NAG Fortran Library Manual
Mark 19

Volume 11

G08 – G13

NAG Fortran Library Manual, Mark 19

September 1999 ISBN 1-85206-169-3

NAG Ltd
Wilkinson House
Jordan Hill Road
Oxford
OX2 8DR
United Kingdom

Tel: +44 (0)1865 511245
Fax: +44 (0)1865 310139

Nihon NAG KK
Nagashima Building 2F
2-24-3 Higashi
Shibuya-ku
Tokyo
Japan

Tel: +81 (0)3 5485 2901
Fax: +81 (0)3 5485 2903

NAG GmbH
Schleißheimerstraße 5
85748 Garching
Deutschland

Tel: +49 (0)89 3207395
Fax: +49 (0)89 3207396

NAG Inc
1400 Opus Place, Suite 200
Downers Grove, IL 60515-5702
USA

Tel: +1 630 971 2337
Fax: +1 630 971 2706

NAG also has a number of distributors throughout the world. Please contact NAG for further details.

[*NP3390/19*]

Chapter G08 – Nonparametric Statistics

Note. Please refer to the Users' Note for your implementation to check that a routine is available.

Routine Name	Mark of Introduction	Purpose
G08AAF	8	Sign test on two paired samples
G08ACF	8	Median test on two samples of unequal size
G08AEF	8	Friedman two-way analysis of variance on k matched samples
G08AFF	8	Kruskal–Wallis one-way analysis of variance on k samples of unequal size
G08AGF	14	Performs the Wilcoxon one-sample (matched pairs) signed rank test
G08AHF	14	Performs the Mann–Whitney U test on two independent samples
G08AJF	14	Computes the exact probabilities for the Mann–Whitney U statistic, no ties in pooled sample
G08AKF	14	Computes the exact probabilities for the Mann–Whitney U statistic, ties in pooled sample
G08ALF	15	Performs the Cochran Q test on cross-classified binary data
G08BAF	8	Mood's and David's tests on two samples of unequal size
G08CBF	14	Performs the one-sample Kolmogorov–Smirnov test for standard distributions
G08CCF	14	Performs the one-sample Kolmogorov–Smirnov test for a user-supplied distribution
G08CDF	14	Performs the two-sample Kolmogorov–Smirnov test
G08CGF	14	Performs the χ^2 goodness of fit test, for standard continuous distributions
G08DAF	8	Kendall's coefficient of concordance
G08EAF	14	Performs the runs up or runs down test for randomness
G08EBF	14	Performs the pairs (serial) test for randomness
G08ECF	14	Performs the triplets test for randomness
G08EDF	14	Performs the gaps test for randomness
G08RAF	12	Regression using ranks, uncensored data
G08RBF	12	Regression using ranks, right-censored data

Chapter G08

Nonparametric Statistics

Contents

1 Scope of the Chapter

The routines in this chapter perform nonparametric statistical tests which are based on distribution-free methods of analysis. For convenience, the chapter contents are divided into five types of test: tests of location, tests of dispersion, tests of distribution, tests of association and correlation, and tests of randomness. There are also routines to fit linear regression models using the ranks of the observations.

The emphasis in this chapter is on testing; users wishing to compute nonparametric correlations are referred to Chapter G02, which contains several routines for that purpose.

There are a large number of nonparametric tests available. A selection of some of the more commonly used tests are included in this chapter.

2 Background to the Problems

2.1 Parametric and Nonparametric Hypothesis Testing

Classical techniques of statistical inference often make numerous or stringent assumptions about the nature of the population or populations from which the observations have been drawn. For instance, a testing procedure might assume that the set of data was obtained from Normally distributed populations. It might be further assumed that the populations involved have equal variances, or that there is a known relationship between the variances. In the Normal case, the test statistic derived would usually be a function of the sample means and variances, since a Normal distribution is completely characterised by its mean and variance. Alternatively, it might be assumed that the set of data was obtained from other distributions of known form, such as the gamma or the exponential. Again, a testing procedure would be devised based upon the parameters characterising such a distribution.

The type of hypothesis testing just described is usually termed **parametric** inference. Distributional assumptions are made which imply that the parameters of the chosen distribution, as estimated from the data, are sufficient to characterise the difference in distribution between the populations.

However, problems arise with parametric methods of inference when these assumptions cannot be made, either because they are contrary to the known nature of the mechanism generating a population, or because the data obviously do not satisfy the assumptions. Some parametric procedures become unreliable under relatively minor departures from the hypothesised distributional form. In the Normal case for example, tests on variances are extremely sensitive to departures from Normality in the underlying distribution.

There are also common situations, particularly in the behavioural sciences, where much more basic assumptions than that of Normality cannot be made. Data values are not always measured on continuous or even numerical scales. They may be simply categorical in nature, relating to such quantities as voting intentions or food preferences.

Techniques of inference are therefore required which do not involve making detailed assumptions about the underlying mechanism generating the observations. The routines in this chapter perform such distribution-free tests, evaluating from a set of data the value of a test statistic, together with an estimate of its significance.

For a comparison of some distribution-based and distribution-free tests, the interested reader is referred to Chapter 31 of Kendall and Stuart [2]. For a briefer and less mathematical account, see Conover [1] or Siegel [3].

2.2 Types of Nonparametric Test

This introduction is concerned with explaining the basic concepts of hypothesis testing, and some familiarity with the subject is assumed. Chapter 22 of Kendall and Stuart [2] contains a detailed account, and the outline given in Conover [1] or Siegel [3] should be sufficient to understand this section.

Nonparametric tests may be grouped into five categories:

(1) Tests of location
(2) Tests of dispersion
(3) Distribution-free tests of fit
(4) Tests of association or correlation

(5) Tests of randomness

Tests can also be categorised by the design that they can be applied to:

(1) One sample
(2) Two related (paired) samples
(3) Two independent samples
(4) k (> 2) related (matched) samples
(5) k (> 2) independent samples

A third classification of a test relates to the type of data to which it may be applied. Variables are recorded on four scales of measurement: nominal (categorical), ordinal, interval, and ratio.

The nominal scale is used only to categorise data; for each category a name, perhaps numeric, is assigned so that two different categories will be identified by distinct names. The ordinal scale, as well as categorising the observations, orders the categories. Each is assigned a distinct identifying symbol, in such a way that the order of the symbols corresponds to the order of the categories. (The most common system for ordinal variables is to assign numerical identifiers to the categories, though if they have previously been assigned alphabetic characters, these may be transformed to a numerical system by any convenient method which preserves the ordering of the categories.) The interval scale not only categorises and orders the observations, but also quantifies the comparison between categories; this necessitates a common unit of measurement and an arbitrary zero-point. Finally, the ratio scale is similar to the interval scale, except that it has an absolute (as opposed to arbitrary) zero-point.

It is apparent that there are many possible combinations of these three characteristics of a problem, and many nonparametric tests have been derived to meet the different experimental situations. However, it is not usually a difficult matter to choose an appropriate test given the nature of the data and the type of test which one wishes to perform.

2.3 Principles of Nonparametric Tests

In this section, each type of test is considered in turn, and remarks are made on the design principles on which each is based.

2.3.1 Location tests

These tests are primarily concerned with inferences about differences in the location of the population distributions. In some cases however, the tests are only concerned with inferences about the population distributions unless added assumptions are made which allow the hypotheses to be stated in terms of the location parameters.

For most of these tests, data must be measured numerically on at least an ordinal scale, in order that a measure of location may be devised. Ordinal measurement implies that pairs of values may be compared and numerically ordered. A vector of n values may therefore be **ranked** from smallest to largest using the ordering operation. The resultant **ranks** contain all the information in the original data, but have the advantage that tests may be derived easily based on them, and no testing bias is introduced by the use of ordinal values as though they were measured on an interval scale. Note that the requirement of the measurement scale being ordinal does not imply that all tests of this type involve the actual ranking of the original data.

For the one-sample or matched pairs case, test statistics may be derived based on the number of observations (or differences) lying either side of zero (or some other fixed value), as in the sign test for example. Under the hypothesis that the median of the single population is zero or the difference in the medians of the paired populations is zero the number of positive and negative values should be similar. The Wilcoxon signed rank test goes further than the sign test by taking into account the magnitude of the single sample values or of the differences.

For the two-sample case, if median equality is hypothesised, the distribution of the ranks of each sample in the total pooled sample should be similar. Test statistics, such as the Mann–Whitney U statistic, which are based on the ranks of each sample and summarise the differences in rank sums for each sample, may be computed. These statistics are referred to their expected distributions under the null hypothesis. The above hypothesis can also be tested using the median test. Its test statistic is based on the number of values in each sample which are greater than or less than the pooled median of the two samples, rather than the ranks of each sample.

If median equality is hypothesised for several samples, the distribution ranks of the members of each sample in the total pooled sample should be 'homogeneous'. Test statistics can be derived which summarise the differences in rank sums for the various samples, and again referred to their expected distributions under the null hypothesis.

2.3.2 Dispersion tests

These provide a distribution-free alternative to such tests as the F-(variance-ratio) test for variance equality, which is very sensitive to non-Normality in the generating distribution.

The dispersions of two or more samples may be compared by pooling the samples and observing the distribution of ranks in the ranked pooled sample. Equal dispersions should be recognisable by there being a wide distribution of the extreme ranks between the members of different samples. Statistics are evaluated which quantify the dispersion of ranks between samples, and their significance may be found by evaluating their permutation distributions assuming that no dispersion difference exists.

2.3.3 Tests of fit

In the one-sample case, these are tests which investigate whether or not a sample of observations can be considered to follow a specified distribution. In the two-sample case, a test of fit investigates whether the two samples can be considered to have arisen from a common probability distribution.

For the one sample problem, the null hypothesis may specify only the distributional form, for example Normal(μ,σ^2), or it may incorporate actual parameter values, for example, Poisson with mean 10.

Some tests of this type proceed by forming the sample cumulative distribution function of the observations and computing a statistic whose value measures the departure of the sample cumulative distribution function from that of the null distribution. In the two-sample case, a statistic is computed which provides a measure of the difference between the sample cumulative distribution function of each sample. These tests are known as one- or two-sample Kolmogorov–Smirnov tests.

The significance for these test statistics can be computed directly for moderate sample sizes but for larger sample sizes asymptotic results are often used.

Another goodness of fit test is the χ^2 test. For this test, the data is first grouped into intervals and then the difference between the observed number of observations in each interval and the number expected, if the null hypothesis is true, is computed. A statistic based on these differences has asymptotically a χ^2-distribution.

2.3.4 Association and correlation tests

These are distribution-free analogues of tests based on such statistics as Pearson product-moment correlation coefficients.

Essentially they are based on rankings rather than the observed data values, and involve summing some function of the rank differences between the samples to obtain an overall measure of the concordance of ranks. This measure can be standardised by dividing by its theoretical maximum value for the given sample size and number of samples.

Significance levels may be calculated for quite small sample sizes by using an approximation to a χ^2-distribution.

2.3.5 Tests of randomness

These tests are designed to investigate sequences of observations and attempt to identify any deviations from randomness. There are clearly many ways in which a sequence may deviate from randomness. The tests provided here primarily detect some form of dependency between the observations in the sequence.

The most common application of this type of tests is in the area of random number generation. The tests are used as empirical tests on a sample of output from a generator to establish local randomness. Theoretical tests are necessary and useful for testing global randomness. Some of the more common empirical tests are discussed below.

A runs-up or runs-down test investigates whether runs of different lengths are occuring with greater or lesser frequency than would be expected under the null hypothesis of randomness. A run up is defined as a sequence of observations in which each observation is larger than the previous observation. The run up ends when an observation is smaller than the previous observation. A test statistic, modified to take into account the dependency between successive run lengths, is computed. The test statistic has an asymptotic χ^2-distribution.

The pairs test investigates the condition that, under the null hypothesis of randomness, the non-overlapping 2-tuples (pairs) of a sequence of observations from the interval [0,1] should be uniformly distributed over the unit square ($[0,1]^2$). The triplets test follows the same idea but considers 3-tuples and checks for uniformity over the unit cube ($[0,1]^3$). In each test, a test statistic, based on differences between the observed and expected distribution of the 2- or 3-tuples, is computed which has an asymptotic χ^2-distribution.

The gaps test considers the 'gaps' between successive occurences of observations in the sequence lying in a specified range. Under the null hypothesis of randomness, the gap length should follow a geometric distribution with a parameter based on the length of the specified range, relative to the overall length of the interval containing all possible observations. The expected number of 'gaps', of a certain length, under the null hypothesis may thus be computed together with a test statistic based on the differences between the observed and expected numbers of 'gaps' of different length. Again the test statistic has an asymptotic χ^2-distribution.

Other empirical tests such as the χ^2 goodness of fit test and the one-sample Kolmogorov–Smirnov test may be used to investigate a sequence for non-uniformity.

2.4 Regression using ranks

If the user wishes to fit a regression model but is unsure about what transformation to take for the observed response to obtain a linear model, then one strategy is to replace response observations by their ranks. Estimates for regression parameters can be found by maximizing a likelihood function based on the ranks and the proposed regression model. The present routines give approximate estimates which are adequate when the signal-to-noise ratio is small, which is often the case with data from the medical and social sciences. Approximate standard errors of estimated regression coefficients are found. Also χ^2 statistics can be used to test the null hypothesis of no regression effect.

3 Recommendations on Choice and Use of Available Routines

Note. Refer to the Users' Note for your implementation to check that a routine is available.

The routines are grouped into six categories. The fourth character of the routine name is used to denote this categorisation.

Sub-chapter	**Type of test**
G08A	Location
G08B	Dispersion
G08C	Fit
G08D	Correlation and Association
G08E	Randomness
G08R	Regression using Ranks

3.1 G08A – Location Tests

3.1.1 One sample or matched-pairs case

Note that a random sample of matched pairs, (x_i,y_i), may be reduced to a single sample by considering the differences, $d_i = x_i - y_i$ say, of each pair. The matched pair may be thought of as a single observation on a bivariate random variable.

G08AAF performs the sign test on two paired samples. Each pair is classified as a $+$ or $-$ depending on the sign of the difference between the two data values within the pair. Under the assumptions that the d_i are mutually independent and that the observations are measured on at least an ordinal scale, the sign test tests the hypothesis that for any pair sampled from the population distribution, Probabilty(+) = Probability(−). The hypothesis may be stated in terms of the equality of the location parameters but the test is no longer regarded as unbiased and consistent unless further assumptions are made. If the user wishes to test the hypothesis that the location parameters differ by a fixed amount then that amount must be added or subtracted from one of the samples as required before calling G08AAF.

G08AGF performs the one-sample Wilcoxon signed-rank test. The test may be used to test if the median of the population from which the random sample was taken is equal to some specified value (commonly used to test if the median is zero). In this test not only is the sign of the difference between the data values and the hypothesised median value important but also the magnitude of this difference. Thus, where the magnitude of the differences (or the data values themselves if the hypothesised median value is zero) is important this test is preferred to the sign test because it is more powerful. The test may easily be used to test whether the medians of two related populations are equal by taking the differences between the paired sample values and then testing the hypothesis that the median of the differences is zero, using the single sample of differences. The significance of the test statistic may be computed exactly for a moderate sample size but for a larger sample a Normal approximation is used. The exact method allows for ties in the differences.

3.1.2 Two independent samples

G08ACF performs the median test and G08AHF performs the Mann–Whitney U test.

For both tests the two samples are assumed to be random samples from their respective populations and mutually independent. The measurement scale must be at least ordinal.

Note that, although the median test may be generalised to more than two samples, G08ACF only deals with the two-sample case. For the median test, each observation is classified as being above or below the pooled median of the two samples. It may be used to test the hypothesis that the two population medians are equal; under the assumption that if the two population medians are equal then the probability of an observation exceeding the pooled median is the same for both populations.

The Mann–Whitney U test involves the ranking of the pooled sample. The Mann–Whitney test thus attaches importance to the position of each observation relative to the others and not just its position relative to the median of the pooled sample as in the median test. Thus when the magnitude of the differences between the observations is meaningful the Mann–Whitney U test is preferred as it is more powerful than the median test. The test tests whether the two population distributions are the same or not. If it is assumed that any difference between the the two population distributions is a difference in the location then the test is testing whether the population means are the same or not.

In G08AHF, the significance of the U test statistic is computed using a Normal approximation. If the exact significance is desired then either G08AJF or G08AKF must be used. G08AJF computes the exact significance of the U test statistic for the case where there are no ties in the pooled sample. It requires only the value of the statistic and the two sample sizes. G08AKF computes the exact significance of the U test statistic for the case where there are ties in the pooled sample. It requires the value of the statistic and the two sample sizes and the ranks of the observations of the two samples as provided by G08AHF. G08AHF returns an indicator to inform the user whether or not ties were found in the pooled sample.

3.1.3 More than two related samples

G08AEF performs the Friedman two-way analysis of variance. This test may in some ways be regarded as an extension of the sign test to the case of k, $(k > 2)$, related samples. The data is in the form of a number of multivariate observations which are assumed to be mutually independent. This test also assumes that the measurement within each observation across the k variates is at least ordinal so that the observation for each variate may be ranked according to some criteria.

For data which may be defined as either a zero or one, that is binary response data, G08ALF performs Cochran's Q-test to examine differences between the treatments within blocks.

3.1.4 More than two independent samples

G08AFF performs the Kruskal–Wallis one-way analysis of variance. The test assumes that each sample is a random sample from its respective distributions and in addition that there is both independence within the samples and mutual independence among the various samples. The test requires that the measurement scale is at least ordinal so that the pooled sample may be ranked.

3.2 G08B – Dispersion Tests

G08BAF performs either Mood's or David's test for dispersion differences, or both, for two independent samples of possibly unequal size.

For both tests the null hypothesis is that the two samples have equal dispersions, the routine returning a probability value which may be used to perform the test against a one-sided or two-sided alternative, in a way described in the routine document.

3.3 G08C – Tests of Fit

G08CBF and G08CCF both perform the one sample Kolmogorov–Smirnov distribution test. This test is used to test the null hypothesis that the random sample arises from a specified null distribution against one of three possible alternatives.

With G08CBF the user may choose a null distribution from one of the following: the uniform, Normal, gamma, beta, binomial, exponential, and Poisson. The parameter values may either be specified by the user or estimated from the data by the routine. With G08CCF the user must provide a function which will compute the value of the cumulative distribution function at any specified point for the null distribution. The alternative hypotheses available correspond to one- and two-sided tests. The distribution of the test statistic is computed using an exact method for a moderate sample size. For a larger sample size an asymptotic result is used.

G08CDF performs the two-sample Kolmogorov–Smirnov test which tests the null hypothesis that the two samples may be considered to have arisen from the same population distribution against one of three possible alternative hypotheses, again corresponding to one-sided and two-sided tests. The distribution of the test statistic is computed using an exact method for moderate sample sizes but for larger samples, approximations based on asymptotic results are used.

Note that G01EYF and G01EZF are available for computing the distributions of the one-sample and two-sample Kolmogorov–Smirnov statistics respectively.

G08CGF performs the χ^2 goodness of fit test on a single sample which again tests the null hypothesis that the sample arises from a specified null distribution. The user may choose a null distribution from one of the following: the Normal, uniform, exponential, χ^2, and gamma, or may define the distribution by specifying the probability that an observation lies in a certain interval for a range of intervals covering the support of the null distibution. The significance of this test is computed using the χ^2 distribution as an approximation to the distribution of the test statistic.

Tests of Normality may also be carried out using routines in Chapter G01.

3.4 G08D – Association and Correlation Tests

G08DAF computes Kendall's coefficient of concordance on k independent ranks of n objects. An example of its application would be to compare for consistency the results of a group of IQ tests performed on the same set of people. Allowance is made for tied rankings, and the approximate significance of the computed coefficient is found.

3.5 G08E – Tests of Randomness

G08EAF performs the runs-up test on a sequence of observations. The runs-down test may be performed by multiplying each observation by -1 before calling the routine. All runs whose length is greater than or equal to a certain chosen length will be treated as a single group.

G08EBF	performs the pairs (serial) test on a sequence of observations from the interval [0,1]. The number of equal sub-intervals into which the interval [0,1] is to be divided must be specified.
G08ECF	performs the triplets test on a sequence of observations from the interval [0,1]. The number of equal sub-intervals into which the interval [0,1] is to be divided must be specified.
G08EDF	performs the gaps test on a sequence of observations. The total of the interval containing all possible values the observations could take must be specified together with the interval being used to define the 'gaps'. All 'gaps' whose length is greater than or equal to a certain chosen length will be treated as a single group.

3.6 G08R – Regression Using Ranks

G08RAF	fits a multiple linear regression model in which the observations on the response variable are replaced by their ranks.
G08RBF	performs the same function but takes into account observations which may be right-censored.

3.7 Related Routines

Tests of location and distribution may be based on scores which are estimates of the expected values of the order statistics. G01DHF may be used to compute Normal scores, an approximation to the Normal scores (Blom, Tukey or van der Waerden scores) or Savage (exponential) scores. For more accurate Normal scores G01DAF may be used. Other routines in subchapter G01D may be used to test for Normality.

4 Routines Withdrawn or Scheduled for Withdrawal

Since Mark 13 the following routines have been withdrawn. Advice on replacing calls to these routines is given in the document 'Advice on Replacement Calls for Withdrawn/Superseded Routines'.

G08ABF G08ADF G08CAF

5 References

[1] Conover W J (1980) *Practical Nonparametric Statistics* Wiley

[2] Kendall M G and Stuart A (1973) *The Advanced Theory of Statistics (Volume 2)* Griffin (3rd Edition)

[3] Siegel S (1956) *Non-parametric Statistics for the Behavioral Sciences* McGraw-Hill

G08AAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08AAF performs the Sign test on two related samples of size n.

2. Specification

```
SUBROUTINE G08AAF (X, Y, N, IS, N1, P, IFAIL)
INTEGER        N, IS, N1, IFAIL
real           X(N), Y(N), P
```

3. Description

The Sign test investigates the median difference between pairs of scores from two matched samples of size n, denoted by $\{x_i,y_i\}$, for $i = 1,2,...,n$. The hypothesis under test, H_0, often called the null hypothesis, is that the medians are the same, and this is to be tested against a one- or two-sided alternative H_1 (see below).

G08AAF computes:

(a) The test statistic S, which is the number of pairs for which $x_i < y_i$;

(b) The number n_1 of non-tied pairs $(x_i \neq y_i)$;

(c) The lower tail probability p corresponding to S (adjusted to allow the complement $(1-p)$ to be used in an upper 1-tailed or a 2-tailed test). p is the probability of observing a value $\leq S$ if $S < \frac{1}{2}n_1$; or of observing a value $< S$ if $S > \frac{1}{2}n_1$, given that H_0 is true. If $S = \frac{1}{2}n_1$, p is set to 0.5.

Suppose that a significance test of a chosen size α is to be performed (i.e. α is the probability of rejecting H_0 when H_0 is true; typically α is a small quantity such as 0.05 or 0.01). The returned value of p can be used to perform a significance test on the median difference, against various alternative hypotheses H_1, as follows:

(i) H_1 : median of $x \neq$ median of y. H_0 is rejected if $2\times\min(p,1-p) < \alpha$.

(ii) H_1 : median of $x >$ median of y. H_0 is rejected if $p < \alpha$.

(iii) H_1 : median of $x <$ median of y. H_0 is rejected if $1 - p < \alpha$.

4. References

[1] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences.
McGraw-Hill, 1956.

5. Parameters

1: X(N) – ***real*** array. *Input*

2: Y(N) – ***real*** array. *Input*

On entry: X(i) and Y(i) must be set to the ith pair of data values, $\{x_i,y_i\}$, for $i = 1,2,...,n$.

3: N – INTEGER. *Input*

On entry: the size of each sample, n.

Constraint: N $\geq$ 1.

4: IS – INTEGER. *Output*

On exit: the Sign test statistic, S.

5: N1 – INTEGER. *Output*

On exit: the number of non-tied pairs, n_1.

6: P – ***real***. *Output*

On exit: the lower tail probability, p, corresponding to S.

7: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, N < 1.

IFAIL = 2

N1 = 0, i.e. the samples are identical.

7. Accuracy

The tail probability, p, is computed using the relationship between the binomial and beta distributions. For $n_1 < 120, p$ should be accurate to at least 4 significant figures, assuming that the machine has a precision of 7 or more digits. For $n_1 \geq 120, p$ should be computed with an absoulute error of less than 0.005. For further details see G01EEF.

8. Further Comments

The time taken by the routine is small, and increases with n.

9. Example

This example is taken from page 69 of Seigel [1]. The data relate to ratings of 'insight into paternal discipline' for 17 sets of parents, recorded on a scale from 1 to 5.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08AAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N
      PARAMETER        (N=17)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             SIG
      INTEGER          IFAIL, IS, N1
*     .. Local Arrays ..
      real             X(N), Y(N)
*     .. External Subroutines ..
      EXTERNAL         G08AAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08AAF Example Program Results'
```

```
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) X, Y
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Sign test'
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Data values'
      WRITE (NOUT,*)
      WRITE (NOUT,99999) X, Y
      IFAIL = 0
*
      CALL G08AAF(X,Y,N,IS,N1,SIG,IFAIL)
*
      WRITE (NOUT,*)
      WRITE (NOUT,99998) 'Test statistic   ', IS
      WRITE (NOUT,99998) 'Observations     ', N1
      WRITE (NOUT,99997) 'Lower tail prob. ', SIG
      STOP
*
99999 FORMAT (4X,17F3.0)
99998 FORMAT (1X,A,I5)
99997 FORMAT (1X,A,F5.3)
      END
```

9.2. Program Data

```
G08AAF Example Program Data
 4 4 5 5 3 2 5 3 1 5 5 5 4 5 5 5 5
 2 3 3 3 3 3 3 3 2 3 2 2 5 2 5 3 1
```

9.3. Program Results

```
 G08AAF Example Program Results

 Sign test

 Data values

     4. 4. 5. 5. 3. 2. 5. 3. 1. 5. 5. 5. 4. 5. 5. 5. 5.
     2. 3. 3. 3. 3. 3. 3. 3. 2. 3. 2. 2. 5. 2. 5. 3. 1.

 Test statistic       3
 Observations        14
 Lower tail prob. 0.029
```

G08ACF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08ACF performs the Median test on two independent samples of possibly unequal size.

2. Specification

```
SUBROUTINE G08ACF (X, N, N1, W, I1, I2, P, IFAIL)
INTEGER        N, N1, I1, I2, IFAIL
real           X(N), W(N), P
```

3. Description

The Median test investigates the difference between the medians of two independent samples of sizes n_1 and n_2, denoted by:

$$x_1,x_2,\ldots,x_{n_1}$$

and

$$x_{n_1+1},x_{n_1+2},\ldots,x_n, \qquad n = n_1+n_2.$$

The hypothesis under test, H_0, often called the null hypothesis, is that the medians are the same, and this is to be tested against the alternative hypothesis H_1 that they are different.

The test proceeds by forming a 2×2 frequency table, giving the number of scores in each sample above and below the median of the pooled sample:

	Sample 1	Sample 2	Total
Scores ≤ pooled median	i_1	i_2	$i_1 + i_2$
Scores ≥ pooled median	$n_1 - i_1$	$n_2 - i_2$	$n - (i_1+i_2)$
Total	n_1	n_2	n

Under the null hypothesis, H_0, we would expect about half of each group's scores to be above the pooled median and about half below, that is we would expect i, to be about $n_1/2$ and i_2 to be about $n_2/2$.

G08ACF returns:

(a) The frequencies i_1 and i_2;

(b) The probability, p, of observing a table at least as 'extreme' as that actually observed, given that H_0 is true. If $n < 40$, p is computed directly ('Fisher's exact test'); otherwise a χ_1^2 approximation is used (see G01AFF).

H_0 is rejected by a test of chosen size α if $p < \alpha$.

4. References

[1] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences.
McGraw-Hill, 1956.

5. Parameters

1: X(N) – ***real*** array. *Input*

On entry: the first n_1 elements of X must be set to the data values in the first sample, and the next n_2 (= N$-n_1$) elements to the data values in the second sample.

2: N – INTEGER. *Input*

On entry: the total of the two sample sizes, n $(= n_1+n_2)$.

Constraint: N $\geq$ 2.

3: N1 – INTEGER. *Input*

On entry: the size of the first sample n_1.

Constraint: 1 $\leq$ N1 $<$ N.

4: W(N) – ***real*** array. *Workspace*

5: I1 – INTEGER. *Output*

On exit: the number of scores in the first sample which lie below the pooled median, i_1.

6: I2 – INTEGER. *Output*

On exit: the number of scores in the second sample which lie below the pooled median, i_2.

7: P – ***real.*** *Output*

On exit: the tail probability p corresponding to the observed dichotomy of the two samples.

8: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, N < 2.

IFAIL = 2

On entry, N1 < 1,
or N1 $\geq$ N.

7. Accuracy

The probability returned should be accurate enough for practical use.

8. Further Comments

The time taken by the routine is small, and increases with n.

9. Example

This example is taken from page 112 of Seigel [1]. The data relate to scores of 'oral socialisation anxiety' in 39 societies, which can be separated into groups of size 16 and 23 on the basis of their attitudes to illness.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08ACF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N
      PARAMETER        (N=39)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             P
      INTEGER          I, I1, I2, IFAIL, N1
*     .. Local Arrays ..
      real             W1(N), X(N)
*     .. External Subroutines ..
      EXTERNAL         G08ACF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08ACF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) (X(I),I=1,N)
      N1 = 16
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Median test'
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Data values'
      WRITE (NOUT,*)
      WRITE (NOUT,99999) '    Group 1  ', (X(I),I=1,N1)
      WRITE (NOUT,*)
      WRITE (NOUT,99999) '    Group 2  ', (X(I),I=N1+1,N)
      IFAIL = 0
*
      CALL G08ACF(X,N,N1,W1,I1,I2,P,IFAIL)
*
      WRITE (NOUT,*)
      WRITE (NOUT,99998) I1, ' scores below median in group 1'
      WRITE (NOUT,99998) I2, ' scores below median in group 2'
      WRITE (NOUT,*)
      WRITE (NOUT,99997) '     Significance  ', P
      STOP
*
99999 FORMAT (1X,A,8F4.0,/(14X,8F4.0))
99998 FORMAT (1X,I6,A)
99997 FORMAT (1X,A,F8.5)
      END
```

9.2. Program Data

```
G08ACF Example Program Data
 13  6 12  7 12  7 10  7 10  7 10  7 10  8  9  8
 17  6 16  8 15  8 15 10 15 10 14 10 14 11 14 11
 13 12 13 12 13 12 12
```

9.3. Program Results

```
G08ACF Example Program Results

Median test

Data values

    Group 1   13.  6. 12.  7. 12.  7. 10.  7.
              10.  7. 10.  7. 10.  8.  9.  8.

    Group 2   17.  6. 16.  8. 15.  8. 15. 10.
              15. 10. 14. 10. 14. 11. 14. 11.
              13. 12. 13. 12. 13. 12. 12.

    13 scores below median in group 1
     6 scores below median in group 2

     Significance   0.00088
```

G08AEF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08AEF performs the Friedman two-way analysis of variance by ranks on k related samples of size n.

2. Specification

```
SUBROUTINE G08AEF (X, IX, K, N, W1, W2, FR, P, IFAIL)
INTEGER        IX, K, N, IFAIL
real           X(IX,N), W1(K), W2(K), FR, P
```

3. Description

The Friedman test investigates the score differences between k matched samples of size n, the scores in the ith sample being denoted by:

$x_{i1},x_{i2},\ldots,x_{in}$.

(Thus the sample scores may be regarded as a two-way table with k rows and n columns.) The hypothesis under test, H_0, often called the null hypothesis, is that the samples come from the same population, and this is to be tested against the alternative hypothesis H_1 that they come from different populations.

The test is based on the observed distribution of score rankings between the matched observations in different samples.

The test proceeds as follows:

(a) The scores in each column are ranked, r_{ij} denoting the rank within column j of the observation in row i. Average ranks are assigned to tied scores.

(b) The ranks are summed over each row to give rank sums $t_i = \sum_{j=1}^{n} r_{ij}$, for $i = 1,2,\ldots,k$.

(c) The Friedman test statistic FR is computed, where

$$FR = \frac{12}{nk(k+1)}\sum_{i=1}^{k}\{t_i-\tfrac{1}{2}n(k+1)\}^2.$$

G08AEF returns the value of FR, and also an approximation, p, to the significance of this value. (FR approximately follows a χ^2_{k-1} distribution, so large values of FR imply rejection of H_0). H_0 is rejected by a test of chosen size α if $p < \alpha$. The approximation p is acceptable unless $k = 4$ and $n < 5$, or $k = 3$ and $n < 10$, or $k = 2$ and $n < 20$; for $k = 3$ or 4, tables should be consulted; (e.g. N of Siegel [1]) for $k = 2$ the Sign test (see G08AAF) or Wilcoxon test (see G08AGF) is in any case more appropriate.

4. References

[1] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences.
McGraw-Hill, 1956.

5. Parameters

1: X(IX,N) – ***real*** array. *Input*

On entry: X(i,j) must be set to the value, x_{ij}, of observation j in sample i, for $i = 1,2,\ldots,k$; $j = 1,2,\ldots,n$.

2: IX – INTEGER. *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G08AEF is called.

Constraint: IX $\geq$ K.

3: K – INTEGER. *Input*

On entry: the number of samples, k.

Constraint: K $>$ 1.

4: N – INTEGER. *Input*

On entry: the size, n, of each sample.

Constraint: N $\geq$ 1.

5: W1(K) – ***real*** array. *Workspace*

6: W2(K) – ***real*** array. *Workspace*

7: FR – ***real.*** *Output*

On exit: the value of the Friedman test statistic, FR.

8: P – ***real.*** *Output*

On exit: the approximate significance, p, of the Friedman test statistic.

9: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, N $<$ 1.

IFAIL = 2

On entry, IX $<$ K.

IFAIL = 3

On entry, K $\leq$ 1.

7. Accuracy

For estimates of the accuracy of the significance p, see G01ECF. The χ^2 approximation is acceptable unless $k = 4$ and $n < 5$, or $k = 3$ and $n < 10$, or $k = 2$ and $n < 20$.

8. Further Comments

The time taken by the routine is approximately proportional to the product nk.

If $k = 2$, the Sign test (see G08AAF) or Wilcoxon test (see G08AGF) is more appropriate.

9. Example

This example is taken from page 169 of Siegel [1]. The data relate to training scores of three matched samples of 18 rats, trained under three different patterns of reinforcement.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08AEF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          K, N, IX
      PARAMETER        (K=3,N=18,IX=K)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             FR, SIG
      INTEGER          I, IFAIL, J
*     .. Local Arrays ..
      real             W1(K), W2(K), X(IX,N)
*     .. External Subroutines ..
      EXTERNAL         G08AEF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08AEF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) ((X(I,J),J=1,18),I=1,3)
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Friedman test'
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Data values'
      WRITE (NOUT,*)
      WRITE (NOUT,*) '   Group Group Group'
      WRITE (NOUT,*) '     1     2     3'
      WRITE (NOUT,99997) ((X(I,J),I=1,3),J=1,18)
      IFAIL = 0
*
      CALL G08AEF(X,IX,K,N,W1,W2,FR,SIG,IFAIL)
*
      WRITE (NOUT,*)
      WRITE (NOUT,99999) 'Test statistic      ', FR
      WRITE (NOUT,99998) 'Degrees of freedom  ', K - 1
      WRITE (NOUT,99999) 'Significance        ', SIG
      STOP
*
99999 FORMAT (1X,A,F6.3)
99998 FORMAT (1X,A,I6)
99997 FORMAT (1X,F7.1,2F6.1)
      END
```

9.2. Program Data

```
G08AEF Example Program Data
  1  2  1  1  3  2  3  1  3  3  2  2  3  2  2.5  3  3  2
  3  3  3  2  1  3  2  3  1  1  3  3  2  3  2.5  2  2  3
  2  1  2  3  2  1  1  2  2  2  1  1  1  1  1    1  1  1
```

9.3. Program Results

```
G08AEF Example Program Results

Friedman test

Data values

   Group Group Group
     1     2     3
    1.0   3.0   2.0
    2.0   3.0   1.0
    1.0   3.0   2.0
    1.0   2.0   3.0
    3.0   1.0   2.0
    2.0   3.0   1.0
    3.0   2.0   1.0
    1.0   3.0   2.0
    3.0   1.0   2.0
    3.0   1.0   2.0
    2.0   3.0   1.0
    2.0   3.0   1.0
    3.0   2.0   1.0
    2.0   3.0   1.0
    2.5   2.5   1.0
    3.0   2.0   1.0
    3.0   2.0   1.0
    2.0   3.0   1.0

Test statistic        8.583
Degrees of freedom        2
Significance          0.014
```

G08AFF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08AFF performs the Kruskal-Wallis one-way analysis of variance by ranks on k independent samples of possibly unequal sizes.

2. Specification

```
SUBROUTINE G08AFF (X, LX, L, K, W, H, P, IFAIL)
INTEGER        LX, L(K), K, IFAIL
real           X(LX), W(LX), H, P
```

3. Description

The Kruskal-Wallis test investigates the differences between scores from k independent samples of unequal sizes, the ith sample containing l_i observations. The hypothesis under test, H_0, often called the null hypothesis, is that the samples come from the same population, and this is to be tested against the alternative hypothesis H_1 that they come from different populations.

The test proceeds as follows:

(a) The pooled sample of all the observations is ranked. Average ranks are assigned to tied scores.

(b) The ranks of the observations in each sample are summed, to give the rank sums R_i, for $i = 1,2,...,k$.

(c) The Kruskal-Wallis' test statistic H is computed as:

$$H = \frac{12}{N(N+1)}\sum_{i=1}^{k}\frac{R_i^2}{l_i} - 3(N+1), \qquad \text{where } N = \sum_{i=1}^{k} l_i,$$

i.e. N is the total number of observations. If there are tied scores, H is corrected by dividing by:

$$1 - \frac{\sum(t^3-t)}{N^3-N}$$

where t is the number of tied scores in a group and the summation is over all tied groups.

G08AFF returns the value of H, and also an approximation, p, to the probability of a value of at least H being observed, H_0 is true. (H approximately follows a χ^2_{k-1} distribution). H_0 is rejected by a test of chosen size α if $p < \alpha$. The approximation p is acceptable unless $k = 3$ and l_1, l_2 or $l_3 \leq 5$ in which case tables should be consulted (e.g. O of Seigel [1]) or $k = 2$ (in which case the Median test (see G08ACF) or the Mann-Whitney U test (see G08AHF) is more appropriate).

4. References

[1] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences.
McGraw-Hill, 1956.

[2] MOORE, P.G. SHIRLEY, E.A. and EDWARDS, D.E.
Standard Statistical Calculations.
Pitman, 1972.

5. Parameters

1: X(LX) – ***real*** array. *Input*

On entry: the elements of X must contain the observations in the K groups. The first l_1 elements must contain the scores in the first group, the next l_2 those in the second group, and so on.

2: LX – INTEGER. *Input*

On entry: the total number of observations, N.

Constraint: $\mathrm{LX} = \sum_{i=1}^{k} \mathrm{L}(i)$.

3: L(K) – INTEGER array. *Input*

On entry: $\mathrm{L}(i)$ must contain the number of observations l_i in sample i, for $i = 1,2,...,k$.

Constraint: $\mathrm{L}(i) > 0$, for $i = 1,2,...k$.

4: K – INTEGER. *Input*

On entry: the number of samples, k.

Constraint: $\mathrm{K} \geq 2$.

5: W(LX) – ***real*** array. *Workspace*

6: H – ***real.*** *Output*

On exit: the value of the Kruskal-Wallis test statistic, H.

7: P – ***real.*** *Input*

On exit: the approximate significance, p, of the Kruskal-Wallis test statistic.

8: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, $\mathrm{K} < 2$.

IFAIL = 2

On entry, $\mathrm{L}(i) \leq 0$ for some i, $i = 1,2,...,k$.

IFAIL = 3

On entry, $\mathrm{LX} \neq \sum_{i=1}^{k} \mathrm{L}(i)$.

IFAIL = 4

On entry, all the observations were equal.

7. Accuracy

For estimates of the accuracy of the significance p, see G01ECF. The χ^2 approximation is acceptable unless $k = 3$ and l_1, l_2 or $l_3 \leq 5$.

8. Further Comments

The time taken by the routine is small, and increases with N and k.

If $k = 2$, the Median test (see G08ACF) or the Mann-Whitney U test (see G08AHF) is more appropriate.

9. Example

This example is taken from Moore *et al.* [2]. There are 5 groups of sizes 5, 8, 6, 8 and 8. The data represent the weight gain, in pounds, of pigs from five different litters under the same conditions.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08AFF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          K, LMAX
      PARAMETER        (K=5,LMAX=35)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             H, P
      INTEGER          I, IFAIL, II, LX, NHI, NI, NLO
*     .. Local Arrays ..
      real             W1(LMAX), X(LMAX)
      INTEGER          L(K)
*     .. External Subroutines ..
      EXTERNAL         G08AFF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08AFF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) L
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Kruskal-Wallis test'
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Data values'
      WRITE (NOUT,*)
      WRITE (NOUT,*) '  Group    Observations'
      LX = 0
      DO 20 I = 1, K
         LX = LX + L(I)
   20 CONTINUE
      IF (LX.LE.LMAX) THEN
         READ (NIN,*) (X(I),I=1,LX)
         IFAIL = 0
         NLO = 1
         DO 40 I = 1, K
            NI = L(I)
            NHI = NLO + NI - 1
            WRITE (NOUT,99999) I, (X(II),II=NLO,NHI)
            NLO = NLO + NI
   40    CONTINUE
*
```

```
         CALL G08AFF(X,LX,L,K,W1,H,P,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'Test statistic       ', H
         WRITE (NOUT,99997) 'Degrees of freedom   ', K - 1
         WRITE (NOUT,99998) 'Significance         ', P
      END IF
      STOP
*
99999 FORMAT (1X,I5,5X,10F4.0)
99998 FORMAT (1X,A,F9.3)
99997 FORMAT (1X,A,I9)
      END
```

9.2. Program Data

```
G08AFF Example Program Data
 5 8 6 8 8
 23 27 26 19 30 29 25 33 36 32
 28 30 31 38 31 28 35 33 36 30
 27 28 22 33 34 34 32 31 33 31
 28 30 24 29 30
```

9.3. Program Results

```
 G08AFF Example Program Results

 Kruskal-Wallis test

 Data values

   Group     Observations
     1       23. 27. 26. 19. 30.
     2       29. 25. 33. 36. 32. 28. 30. 31.
     3       38. 31. 28. 35. 33. 36.
     4       30. 27. 28. 22. 33. 34. 34. 32.
     5       31. 33. 31. 28. 30. 24. 29. 30.

 Test statistic           10.537
 Degrees of freedom            4
 Significance              0.032
```

G08AGF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08AGF performs the Wilcoxon signed rank test on a single sample of size n.

2. Specification

```
      SUBROUTINE G08AGF (N, X, XME, TAIL, ZEROS, W, WNOR, P, N1,
     1                   WRK, IFAIL)
      INTEGER          N, N1, IFAIL
      real             X(N), XME, W, WNOR, P, WRK(3*N)
      CHARACTER*1      TAIL, ZEROS
```

3. Description

The Wilcoxon one sample signed rank test may be used to test whether a particular sample came from a population with a specified median. It is assumed that the population distribution is symmetric. The data consist of a single sample of n observations denoted by $x_1,x_2,...,x_n$. This sample may arise from the difference between pairs of observations from two matched samples of equal size taken from two populations, in which case the test may be used to test whether the median of the first population is the same as that of the second population.

The hypothesis under test, H_0, often called the null hypothesis, is that the median is equal to some given value (X_{med}), and this is to be tested against an alternative hypothesis H_1 which is

H_1 : population median $\neq X_{med}$; or

H_1 : population median $> X_{med}$; or

H_1 : population median $< X_{med}$,

using a two-tailed, upper-tailed or lower-tailed probability respectively. The user selects the alternative hypothesis by choosing the appropriate tail probability to be computed (see the description of argument TAIL in Section 5).
The Wilcoxon test differs from the Sign test (see G08AAF) in that the magnitude of the scores is taken into account, rather than simply the direction of such scores.

The test procedure is as follows:

(a) For each x_i, for $i = 1,2,...,n$, the signed difference $d_i = x_i - X_{med}$ is found, where X_{med} is a given test value for the median of the sample.

(b) The absolute differences $|d_i|$ are ranked with rank r_i and any tied values of $|d_i|$ are assigned the average of the tied ranks. The user may choose whether or not to ignore any cases where $d_i = 0$ by removing them before or after ranking (see the description of the argument ZEROS in Section 5).

(c) The number of non-zero d_i's is found.

(d) To each rank is affixed the sign of the d_i to which it corresponds. Let $s_i = \mathrm{sign}(d_i)r_i$

(e) The sum of the positive-signed ranks, $W = \sum_{s_i > 0} s_i = \sum_{i=1}^{n} \max(s_i,0.0)$, is calculated.

G08AGF returns:

(a) The test statistic W;

(b) The number n_1 of non-zero d_i's;

(c) The approximate Normal test statistic z,

$$\text{where } z = \frac{\left(W-\frac{n_1(n_1+1)}{4}\right) - \operatorname{sign}\left(W-\frac{n_1(n_1+1)}{4}\right)\times\frac{1}{2}}{\sqrt{\frac{1}{4}\sum_{i=1}^{n} s_i^2}}$$

(d) The tail probability, p, corresponding to W, depending on the choice of the alternative hypothesis, H_1.

If $n_1 \leq 80$, p is computed exactly; otherwise, an approximation to p is returned based on an approximate Normal statistic corrected for continuity according to the tail specified.

The value of p can be used to perform a significance test on the median against the alternative hypothesis. Let α be the size of the significance test (that is, α is the probability of rejecting H_0 when H_0 is true). If $p < \alpha$ then the null hypothesis is rejected. Typically α might be 0.05 or 0.01.

4. References

[1] CONOVER, W.J.
Practical Nonparametric Statistics, Ch. 5, pp. 280-293.
John Wiley & Sons, New York, 1980.

[2] NEUMANN, N.
Some Procedures for Calculating the Distributions of Elementary Nonparametric Teststatistics.
Statistical Software Newsletter, 14, 3, pp. 120-126, 1988.

[3] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences, Ch. 5, pp. 75-83.
McGraw-Hill, 1956.

5. Parameters

1: N – INTEGER. *Input*

On entry: the size of the sample, n.

Constraint: N $\geq$ 1.

2: X(N) – ***real*** array. *Input*

On entry: the sample observations, $x_1,x_2,\ldots,x_n$.

3: XME – ***real.*** *Input*

On entry: the median test value, X_{med}.

4: TAIL – CHARACTER*1. *Input*

On entry: indicates the choice of tail probability, and hence the alternative hypothesis.

If TAIL = 'T' or 't', then a two-tailed probability is calculated and the alternative hypothesis is H_1 : population median $\neq X_{med}$.

If TAIL = 'U' or 'u', then a upper-tailed probability is calculated and the alternative hypothesis is H_1 : population median $> X_{med}$.

If TAIL = 'L' or 'l', then a lower-tailed probability is calculated and the alternative hypothesis is H_1 : population median $< X_{med}$.

Constraint: TAIL = 'T', 't', 'U', 'u', 'L' or 'l'.

5: ZEROS – CHARACTER*1. *Input*

On entry: indicates whether or not to include the cases where $d_i = 0.0$ in the ranking of the d_i's.

If ZEROS = 'Y' or 'y', all $d_i = 0.0$ are included when ranking.
If ZEROS = 'N' or 'n', all $d_i = 0.0$, are ignored, that is all cases where $d_i = 0.0$ are removed before ranking.

Constraint: ZEROS = 'Y', 'y', 'N' or 'n'.

6: W – ***real.*** *Output*

On exit: the Wilcoxon rank sum statistic, W, being the sum of the positive ranks.

7: WNOR – ***real.*** *Output*

On exit: the approximate Normal test statistic, z, as described in Section 3.

8: P – ***real.*** *Output*

On exit: the tail probability, p, as specified by the parameter TAIL.

9: N1 – INTEGER. *Output*

On exit: the number of non-zero d_i's, n_1.

10: WRK(3*N) – ***real*** array. *Workspace*

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, TAIL ≠ 'T', 't', 'U', 'u', 'L' or 'l'.
or ZEROS ≠ 'Y', 'y', 'N' or 'n'.

IFAIL = 2

On entry, N < 1.

IFAIL = 3

The whole sample is identical to the given median test value.

7. Accuracy

The approximation used to calculate p when $n_1 > 80$ will return a value with a relative error of less than 10 percent for most cases. The error may increase for cases where there are a large number of ties in the sample.

8. Further Comments

The time taken by the routine increases with n_1, until $n_1 > 80$, from which point on the approximation is used. The time decreases significantly at this point and increases again modestly with n_1 for $n_1 > 80$.

9. Example

The following example performs the Wilcoxon signed rank test on two matched samples of size 8, taken from two populations. The distribution of the differences between pairs of observations from the two populations is assumed to be symmetric. The test is used to test whether the medians of the two distributions of the populations are equal or not. The test statistic, the approximate Normal statistic and the two-tailed probability are computed and printed.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08AGF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          MAXN
      PARAMETER        (MAXN=10)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      real             XME
      PARAMETER        (XME=0.0e0)
*     .. Local Scalars ..
      real             P, RS, RSNOR
      INTEGER          I, IFAIL, N, NZ1
*     .. Local Arrays ..
      real             WRK(3*MAXN), X(MAXN), Y(MAXN), Z(MAXN)
*     .. External Subroutines ..
      EXTERNAL         G08AGF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08AGF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.MAXN) THEN
         READ (NIN,*) (X(I),I=1,N), (Y(I),I=1,N)
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Wilcoxon one sample signed ranks test'
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Data values'
         WRITE (NOUT,99999) (X(I),I=1,N), (Y(I),I=1,N)
         DO 20 I = 1, N
            Z(I) = X(I) - Y(I)
   20    CONTINUE
         IFAIL = 0
*
         CALL G08AGF(N,Z,XME,'Two-tail','Nozeros',RS,RSNOR,P,NZ1,WRK,
     +               IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'Test statistic            = ', RS
         WRITE (NOUT,99998) 'Normalized test statistic = ', RSNOR
         WRITE (NOUT,99997) 'Degrees of freedom        = ', NZ1
         WRITE (NOUT,99998) 'Two tail probability      = ', P
      ELSE
         WRITE (NOUT,99996) 'N is too large : N = ', N
      END IF
      STOP
*
99999 FORMAT (4X,8F5.1)
99998 FORMAT (1X,A,F8.4)
99997 FORMAT (1X,A,I8)
99996 FORMAT (1X,A,I16)
      END
```

9.2. Program Data

```
G08AGF Example Program Data
 8
 82 69 73 43 58 56 76 65
 63 42 74 37 51 43 80 62
```

9.3. Program Results

```
G08AGF Example Program Results

Wilcoxon one sample signed ranks test

Data values
    82.0 69.0 73.0 43.0 58.0 56.0 76.0 65.0
    63.0 42.0 74.0 37.0 51.0 43.0 80.0 62.0

Test statistic            =  32.0000
Normalized test statistic =   1.8904
Degrees of freedom        =        8
Two tail probability      =   0.0547
```

G08AHF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G08AHF performs the Mann–Whitney U test on two independent samples of possibly unequal size.

2 Specification

```
 SUBROUTINE G08AHF(N1, X, N2, Y, TAIL, U, UNOR, P, TIES, RANKS,
1                  WRK, IFAIL)
 INTEGER           N1, N2, IFAIL
 real              X(N1), Y(N2), U, UNOR, P, RANKS(N1+N2),
1                  WRK(N1+N2)
 LOGICAL           TIES
 CHARACTER*1       TAIL
```

3 Description

The Mann–Whitney U test investigates the difference between two populations defined by the distribution functions $F(x)$ and $G(y)$ respectively. The data consist of two independent samples of size n_1 and n_2, denoted by $x_1, x_2, \ldots, x_{n_1}$ and $y_1, y_2, \ldots, y_{n_2}$, taken from the two populations.

The hypothesis under test, H_0, often called the null hypothesis, is that the two distributions are the same, that is $F(x) = G(x)$, and this is to be tested against an alternative hypothesis H_1 which is

$H_1 : F(x) \neq G(y)$; or

$H_1 : F(x) < G(y)$, i.e., the x's tend to be greater than the y's; or

$H_1 : F(x) > G(y)$, i.e., the x's tend to be less than the y's,

using a two-tailed, upper-tailed or lower-tailed probability respectively. The user selects the alternative hypothesis by choosing the appropriate tail probability to be computed (see the description of argument TAIL in Section 5).

Note that when using this test to test for differences in the distributions one is primarily detecting differences in the location of the two distributions. That is to say, if we reject the null hypothesis H_0 in favour of the alternative hypothesis H_1: $F(x) > G(y)$ we have evidence to suggest that the location, of the distribution defined by $F(x)$, is less than the location, of the distribution defined by $G(y)$.

The Mann–Whitney U test differs from the Median test (see G08ACF) in that the ranking of the individual scores within the pooled sample is taken into account, rather than simply the position of a score relative to the median of the pooled sample. It is therefore a more powerful test if score differences are meaningful.

The test procedure involves ranking the pooled sample, average ranks being used for ties. Let r_{1i} be the rank assigned to x_i, $i = 1, 2, \ldots, n_1$ and r_{2j} the rank assigned to y_j, $j = 1, 2, \ldots, n_2$. Then the test statistic U is defined as follows;

$$U = \sum_{i=1}^{n_1} r_{1i} - \frac{n_1(n_1+1)}{2}$$

U is also the number of times a score in the second sample precedes a score in the first sample (where we only count a half if a score in the second sample actually equals a score in the first sample).

G08AHF returns:

(a) The test statistic U.

(b) The approximate Normal test statistic,

$$z = \frac{U - \text{mean}(U) \pm \frac{1}{2}}{\sqrt{\text{var}(U)}}$$

where

$$\text{mean}(U) = \frac{n_1 n_2}{2}$$

and

$$\text{var}(U) = \frac{n_1 n_2(n_1+n_2+1)}{12} - \frac{n_1 n_2}{(n_1+n_2)(n_1+n_2-1)} \times TS$$

where

$$TS = \sum_{j=1}^{\tau} \frac{(t_j)(t_j-1)(t_j+1)}{12}$$

τ is the number of groups of ties in the sample and t_j is the number of ties in the jth group.

Note that if no ties are present the variance of U reduces to $\frac{n_1 n_2}{12}(n_1+n_2+1)$.

(c) An indicator as to whether ties were present in the pooled sample or not.

(d) The tail probability, p, corresponding to U (adjusted to allow the complement to be used in an upper 1-tailed or a 2-tailed test), depending on the choice of TAIL, i.e., the choice of alternative hypothesis, H_1. The tail probability returned is an approximation of p is based on an approximate Normal statistic corrected for continuity according to the tail specified. If n_1 and n_2 are not very large an exact probability may be desired. For the calculation of the exact probability see G08AJF (no ties in the pooled sample) or G08AKF (ties in the pooled sample).

The value of p can be used to perform a significance test on the null hypothesis H_0 against the alternative hypothesis H_1. Let α be the size of the significance test (that is, α is the probability of rejecting H_0 when H_0 is true). If $p < \alpha$ then the null hypothesis is rejected. Typically α might be 0.05 or 0.01.

4 References

[1] Conover W J (1980) *Practical Nonparametric Statistics* Wiley

[2] Neumann N (1988) Some procedures for calculating the distributions of elementary nonparametric teststatistics *Statistical Software Newsletter* **14 (3)** 120–126

[3] Siegel S (1956) *Non-parametric Statistics for the Behavioral Sciences* McGraw-Hill

5 Parameters

1: N1 — INTEGER *Input*

On exit: the number of non-tied pairs, n_1.

2: X(N1) — ***real*** array *Input*

On entry: the first vector of observations, $x_1, x_2, \ldots, x_{n_1}$.

3: N2 — INTEGER *Input*

On entry: the size of the second sample, n_2.

Constraint: N2 $\geq$ 1.

4: Y(N2) — ***real*** array *Input*

On entry: the second vector of observations. $y_1, y_2, \ldots, y_{n_2}$.

5: TAIL — CHARACTER*1 *Input*

On entry: indicates the choice of tail probability, and hence the alternative hypothesis.

If TAIL = 'T', then a two-tailed probability is calculated and the alternative hypothesis is $H_1 : F(x) \neq G(y)$.

If TAIL = 'U', then an upper-tailed probability is calculated and the alternative hypothesis $H_1 : F(x) < G(y)$, i.e., the x's tend to be greater than the y's.

If TAIL = 'L', then a lower-tailed probability is calculated and the alternative hypothesis $H_1 : F(x) > G(y)$, i.e., the x's tend to be less than the y's.

Constraint: TAIL = 'T', 'U' or 'L'.

6: U — ***real*** *Output*

On exit: the Mann–Whitney rank sum statistic, U.

7: UNOR — ***real*** *Output*

On exit: the approximate Normal test statistic, z, as described in Section 3.

8: P — ***real*** *Output*

On exit: the tail probability, p, as specified by the parameter TAIL.

9: TIES — LOGICAL *Output*

On exit: indicates whether the pooled sample contained ties or not. This will be useful in checking which routine to use should one wish to calculate an exact tail probability.

TIES = .FALSE., no ties were present (use G08AJF for an exact probability).

TIES = .TRUE., ties were present (use G08AKF for an exact probability).

10: RANKS(N1+N2) — ***real*** array *Output*

On exit: contains the ranks of the pooled sample. The ranks of the first sample are contained in the first N1 elements and those of the second sample are contained in the next N2 elements.

11: WRK(N1+N2) — ***real*** array *Workspace*

12: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N1 < 1,
or N2 < 1.

IFAIL = 2

On entry, TAIL ≠ 'T', 'U' or 'L'.

IFAIL = 3

The pooled sample values are all the same, that is the variance of U = 0.0.

7 Accuracy

The approximate tail probaility, p, returned by G08AHF is a good approximation to the exact probability for cases where $\max(n_1, n_2) \geq 30$ and $(n_1 + n_2) \geq 40$. The relative error of the approximation should be less than 10 percent, for most cases falling in this range.

8 Further Comments

The time taken by the routine increases with n_1 and n_2.

9 Example

The example program performs the Mann–Whitney test on two independent samples of sizes 16 and 23 respectively. This is used to test the null hypothesis that the distributions of the two populations from which the samples were taken are the same against the alternative hypothesis that the distributions are different. The test statistic, the approximate Normal statistic and the approximate two-tail probability are printed. An exact tail probability is also calculated and printed depending on whether ties were found in the pooled sample or not.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08AHF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MAXN1, MAXN2, MAXLW, MAXIW
      PARAMETER        (MAXN1=25,MAXN2=25,MAXLW=8000,MAXIW=100)
*     .. Local Scalars ..
      real             P, PEXACT, U, UNOR
      INTEGER          I, IFAIL, LWRK, N, N1, N2, NSUM
      LOGICAL          TIES
*     .. Local Arrays ..
      real             RANKS(MAXN1+MAXN2), WRK(MAXLW), X(MAXN1),
     +                 Y(MAXN2)
      INTEGER          IWRK(MAXIW)
*     .. External Subroutines ..
      EXTERNAL         G08AHF, G08AJF, G08AKF
*     .. Intrinsic Functions ..
      INTRINSIC        INT, MIN
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08AHF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N1, N2
      WRITE (NOUT,*)
      WRITE (NOUT,99999) 'Sample size of group 1 = ', N1
      WRITE (NOUT,99999) 'Sample size of group 2 = ', N2
      WRITE (NOUT,*)
      IF (N1.LE.MAXN1 .AND. N2.LE.MAXN2) THEN
         READ (NIN,*) (X(I),I=1,N1)
         WRITE (NOUT,*) 'Mann-Whitney U test'
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Data values'
         WRITE (NOUT,*)
         WRITE (NOUT,99998) '    Group 1  ', (X(I),I=1,N1)
         READ (NIN,*) (Y(I),I=1,N2)
         WRITE (NOUT,*)
         WRITE (NOUT,99998) '    Group 2  ', (Y(I),I=1,N2)
         IFAIL = 0
*
         CALL G08AHF(N1,X,N2,Y,'Lower-tail',U,UNOR,P,TIES,RANKS,WRK,
     +               IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99997) 'Test statistic           = ', U
         WRITE (NOUT,99997) 'Normal Statistic         = ', UNOR
         WRITE (NOUT,99997) 'Approx. tail probability = ', P
         WRITE (NOUT,*)
         IF ( .NOT. TIES) THEN
            WRITE (NOUT,*) 'There are no ties in the pooled sample'
            LWRK = INT(N1*N2/2) + 1
*
            CALL G08AJF(N1,N2,'Lower-tail',U,PEXACT,WRK,LWRK,IFAIL)
*
         ELSE
            WRITE (NOUT,*) 'There are ties in the pooled sample'
            N = MIN(N1,N2)
            NSUM = N1 + N2
            LWRK = N + N*(N+1)*NSUM - N*(N+1)*(2*N+1)/3 + 1
*
            CALL G08AKF(N1,N2,'Lower-tail',RANKS,U,PEXACT,WRK,LWRK,IWRK,
     +                  IFAIL)
*
         END IF
         WRITE (NOUT,*)
         WRITE (NOUT,99997) 'Exact tail probability   = ', PEXACT
      ELSE
         WRITE (NOUT,*) 'Either N1 or N2 is out of range :'
         WRITE (NOUT,99996) 'N1 = ', N1, ' and N2 = ', N2
      END IF
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,A,8F5.1,2(/14X,8F5.1))
99997 FORMAT (1X,A,F10.4)
99996 FORMAT (1X,A,I16,A,I16)
      END
```

9.2 Program Data

```
G08AHF Example Program Data
 16 23
 13.0  6.0 12.0  7.0 12.0  7.0 10.0  7.0
 10.0  7.0 16.0  7.0 10.0  8.0  9.0  8.0
 17.0  6.0 10.0  8.0 15.0  8.0 15.0 10.0 15.0 10.0 14.0 10.0
 14.0 11.0 14.0 11.0 13.0 12.0 13.0 12.0 13.0 12.0 12.0
```

9.3 Program Results

```
G08AHF Example Program Results

Sample size of group 1 =    16
Sample size of group 2 =    23

Mann-Whitney U test

Data values

    Group 1   13.0  6.0 12.0  7.0 12.0  7.0 10.0  7.0
              10.0  7.0 16.0  7.0 10.0  8.0  9.0  8.0


    Group 2   17.0  6.0 10.0  8.0 15.0  8.0 15.0 10.0
              15.0 10.0 14.0 10.0 14.0 11.0 14.0 11.0
              13.0 12.0 13.0 12.0 13.0 12.0 12.0

Test statistic           =     86.0000
Normal Statistic         =     -2.8039
Approx. tail probability =      0.0025

There are ties in the pooled sample

Exact tail probability   =      0.0020
```

G08AJF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G08AJF calculates the exact tail probability for the Mann–Whitney rank sum test statistic for the case where there are no ties in the two samples pooled together.

2 Specification

```
SUBROUTINE G08AJF(N1, N2, TAIL, U, P, WRK, LWRK, IFAIL)
INTEGER          N1, N2, LWRK, IFAIL
real             U, P, WRK(LWRK)
CHARACTER*1      TAIL
```

3 Description

G08AJF computes the exact tail probability for the Mann–Whitney U test statistic (calculated by G08AHF and returned through the parameter U) using a method based on an algorithm developed by Harding [2], and presented by Neumann [3], for the case where there are no ties in the pooled sample.

The Mann–Whitney U test investigates the difference between two populations defined by the distribution functions $F(x)$ and $G(y)$ respectively. The data consist of two independent samples of size n_1 and n_2, denoted by $x_1, x_2, \ldots, x_{n_1}$ and $y_1, y_2, \ldots, y_{n_2}$, taken from the two populations.

The hypothesis under test, H_0, often called the null hypothesis, is that the two distributions are the same, that is $F(x) = G(x)$, and this is to be tested against an alternative hypothesis H_1 which is

$H_1 : F(x) \neq G(y)$; or

$H_1 : F(x) < G(y)$, i.e., the x's tend to be greater than the y's; or

$H_1 : F(x) > G(y)$, i.e., the x's tend to be less than the y's,

using a two-tailed, upper-tailed or lower-tailed probability respectively. The user selects the alternative hypothesis by choosing the appropriate tail probability to be computed (see the description of argument TAIL in Section 5).

Note that when using this test to test for differences in the distributions one is primarily detecting differences in the location of the two distributions. That is to say, if we reject the null hypothesis H_0 in favour of the alternative hypothesis H_1: $F(x) > G(y)$ we have evidence to suggest that the location, of the distribution defined by $F(x)$, is less than the location, of the distribution defined by $G(y)$.

G08AJF returns the exact tail probability, p, corresponding to U, depending on the choice of alternative hypothesis, H_1.

The value of p can be used to perform a significance test on the null hypothesis H_0 against the alternative hypothesis H_1. Let α be the size of the significance test (that is, α is the probability of rejecting H_0 when H_0 is true). If $p < \alpha$ then the null hypothesis is rejected. Typically α might be 0.05 or 0.01.

4 References

[1] Conover W J (1980) *Practical Nonparametric Statistics* Wiley

[2] Harding E F (1983) An efficient minimal-storage procedure for calculating the Mann–Whitney U, generalised U and similar distributions *Appl. Statist.* **33** 1–6

[3] Neumann N (1988) Some procedures for calculating the distributions of elementary nonparametric teststatistics *Statistical Software Newsletter* **14** **(3)** 120–126

[4] Siegel S (1956) *Non-parametric Statistics for the Behavioral Sciences* McGraw-Hill

5 Parameters

1: N1 — INTEGER *Input*

On exit: the number of non-tied pairs, n_1.

2: N2 — INTEGER *Input*

On entry: the size of the second sample, n_2.

Constraint: N2 $\geq$ 1.

3: TAIL — CHARACTER*1 *Input*

On entry: indicates the choice of tail probability, and hence the alternative hypothesis.

If TAIL = 'T', then a two-tailed probability is calculated and the alternative hypothesis is $H_1 : F(x) \neq G(y)$.

If TAIL = 'U', then an upper-tailed probability is calculated and the alternative hypothesis $H_1 : F(x) < G(y)$, i.e., the x's tend to be greater than the y's.

If TAIL = 'L', then a lower-tailed probability is calculated and the alternative hypothesis $H_1 : F(x) > G(y)$, i.e., the x's tend to be less than the y's.

Constraint: TAIL = 'T', 'U' or 'L'.

4: U — ***real*** *Input*

On entry: the value of the Mann–Whitney rank sum test statistic, U. This is the statistic returned through the parameter U by G08AHF.

Constraint: U $\geq$ 0.0.

5: P — ***real*** *Output*

On exit: the exact tail probability, p, as specified by the parameter TAIL.

6: WRK(LWRK) — ***real*** array *Workspace*

7: LWRK — INTEGER *Input*

On entry: the dimension of the array WRK as declared in the (sub)program from which G08AJF is called.

Constraint: LWRK $\geq$ (N1 $\times$ N2)/2 + 1.

8: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N1 < 1,
or N2 < 1.

IFAIL = 2

On entry, TAIL ≠ 'T', 'U' or 'L'.

IFAIL = 3

On entry, U < 0.0.

IFAIL = 4

On entry, LWRK < (N1 × N2)/2 + 1.

7 Accuracy

The exact tail probability, p, is computed to an accuracy of at least 4 significant figures.

8 Further Comments

The time taken by the routine increases with n_1 and n_2 and the product $n_1 n_2$.

9 Example

The example program finds the Mann–Whitney test statistic, using G08AHF for two independent samples of size 16 and 23 respectively. This is used to test the null hypothesis that the distributions of the two populations from which the samples were taken are the same against the alternative hypothesis that the distributions are different. The test statistic, the approximate normal statistic and the approximate two-tail probability are printed. G08AJF is then called to obtain the exact two-tailed probability. The exact probability is also printed.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08AJF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MAXN1, MAXN2, MAXL
      PARAMETER        (MAXN1=25,MAXN2=25,MAXL=200)
*     .. Local Scalars ..
      real             P, PEXACT, U, UNOR
      INTEGER          I, IFAIL, LWRK, N1, N2
      LOGICAL          TIES
*     .. Local Arrays ..
      real             RANKS(MAXN1+MAXN2), WRK(MAXL), X(MAXN1), Y(MAXN2)
*     .. External Subroutines ..
      EXTERNAL         G08AHF, G08AJF
*     .. Intrinsic Functions ..
      INTRINSIC        INT
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08AJF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N1, N2
      WRITE (NOUT,*)
      IF (N1.LE.MAXN1 .AND. N2.LE.MAXN2) THEN
```

```
         WRITE (NOUT,99999) 'Sample size of group 1 = ', N1
         WRITE (NOUT,99999) 'Sample size of group 2 = ', N2
         WRITE (NOUT,*)
         READ (NIN,*) (X(I),I=1,N1)
         WRITE (NOUT,*) 'Mann-Whitney U test'
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Data values'
         WRITE (NOUT,*)
         WRITE (NOUT,99998) '    Group 1   ', (X(I),I=1,N1)
         READ (NIN,*) (Y(I),I=1,N2)
         WRITE (NOUT,*)
         WRITE (NOUT,99998) '    Group 2   ', (Y(I),I=1,N2)
         IFAIL = 0
*
         CALL G08AHF(N1,X,N2,Y,'Lower-tail',U,UNOR,P,TIES,RANKS,WRK,
     +               IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99997) 'Test statistic     = ', U
         WRITE (NOUT,99997) 'Normal statistic   = ', UNOR
         WRITE (NOUT,99997) 'Tail probability   = ', P
         WRITE (NOUT,*)
         IF ( .NOT. TIES) THEN
            LWRK = INT(N1*N2/2) + 1
            WRITE (NOUT,99996)
     +        'The length of the workspace is calculated as ', LWRK
            IFAIL = 0
*
            CALL G08AJF(N1,N2,'Lower-tail',U,PEXACT,WRK,LWRK,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,99997) 'Exact tail probability = ', PEXACT
         ELSE
            WRITE (NOUT,*)
     +   'There are ties in the pooled sample so G08AJF was not called.'
         END IF
      ELSE
         WRITE (NOUT,*) 'Either N1 or N2 is out of range :'
         WRITE (NOUT,99995) 'N1 = ', N1, ' and N2 = ', N2
      END IF
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,A,8F5.1,2(/14X,8F5.1))
99997 FORMAT (1X,A,F10.4)
99996 FORMAT (1X,A,I10)
99995 FORMAT (1X,A,I16,A,I16)
      END
```

9.2 Program Data

```
G08AJF Example Program Data
 16 23
 13.0  5.8 11.7  6.5 12.3  6.7  9.2  6.9
 10.0  7.3 16.0  7.0 10.5  8.5  9.0  7.5
 17.0  6.2 10.1  8.0 15.3  8.2 15.0  9.6 14.9 10.4 14.2  9.8
 13.8 11.0 14.0 11.1 12.9 11.6 12.8 12.0 13.1 12.4 11.9
```

9.3 Program Results

```
G08AJF Example Program Results

Sample size of group 1 =    16
Sample size of group 2 =    23

Mann-Whitney U test

Data values

    Group 1   13.0  5.8 11.7  6.5 12.3  6.7  9.2  6.9
              10.0  7.3 16.0  7.0 10.5  8.5  9.0  7.5


    Group 2   17.0  6.2 10.1  8.0 15.3  8.2 15.0  9.6
              14.9 10.4 14.2  9.8 13.8 11.0 14.0 11.1
              12.9 11.6 12.8 12.0 13.1 12.4 11.9

Test statistic     =    86.0000
Normal statistic   =    -2.7838
Tail probability   =     0.0027

The length of the workspace is calculated as        185

Exact tail probability =     0.0022
```

G08AKF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G08AKF calculates the exact tail probability for the Mann–Whitney rank sum test statistic for the case where there are ties in the two samples pooled together.

2 Specification

```
SUBROUTINE G08AKF(N1, N2, TAIL, RANKS, U, P, WRK, LWRK, IWRK, IFAIL)
INTEGER          N1, N2, LWRK, IWRK(2*(N1+N2+1)), IFAIL
real             RANKS(N1+N2), U, P, WRK(LWRK)
CHARACTER*1      TAIL
```

3 Description

G08AKF computes the exact tail probability for the Mann–Whitney U test statistic (calculated by G08AHF and returned through the parameter U) using a method based on an algorithm developed by Neumann [2], for the case where there are ties in the pooled sample.

The Mann–Whitney U test investigates the difference between two populations defined by the distribution functions $F(x)$ and $G(y)$ respectively. The data consist of two independent samples of size n_1 and n_2, denoted by $x_1, x_2, \ldots, x_{n_1}$ and $y_1, y_2, \ldots, y_{n_2}$, taken from the two populations.

The hypothesis under test, H_0, often called the null hypothesis, is that the two distributions are the same, that is $F(x) = G(x)$, and this is to be tested against an alternative hypothesis H_1 which is

$H_1 : F(x) \neq G(y)$; or

$H_1 : F(x) < G(y)$, i.e., the x's tend to be greater than the y's; or

$H_1 : F(x) > G(y)$, i.e., the x's tend to be less than the y's,

using a two-tailed, upper-tailed or lower-tailed probability respectively. The user selects the alternative hypothesis by choosing the appropriate tail probability to be computed (see the description of argument TAIL in Section 5).

Note that when using this test to test for differences in the distributions one is primarily detecting differences in the location of the two distributions. That is to say, if we reject the null hypothesis H_0 in favour of the alternative hypothesis H_1: $F(x) > G(y)$ we have evidence to suggest that the location, of the distribution defined by $F(x)$, is less than the location, of the distribution defined by $G(y)$.

G08AKF returns the exact tail probability, p, corresponding to U, depending on the choice of alternative hypothesis, H_1.

The value of p can be used to perform a significance test on the null hypothesis H_0 against the alternative hypothesis H_1. Let α be the size of the significance test (that is α is the probability of rejecting H_0 when H_0 is true). If $p < \alpha$ then the null hypothesis is rejected. Typically α might be 0.05 or 0.01.

4 References

[1] Conover W J (1980) *Practical Nonparametric Statistics* Wiley

[2] Neumann N (1988) Some procedures for calculating the distributions of elementary nonparametric teststatistics *Statistical Software Newsletter* **14 (3)** 120–126

[3] Siegel S (1956) *Non-parametric Statistics for the Behavioral Sciences* McGraw-Hill

5 Parameters

1: N1 — INTEGER *Input*

On exit: the number of non-tied pairs, n_1.

2: N2 — INTEGER *Input*

On entry: the size of the second sample, n_2.

Constraint: N2 $\geq$ 1.

3: TAIL — CHARACTER*1 *Input*

On entry: indicates the choice of tail probability, and hence the alternative hypothesis.

If TAIL = 'T', then a two-tailed probability is calculated and the alternative hypothesis is $H_1 : F(x) \neq G(y)$.

If TAIL = 'U', then an upper-tailed probability is calculated and the alternative hypothesis $H_1 : F(x) < G(y)$, i.e., the x's tend to be greater than the y's.

If TAIL = 'L', then a lower-tailed probability is calculated and the alternative hypothesis $H_1 : F(x) > G(y)$, i.e., the x's tend to be less than the y's.

Constraint: TAIL = 'T', 'U' or 'L'.

4: RANKS(N1+N2) — ***real*** array *Input*

On entry: the ranks of the pooled sample. These ranks are output in the array RANKS by G08AHF and should not be altered in any way if the user is using the same n_1, n_2 and U as used in G08AHF.

5: U — ***real*** *Input*

On entry: the value of the Mann–Whitney rank sum test statistic, U. This is the statistic returned through the parameter U by G08AHF.

6: P — ***real*** *Output*

On exit: the tail probability, p, as specified by the parameter TAIL.

7: WRK(LWRK) — ***real*** array *Workspace*

8: LWRK — INTEGER *Input*

On entry: the dimension of the array WRK as declared in the (sub)program from which G08AKF is called.

Constraint: LWRK $\geq n + n(n+1)(n+m) - \frac{n(n+1)(2\times n+1)}{3} + 1$, where n = min(N1,N2) and m = max(N1,N2).

9: IWRK(2*(N1+N2+1)) — INTEGER array *Workspace*

10: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N1 < 1,
or N2 < 1.

IFAIL = 2

On entry, TAIL ≠ 'T', 'U' or 'L'.

IFAIL = 3

On entry, U < 0.0.

IFAIL = 4

On entry, LWRK is too small.

7 Accuracy

The exact tail probability, p, is computed to an accuracy of at least 4 significant figures.

8 Further Comments

The time taken by the routine increases with n_1 and n_2 and the product n_1n_2. Note that the amount of workspace required becomes very large for even moderate sizes of n_1 and n_2.

9 Example

The example program finds the Mann–Whitney test statistic, using G08AHF for two independent samples of size 16 and 23 respectively. This is used to test the null hypothesis that the distributions of the two populations from which the samples were taken are the same against the alternative hypothesis that the distributions are different. The test statistic, the approximate Normal statistic and the approximate two-tail probability are printed. G08AKF is then called to obtain the exact two-tailed probability. The exact probability is also printed.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08AKF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MAXN1, MAXN2, MAXL, MAXIW
      PARAMETER        (MAXN1=25,MAXN2=25,MAXL=8000,MAXIW=100)
*     .. Local Scalars ..
      real             P, PEXACT, U, UNOR
      INTEGER          I, IFAIL, LWRK, N, N1, N2, NSUM
      LOGICAL          TIES
```

```
*     .. Local Arrays ..
      real             RANKS(MAXN1+MAXN2), WRK(MAXL), X(MAXN1), Y(MAXN2)
      INTEGER          IWRK(MAXIW)
*     .. External Subroutines ..
      EXTERNAL         G08AHF, G08AKF
*     .. Intrinsic Functions ..
      INTRINSIC        MIN
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08AKF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N1, N2
      WRITE (NOUT,*)
      IF ((N1.LE.MAXN1) .AND. (N2.LE.MAXN2)) THEN
         WRITE (NOUT,99999) 'Sample size of group 1 = ', N1
         WRITE (NOUT,99999) 'Sample size of group 2 = ', N2
         WRITE (NOUT,*)
         READ (NIN,*) (X(I),I=1,N1)
         WRITE (NOUT,*) 'Mann-Whitney U test'
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Data values'
         WRITE (NOUT,*)
         WRITE (NOUT,99998) '    Group 1  ', (X(I),I=1,N1)
         READ (NIN,*) (Y(I),I=1,N2)
         WRITE (NOUT,*)
         WRITE (NOUT,99998) '    Group 2  ', (Y(I),I=1,N2)
         IFAIL = 0
*
         CALL G08AHF(N1,X,N2,Y,'Lower-tail',U,UNOR,P,TIES,RANKS,WRK,
     +               IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99997) 'Test statistic     = ', U
         WRITE (NOUT,99997) 'Normal statistic   = ', UNOR
         WRITE (NOUT,99997) 'Tail probability   = ', P
         WRITE (NOUT,*)
         IF (TIES) THEN
            NSUM = N1 + N2
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Ranks'
            WRITE (NOUT,*)
            WRITE (NOUT,99998) '    Group 1  ', (RANKS(I),I=1,N1)
            WRITE (NOUT,*)
            WRITE (NOUT,99998) '    Group 2  ', (RANKS(I),I=N1+1,NSUM)
            N = MIN(N1,N2)
            LWRK = N + N*(N+1)*NSUM - N*(N+1)*(2*N+1)/3 + 1
            WRITE (NOUT,*)
            WRITE (NOUT,99996)
     +        'The length of the workspace is calculated as ', LWRK
            IFAIL = 0
*
            CALL G08AKF(N1,N2,'Lower-tail',RANKS,U,PEXACT,WRK,LWRK,IWRK,
     +                  IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,99997) 'Exact tail probability = ', PEXACT
         ELSE
```

```
            WRITE (NOUT,*)
     +'There are no ties in the pooled sample so G08AKF was not called.'
         END IF
      ELSE
         WRITE (NOUT,*) 'Either N or M is out of range :'
         WRITE (NOUT,99995) 'N1 = ', N1, ' AND N2 = ', N2
      END IF
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,A,8F5.1,2(/14X,8F5.1))
99997 FORMAT (1X,A,F10.4)
99996 FORMAT (1X,A,I10)
99995 FORMAT (1X,A,I16,A,I16)
      END
```

9.2 Program Data

```
G08AKF Example Program Data
 16 23
 13.0  6.0 12.0  7.0 12.0  7.0 10.0  7.0
 10.0  7.0 16.0  7.0 10.0  8.0  9.0  8.0
 17.0  6.0 10.0  8.0 15.0  8.0 15.0 10.0 15.0 10.0 14.0 10.0
 14.0 11.0 14.0 11.0 13.0 12.0 13.0 12.0 13.0 12.0 12.0
```

9.3 Program Results

```
G08AKF Example Program Results

Sample size of group 1 =    16
Sample size of group 2 =    23

Mann-Whitney U test

Data values

    Group 1   13.0  6.0 12.0  7.0 12.0  7.0 10.0  7.0
              10.0  7.0 16.0  7.0 10.0  8.0  9.0  8.0


    Group 2   17.0  6.0 10.0  8.0 15.0  8.0 15.0 10.0
              15.0 10.0 14.0 10.0 14.0 11.0 14.0 11.0
              13.0 12.0 13.0 12.0 13.0 12.0 12.0

Test statistic     =    86.0000
Normal statistic   =    -2.8039
Tail probability   =     0.0025


Ranks

    Group 1   29.5  1.5 24.5  5.0 24.5  5.0 16.0  5.0
              16.0  5.0 38.0  5.0 16.0  9.5 12.0  9.5
```

```
 Group 2   39.0  1.5 16.0  9.5 36.0  9.5 36.0 16.0
                36.0 16.0 33.0 16.0 33.0 20.5 33.0 20.5
                29.5 24.5 29.5 24.5 29.5 24.5 24.5

  The length of the workspace is calculated as       7633

  Exact tail probability =     0.0020
```

G08ALF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08ALF performs the Cochran Q test on cross-classified binary data.

2. Specification

```
SUBROUTINE G08ALF (N, K, X, LDX, Q, PROB, IFAIL)
INTEGER       N, K, LDX, IFAIL
real          X(LDX,K), Q, PROB
```

3. Description

Cochran's Q test may be used to test for differences between k treatments applied independently to n individuals or blocks (k related samples of equal size n), where the observed response can take only one of two possible values; for example a treatment may result in a 'success' or 'failure'. The data is recorded as either 1 or 0 to represent this dichotomization.

The use of this 'randomized block design' allows the effect of differences between the blocks to be separated from the differences between the treatments. The test assumes that the blocks were randomly selected from all possible blocks and that the result may be one of two possible outcomes common to all treatments within blocks.

The null and alternative hypotheses to be tested may be stated as follows;

H_0 : The treatments are equally effective, that is the probability of obtaining a 1 within a block is the same for each treatment.

H_1 : There is a difference between the treatments, that is the probability of obtaining a 1 is not the same for different treatments within blocks.

The data is often represented in the form of a table with the n rows representing the blocks and the k columns the treatments. Let R_i represent the row totals, for $i = 1,2,...,n$, and C_j represent the column totals, for $j = 1,2,...,k$. Let x_{ij} represent the response or result where $x_{ij} = 0$ or 1.

	Treatments				
Blocks	1	2		k	Row Totals
1	x_{11}	x_{12}	...	x_{1k}	R_1
2	x_{21}	x_{22}	...	x_{2k}	R_2
...	...	...	...	...	...
n	x_{n1}	x_{n2}	...	x_{nk}	R_n
Column Totals	C_1	C_2		C_k	N = Grand Total

If $p_{ij} = \Pr(x_{ij} = 1)$, for $i = 1,2,...,n$; $j = 1,2,...,k$, then the hypotheses may be restated as follows,

H_0 : $p_{i1} = p_{i2} = ... = p_{ik}$, for each $i = 1,2,...,n$,

H_1 : $p_{ij} \neq p_{ik}$, for some j and k, and for some i.

The test statistic is defined as:

$$Q = \frac{k(k-1)\sum_{j=1}^{k}\left(C_j - \frac{N}{k}\right)^2}{\sum_{i=1}^{n} R_i(k-R_i)}.$$

When the number of blocks, n, is large relative to the number of treatments, k, Q has an approximate χ^2 distribution with $k-1$ degrees of freedom. This is used to find the probability, p, of obtaining a statistic greater than or equal to the computed value of Q. Thus p is the upper-tail probability associated with the computed value of Q, where the χ^2 distribution is used to approximate the true distribution of Q.

4. References

[1] CONOVER, W.J.
Practical Nonparametric Statistics.
John Wiley & Sons, New York, Ch. 5, pp. 280-293, 1980.

[2] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences.
McGraw-Hill, Ch. 5, pp. 75-83, 1956.

5. Parameters

1: N – INTEGER. *Input*

On entry: the number of blocks, n.

Constraint: N $\geq$ 2.

2: K – INTEGER. *Input*

On entry: the number of treatments, k.

Constraint: K $\geq$ 2.

3: X(LDX,K) – ***real*** array. *Input*

On entry: the matrix of observed zero-one data. X(i,j) must contain the value x_{ij}, for $i = 1,2,...,n$; $j = 1,2,...,k$.

Constraint: X(i,j) = 0.0 or 1.0, for all $i = 1,2,...,n$; $j = 1,2,...,k$.

4: LDX – INTEGER. *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G08ALF is called.

Constraint: LDX $\geq$ N.

5: Q – ***real.*** *Output*

On exit: the value of the Cochran Q test statistic.

6: PROB – ***real.*** *Output*

On exit: the upper tail probability, p, associated with the Cochran Q test statistic, that is the probability of obtaining a value greater than or equal to the observed value (the output value of Q).

7: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N < 2,
or K < 2,
or LDX < N.

IFAIL = 2

On entry, X$(i,j) \neq 0.0$ or 1.0 for some i and j, $i = 1,2,...,n$; $j = 1,2,...,k$.

IFAIL = 3

The approximation process used to calculate the tail probability has failed to converge. The result returned in PROB may still be a reasonable approximation.

7. Accuracy

The use of the χ^2 distribution as an approximation to the true distribution of the Cochran Q test statistic improves as k increases and as n increases relative to k. This approximation should be a reasonable one when the total number of observations left, after omitting those rows containing all 0's or 1's, is greater than about 25 and the number of rows left is larger than 5.

8. Further Comments

None.

9. Example

The following example is taken from Conover [1], page 201. The data represent the success of three basketball enthusiasts in predicting the outcome of 12 collegiate basketball games, selected at random, using 1 for successful prediction of the outcome and 0 for unsuccessful prediction. This data is read in and the Cochran Q test statistic and its corresponding upper-tail probability are computed and printed.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08ALF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          LDX, KMAX
      PARAMETER        (LDX=20,KMAX=3)
*     .. Local Scalars ..
      real             DF, P, Q
      INTEGER          I, IFAIL, J, K, N
*     .. Local Arrays ..
      real             X(LDX,KMAX)
*     .. External Subroutines ..
      EXTERNAL         G08ALF
*     .. Intrinsic Functions ..
      INTRINSIC        real
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08ALF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, K
      IF (N.LE.LDX .AND. K.LE.KMAX) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Cochrans Q test'
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Data matrix'
         DO 20 I = 1, N
            READ (NIN,*) (X(I,J),J=1,K)
            WRITE (NOUT,99999) (X(I,J),J=1,K)
   20    CONTINUE
         IFAIL = 0
*
```

```
         CALL G08ALF(N,K,X,LDX,Q,P,IFAIL)
*
         DF = real(K-1)
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'Cochrans Q test statistic = ', Q
         WRITE (NOUT,99997) 'Degrees of freedom = ', DF
         WRITE (NOUT,99998) 'Upper-tail probability = ', P
      END IF
      STOP
*
99999 FORMAT (1X,3F6.1)
99998 FORMAT (1X,A,F12.4)
99997 FORMAT (1X,A,F6.1)
      END
```

9.2. Program Data

```
G08ALF Example Program Data
 12  3
 1 1 1
 1 1 1
 0 1 0
 1 1 0
 0 0 0
 1 1 1
 1 1 1
 1 1 0
 0 0 1
 0 1 0
 1 1 1
 1 1 1
```

9.3. Program Results

```
 G08ALF Example Program Results

 Cochrans Q test

 Data matrix
    1.0   1.0   1.0
    1.0   1.0   1.0
    0.0   1.0   0.0
    1.0   1.0   0.0
    0.0   0.0   0.0
    1.0   1.0   1.0
    1.0   1.0   1.0
    1.0   1.0   0.0
    0.0   0.0   1.0
    0.0   1.0   0.0
    1.0   1.0   1.0
    1.0   1.0   1.0

 Cochrans Q test statistic =       2.8000
 Degrees of freedom =    2.0
 Upper-tail probability =       0.2466
```

G08BAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08BAF performs Mood's and David's tests for dispersion differences between two independent samples of possibly unequal size.

2. Specification

```
SUBROUTINE G08BAF (X, N, N1, R, ITEST, W, V, PW, PV, IFAIL)
INTEGER          N, N1, ITEST, IFAIL
real             X(N), R(N), W, V, PW, PV
```

3. Description

Mood's and David's tests investigate the difference between the dispersions of two independent samples of sizes n_1 and n_2, denoted by:

$$x_1,x_2,\ldots,x_{n_1}$$

and

$$x_{n_1+1},x_{n_1+2},\ldots,x_n, \quad n = n_1+n_2.$$

The hypothesis under test, H_0, often called the null hypothesis, is that the dispersion difference is zero, and this is to be tested against a one- or two-sided alternative hypothesis H_1 (see below).

Both tests are based on the rankings of the sample members within the pooled sample formed by combining both samples. If there is some difference in dispersion, more of the extreme ranks will tend to be found in one sample than in the other.

Let the rank of x_i be denoted by r_i, for $i = 1,2,\ldots,n$.

(a) Mood's test.

The test statistic $W = \sum_{i=1}^{n_1}\left(r_i-\frac{n+1}{2}\right)^2$ is found.

W is the sum of squared deviations from the average rank in the pooled sample. For large n, W approaches normality, and so an approximation, p_w, to the probability of observing W not greater than the computed value, may be found.

G08BAF returns W and p_w if Mood's test is selected.

(b) David's test.

The disadvantage of Mood's test is that it assumes that the means of the two samples are equal. If this assumption is unjustified a high value of W could merely reflect the difference in means. David's test reduces this effect by using the variance of the ranks of the first sample about their mean rank, rather than the overall mean rank.

The test statistic for David's test is

$$V = \frac{1}{n_1 - 1}\sum_{i=1}^{n_1}(r_i-\bar{r})^2$$

where

$$\bar{r} = \frac{\sum_{i=1}^{n_1} r_i}{n_1}.$$

For large n, V approaches normality, enabling an approximate probability p_v to be computed, similarly to p_w.

G08BAF returns V and p_v if David's test is selected.

Hypothesis testing:

Suppose that a significance test of a chosen size α is to be performed (i.e. α is the probability of rejecting H_0 when H_0 is true; typically α is a small quantity such as 0.05 or 0.01).

The returned value p $(= p_v$ or $p_w)$ can be used to perform a significance test, against various alternative hypotheses H_1, as follows.

(i) H_1 : dispersions are unequal. H_0 is rejected if $2\times\min(p,1-p) < \alpha$.

(ii) H_1 : dispersion of sample 1 > dispersion of sample 2. H_0 is rejected if $1 - p < \alpha$.

(iii) H_1 : dispersion of sample 2 > dispersion of sample 1. H_0 is rejected if $p < \alpha$.

4. References

[1] COOPER, B.E.
Statistics for Experimentalists.
Pergamon Press, 1975.

5. Parameters

1: X(N) – ***real*** array. *Input*

On entry: the first n_1 elements of X must be set to the data values in the first sample, and the next n_2 $(= \text{N}-n_1)$ elements to the data values in the second sample.

2: N – INTEGER. *Input*

On entry: the total of the two sample sizes, n $(= n_1+n_2)$.

Constraint: N > 2.

3: N1 – INTEGER. *Input*

On entry: the size of the first sample, n_1.

Constraint: 1 < N1 < N.

4: R(N) – ***real*** array. *Output*

On exit: the ranks r_i, assigned to the data values x_i, for $i = 1,2,...,n$.

5: ITEST – INTEGER. *Input*

On entry: the test(s) to be carried out, using the codes:

0 – both Mood's and David's tests
1 – David's test only
2 – Mood's test only

Constraint: ITEST = 0, 1 or 2.

6: W – ***real.*** *Output*

On exit: Mood's test statistic, W, if requested.

7: V – ***real.*** *Output*

On exit: David's test statistic, V, if requested.

8: PW – ***real.*** *Output*

On exit: the lower tail probability, p_w, corresponding to the value of W, if Mood's test was requested.

9: PV – ***real.*** *Output*

On exit: the lower tail probability, p_v, corresponding to the value of V, if David's test was requested.

10: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, N ≤ 2.

IFAIL = 2

On entry, N1 ≤ 1,
or N1 ≥ N.

IFAIL = 3

On entry, ITEST < 0,
or ITEST > 2.

7. Accuracy

All computations are believed to be stable. The statistics V and W should be accurate enough for all practical uses.

8. Further Comments

The time taken by the routine is small, and increases with n.

9. Example

This example is taken from page 280 of Cooper [1]. The data consist of two samples of six observations each. Both Mood's and David's test statistics and significances are computed. Note that Mood's statistic is inflated owing to the difference in location of the two samples, the median ranks differing by a factor of two.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08BAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N
      PARAMETER        (N=12)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             PV, PW, V, W
      INTEGER          I, IFAIL, ITEST, N1
*     .. Local Arrays ..
      real             WK(N), X(N)
*     .. External Subroutines ..
      EXTERNAL         G08BAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08BAF Example Program Results'
```

```
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) X
      N1 = 6
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Mood''s test and David''s test'
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Data values'
      WRITE (NOUT,*)
      WRITE (NOUT,99999) '    Group 1  ', (X(I),I=1,N1)
      WRITE (NOUT,*)
      WRITE (NOUT,99999) '    Group 2  ', (X(I),I=N1+1,N)
      ITEST = 0
      IFAIL = 0
*
      CALL G08BAF(X,N,N1,WK,ITEST,W,V,PW,PV,IFAIL)
*
      WRITE (NOUT,99998) '    Mood''s measure  = ', W,
     +  '   Significance = ', PW
      WRITE (NOUT,99998) '    David''s measure = ', V,
     +  '   Significance = ', PV
      STOP
*
99999 FORMAT (1X,A,8F4.0,/(13X,8F4.0))
99998 FORMAT (1X,A,F8.3,A,F8.4)
      END
```

9.2. Program Data

```
G08BAF Example Program Data
  6.0  9.0 12.0  4.0 10.0 11.0
  8.0  1.0  3.0  7.0  2.0  5.0
```

9.3. Program Results

```
 G08BAF Example Program Results

 Mood's test and David's test

 Data values

     Group 1    6.  9. 12.  4. 10. 11.

     Group 2    8.  1.  3.  7.  2.  5.
     Mood's measure  =   75.500  Significance =   0.5830
     David's measure =    9.467  Significance =   0.1986
```

G08CBF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08CBF performs the one sample Kolmogorov-Smirnov test, using one of the standard distributions provided.

2. Specification

```
      SUBROUTINE G08CBF (N, X, DIST, PAR, ESTIMA, NTYPE, D, Z, P, SX,
     1                   IFAIL)
      INTEGER          N, NTYPE, IFAIL
      real             X(N), PAR(2), D, Z, P, SX(N)
      CHARACTER*(*)    DIST
      CHARACTER*1      ESTIMA
```

3. Description

The data consist of a single sample of n observations denoted by $x_1,x_2,...,x_n$. Let $S_n(x_{(i)})$ and $F_0(x_{(i)})$ represent the sample cumulative distribution function and the theoretical (null) cumulative distribution function respectively at the point $x_{(i)}$ where $x_{(i)}$ is the ith smallest sample observation.

The Kolmogorov-Smirnov test provides a test of the null hypothesis H_0 : the data are a random sample of observations from a theoretical distribution specified by the user against one of the following alternative hypotheses:

(i) H_1 : the data cannot be considered to be a random sample from the specified null distribution.

(ii) H_2 : the data arise from a distribution which dominates the specified null distribution. In practical terms, this would be demonstrated if the values of the sample cumulative distribution function $S_n(x)$ tended to exceed the corresponding values of the theoretical cumulative distribution function $F_0(x)$.

(iii) H_3 : the data arise from a distribution which is dominated by the specified null distribution. In practical terms, this would be demonstrated if the values of the theoretical cumulative distribution function $F_0(x)$ tended to exceed the corresponding values of the sample cumulative distribution function $S_n(x)$.

One of the following test statistics is computed depending on the particular alternative null hypothesis specified (see the description of the parameter NTYPE in Section 5).

For the alternative hypothesis H_1.

D_n – the largest absolute deviation between the sample cumulative distribution function and the theoretical cumulative distribution function. Formally $D_n = \max\{D_n^+,D_n^-\}$.

For the alternative hypothesis H_2.

D_n^+ – the largest positive deviation between the sample cumulative distribution function and the theoretical cumulative distribution function. Formally $D_n^+ = \max\{S_n(x_{(i)})-F_0(x_{(i)}),0\}$ for both discrete and continuous null distributions.

For the alternative hypothesis H_3.

D_n^- – the largest positive deviation between the theoretical cumulative distribution function and the sample cumulative distribution function. Formally if the null distribution is discrete then $D_n^- = \max\{F_0(x_{(i)})-S_n(x_{(i)}),0\}$ and if the null distribution is continuous then $D_n^- = \max\{F_0(x_{(i)})-S_n(x_{(i-1)}),0\}$.

The standardized statistic $Z = D\times\sqrt{n}$ is also computed where D may be D_n,D_n^+ or D_n^- depending on the choice of the alternative hypothesis. This is the standardised value of D with no correction for continuity applied and the distribution of Z converges asymptotically to a limiting distribution, first derived by Kolmogorov [4], and then tabulated by Smirnov [6]. The asymptotic distributions for the one-sided statistics were obtained by Smirnov [5].

The probability, under the null hypothesis, of obtaining a value of the test statistic as extreme as that observed, is computed. If $n \leq 100$ an exact method given by Conover [1], is used. Note that the method used is only exact for continuous theoretical distributions and does not include Conover's modification for discrete distributions. This method computes the one-sided probabilities. The two-sided probabilities are estimated by doubling the one-sided probability. This is a good estimate for small p, that is $p \leq 0.10$, but it becomes very poor for larger p. If $n > 100$ then p is computed using the Kolmogorov-Smirnov limiting distributions, see Feller [2], Kendall and Stuart [3], Kolmogorov [4], Smirnov [5] and [6].

4. References

[1] CONOVER, W.J.
Practical Nonparametric Statistics, Ch 6.
John Wiley & Sons, New York, 1980.

[2] FELLER W.
On the Kolmogorov-Smirnov Limit Theorems for Empirical Distributions.
Annals of Math. Stat., 19, pp. 179-181, 1948.

[3] KENDALL, M.G. and STUART, A.
The Advanced Theory of Statistics, Vol. 2, Ch. 30.
Griffin, London, 1973.

[4] KOLMOGOROV, A.N.
Sulla determinazione empirica di una legge di distribuzione.
Giornale dell' Istituto Italiano degli Attuari, 4, pp. 83-91, 1933.

[5] SMIRNOV, N.
Estimate of deviation between empirical distribution functions in two independent samples.
Bulletin Moscow University, 2(2), pp. 3-16, 1933.

[6] SMIRNOV, N.
Table for estimating the goodness of fit of empirical distributions.
Annals of Math. Stat., 19, pp. 279-281, 1948.

[7] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences, Ch. 4.
McGraw-Hill, 1956.

5. Parameters

1: N – INTEGER. *Input*

On entry: the number of observations in the sample, n.

Constraint: N $\geq$ 3.

2: X(N) – ***real*** array. *Input*

On entry: the sample observations $x_1,x_2,...,x_n$.

Constraint: the sample observations supplied must be consistent, in the usual manner, with the null distribution chosen, as specified by the parameters DIST and PAR. For further details see Section 8.

3: DIST – CHARACTER*(*). *Input*

On entry: the theoretical (null) distribution from which it is suspected the data may arise, as follows:

DIST = 'U' or 'u', uniform distribution over (a,b) – $U(a,b)$.

DIST = 'N' or 'n', Normal distribution with mean μ and variance σ^2 – $N(\mu,\sigma^2)$.

DIST = 'G' or 'g', gamma distribution with shape parameter α and scale parameter β, where the mean = $\alpha\beta$.

DIST = 'BE', 'Be', 'bE' or 'be', beta distribution with shape parameters α and β, where the mean = $\alpha/(\alpha+\beta)$.

DIST = 'BI', 'Bi', 'bI' or 'bi', binomial distribution with the number of trials, m, and the probability of a success, p.

DIST = 'E' or 'e', exponential distribution with parameter λ, where the mean = $1/\lambda$.

DIST = 'P' or 'p', poisson distribution with parameter μ, where the mean = μ.

Any number of characters may be supplied as the actual argument, however only the characters, maximum 2, required to uniquely identify the distribution are referenced.

4: PAR(2) – ***real*** array. *Input/Output*

On entry: if ESTIMA = 'S' or 's' , PAR must contain the known values of the parameter(s) of the null distribution as follows :

If a uniform distribution is used then PAR(1) and PAR(2) must contain the boundaries a and b respectively.

If a Normal distribution is used then PAR(1) and PAR(2) must contain the mean, μ, and the variance, σ^2, respectively.

If a gamma distribution is used then PAR(1) and PAR(2) must contain the parameters α and β respectively.

If a beta distribution is used then PAR(1) and PAR(2) must contain the parameters α and β respectively.

If a binomial distribution is used then PAR(1) and PAR(2) must contain the parameters m and p respectively.

If a exponential distribution is used then PAR(1) must contain the parameter λ.

If a poisson distribution is used then PAR(1) must contain the parameter μ.

If ESTIMA = 'E' or 'e', PAR need not be set except when the null distribution requested is the binomial distribution in which case PAR(1) must contain the parameter m.

On exit: if ESTIMA = 'S' or 's' PAR is unchanged. If ESTIMA = 'E' or 'e' then PAR(1) and PAR(2) are set to values as estimated from the data.

Constraints: if DIST = 'U' or 'u', PAR(1) < PAR(2),
if DIST = 'N' or 'n', PAR(2) > 0.0,
if DIST = 'G' or 'g', PAR(1) > 0.0 and PAR(2) > 0.0,
if DIST = 'BE', 'Be', 'bE' or 'be', PAR(1) > 0.0 and PAR(2) > 0.0, and
PAR(1) ≤ 10^6 and PAR(2) ≤ 10^6,
if DIST = 'BI', 'Bi', 'bI' or 'bi',
PAR(1) ≥ 1.0 and 0.0 < PAR(2) < 1.0 and,
PAR(1)×PAR(2)×(1.0–PAR(2)) ≤ 10^6 and
PAR(1) < 1/eps where eps = the machine precision, see X02AJF.
if DIST = 'E' or 'e' PAR(1) > 0.0
if DIST = 'P' or 'p' PAR(1) > 0.0 and PAR(1) ≤ 10^6.

5: ESTIMA – CHARACTER*1. *Input*

On entry: ESTIMA must specify whether values of the parameters of the null distribution are known or are to be estimated from the data:

If ESTIMA = 'S' or 's', values of the parameters will be supplied in the array PAR described above.

If ESTIMA = 'E' or 'e', parameters are to be estimated from the data except when the null distribution requested is the binomial distribution in which case the first parameter, m, must be supplied in PAR(1) and only the second parameter, p is estimated from the data.

Constraint: ESTIMA = 'S', 's', 'E' or 'e'

6: NTYPE – INTEGER. *Input*

On entry: the test statistic to be calculated, i.e. the choice of alternative hypothesis.

NTYPE = 1 : Computes D_n, to test H_0 against H_1,

NTYPE = 2 : Computes D_n^+ to test H_0 against H_2,

NTYPE = 3 : Computes D_n^-, to test H_0 against H_3.

Constraint: NTYPE = 1, 2 or 3.

7: D – ***real.*** *Output*

On exit: the Kolmogorov-Smirnov test statistic (D_n, D_n^+ or D_n^- according to the value of NTYPE).

8: Z – ***real.*** *Output*

On exit: a standardized value, Z, of the test statistic, D, without any continuity correction applied.

9: P – ***real.*** *Output*

On exit: the probability, p, associated with the observed value of D where D may be D_n,D_n^+ or D_n^- depending on the value of NTYPE, (see Section 3).

10: SX(N) – ***real*** array. *Output*

On exit: the sample observations, $x_1,x_2,...,x_n$, sorted in ascending order,

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N < 3.

IFAIL = 2

On entry, an invalid code for DIST has been specified.

IFAIL = 3

On entry, NTYPE ≠ 1, 2 or 3.

IFAIL = 4

On entry, ESTIMA ≠ 'S' or 's' or 'E' or 'e'.

IFAIL = 5

On entry, the parameters supplied for the specified null distribution are out of range (see Section 5). Apart from a check on the first parameter for the binomial distribution (DIST = 'BI', 'Bi', 'bI' or 'bi') this error will only occur if ESTIMA = 'S' or 's'.

IFAIL = 6

The data supplied in X could not arise from the chosen null distribution, as specified by the parameters DIST and PAR. For further details see Section 8.

IFAIL = 7

The whole sample is constant i.e. the variance is zero. This error may only occur if (DIST = 'U', 'u', 'N', 'n', 'G', 'g', 'BE', 'Be', 'bE' or 'be') and ESTIMA = 'E' or 'e'.

IFAIL = 8

The variance of the binomial distribution (DIST = 'BI', 'Bi', 'bI' or 'i') is too large. That is, $mp(1-p) > 1000000$.

IFAIL = 9

When DIST = 'G' or 'g', in the computation of the incomplete gamma function by S14BAF the convergence of the Taylor series or Legendre continued fraction fails within 600 iterations, this is an unlikely error exit.

7. Accuracy

The approximation for p, given when $n > 100$, has a relative error of at most 2.5% for most cases. The two-sided probability is approximated by doubling the one-sided probability. This is only good for small p, i.e $p < .10$ but very poor for large p. The error is always on the conservative side, that is the tail probability p, is over estimated.

8. Further Comments

The time taken by the routine increases with n until $n > 100$ at which point it drops and then increases slowly with n. The time may also depend on the choice of null distribution and on whether or not the parameters are to be estimated.

The data supplied in the parameter X must be consistent with the chosen null distribution as follows:

When DIST = 'U' or 'u',	then PAR(1) $\le x_i \le$ PAR(2) for $i = 1,2,...,n$.
When DIST = 'N' or 'n',	then there is no constraint on the x_i's.
When DIST = 'G' or 'g',	then $x_i \ge 0.0$ for $i = 1,2,...,n$.
When DIST = 'BE', 'Be', 'bE' or 'be',	then $0.0 \le x_i \le 1.0$ for $i = 1,2,...,n$.
When DIST = 'B', 'Bi', 'bI' or 'bi',	then $0.0 \le x_i \le$ PAR(1) for $i = 1,2,...,n$.
When DIST = 'E' or 'e'	then $x_i \ge 0.0$ for $i = 1,2,...,n$.
When DIST = 'P' or 'p'	then $x_i \ge 0.0$ for $i = 1,2,...,n$.

9. Example

The following example program reads in a set of data consisting of 30 observations. The Kolmogorov-Smirnov test is then applied twice, firstly to test whether the sample is taken from a uniform distribution, $U(0,2)$ and secondly to test whether the sample is taken from a Normal distribution where the mean and variance are estimated from the data. In both cases we are testing against H_1 – that is we are doing a two-tailed test. The values of D, Z and P are printed for each case.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08CBF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MAXP
      PARAMETER        (NMAX=30,MAXP=2)
*     .. Local Arrays ..
      real             PAR(MAXP), SX(NMAX), X(NMAX)
```

```
*     .. Local Scalars ..
      real              D, P, Z
      INTEGER           I, IFAIL, N, NP, NTYPE
*     .. External Subroutines ..
      EXTERNAL          G08CBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08CBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      WRITE (NOUT,*)
      IF (N.LE.NMAX) THEN
         READ (NIN,*) (X(I),I=1,N)
         READ (NIN,*) NP, (PAR(I),I=1,NP), NTYPE
         IFAIL = 0
*
         CALL G08CBF(N,X,'Uniform',PAR,'Supplied',NTYPE,D,Z,P,SX,IFAIL)
*
         WRITE (NOUT,*) 'Test against uniform distribution on (0,2)'
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'Test statistic D = ', D
         WRITE (NOUT,99999) 'Z statistic      = ', Z
         WRITE (NOUT,99999) 'Tail probability = ', P
         WRITE (NOUT,*)
*
         READ (NIN,*) NP, (PAR(I),I=1,NP), NTYPE
         IFAIL = 0
*
         CALL G08CBF(N,X,'Normal',PAR,'Estimate',NTYPE,D,Z,P,SX,IFAIL)
*
         WRITE (NOUT,*)
     +'Test against Normal distribution with parameters estimated from t
     +he data'
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'Mean = ', PAR(1), '  and variance = ',
     +     PAR(2)
         WRITE (NOUT,99999) 'Test statistic D = ', D
         WRITE (NOUT,99999) 'Z statistic      = ', Z
         WRITE (NOUT,99999) 'Tail probability = ', P
      ELSE
         WRITE (NOUT,99997) 'N is out of range: N = ', N
      END IF
      STOP
*
99999 FORMAT (1X,A,F8.4)
99998 FORMAT (1X,A,F6.4,A,F6.4)
99997 FORMAT (1X,A,I7)
      END
```

9.2. Program Data

```
G08CBF Example Program Data
 30
 0.01 0.30 0.20 0.90 1.20 0.09 1.30 0.18 0.90 0.48
 1.98 0.03 0.50 0.07 0.70 0.60 0.95 1.00 0.31 1.45
 1.04 1.25 0.15 0.75 0.85 0.22 1.56 0.81 0.57 0.55
 2  0.0  2.0  1
 2  0.0  1.0  1
```

9.3. Program Results

```
G08CBF Example Program Results

Test against uniform distribution on (0,2)

Test statistic D =   0.2800
Z statistic      =   1.5336
Tail probability =   0.0143

Test against Normal distribution with parameters estimated from the data

Mean = 0.6967  and variance = 0.2564
Test statistic D =   0.1108
Z statistic      =   0.6068
Tail probability =   0.8925
```

G08CCF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08CCF performs the one sample Kolmogorov-Smirnov distribution test, using a user specified distribution.

2. Specification

```
SUBROUTINE G08CCF (N, X, CDF, NTYPE, D, Z, P, SX, IFAIL)
INTEGER          N, NTYPE, IFAIL
real             X(N), CDF, D, Z, P, SX(N)
EXTERNAL         CDF
```

3. Description

The data consist of a single sample of n observations, denoted by $x_1,x_2,\ldots,x_n$.

Let $S_n(x_{(i)})$ and $F_0(x_{(i)})$ represent the sample cumulative distribution function and the theoretical (null) cumulative distribution function respectively at the point $x_{(i)}$ where $x_{(i)}$ is the ith smallest sample observation.

The Kolmogorov-Smirnov test provides a test of the null hypothesis H_0 : the data are a random sample of observations from a theoretical distribution specified by the user (in the function CDF) against one of the following alternative hypotheses:

(i) H_1 : the data cannot be considered to be a random sample from the specified null distribution.

(ii) H_2 : the data arise from a distribution which dominates the specified null distribution. In practical terms, this would be demonstrated if the values of the sample cumulative distribution function $S_n(x)$ tended to exceed the corresponding values of the theoretical cumulative distribution function $F_0(x)$.

(iii) H_3 : the data arise from a distribution which is dominated by the specified null distribution. In practical terms, this would be demonstrated if the values of the theoretical cumulative distribution function $F_0(x)$ tended to exceed the corresponding values of the sample cumulative distribution function $S_n(x)$.

One of the following test statistics is computed depending on the particular alternative hypothesis specified (see the description of the parameter NTYPE in Section 5).

For the alternative hypothesis H_1.

D_n – the largest absolute deviation between the sample cumulative distribution function and the theoretical cumulative distribution function. Formally $D_n = \max\{D_n^+,D_n^-\}$.

For the alternative hypothesis H_2.

D_n^+ – the largest positive deviation between the sample cumulative distribution function and the theoretical cumulative distribution function. Formally $D_n^+ = \max\{S_n(x_{(i)})-F_0(x_{(i)}),0\}$

For the alternative hypothesis H_3.

D_n^- – the largest positive deviation between the theoretical cumulative distribution function and the sample cumulative distribution function. Formally $D_n^- = \max\{F_0(x_{(i)})-S_n(x_{(i-1)}),0\}$. This is only true for continuous distributions. See Section 8 for comments on discrete distributions.

The standardized statistic, $Z = D\times\sqrt{n}$, is also computed where D may be D_n,D_n^+ or D_n^- depending on the choice of the alternative hypothesis. This is the standardized value of D with no continuity correction applied and the distribution of Z converges asymptotically to a limiting distribution, first derived by Kolmogorov [4], and then tabulated by Smirnov [6]. The asymptotic distributions for the one-sided statistics were obtained by Smirnov [5].

The probability, under the null hypothesis, of obtaining a value of the test statistic as extreme as that observed, is computed. If $n \leq 100$ an exact method given by Conover [1], is used. Note that the method used is only exact for continuous theoretical distributions and does not include Conover's modification for discrete distributions. This method computes the one-sided probabilities. The two-sided probabilities are estimated by doubling the one-sided probability. This is a good estimate for small p, that is $p \leq 0.10$, but it becomes very poor for larger p. If $n > 100$ then p is computed using the Kolmogorov-Smirnov limiting distributions, see Feller [2], Kendall and Stuart [3], Kolmogorov [4], Smirnov [5] and [6].

4. References

[1] CONOVER, W.J.
Practical Nonparametric Statistics, Ch. 6.
John Wiley & Sons, New York, 1980.

[2] FELLER W.
On the Kolmogorov-Smirnov Limit Theorems for Empirical Distributions.
Annals of Math. Stat., 19, pp. 179-181, 1948.

[3] KENDALL, M.G. and STUART, A.
The Advanced Theory of Statistics, Vol. 2, Ch. 30.
Griffin, London, 1973.

[4] KOLMOGOROV, A.N.
Sulla determinazione empirica di una legge di distribuzione.
Giornale dell' Istituto Italiano degli Attuari, 4, pp. 83-91, 1933.

[5] SMIRNOV, N.
Estimate of deviation between empirical distribution functions in two independent samples.
Bulletin Moscow University, 2(2), pp. 3-16, 1933.

[6] SMIRNOV, N.
Table for estimating the goodness of fit of empirical distributions.
Annals of Math. Stat., 19, pp. 279-281, 1948.

[7] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences, Ch. 4.
McGraw-Hill, 1956.

5. Parameters

1: N – INTEGER. *Input*

On entry: the number of observations in the sample, n.

Constraint: N ≥ 1.

2: X(N) – ***real*** array. *Input*

On entry: the sample observations $x_1,x_2,...,x_n$.

3: CDF – ***real*** FUNCTION, supplied by the user. *External Procedure*

CDF must return the value of the theoretical (null) cumulative distribution function for a given value of its argument.

Its specification is:

```
real FUNCTION CDF(X)
real          X
```

1: X – ***real.*** *Input*

On entry: the argument for which CDF must be evaluated.

CDF must be declared as EXTERNAL in the (sub)program from which G08CCF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

Constraint: CDF must always return a value in the range [0.0,1.0] and CDF must always satisfy the condition that CDF(x_1) $\le$ CDF(x_2) for any $x_1 \le x_2$.

4: NTYPE – INTEGER. *Input*

On entry: the statistic to be calculated, i.e. the choice of alternative hypothesis.

NTYPE = 1 : Computes D_n, to test H_0 against H_1,

NTYPE = 2 : Computes D_n^+, to test H_0 against H_2,

NTYPE = 3 : Computes D_n^-, to test H_0 against H_3.

Constraint: NTYPE = 1, 2 or 3.

5: D – ***real***. *Output*

On exit: the Kolmogorov-Smirnov test statistic (D_n, D_n^+ or D_n^- according to the value of NTYPE).

6: Z – ***real***. *Output*

On exit: a standardized value, Z, of the test statistic, D, without the continuity correction applied.

7: P – ***real***. *Output*

On exit: the probability, p, associated with the observed value of D where D may D_n,D_n^+ or D_n^- depending on the value of NTYPE, (see Section 3).

8: SX(N) – ***real*** array. *Output*

On exit: the sample observations, x_1,x_2,...,x_n, sorted in ascending order.

9: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N < 1

IFAIL = 2

On entry, NTYPE ≠ 1, 2 or 3.

IFAIL = 3

The supplied theoretical cumulative distribution function returns a value less than 0.0 or greater than 1.0, thereby violating the definition of the cumulative distribution function.

IFAIL = 4

The supplied theoretical cumulative distribution function is not a non-decreasing function thereby violating the definition of a cumulative distribution function, that is $F_0(x) > F_0(y)$ for some $x < y$.

7. Accuracy

For most cases the approximation for p given when $n > 100$ has a relative error of less than 0.01. The two-sided probability is approximated by doubling the one-sided probability. This is only good for small p, that is $p < 0.10$, but very poor for large p. The error is always on the conservative side.

8. Further Comments

The time taken by the routine increases with n until $n > 100$ at which point it drops and then increases slowly.

For a discrete theoretical cumulative distribution function $F_0(x)$, $D_n^- = \max\{F_0(x_{(i)})-S_n(x_{(i)}),0\}$. Thus if the user wishes to provide a discrete distribution function the following adjustment needs to be made,

For D_n^+ – return $F(x)$ as x as usual

For D_n^- – return $F(x-d)$ at x where d is the discrete jump in the distribution. For example $d = 1$ for the Poisson or Binomial distributions.

9. Example

The following example performs the one sample Kolmogorov-Smirnov test to test whether a sample of 30 observations arise firstly from a uniform distribution $U(0,1)$ or secondly from a Normal distribution with mean 0.75 and standard deviation 0.5. The two-sided test statistic, D_n, the standardized test statistic, Z, and the upper tail probability, p, are computed and then printed for each test.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08CCF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX
      PARAMETER        (NMAX=30)
*     .. Local Arrays ..
      real             SX(NMAX), X(NMAX)
*     .. Local Scalars ..
      real             D, P, Z
      INTEGER          I, IFAIL, N, NTYPE
*     .. External Functions ..
      real             CDF1, CDF2
      EXTERNAL         CDF1, CDF2
*     .. External Subroutines ..
      EXTERNAL         G08CCF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08CCF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      WRITE (NOUT,*)
      IF (N.LE.NMAX) THEN
         READ (NIN,*) (X(I),I=1,N)
         READ (NIN,*) NTYPE
         IFAIL = 0
*
```

```
            CALL G08CCF(N,X,CDF1,NTYPE,D,Z,P,SX,IFAIL)
*
            WRITE (NOUT,*) 'Test against uniform distribution on (0,2)'
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Test statistic D = ', D
            WRITE (NOUT,99999) 'Z statistic      = ', Z
            WRITE (NOUT,99999) 'Tail probability = ', P
*
            CALL G08CCF(N,X,CDF2,NTYPE,D,Z,P,SX,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +        'Test against normal distribution with mean = 0.75'
            WRITE (NOUT,*) 'and standard deviation = 0.5.'
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Test statistic D = ', D
            WRITE (NOUT,99999) 'Z statistic      = ', Z
            WRITE (NOUT,99999) 'Tail probability = ', P
         ELSE
            WRITE (NOUT,99998) 'N is out of range: N = ', N
         END IF
         STOP
*
99999 FORMAT (1X,A,F8.4)
99998 FORMAT (1X,A,I7)
      END
*
      real FUNCTION CDF1(X)
*     .. Parameters ..
      real              A, B
      PARAMETER         (A=0.0e0,B=2.0e0)
*     .. Scalar Arguments ..
      real              X
*     .. Executable Statements ..
      IF (X.LT.A) THEN
         CDF1 = 0.0e0
      ELSE IF (X.GT.B) THEN
         CDF1 = 1.0e0
      ELSE
         CDF1 = (X-A)/(B-A)
      END IF
      RETURN
      END
*
      real FUNCTION CDF2(X)
*     .. Parameters ..
      real              XMEAN, STD
      PARAMETER         (XMEAN=0.75e0,STD=0.5e0)
*     .. Scalar Arguments ..
      real              X
*     .. Local Scalars ..
      real              Z
      INTEGER           IFAIL
*     .. External Functions ..
      real              S15ABF
      EXTERNAL          S15ABF
*     .. Executable Statements ..
      Z = (X-XMEAN)/STD
      CDF2 = S15ABF(Z,IFAIL)
      RETURN
      END
```

9.2. Program Data

```
G08CCF Example Program Data
 30
 0.01 0.30 0.20 0.90 1.20 0.09 1.30 0.18 0.90 0.48
 1.98 0.03 0.50 0.07 0.70 0.60 0.95 1.00 0.31 1.45
 1.04 1.25 0.15 0.75 0.85 0.22 1.56 0.81 0.57 0.55
 1
```

9.3. Program Results

```
 G08CCF Example Program Results

 Test against uniform distribution on (0,2)

 Test statistic D =   0.2800
 Z statistic      =   1.5336
 Tail probability =   0.0143

 Test against normal distribution with mean = 0.75
 and standard deviation = 0.5.

 Test statistic D =   0.1439
 Z statistic      =   0.7882
 Tail probability =   0.5262
```

G08CDF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08CDF performs the two sample Kolmogorov-Smirnov distribution test.

2. Specification

```
SUBROUTINE G08CDF (N1, X, N2, Y, NTYPE, D, Z, P, SX, SY, IFAIL)
INTEGER        N1, N2, NTYPE, IFAIL
real           X(N1), Y(N2), D, Z, P, SX(N1), SY(N2)
```

3. Description

The data consist of two independent samples, one of size n_1, denoted by $x_1,x_2,...,x_{n_1}$, and the other of size n_2 denoted by $y_1,y_2,...,y_{n_2}$. Let $F(x)$ and $G(x)$ represent their respective, unknown, distribution functions. Also let $S_1(x)$ and $S_2(x)$ denote the values of the sample cumulative distribution functions at the point x for the two samples respectively.

The Kolmogorov-Smirnov test provides a test of the null hypothesis $H_0 : F(x) = G(x)$ against one of the following alternative hypotheses:

(i) $H_1 : F(x) \neq G(x)$.

(ii) $H_2 : F(x) > G(x)$. This alternative hypothesis is sometimes stated as, 'The x's tend to be smaller than the y's', i.e. it would be demonstrated in practical terms if the values of $S_1(x)$ tended to exceed the corresponding values of $S_2(x)$.

(iii) $H_3 : F(x) < G(x)$. This alternative hypothesis is sometimes stated as, 'The x's tend to be larger than the y's', i.e. it would be demonstrated in practical terms if the values of $S_2(x)$ tended to exceed the corresponding values of $S_1(x)$.

One of the following test statistics is computed depending on the particular alternative null hypothesis specified (see the description of the parameter NTYPE in Section 5).

For the alternative hypothesis H_1.

D_{n_1,n_2} – the largest absolute deviation between the two sample cumulative distribution functions.

For the alternative hypothesis H_2.

$D^+_{n_1,n_2}$ – the largest positive deviation between the sample cumulative distribution function of the first sample, $S_1(x)$, and the sample cumulative distribution function of the second sample, $S_2(x)$. Formally $D^+_{n_1,n_2} = \max\{S_1(x) - S_2(x),0\}$

For the alternative hypothesis H_3.

$D^-_{n_1,n_2}$ – the largest positive deviation between the sample cumulative distribution function of the second sample, $S_2(x)$, and the sample cumulative distribution function of the first sample, $S_1(x)$. Formally $D^-_{n_1,n_2} = \max\{S_2(x) - S_1(x),0\}$

G08CDF also returns the standardized statistic $Z = \sqrt{\frac{n_1+n_2}{n_1n_2}}\times D$ where D may be D_{n_1,n_2}, $D^+_{n_1,n_2}$ or $D^-_{n_1,n_2}$ depending on the choice of the alternative hypothesis. The distribution of this statistic converges asymptotically to a distribution given by Smirnov as n_1 and n_2 increase, see Feller [2], Kendall *et al.* [3], Kim *et al.* [4}, Smirnov [5] or Smirnov [6].

The probability, under the null hypothesis, of obtaining a value of the test statistic as extreme as that observed, is computed. If $\max(n_1,n_2) \leq 2500$ and $n_1n_2 \leq 10000$ then an exact method given by Kim and Jenrich see [4] is used. Otherwise p is computed using the approximations suggested by Kim and Jenrich [4]. Note that the method used is only exact for continuous theoretical distributions. This method computes the two-sided probability. The one-sided probabilities are estimated by halving the two-sided probability. This is a good estimate for small p, that is $p \leq 0.10$, but it becomes very poor for larger p.

4. References

[1] CONOVER, W.J.
Practical Nonparametric Statistics, Ch. 6.
John Wiley & Sons, New York, 1980.

[2] FELLER W.
On the Kolmogorov-Smirnov Limit Theorems for Empirical Distributions.
Annals of Math. Stat., 19, pp. 179-181, 1948.

[3] KENDALL, M.G. and STUART, A.
The Advanced Theory of Statistics, Vol. 2, Ch. 30.
Griffin, London, 1973.

[4] KIM, P.J. and JENRICH, R.I.
Tables of exact sampling distribution of the two sample Kolmogorov-Smirnov criterion $D_{mn}(m<n)$.
In: 'Selected Tables in Mathematical Statistics', Vol. 1, H.L. Harter and D.B. Owen (eds). American Mathematical Society, Providence, Rhode Island, pp. 80-129, 1973.

[5] SMIRNOV, N.
Estimate of deviation between empirical distribution functions in two independent samples.
Bulletin Moscow University, 2(2), pp. 3-16, 1939.

[6] SMIRNOV, N.
Table for estimating the goodness of fit of empirical distributions.
Annals of Math. Stat., 19, pp. 279-281, 1948.

[7] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences, Ch. 4.
McGraw-Hill, 1956.

5. Parameters

1: N1 – INTEGER. *Input*

On entry: the number of observations in the first sample, n_1.

Constraint: N1 $\geq$ 1.

2: X(N1) – ***real*** array. *Input*

On entry: the observations from the first sample, $x_1,x_2,...,x_{n_1}$.

3: N2 – INTEGER. *Input*

On entry: the number of observations in the second sample, n_2.

Constraint: N2 $\geq$ 1.

4: Y(N2) – ***real*** array. *Input*

On entry: the observations from the second sample, $y_1,y_2,...,y_{n_2}$.

5: NTYPE – INTEGER. *Input*

On entry: the statistic to be computed, i.e the choice of alternative hypothesis.

NTYPE = 1 : Computes $D_{n_1n_2}$, to test against H_1.

NTYPE = 2 : Computes $D^{+}_{n_1n_2}$, to test against H_2.

NTYPE = 3 : Computes $D^{-}_{n_1n_2}$, to test against H_3.

Constraint: NTYPE = 1, 2 or 3.

6: D – ***real.*** *Output*

On exit: the Kolmogorov-Smirnov test statistic ($D_{n_1n_2}$, $D^{+}_{n_1n_2}$ or $D^{-}_{n_1n_2}$ according to the value of NTYPE).

7: Z – ***real.*** *Output*

On exit: a standardized value Z of the test statistic, D, without any correction for continuity.

8: P – ***real.*** *Output*

On exit: the tail probability associated with the observed value of D, where D may be D_{n_1,n_2},D_{n_1,n_2} or D_{n_1,n_2} depending on the value of NTYPE, (see Section 3).

9: SX(N1) – ***real*** array. *Output*

On exit: the observations from the first sample sorted in ascending order.

10: SY(N2) – ***real*** array. *Output*

On exit: the observations from the second sample sorted in ascending order.

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N1 < 1,
or N2 < 1.

IFAIL = 2

On entry, NTYPE ≠ 1, 2 or 3.

IFAIL = 3

The iterative procedure used in the approximation of the probability for large n_1 and n_2 did not converge. For the two-sided test, $p = 1$ is returned. For the one-sided test $p = 0.5$ is returned.

7. Accuracy

The large sample distributions used as approximations to the exact distribution should have a relative error of less than 5% for most cases.

8. Further Comments

The time taken by the routine increases with n_1 and n_2, until $n_1n_2 > 10000$ or $\max(n_1,n_2) \geq 2500$. At this point one of the approximations is used and the time decreases significantly. The time then increases again modestly with n_1 and n_2.

9. Example

The following example computes the two-sided Kolmogorov-Smirnov test statistic for two independent samples of size 100 and 50 respectively. The first sample is from a uniform distribution $U(0,2)$. The second sample is from a uniform distribution $U(0.25,2.25)$. The test statistic, D_{n_1,n_2}, the standardized test statistic, Z, and the tail probability, p, are computed and printed.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08CDF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX
      PARAMETER        (NMAX=100,MMAX=50)
*     .. Local Arrays ..
      real             SX(NMAX), SY(MMAX), X(NMAX), Y(MMAX)
*     .. Local Scalars ..
      real             D, P, Z
      INTEGER          IFAIL, M, N, NTYPE
*     .. External Subroutines ..
      EXTERNAL         G05CBF, G05FAF, G08CDF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08CDF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, M
      WRITE (NOUT,*)
      IF (N.LE.NMAX .AND. M.LE.MMAX) THEN
         CALL G05CBF(0)
         CALL G05FAF(0.0e0,2.0e0,N,X)
         CALL G05FAF(0.25e0,2.25e0,M,Y)
         READ (NIN,*) NTYPE
         IFAIL = -1
*
         CALL G08CDF(N,X,M,Y,NTYPE,D,Z,P,SX,SY,IFAIL)
*
         IF (IFAIL.NE.0) WRITE (NOUT,99999) '** IFAIL = ', IFAIL
         WRITE (NOUT,99998) 'Test statistic D = ', D
         WRITE (NOUT,99998) 'Z statistic      = ', Z
         WRITE (NOUT,99998) 'Tail probability = ', P
      ELSE
         WRITE (NOUT,99997) 'N or M is out of range: N = ', N,
     +     ' and M = ', M
      END IF
      STOP
*
99999 FORMAT (1X,A,I2)
99998 FORMAT (1X,A,F8.4)
99997 FORMAT (1X,A,I7,A,I7)
      END
```

9.2. Program Data

```
G08CDF Example Program Data
 100 50
 1
```

9.3. Program Results

```
G08CDF Example Program Results

Test statistic D =   0.3600
Z statistic      =   0.0624
Tail probability =   0.0003
```

G08CGF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08CGF computes the test statistic for the χ^2 goodness of fit test for data with a chosen number of class intervals.

2. Specification

```
      SUBROUTINE G08CGF (NCLASS, IFREQ, CINT, DIST, PAR, NPEST, PROB,
     1                   CHISQ, P, NDF, EVAL, CHISQI, IFAIL)
      INTEGER          NCLASS, IFREQ(NCLASS), NPEST, NDF, IFAIL
      real             CINT(NCLASS-1), PAR(2), PROB(NCLASS), CHISQ, P,
     1                 EVAL(NCLASS), CHISQI(NCLASS)
      CHARACTER*1      DIST
```

3. Description

The χ^2 goodness of fit test performed by G08CGF is used to test the null hypothesis that a random sample arises from a specified distribution against the alternative hypothesis that the sample does not arise from the specified distribution.

Given a sample of size n, denoted by $x_1,x_2,\ldots,x_n$, drawn from a random variable X, and that the data have been grouped into k classes,

$$x \le c_1,$$
$$c_{i-1} < x \le c_i, \qquad i = 2,3,\ldots k-1,$$
$$x > c_{k-1},$$

then the χ^2 goodness of fit test statistic is defined by;

$$X^2 = \sum_{i=1}^{k} \frac{(O_i-E_i)^2}{E_i}$$

where O_i is the observed frequency of the ith class, and E_i is the expected frequency of the ith class.

The expected frequencies are computed as

$$E_i = p_i \times n,$$

where p_i is the probability that X lies in the ith class, that is

$$p_1 = P(X \le c_1),$$
$$p_i = P(c_{i-1} < X \le c_i), \qquad i = 2,3,\ldots k-1,$$
$$p_k = P(X > c_{k-1}).$$

These probabilities are either taken from a common probability distribution or are supplied by the user. The available probability distributions within this routine are:

Normal distribution with mean μ, variance σ^2;

uniform distribution on the interval $[a,b]$;

exponential distribution with $pdf = \lambda e^{-\lambda x}$;

χ^2 distribution with f degrees of freedom; and

gamma distribution with $pdf = \dfrac{x^{\alpha-1}e^{-x/\beta}}{\Gamma(\alpha)\beta^\alpha}$.

The user must supply the frequencies and classes. Given a set of data and classes the frequencies may be calculated using G01AEF.

G08CGF returns the χ^2 test statistic, X^2, together with it's degrees of freedom and the upper tail probability from the χ^2 distribution associated with the test statistic. Note that the use of the χ^2 distribution as an approximation to the distribution of the test statistic improves as the expected values in each class increase.

4. References

[1] CONOVER, W.J.
Practical Nonparametric Statistics, Ch. 6.
John Wiley & Sons, New York, 1980.

[2] KENDALL, M.G. and STUART, A.
The Advanced Theory of Statistics, Vol. 2, Ch. 30.
Griffin, London, 1973.

[3] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences, Ch. 4.
McGraw-Hill, 1956.

5. Parameters

1: NCLASS – INTEGER. *Input*

On entry: the number of classes, k, into which the data is divided.

Constraint: NCLASS $\geq$ 2.

2: IFREQ(NCLASS) – INTEGER array. *Input*

On entry: IFREQ(i) must specify the frequency of the ith class, O_i, for $i = 1,2,...,k$.

Constraint: IFREQ(i) ≥ 0, for $i = 1,2,....,k$.

3: CINT(NCLASS–1) – ***real*** array. *Input*

On entry: CINT(i) must specify the upper boundary value for the ith class, for $i = 1,2,...,k-1$.

Constraint: CINT(1) < CINT(2) < ... < CINT(NCLASS–1). For the exponential, gamma and χ^2 distributions CINT(1) $\geq$ 0.0.

4: DIST – CHARACTER*1. *Input*

On entry: indicates for which distribution the test is to be carried out;

If DIST = 'N' or 'n', the Normal distribution is used.
If DIST = 'U' or 'u', the uniform distribution is used.
If DIST = 'E' or 'e', the exponential distribution is used.
If DIST = 'C' or 'c', the χ^2 distribution is used.
If DIST = 'G' or 'g', the gamma distribution is used.
If DIST = 'A' or 'a', the user must supply the class probabilities in the array PROB.

Constraint: DIST = 'N', 'n', 'U', 'u', 'E', 'e', 'C', 'c', 'G', 'g', 'A' or 'a'.

5: PAR(2) – ***real*** array. *Input*

On entry: PAR must contain the parameters of the distribution which is being tested. If the user supplies the probabilities (that is, DIST = 'A' or 'a') the array PAR is not referenced.

If a Normal distribution is used then PAR(1) and PAR(2) must contain the mean, μ, and the variance, σ^2, respectively.

If a uniform distribution is used then PAR(1) and PAR(2) must contain the boundaries a and b respectively.

If an exponential distribution is used then PAR(1) must contain the parameter λ. PAR(2) is not used.

If a χ^2 distribution is used then PAR(1) must contain the number of degrees of freedom. PAR(2) is not used.

If a gamma distribution is used PAR(1) and PAR(2) must contain the parameters α and β respectively.

Constraints: if DIST = 'N' or 'n', PAR(2) > 0.0,
if DIST = 'U' or 'u', PAR(1) < PAR(2), PAR(1) ≤ CINT(1),
PAR(2) ≥ CINT(NCLASS−1),
if DIST = 'E' or 'e', PAR(1) > 0.0,
if DIST = 'C' or 'c', PAR(1) > 0.0,
if DIST = 'G' or 'g', PAR(1), PAR(2) > 0.0.

6: NPEST – INTEGER. *Input*

On entry: the number of estimated parameters of the distribution.

Constraint: 0 ≤ NPEST < NCLASS − 1.

7: PROB(NCLASS) – ***real*** array. *Input*

On entry: if the user is supplying the probability distribution (that is, DIST = 'A' or 'a') then PROB(i) must contain the probability that X lies in the ith class.

If DIST ≠ 'A' or 'a', PROB is not referenced.

Constraints: if DIST = 'A' or 'a' then PROB(i) > 0.0, for $i = 1,2,...,k$ and

$$\sum_{i=1}^{k} \mathrm{PROB}(i) = 1.0.$$

8: CHISQ – ***real.*** *Output*

On exit: the test statistic, X^2, for the χ^2 goodness of fit test.

9: P – ***real.*** *Output*

On exit: the upper tail probability from the χ^2 distribution associated with the test statistic, X^2, and the number of degrees of freedom.

10: NDF – INTEGER. *Output*

On exit: contains (NCLASS−1−NPEST), the degrees of freedom associated with the test.

11: EVAL(NCLASS) – ***real*** array. *Output*

On exit: EVAL(i) contains the expected frequency for the ith class, E_i, for $i = 1,2,...,k$.

12: CHISQI(NCLASS) – ***real*** array. *Output*

On exit: CHISQI(i) contains the contribution from the ith class to the test statistic, that is $(O_i-E_i)^2/E_i$, for $i = 1,2,...,k$.

13: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to −1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, NCLASS < 2.

IFAIL = 2

On entry, DIST is invalid.

IFAIL = 3

On entry, NPEST < 0,
or NPEST ≥ NCLASS − 1.

IFAIL = 4

On entry, IFREQ(i) < 0.0 for some i, for $i = 1,2,...k$.

IFAIL = 5

On entry, the elements of CINT are not in ascending order. That is CINT(i) ≤ CINT(i−1) for some i, for $i = 2,3,...k-1$.

IFAIL = 6

On entry, DIST = 'E', 'e', 'C', 'c', 'G' or 'g' and CINT(1) < 0.0. No negative class boundary values are valid for the exponential, gamma or χ^2 distributions.

IFAIL = 7

On entry, the values provided in PAR are invalid.

IFAIL = 8

On entry, with DIST = 'A' or 'a', PROB(i) ≤ 0.0 for some i, for $i = 1,2,...k$,
or $\sum_{i=1}^{k}$ PROB(i) ≠ 1.0.

IFAIL = 9

An expected frequency is equal to zero when the observed frequency was not.

IFAIL = 10

This is a warning that expected values for certain classes are less than 1.0. This implies that we one cannot be confident that the χ^2 distribution is a good approximation to the distribution of the test statistic.

IFAIL = 11

The solution obtained when calculating the probability for a certain class for the gamma or χ^2 distribution did not converge in 600 iterations. The solution may be an adequate approximation.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is dependent both on the distribution chosen and on the number of classes, k.

9. Example

The following program applies the χ^2 goodness of fit test to test whether there is evidence to suggest that a sample of 100 observations generated by G05DAF do not arise from a uniform distribution $U(0,1)$. The class intervals are calculated such that the interval $(0,1)$ is divided into 5 equal classes. The frequencies for each class are calculated using G01AEF.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08CGF Example Program Text
*     Mark 15 Revised.  NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT, NMAX, NCLMAX
      PARAMETER        (NIN=5,NOUT=6,NMAX=100,NCLMAX=10)
*     .. Local Scalars ..
      real             CHISQ, P, XMAX, XMIN
      INTEGER          I, ICLASS, IFAIL, NPEST, NCLASS, N, NDF
      CHARACTER*1      CDIST
*     .. Local Arrays ..
      real             CHISQI(NCLMAX), CINT(NCLMAX), EVAL(NCLMAX),
     +                 PAR(2), PROB(NCLMAX), X(NMAX)
      INTEGER          IFREQ(NCLMAX)
*     .. External Subroutines ..
      EXTERNAL         G01AEF, G05CBF, G05FAF, G08CGF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08CGF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NCLASS, CDIST
      READ (NIN,*) (PAR(I),I=1,2)
      NPEST = 0
*
*     Generate random numbers from a uniform distribution
      CALL G05CBF(0)
*
      CALL G05FAF(PAR(1),PAR(2),N,X)
      ICLASS = 0
*
*     Determine suitable intervals
      IF (CDIST.EQ.'U' .OR. CDIST.EQ.'u') THEN
         ICLASS = 1
         CINT(1) = PAR(1) + (PAR(2)-PAR(1))/NCLASS
         DO 20 I = 2, NCLASS - 1
            CINT(I) = CINT(I-1) + (PAR(2)-PAR(1))/NCLASS
   20    CONTINUE
      END IF
      IFAIL = 0
*
      CALL G01AEF(N,NCLASS,X,ICLASS,CINT,IFREQ,XMIN,XMAX,IFAIL)
*
      IFAIL = 0
*
      CALL G08CGF(NCLASS,IFREQ,CINT,CDIST,PAR,NPEST,PROB,CHISQ,P,NDF,
     +            EVAL,CHISQI,IFAIL)
*
      IF (IFAIL.NE.0) WRITE (NOUT,99999) '** IFAIL = ', IFAIL
      WRITE (NOUT,*)
      WRITE (NOUT,99998) 'Chi-squared test statistic    = ', CHISQ
      WRITE (NOUT,99997) 'Degrees of freedom.           = ', NDF
      WRITE (NOUT,99998) 'Significance level            = ', P
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'The contributions to the test statistic are :-'
      DO 40 I = 1, NCLASS
         WRITE (NOUT,99996) CHISQI(I)
   40 CONTINUE
      STOP
```

```
*
99999 FORMAT (1X,A,I2)
99998 FORMAT (1X,A,F10.4)
99997 FORMAT (1X,A,I5)
99996 FORMAT (1X,F10.4)
      END
```

9.2. Program Data

```
G08CGF Example Program Data.
100 5 'U'      :N  NCLASS  CDIST
0.0 1.0        :PAR(1) PAR(2)
```

9.3. Program Results

```
 G08CGF Example Program Results

 Chi-squared test statistic    =     3.3000
 Degrees of freedom.           =     4
 Significance level            =     0.5089

 The contributions to the test statistic are :-
     1.8000
     0.8000
     0.2000
     0.0500
     0.4500
```

G08DAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08DAF calculates Kendall's coefficient of concordance on k independent rankings of n objects or individuals.

2. Specification

```
SUBROUTINE G08DAF (X, IX, K, N, RNK, W, P, IFAIL)
INTEGER        IX, K, N, IFAIL
real           X(IX,N), RNK(IX,N), W, P
```

3. Description

Kendall's coefficient of concordance measures the degree of agreement between k comparisons of n objects, the scores in the ith comparison being denoted by

$$x_{i1},x_{i2},\ldots,x_{in}.$$

The hypothesis under test, H_0, often called the null hypothesis, is that there is no agreement between the comparisons, and this is to be tested against the alternative hypothesis H_1 that there is some agreement.

The n scores for each comparison are ranked, the rank r_{ij} denoting the rank of object j in comparison i, and all ranks lying between 1 and n. Average ranks are assigned to tied scores.

For each of the n objects, the k ranks are totalled, giving rank sums R_j, for $j = 1,2,\ldots,n$. Under H_0, all the R_j would be approximately equal to the average rank sum $\frac{1}{2}k(n+1)$. The total squared deviation of the R_j's from this average value is therefore a measure of the departure from H_0 exhibited by the data. If there were complete agreement between the comparisons, the rank sums R_j would have the values $k,2k,\ldots,nk$ (or some permutation thereof). The total squared deviation of these values is $k^2(n^3-n)/12$.

Kendall's coefficient of concordance is the ratio

$$W = \frac{\sum_{j=1}^{n}(R_j-\frac{1}{2}k(n+1))^2}{\frac{1}{12}k^2(n^3-n)}$$

and lies between 0 and 1, the value 0 indicating complete disagreement, and 1 indicating complete agreement.

If there are tied rankings within comparisons, W is corrected by subtracting $k\sum T$ from the denominator, where $T = \sum(t^3-t)/12$, each t being the number of occurrences of each tied rank within a comparison, and the summation of T being over all comparisons containing ties.

G08DAF returns the value of W, and also an approximation, p, of the significance of the observed W. (For $n > 7$, $k(n-1)W$ approximately follows a χ^2_{n-1} distribution, so large values of W imply rejection of H_0). H_0 is rejected by a test of chosen size α if $p < \alpha$. If $n \le 7$, tables should be used to establish the significance of W (e.g. table R of Siegel [1]).

4. References

[1] SIEGEL, S.
Nonparametric Statistics for the Behavioral Sciences, Ch. 9.
McGraw-Hill, 1956.

5. Parameters

1: X(IX,N) – ***real*** array. *Input*

On entry: X(i,j) must be set to the value x_{ij} of object j in comparison i, for $i = 1,2,...,k$; $j = 1,2,...,n$.

2: IX – INTEGER. *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G08DAF is called.

Constraint: IX ≥ K.

3: K – INTEGER. *Input*

On entry: the number of comparisons, k.

Constraint: K ≥ 2.

4: N – INTEGER. *Input*

On entry: the number of objects, n.

Constraint: N ≥ 2.

5: RNK(IX,N) – ***real*** array. *Workspace*

6: W – ***real.*** *Output*

On exit: the value of Kendall's coefficient of concordance, W.

7: P – ***real.*** *Output*

On exit: the approximate significance, p, of W.

8: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, N < 2.

IFAIL = 2

On entry, IX < K.

IFAIL = 3

On entry, K ≤ 1.

7. Accuracy

All computations are believed to be stable. The statistic W should be accurate enough for all practical uses.

8. Further Comments

The time taken by the routine is approximately proportional to the product nk.

9. Example

This example is taken from page 234 of Seigel [1]. The data consists of 10 objects ranked on three different variables: X, Y and Z. The computed values of Kendall's coefficient is significant at the 1% level of significance ($p = 0.008 < 0.01$), indicating that the null hypothesis of there being no agreement between the three rankings X, Y, Z may be rejected with reasonably high confidence.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08DAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          N, K, IX
      PARAMETER        (N=10,K=3,IX=K)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             P, W
      INTEGER          I, IFAIL, J
*     .. Local Arrays ..
      real             RNK(IX,N), X(IX,N)
*     .. External Subroutines ..
      EXTERNAL         G08DAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08DAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) ((X(I,J),J=1,N),I=1,K)
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Kendall''s coefficient of concordance'
      WRITE (NOUT,*)
      WRITE (NOUT,*) 'Data values'
      WRITE (NOUT,*)
      WRITE (NOUT,99999) ('Comparison ',I,' scores  ',(X(I,J),J=1,N),
     +  I=1,K)
      IFAIL = 0
*
      CALL G08DAF(X,IX,K,N,RNK,W,P,IFAIL)
*
      WRITE (NOUT,*)
      WRITE (NOUT,99998) 'Kendall''s coefficient =', W
      WRITE (NOUT,99998) '          Significance =', P
      STOP
*
99999 FORMAT (1X,A,I1,A,10F5.1)
99998 FORMAT (1X,A,F8.3)
      END
```

9.2. Program Data

```
G08DAF Example Program Data
  1.0  4.5  2.0  4.5  3.0  7.5  6.0  9.0  7.5 10.0
  2.5  1.0  2.5  4.5  4.5  8.0  9.0  6.5 10.0  6.5
  2.0  1.0  4.5  4.5  4.5  4.5  8.0  8.0  8.0 10.0
```

9.3. Program Results

```
G08DAF Example Program Results

Kendall's coefficient of concordance

Data values

Comparison 1 scores    1.0  4.5  2.0  4.5  3.0  7.5  6.0  9.0  7.5 10.0
Comparison 2 scores    2.5  1.0  2.5  4.5  4.5  8.0  9.0  6.5 10.0  6.5
Comparison 3 scores    2.0  1.0  4.5  4.5  4.5  4.5  8.0  8.0  8.0 10.0

Kendall's coefficient =   0.828
         Significance =   0.008
```

G08EAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08EAF performs a runs up (or a runs down) test on a sequence of observations.

2. Specification

```
 SUBROUTINE G08EAF (CL, N, X, M, MAXR, NRUNS, NCOUNT, EX, COV,
1                    LDCOV, CHI, DF, PROB, WRK, LWRK, IFAIL)
 INTEGER          N, M, MAXR, NRUNS, NCOUNT(MAXR), LDCOV, LWRK, IFAIL
 real             X(N), EX(MAXR), COV(LDCOV,MAXR), CHI, DF,
1                 PROB, WRK(LWRK)
 CHARACTER*1      CL
```

3. Description

Runs tests maybe used to investigate for trends in a sequence of observations. G08EAF computes statistics for the runs up test. If the runs down test is desired then each observation must be multiplied by -1 before G08EAF is called with the modified vector of observations.

G08EAF may be used in two different modes:

(i) a single call to G08EAF which computes all test statistics after counting the runs.

(ii) multiple calls to G08EAF with the final test statistics only being computed in the last call.

The second mode is necessary if all the data do not fit into the memory. See parameter CL in Section 5 for details on how to invoke each mode.

A run up is a sequence of numbers in increasing order. A run up ends at x_k when $x_k > x_{k+1}$ and the new run then begins at x_{k+1}. G08EAF counts the number of runs up of different lengths. Let c_i denote the number of runs of length i for $i = 1,2,...,r-1$. The number of runs of length r or greater is then denoted by c_r.

An unfinished run at the end of a sequence is not counted unless the sequence is part of an initial or intermediate call to G08EAF (i.e. unless there is another call to G08EAF to follow) in which case the unfinished run is used together with the beginning of the next sequence of numbers input to G08EAF in the next call. The following is a trivial example.

Suppose we called G08EAF twice with the following two sequences;

(0.20 0.40 0.45 0.40 0.15 0.75 0.95 0.23) and

(0.27 0.40 0.25 0.10 0.34 0.39 0.61 0.12).

Then after the second call G08EAF would have counted the runs up of the following lengths;

3, 1, 3, 3, 1, and 4.

When the counting of runs is complete G08EAF computes the expected values and covariances of the counts, c_i. For the details of the method used see Knuth [1]. An approximate χ^2 statistic with r degrees of freedom is computed where

$$X^2 = (c-\mu_c)^T \Sigma_c^{-1} (c-\mu_c)$$

where c is the vector of counts, c_i, for $i = 1,2,...,r$,

μ_c is the vector of expected values, e_i, for $i = 1,2,...,r$, where e_i is the expected value for c_i under the null hypothesis of randomness, and

Σ_c is the covariance matrix of c under the null hypothesis.

The use of the χ^2 distribution as an approximation to the exact distribution of the test statistic improves as the expected values increase.

The user may specify the total number of runs to be found. If the specified number of runs is found before the end of a sequence G08EAF will exit before counting any further runs. The number of runs actually counted and used to compute the test statistic is returned via NRUNS.

4. References

[1] KNUTH, D.E.
The Art of Computer Programming, Vol. 2, Ch. 3, pp. 60-63.
Addison Wesley, Massachusetts, 1980.

[2] MORGAN, B.J.T.
Elements of Simulation, Ch. 6, pp. 143-145.
Chapman and Hall, London, 1984.

[3] RIPLEY B.D.
Stochastic Simulation, Ch. 2, pp. 44.
Wiley, New York, 1987.

[4] DAGPUNAR, J.
Principles of Random Variate Generation, Ch. 2, pp. 43.
Oxford University Press, Oxford, 1988.

5. Parameters

1: CL – CHARACTER*1. *Input*

On entry: must specify the type of call to G08EAF,

CL = 'S' or 's', this is the one and only call to G08EAF (single call mode). All data are to be input at once. All test statistics are computed after the counting of runs is complete.

CL = 'F' or 'f', this is the first call to the routine. All initializations are carried out and the counting of runs begins. The final test statistics are not computed since further calls will be made to G08EAF.

CL = 'I' or 'i', this is an intermediate call during which the counts of runs are updated. The final test statistics are not computed since further calls will be made to G08EAF.

CL = 'L' or 'l', this is the last call to G08EAF. The test statistics are computed after the final counting of runs is completed.

Constraint: CL = 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

2: N – INTEGER. *Input*

On entry: the length of the current sequence of observations, n.

Constraints: if CL = 'S' or 's', then N $\geq$ 3,
otherwise N $\geq$ 1.

3: X(N) – ***real*** array. *Input*

On entry: the sequence of observations.

4: M – INTEGER. *Input*

On entry: the maximum number of runs to be sought. If M $\leq$ 0 then no limit is placed on the number of runs that are found.

M must not be changed between calls to G08EAF.

Constraint: If CL = 'S' or 's' then M $\leq$ N.

5: MAXR – INTEGER. *Input*

On entry: the length of the longest run for which tabulation is desired, r. That is, all runs with length greater than or equal to r are counted together.

MAXR must not be changed between calls to G08EAF.

Constraint: MAXR $\geq$ 1 and if CL = 'S' or 's', MAXR < N.

6: NRUNS – INTEGER. *Input/Output*

On entry: if CL = 'S', 's', 'F' or 'f', NRUNS need not be set.

If CL = 'I', 'i', 'L' or 'l', NRUNS must contain the value returned by the previous call to G08EAF.

On exit: the number of runs actually found.

7: NCOUNT(MAXR) – INTEGER array. *Input/Output*

On entry: if CL = 'S', 's', 'F' or 'f', NCOUNT need not be set.

If CL = 'I', 'i', 'L' or 'l', NCOUNT must contain the values returned by the previous call to G08EAF.

On exit: the counts of runs of the different lengths, c_i, for $i = 1,2,...,r$.

8: EX(MAXR) – ***real*** array. *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is the final exit) then EX contains the expected values of the counts, e_i, for $i = 1,2,...,r$.

Otherwise the elements of EX are not set.

9: COV(LDCOV,MAXR) – ***real*** array. *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is the final exit) then COV contains the covariance matrix of the counts, Σ_c.

Otherwise the elements of COV are not set.

10: LDCOV – INTEGER. *Input*

On entry: the first dimension of the array COV as declared in the (sub)program from which G08EAF is called.

Constraint: LDCOV $\geq$ MAXR

11: CHI – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is the final exit) then CHI contains the approximate χ^2 test statistic, X^2.

Otherwise CHI is not set.

12: DF – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is the final exit) then DF contains the degrees of freedom of the χ^2 statistic.

Otherwise DF is not set.

13: PROB – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is the final exit) then PROB contains the upper tail probability corresponding to the χ^2 test statistic, i.e. the significance level.

Otherwise PROB is not set.

14: WRK(LWRK) – ***real*** array. *Workspace*

15: LWRK – INTEGER. *Input*

On entry: the dimension of the array WRK as declared in the (sub)program from which G08EAF is called.

Constraint: $\text{LWRK} \geq \dfrac{\text{MAXR}\times(\text{MAXR}+5)}{2} + 1$.

16: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, CL ≠ 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

IFAIL = 2

On entry, N < 3 if CL = 'S' or 's',
or N < 1 otherwise.

IFAIL = 3

On entry, with CL = 'S' or 's', M > N.

IFAIL = 4

On entry, MAXR < 1,
or MAXR ≥ N and CL = 'S' or 's'.

IFAIL = 5

On entry, LDCOV < MAXR.

IFAIL = 6

On entry, LWRK < MAXR×(MAXR+5)/2 + 1.

IFAIL = 7

There is a tie in the sequence of observations.

IFAIL = 8

The total length of the runs found is less than MAXR.

IFAIL = 9

The covariance matrix COV is not positive-definite. This may be because the value of MAXR is too large relative to the full length of the series. Thus the approximate χ^2 test statistic cannot be computed.

IFAIL = 10

The number of runs requested were not found. All statistics are still computed and the information returned may still be of use.

7. Accuracy

The computations are believed to be stable. The computation of PROB given the values of CHI and DF will obtain a relative accuracy of 5 significant figures for most cases.

8. Further Comments

The time taken by the routine increases with the number of observations *n* and also depends to an extent on whether the call to G08EAF is an only, first, intermediate or last call.

9. Example

The following program performs a runs up test on 10000 pseudo-random numbers taken from a uniform distribution $U(0,1)$, generated by G05CAF. G08EAF is called 10 times with 1000 observations each time. No limit is placed on the number of runs to be counted. All runs of length 6 or more are counted together.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08EAF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          M, N, MAXR, LDC, LWRK
      PARAMETER        (M=0,N=1000,MAXR=6,LDC=10,LWRK=34)
*     .. Local Scalars ..
      real             CHI, DF, P
      INTEGER          I, IFAIL, J, NRUNS
      CHARACTER*1      CL
*     .. Local Arrays ..
      real             C(LDC,MAXR), EXPECT(MAXR), WRK(LWRK), X(N)
      INTEGER          NCOUNT(MAXR)
*     .. External Subroutines ..
      EXTERNAL         G05CBF, G05FAF, G08EAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08EAF Example Program Results'
      CALL G05CBF(0)
      DO 20 I = 1, 10
         IF (I.EQ.1) THEN
            CL = 'First'
         ELSE IF (I.EQ.10) THEN
            CL = 'Last'
         ELSE
            CL = 'Intermediate'
         END IF
         CALL G05FAF(0.0e0,1.0e0,N,X)
         IFAIL = -1
*
         CALL G08EAF(CL,N,X,M,MAXR,NRUNS,NCOUNT,EXPECT,C,LDC,CHI,DF,P,
    +                WRK,LWRK,IFAIL)
*
         IF (CL.NE.'L' .AND. CL.NE.'l' .AND. IFAIL.NE.0) GO TO 60
*
   20 CONTINUE
*
```

```
         IF (IFAIL.EQ.0 .OR. IFAIL.EQ.10) THEN
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Total number of runs found = ', NRUNS
            IF (IFAIL.EQ.10) WRITE (NOUT,*)
     +          ' ** Note : the number of runs requested were not found.'
            WRITE (NOUT,*)
            WRITE (NOUT,*) '                                 Count'
            WRITE (NOUT,*)
     +          '        1        2        3        4        5       >5'
            WRITE (NOUT,99998) (NCOUNT(J),J=1,MAXR)
            WRITE (NOUT,*)
            WRITE (NOUT,*) '                                 Expect'
            WRITE (NOUT,*)
     +          '        1        2        3        4        5       >5'
            WRITE (NOUT,99997) (EXPECT(J),J=1,MAXR)
            WRITE (NOUT,*)
            WRITE (NOUT,*) '                           Covariance matrix'
            WRITE (NOUT,*)
     +          '        1        2        3        4        5       >5'
            DO 40 I = 1, MAXR
               WRITE (NOUT,99996) I, (C(I,J),J=1,MAXR)
   40       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,99995) 'Chisq = ', CHI
            WRITE (NOUT,99994) 'DF    = ', DF
            WRITE (NOUT,99995) 'Prob  = ', P
         END IF
   60 STOP
*
99999 FORMAT (1X,A,I10)
99998 FORMAT (3X,6I9)
99997 FORMAT (3X,6F9.2)
99996 FORMAT (1X,I2,6F9.2)
99995 FORMAT (1X,A,F10.4)
99994 FORMAT (1X,A,F7.1)
      END
```

9.2. Program Data

None.

9.3. Program Results

```
 G08EAF Example Program Results

 Total number of runs found =       4991

                                 Count
        1        2        3        4        5       >5
     1657     2071      949      240       56       18

                                 Expect
        1        2        3        4        5       >5
  1667.00  2082.96   916.37   263.77    57.51    11.90

                           Covariance matrix
        1        2        3        4        5       >5
 1  1277.98  -194.57  -148.83   -71.58   -22.92    -6.67
 2  -194.57  1409.86  -490.46  -197.21   -55.19   -14.35
 3  -148.83  -490.46   601.26  -117.40   -31.23    -7.79
 4   -71.58  -197.21  -117.40   222.03   -10.75    -2.61
 5   -22.92   -55.19   -31.23   -10.75    54.80    -0.65
 6    -6.67   -14.35    -7.79    -2.61    -0.65    11.75

 Chisq =     7.0680
 DF    =     6.0
 Prob  =     0.3146
```

G08EBF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08EBF performs a pairs test on a sequence of observations in the interval [0,1].

2. Specification

```
      SUBROUTINE G08EBF (CL, N, X, MSIZE, LAG, NCOUNT, LDC, EX, CHI,
     1                   DF, PROB, WRK, IFAIL)
      INTEGER          N, MSIZE, LAG, NCOUNT(LDC,MSIZE), LDC, IFAIL
      real             X(N), EX, CHI, DF, PROB, WRK(2*LAG)
      CHARACTER*1      CL
```

3. Description

G08EBF computes the statistics for performing a pairs test which may be used to investigate deviations from randomness in a sequence of [0,1] observations.

For a given lag, $l \geq 1$, an m by m matrix, C, of counts is formed as follows; the element c_{jk} of C is the number of pairs $(\mathrm{X}(i),\mathrm{X}(i+1))$ such that

$$\frac{j-1}{m} \leq \mathrm{X}(i) < \frac{j}{m}$$

$$\frac{k-1}{m} \leq \mathrm{X}(i+l) < \frac{k}{m}$$

where $i = 1,3,5,...,n-1$, if $l = 1$
and $i = 1,2,...,l,2l+1,2l+2,...3l,4l+1,...,n-l$ if $l > 1$.

Note that all pairs formed are non-overlapping pairs and are thus independent under the assumption of randomness.

Under the assumption that the sequence is random, the expected number of pairs for each class (i.e. each element of the matrix of counts) is the same, that is the pairs should be uniformly distributed over the unit square $[0,1]^2$. Thus the expected number of pairs for each class is just the total number of pairs, $\sum_{j,k=1}^{m} c_{jk}$, divided by the number of classes, m^2.

The χ^2 test statistic used to test the hypothesis of randomness is defined as;

$$X^2 = \sum_{j,k=1}^{m} \frac{(c_{jk}-e)^2}{e}$$

where $e = \sum_{j,k=1}^{m} c_{jk}/m^2$ = expected number of pairs in each class.

The use of the χ^2 distribution as an approximation to the exact distribution of the test statistic, x^2, improves as the expected value, e, increases.

G08EBF may be used in two different modes:

(i) a single call to G08EBF which computes all test statistics after counting the pairs

(ii) multiple calls to G08EBF with the final test statistics only being computed in the last call.

The second mode is necessary if all the data do not fit into the memory. See parameter CL is Section 5 for details on how to invoke each mode.

4. References

[1] KNUTH, D.E.
The Art of Computer Programming, Vol. 2, Ch. 3, pp. 60-63.
Addison Wesley, Massachusetts, 1980.

[2] MORGAN, B.J.T.
Elements of Simulation, Ch. 6, pp. 143-145.
Chapman and Hall, London, 1984.

[3] RIPLEY B.D.
Stochastic Simulation, Ch. 2, pp. 44.
Wiley, New York, 1987.

[4] DAGPUNAR, J.
Principles of Random Variate Generation, Ch. 2, pp. 43.
Oxford University Press, Oxford, 1988.

5. Parameters

1: CL – CHARACTER*1. *Input*

On entry: indicates the type of call to G08EBF,

CL = 'S' or 's', this is the one and only call to G08EBF (single call mode). All data are to be input at once. All test statistics are computed after the counting of pairs is complete.

CL = 'F' or 'f', this is the first call to the routine. All initializations are carried out and the counting of pairs begins. The final test statistics are not computed since further calls will be made to G08EBF.

CL = 'I' or 'i', this is an intermediate call during which the counts of pairs are updated. The final test statistics are not computed since further calls will be made to G08EBF.

CL = 'L' or 'l', this is the last call to G08EBF. The test statistics are computed after the final counting of runs is complete.

Constraint: CL = 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

2: N – INTEGER. *Input*

On entry: the number of observations, n.

Constraints: if CL = 'S' or 's', then N $\geq$ 2,
otherwise N $\geq$ 1.

3: X(N) – ***real*** array. *Input*

On entry: the sequence of observations.

Constraint: $0.0 \leq \mathrm{X}(i) \leq 1.0$, for $i = 1,2,\ldots,n$.

4: MSIZE – INTEGER. *Input*

On entry: the size of the matrix of counts, m.

MSIZE must not be changed between calls to G08EBF.

Constraint: MSIZE $\geq$ 2.

5: LAG – INTEGER. *Input*

On entry: the lag, l, to be used in choosing pairs.

If LAG = 1, then we consider the pairs $(\mathrm{X}(i), \mathrm{X}(i+1))$ for $i = 1,3,\ldots,n-1$ where n is the number of observations.

If LAG > 1, then we consider the pairs (X(i), X(i+l)) for i = 1,2,...,l,2l+1,2l+2,..., 3l,4l+1,...,n–l where n is the number of observations.

LAG must not be changed between calls to G08EBF.

Constraints: LAG > 0,
if CL = 'S' or 's', LAG < N.

6: NCOUNT(LDC,MSIZE) – INTEGER array. *Input/Output*

On entry: if CL = 'S', 's', 'F' or 'f', NCOUNT need not be set, if CL = 'I', 'i', 'L' or 'l', NCOUNT must contain the values returned by the previous call to G08EBF.

On exit: NCOUNT is an MSIZE by MSIZE matrix containing the counts of the number of pairs in each cell, c_{ij}, for i,j = 1,2,...,m.

7: LDC – INTEGER. *Input*

On entry: the first dimension of the array NCOUNT as declared in the (sub)program from which G08EBF is called.

Constraint: LDC ≥ MSIZE.

8: EX – ***real***. *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then EX contains the expected number of counts in each cell, e.

Otherwise EX is not set.

9: CHI – ***real***. *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then CHI contains the χ^2 test statistic, X^2, for testing the null hypothesis of randomness.

Otherwise CHI is not set.

10: DF – ***real***. *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then DF contains the degrees of freedom for the χ^2 statistic.

Otherwise DF is not set.

11: PROB – ***real***. *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then PROB contains the upper tail probability associated with the χ^2 test statistic, i.e. the significance level.

Otherwise PROB is not set.

12: WRK(2*LAG) – ***real*** array. *Workspace*

WRK is used to store information between successive calls to G08EBF and therefore must not be changed.

13: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, CL ≠ 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

IFAIL = 2

On entry, N < 1.
or CL = 'S' or 's' and N < 2.

IFAIL = 3

On entry, MSIZE ≤ 1.

IFAIL = 4

On entry, LAG < 1,
or CL = 'S' or 's' and LAG ≥ N.

IFAIL = 5

On entry, LDC < MSIZE.

IFAIL = 6

On entry, X(i) < 0.0,
or X(i) > 1.0 for some $i = 1,2,...,n$.

IFAIL = 7

No pairs were found. This will occur if the value of LAG is greater than or equal to the total number of observations.

IFAIL = 8

The expected value for each cell is less than or equal to 5.0. This implies that the χ^2 distribution may not be a very good approximation to the distribution of the test statistic.

7. Accuracy

The computations are believed to be stable. The computation of PROB given the values of CHI and DF will obtain a relative accuracy of 5 significant figures for most cases.

8. Further Comments

If after forming the pairs in an initial or intermediate call to G08EBF there is an observation left over at the end of the seqeunce, this observation is used at the beginning of the new sequence provided by the following call to G08EBF. Clearly an observation left over from an only or final call to G08EBF is ignored.

The time taken by the routine increases with the number of observations n, and depends to some extent whether the call is an only, first, intermediate or last call.

9. Example

The following program performs the pairs test on 10000 pseudo-random numbers from a uniform distribution $U(0,1)$ generated by G05CAF. G08EBF is called 10 times with 1000 observations on each call. LAG = 1 is used and the pairs are tallied into a 10 by 10 matrix.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08EBF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          N, MSIZE, LAG, LDC
      PARAMETER        (N=1000,MSIZE=10,LAG=1,LDC=10)
*     .. Local Scalars ..
      real             CHI, DF, EX, P
      INTEGER          I, IFAIL, IFAIL2
      CHARACTER*1      CL
*     .. Local Arrays ..
      real             WRK(2*LAG), X(N)
      INTEGER          NCOUNT(LDC,MSIZE)
*     .. External Subroutines ..
      EXTERNAL         G05CBF, G05FAF, G08EBF, X04EAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08EBF Example Program Results'
      CALL G05CBF(0)
      DO 20 I = 1, 10
         IF (I.EQ.1) THEN
            CL = 'First'
         ELSE IF (I.EQ.10) THEN
            CL = 'Last'
         ELSE
            CL = 'Intermediate'
         END IF
         CALL G05FAF(0.0e0,1.0e0,N,X)
         IFAIL = -1
*
         CALL G08EBF(CL,N,X,MSIZE,LAG,NCOUNT,LDC,EX,CHI,DF,P,WRK,IFAIL)
*
         IF (CL.NE.'L' .AND. CL.NE.'l' .AND. IFAIL.NE.0) GO TO 40
*
   20 CONTINUE
      IF (IFAIL.EQ.0 .OR. IFAIL.EQ.8) THEN
         WRITE (NOUT,*)
         IFAIL2 = 0
*
*        Print the count matrix
         CALL X04EAF('General',' ',MSIZE,MSIZE,NCOUNT,LDC,
     +               'Count matrix',IFAIL2)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'Expected value = ', EX
         WRITE (NOUT,99997) 'CHISQ          = ', CHI
         WRITE (NOUT,99998) 'DF             = ', DF
         WRITE (NOUT,99997) 'Probability    = ', P
         IF (IFAIL.EQ.8) WRITE (NOUT,*)
     +   ' ** Note : the Chi-squared approximation may not be very good.'
      END IF
   40 STOP
*
99999 FORMAT (1X,I2,10I7)
99998 FORMAT (1X,A,F8.2)
99997 FORMAT (1X,A,F10.4)
      END
```

9.2. Program Data

None.

9.3. Program Results

```
G08EBF Example Program Results

Count matrix
     1  2  3  4  5  6  7  8  9 10
 1  48 40 39 50 62 60 55 46 41 41
 2  46 50 49 49 44 55 68 59 46 38
 3  44 49 48 43 50 36 60 65 50 48
 4  56 46 52 37 45 44 43 50 50 45
 5  44 51 41 43 50 57 43 56 54 57
 6  43 62 57 58 49 49 57 41 40 47
 7  47 49 45 41 62 55 43 45 66 49
 8  52 46 52 52 51 52 47 46 47 49
 9  56 46 48 70 45 64 59 59 53 62
10  60 49 50 45 40 61 59 46 37 49

Expected value =     50.00
CHISQ          =    109.8800
DF             =     99.00
Probability    =      0.2137
```

G08ECF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08ECF performs the triplets test on a sequence of observations from the interval [0,1].

2. Specification

```
      SUBROUTINE G08ECF (CL, N, X, MSIZE, NCOUNT, LDC, EX, CHI, DF, PROB,
     1                   IFAIL)
      INTEGER          N, MSIZE, NCOUNT(LDC,LDC,MSIZE), LDC, IFAIL
      real             X(N), EX, CHI, DF, PROB
      CHARACTER*1      CL
```

3. Description

G08ECF computes the statistics for performing a triplets test which may be used to investigate deviations from randomness in a sequence of [0,1] observations.

An m by m matrix, C, of counts is formed as follows; the element c_{jkl} of C is the number of triplets (X(i), X(i+1), X(i+2)). for $i = 1,4,7,...,n-2$, such that

$$\frac{j-1}{m} \le \mathrm{X}(i) < \frac{j}{m}$$

$$\frac{k-1}{m} \le \mathrm{X}(i+1) < \frac{k}{m}$$

$$\frac{l-1}{m} \le \mathrm{X}(i+2) < \frac{l}{m}$$

Note that all triplets formed are non-overlapping and are thus independent under the assumption of randomness.

Under the assumption that the sequence is random, the expected number of triplets for each class (i.e. each element of the count matrix) is the same, that is the triplets should be uniformly distributed over the unit cube $[0,1]^3$. Thus the expected number of triplets for each class is just the total number of triplets, $\sum_{j,k,l=1}^{m} c_{jkl}$, divided by the number of classes, m^3.

The χ^2 test statistic used to test the hypothesis of randomness is defined as;

$$X^2 = \sum_{j,k,l=1}^{m} \frac{(c_{jkl}-e)^2}{e}$$

where $e = \sum_{j,k,l=1}^{m} c_{jkl}/m^3$ = expected number of triplets in each class.

The use of the χ^2 distribution as an approximation to the exact distribution of the test statistic, X^2, improves as the expected value, e, increases.

G08ECF may be used in two different modes:

(i) a single call to G08ECF which computes all test statistics after counting the triplets.

(ii) multiple calls to G08ECF with the final test statistics only being computed in the last call.

The second mode is necessary if all the data do not fit into the memeory. See parameter CL in Section 5 for details on how to invoke each mode.

4. References

[1] MORGAN, B.J.T.
Elements of Simulation, Ch. 6, pp. 143-145.
Chapman and Hall, London, 1984.

[2] RIPLEY B.D.
Stochastic Simulation, Ch. 2, pp. 44.
Wiley, New York, 1987.

[3] DAGPUNAR, J.
Principles of Random Variate Generation, Ch. 2, pp. 43.
Oxford University Press, Oxford, 1988.

5. Parameters

1: CL – CHARACTER*1. *Input*

On entry: indicates the type of call to G08ECF,

CL = 'S' or 's', this is the one and only call to G08ECF (single call mode). All data are to be input at once. All test statistics are computed after counting of the triplets is complete.

CL = 'F' or 'f', this is the first call to the routine. All initializations are carried out and the counting of triplets begins. The final test statistics are not computed since further calls will be made to G08ECF.

CL = 'I' or 'i', this is an intermediate call during which counts of the triplets are updated. The final test statistics are not computed since further calls will be made to G08ECF.

CL = 'L' or 'l', this is the last call to G08ECF. The test statistics are computed after the final counting of the triplets is complete.

Constraint: CL = 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

2: N – INTEGER. *Input*

On entry: the number of observations, n.

Constraints: if CL = 'S' or 's', then N $\geq$ 3,
otherwise N $\geq$ 1.

3: X(N) – ***real*** array. *Input*

On entry: the sequence of observations.

Constraint: $0.0 \leq \mathrm{X}(i) \leq 1.0$, for $i = 1,2,\ldots,n$.

4: MSIZE – INTEGER. *Input*

On entry: the size of the count matrix to be formed, m. MSIZE must not be changed between calls to G08ECF.

Constraint: MSIZE $\geq$ 2.

5: NCOUNT(LDC,LDC,MSIZE) – INTEGER array. *Input/Output*

On entry: if CL = 'S', 's', 'F' or 'f', NCOUNT need not be set.

If CL = 'I', 'i', 'L' or 'l', NCOUNT must contain the values returned by the previous call to G08ECF.

On exit: NCOUNT is an MSIZE by MSIZE by MSIZE matrix containing the counts of the number of triplets, c_{jkl}, for $j,k,l = 1,2,\ldots,m$.

6: LDC – INTEGER. *Input*

On entry: the first and second dimensions of the array NCOUNT as declared in the (sub)program from which G08ECF is called.

Constraint: LDC $\geq$ MSIZE.

7: EX – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then EX contains the expected number of counts for each element of the count matrix.

Otherwise EX is not set.

8: CHI – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then CHI contains the χ^2 test statistic, X^2 for testing the null hypothesis of randomness.

Otherwise CHI is not set.

9: DF – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then DF contains the degrees of freedom for the χ^2 statistic.

Otherwise DF is not set.

10: PROB – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then PROB contains the upper tail probability associated with the χ^2 test statistic, i.e. the significance level.

Otherwise PROB is not set.

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, CL ≠ 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

IFAIL = 2

On entry, N < 1,
or CL = 'S' or 's' and N < 3.

IFAIL = 3

On entry, MSIZE ≤ 1.

IFAIL = 4

On entry, LDC < MSIZE.

IFAIL = 5

On entry, X(i) < 0.0,
or X(i) > 1.0, for some $i = 1,2,...,n$.

IFAIL = 6

No triplets were found because less than 3 observations were provided in total.

IFAIL = 7

The expected value for the counts in each element of the count matrix is less than or equal to 5.0. This implies that the χ^2 distribution may not be a very good approximation to the distribution of the test statistic.

7. Accuracy

The computations are believed to be stable. The computations of PROB given the values of CHI and DF will obtain a relative accuracy of 5 significant figures for most cases.

8. Further Comments

If the call to G08ECF is an initial call or intermediate call with further calls to follow then any unused observations are saved for use at the beginning of the new sequence provided in the following call. Clearly any observations left over from an only or final call to G08ECF are ignored.

The time taken by the routine increases with the number of observations n, and depends to some extent whether the call is an only, first, intermediate or last call.

9. Example

The following program performs the pairs test on 10000 pseudo-random numbers from a uniform distribution $U(0,1)$ generated by G05CAF. G08ECF is called 10 times with 1000 observations on each call. The triplets are tallied into a 5 by 5 by 5 matrix.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08ECF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          N, MSIZE, LDC
      PARAMETER        (N=1000,MSIZE=5,LDC=5)
*     .. Local Scalars ..
      real             CHI, DF, EX, P
      INTEGER          I, IFAIL, J, K
      CHARACTER*1      CL
*     .. Local Arrays ..
      real             X(N)
      INTEGER          NCOUNT(LDC,LDC,MSIZE)
*     .. External Subroutines ..
      EXTERNAL         G05CBF, G05FAF, G08ECF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08ECF Example Program Results'
      CALL G05CBF(0)
      DO 20 I = 1, 10
         IF (I.EQ.1) THEN
            CL = 'First'
         ELSE IF (I.EQ.10) THEN
            CL = 'Last'
         ELSE
            CL = 'Intermediate'
         END IF
         CALL G05FAF(0.0e0,1.0e0,N,X)
         IFAIL = -1
*
         CALL G08ECF(CL,N,X,MSIZE,NCOUNT,LDC,EX,CHI,DF,P,IFAIL)
*
         IF (CL.NE.'L' .AND. CL.NE.'l' .AND. IFAIL.NE.0) GO TO 80
*
   20 CONTINUE
*
```

```
      IF (IFAIL.EQ.0 .OR. IFAIL.EQ.7) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Count matrix'
         DO 60 I = 1, MSIZE
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'I = ', I
            WRITE (NOUT,*) '        1      2      3      4      5'
            WRITE (NOUT,*)
            DO 40 J = 1, MSIZE
               WRITE (NOUT,99998) J, (NCOUNT(I,J,K),K=1,MSIZE)
   40       CONTINUE
   60    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,99997) 'Expected value = ', EX
         WRITE (NOUT,99996) 'CHISQ          = ', CHI
         WRITE (NOUT,99997) 'DF             = ', DF
         WRITE (NOUT,99996) 'Prob           = ', P
         IF (IFAIL.EQ.7) WRITE (NOUT,*)
     +   ' ** Note : the chi square approximation may not be very good.'
      END IF
   80 STOP
*
99999 FORMAT (1X,A,I2)
99998 FORMAT (1X,I2,5I7)
99997 FORMAT (1X,A,F8.2)
99996 FORMAT (1X,A,F10.4)
      END
```

9.2. Program Data

None.

9.3. Program Results

```
 G08ECF Example Program Results

 Count matrix

 I =  1
        1      2      3      4      5

  1    22     28     22     26     24
  2    24     31     21     27     26
  3    29     33     24     28     22
  4    26     25     34     27     35
  5    31     24     19     23     28

 I =  2
        1      2      3      4      5

  1    30     24     38     32     16
  2    22     24     19     32     23
  3    23     21     17     24     26
  4    18     25     26     28     28
  5    28     35     29     27     30

 I =  3
        1      2      3      4      5

  1    25     30     30     24     19
  2    33     24     23     29     28
  3    33     21     23     35     37
  4    26     21     34     22     32
  5    28     35     30     25     35
 I =   4
         1        2        3        4        5

  1     23       20       38       21       35
  2     30       23       22       32       29
  3     32       17       30       20       22
  4     32       27       32       19       33
  5     24       30       28       25       23

 I =   5
         1        2        3        4        5

  1     25       15       31       30       25
  2     24       28       25       25       31
  3     26       18       25       26       26
  4     25       28       38       32       36
  5     19       22       32       28       20

 Expected value =    26.66
 CHISQ          =   124.0582
 DF             =   124.00
 Prob           =     0.4816
```

G08EDF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08EDF performs a gaps test on a sequence of observations.

2. Specification

```
      SUBROUTINE G08EDF (CL, N, X, M, MAXG, RLO, RUP, TOTLEN, NGAPS, NCOUNT,
     1                   EX, CHI, DF, PROB, IFAIL)
      INTEGER            N, M, MAXG, NGAPS, NCOUNT(MAXG), IFAIL
      real               X(N), RLO, RUP, TOTLEN, EX(MAXG), CHI, DF, PROB
      CHARACTER*1        CL
```

3. Description

Gaps tests are used to test for cyclical trend in a sequence of observations. G08EDF computes certain statistics for the gaps test.

G08EDF may be used in two different modes:

(i) a single call to G08EDF which computes all test statistics after counting the gaps.

(ii) multiple calls to G08EDF with the final test statistics only being computed in the last call.

The second mode is necessary if all the data do not fit into the memory. See parameter CL in Section 5 for details on how to invoke each mode.

The term gap is used to describe the distance between two numbers in the sequence that lie in the interval (r_l,r_u). That is a gap ends at x_j if $r_l \leq x_j \leq r_u$. The next gap then begins at x_{j+1}. The interval (r_l,r_u) should lie within the region of all possible numbers. For example if the test is carried out on a sequence of (0,1) random numbers then the interval (r_l,r_u) must be contained in the whole interval (0,1). Let t_{len} be the length of the interval which specifies all possible numbers.

G08EDF counts the number of gaps of different lengths. Let c_i denote the number of gaps of length i, for $i = 1,2,...,k-1$. The number of gaps of length k or greater is then denoted by c_k. An unfinished gap at the end of a sequence is not counted unless the sequence is part of an initial or intermediate call to G08EDF (i.e. unless there is another call to G08EDF to follow) in which case the unfinished gap is used. The following is a trivial example.

Suppose we called G08EDF twice (i.e. with CL = 'F' and then with CL = 'L') with the following two sequences and with RL = 0.30 and RU = 0.60;

(0.20 0.40 0.45 0.40 0.15 0.75 0.95 0.23) and

(0.27 0.40 0.25 0.10 0.34 0.39 0.61 0.12).

Then after the second call G08EDF would have counted the gaps of the following lengths;

2, 1, 1, 6, 3 and 1.

When the counting of gaps is complete G08EDF computes the expected values of the counts. A approximate χ^2 statistic with MAXG degrees of freedom is computed where

$$X^2 = \frac{\sum_{i=1}^{k} (c_i-e_i)^2}{e_i}$$

where $e_i = ngaps \times p \times (1-p)^{i-1}$ if $i < k$,

$= ngaps \times (1-p)^{i-1}$ if $i = k$.

where $ngaps$ = the number of gaps found and $p = (r_u-r_l)/t_{len}$.

The use of the χ^2 distribution as an approximation to the exact distribution of the test statistic improves as the expected values increase.

The user may specify the total number of gaps to be found. If the specified number of gaps is found before the end of a sequence G08EDF will exit before counting any further gaps. The number of gaps actually counted and used to compute the test statistic is returned via NGAPS.

4. References

[1] KNUTH, D.E.
The Art of Computer Programming, Vol. 2, Ch. 3, pp. 60-63.
Addison Wesley, Massachusetts, 1980.

[2] MORGAN, B.J.T.
Elements of Simulation, Ch. 6, pp. 143-145.
Chapman and Hall, London, 1984.

[3] RIPLEY B.D.
Stochastic Simulation, Ch. 2, pp. 44.
Wiley, New York, 1987.

[4] DAGPUNAR, J.
Principles of Random Variate Generation, Ch. 2, pp. 43.
Oxford University Press, Oxford, 1988.

5. Parameters

1: CL – CHARACTER*1. *Input*

On entry: indicates the type of call to G08EDF,

CL = 'S' or 's', this is the one and only call to G08EDF (single call mode). All data are to be input at once. All test statistics are computed after the counting of gaps is complete.

CL = 'F' or 'f', this is the first call to the routine. All initializations are carried out before the counting of gaps begins. The final test statistics are not computed since further calls will be made to G08EDF.

CL = 'I' or 'i', this is an intermediate call during which the counts of gaps are updated. The final test statistics are not computed since further calls will be made to G08EDF.

CL = 'L' or 'l', this is the last call to G08EDF. The test statistics are computed after the final counting of gaps is complete.

Constraint: CL = 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

2: N – INTEGER. *Input*

On entry: the length of the current sequence of observations, n.

Constraint: N $\geq$ 1.

3: X(N) – ***real*** array. *Input*

On entry: the sequence of observations.

4: M – INTEGER. *Input*

On entry: the maximum number of gaps to be sought. If M $\leq$ 0 then there no limit is placed on the number of gaps that are found.

M should not be changed between calls to G08EDF.

Constraint: M $\leq$ N if CL = 'S' or 's'.

5: MAXG – INTEGER. *Input*

On entry: the length of the longest gap for which tabulation is desired, k.

MAXG must not be changed between calls to G08EDF.

Constraints: MAXG > 1 and MAXG $\leq$ N if CL = 'S' or 's'.

6: RLO – ***real.*** *Input*

On entry: the lower limit of the interval to be used to define the gaps, r_l.

RLO must not be changed between calls to G08EDF.

Constraints: RLO < RUP and RUP − RLO < TOTLEN.

7: RUP – ***real.*** *Input*

On entry: the upper limit of the interval to be used to define the gaps, r_u.

RUP must not be changed between calls to G08EDF.

Constraints: RUP > RLO and RUP − RLO < TOTLEN.

8: TOTLEN – ***real.*** *Input*

On entry: the total length of the interval which contains all possible numbers that may arise in the sequence.

Constraints: TOTLEN > 0.0 and RUP − RLO < TOTLEN.

9: NGAPS – INTEGER. *Input/Output*

On entry: if CL = 'S', 's', 'F' or 'f', NGAPS need not be set.

If CL = 'I', 'i', 'L' or 'l', NGAPS must contain the value returned by the previous call to G08EDF.

On exit: the number of gaps actually found, *ngaps*.

10: NCOUNT(MAXG) – INTEGER array. *Input/Output*

On entry: if CL = 'S', 's', 'F' or 'f', NCOUNT need not be set.

If CL = 'I', 'i', 'L' or 'l', NCOUNT must contain the values returned by the previous call to G08EDF.

On exit: the counts of gaps of the different lengths, c_i, for $i = 1,2,...,k$.

11: EX(MAXG) – ***real*** array. *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then EX contains the expected values of the counts.
Otherwise the elements of EX are not set.

12: CHI – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then CHI contains the χ^2 test statistic, X^2, for testing the null hypothesis of randomness.
Otherwise CHI is not set.

13: DF – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then DF contains the degrees of freedom for the χ^2 statistic.
Otherwise DF is not set.

14: PROB – ***real.*** *Output*

On exit: if CL = 'S', 's', 'L' or 'l', (i.e. if it is a final exit) then PROB contains the upper tail probability associated with the χ^2 test statistic, i.e. the significance level.
Otherwise PROB is not set.

15: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL $\neq$ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, CL $\neq$ 'S', 's', 'F', 'f', 'I', 'i', 'L' or 'l'.

IFAIL = 2

On entry, N < 1.

IFAIL = 3

On entry, with CL = 'S' or 's', M > N.

IFAIL = 4

On entry, MAXG $\leq$ 1,
or with CL = 'S' or 's', MAXG > N.

IFAIL = 5

On entry, RLO $\geq$ RUP,
or TOTLEN $\leq$ 0.0,
or RUP – RLO $\geq$ TOTLEN.

IFAIL = 6

No gaps were found. The user may need to use a longer sequence or increase the size of the interval (r_l,r_u).

IFAIL = 7

The expected frequency of a certain class is zero, that is $e_i = 0$, for some $i = 1,2,...,k$.

IFAIL = 8

The number of gaps requested were not found.

IFAIL = 9

Some classes have expected frequencies less than 1.0. This implies that the χ^2 distribution may not be a very good approximation to the distribution of the test statistic.

7. Accuracy

The computations are believed to be stable. The computation of PROB given the values of CHI and DF will obtain a relative accuracy of 5 significant places for most cases.

8. Further Comments

The time taken by the routine increases with the number of observations n, and depends to some extent whether the call is an only, first, intermediate or last call.

9. Example

The following program performs the pairs test on 10000 pseudo-random numbers from a uniform distribution between 0 and 1 generated by G05CAF. G08EDF is called 10 times with 1000 observations on each call. All gaps of length 10 or more are counted together.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08EDF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
      INTEGER          M, N, MAXG
      real             RLO, RUP, TOTLEN
      PARAMETER        (M=0,N=1000,MAXG=10,RLO=0.4e0,RUP=0.6e0,
     +                 TOTLEN=1.0e0)
*     .. Local Scalars ..
      real             CHI, DF, P
      INTEGER          I, IFAIL, J, NGAP
      CHARACTER*1      CL
*     .. Local Arrays ..
      real             EX(MAXG), X(N)
      INTEGER          NCOUNT(MAXG)
*     .. External Subroutines ..
      EXTERNAL         G05CBF, G05FAF, G08EDF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08EDF Example Program Results'
      CALL G05CBF(0)
      DO 20 I = 1, 5
         IF (I.EQ.1) THEN
            CL = 'First'
         ELSE IF (I.EQ.5) THEN
            CL = 'Last'
         ELSE
            CL = 'Intermediate'
         END IF
         CALL G05FAF(0.0e0,1.0e0,N,X)
         IFAIL = -1
*
         CALL G08EDF(CL,N,X,M,MAXG,RLO,RUP,TOTLEN,NGAP,NCOUNT,EX,CHI,DF,
     +               P,IFAIL)
*
         IF (CL.NE.'L' .AND. CL.NE.'l' .AND. IFAIL.NE.0) GO TO 40
*
   20 CONTINUE
      IF (IFAIL.EQ.0 .OR. IFAIL.GE.8) THEN
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'Total number of gaps found = ', NGAP
         IF (IFAIL.EQ.8) WRITE (NOUT,*)
     +      ' ** Note : the number of gaps requested were not found.'
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Count'
         WRITE (NOUT,*)
     +'     0      1      2      3      4      5      6      7      8
     +  >9'
         WRITE (NOUT,99998) (NCOUNT(J),J=1,MAXG)
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Expect'
         WRITE (NOUT,*)
     +'     0      1      2      3      4      5      6      7      8
     +  >9'
         WRITE (NOUT,99997) (EX(J),J=1,MAXG)
         WRITE (NOUT,*)
         WRITE (NOUT,99996) 'Chisq = ', CHI
         WRITE (NOUT,99995) 'DF    = ', DF
         WRITE (NOUT,99996) 'Prob  = ', P
         IF (IFAIL.EQ.9) WRITE (NOUT,*)
     +   ' ** Note : the chi square approximation may not be very good.'
      END IF
   40 STOP
*
99999 FORMAT (1X,A,I10)
99998 FORMAT (1X,10I7)
99997 FORMAT (1X,10F7.1)
99996 FORMAT (1X,A,F10.4)
99995 FORMAT (1X,A,F7.1)
      END
```

9.2. Program Data

None.

9.3. Program Results

```
G08EDF Example Program Results

Total number of gaps found =       1067

Count
      0      1      2      3      4      5      6      7      8     >9
    221    182    143    104     98     74     51     36     39    119

Expect
      0      1      2      3      4      5      6      7      8     >9
  213.4  170.7  136.6  109.3   87.4   69.9   55.9   44.8   35.8  143.2

Chisq =     9.6190
DF    =     9.0
Prob  =     0.3822
```

G08RAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08RAF calculates the parameter estimates, score statistics and their variance-covariance matrices for the linear model using a likelihood based on the ranks of the observations.

2. Specification

```
      SUBROUTINE G08RAF (NS, NV, NSUM, Y, IP, X, NX, IDIST, NMAX, TOL,
     1                   PARVAR, NPVAR, IRANK, ZIN, ETA, VAPVEC, PAREST,
     2                   WORK, LWORK, IWA, IFAIL)
      INTEGER          NS, NV(NS), NSUM, IP, NX, IDIST, NMAX, NPVAR,
     1                 IRANK(NMAX), LWORK, IWA(NMAX), IFAIL
      real             Y(NSUM), X(NX,IP), TOL, PARVAR(NPVAR,IP),
     1                 ZIN(NMAX), ETA(NMAX), VAPVEC(NMAX*(NMAX+1)/2),
     2                 PAREST(4*IP+1), WORK(LWORK)
```

3. Description

Analysis of data can be made by replacing observations by their ranks. The analysis produces inference for regression parameters arising from the following model.

For random variables $Y_1,Y_2,\ldots,Y_n$ we assume that, after an arbitrary monotone increasing differentiable transformation, $h(.)$, the model

$$h(Y_i) = x_i^T\beta + \varepsilon_i \qquad (1)$$

holds, where x_i is a known vector of explanatory variables and β is a vector of p unknown regression coefficients. The ε_i are random variables assumed to be independent and identically distributed with a completely known distribution which can be one of the following: Normal, logistic, extreme value or double-exponential. In Pettitt [1] an estimate for β is proposed as $\hat{\beta} = MX^Ta$ with estimated variance-covariance matrix M. The statistics a and M depend on the ranks r_i of the observations Y_i and the density chosen for ε_i.

The matrix X is the n by p matrix of explanatory variables. It is assumed that X is of rank p and that a column or a linear combination of columns of X is not equal to the column vector of 1's or a multiple of it. This means that a constant term cannot be included in the model (1). The statistics a and M are found as follows. Let ε_i have pdf $f(\varepsilon)$ and let $g = -f'/f$. Let $W_1,W_2,\ldots,W_n$ be order statistics for a random sample of size n with the density $f(.)$. Define $Z_i = g(W_i)$, then $a_i = E(Z_{r_i})$. To define M we need $M^{-1} = X^T(B-A)X$, where B is an n by n diagonal matrix with $B_{ii} = E(g'(W_{r_i}))$ and A is a symmetric matrix with $A_{ij} = \text{cov}\,(Z_{r_i},Z_{r_j})$. In the case of the Normal distribution, the $Z_1 < \ldots < Z_n$ are standard Normal order statistics and $E(g'(W_i)) = 1$ for $i = 1,2,\ldots,n$.

The analysis can also deal with ties in the data. Two observations are adjudged to be tied if $|Y_i-Y_j| <$ TOL , where TOL is a user-supplied tolerance level.

Various statistics can be found from the analysis:

(a) The score statistic X^Ta. This statistic is used to test the hypothesis H_0: $\beta = 0$, see (e).

(b) The estimated variance-covariance matrix $X^T(B-A)X$ of the score statistic in (a).

(c) The estimate $\hat{\beta} = MX^Ta$.

(d) The estimated variance-covariance matrix $M = (X^T(B-A)X)^{-1}$ of the estimate $\hat{\beta}$.

(e) The χ^2 statistic $Q = \hat{\beta}^TM^{-1}\hat{\beta} = a^TX(X^T(B-A)X)^{-1}X^Ta$ used to test H_0: $\beta = 0$,. Under H_0, Q has an approximate χ^2 distribution with p degrees of freedom.

(f) The standard errors $M_{ii}^{\frac{1}{2}}$ of the estimates given in (c).

(g) Approximate zstatistics i.e. $Z_i = \hat{\beta}_i / s.e.(\hat{\beta}_i)$ for testing H_0: $\beta_i = 0$. For $i = 1,2,...,n$, Z_i has an approximate $N(0,1)$ distribution.

In many situations, more than one sample of observations will be available. In this case we assume the model,

$$h_k(Y_k) = X_k^T \beta + e_k, \qquad k = 1,2,...,\mathrm{NS}$$

where NS is the number of samples. In an obvious manner, Y_k and X_k are the vector of observations and the design matrix for the kth sample respectively. Note that the arbitrary transformation h_k can be assumed different for each sample since observations are ranked within the sample.

The earlier analysis can be extended to give a combined estimate of β as $\hat{\beta} = Dd$, where

$$D^{-1} = \sum_{k=1}^{\mathrm{NS}} X_k^T (B_k - A_k) X_k$$

and

$$d = \sum_{k=1}^{\mathrm{NS}} X_k^T a_k$$

with a_k, B_k and A_k defined as a, B and A above but for the kth sample.

The remaining statistics are calculated as for the one sample case.

4. References

[1] PETTITT, A.N.
Inference for the linear model using a likelihood based on ranks.
J. Roy. Statist. Soc. Ser. B, pp. 234-243, 1982.

5. Parameters

1: NS – INTEGER. *Input*

On entry: the number of samples.

Constraint: NS $\geq$ 1.

2: NV(NS) – INTEGER array. *Input*

On entry: the number of observations in the ith sample, for $i = 1,2,...,\mathrm{NS}$.

Constraint: NV(i) $\geq$ 1, for $i = 1,2,...,\mathrm{NS}$.

3: NSUM – INTEGER. *Input*

On entry: the total number of observations.

Constraint: $\mathrm{NSUM} = \sum_{i=1}^{\mathrm{NS}} \mathrm{NV}(i)$.

4: Y(NSUM) – ***real*** array. *Input*

On entry: the observations in each sample. Specifically, $\mathrm{Y}\left(\sum_{k=1}^{i-1} \mathrm{NV}(k) + j\right)$ must contain the jth observation in the ith sample.

5: IP – INTEGER. *Input*

On entry: the number of parameters to be fitted.

Constraint: IP $\geq$ 1.

6: X(NX,IP) – ***real*** array. *Input*

On entry: the design matrices for each sample. Specifically, $\mathrm{X}\left(\sum_{k=1}^{i-1} \mathrm{NV}(k)+j,l\right)$ must contain the value of the lth explanatory variable for the jth observation in the ith sample.

Constraint: X must not contain a column with all elements equal.

7: NX – INTEGER. *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G08RAF is called.

Constraint: NX $\geq$ NSUM.

8: IDIST – INTEGER. *Input*

On entry: the error distribution to be used in the analysis as follows:

IDIST = 1 – Normal
IDIST = 2 – Logistic
IDIST = 3 – Extreme value
IDIST = 4 – Double-exponential

Constraint: IDIST = 1, 2, 3 or 4

9: NMAX – INTEGER. *Input*

On entry: the value of the largest sample size.

Constraint: NMAX = $\max_{1 \leq i \leq \mathrm{NS}} (\mathrm{NV}(i))$ and NMAX > IP.

10: TOL – ***real.*** *Input*

On entry: the tolerance for judging whether two observations are tied. Thus, observations Y_i and Y_j are adjudged to be tied if $|Y_i - Y_j| <$ TOL.

Constraint: TOL > 0.0.

11: PARVAR(NPVAR,IP) – ***real*** array. *Output*

On exit: the variance-covariance matrices of the score statistics and the parameter estimates; the former being stored in the upper triangle and the latter in the lower triangle. Thus for $1 \leq i \leq j \leq$ IP, PARVAR(i,j) contains an estimate of the covariance between the ith and jth score statistics. For $1 \leq j \leq i \leq$ IP-1, PARVAR$(i+1,j)$ contains an estimate of the covariance between the ith and jth parameter estimates.

12: NPVAR – INTEGER. *Input*

On entry: the first dimension of the array PARVAR as declared in the (sub)program from which G08RAF is called.

Constraint: NPVAR $\geq$ IP + 1.

13: IRANK(NMAX) – INTEGER array. *Output*

On exit: for the one sample case, IRANK contains the ranks of the observations.

14: ZIN(NMAX) – ***real*** array. *Output*

On exit: for the one sample case, ZIN contains the expected values of the function $g(.)$ of the order statistics.

15: ETA(NMAX) – ***real*** array. *Output*

On exit: for the one sample case, ETA contains the expected values of the function $g'(.)$ of the order statistics.

16: VAPVEC(NMAX×(NMAX+1)/2) – ***real*** array. *Output*

On exit: for the one sample case, VAPVEC contains the upper triangle of the variance-covariance matrix of the function $g(.)$ of the order statistics stored column-wise.

17: PAREST(4*IP+1) – ***real*** array. *Output*

On exit: the statistics calculated by the routine as follows. The first IP components of PAREST contain the score statistics. The next IP elements contain the parameter estimates. PAREST(2×IP+1) contains the value of the χ^2 statistic. The next IP elements of PAREST contain the standard errors of the parameter estimates. Finally, the remaining IP elements of PAREST contain the z-statistics.

18: WORK(LWORK) – ***real*** array. *Workspace*

19: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G08RAF is called.

Constraint: LWORK ≥ NMAX×(IP+1).

20: IWA(NMAX) – INTEGER array. *Workspace*

21: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, NS < 1,
or TOL ≤ 0.0,
or NMAX ≤ IP,
or NPVAR < IP + 1,
or NX < NSUM,
or NMAX ≠ $\max_{1 \le i \le NS}(NV(i))$,
or NV(i) ≤ 0 for some i, NV(i),
or NSUM ≠ $\sum_{i=1}^{NS} NV(i)$,
or IP < 1,
or LWORK < NMAX×(IP+1).

IFAIL = 2

On entry, IDIST < 1,
or IDIST > 4.

IFAIL = 3

On entry, all the observations are adjudged to be tied. The user is advised to check the value set for TOL.

IFAIL = 4

The matrix $X^T(B-A)X$ is either ill-conditioned or not positive-definite. This fault should only occur with extreme rankings of the data.

IFAIL = 5

The matrix X has at least one of its columns with all elements equal.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine depends on the number of samples, the total number of observations and the number of parameters fitted.

In extreme cases the parameter estimates for certain models can be infinite, although this is unlikely to occur in practice. See Pettitt [1] for further details.

9. Example

A program to fit a regression model to a single sample of 20 observations using two explanatory variables. The error distribution will be taken to be logistic.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08RAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NSMAX, NX, NMXMAX, NSMMAX, IPMAX, NPVAR, LWORK
      PARAMETER        (NSMAX=5,NX=100,NMXMAX=NX,NSMMAX=NX,IPMAX=6,
     +                 NPVAR=IPMAX+1,LWORK=NMXMAX*(IPMAX+1))
*     .. Local Scalars ..
      real             TOL
      INTEGER          I, IDIST, IFAIL, IP, J, NMAX, NS, NSUM
*     .. Local Arrays ..
      real             ETA(NMXMAX), PAREST(4*IPMAX+1),
     +                 PARVAR(NPVAR,IPMAX), VAPVEC(NMXMAX*(NMXMAX+1)/2),
     +                 WORK(LWORK), X(NX,IPMAX), Y(NSMMAX), ZIN(NMXMAX)
      INTEGER          IRANK(NMXMAX), IWA(NMXMAX), NV(NSMAX)
*     .. External Subroutines ..
      EXTERNAL         G08RAF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08RAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*     Read number of samples, number of parameters to be fitted,
*     error distribution parameter and tolerance criterion for ties.
      READ (NIN,*) NS, IP, IDIST, TOL
      WRITE (NOUT,*)
      IF (NS.GT.0 .AND. NS.LE.NSMAX .AND. IP.GT.0 .AND. IP.LE.IPMAX)
     +    THEN
         WRITE (NOUT,99999) 'Number of samples =', NS
         WRITE (NOUT,99999) 'Number of parameters fitted =', IP
         WRITE (NOUT,99999) 'Distribution =', IDIST
         WRITE (NOUT,99998) 'Tolerance for ties =', TOL
*        Read the number of observations in each sample.
         READ (NIN,*) (NV(I),I=1,NS)
         NMAX = 0
         NSUM = 0
         DO 20 I = 1, NS
            NSUM = NSUM + NV(I)
            NMAX = MAX(NMAX,NV(I))
   20    CONTINUE
```

```
         IF (NMAX.GT.0 .AND. NMAX.LE.NMXMAX .AND. NSUM.GT.0 .AND.
     +       NSUM.LE.NSMMAX) THEN
*           Read in observations and design matrices for each sample.
            READ (NIN,*) (Y(I),(X(I,J),J=1,IP),I=1,NSUM)
            IFAIL = 0
*
            CALL G08RAF(NS,NV,NSUM,Y,IP,X,NX,IDIST,NMAX,TOL,PARVAR,
     +                  NPVAR,IRANK,ZIN,ETA,VAPVEC,PAREST,WORK,LWORK,
     +                  IWA,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Score statistic'
            WRITE (NOUT,99997) (PAREST(I),I=1,IP)
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Covariance matrix of score statistic'
            DO 40 J = 1, IP
               WRITE (NOUT,99997) (PARVAR(I,J),I=1,J)
   40       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Parameter estimates'
            WRITE (NOUT,99997) (PAREST(IP+I),I=1,IP)
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Covariance matrix of parameter estimates'
            DO 60 I = 1, IP
               WRITE (NOUT,99997) (PARVAR(I+1,J),J=1,I)
   60       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,99996) 'Chi-squared statistic =',
     +        PAREST(2*IP+1), ' with', IP, ' d.f.'
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Standard errors of estimates and'
            WRITE (NOUT,*) 'approximate z-statistics'
            WRITE (NOUT,99995) (PAREST(2*IP+1+I),PAREST(3*IP+1+I),I=1,
     +        IP)
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I2)
99998 FORMAT (1X,A,F8.5)
99997 FORMAT (1X,2F9.3)
99996 FORMAT (1X,A,F9.3,A,I2,A)
99995 FORMAT (1X,F9.3,F14.3)
      END
```

9.2. Program Data

```
G08RAF Example Program Data
 1 2 2 0.00001
 20
 1.0 1.0 23.0
 1.0 1.0 32.0
 3.0 1.0 37.0
 4.0 1.0 41.0
 2.0 1.0 41.0
 4.0 1.0 48.0
 1.0 1.0 48.0
 5.0 1.0 55.0
 4.0 1.0 55.0
 4.0 0.0 56.0
 4.0 1.0 57.0
 4.0 1.0 57.0
 4.0 1.0 57.0
 1.0 0.0 58.0
 4.0 1.0 59.0
 5.0 0.0 59.0
 5.0 0.0 60.0
 4.0 1.0 61.0
 4.0 1.0 62.0
 3.0 1.0 62.0
```

9.3. Program Results

```
G08RAF Example Program Results

Number of samples = 1
Number of parameters fitted = 2
Distribution = 2
Tolerance for ties = 0.00001

Score statistic
   -1.048   64.333

Covariance matrix of score statistic
    0.673
   -4.159  533.670

Parameter estimates
   -0.852    0.114

Covariance matrix of parameter estimates
    1.560
    0.012    0.002

Chi-squared statistic =    8.221 with 2 d.f.

Standard errors of estimates and
approximate z-statistics
    1.249        -0.682
    0.044         2.567
```

G08RBF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G08RBF calculates the parameter estimates, score statistics and their variance-covariance matrices for the linear model using a likelihood based on the ranks of the observations when some of the observations may be right censored.

2. Specification

```
SUBROUTINE G08RBF (NS, NV, NSUM, Y, IP, X, NX, ICEN, GAMMA, NMAX,
1                  TOL, PARVAR, NPVAR, IRANK, ZIN, ETA, VAPVEC,
2                  PAREST, WORK, LWORK, IWA, IFAIL)
INTEGER         NS, NV(NS), NSUM, IP, NX, ICEN(NSUM), NMAX, NPVAR,
1               IRANK(NMAX), LWORK, IWA(4*NMAX), IFAIL
real            Y(NSUM), X(NX,IP), GAMMA, TOL, PARVAR(NPVAR,IP),
1               ZIN(NMAX), ETA(NMAX), VAPVEC(NMAX*(NMAX+1)/2),
2               PAREST(4*IP+1), WORK(LWORK)
```

3. Description

Analysis of data can be made by replacing observations by their ranks. The analysis produces inference for the regression model where the location parameters of the observations, θ_i, $i = 1,2,...,n$, are related by $\theta = X\beta$. Here X is an n by p matrix of explanatory variables and β is a vector of p unknown regression parameters. The observations are replaced by their ranks and an approximation, based on Taylor's series expansion, made to the rank marginal likelihood. For details of the approximation see Pettitt [2].

An observation is said to be right censored if we can only observe Y_j^* with $Y_j^* \le Y_j$. We rank censored and uncensored observations as follows. Suppose we can observe Y_j, for $j = 1,2,...,n$, directly but Y_j^*, for $j = n+1,n+2,...,q$; $n \le q$, are censored on the right. We define the rank r_j of Y_j, for $j = 1,2,...,n$, in the usual way; r_j equals i if and only if Y_j is the ith smallest amongst the $Y_1,Y_2,...,Y_n$. The right censored Y_j^*, for $j = n+1,n+2,...,q$ has rank r_j if and only if Y_j^* lies in the interval $[Y_{(r_j)},Y_{(r_j+1)}]$, with $Y_0 = -\infty$, $Y_{(n+1)} = +\infty$ and $Y_{(1)} < ... < Y_{(n)}$ the ordered Y_j, for $j = 1,2,...,n$.

The distribution of the Y's is assumed to be of the following form. Let $F_L(y) = e^y/(1+e^y)$, the logistic distribution function, and consider the distribution function $F_\gamma(y)$ defined by $1-F_\gamma = [1-F_L(y)]^{1/\gamma}$. This distribution function can be thought of as either the distribution function of the minimum, $X_{1,\gamma}$, of a random sample of size γ^{-1} from the logistic distribution, or as the $F_\gamma(y-\log \gamma)$ being the distribution function of a random variable having the F-distribution with 2 and $2\gamma^{-1}$ degrees of freedom. This family of generalized logistic distribution functions $[F_\gamma(.);\ 0 \le \gamma < \infty]$ naturally links the symmetric logistic distribution $(\gamma = 1)$ with the skew extreme value distribution ($\lim \gamma \to 0$) and with the limiting negative exponential distribution ($\lim \gamma \to \infty$). For this family explicit results are available for right censored data. See Pettitt [3] for details.

Let l_R denote the logarithm of the rank marginal likelihood of the observations and define the $q\times 1$ vector a by $a = l_R'\ (\theta = 0)$, and let the q by q diagonal matrix B and q by q symmetric matrix A be given by $B - A = -l_R''\ (\theta = 0)$. Then various statistics can be found from the analysis.

(a) The score statistic X^Ta. This statistic is used to test the hypothesis H_0: $\beta = 0$ (see (e)).

(b) The estimated variance-covariance matrix of the score statistic in (a).

(c) The estimate $\hat{\beta}_R = MX^Ta$.

(d) The estimated variance-covariance matrix $M = (X^T(B-A)X)^{-1}$ of the estimate $\hat{\beta}_R$.

(e) The χ^2 statistic $Q = \hat{\beta}_R M^{-1}\hat{\beta}_r = a^T X(X^T(B-A)X)^{-1}X^T a$, used to test H_0: $\beta = 0$. Under H_0, Q has an approximate χ^2 distribution with p degrees of freedom.

(f) The standard errors $M_{ii}^{\frac{1}{2}}$ of the estimates given in (c).

(g) Approximate z-statistics, i.e. $Z_i = \hat{\beta}_{R_i}/s.e.(\hat{\beta}_{R_i})$ for testing H_0: $\beta_i = 0$. For $i = 1,2,\ldots,n$, Z_i has an approximate $N(0,1)$ distribution.

In many situations, more than one sample of observations will be available. In this case we assume the model,

$$h_k(Y_k) = X_k^T\beta + e_k, \qquad k = 1,2,\ldots,\mathrm{NS},$$

where NS is the number of samples. In an obvious manner, Y_k and X_k are the vector of observations and the design matrix for the kth sample respectively. Note that the arbitrary transformation h_k can be assumed different for each sample since observations are ranked within the sample.

The earlier analysis can be extended to give a combined estimate of β as $\hat{\beta} = Dd$, where

$$D^{-1} = \sum_{k=1}^{\mathrm{NS}} X^T(B_k-A_k)\ X_k$$

and

$$d = \sum_{k=1}^{\mathrm{NS}} X_k^T a_k.$$

with a_k, B_k and A_k defined as a, B and A above but for the kth sample.

The remaining statistics are calculated as for the one sample case.

4. References

[1] KALBFLEISCH, J.D. and PRENTICE, R.L.
The Statistical Analysis of Failure Time Data.
Wiley, New York, 1980.

[2] PETTITT, A.N.
Inference for the linear model using a likelihood based on ranks.
J. Roy. Statist. Soc. Ser. B, pp. 234-243, 1982.

[3] PETTITT, A.N.
Approximate methods using ranks for regression with censored data.
Biometrika, 70, pp. 121-132, 1983.

5. Parameters

1: NS – INTEGER. *Input*

On entry: the number of samples.

Constraint: NS $\geq$ 1.

2: NV(NS) – INTEGER array. *Input*

On entry: the number of observations in the ith sample, for $i = 1,2,\ldots,\mathrm{NS}$.

Constraint: NV(i) $\geq$ 1, for $i = 1,2,\ldots,\mathrm{NS}$.

3: NSUM – INTEGER. *Input*

On entry: the total number of observations.

Constraint: $\mathrm{NSUM} = \sum_{i=1}^{\mathrm{NS}} \mathrm{NV}(i)$.

4: Y(NSUM) – ***real*** array. *Input*

On entry: the observations in each sample. Specifically, $Y\left(\sum_{k=1}^{i-1} NV(k)+j\right)$ must contain the jth observation in the ith sample.

5: IP – INTEGER. *Input*

On entry: the number of parameters to be fitted.

Constraint: IP $\geq$ 1.

6: X(NX,IP) – ***real*** array. *Input*

On entry: the design matrices for each sample. Specifically, $X\left(\sum_{k=1}^{i-1} NV(k)+j,l\right)$ must contain the value of the lth explanatory variable for the jth observations in the ith sample.

Constraint: X must not contain a column with all elements equal.

7: NX – INTEGER. *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G08RBF is called.

Constraint: NX $\geq$ NSUM.

8: ICEN(NSUM) – INTEGER array. *Input*

On entry: defines the censoring variable for the observations in Y as follows:

ICEN(i) = 0 if Y(i) is uncensored.
ICEN(i) = 1 if Y(i) is censored.

Constraint: ICEN(i) = 0 or 1, for i = 1,2,...,NSUM.

9: GAMMA – ***real.*** *Input*

On entry: the value of the parameter defining the generalized logistic distribution. For GAMMA $\leq$ 0.0001, the limiting extreme value distribution is assumed.

Constraint: GAMMA > 0.0.

10: NMAX – INTEGER. *Input*

On entry: the value of the largest sample size.

Constraint: NMAX = $\max_{1\leq i\leq NS}(NV(i))$ and NMAX > IP.

11: TOL – ***real.*** *Input*

On entry: the tolerance for judging whether two observations are tied. Thus, observations Y_i and Y_j are adjudged to be tied if $|Y_i-Y_j| <$ TOL.

Constraint: TOL > 0.0.

12: PARVAR(NPVAR,IP) – ***real*** array. *Output*

On exit: the variance-covariance matrices of the score statistics and the parameter estimates; the former being stored in the upper triangle and the latter in the lower triangle. Thus for $1 \leq i \leq j \leq$ IP, PARVAR(i,j) contains an estimate of the covariance between the ith and jth score statistics. For $1 \leq j \leq i \leq$ IP$-$1, PARVAR(i+1,j) contains an estimate of the covariance between the ith and jth parameter estimates.

13: NPVAR – INTEGER. *Input*

On entry: the first dimension of the array PARVAR as declared in the (sub)program from which G08RBF is called.

Constraint: NPVAR $\geq$ IP + 1.

14: IRANK(NMAX) – INTEGER array. *Output*

On exit: for the one sample case, IRANK contains the ranks of the observations.

15: ZIN(NMAX) – ***real*** array. *Output*

On exit: for the one sample case, ZIN contains the expected values of the function g(.) of the order statistics.

16: ETA(NMAX) – ***real*** array. *Output*

On exit: for the one sample case, ETA contains the expected values of the function $g'(.)$ of the order statistics.

17: VAPVEC(NMAX*(NMAX+1)/2) – ***real*** array. *Output*

On exit: for the one sample case, VAPVEC contains the upper triangle of the variance-covariance matrix of the function g(.) of the order statistics stored column-wise.

18: PAREST(4*IP+1) – ***real*** array. *Output*

On exit: the statistics calculated by the routine as follows. The first IP components of PAREST contain the score statistics. The next IP elements contain the parameter estimates. PAREST(2×IP+1) contains the value of the χ^2 statistic. The next IP elements of PAREST contain the standard errors of the parameter estimates. Finally, the remaining IP elements of PAREST contain the *z*-statistics.

19: WORK(LWORK) – ***real*** array. *Workspace*

20: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G08RBF is called.

Constraint: LWORK $\geq$ NMAX×(IP+1).

21: IWA(4*NMAX) – INTEGER array. *Workspace*

22: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, NS < 1,
or TOL $\leq$ 0.0,
or NMAX $\leq$ IP,
or NPVAR < IP + 1,
or NX < NSUM,
or NMAX $\neq \max_{1 \leq i \leq NS}$ (NV(i)),
or NV(i) $\leq$ 0 for some i, i = 1,2,...,NS

or $\text{NSUM} \neq \sum_{i=1}^{\text{NS}} \text{NV}(i)$,

or IP < 1,

or GAMMA < 0.0,

or LWORK < NMAX×(IP+1).

IFAIL = 2

On entry, ICEN(i) ≠ 0 or 1 for some $1 \le i \le$ NSUM.

IFAIL = 3

On entry, all the observations are adjudged to be tied. The user is advised to check the value set for TOL.

IFAIL = 4

The matrix $X^T(B-A)X$ is either ill-conditioned or not positive-definite. This fault should only occur with extreme rankings of the data.

IFAIL = 5

On entry, at least one column of the matrix X has all its elements equal.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine depends on the number of samples, the total number of observations and the number of parameters fitted.

In extreme cases the parameter estimates for certain models can be infinite, although this is unlikely to occur in practice. See Pettitt [2] for further details.

9. Example

A program to fit a regression model to a single sample of 40 observations using just one explanatory variable.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G08RBF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NSMAX, NX, NMXMAX, NSMMAX, IPMAX, NPVAR, LWORK
      PARAMETER        (NSMAX=5,NX=100,NMXMAX=NX,NSMMAX=NX,IPMAX=6,
     +                 NPVAR=IPMAX+1,LWORK=NMXMAX*(IPMAX+1))
*     .. Local Scalars ..
      real             GAMMA, TOL
      INTEGER          I, IFAIL, IP, J, NMAX, NS, NSUM
*     .. Local Arrays ..
      real             ETA(NMXMAX), PAREST(4*IPMAX+1),
     +                 PARVAR(NPVAR,IPMAX), VAPVEC(NMXMAX*(NMXMAX+1)/2),
     +                 WORK(LWORK), X(NX,IPMAX), Y(NSMMAX), ZIN(NMXMAX)
      INTEGER          ICEN(NSMMAX), IRANK(NMXMAX), IWA(4*NMXMAX),
     +                 NV(NSMAX)
*     .. External Subroutines ..
      EXTERNAL         G08RBF
```

```
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G08RBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*     Read number of samples, number of parameters to be fitted,
*     distribution power parameter and tolerance criterion for ties.
      READ (NIN,*) NS, IP, GAMMA, TOL
      WRITE (NOUT,*)
      IF (NS.GT.0 .AND. NS.LE.NSMAX .AND. IP.GT.0 .AND. IP.LE.IPMAX)
     +    THEN
         WRITE (NOUT,99999) 'Number of samples =', NS
         WRITE (NOUT,99999) 'Number of parameters fitted =', IP
         WRITE (NOUT,99998) 'Distribution power parameter =', GAMMA
         WRITE (NOUT,99998) 'Tolerance for ties =', TOL
*        Read the number of observations in each sample
         READ (NIN,*) (NV(I),I=1,NS)
         NMAX = 0
         NSUM = 0
         DO 20 I = 1, NS
            NSUM = NSUM + NV(I)
            NMAX = MAX(NMAX,NV(I))
   20    CONTINUE
         IF (NMAX.GT.0 .AND. NMAX.LE.NMXMAX .AND. NSUM.GT.0 .AND.
     +       NSUM.LE.NSMMAX) THEN
*           Read in observations, design matrix and censoring variable
            READ (NIN,*) (Y(I),(X(I,J),J=1,IP),ICEN(I),I=1,NSUM)
            IFAIL = 0
*
            CALL G08RBF(NS,NV,NSUM,Y,IP,X,NX,ICEN,GAMMA,NMAX,TOL,PARVAR,
     +                  NPVAR,IRANK,ZIN,ETA,VAPVEC,PAREST,WORK,LWORK,
     +                  IWA,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Score statistic'
            WRITE (NOUT,99997) (PAREST(I),I=1,IP)
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Covariance matrix of score statistic'
            DO 40 J = 1, IP
               WRITE (NOUT,99997) (PARVAR(I,J),I=1,J)
   40       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Parameter estimates'
            WRITE (NOUT,99997) (PAREST(IP+I),I=1,IP)
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Covariance matrix of parameter estimates'
            DO 60 I = 1, IP
               WRITE (NOUT,99997) (PARVAR(I+1,J),J=1,I)
   60       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,99996) 'Chi-squared statistic =',
     +        PAREST(2*IP+1), ' with', IP, ' d.f.'
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Standard errors of estimates and'
            WRITE (NOUT,*) 'approximate z-statistics'
            WRITE (NOUT,99995) (PAREST(2*IP+1+I),PAREST(3*IP+1+I),I=1,
     +        IP)
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I2)
99998 FORMAT (1X,A,F10.5)
99997 FORMAT (1X,F9.3)
99996 FORMAT (1X,A,F9.3,A,I2,A)
99995 FORMAT (1X,F9.3,F14.3)
      END
```

9.2. Program Data

```
G08RBF Example Program Data
 1 1 0.00001 0.00001
 40
 143.0 0.0 0 164.0 0.0 0 188.0 0.0 0 188.0 0.0 0 190.0 0.0 0
 192.0 0.0 0 206.0 0.0 0 209.0 0.0 0 213.0 0.0 0 216.0 0.0 0
 220.0 0.0 0 227.0 0.0 0 230.0 0.0 0 234.0 0.0 0 246.0 0.0 0
 265.0 0.0 0 304.0 0.0 0 216.0 0.0 1 244.0 0.0 1 142.0 1.0 0
 156.0 1.0 0 163.0 1.0 0 198.0 1.0 0 205.0 1.0 0 232.0 1.0 0
 232.0 1.0 0 233.0 1.0 0 233.0 1.0 0 233.0 1.0 0 233.0 1.0 0
 239.0 1.0 0 240.0 1.0 0 261.0 1.0 0 280.0 1.0 0 280.0 1.0 0
 296.0 1.0 0 296.0 1.0 0 323.0 1.0 0 204.0 1.0 1 344.0 1.0 1
```

9.3. Program Results

```
 G08RBF Example Program Results

 Number of samples = 1
 Number of parameters fitted = 1
 Distribution power parameter =   0.00001
 Tolerance for ties =   0.00001

 Score statistic
     4.584

 Covariance matrix of score statistic
     7.653

 Parameter estimates
     0.599

 Covariance matrix of parameter estimates
     0.131

 Chi-squared statistic =    2.746 with 1 d.f.

 Standard errors of estimates and
 approximate z-statistics
     0.361         1.657
```

Chapter G10 – Smoothing in Statistics

Note. Please refer to the Users' Note for your implementation to check that a routine is available.

Routine Name	Mark of Introduction	Purpose
G10ABF	16	Fit cubic smoothing spline, smoothing parameter given
G10ACF	16	Fit cubic smoothing spline, smoothing parameter estimated
G10BAF	16	Kernel density estimate using Gaussian kernel
G10CAF	16	Compute smoothed data sequence using running median smoothers
G10ZAF	16	Reorder data to give ordered distinct observations

Chapter G10

Smoothing in Statistics

Contents

1 Scope of the Chapter

This chapter is concerned with methods for smoothing data. Included are methods for density estimation, smoothing time series data, and statistical applications of splines. These methods may also be viewed as nonparametric modelling.

2 Background to the Problems

2.1 Smoothing Methods

Many of the methods used in statistics involve fitting a model, the form of which is determined by a small number of parameters, for example, a distribution model like the gamma distribution, a linear regression model or an autoregression model in time series. In these cases the fitting involves the estimation of the small number of parameters from the data. In modelling data with these models there are two important stages in addition to the estimation of the parameters; these are: the identification of a suitable model, for example, the selection of a gamma distribution rather than a Weibull distribution, and the checking to see if the fitted model adequately fits the data. While these parametric models can be fairly flexible, they will not adequately fit all data sets, especially if the number of parameters is to be kept small.

Alternative models based on smoothing can be used. These models will not be written explicitly in terms of parameters. They are sufficiently flexible for a much wider range of situations than parametric models. The main requirement for such a model to be suitable is that the underlying models would be expected to be smooth, so excluding those situations where, for example, a step function would be expected.

These smoothing methods can be used in a variety of ways, for example:

(1) producing smoothed plots to aid understanding;
(2) identifying of a suitable parametric model from the shape of the smoothed data;
(3) eliminating complex effects that are not of direct interest so that attention can be focused on the effects of interest.

Several smoothing techniques make use of a smoothing parameter which can be either chosen by the user or estimated from the data. The smoothing parameter balances the two criterion of smoothness of the fitted model and the closeness of the fit of the model to the data. Generally, the larger the smoothing parameter is, the smoother the fitted model will be, but for small values of the smoothing parameter the model will closely follow the data, and for large values the fit will be poorer.

The smoothing parameter can either be chosen using previous experience of a suitable value for such data, or estimated from the data. The estimation can be either formal, using a criterion such as the cross-validation, or informal by trying different values and examining the result by means of suitable graphs.

Smoothing methods can be used in three important areas of of statistics: regression modelling, distribution modelling and time series modelling.

2.2 Smoothing Splines and Regression Models

For a set of n observations (y_i,x_i), $i = 1,2,\ldots,n$, the spline provides a flexible smooth function for situations in which a simple polynomial or nonlinear regression model is not suitable.

Cubic smoothing splines arise as the function, f, with continuous first derivative which minimises

$$\sum_{i=1}^{n} w_i\{y_i - f(x_i)\}^2 + \rho\int_{-\infty}^{\infty}(f''(x))^2dx,$$

where w_i is the (optional) weight for the ith observation and ρ is the smoothing parameter. This criterion consists of two parts: the first measures the fit of the curve and the second the smoothness of the curve. The value of the smoothing parameter, ρ, weights these two aspects: larger values of ρ give a smoother fitted curve but, in general, a poorer fit.

Splines are linear smoothers since the fitted values, $\hat{y} = (\hat{y}_1,\hat{y}_2,\ldots,\hat{y}_n)^T$, can be written as a linear function of the observed values $y = (y_1,y_2,\ldots,y_n)^T$, that is,

$$\hat{y} = Hy$$

for a matrix H. The degrees of freedom for the spline is trace(H) giving residual degrees of freedom

$$\mathrm{trace}(I - H) = \sum_{i=1}^{n}(1 - h_{ii}).$$

The diagonal elements of H, h_{ii}, are the leverages.

The parameter ρ can be estimated in a number of ways.

(1) The degrees of freedom for the spline can be specified, i.e., find ρ such that trace(H) $= \nu_0$ for given ν_0.

(2) Minimise the cross-validation (CV), i.e., find ρ such that the CV is mimimised, where

$$\mathrm{CV} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{r_i}{1 - h_{ii}}\right)^2.$$

(3) Minimise generalised cross-validation (GCV), i.e., find ρ such that the GCV is minimised, where

$$\mathrm{GCV} = n\left(\frac{\sum_{i=1}^{n} r_i^2}{\left(\sum_{i=1}^{n}(1 - h_{ii})\right)^2}\right).$$

2.3 Density Estimation

The object of density estimation is to produce from a set of observations a smooth nonparametric estimate of the unknown density function from which the observations were drawn. That is, given a sample of n observations, $x_1, x_2, \ldots, x_n$, from a distribution with unknown density function, $f(x)$, find an estimate of the density function, $\hat{f}(x)$. The simplest form of density estimator is the histogram; this may be defined by:

$$\hat{f}(x) = \frac{1}{nh}n_j; \quad a + (j-1)h < x < a + jh; \quad j = 1, 2, \ldots, n_s,$$

where n_j is the number of observations falling in the interval $a + (j-1)h$ to $a + jh$, a is the lower bound of the histogram and $b = n_s h$ is the upper bound. The value h is known as the window width. A simple development of this estimator would be the running histogram estimator

$$\hat{f}(x) = \frac{1}{2nh}n_x; \quad a \le x \le b,$$

where n_x is the number of observations falling in the interval $[x - h : x + h]$. This estimator can be written as

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} w\left(\frac{x - x_i}{h}\right)$$

for a function w where

$$\begin{aligned} w(x) &= \tfrac{1}{2} \quad \text{if } -1 < x < 1 \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

The function w can be considered as a kernel function. To produce a smoother density estimate, the kernel function, $K(t)$, which satisfies the following conditions can be used:

$$\int_{-\infty}^{\infty} K(t)dt = 1 \text{ and } K(t) \ge 0.0.$$

The kernel density estimator is therefore defined as:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right).$$

The choice of $K(\cdot)$ is usually not important but to ease computational burden, use can be made of Gausssian kernel defined as:

$$K(t) = \frac{1}{\sqrt{2\pi}}e^{-t^2/2}.$$

The smoothness of the estimator, $\hat{f}(x)$, depends on the window width, h. In general, the larger the value h is, the smoother the resulting density estimate is. There is, however, the problem of oversmoothing when the value of h is too large and essential features of the distribution function are removed. For example, if the distribution was bimodal, a large value of h may result in a unimodal estimate. The value of h has to be chosen such that the essential shape of the distribution is retained while effects due only to the observed sample are smoothed out. The choice of h can be aided by looking at plots of the density estimate for different values of h, or by using cross-validation methods; see Silverman [2].

Silverman [2] shows how the Gaussian kernel density estimator can be computed using a fast Fourier transform (FFT).

2.4 Smoothers for Time Series

If the data consists of a sequence of n observations recorded at equally spaced intervals, usually a time series, several robust smoothers are available. The fitted curve is intended to be robust to any outlying observations in the sequence, hence the techniques employed primarily make use of medians rather than means. These ideas come from the field of exploratory data analysis (EDA); see Tukey [3] and Velleman and Hoaglin [4]. The smoothers are based on the use of running medians to summarize overlapping segments; these provide a simple but flexible curve.

In EDA terminology, the fitted curve and the residuals are called the smooth and the rough respectively, so that

$$\text{Data} = \text{Smooth} + \text{Rough}.$$

Using the notation of Tukey, one of the smoothers commonly used is 4253H, twice. This consists of a running median of 4, then 2, then 5, then 3. This is then followed by what is known as Hanning. Hanning is a running weighted mean, the weights being 1/4, 1/2 and 1/4. The result of this smoothing is then 'reroughed'. This involves computing residuals from the computed fit, applying the same smoother to the residuals and adding the result to the smooth of the first pass.

3 Recommendations on Choice and Use of Available Routines

Note. Refer to the Users' Note for your implementation to check that a routine is available.

The following routines fit smoothing splines:

G10ABF	computes a cubic smoothing spline for a given value of the smoothing parameter. The results returned include the values of leverages and the coefficients of the cubic spline. Options allow only parts of the computation to be performed when the routine is used to estimate the value of the smoothing parameter or as when it is part of an iterative procedure such as that used in fitting generalized additive models; see Hastie and Tibshrani [1].
G10ACF	estimates the value of the smoothing parameter using one of three criteria and fits the cubic smoothing spline using that value.

G10ABF and G10ACF require the x_i to be strictly increasing. If two or more observations have the same x_i-value then they should be replaced by a single observation with y_i equal to the (weighted) mean of the y-values and weight, w_i, equal to the sum of the weights. This operation can be performed by G10ZAF.

The following routine produces an estimate of the density function:

G10BAF	computes a density estimate using a Normal kernel.

The following routine produces a smoothed estimate for a time series:

G10CAF	computes a smoothed series using running median smoothers.

The following service routine is also available:

G10ZAF	orders and weights the (x, y) input data to produce a data set strictly monotonic in x.

4 References

[1] Hastie T J and Tibshirani R J (1990) *Generalized Additive Models* Chapman and Hall

[2] Silverman B W (1990) *Density Estimation* Chapman and Hall

[3] Tukey J W (1977) *Exploratory Data Analysis* Addison-Wesley

[4] Velleman P F and Hoaglin D C (1981) *Applications, Basics, and Computing of Exploratory Data Analysis* Duxbury Press, Boston, MA

G10ABF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G10ABF fits a cubic smoothing spline for a given smoothing parameter.

2. Specification

```
      SUBROUTINE G10ABF (MODE, WEIGHT, N, X, Y, WT, RHO, YHAT, C, LDC, RSS,
     1                   DF, RES, H, WK, IFAIL)
      INTEGER          N, LDC, IFAIL
      real             X(N), Y(N), WT(*), RHO, YHAT(N), C(LDC,3), RSS, DF,
     1                 RES(N), H(N), WK(9*N+14)
      CHARACTER*1      MODE, WEIGHT
```

3. Description

G10ABF fits a cubic smoothing spline to a set of n observations (x_i, y_i), for $i = 1,2,...,n$. The spline provides a flexible smooth function for situations in which a simple polynomial or non-linear regression model is unsuitable.

Cubic smoothing splines arise as the unique real-valued solution function f, with absolutely continuous first derivative and squared-integrable second derivative, which minimises:

$$\sum_{i=1}^{n} w_i\{y_i - f(x_i)\}^2 + \rho \int_{-\infty}^{\infty} \{f''(x)\}^2 dx,$$

where w_i is the (optional) weight for the ith observation and ρ is the smoothing parameter. This criterion consists of two parts: the first measures the fit of the curve, and the second the smoothness of the curve. The value of the smoothing parameter ρ weights these two aspects, larger values of ρ give a smoother fitted curve but, in general, a poorer fit. For details of how the cubic spline can be estimated see Hutchinson and de Hoog [3] and Reinsch [4].

The fitted values, $\hat{y} = (\hat{y}_1,\hat{y}_2,...,\hat{y}_n)^T$, and weighted residuals, r_i, can be written as:

$$\hat{y} = Hy \text{ and } r_i = \sqrt{w_i}(y_i - \hat{y}_i)$$

for a matrix H. The residual degrees of freedom for the spline is trace$(I-H)$ and the diagonal elements of H, h_{ii}, are the leverages.

The parameter ρ can be chosen in a number of ways. The fit can be inspected for a number of different values of ρ. Alternatively the degrees of freedom for the spline, which determines the value of ρ, can be specified, or the (generalised) cross-validation can be minimised to give ρ; see G10ACF for further details.

G10ABF requires the x_i's to be strictly increasing. If two or more observations have the same x_i value then they should be replaced by a single observation with y_i equal to the (weighted) mean of the y values and weight, w_i, equal to the sum of the weights. This operation can be performed by G10ZAF.

The computation is split into three phases.

(1) Compute matrices needed to fit spline.

(2) Fit spline for a given value of ρ.

(3) Compute spline coefficients.

When fitting the spline for several different values of ρ, phase (1) need only be carried out once and then phase (2) repeated for different values of ρ. If the spline is being fitted as part of an iterative weighted least squares procedure phases (1) and (2) have to be repeated for each set of weights. In either case, phase (3) will often only have to be performed after the final fit has been computed.

The algorithm is based on Hutchinson [2].

4. References

[1] HASTIE, T.J. and TIBSHIRANI, R.J.
Generalized Additive Models.
Chapman and Hall, 1990.

[2] HUTCHINSON, M.F.
Algorithm 642: A fast procedure for calculating minimum cross-validation cubic smoothing splines.
ACM Trans. Math. Softw., 12, pp. 150-153, 1986.

[3] HUTCHINSON, M.F. and DE HOOG, F.R.
Smoothing noisy data with spline functions.
Numer. Math., 47, pp. 99-106, 1985.

[4] REINSCH, C.H.
Smoothing by spline functions.
Numer. Math. 10, pp. 177-183, 1967.

5. Parameters

1: MODE – CHARACTER*1. *Input*

On entry: indicates in which mode the routine is to be used.

If MODE = 'P', initialisation and fitting is performed. This Partial fit can be used in an iterative weighted least-squares context where the weights are changing at each call to G10ABF or when the coefficients are not required.
If MODE = 'Q', fitting only is performed, initialisation must have been performed previously by a call to G10ABF with MODE = 'P'. This Quick fit may be called repeatedly with different values of RHO without re-initialisation.
If MODE = 'F', initialisation and Full fitting is performed and the function coefficients are calculated.

Constraint: MODE = 'P', 'Q' or 'F'.

2: WEIGHT – CHARACTER*1. *Input*

On entry: indicates whether user-defined weights are to be used.

If WEIGHT = 'W', user-defined weights should be supplied in WT.
If WEIGHT = 'U', the data is treated as unweighted.

Constraint: WEIGHT = 'W' or 'U'.

3: N – INTEGER. *Input*

On entry: the number of distinct observations, n.

Constraint: $N \geq 3$.

4: X(N) – ***real*** array. *Input*

On entry: the distinct and ordered values x_i for $i = 1,2,...,n$.

Constraint: $X(i) < X(i+1)$, for $i = 1,2,...,n-1$.

5: Y(N) – ***real*** array. *Input*

On entry: the values y_i for $i = 1,2,...,n$.

6: WT(*) – ***real*** array. *Input*

Note: the dimension of the array WT must be at least 1 if WEIGHT = 'U' and N if WEIGHT = 'W'.

On entry: if WEIGHT = 'W', then WT must contain the n weights. If WEIGHT = 'U', WT is not referenced and unit weights are assumed.

Constraint: if WEIGHT = 'W' then WT(i) > 0.0 for $i = 1,2,...,n$.

7: RHO – ***real.*** ***Input***

On entry: the smoothing parameter, ρ.

Constraint: RHO ≥ 0.0.

8: YHAT(N) – ***real*** array. ***Output***

On exit: the fitted values, $\hat{y}_i$ for $i = 1,2,...,n$.

9: C(LDC,3) – ***real*** array. ***Input/Output***

On entry: if MODE = 'Q', C must be unaltered from the previous call to G10ABF with MODE = 'P'. Otherwise C need not be set.

On exit: if MODE = 'F', C contains the the spline coefficients. More precisely, the value of the spline at t is given by $((C(i,3)\times d + C(i,2))\times d + C(i,1))\times d + \hat{y}_i$, where $x_i \leq t < x_{i+1}$ and $d = t-x_i$.

If MODE = 'P' or 'Q', C contains information that will be used in a subsequent call to G10ABF with MODE = 'Q'.

10: LDC – INTEGER. ***Input***

On entry: the dimension of the array C as declared in the (sub)program from which G10ABF is called.

Constraint: LDC ≥ N – 1.

11: RSS – ***real.*** ***Output***

On exit: the (weighted) residual sum of squares.

12: DF – ***real.*** ***Output***

On exit: the residual degrees of freedom.

13: RES(N) – ***real*** array. ***Output***

On exit: the (weighted) residuals, r_i for $i = 1,2,...,n$.

14: H(N) – ***real*** array. ***Output***

On exit: the leverages, h_{ii} for $i = 1,2,...,n$.

15: WK(9*N+14) – ***real*** array. ***Input/Output***

On entry: if MODE = 'Q', WK must be unaltered from the previous call to G10ABF with MODE = 'P'. Otherwise WK is used as workspace and need not be set.

On exit: if MODE = 'P' or 'Q', WK contains information that will be used in a subsequent call to G10ABF with MODE = 'Q'.

16: IFAIL – INTEGER. ***Input/Output***

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N < 3,
or LDC < N − 1,
or RHO < 0.0,
or MODE ≠ 'Q', 'P' or 'F',
or WEIGHT ≠ 'W' or 'U'.

IFAIL = 2

On entry, WEIGHT = 'W' and at least one element of WT ≤ 0.0.

IFAIL = 3

On entry, X(i) ≥ X(i+1), for some i, i = 1,2,...,n−1.

7. Accuracy

Accuracy depends on the value of ρ and the position of the x values. The values of $x_i - x_{i-1}$ and w_i are scaled and ρ is transformed to avoid underflow and overflow problems.

8. Further Comments

The time taken by the routine is of order n.

Regression splines with a small ($< n$) number of knots can be fitted by E02BAF and E02BEF.

9. Example

The data, given by Hastie and Tibshirani [1], is the age, x_i, and C-peptide concentration (pmol/ml), y_i, from a study of the factors affecting insulin-dependent diabetes mellitus in children. The data is input, reduced to a strictly ordered set by G10ZAF and a spline is fitted by G10ABF with ρ = 10.0. The fitted values and residuals are printed.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G10ABF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LDC
      PARAMETER        (NMAX=43,LDC=NMAX-1)
*     .. Local Scalars ..
      real             DF, RHO, RSS
      INTEGER          I, IFAIL, N, NORD
      CHARACTER        MODE, WEIGHT
*     .. Local Arrays ..
      real             C(LDC,3), H(NMAX), RES(NMAX), WK(9*NMAX+14),
     +                 WT(NMAX), WWT(NMAX), X(NMAX), XORD(NMAX),
     +                 Y(NMAX), YHAT(NMAX), YORD(NMAX)
      INTEGER          IWRK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         G10ABF, G10ZAF
*     .. Executable Statements ..
      WRITE (NOUT,*) ' G10ABF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.GT.0 .AND. N.LE.NMAX) THEN
         READ (NIN,*) MODE, WEIGHT
         READ (NIN,*) RHO
         IF (WEIGHT.EQ.'U' .OR. WEIGHT.EQ.'u') THEN
            READ (NIN,*) (X(I),Y(I),I=1,N)
         ELSE
```

```
            READ (NIN,*) (X(I),Y(I),WT(I),I=1,N)
         END IF
         IFAIL = 0
*
*        Sort data into increasing X and
*        remove tied observations and weight accordingly
*
         IFAIL = 0
*
         CALL G10ZAF(WEIGHT,N,X,Y,WT,NORD,XORD,YORD,WWT,RSS,IWRK,IFAIL)
*
*        Fit cubic spline
*
         CALL G10ABF(MODE,'W',NORD,XORD,YORD,WWT,RHO,YHAT,C,LDC,RSS,DF,
     +               RES,H,WK,IFAIL)
*
*        Print results
*
         WRITE (NOUT,*)
         WRITE (NOUT,99999) ' RHO  = ', RHO
         WRITE (NOUT,*)
         WRITE (NOUT,99999) ' Residual sum of squares  = ', RSS
         WRITE (NOUT,99999) ' Degrees of freedom       = ', DF
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Ordered input data     Output results'
         WRITE (NOUT,*)
         WRITE (NOUT,*) '    X       Y           Fitted Values'
         WRITE (NOUT,*)
         DO 20 I = 1, NORD
            WRITE (NOUT,99998) XORD(I), YORD(I), YHAT(I)
   20    CONTINUE
      END IF
      STOP
*
99999 FORMAT (A,F10.3)
99998 FORMAT (1X,2F8.4,8X,F8.4)
      END
```

9.2. Program Data

```
G10ABF Example Program Data
43
'F', 'U'
10.0
 5.2 4.8    8.8 4.1   10.5 5.2   10.6 5.5   10.4 5.0
 1.8 3.4   12.7 3.4   15.6 4.9    5.8 5.6    1.9 3.7
 2.2 3.9    4.8 4.5    7.9 4.8    5.2 4.9    0.9 3.0
11.8 4.6    7.9 4.8   11.5 5.5   10.6 4.5    8.5 5.3
11.1 4.7   12.8 6.6   11.3 5.1    1.0 3.9   14.5 5.7
11.9 5.1    8.1 5.2   13.8 3.7   15.5 4.9    9.8 4.8
11.0 4.4   12.4 5.2   11.1 5.1    5.1 4.6    4.8 3.9
 4.2 5.1    6.9 5.1   13.2 6.0    9.9 4.9   12.5 4.1
13.2 4.6    8.9 4.9   10.8 5.1
```

9.3. Program Results

```
 G10ABF Example Program Results

RHO   =      10.000

Residual sum of squares  =      11.288
Degrees of freedom       =      27.785

 Ordered input data      Output results

    X        Y           Fitted Values

  0.9000  3.0000           3.3674
  1.0000  3.9000           3.4008
  1.8000  3.4000           3.6642
  1.9000  3.7000           3.7016
  2.2000  3.9000           3.8214
  4.2000  5.1000           4.5265
  4.8000  4.2000           4.6471
  5.1000  4.6000           4.7561
  5.2000  4.8500           4.7993
  5.8000  5.6000           5.0458
  6.9000  5.1000           5.1204
  7.9000  4.8000           4.9590
  8.1000  5.2000           4.9262
  8.5000  5.3000           4.8595
  8.8000  4.1000           4.8172
  8.9000  4.9000           4.8095
  9.8000  4.8000           4.8676
  9.9000  4.9000           4.8818
 10.4000  5.0000           4.9445
 10.5000  5.2000           4.9521
 10.6000  5.0000           4.9572
 10.8000  5.1000           4.9613
 11.0000  4.4000           4.9614
 11.1000  4.9000           4.9618
 11.3000  5.1000           4.9623
 11.5000  5.5000           4.9568
 11.8000  4.6000           4.9338
 11.9000  5.1000           4.9251
 12.4000  5.2000           4.8943
 12.5000  4.1000           4.8944
 12.7000  3.4000           4.9051
 12.8000  6.6000           4.9138
 13.2000  5.3000           4.9239
 13.8000  3.7000           4.8930
 14.5000  5.7000           4.9938
 15.5000  4.9000           4.9773
 15.6000  4.9000           4.9682
```

G10ACF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G10ACF estimates the values of the smoothing parameter and fits a cubic smoothing spline to a set of data.

2 Specification

```
      SUBROUTINE G10ACF(METHOD, WEIGHT, N, X, Y, WT, YHAT, C, LDC, RSS,
     1                  DF, RES, H, CRIT, RHO, U, TOL, MAXCAL, WK, IFAIL)
      INTEGER           N, LDC, MAXCAL, IFAIL
      real              X(N), Y(N), WT(*), YHAT(N), C(LDC,3), RSS, DF,
     1                  RES(N), H(N), CRIT, RHO, U, TOL, WK(7*(N+2))
      CHARACTER*1       METHOD, WEIGHT
```

3 Description

For a set of n observations (x_i, y_i), $i = 1, 2, \ldots, n$, the spline provides a flexible smooth function for situations in which a simple polynomial or non-linear regression model is not suitable.

Cubic smoothing splines arise as the unique real-valued solution function f, with absolutely continuous first derivative and squared-integrable second derivative, which minimises:

$$\sum_{i=1}^{n} w_i \{y_i - f(x_i)\}^2 + \rho \int_{-\infty}^{\infty} \{f''(x)\}^2 \, dx,$$

where w_i is the (optional) weight for the ith observation and ρ is the smoothing parameter. This criterion consists of two parts: the first measures the fit of the curve and the second the smoothness of the curve. The value of the smoothing parameter ρ weights these two aspects, larger values of ρ give a smoother fitted curve but, in general, a poorer fit. For details of how the cubic spline can be fitted see Hutchinson and de Hoog [3] and Reinsch [4].

The fitted values, $\hat{y} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)^T$, and weighted residuals, r_i, can be written as:

$$\hat{y} = Hy \text{ and } r_i = \sqrt{w_i}(y_i - \hat{y}_i)$$

for a matrix H. The residual degrees of freedom for the spline is trace$(I - H)$ and the diagonal elements of H are the leverages.

The parameter ρ can be estimated in a number of ways.

(1) The degrees of freedom for the spline can be specified, i.e., find ρ such that trace$(H) = \nu_0$ for given ν_0.

(2) Minimise the cross-validation (CV), i.e., find ρ such that the CV is minimised, where

$$\mathrm{CV} = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} \left[\frac{r_i}{1 - h_{ii}} \right]^2 .$$

(3) Minimise the generalised cross-validation (GCV), i.e., find ρ such that the GCV is minimised, where

$$\mathrm{GCV} = \frac{n^2}{\sum_{i=1}^{n} w_i} \left[\frac{\sum_{i=1}^{n} r_i^2}{\left(\sum_{i=1}^{n} (1 - h_{ii}) \right)^2} \right] .$$

G10ACF requires the x_i's to be strictly increasing. If two or more observations have the same x_i value then they should be replaced by a single observation with y_i equal to the (weighted) mean of the y values and weight, w_i, equal to the sum of the weights. This operation can be performed by G10ZAF.

The algorithm is based on Hutchinson [2]. C05AZF is used to solve for ρ given ν_0 and the method of E04ABF is used to minimise the GCV or CV.

4 References

[1] Hastie T J and Tibshirani R J (1990) *Generalized Additive Models* Chapman and Hall

[2] Hutchinson M F (1986) Algorithm 642: A fast procedure for calculating minimum cross-validation cubic smoothing splines *ACM Trans. Math. Software* **12** 150–153

[3] Hutchinson M F and De Hoog F R (1985) Smoothing noisy data with spline functions *Numer. Math.* **47** 99–106

[4] Reinsch C H (1967) Smoothing by spline functions *Numer. Math.* **10** 177–183

5 Parameters

1: METHOD — CHARACTER*1 *Input*

On entry: indicates whether the smoothing parameter is to be found by minimization of the CV or GCV functions, or by finding the smoothing parameter corresponding to a specified degrees of freedom value.

If METHOD = 'C' cross-validation is used.

If METHOD = 'D' the degrees of freedom are specified.

If METHOD = 'G' generalized cross-validation is used.

Constraint: METHOD = 'C', 'D' or 'G'.

2: WEIGHT — CHARACTER*1 *Input*

On entry: indicates whether user-defined weights are to be used.

If WEIGHT = 'W' user-defined weights should be supplied in WT.

If WEIGHT = 'U' the data is treated as unweighted.

Constraint: WEIGHT = 'W' or 'U'.

3: N — INTEGER *Input*

On entry: the number of observations, n.

Constraint: N ≥ 3.

4: X(N) — ***real*** array *Input*

On entry: the distinct and ordered values x_i for $i = 1, 2, \ldots, n$.

Constraint: X(i) < X($i+1$), $i = 1, 2, \ldots, n-1$.

5: Y(N) — ***real*** array *Input*

On entry: the values y_i for $i = 1, 2, \ldots, n$.

6: WT(*) — ***real*** array *Input*

Note: the dimension of the array WT must be at least 1 if WEIGHT = 'U' and N if WEIGHT = 'W'.

On entry: if WEIGHT = 'W' then WT must contain the n weights. If WEIGHT = 'U' then WT is not referenced and unit weights are assumed.

Constraint: if WEIGHT = 'W' then WT(i) > 0.0 for $i = 1, 2, \ldots, n$.

7: YHAT(N) — ***real*** array *Output*

On exit: the fitted values, $\hat{y}_i$ for $i = 1, 2, \ldots, n$.

8: C(LDC,3) — ***real*** array *Output*

On exit: the spline coefficients. More precisely, the value of the spline approximation at t is given by $((\mathrm{C}(i,3) \times d + \mathrm{C}(i,2)) \times d + \mathrm{C}(i,1)) \times d + \hat{y}_i$, where $x_i \leq t < x_{i+1}$ and $d = t - x_i$.

9: LDC — INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which G10ACF is called.

Constraint: LDC $\geq$ N $-$ 1.

10: RSS — ***real*** *Output*

On exit: the (weighted) residual sum of squares.

11: DF — ***real*** *Output*

On exit: the residual degrees of freedom. If METHOD = 'D' this will be n − CRIT to the required accuracy.

12: RES(N) — ***real*** array *Output*

On exit: the (weighted) residuals, r_i for $i = 1, 2, \ldots, n$.

13: H(N) — ***real*** array *Output*

On exit: the leverages, h_{ii} for $i = 1, 2, \ldots, n$.

14: CRIT — ***real*** *Input/Output*

On entry: if METHOD = 'D', the required degrees of freedom for the spline. If METHOD = 'C' or 'G', CRIT need not be set.

Constraint: 2.0 < CRIT $\leq$ N.

On exit: if METHOD = 'C', the value of the cross-validation, or if METHOD = 'G' the value of the generalized cross-validation function, evaluated at the value of ρ returned in RHO.

15: RHO — ***real*** *Output*

On exit: the smoothing parameter, ρ.

16: U — ***real*** *Input*

On entry: the upper bound on the smoothing parameter. See Section 8 for details on how this parameter is used.

Constraint: U > TOL.

Suggested value: U = 1000.0.

17: TOL — ***real*** *Input*

On entry: the accuracy to which the smoothing parameter RHO is required. TOL should be preferably not much less than $\sqrt{\epsilon}$, where ϵ is the ***machine precision***.

Constraint: TOL $\geq$ ***machine precision***.

18: MAXCAL — INTEGER *Input*

On entry: the maximum number of spline evaluations to be used in finding the value of ρ.

Constraint: MAXCAL $\geq$ 3.

Suggested value: MAXCAL = 30.

19: WK(7*(N+2)) — ***real*** array *Workspace*

20: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N < 3,
or LDC < N − 1,
or METHOD is not 'C', 'G' or 'D',
or WEIGHT is not 'W' or 'U',
or METHOD = 'D' and CRIT ≤ 2.0,
or METHOD = 'D' and CRIT > N,
or TOL < ***machine precision***,
or U ≤ TOL,
or MAXCAL < 3.

IFAIL = 2

On entry, WEIGHT = 'W' and at least one element of WT ≤ 0.0.

IFAIL = 3

On entry, $X(i) \geq X(i+1)$, for some i, $i = 1, 2, \ldots, n-1$.

IFAIL = 4

METHOD = 'D' and the required value of ρ for specified degrees of freedom > U. Try a larger value of U, see Section 8.

IFAIL = 5

METHOD = 'D' and the accuracy given by TOL cannot be achieved. Try increasing the value of TOL.

IFAIL = 6

A solution to the accuracy given by TOL has not been achieved in MAXCAL iterations. Try increasing the value of TOL and/or MAXCAL.

IFAIL = 7

METHOD = 'C' or 'G' and the optimal value of ρ > U. Try a larger value of U, see Section 8.

7 Accuracy

When minimising the cross-validation or generalised cross-validation, the error in the estimate of ρ should be within $\pm 3(\text{TOL} \times \text{RHO} + \text{TOL})$. When finding ρ for a fixed number of degrees of freedom the error in the estimate of ρ should be within $\pm 2 \times \text{TOL} \times \max(1, \text{RHO})$.

Given the value of ρ, the accuracy of the fitted spline depends on the value of ρ and the position of the x values. The values of $x_i - x_{i-1}$ and w_i are scaled and ρ is transformed to avoid underflow and overflow problems.

8 Further Comments

The time to fit the spline for a given value of ρ is of order n.

When finding the value of ρ that gives the required degrees of freedom, the algorithm examines the interval 0.0 to U. For small degrees of freedom the value of ρ can be large, as in the theoretical case of two degrees of freedom when the spline reduces to a straight line and ρ is infinite. If the CV or GCV is to be minimsed then the algorithm searches for the minimum value in the interval 0.0 to U. If the function is decreasing in that range then the boundary value of U will be returned. In either case, the larger the value of U the more likely is the interval to contain the required solution, but the process will be less efficient.

Regression splines with a small ($< n$) number of knots can be fitted by E02BAF and E02BEF.

9 Example

The data, given by Hastie and Tibshirani [1], is the age, x_i, and C-peptide concentration (pmol/ml), y_i, from a study of the factors affecting insulin-dependent diabetes mellitus in children. The data is input, reduced to a strictly ordered set by G10ZAF and a spline with 5 degrees of freedom is fitted by G10ACF. The fitted values and residuals are printed.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G10ACF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LDC
      PARAMETER        (NMAX=50,LDC=49)
*     .. Local Scalars ..
      real             CRIT, DF, RHO, RSS, TOL, U
      INTEGER          I, IFAIL, MAXCAL, N, NORD
      CHARACTER        METHOD, WEIGHT
*     .. Local Arrays ..
      real             C(LDC,3), H(NMAX), RES(NMAX), WK(7*(NMAX+2)),
     +                 WT(NMAX), WWT(NMAX), X(NMAX), XORD(NMAX),
     +                 Y(NMAX), YHAT(NMAX), YORD(NMAX)
      INTEGER          IWRK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         G10ACF, G10ZAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G10ACF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
         READ (NIN,*) METHOD, WEIGHT
         IF (WEIGHT.EQ.'U' .OR. WEIGHT.EQ.'u') THEN
            READ (NIN,*) (X(I),Y(I),I=1,N)
         ELSE
            READ (NIN,*) (X(I),Y(I),WT(I),I=1,N)
         END IF
         READ (NIN,*) U, TOL, MAXCAL, CRIT
         IFAIL = 0
*
```

```
            IFAIL = 0
*
*           Sort data, removing ties and weighting accordingly
*
            CALL G10ZAF(WEIGHT,N,X,Y,WT,NORD,XORD,YORD,WWT,RSS,IWRK,IFAIL)
*
*           Fit cubic spline
*
            CALL G10ACF(METHOD,'W',NORD,XORD,YORD,WWT,YHAT,C,LDC,RSS,DF,
     +                  RES,H,CRIT,RHO,U,TOL,MAXCAL,WK,IFAIL)
*
*           Print results
*
            WRITE (NOUT,*)
            WRITE (NOUT,99999) RSS
            WRITE (NOUT,99998) DF
            WRITE (NOUT,99997) RHO
            WRITE (NOUT,99996)
            DO 20 I = 1, NORD
               WRITE (NOUT,99995) I, XORD(I), YORD(I), YHAT(I), H(I)
   20       CONTINUE
         END IF
         STOP
*
99999 FORMAT (' Residual sum of squares = ',F10.2)
99998 FORMAT (' Degrees of freedom = ',F10.2)
99997 FORMAT (' RHO = ',F10.2)
99996 FORMAT (/'     Input data',16X,'Output results',/'   I    X     ',
     +       '  Y   ',9X,'YHAT     H')
99995 FORMAT (I4,2F8.3,6X,2F8.3)
      END
```

9.2 Program Data

```
G10ACF Example Program Data
43
'D', 'U'
 5.2 4.8    8.8 4.1   10.5 5.2   10.6 5.5   10.4 5.0
 1.8 3.4   12.7 3.4   15.6 4.9    5.8 5.6    1.9 3.7
 2.2 3.9    4.8 4.5    7.9 4.8    5.2 4.9    0.9 3.0
11.8 4.6    7.9 4.8   11.5 5.5   10.6 4.5    8.5 5.3
11.1 4.7   12.8 6.6   11.3 5.1    1.0 3.9   14.5 5.7
11.9 5.1    8.1 5.2   13.8 3.7   15.5 4.9    9.8 4.8
11.0 4.4   12.4 5.2   11.1 5.1    5.1 4.6    4.8 3.9
 4.2 5.1    6.9 5.1   13.2 6.0    9.9 4.9   12.5 4.1
13.2 4.6    8.9 4.9   10.8 5.1
10000  0.001  40  12.0
```

9.3 Program Results

```
G10ACF Example Program Results

Residual sum of squares =      10.35
Degrees of freedom =      25.00
RHO =       2.68

    Input data                Output results
 I    X        Y             YHAT      H
 1   0.900   3.000          3.373   0.534
 2   1.000   3.900          3.406   0.427
 3   1.800   3.400          3.642   0.313
 4   1.900   3.700          3.686   0.313
 5   2.200   3.900          3.839   0.448
 6   4.200   5.100          4.614   0.564
 7   4.800   4.200          4.576   0.442
 8   5.100   4.600          4.715   0.189
 9   5.200   4.850          4.783   0.407
10   5.800   5.600          5.193   0.455
11   6.900   5.100          5.184   0.592
12   7.900   4.800          4.958   0.530
13   8.100   5.200          4.931   0.235
14   8.500   5.300          4.845   0.245
15   8.800   4.100          4.763   0.271
16   8.900   4.900          4.748   0.292
17   9.800   4.800          4.850   0.301
18   9.900   4.900          4.875   0.277
19  10.400   5.000          4.970   0.173
20  10.500   5.200          4.977   0.154
21  10.600   5.000          4.979   0.285
22  10.800   5.100          4.970   0.136
23  11.000   4.400          4.961   0.137
24  11.100   4.900          4.964   0.284
25  11.300   5.100          4.975   0.162
26  11.500   5.500          4.975   0.186
27  11.800   4.600          4.930   0.213
28  11.900   5.100          4.911   0.220
29  12.400   5.200          4.852   0.206
30  12.500   4.100          4.857   0.196
31  12.700   3.400          4.900   0.189
32  12.800   6.600          4.932   0.193
33  13.200   5.300          4.955   0.488
34  13.800   3.700          4.797   0.408
35  14.500   5.700          5.076   0.559
36  15.500   4.900          4.979   0.445
37  15.600   4.900          4.946   0.535
```

G10BAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G10BAF performs kernel density estimation using a Gaussian kernel.

2. Specification

```
      SUBROUTINE G10BAF (N, X, WINDOW, SLO, SHI, NS, SMOOTH, T, USEFFT, FFT,
     1                   IFAIL)
      INTEGER            N, NS, IFAIL
      real               X(N), WINDOW, SLO, SHI, SMOOTH(NS), T(NS), FFT(NS)
      LOGICAL            USEFFT
```

3. Description

Given a sample of n observations, $x_1,x_2,...,x_n$, from a distribution with unknown density function, $f(x)$, an estimate of the density function, $\hat{f}(x)$, may be required. The simplest form of density estimator is the histogram. This may be defined by:

$$\hat{f}(x) = \frac{1}{nh}n_j;\quad a+(j-1)h < x < a+jh,\ j = 1,2,...,n_s,$$

where n_j is the number of observations falling in the interval $a+(j-1)h$ to $a+jh$, a is the lower bound to the histogram and $b = n_s h$ is the upper bound. The value h is known as the window width. To produce a smoother density estimate a kernel method can be used. A kernel function, $K(t)$, satisfies the conditions:

$$\int_{-\infty}^{\infty} K(t)dt = 1 \text{ and } K(t) \geq 0.$$

The kernel density estimator is then defined as:

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n}K\left(\frac{x-x_i}{h}\right).$$

The choice of K is usually not important but to ease the computational burden use can be made of the Gaussian kernel defined as:

$$K(t) = \frac{1}{\sqrt{2\pi}}e^{-t^2/2}.$$

The smoothness of the estimator depends on the window width h. The larger the value of h the smoother the density estimate. The value of h can be chosen by examining plots of the smoothed density for different values of h or by using cross-validation methods, see Silverman [3].

Silverman [2] and [3] shows how the Gaussian kernel density estimator can be computed using a fast Fourier transform (FFT). In order to compute the kernel density estimate over the range a to b the following steps are required:

(1) Discretize the data to give n_s equally spaced points t_l with weights ξ_l, see Jones and Lotwick [1].

(2) Compute the FFT of the weights, ξ_l to give Y_l.

(3) Compute $\zeta_l = e^{-\frac{1}{2}h^2 s_l^2}Y_l$ where $s_l = 2\pi l/(b-a)$.

(4) Find the inverse FFT of ζ_l to give $\hat{f}(x)$.

To compute the kernel density estimate for further values of h only steps (3) and (4) need be repeated.

4. References

[1] JONES, M.C. and LOTWICK, H.W.
Remark AS R50. A remark on Algorithm AS 176.
Appl. Statist., 33, pp. 120-122, 1984.

[2] SILVERMAN, B.W.
Algorithm AS 176. Kernel density estimation using the fast Fourier transform.
Appl. Statist., 31, pp. 93-99, 1982.

[3] SILVERMAN, B.W.
Density Estimation.
Chapman and Hall, 1990

5. Parameters

1: N – INTEGER. *Input*

On entry: the number of observations in the sample, n.

Constraint: $N > 0$.

2: X(N) – ***real*** array. *Input*

On entry: the n observations, x_i for $i = 1,2,...,n$.

3: WINDOW – ***real.*** *Input*

On entry: the window width, h.

Constraint: WINDOW > 0.0.

4: SLO – ***real.*** *Input*

On entry: the lower limit of the interval on which the estimate is calculated, a. For most applications SLO should be at least three window widths below the lowest data point.

Constraint: SLO $<$ SHI.

5: SHI – ***real.*** *Input*

On entry: the upper limit of the interval on which the estimate is calculated, b. For most applications SHI should be at least three window widths above the highest data point.

6: NS – INTEGER. *Input*

On entry: the number of points at which the estimate is calculated, n_s.

Constraints: NS ≥ 2.
The largest prime factor of NS must not exceed 19, and the total number of prime factors of NS, counting repetitions, must not exceed 20.

7: SMOOTH(NS) – ***real*** array. *Output*

On exit: the n_s values of the density estimate, $\hat{f}(t_l)$ for $l = 1,2,...,n_s$.

8: T(NS) – ***real*** array. *Output*

On exit: the points at which the estimate is calculated, t_l for $l = 1,2,...,n_s$.

9: USEFFT – LOGICAL. *Input*

On entry: must be set to .FALSE. if the values of Y_l are to be calculated by G10BAF and to .TRUE. if they have been computed by a previous call to G10BAF and are provided in FFT. If USEFFT = .TRUE. then the arguments N, SLO, SHI, NS and FFT must remain unchanged from the previous call to G10BAF with USEFFT = .FALSE..

10: FFT(NS) – ***real*** array. *Input/Output*

On entry: if USEFFT = .TRUE., then FFT must contain the fast Fourier transform of the weights of the discretized data, ξ_l, for $l = 1,2,...,n_s$. Otherwise FFT need not be set.

On exit: the fast Fourier transform of the weights of the discretized data, ξ_l, for $l = 1,2,...,n_s$.

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N ≤ 0,
or NS < 2,
or SHI ≤ SLO,
or WINDOW ≤ 0.0.

IFAIL = 2

On entry, G10BAF has been called with USEFFT = .TRUE. but the routine has not been called previously with USEFFT = .FALSE.,
or G10BAF has been called with USEFFT = .TRUE. but some of the arguments N, SLO, SHI, NS have been changed since the previous call to G10BAF with USEFFT = .FALSE..

IFAIL = 3

On entry, at least one prime factor of NS is greater than 19 or NS has more than 20 prime factors (see C06EAF).

IFAIL = 4

On entry, the interval given by SLO to SHI does not extend beyond three window widths at either extreme of the data set. This may distort the density estimate in some cases.

7. Accuracy

See Jones and Lotwick [1] for a discussion of the accuracy of this method.

8. Further Comments

The time for computing the weights of the discretized data is of order n while the time for computing the FFT is of order $n_s \log(n_s)$ as is the time for computing the inverse of the FFT.

9. Example

A sample of 1000 standard Normal (0,1) variates are generated using G05FDF and the density estimated on 100 points with a window width of 0.1. The resulting estimate of the density function is plotted using G01AGF.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G10BAF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          N, NS
      PARAMETER        (N=1000,NS=100)
*     .. Local Scalars ..
      real             SHI, SLO, WINDOW
      INTEGER          IFAIL, NSTEPX, NSTEPY
      LOGICAL          USEFFT
*     .. Local Arrays ..
      real             FFT(NS), S(NS), SMOOTH(NS), X(N)
      INTEGER          ISORT(NS)
*     .. External Subroutines ..
      EXTERNAL         G01AGF, G05FDF, G10BAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G10BAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) WINDOW
      READ (NIN,*) SLO, SHI
*
*     Generate Normal (0,1) Distribution
*
      CALL G05FDF(0.0e0,1.0e0,N,X)
*
*     Perform kernel density estimation
*
      USEFFT = .FALSE.
      IFAIL = 0
*
      CALL G10BAF(N,X,WINDOW,SLO,SHI,NS,SMOOTH,S,USEFFT,FFT,IFAIL)
*
*     Display smoothed data
*
      WRITE (NOUT,*)
      NSTEPX = 40
      NSTEPY = 20
      IFAIL = 0
*
      CALL G01AGF(S,SMOOTH,NS,ISORT,NSTEPX,NSTEPY,IFAIL)
      STOP
*
      END
```

9.2. Program Data

```
G10BAF Example Program Data
0.1
-4.0, 4.0
```

9.3. Program Results

```
G10BAF Example Program Results

           .+....+....+....+....+....+....+....+....+.
     0.5000+                    +                    +
           .                    .                    .
           .                    .                    .
           .                   11                    .
           .                   111                   .
     0.3750+                    +121                 +
           .                   1.1 2                 .
           .                  2 .   1                .
           .                 3  .   1                .
           .                1   .    1               .
     0.2500+                    +    1               +
           .               21   .    1               .
           .              21    .     1              .
           .             1      .     12             .
           .                    .      1             .
     0.1250+            22      +       1            +
           .           2        .       11           .
           .          11        .        2           .
           .         11         .         21         .
           .       322          .          22        .
     0.0000+1323232...+....+....+....+....+..32323231+
           .+....+....+....+....+....+....+....+....+.
          -4.000    -2.000     0.000     2.000     4.000
               -3.000    -1.000     1.000     3.000
```

G10CAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G10CAF computes a smoothed data sequence using running median smoothers.

2. Specification

```
SUBROUTINE G10CAF (ITYPE, N, Y, SMOOTH, ROUGH, IFAIL)
INTEGER        ITYPE, N, IFAIL
real           Y(N), SMOOTH(N), ROUGH(N)
```

3. Description

Given a sequence of n observations recorded at equally spaced intervals, G10CAF fits a smooth curve through the data using one of two smoothers. The two smoothers are based on the use of running medians and averages to summarize overlapping segments. The fit and the residuals are called the smooth and the rough respectively. They obey the following:

Data = Smooth + Rough.

The two smoothers are:

1) 4253H,twice consisting of a running median of 4, then 2, then 5, then 3 followed by Hanning. Hanning is a running weighted average, the weights being 1/4, 1/2 and 1/4. The result of this smoothing is then reroughed by computing residuals, applying the same smoother to them and adding the result to the smooth of the first pass.

2) 3RSSH,twice consisting of a running median of 3, two splitting operations named S to improve the smooth sequence, each of which is followed by a running median of 3, and finally Hanning. The end points are dealt with using the method described by Velleman and Hoaglin [2]. The full smoother 3RSSH,twice is produced by reroughing as described above.

The compound smoother 4253H,twice is recommended. The smoother 3RSSH,twice is popular when calculating by hand as it requires simpler computations and is included for comparison purposes.

4. References

[1] TUKEY, J.W.
Exploratory Data Analysis.
Addison Wesley, 1977.

[2] VELLEMAN, P.F. and HOAGLIN, D.C.
Applications, Basics and Computing of Exploratory Data Analysis.
Duxbury Press, Boston, Massachusetts, Ch. 6, 1981.

5. Parameters

1: ITYPE – INTEGER. *Input*

On entry: specifies the method to be used.

If ITYPE = 0, 4253H,twice is used.
If ITYPE = 1, 3RSSH,twice is used.

Constraint: ITYPE = 0 or 1.

2: N – INTEGER. *Input*

On entry: the number of observations, n.

Constraint: N > 6.

3: Y(N) – ***real*** array. *Input*

On entry: the sample observations.

4: SMOOTH(N) – ***real*** array. *Output*

On exit: contains the smooth.

5: ROUGH(N) – ***real*** array. *Output*

On exit: contains the rough.

6: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, ITYPE < 0,
or ITYPE > 1.

IFAIL = 2

On entry, N ≤ 6.

7. Accuracy

Not applicable.

8. Further Comments

Alternative methods of smoothing include the use of splines, see G10ABF and G10ACF.

9. Example

The example program reads in a sequence of 49 observations on bituminous coal production (in millions of net tons per year) in the U.S.A., 1920-1968 and is taken from Tukey [1]. For comparison purposes, both smoothers are applied to the data and the results are printed.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G10CAF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX
      PARAMETER        (NMAX=100)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, ITYPE, N
*     .. Local Arrays ..
      real             ROUGH(NMAX), ROUGH1(NMAX), SMOOT1(NMAX),
     +                 SMOOTH(NMAX), Y(NMAX)
*     .. External Subroutines ..
      EXTERNAL         G10CAF
```

```
*     .. Executable Statements ..
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.GT.0 .AND. N.LE.NMAX) THEN
         READ (NIN,*) (Y(I),I=1,N)
*
         ITYPE = 1
         IFAIL = 0
         CALL G10CAF(ITYPE,N,Y,SMOOTH,ROUGH,IFAIL)
         ITYPE = 0
         IFAIL = 0
         CALL G10CAF(ITYPE,N,Y,SMOOT1,ROUGH1,IFAIL)
*
         WRITE (NOUT,*) ' G10CAF Example Program Results'
         WRITE (NOUT,*)
         WRITE (NOUT,99999)
         WRITE (NOUT,99998)
         DO 20 I = 1, N
            WRITE (NOUT,99997) I, Y(I), SMOOTH(I), ROUGH(I), SMOOT1(I),
     +        ROUGH1(I)
   20    CONTINUE
      ELSE
         WRITE (NOUT,*) ' N is out of range'
      END IF
      STOP
*
99999 FORMAT ('                         Using 3RSSH,twice          Using 4',
     +       '253H,twice')
99998 FORMAT (' Index      Data      Smooth         Rough        Smooth    ',
     +       '       Rough')
99997 FORMAT (1X,I4,F11.1,4F13.4)
      END
```

9.2. Program Data

```
G10CAF Example Program Data
49
569.0 416.0 422.0 565.0 484.0 520.0 573.0 518.0 501.0 505.0
468.0 382.0 310.0 334.0 359.0 372.0 439.0 446.0 349.0 395.0
461.0 511.0 583.0 590.0 620.0 578.0 534.0 631.0 600.0 438.0
516.0 534.0 467.0 457.0 392.0 467.0 500.0 493.0 410.0 412.0
416.0 403.0 422.0 459.0 467.0 512.0 534.0 552.0 545.0
```

9.3. Program Results

```
 G10CAF Example Program Results

                         Using 3RSSH,twice          Using 4253H,twice
 Index      Data      Smooth         Rough        Smooth           Rough
    1      569.0    416.0000     153.0000     491.3750      77.6250
    2      416.0    416.0000       0.0000     491.3750     -75.3750
    3      422.0    431.5000      -9.5000     491.3750     -69.3750
    4      565.0    473.0000      92.0000     498.8828      66.1172
    5      484.0    509.5000     -25.5000     514.9375     -30.9375
    6      520.0    520.6875      -0.6875     524.6602      -4.6602
    7      573.0    521.5625      51.4375     525.0352      47.9648
    8      518.0    518.0000       0.0000     521.1602      -3.1602
    9      501.0    510.0000      -9.0000     512.5742     -11.5742
   10      505.0    496.5000       8.5000     493.1680      11.8320
   11      468.0    455.2500      12.7500     449.7422      18.2578
   12      382.0    387.5000      -5.5000     391.6133      -9.6133
   13      310.0    339.7500     -29.7500     353.4297     -43.4297
   14      334.0    334.9375      -0.9375     343.8438      -9.8438
   15      359.0    353.9375       5.0625     355.1602       3.8398
   16      372.0    376.1250      -4.1250     382.7930     -10.7930
   17      439.0    392.2500      46.7500     405.5469      33.4531
   18      446.0    396.2500      49.7500     411.8633      34.1367
   19      349.0    403.0000     -54.0000     411.5586     -62.5586
   20      395.0    427.2500     -32.2500     420.9375     -25.9375
```

```
      21      461.0     461.3750      -0.3750     456.1250       4.8750
      22      511.0     513.3125      -2.3125     513.8516      -2.8516
      23      583.0     567.5625      15.4375     565.2422      17.7578
      24      590.0     590.0000       0.0000     589.4688       0.5313
      25      620.0     593.5000      26.5000     594.7188      25.2813
      26      578.0     595.2500     -17.2500     594.5625     -16.5625
      27      534.0     590.9375     -56.9375     591.8125     -57.8125
      28      631.0     566.8125      64.1875     583.8438      47.1563
      29      600.0     531.5000      68.5000     569.0313      30.9688
      30      438.0     516.0000     -78.0000     546.3438    -108.3438
      31      516.0     516.0000       0.0000     517.2578      -1.2578
      32      534.0     501.8750      32.1250     489.6445      44.3555
      33      467.0     473.6250      -6.6250     471.2383      -4.2383
      34      457.0     457.0000       0.0000     463.4844      -6.4844
      35      392.0     452.0000     -60.0000     464.1875     -72.1875
      36      467.0     440.1250      26.8750     468.4688      -1.4688
      37      500.0     421.3750      78.6250     470.6094      29.3906
      38      493.0     412.0000      81.0000     462.2617      30.7383
      39      410.0     412.0000      -2.0000     438.5703     -28.5703
      40      412.0     412.0000       0.0000     416.1094      -4.1094
      41      416.0     411.0625       4.9375     408.8711       7.1289
      42      403.0     410.6875      -7.6875     412.1836      -9.1836
      43      422.0     422.0000       0.0000     424.8750      -2.8750
      44      459.0     446.6250      12.3750     448.1445      10.8555
      45      467.0     476.3750      -9.3750     478.7578     -11.7578
      46      512.0     509.0000       3.0000     510.0234       1.9766
      47      534.0     534.0000       0.0000     534.1250      -0.1250
      48      552.0     545.0000       7.0000     547.0000       5.0000
      49      545.0     547.7500      -2.7500     550.9375      -5.9375
```

G10ZAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G10ZAF orders and weights data which is entered unsequentially, weighted or unweighted.

2. Specification

```
   SUBROUTINE G10ZAF (WEIGHT, N, X, Y, WT, NORD, XORD, YORD, WTORD, RSS,
  1                   IWRK, IFAIL)
   INTEGER       N, NORD, IWRK(N), IFAIL
   real          X(N), Y(N), WT(*), XORD(N), YORD(N), WTORD(N), RSS
   CHARACTER*1   WEIGHT
```

3. Description

Given a set of observations (x_i, y_i) for $i = 1,2,...,n$, with corresponding weights w_i, G10ZAF rearranges the observations so that the x_i's are in ascending order.

For any equal x_i in the ordered set, say $x_j = x_{j+1} = ... = x_{j+k}$ a single observation x_j is returned with a corresponding y' and w', calculated as:

$$w' = \sum_{l=0}^{k} w_{i+l}$$

and

$$y' = \frac{\sum_{l=0}^{k} w_{i+l} y_{i+l}}{w'}.$$

Observations with zero weight are ignored. If no weights are supplied by the user, then unit weights are assumed; that is $w_i = 1$ for $i = 1,2,...,n$.

In addition, the within group sum of squares is computed for the tied observations using West's algorithm [2].

4. References

[1] DRAPER, N.R. and SMITH, H.
Applied Regression Analysis.
Wiley, 1966.

[2] WEST, D.H.D.
Updating mean and variance estimates: an improved method.
CACM, 22, pp. 532-555, 1979.

5. Parameters

1: WEIGHT – CHARACTER*1. *Input*

On entry: indicates whether user-defined weights are to be used;

If WEIGHT = 'W' user-defined weights are to be used and must be supplied in WT;
If WEIGHT = 'U' the data is treated as unweighted.

Constraint: WEIGHT = 'W' or 'U'.

2: N – INTEGER. *Input*

On entry: the number of observations, n.

Constraint: N ≥ 1.

3: X(N) – ***real*** array. *Input*

On entry: the values, x_i for $i = 1,2,...,n$.

4: Y(N) – ***real*** array. *Input*

On entry: the values, y_i for $i = 1,2,...,n$.

5: WT(*) – ***real*** array. *Input*

Note: the dimension of the array WT must be at least 1 if WEIGHT = 'U' and N if WEIGHT = 'W'.

On entry: if WEIGHT = 'W' then WT must contain n weights, w_i, for $i = 1,2,...,n$. If WEIGHT = 'U' then WT is not referenced.

Constraint: if WEIGHT = 'W' then WT(i) $\geq$ 0.0 for $i = 1,2,...,n$, and at least one WT(i) > 0.0 for some i.

6: NORD – INTEGER. *Output*

On exit: the number of distinct observations.

7: XORD(N) – ***real*** array. *Output*

On exit: the first NORD elements contain the ordered and distinct x_i.

8: YORD(N) – ***real*** array. *Output*

On exit: the first NORD elements contain the values y' corresponding to the values in XORD.

9: WTORD(N) – ***real*** array. *Output*

On exit: the first NORD elements contain the values w' corresponding to the values of XORD and YORD.

10: RSS – ***real***. *Output*

On exit: the within group sum of squares for tied observations.

11: IWRK(N) – INTEGER array. *Workspace*

12: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, WEIGHT $\neq$ 'W' or 'U',
or N < 1.

IFAIL = 2

On entry, WEIGHT = 'W' and at least one element of WT is < 0.0, or all elements of WT are 0.0.

7. Accuracy

For a discussion on the accuracy of the algorithm for computing mean and variance see West [2].

8. Further Comments

The routine may be used to compute the pure error sum of squares in simple linear regression along with G02DAF, see Draper and Smith [1].

9. Example

A set of unweighted observations are input and G10ZAF used to produce a set of strictly increasing weighted observations.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G10ZAF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX
      PARAMETER        (NMAX=100)
*     .. Local Scalars ..
      real             RSS
      INTEGER          I, IFAIL, N, NORD
      CHARACTER        WEIGHT
*     .. Local Arrays ..
      real             WT(NMAX), WTORD(NMAX), X(NMAX), XORD(NMAX),
     +                 Y(NMAX), YORD(NMAX)
      INTEGER          IWRK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         G10ZAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G10ZAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
         READ (NIN,*) WEIGHT
         DO 20 I = 1, N
            READ (NIN,*) X(I), Y(I)
   20    CONTINUE
         IFAIL = 0
*
         CALL G10ZAF(WEIGHT,N,X,Y,WT,NORD,XORD,YORD,WTORD,RSS,IWRK,
     +               IFAIL)
*
*        Print results
*
         WRITE (NOUT,*)
         WRITE (NOUT,99999) NORD
         WRITE (NOUT,99998) RSS
         WRITE (NOUT,*)
         WRITE (NOUT,99997)
         DO 40 I = 1, NORD
            WRITE (NOUT,99996) XORD(I), YORD(I), WTORD(I)
   40    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,'Number of distinct observations = ',I6)
99998 FORMAT (1X,'Residual sum of squares = ',F13.5)
99997 FORMAT (13X,'X                Y                WT')
99996 FORMAT (5X,F13.5,5X,F13.5,5X,F13.5)
      END
```

9.2. Program Data

```
G10ZAF Example Program Data
10
'U'
1.0 4.0
3.0 4.0
5.0 1.0
5.0 2.0
3.0 5.0
4.0 3.0
9.0 4.0
6.0 9.0
9.0 7.0
9.0 4.0
```

9.3. Program Results

```
 G10ZAF Example Program Results

 Number of distinct observations =      6
 Residual sum of squares =       7.00000

             X                  Y                  WT
          1.00000            4.00000            1.00000
          3.00000            4.50000            2.00000
          4.00000            3.00000            1.00000
          5.00000            1.50000            2.00000
          6.00000            9.00000            1.00000
          9.00000            5.00000            3.00000
```

Chapter G11 – Contingency Table Analysis

Note. Please refer to the Users' Note for your implementation to check that a routine is available.

Routine Name	Mark of Introduction	Purpose
G11AAF	16	χ^2 statistics for two-way contingency table
G11BAF	17	Computes multiway table from set of classification factors using selected statistic
G11BBF	17	Computes multiway table from set of classification factors using given percentile/quantile
G11BCF	17	Computes marginal tables for multiway table computed by G11BAF or G11BBF
G11CAF	19	Returns parameter estimates for the conditional analysis of stratified data
G11SAF	12	Contingency table, latent variable model for binary data
G11SBF	12	Frequency count for G11SAF

Chapter G11

Contingency Table Analysis

Contents

1 Scope of the Chapter

The routines in this chapter are for the analysis of discrete multivariate data. One suite of routines computes tables while other routines are for the analysis of two-way contingency tables, conditional logistic models and one-factor analysis of binary data.

Routines in Chapter G02 may be used to fit generalized linear models to discrete data including binary data and contingency tables.

2 Background to the Problems

2.1 Discrete Data

Discrete variables can be defined as variables which take a limited range of values. Discrete data can be usefully categorized into three types.

Binary data. The variables can take one of two values, for example, yes or no. The data may be grouped, for example, the number of yes responses in ten questions.

Categorical data. The variables can take one of two or more values or levels, but the values are not considered to have any ordering, for example, the values may be red, green, blue or brown.

Ordered categorical data. This is similar to categorical data but an ordering can be placed on the levels, for example: poor, average or good.

Data containing discrete variables can be analysed by computing summaries and measures of association and by fitting models.

2.2 Tabulation

The basic summary for multivariate discrete data is the multidimensional table in which each dimension is specified by a discrete variable. If the cells of the table are the number of observations with the corresponding values of the discrete variables then it is a contingency table. The discrete variables that can be used to classify a table are known as factors. For example, the factor sex would have the levels male and female. These can be coded as 1 and 2 respectively. Given several factors a multi-way table can be constructed such that each cell of the table represents one level from each factor. For example, a sample of 120 observations with the two factors sex and habitat, habitat having three levels: inner-city, suburban and rural, would give the 2×3 contingency table:

Sex	**Habitat**		
	Inner-city	Suburban	Rural
Male	32	27	15
Female	21	19	6

If the sample also contains continuous variables such as age, the average for the observations in each cell could be computed,

Sex	**Habitat**		
	Inner-city	Suburban	Rural
Male	25.5	30.3	35.6
Female	23.2	29.1	30.4

or other summary statistics.

Given a table, the totals or means for rows, columns etc. may be required. Thus the above contingency table with marginal totals is:

Sex	**Habitat**			
	Inner-city	Suburban	Rural	Total
Male	32	27	15	74
Female	21	19	6	46
Total	53	46	21	120

Note that the marginal totals for columns is itself a 2×1 table. Also, other summary statistics could be used to produce the marginal tables such as means or medians. Having computed the marginal tables, the cells of the original table may be expressed in terms of the margins, for example, in the above table the cells could be expressed as percentages of the column totals.

2.3 Discrete Response Variables and Logistic Regression

A second important categorization in addition to that given in Section 2.1 is whether one of the discrete variables can be considered as a response variable or whether it is just the association between the discrete variables that is being considered.

If the response variable is binary, for example, success or failure, then a logistic or probit regression model can be used. The logistic regression model relates the logarithm of the odds-ratio to a linear model. So if p_i is the probability of success, the model relates $\log(p_i/(1-p_i))$ to the explanatory variables. If the responses are independent then these models are special cases of the generalized linear model with binomial errors. However, there are cases when the binomial model is not suitable. For example, in a case-control study a number of cases (successes) and number of controls (failures) is chosen for a number of sets of case-controls. In this situation a conditional logistic analysis is required.

Handling a categorical or ordered categorical response variable is more complex, for a discussion on the appropriate models see McCullagh and Nelder [7]. These models generally use a Poisson distribution.

Note that if the response variable is a continuous variable and it is only the explanatory variables that are discrete then the regression models described in Chapter G02 should be used.

2.4 Contingency Tables

If there is no response variable then to investigate the association between discrete variables a contingency table can be computed and a suitable test performed on the table. The simplest case is the two-way table formed when considering two discrete variables. For a data set of n observations classified by the two variables with r and c levels respectively, a two-way table of frequencies or counts with r rows and c columns can be computed.

n_{11}	n_{12}	$\dots$	n_{1c}	$n_{1.}$
n_{21}	n_{22}	$\dots$	n_{2c}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n_{r1}	n_{r2}	$\dots$	n_{rc}	$n_{r.}$
$n_{.1}$	$n_{.2}$	$\dots$	$n_{.c}$	n

If p_{ij} is the probability of an observation in cell ij then the the model which assumes no association between the two variables is the model

$$p_{ij} = p_{i.}p_{.j}$$

where $p_{i.}$ is the marginal probability for the row variable and $p_{.j}$ is the marginal probability for the column variable, the marginal probability being the probability of observing a particular value of the variable ignoring all other variables. The appropriateness of this model can be assessed by two commonly used statistics:

the Pearson χ^2 statistic

$$\sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(n_{ij}-f_{ij})^2}{f_{ij}},$$

and the likelihood ratio test statistic

$$2\sum_{i=1}^{r}\sum_{j=1}^{c} n_{ij} \times \log(n_{ij}/f_{ij}).$$

The f_{ij} are the fitted values from the model; these values are the expected cell frequencies and are given by

$$f_{ij} = n\hat{p}_{ij} = n\hat{p}_{i.}\hat{p}_{.j} = n(n_{i.}/n)(n_{.j}/n) = n_{i.}n_{.j}/n.$$

Under the hypothesis of no association between the two classification variables, both these statistics have, approximately, a χ^2 distribution with $(c-1)(r-1)$ degrees of freedom. This distribution is arrived at under the assumption that the expected cell frequencies, f_{ij}, are not too small.

In the case of the 2×2 table, i.e., $c = 2$ and $r = 2$, the χ^2 approximation can be improved by using Yates's continuity correction factor. This decreases the absolute value of $(n_{ij} - f_{ij})$ by $1/2$. For 2×2 tables with a small values of n the exact probabilities can be computed; this is known as Fisher's exact test.

An alternative approach, which can easily be generalized to more than two variables, is to use log-linear models. A log-linear model for two variables can be written as

$$\log(p_{ij}) = \log(p_{i.}) + \log(p_{.j}).$$

A model like this can be fitted as a generalized linear model with Poisson error with the cell counts, n_{ij}, as the response variable.

2.5 Latent Variable Models

Latent variable models play an important role in the analysis of multivariate data. They have arisen in response to practical needs in many sciences, especially in psychology, educational testing and other social sciences.

Large-scale statistical enquiries, such as social surveys, generate much more information than can be easily absorbed without condensation. Elementary statistical methods help to summarize the data by looking at individual variables or the relationship between a small number of variables. However, with many variables it may still be difficult to see any pattern of inter-relationships. Our ability to visualize relationships is limited to two or three dimensions putting us under strong pressure to reduce the dimensionality of the data and yet preserve as much of the structure as possible. The question is thus one of how to replace the many variables with which we start by a much smaller number, with as little loss of information as possible.

One approach to the problem is to set up a model in which the dependence between the observed variables is accounted for by one or more latent variables. Such a model links the large number of observable variables with a much smaller number of latent variables.

Factor analysis, as described in Chapter G03, is based on a linear model of this kind when the observed variables are continuous. Here we consider the case where the observed variables are binary (e.g., coded 0/1 or true/false) and where there is one latent variable. In educational testing this is known as latent trait analysis, but, more generally, as factor analysis of binary data.

A variety of methods and models have been proposed for this problem. The models used here are derived from the general approach of Bartholomew [1] and [2]. The user is referred to [1] for further information on the models and to [3] for details of the method and application.

3 Recommendations on Choice and Use of Available Routines

3.1 Tabulation

The following routines can be used to perform the tabulation of discrete data.

G11BAF	computes a multidimensional table from a set of discrete variables or classification factors. The cells of the table may be counts or a summary statistic {total, mean, variance, largest or smallest} computed for an associated continuous variable. Alternatively, G11BAF will update an existing table with further data.
G11BBF	computes a multidimensional table from a set of discrete variables or classification factor where the cells are the percentile or quantile for an associated variable. For example, G11BBF can be used to produce a table of medians.
G11BCF	computes a marginal table from a table computed by G11BAF or G11BBF using a summary statistic {total, mean, median variance, largest or smallest}.

3.2 Analysis of Contingency Tables

G11AAF	computes the Pearson and likelihood ratio χ^2 statistics for a two-way contingency table. For 2×2 tables Yates's correction factor is used and for small samples, $n \leq 40$, Fisher's exact test is used.

In addition, G02GCF can be used to fit a log-linear model to a contingency table.

3.3 Binary data

The following routines can be used to analyse binary data.

G11SAF	fits a latent variable model to binary data. The frequency distribution of score patterns is required as input data. If the user's data is in the form of individual score patterns, then the service routine G11SBF may be used to calculate the frequency distribution.
G11CAF	estimates the parameters for a conditional logistic model.

In addition, G02GBF fits generalized linear models to binary data.

4 Routines Withdrawn or Scheduled for Withdrawal

None since Mark 13.

5 References

[1] Bartholomew D J (1980) Factor analysis for categorical data (with Discussion) *J. Roy. Statist. Soc. Ser. B* **42** 293–321

[2] Bartholomew D J (1984) The foundations of factor analysis *Biometrika* **71** 221–232

[3] Bartholomew D J (1987) *Latent Variable Models and Factor Analysis* Griffin

[4] Everitt B S (1977) *The Analysis of Contingency Tables* Chapman and Hall

[5] Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Volume 1)* Griffin (3rd Edition)

[6] Kendall M G and Stuart A (1973) *The Advanced Theory of Statistics (Volume 2)* Griffin (3rd Edition)

[7] McCullagh P and Nelder J A (1983) *Generalized Linear Models* Chapman and Hall

G11AAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G11AAF computes χ^2 statistics for a two-way contingency table. For a 2×2 table with a small number of observations exact probabilities are computed.

2. Specification

```
      SUBROUTINE G11AAF (NROW, NCOL, NOBST, LDT, EXPT, CHIST, PROB, CHI, G,
     1                   DF, IFAIL)
      INTEGER            NROW, NCOL, NOBST(LDT,NCOL), LDT, IFAIL
      real               EXPT(LDT,NCOL), CHIST(LDT,NCOL), PROB, CHI, G, DF
```

3. Description

For a set of n observations classified by two variables, with r and c levels respectively, a two-way table of frequencies with r rows and c columns can be computed.

$$\begin{array}{cccc|c}
n_{11} & n_{12} & \cdots & n_{1c} & n_{1.} \\
n_{21} & n_{22} & \cdots & n_{2c} & n_{2.} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
n_{r1} & n_{r2} & \cdots & n_{rc} & n_{r.} \\
\hline
n_{.1} & n_{.2} & \cdots & n_{.c} & n
\end{array}$$

To measure the association between the two classification variables two statistics that can be used are:

$$\text{The Pearson } \chi^2 \text{ statistic} = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(n_{ij}-f_{ij})^2}{f_{ij}},$$

and

$$\text{The likelihood ratio test statistic} = 2\sum_{i=1}^{r}\sum_{j=1}^{c}n_{ij}\times\log(n_{ij}/f_{ij}).$$

Where f_{ij} are the fitted values from the model that assumes the effects due to the classification variables are additive, i.e. there is no association. These values are the expected cell frequencies and are given by,

$$f_{ij} = n_{i.}n_{.j}/n.$$

Under the hypothesis of no association between the two classification variables, both these statistics have, approximately, a χ^2 distribution with $(c-1)(r-1)$ degrees of freedom. This distribution is arrived at under the assumption that the expected cell frequencies, f_{ij}, are not too small. For a discussion of this point see Everitt [1]. He concludes by saying, "... in the majority of cases the chi-square criterion may be used for tables with expectations in excess of 0.5 in the smallest cell".

In the case of the 2×2 table, i.e. $c = 2$ and $r = 2$, the χ^2 approximation can be improved by using Yates' continuity correction factor. This decreases the absolute value of $(n_{ij}-f_{ij})$ by $\frac{1}{2}$. For 2×2 tables with a small values of n the exact probabilities from Fisher's test are computed. These are based on the hypergeometric distribution and are computed using G01BLF. A two-tail probability is computed as $\min(1,2p_u,2p_l)$, where p_u and p_l are the upper and lower one-tail probabilities from the hypergeometric distribution.

4. References

[1] EVERITT, B.S.
The Analysis of Contingency Tables.
Chapman and Hall, 1977.

[2] KENDALL, M.G. and STUART, A.
The Advanced Theory of Statistics, Volume 2.
Griffin, 1973.

5. Parameters

1: NROW – INTEGER. *Input*

On entry: the number of rows in the contingency table, r.

Constraint: NROW $\geq$ 2.

2: NCOL – INTEGER. *Input*

On entry: the number of columns in the contingency table, c.

Constraint: NCOL $\geq$ 2.

3: NOBST(LDT,NCOL) – INTEGER array. *Input*

On entry: the contingency table, NOBST(i,j) must contain n_{ij} for $i = 1,2,...,r; j = 1,2,...,c$.

Constraint: NOBST$(i,j) \geq 0$ for $i = 1,2,...,r; j = 1,2,...,c$.

4: LDT – INTEGER. *Input*

On entry: the first dimension of the arrays NOBST, EXPT and CHIST as declared in the (sub)program from which G11AAF is called.

Constraint: LDT $\geq$ NROW.

5: EXPT(LDT,NCOL) – ***real*** array. *Output*

On exit: the table of expected values, EXPT(i,j) contains f_{ij} for $i = 1,2,...,r; j = 1,2,...,c$.

6: CHIST(LDT,NCOL) – ***real*** array. *Output*

On exit: the table of χ^2 contributions, CHIST(i,j) contains $\frac{(n_{ij}-f_{ij})^2}{f_{ij}}$ for $i = 1,2,...,r$; $j = 1,2,...,c$.

7: PROB – ***real.*** *Output*

On exit: if $c = 2$, $r = 2$ and $n \leq 40$ then PROB contains the two-tail significance level for Fisher's exact test, otherwise PROB contains the significance level from the Pearson χ^2 statistic.

8: CHI – ***real.*** *Output*

On exit: the Pearson χ^2 statistic.

9: G – ***real.*** *Output*

On exit: the likelihood ratio test statistic.

10: DF – ***real.*** *Output*

On exit: the degrees of freedom for the statistics.

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL $\neq$ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, NROW < 2,
or NCOL < 2,
or LDT < NROW.

IFAIL = 2

On entry, a value in NOBST < 0, or all values in NOBST are zero.

IFAIL = 3

On entry, a 2×2 table has a row or column with both values 0.

IFAIL = 4

At least one cell has expected frequency, f_{ij}, $\leq$ 0.5. The χ^2 approximation may be poor.

7. Accuracy

For the accuracy of the probabilities for Fisher's exact test see G01BLF.

8. Further Comments

The routine G01AFF allows for the automatic amalgamation of rows and columns. In most circumstances this is not recommended, see Everitt [1].

Multi-dimensional contingency tables can be analysed using log-linear models fitted by G02GBF.

9. Example

The data below, taken from Everitt [1], is from 141 patients with brain tumours. The row classification variable is the site of the tumour: frontal lobes, temporal lobes and other cerebal areas. The column classification variable is the type of tumour: benign, malignant and other cerebal tumours.

23	9	6	38
21	4	3	28
34	24	17	75
78	37	26	141

The data is read in and the statistics computed and printed.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G11AAF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          CMAX, RMAX
      PARAMETER        (CMAX=3,RMAX=3)
*     .. Local Scalars ..
      real             CHI, DF, G, PROB
      INTEGER          I, IFAIL, J, NCOL, NROW
*     .. Local Arrays ..
      real             CHIST(RMAX,CMAX), EXPT(RMAX,CMAX)
      INTEGER          NOBST(RMAX,CMAX)
*     .. External Subroutines ..
      EXTERNAL         G11AAF
*     .. Executable Statements ..
      WRITE (NOUT,*) ' G11AAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NROW, NCOL
      IF (NROW.LE.RMAX .AND. NCOL.LE.CMAX) THEN
         DO 20 I = 1, NROW
            READ (NIN,*) (NOBST(I,J),J=1,NCOL)
   20    CONTINUE
         IFAIL = -1
*
         CALL G11AAF(NROW,NCOL,NOBST,RMAX,EXPT,CHIST,PROB,CHI,G,DF,
     +               IFAIL)
*
         IF (IFAIL.EQ.0 .OR. IFAIL.EQ.3) THEN
            WRITE (NOUT,*)
            WRITE (NOUT,99999) ' Probability = ', PROB
            WRITE (NOUT,99998) ' Pearson Chi-square statistic = ', CHI
            WRITE (NOUT,99998) ' Likelihood ratio test statistic = ', G
            WRITE (NOUT,99997) ' Degrees of freedom = ', DF
         END IF
      END IF
      STOP
*
99999 FORMAT (A,F6.4)
99998 FORMAT (A,F8.3)
99997 FORMAT (A,F4.0)
      END
```

9.2. Program Data

```
G11AAF Example Program Data
3 3                          : NROW NCOL
23  9  6                     : NOBST
21  4  3
34 24 17
```

9.3. Program Results

```
 G11AAF Example Program Results

Probability = 0.0975
Pearson Chi-square statistic =    7.844
Likelihood ratio test statistic =    8.096
Degrees of freedom =   4.
```

G11BAF – NAG Fortran Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G11BAF computes a table from a set of classification factors using a selected statistic.

2 Specification

```
      SUBROUTINE G11BAF(STAT, UPDATE, WEIGHT, N, NFAC, ISF, LFAC, IFAC,
     1                  LDF, Y, WT, TABLE, MAXT, NCELLS, NDIM, IDIM,
     2                  ICOUNT, AUXT, IWK, IFAIL)
      INTEGER           N, NFAC, ISF(NFAC), LFAC(NFAC), IFAC(LDF,NFAC),
     1                  LDF, MAXT, NCELLS, NDIM, IDIM(NFAC),
     2                  ICOUNT(MAXT), IWK(2*NFAC), IFAIL
      real              Y(N), WT(*), TABLE(MAXT), AUXT(*)
      CHARACTER*1       STAT, UPDATE, WEIGHT
```

3 Description

A data set may include both classification variables and general variables. The classification variables, known as factors, take a small number of values known as levels. For example, the factor sex would have the levels male and female. These can be coded as 1 and 2 respectively. Given several factors, a multi-way table can be constructed such that each cell of the table represents one level from each factor. For example, the two factors sex and habitat, habitat having three levels: inner-city, suburban and rural, define the 2 × 3 contingency table:

Sex	**Habitat**		
	Inner-city	Suburban	Rural
Male			
Female			

For each cell statistics can be computed. If a third variable in the data set was age, then for each cell the average age could be computed:

Sex	**Habitat**		
	Inner-city	Suburban	Rural
Male	25.5	30.3	35.6
Female	23.2	29.1	30.4

That is the average age for all observations for males living in rural areas is 35.6. Other statistics can also be computed: the number of observations, the total, the variance, the largest value and the smallest value.

G11BAF computes a table for one of the selected statistics. The factors have to be coded with levels 1,2,... Weights can be used to eliminate values from the calculations, e.g. if they represent 'missing values'. There is also the facility to update an existing table with the addition of new observations.

4 References

[1] West D H D (1979) Updating mean and variance estimates: An improved method *Comm. ACM* **22** 532–535

[2] John J A and Quenouille M H (1977) *Experiments: Design and Analysis* Griffin

[3] Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Volume 1)* Griffin (3rd Edition)

5 Parameters

1: STAT — CHARACTER*1 *Input*

On entry: indicates which statistic is to be computed for the table cells.

If STAT = 'N', the number of observations for each cell.
If STAT = 'T', the total for the variable in Y for each cell.
If STAT = 'A', the average (mean) for the variable in Y for each cell.
If STAT = 'V', the variance for the variable in Y for each cell.
If STAT = 'L', the largest value for the variable in Y for each cell.
If STAT = 'S', the smallest value for the variable in Y for each cell.

Constraint: STAT = 'N', 'T', 'A', 'V', 'L' or 'S'.

2: UPDATE — CHARACTER*1 *Input*

On entry: indicates if an existing table is to be updated by further observation.

If UPDATE = 'I' the table cells will be initialised to zero before tabulations take place.

If UPDATE = 'U' the table input in TABLE will be updated. The parameters NCELLS, TABLE, ICOUNT and AUXT must remain unchanged from the previous call to G11BAF.

Constraint: UPDATE = 'I' or 'U'.

3: WEIGHT — CHARACTER*1 *Input*

On entry: indicates if weights are to be used.

If WEIGHT = 'U' weights are not used and unit weights are assumed.

If WEIGHT = 'W' or 'V' weights are used and must be supplied in WT. The only difference between WEIGHT = 'W' and WEIGHT = 'V' is if the variance is computed. If WEIGHT = 'W' the divisor for the variance is the sum of the weights minus one and if WEIGHT = 'V' the divisor is the number of observations with non-zero weights minus one. The former is useful if the weights represent the frequency of the observed values. If STAT = 'T' or 'A' the weighted total or mean is computed respectively, if STAT = 'N', 'L' or 'S' the only effect of weights is to eliminate values with zero weights from the computations.

Constraint: WEIGHT = 'U', 'V' or 'W'.

4: N — INTEGER *Input*

On entry: the number of observations.

Constraint: N $\geq$ 2.

5: NFAC — INTEGER *Input*

On entry: the number of classifying factors in IFAC.

Constraint: NFAC $\geq$ 1.

6: ISF(NFAC) — INTEGER array *Input*

On entry: indicates which factors in IFAC are to be used in the tabulation.

If ISF(i) > 0 the ith factor in IFAC is included in the tabulation.

Note that if ISF(i) $\leq$ 0 for i = 1,2,...,NFAC then the statistic for the whole sample is calculated and returned in a 1 $\times$ 1 table.

7: LFAC(NFAC) — INTEGER array *Input*

On entry: the number of levels of the classifying factors in IFAC.

Constraint: if ISF(i) > 0, LFAC(i) $\geq$ 2 for i = 1,2,...,NFAC.

8: IFAC(LDF,NFAC) — INTEGER array *Input*

On entry: the NFAC coded classification factors for the N observations.

Constraint: $1 \leq \text{IFAC}(i,j) \leq \text{LFAC}(j)$ for $i = 1,2,\ldots,\text{N}$; $j = 1,2,\ldots,\text{NFAC}$.

9: LDF — INTEGER *Input*

On entry: the first dimension of the array IFAC as declared in the (sub)program from which G11BAF is called.

Constraint: LDF $\geq$ N.

10: Y(N) — ***real*** array *Input*

On entry: the variable to be tabulated. If STAT = 'N', Y is not referenced.

11: WT(*) — ***real*** array *Input*

Note: the dimension of the array WT must be at least N if WEIGHT = 'W' or 'V' and 1 otherwise.

On entry: if WEIGHT = 'W' or 'V', WT must contain the N weights. Otherwise WT is not referenced.

Constraint: if WEIGHT = 'W' or 'V', WT(i) $\geq$ 0.0 for i = 1,2,...,N.

12: TABLE(MAXT) — ***real*** array *Input/Output*

On entry: if UPDATE = 'U', TABLE must be unchanged from the previous call to G11BAF, otherwise TABLE need not be set.

On exit: the computed table. The NCELLS cells of the table are stored so that for any two factors the index relating to the factor referred to later in LFAC and IFAC changes faster. For further details see Section 8.

13: MAXT — INTEGER *Input*

On entry: the maximum size of the table to be computed.

Constraint: MAXT $\geq$ product of the levels of the factors included in the tabulation.

14: NCELLS — INTEGER *Input/Output*

On entry: if UPDATE = 'U', NCELLS must be unchanged from the previous call to G11BAF, otherwise NCELLS need not be set.

On exit: the number of cells in the table.

15: NDIM — INTEGER *Output*

On exit: the number of factors defining the table.

16: IDIM(NFAC) — INTEGER array *Output*

On exit: the first NDIM elements contain the number of levels for the factors defining the table.

17: ICOUNT(MAXT) — INTEGER array *Input/Output*

On entry: if UPDATE = 'U', ICOUNT must be unchanged from the previous call to G11BAF, otherwise ICOUNT need not be set.

On exit: a table containing the number of observations contributing to each cell of the table, stored identically to TABLE. Note if STAT = 'N' this is the same as is returned in TABLE.

18: AUXT(*) — ***real*** array *Input/Output*

Note: the dimension of the array AUXT must be at least NCELLS if STAT = 'A', at least 2×NCELLS if STAT = 'V' and 1 otherwise .

On entry: if UPDATE = 'U', AUXT must be unchanged from the previous call to G11BAF, otherwise AUXT need not be set.

On exit: if STAT = 'A' or 'V' the first NCELLS values hold the table containing the sum of the weights for the observations contributing to each cell, stored identically to TABLE. If STAT = 'V' then the second set of NCELLS values hold the table of cell means. Otherwise AUXT is not referenced.

19: IWK(2*NFAC) — INTEGER array *Workspace*

20: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N < 2,
or NFAC < 1,
or LDF < N,
or UPDATE ≠ 'I' or 'U',
or WEIGHT ≠ 'U', 'W' or 'V',
or STAT ≠ 'N', 'T', 'A', 'V', 'L' or 'S'.

IFAIL = 2

On entry, ISF(i) > 0 and LFAC(i) < 2 for some i,
or IFAC(i,j) < 1 for some i,j,
or IFAC(i,j) > LFAC(j) for some i,j,
or MAXT is too small,
or WEIGHT = 'W' or 'V' and WT(i) < 0.0 for some i.

IFAIL = 3

STAT = 'V' and the divisor for the variance is ≤ 0.0.

IFAIL = 4

UPDATE = 'U' and at least one of NCELLS, TABLE, AUXT or ICOUNT have been changed since previous call to G11BAF.

7 Accuracy

Only applicable when STAT = 'V'. In this case a one pass algorithm is used as described by West [1].

8 Further Comments

The tables created by G11BAF and stored in TABLE, ICOUNT and, depending on STAT, also in AUXT are stored in the following way. Let there be n factors defining the table with factor k having l_k levels, then the cell defined by the levels $i_1, i_2, \ldots, i_n$ of the factors is stored in mth cell given by:

$$m = 1 + \sum_{k=1}^{n} \{(i_k - 1)c_k\},$$

where $c_j = \prod_{k=j+1}^{n} l_k$ for $j = 1, 2, \ldots, n-1$ and $c_n = 1$.

9 Example

The data, given by John and Quenouille [2], is for a 3×6 factorial experiment in 3 blocks of 18 units. The data is input in the order: blocks, factor with 3 levels, factor with 6 levels, yield. The 3×6 table of treatment means for yield over blocks is computed and printed.

9.1 Example Text

Note: the listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G11BAF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX, LTMAX
      PARAMETER        (NMAX=54,MMAX=3,LTMAX=18)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, K, LDF, MAXT, N, NCELLS, NCOL, NDIM,
     +                 NFAC, NROW
      CHARACTER        STAT, WEIGHT
*     .. Local Arrays ..
      real             AUXT(2*LTMAX), TABLE(LTMAX), WT(NMAX), Y(NMAX)
      INTEGER          ICOUNT(LTMAX), IDIM(MMAX), IFAC(NMAX,MMAX),
     +                 ISF(MMAX), IWK(2*MMAX), LFAC(MMAX)
*     .. External Subroutines ..
      EXTERNAL         G11BAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G11BAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) STAT, WEIGHT, N, NFAC
      IF (N.LE.NMAX .AND. NFAC.LE.MMAX) THEN
         IF (WEIGHT.EQ.'W' .OR. WEIGHT.EQ.'w' .OR. WEIGHT.EQ.'V' .OR.
     +       WEIGHT.EQ.'v') THEN
            DO 20 I = 1, N
               READ (NIN,*) (IFAC(I,J),J=1,NFAC), Y(I), WT(I)
   20       CONTINUE
         ELSE
            DO 40 I = 1, N
               READ (NIN,*) (IFAC(I,J),J=1,NFAC), Y(I)
   40       CONTINUE
         END IF
         READ (NIN,*) (LFAC(J),J=1,NFAC)
         READ (NIN,*) (ISF(J),J=1,NFAC)
         LDF = NMAX
         MAXT = LTMAX
         IFAIL = 0
*
         CALL G11BAF(STAT,'I',WEIGHT,N,NFAC,ISF,LFAC,IFAC,LDF,Y,WT,
     +               TABLE,MAXT,NCELLS,NDIM,IDIM,ICOUNT,AUXT,IWK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' TABLE'
         WRITE (NOUT,*)
         NCOL = IDIM(NDIM)
         NROW = NCELLS/NCOL
         K = 1
         DO 60 I = 1, NROW
            WRITE (NOUT,99999) (TABLE(J),'(',ICOUNT(J),')',J=K,K+NCOL-1)
            K = K + NCOL
   60    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,6(F8.2,A,I2,A))
      END
```

9.2 Example Data

```
G11BAF Example Program Data

'A' 'U' 54 3

1 1 1 274
1 2 1 361
1 3 1 253
1 1 2 325
1 2 2 317
1 3 2 339
1 1 3 326
1 2 3 402
1 3 3 336
1 1 4 379
1 2 4 345
1 3 4 361
1 1 5 352
1 2 5 334
1 3 5 318
1 1 6 339
1 2 6 393
1 3 6 358
2 1 1 350
2 2 1 340
2 3 1 203
2 1 2 397
2 2 2 356
2 3 2 298
2 1 3 382
2 2 3 376
2 3 3 355
2 1 4 418
```

```
2 2 4 387
2 3 4 379
2 1 5 432
2 2 5 339
2 3 5 293
2 1 6 322
2 2 6 417
2 3 6 342
3 1 1  82
3 2 1 297
3 3 1 133
3 1 2 306
3 2 2 352
3 3 2 361
3 1 3 220
3 2 3 333
3 3 3 270
3 1 4 388
3 2 4 379
3 3 4 274
3 1 5 336
3 2 5 307
3 3 5 266
3 1 6 389
3 2 6 333
3 3 6 353

3 3 6
0 1 1
```

9.3 Example Results

```
G11BAF Example Program Results

 TABLE

  235.33( 3)  342.67( 3)  309.33( 3)  395.00( 3)  373.33( 3)  350.00( 3)
  332.67( 3)  341.67( 3)  370.33( 3)  370.33( 3)  326.67( 3)  381.00( 3)
  196.33( 3)  332.67( 3)  320.33( 3)  338.00( 3)  292.33( 3)  351.00( 3)
```

G11BBF – NAG Fortran Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G11BBF computes a table from a set of classification factors using a given percentile or quantile, for example the median.

2 Specification

```
SUBROUTINE G11BBF(TYPE, WEIGHT, N, NFAC, ISF, LFAC, IFAC, LDF,
1                 PERCNT, Y, WT, TABLE, MAXT, NCELLS, NDIM, IDIM,
2                 ICOUNT, IWK, WK, IFAIL)
 INTEGER          N, NFAC, ISF(NFAC), LFAC(NFAC), IFAC(LDF,NFAC),
1                 LDF, MAXT, NCELLS, NDIM, IDIM(NFAC),
2                 ICOUNT(MAXT), IWK(2*NFAC+N), IFAIL
 real             PERCNT, Y(N), WT(*), TABLE(MAXT), WK(2*N)
 CHARACTER*1      TYPE, WEIGHT
```

3 Description

A data set may include both classifcation variables and general variables. The classification variables, known as factors, take a small number of values known as levels. For example, the factor sex would have the levels male and female. These can be coded as 1 and 2 respectively. Given several factors, a multi-way table can be constructed such that each cell of the table represents one level from each factor. For example, the two factors sex and habitat, habitat having three levels: inner-city, suburban and rural, define the 2×3 contingency table:

Sex	**Habitat**		
	Inner-city	Suburban	Rural
Male			
Female			

For each cell statistics can be computed. If a third variable in the data set was age then for each cell the median age could be computed:

Sex	**Habitat**		
	Inner-city	Suburban	Rural
Male	24	31	37
Female	21.5	28.5	33

That is the median age for all observations for males living in rural areas is 37. The median being the 50% quantile. Other quantiles can also be computed: the p percent quantile or percentile, q_p, is the estimate of the value such that p percent of observations are less than q_p. This is calculated in two different ways depending on whether the tabulated variable is continuous or discrete. Let there be m values in a cell and let $y_{(1)}, y_{(2)}, \ldots, y_{(m)}$ be the values for that cell sorted into ascending order. Also, associated with each value there is a weight, $w_{(1)}, w_{(2)}, \ldots, w_{(m)}$, which could represent the observed frequency for that value, with $W_j = \sum_{i=1}^{j} w_{(i)}$ and $W'_j = \sum_{i=1}^{j} w_{(i)} - \frac{1}{2} w_{(j)}$. For the p percentile let $p_w = (p/100)W_m$ and $p'_w = (p/100)W'_m$ then the percentiles for the two cases are as given below.

If the variable is discrete, that is takes only a limited number of (usually integer) values then the percentile is defined as:

$$y_{(j)} \qquad \text{if } W_{j-1} < p_W < W_j$$

$$\frac{y_{(j+1)} + y_{(j)}}{2} \qquad \text{if } p_w = W_j$$

If the data is continuous then the quantiles are estimated by linear interpolation.

$$\begin{array}{ll} y_{(1)} & \text{if } p'_w \leq W'_1 \\ (1-f)y_{(j-1)} + fy_{(j)} & \text{if } W'_{j-1} < p'_w \leq W'_j \\ y_{(m)} & \text{if } p'_w > W'_m \end{array}$$

where $f = (p'_w - W'_{j-1})/(W'_j - W'_{j-1})$.

4 References

[1] John J A and Quenouille M H (1977) *Experiments: Design and Analysis* Griffin

[2] Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Volume 1)* Griffin (3rd Edition)

5 Parameters

1: TYPE — CHARACTER*1 *Input*

On entry: indicates if the variable to be tabulated is discrete or continuous.

If TYPE = 'D', the percentiles are computed for a discrete variable.

If TYPE = 'C', the percentiles are computed for a continuous variable using linear interpolation.

Constraint: TYPE = 'D' or 'C'.

2: WEIGHT — CHARACTER*1 *Input*

On entry: indicates if there are weights associated with the variable to be tabulated.

If WEIGHT = 'U', weights are not input and unit weights are assumed.

If WEIGHT = 'W', weights must be supplied in WT.

Constraint: WEIGHT = 'U' or 'W'.

3: N — INTEGER *Input*

On entry: the number of observations.

Constraint: N $\geq$ 2.

4: NFAC — INTEGER *Input*

On entry: the number of classifying factors in IFAC.

Constraint: NFAC $\geq$ 1.

5: ISF(NFAC) — INTEGER array *Input*

On entry: indicates which factors in IFAC are to be used in the tabulation.

If ISF(i) > 0 the ith factor in IFAC is included in the tabulation.

Note that if ISF(i) $\leq$ 0 for $i = 1, 2, \ldots,$ NFAC then the statistic for the whole sample is calculated and returned in a 1×1 table.

6: LFAC(NFAC) — INTEGER array *Input*

On entry: the number of levels of the classifying factors in IFAC.

Constraint: if ISF(i) > 0, LFAC(i) $\geq$ 2 for $i = 1, 2, \ldots,$ NFAC.

7: IFAC(LDF,NFAC) — INTEGER array *Input*

On entry: the NFAC coded classification factors for the N observations.

Constraint: $1 \leq \text{IFAC}(i,j) \leq \text{LFAC}(j)$ for $i = 1,2,\ldots,\text{N}$; $j = 1,2,\ldots,\text{NFAC}$.

8: LDF — INTEGER *Input*

On entry: the first dimension of the array IFAC as declared in the (sub)program from which G11BBF is called.

Constraint: LDF $\geq$ N.

9: PERCNT — ***real*** *Input*

On entry: the percentile to be tabulated, p.

Constraint: $0.0 < p < 100.0$.

10: Y(N) — ***real*** array *Input*

On entry: the variable to be tabulated.

11: WT(*) — ***real*** array *Input*

Note: the dimension of the array WT must be at least N if WEIGHT = 'W' and 1 otherwise.

On entry: if WEIGHT = 'W', WT must contain the N weights. Otherwise WT is not referenced.

Constraint: if WEIGHT = 'W', $\text{WT}(i) \geq 0.0$ for $i = 1,2,\ldots,\text{N}$.

12: TABLE(MAXT) — ***real*** array *Output*

On exit: the computed table. The NCELLS cells of the table are stored so that for any two factors the index relating to the factor occuring later in LFAC and IFAC changes faster. For further details see Section 8.

13: MAXT — INTEGER *Input*

On entry: the maximum size of the table to be computed.

Constraint: MAXT $\geq$ product of the levels of the factors included in the tabulation.

14: NCELLS — INTEGER *Output*

On exit: the number of cells in the table.

15: NDIM — INTEGER *Output*

On exit: the number of factors defining the table.

16: IDIM(NFAC) — INTEGER array *Output*

On exit: the first NDIM elements contain the number of levels for the factors defining the table.

17: ICOUNT(MAXT) — INTEGER array *Output*

On exit: a table containing the number of observations contributing to each cell of the table, stored identically to TABLE.

18: IWK(2*NFAC+N) — ***real*** array *Workspace*

19: WK(2*N) — ***real*** array *Workspace*

20: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N < 2,

or NFAC < 1,

or LDF < N,

or TYPE $\neq$ 'D' or 'C',

or WEIGHT $\neq$ 'U' or 'W',

or PERCNT $\leq$ 0.0,

or PERCNT $\geq$ 100.0.

IFAIL = 2

On entry, ISF(i) > 0 and LFAC(i) $\leq$ 1 for some i,

or IFAC(i,j) < 1 for some i,j,

or IFAC(i,j) > LFAC(j) for some i,j,

or MAXT is too small,

or WEIGHT = 'W' and WT(i) < 0.0 for some i.

IFAIL = 3

At least one cell is empty.

7 Accuracy

Not applicable.

8 Further Comments

The tables created by G11BBF and stored in TABLE and ICOUNT are stored in the following way. Let there be n factors defining the table with factor k having l_k levels, then the cell defined by the levels i_1, $i_2,\ldots,i_n$ of the factors is stored in mth cell given by:

$$m = 1 + \sum_{k=1}^{n} \{(i_k - 1)c_k\},$$

where $c_j = \prod_{k=j+1}^{n} l_k$ for $j = 1,2,\ldots,n-1$ and $c_n = 1$.

9 Example

The data, given by John and Quenouille [1], are for a 3×6 factorial experiment in 3 blocks of 18 units. The data is input in the order: blocks, factor with 3 levels, factor with 6 levels, yield, and the 3×6 table of treatment medians for yield over blocks is computed and printed.

9.1 Example Text

Note: the listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G11BBF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX, LTMAX
      PARAMETER        (NMAX=54,MMAX=3,LTMAX=18)
*     .. Local Scalars ..
      real             PERCNT
      INTEGER          I, IFAIL, J, K, LDF, MAXT, N, NCELLS, NCOL, NDIM,
     +                 NFAC, NROW
      CHARACTER        TYPE, WEIGHT
*     .. Local Arrays ..
      real             TABLE(LTMAX), WK(2*NMAX), WT(NMAX), Y(NMAX)
      INTEGER          ICOUNT(LTMAX), IDIM(MMAX), IFAC(NMAX,MMAX),
     +                 ISF(MMAX), IWK(2*MMAX+NMAX), LFAC(MMAX)
*     .. External Subroutines ..
      EXTERNAL         G11BBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G11BBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) TYPE, WEIGHT, N, NFAC, PERCNT
      IF (N.LE.NMAX .AND. NFAC.LE.MMAX) THEN
         IF (WEIGHT.EQ.'W' .OR. WEIGHT.EQ.'w' .OR. WEIGHT.EQ.'V' .OR.
     +       WEIGHT.EQ.'v') THEN
            DO 20 I = 1, N
               READ (NIN,*) (IFAC(I,J),J=1,NFAC), Y(I), WT(I)
   20       CONTINUE
         ELSE
            DO 40 I = 1, N
               READ (NIN,*) (IFAC(I,J),J=1,NFAC), Y(I)
   40       CONTINUE
         END IF
         READ (NIN,*) (LFAC(J),J=1,NFAC)
         READ (NIN,*) (ISF(J),J=1,NFAC)
         LDF = NMAX
         MAXT = LTMAX
         IFAIL = 0
*
         CALL G11BBF(TYPE,WEIGHT,N,NFAC,ISF,LFAC,IFAC,LDF,PERCNT,Y,WT,
     +               TABLE,MAXT,NCELLS,NDIM,IDIM,ICOUNT,IWK,WK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99999) ' TABLE for ', PERCNT, 'th percentile'
         WRITE (NOUT,*)
         NCOL = IDIM(NDIM)
         NROW = NCELLS/NCOL
         K = 1
         DO 60 I = 1, NROW
            WRITE (NOUT,99998) (TABLE(J),'(',ICOUNT(J),')',J=K,K+NCOL-1)
            K = K + NCOL
   60    CONTINUE
      END IF
      STOP
*
99999 FORMAT (A,F4.0,A)
99998 FORMAT (1X,6(F8.2,A,I2,A))
      END
```

9.2 Example Data

```
G11BBF Example Program Data

'C' 'U' 54 3 50.0

1 1 1 274
1 2 1 361
1 3 1 253
1 1 2 325
1 2 2 317
1 3 2 339
1 1 3 326
1 2 3 402
1 3 3 336
1 1 4 379
1 2 4 345
1 3 4 361
1 1 5 352
1 2 5 334
1 3 5 318
1 1 6 339
1 2 6 393
1 3 6 358
2 1 1 350
2 2 1 340
2 3 1 203
2 1 2 397
2 2 2 356
2 3 2 298
2 1 3 382
2 2 3 376
2 3 3 355
2 1 4 418
2 2 4 387
2 3 4 379
2 1 5 432
2 2 5 339
2 3 5 293
2 1 6 322
2 2 6 417
2 3 6 342
3 1 1  82
3 2 1 297
3 3 1 133
3 1 2 306
3 2 2 352
3 3 2 361
3 1 3 220
3 2 3 333
3 3 3 270
3 1 4 388
```

```
3 2 4 379
3 3 4 274
3 1 5 336
3 2 5 307
3 3 5 266
3 1 6 389
3 2 6 333
3 3 6 353

3 3 6
0 1 1
```

9.3 Example Results

```
G11BBF Example Program Results

TABLE for  50.th percentile

  226.00( 3)  320.25( 3)  299.50( 3)  385.75( 3)  348.00( 3)  334.75( 3)
  329.25( 3)  343.25( 3)  365.25( 3)  370.50( 3)  327.25( 3)  378.00( 3)
  185.50( 3)  328.75( 3)  319.50( 3)  339.25( 3)  286.25( 3)  350.25( 3)
```

G11BCF – NAG Fortran Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G11BCF computes a marginal table from a table computed by G11BAF or G11BBF using a selected statistic.

2 Specification

```
      SUBROUTINE G11BCF(STAT, TABLE, NCELLS, NDIM, IDIM, ISDIM, STABLE,
     1                  MAXST, MCELLS, MDIM, MLEVEL, AUXT, IWK, WK,
     2                  IFAIL)
      INTEGER           NCELLS, NDIM, IDIM(NDIM), ISDIM(NDIM), MAXST,
     1                  MCELLS, MDIM, MLEVEL(NDIM), IWK(3*NDIM), IFAIL
      real              TABLE(NCELLS), STABLE(MAXST), AUXT(*), WK(NCELLS)
      CHARACTER*1       STAT
```

3 Description

For a data set containing classification variables known as factors the routines G11BAF and G11BBF compute a table using selected statistics, for example the mean or the median. The table is indexed by the levels of the selected factors, for example if there were three factors A, B and C with 3, 2 and 4 levels respectively and the mean was to be tabulated the resulting table would be $3 \times 2 \times 4$ with each cell being the mean of all observations with the appropriate combination of levels of the three factors. In further analysis the table of means averaged over C for A and B may be required, this can be computed from the full table by taking the mean over the third dimension of the table, C.

In general, given a table computed by G11BAF or G11BBF, G11BCF computes a sub-table defined by a subset of the factors used to define the table such that each cell of the sub-table is the selected statistic computed over the remaining factors. The statistics that can be used are the total, the mean, the median, the variance, the smallest and the largest value.

4 References

[1] West D H D (1979) Updating mean and variance estimates: An improved method *Comm. ACM* **22** 532–535

[2] John J A and Quenouille M H (1977) *Experiments: Design and Analysis* Griffin

[3] Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Volume 1)* Griffin (3rd Edition)

5 Parameters

1: STAT — CHARACTER*1 *Input*

On entry: indicates which statistic is to be used to compute the marginal table.

If STAT = 'T' the total.
If STAT = 'A' the average or mean.
If STAT = 'M' the median.
If STAT = 'V' the variance.
If STAT = 'L' the largest value.
If STAT = 'S' the smallest value.

Constraint: STAT = 'T', 'A', 'M', 'V', 'L' or 'S'.

2: TABLE(NCELLS) — ***real*** array *Input*

On entry: the table as computed by G11BAF or G11BBF.

3: NCELLS — INTEGER *Input*

On entry: the number of cells in TABLE as returned by G11BAF or G11BBF.

4: NDIM — INTEGER *Input*

On entry: the number of dimensions for TABLE as returned by G11BAF or G11BBF.

Constraint: NDIM ≥ 2.

5: IDIM(NDIM) — INTEGER array *Input*

On entry: the number of levels for each dimension of TABLE as returned by G11BAF or G11BBF.

Constraint: IDIM(i) ≥ 2 for $i = 1, 2, \ldots,$ NDIM.

6: ISDIM(NDIM) — INTEGER array *Input*

On entry: indicates which dimensions of TABLE are to be included in the sub-table. If ISDIM(i) > 0 the dimension or factor indicated by IDIM(i) is to be included in the sub-table otherwise it is excluded.

7: STABLE(MAXST) — ***real*** array *Output*

On exit: the first MCELLS elements contain the sub-table computed using the statistic indicated by STAT. The table is stored in a similar way to TABLE with the MCELLS cells stored so that for any two dimensions the index relating to the dimension given later in IDIM changes faster. For further details see Section 8.

8: MAXST — INTEGER *Input*

On entry: the maximum size of sub-table to be computed.

Constraint: MAXST $\geq$ the product of the levels of the dimensions of TABLE included in the sub-table, STABLE.

9: MCELLS — INTEGER *Output*

On exit: the number of cells in the sub-table in STABLE.

10: MDIM — INTEGER *Output*

On exit: the number of dimensions to the sub-table in STABLE.

11: MLEVEL(NDIM) — INTEGER array *Output*

On exit: the first MDIM elements contain the number of levels for the dimensions of the sub-table in STABLE. The remaining elements are not referenced.

12: AUXT(*) — ***real*** array *Output*

Note: the dimension of the array AUXT must be at least MAXST if STAT = 'V' and 1 otherwise.

On exit: if STAT = 'V' AUXT contains the sub-table of means corresponding to the sub-table of variances in STABLE. Otherwise AUXT is not referenced.

13: IWK(3*NDIM) — INTEGER array *Workspace*
14: WK(NCELLS) — ***real*** array *Workspace*
15: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, NDIM < 2,
or STAT ≠ 'T', 'A', 'M', 'V', 'L' or 'S'.

IFAIL = 2

On entry, IDIM(i) ≤ 1, for some $i = 1, 2, \ldots,$ NDIM,
or NCELLS is incompatible with IDIM,
or the requested sub-table is of dimension 0,
or the requested sub-table is the full table,
or MAXST is too small, the minimum value is returned in MDIM.

7 Accuracy

Only applicable when STAT = 'V'. In this case a one pass algorithm is used as describe in West [1].

8 Further Comments

The sub-tables created by G11BCF and stored in STABLE and, depending on STAT, also in AUXT are stored in the following way. Let there be m dimensions defining the table with dimension k having l_k levels, then the cell defined by the levels $i_1, i_2, \ldots, i_m$ of the factors is stored in sth cell given by:

$$s = 1 + \sum_{k=1}^{m} [(i_k - 1)c_k]$$

where

$$c_j = \prod_{k=j+1}^{m} l_k \quad \text{for } j = 1, 2, \ldots, n-1 \text{ and } c_m = 1$$

9 Example

The data, given by John and Quenouille [2], are for 3 blocks of a 3 × 6 factorial experiment. The data can be considered as a 3 × 6 × 3 table (i.e. blocks × treatment with 6 levels × treatment with 3 levels). This table is input and the 6 × 3 table of treatment means for over blocks is computed and printed.

9.1 Example Text

Note: the listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G11BCF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, LTMAX
      PARAMETER        (MMAX=5,LTMAX=54)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, K, MAXST, MCELLS, MDIM, NCELLS,
     +                 NCOL, NDIM, NROW
      CHARACTER        STAT
```

```
*     .. Local Arrays ..
      real              AUXT(LTMAX), STABLE(LTMAX), TABLE(LTMAX),
     +                  WK(LTMAX)
      INTEGER           IDIM(MMAX), ISDIM(MMAX), IWK(3*MMAX),
     +                  MLEVEL(MMAX)
*     .. External Subroutines ..
      EXTERNAL          G11BCF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G11BCF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) STAT, NCELLS, NDIM
      IF (NCELLS.LE.LTMAX .AND. NDIM.LT.MMAX) THEN
         READ (NIN,*) (TABLE(I),I=1,NCELLS)
         READ (NIN,*) (IDIM(J),J=1,NDIM)
         READ (NIN,*) (ISDIM(J),J=1,NDIM)
         MAXST = LTMAX
         IFAIL = 0
*
         CALL G11BCF(STAT,TABLE,NCELLS,NDIM,IDIM,ISDIM,STABLE,MAXST,
     +               MCELLS,MDIM,MLEVEL,AUXT,IWK,WK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Marginal Table'
         WRITE (NOUT,*)
         NCOL = MLEVEL(MDIM)
         NROW = MCELLS/NCOL
         K = 1
         DO 20 I = 1, NROW
            WRITE (NOUT,99999) (STABLE(J),J=K,K+NCOL-1)
            K = K + NCOL
   20    CONTINUE
      END IF
      STOP
*
99999 FORMAT (10F8.2)
      END
```

9.2 Example Data

```
G11BCF Example Program Data

'A' 54 3

274 361 253 325 317 339 326 402 336 379 345 361 352 334 318 339 393 358
350 340 203 397 356 298 382 376 355 418 387 379 432 339 293 322 417 342
 82 297 133 306 352 361 220 333 270 388 379 274 336 307 266 389 333 353

3 6 3
0 1 1
```

9.3 Example Results

```
G11BCF Example Program Results

 Marginal Table

 235.33  332.67  196.33
 342.67  341.67  332.67
 309.33  370.33  320.33
 395.00  370.33  338.00
 373.33  326.67  292.33
 350.00  381.00  351.00
```

G11CAF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G11CAF returns parameter estimates for the conditional logistic analysis of stratified data, for example, data from case-control studies and survival analyses.

2 Specification

```
 SUBROUTINE G11CAF(N, M, NS, Z, LDZ, ISZ, IP, IC, ISI, DEV, B, SE,
1                  SC, COV, NCA, NCT, TOL, MAXIT, IPRINT, WK, LWK,
2                  IFAIL)
 real              Z(LDZ,M), DEV, B(IP), SE(IP), SC(IP),
1                  COV(IP*(IP+1)/2), TOL, WK(LWK)
 INTEGER           N, M, NS, LDZ, ISZ(M), IP, IC(N), ISI(N),
1                  NCA(NS), NCT(NS), MAXIT, IPRINT, LWK, IFAIL
```

3 Description

In the analysis of binary data, the logistic model is commonly used. This relates the probability of one of the outcomes, say $y = 1$, to p explanatory variates or covariates by

$$Prob(y=1) = \frac{\exp(\alpha + z^T\beta)}{1+\exp(\alpha + z^T\beta)}$$

where β is a vector of unknown coefficients for the covariates z and α is a constant term. If the observations come from different strata or groups, α would vary from strata to strata. If the observed outcomes are independent then the y's follow a Bernoulli distribution, i.e., a binomial distribution with sample size one and the model can be fitted as a generalized linear model with binomial errors.

In some situations the number of observations for which $y = 1$ may not be independent. For example, in epidemiological research, case-control studies are widely used in which one or more observed cases are matched with one or more controls. The matching is based on fixed characteristics such as age, sex etc. and is designed to eliminate the effect of such characteristics in order to more accurately determine the effect of other variables. Each case-control group can be considered as a stratum. In this type of study the binomial model is not appropriate, except if the strata are large, and a conditional logistic model is used. This considers the probability of the cases having the observed vectors of covariates given the set of vectors of covariates in the strata. In the situation of one case per stratum, the conditional likelihood for n_s strata can be written as

$$L = \prod_{i=1}^{n_s} \frac{\exp(z_i^T\beta)}{\left[\sum_{l\in S_i}\exp(z_l^T\beta)\right]} \tag{1}$$

where S_i is the set of observations in the ith stratum, with associated vectors of covariates z_l, $l \in S_i$, and z_i is the vector of covariates of the case in the ith stratum. In the general case of c_i cases per strata then the full conditional likelihood is

$$L = \prod_{i=1}^{n_s} \frac{\exp(s_i^T\beta)}{\left[\sum_{l\in C_i}\exp(s_l^T\beta)\right]} \tag{2}$$

where s_i is the sum of the vectors of covariates for the cases in the ith stratum and s_l, $l \in C_i$ refer to the sum of vectors of covariates for all distinct sets of c_i observations drawn from the ith stratum. The conditional likelihood can be maximized by a Newton-Raphson procedure. The covariances of the parameter estimates can be estimated from the inverse of the matrix of second derivatives of the logarithm of the conditional likelihood, while the first derivatives provide the score function, $U_j(\beta)$, $j = 1, 2, \ldots, p$, which can be used for testing the significance of parameters.

If the strata are not small, C_i can be large so to improve the speed of computation, the algorithm given by Howard in [3] and described by [4] is used.

A second situation in which the above conditional likelihood arises is in fitting Cox's proportional hazard model (see G12BAF) in which the strata refer to the risk sets for each failure time and the where failures are cases. When ties are present in the data G12BAF uses an approximation. For an exact estimate, the data can be expanded using G12ZAF to create the risk sets/strata and G11CAF used.

4 References

[1] Cox D R (1972) Regression models in life tables (with discussion) *J. Roy. Statist. Soc. Ser. B* **34** 187–220

[2] Cox D R and Hinkley D V (1974) *Theoretical Statistics* Chapman and Hall

[3] Howard S (1972) Remark on the paper by Cox, D.R. (1972): Regression methods *J. R. Statist. Soc.* **B 34** and life tables 187–220

[4] Krailo M D and Pike M C (1984) Algorithm AS 196. Conditional multivariate *Appl. Statist.* **33** logistic analysis of stratified case-control studies 95–103

[5] Smith P G, Pike M C, Hill P, Breslow N E and Day N E (1981) Multivariate conditional logistic analysis of stratum-matched case-control *Appl. Statist.* (Algorithm AS 162.) **30** studies 190–197

5 Parameters

1: N — INTEGER *Input*

On entry: the number of observations, n.

Constraint: N $\geq$ 2.

2: M — INTEGER *Input*

On entry: the number of covariates in array Z.

Constraint: M $\geq$ 1.

3: NS — INTEGER *Input*

On entry: the number of strata, n_s.

Constraint: NS $\geq$ 1.

4: Z(LDZ,M) — ***real*** array *Input*

On entry: the ith row must contain the covariates which are associated with the ith observation.

5: LDZ — INTEGER *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which G11CAF is called.

Constraint: LDZ $\geq$ N.

6: ISZ(M) — INTEGER array *Input*

On entry: indicates which subset of covariates are to be included in the model.

If ISZ(j) $\geq$ 1 the jth covariate is included in the model.
If ISZ(j) $=$ 0 the jth covariate is excluded from the model and not referenced.

Constraints: ISZ(j) $\geq$ 0 and at least one value must be non-zero.

7: IP — INTEGER *Input*

On entry: the number of covariates included in the model, p as indicated by ISZ.

Constraints: IP $\geq$ 1 and IP = number of non-zero values of ISZ.

8: IC(N) — INTEGER array *Input*

On entry: indicates whether the ith observation is a case or a control.

IC(i) = 0 indicates that the ith observation is a case.
IC(i) = 1 indicates that the ith observation is a control.

Constraint: IC(i) = 0 or 1 for $i = 1, 2, \ldots$,N.

9: ISI(N) — INTEGER array *Input*

On entry: stratum indicators which also allow data points to be excluded from the analysis.

ISI(i) = k indicates that the ith observation is from the kth stratum, where $k = 1, 2, \ldots$,NS.
ISI(i) = 0 indicates that the ith observation is to be omitted from the analysis.

Constraints: $0 \leq$ ISI(i) $\leq$ NS for $i = 1, 2, \ldots$,N, and more than IP values of ISI(i) > 0.

10: DEV — ***real*** *Output*

On exit: the deviance, that is $-2\times$(maximized log marginal likelihood).

11: B(IP) — ***real*** array *Input/Output*

On entry: initial estimates of the covariate coefficient parameters β. B(j) must contain the initial estimate of the coefficent of the covariate in Z corresponding to the jth non-zero value of ISZ.

Suggested values: in many cases an initial value of zero for B(j) may be used. For another suggestion see Section 8.

On exit: B(j) contains the estimate $\hat{\beta}_i$ the coefficient of the covariate stored in the ith column of Z where i is the jth non-zero value in the array ISZ.

12: SE(IP) — ***real*** array *Output*

On exit: SE(j) is the asymptotic standard error of the estimate contained in B(j) and score function in SC(j) for $j = 1, 2, \ldots$,IP.

13: SC(IP) — ***real*** array *Output*

On exit: SC(j) is the value of the score function $U_j(\beta)$ for the estimate contained in B(j).

14: COV(IP*(IP+1)/2) — ***real*** array *Output*

On exit: the variance-covariance matrix of the parameter estimates in B stored in packed form by column, i.e., the covariance between the parameter estimates given in B(i) and B(j), $j \geq i$, is given in COV($j(j-1)/2+i$).

15: NCA(NS) — INTEGER array *Output*

On exit: NCA(i) contains the number of cases in the ith stratum for $i = 1, 2, \ldots$,NS.

16: NCT(NS) — INTEGER array *Output*

On exit: NCT(i) contains the number of controls in the ith stratum for $i = 1, 2, \ldots$,NS.

17: TOL — ***real*** *Input*

On entry: indicates the accuracy required for the estimation. Convergence is assumed when the decrease in deviance is less than TOL$\times$(1.0+CurrentDeviance). This corresponds approximately to an absolute accuracy if the deviance is small and a relative accuracy if the deviance is large.

Constraint: TOL $\geq 10 \times$ ***machine precision***.

18: MAXIT — INTEGER *Input*

On entry: the maximum number of iterations required for computing the estimates. If MAXIT is set to 0 then the standard errors, the score functions and the variance-covariance matrix are computed for the input value of β in B but β is not updated.

Constraint: MAXIT ≥ 0.

19: IPRINT — INTEGER *Input*

On entry: indicates if the printing of information on the iterations is required. If IPRINT ≤ 0, there is no printing. If IPRINT ≥ 1, the deviance and the current estimates are printed every IPRINT iterations.

When printing occurs the output is directed to the current advisory message unit (see X04ABF).

20: WK(LWK) — ***real*** array *Workspace*

21: LWK — INTEGER *Input*

On entry: the length of the workspace array WK.

Constraint: LWK $\geq pn_0 + (c_m + 1)(p + 1)(p + 2)/2 + c_m$, where n_0 is the number of observations included in the model, i.e., the number of observations for which ISI(i) $\neq 0$ and c_m is the maximum number of observations in any stratum.

22: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, M < 1,
or N < 2,
or NS < 1,
or IP < 1,
or LDZ < N,
or TOL < 10 × ***machine precision***,
or MAXIT < 0.

IFAIL = 2

On entry, ISZ(i) < 0 for some i,
or the value of IP is incompatible with ISZ,
or IC(i) $\neq$ 1 or 0.
or ISI(i) < 0 or ISI(i) > NS,
or number of values of ISZ(i) > 0 is greater than or equal to n_0, the number of observations excluding any with ISI(i) = 0.

IFAIL = 3

The value of LWK is too small.

IFAIL = 4

Overflow has been detected. Try using different starting values.

IFAIL = 5

The matrix of second partial derivatives is singular. Try different starting values or include fewer covariates.

IFAIL = 6

Convergence has not been achieved in MAXIT iterations. The progress towards convergence can be examined by using a non-zero value of IPRINT. Any non-convergence may be due to a linear combination of covariates being monotonic with time.

Full results are returned.

7 Accuracy

The accuracy is specified by TOL.

8 Further Comments

The other models described in Section 3 can be fitted using the generalized linear modelling routines G02GBF and G02GCF.

The case with one case per stratum can be analysed by having a dummy response variable y such that y $= 1$ for a case and $y = 0$ for a control, and fitting a Poisson generalized linear model with a log link and including a factor with a level for each strata. These models can be fitted by using G02GCF.

G11CAF uses mean centering which involves subtracting the means from the covariables prior to computation of any statistics. This helps to minimize the effect of outlying observations and accelerates convergence. In order to reduce the risk of the sums computed by Howard's algorithm becoming too large, the scaling factor described in [4] is used.

If the initial estimates are poor then there may be a problem with overflow in calculating $\exp(\beta^T z_i)$ or there may be non-convergence. Reasonable estimates can often be obtained by fitting an unconditional model.

9 Example

The data was used for illustrative purposes by [5] and consists of two strata and two covariates. The data is input, the model is fitted and the results are printed.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G11CAF Example Program Text.
*     Mark 19 Release. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX, SMAX, MLWK
      PARAMETER        (NMAX=500,MMAX=20,SMAX=10,MLWK=10000)
*     .. Local Scalars ..
      real             DEV, TOL
      INTEGER          I, IFAIL, IP, IPRINT, J, LDZ, LWK, M, MAXIT, N,
     +                 NS
*     .. Local Arrays ..
```

```
      real             B(MMAX), COV(MMAX*(MMAX+1)/2), SC(MMAX),
     +                 SE(MMAX), WK(MLWK), Z(NMAX,MMAX)
      INTEGER          IC(NMAX), ISI(NMAX), ISZ(MMAX), NCA(SMAX),
     +                 NCT(SMAX)
*     .. External Subroutines ..
      EXTERNAL         G11CAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G11CAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      LDZ = NMAX
      LWK = MLWK
      TOL = 1.0e-5
      READ (NIN,*) N, M, NS, MAXIT, IPRINT
      IF (N.LE.NMAX .AND. M.LE.MMAX .AND. NS.LE.SMAX) THEN
         DO 20 I = 1, N
            READ (NIN,*) ISI(I), IC(I), (Z(I,J),J=1,M)
   20    CONTINUE
         READ (NIN,*) (ISZ(J),J=1,M), IP
         READ (NIN,*) (B(J),J=1,M)
*
         IFAIL = 0
*
         CALL G11CAF(N,M,NS,Z,LDZ,ISZ,IP,IC,ISI,DEV,B,SE,SC,COV,NCA,NCT,
     +               TOL,MAXIT,IPRINT,WK,LWK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99999) ' Deviance = ', DEV
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Strata     No. Cases   No. Controls'
         WRITE (NOUT,*)
         DO 40 I = 1, NS
            WRITE (NOUT,99998) I, NCA(I), NCT(I)
   40    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Parameter       Estimate',
     +     '       Standard Error'
         WRITE (NOUT,*)
         DO 60 I = 1, IP
            WRITE (NOUT,99997) I, B(I), SE(I)
   60    CONTINUE
      END IF
*
99999 FORMAT (A,e13.4)
99998 FORMAT (3X,I3,2(10X,I2))
99997 FORMAT (I6,2(10X,F8.4))
      END
```

9.2 Program Data

```
G11CAF Example Program Data

7 2 2 10 0

1 0 0 1
1 0 1 2
1 1 0 1
```

```
1 1 1 3
2 0 0 1
2 1 1 0
2 1 0 2

    1 1 2

0.0 0.0
```

9.3 Program Results

```
G11CAF Example Program Results

Deviance =    0.5475E+01

 Strata     No. Cases   No. Controls

    1          2           2
    2          1           2

 Parameter      Estimate        Standard Error

    1          -0.5223             1.3901
    2          -0.2674             0.8473
```

G11SAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G11SAF fits a latent variable model (with a single factor) to data consisting of a set of measurements on individuals in the form of binary-valued sequences (generally referred to as score patterns). Various measures of goodness of fit are calculated along with the factor (theta) scores.

2. Specification

```
      SUBROUTINE G11SAF (IP, N, GPROB, IS, X, NRX, IRL, A, C, IPRINT,
     1                   CGETOL, MAXIT, CHISQR, ISHOW, NITER, ALPHA,
     2                   PIGAM, CM, ICM, G, EXPP, NRE, OBS, EXF, Y, XL,
     3                   IOB, RLOGL, CHI, IDF, SIGLEV, W, LW, IFAIL)
      INTEGER        IP, N, IS, NRX, IRL(IS), IPRINT, MAXIT, ISHOW,
     1               NITER, ICM, NRE, IOB(IS), IDF, LW, IFAIL
      real           A(IP), C(IP), CGETOL, ALPHA(IP), PIGAM(IP),
     1               CM(ICM,2*IP), G(2*IP), EXPP(NRE,IP), OBS(NRE,IP),
     2               EXF(IS), Y(IS), XL(IS), RLOGL, CHI, SIGLEV, W(LW)
      LOGICAL        GPROB, X(NRX,IP), CHISQR
```

3. Description

Given a set of p dichotomous variables $\underset{\sim}{x} = (x_1,x_2,...,x_p)'$, where $'$ denotes vector or matrix transpose, the objective is to investigate whether the association between them can be adequately explained by a latent variable model of the form (see Bartholomew [1] and [2]):

$$G\{\pi_i(\theta)\} = \alpha_{i0} + \alpha_{i1}\theta \qquad (1)$$

The x_i are called item responses and take the value 0 or 1. θ denotes the latent variable assumed to have a standard Normal distribution over a population of individuals to be tested on p items. Call $\pi_i(\theta) = P(x_i{=}1|\theta)$ the item response function: it represents the probability that an individual with latent ability θ will produce a positive response (1) to item i. α_{i0} and α_{i1} are item parameters which can assume any real values. The set of parameters $\{\alpha_{i1}$, for $i = 1,2,...,p\}$ being coefficients of the unobserved variable θ, can be interpreted as 'factor loadings'.

G is a function selected by the user as either Φ^{-1} or logit, mapping the interval (0,1) onto the whole real line. Data from a random sample of n individuals takes the form of the matrices X and R defined below:

$$X_{s\times p} = \begin{bmatrix} x_{11} & x_{12} & . & . & . & x_{1p} \\ x_{21} & x_{22} & . & . & . & x_{2p} \\ . & . & & & & . \\ . & . & & & & . \\ . & . & & & & . \\ x_{s1} & x_{s2} & . & . & . & x_{sp} \end{bmatrix} = \begin{bmatrix} \underset{\sim}{x}_1' \\ \underset{\sim}{x}_2' \\ . \\ . \\ . \\ \underset{\sim}{x}_s' \end{bmatrix}, \quad R_{s\times 1} = \begin{bmatrix} r_1 \\ r_2 \\ . \\ . \\ . \\ r_s \end{bmatrix}$$

where $\underset{\sim}{x}_l = (x_{l1},x_{l2},...,x_{lp})'$ denotes the lth score pattern in the sample, r_l the frequency with which $\underset{\sim}{x}_l$ occurs and s the number of different score patterns observed. (Thus $\sum_{l=1}^{s} r_l = n$). It can be shown that the log likelihood function is proportional to

$$\sum_{l=1}^{s} r_l \log P_l$$

$$\text{where } P_l = P(\underset{\sim}{x}=\underset{\sim}{x}_l) = \int_{-\infty}^{\infty} P(\underset{\sim}{x}=\underset{\sim}{x}_l|\theta)\phi(\theta)\, d\theta \qquad (2)$$

($\phi(\theta)$ being the probability density function of a standard Normal random variable).

P_l denotes the unconditional probability of observing score pattern $\underset{\sim}{x}_l$. The integral in (2) is approximated using Gauss-Hermite quadrature. If we take $G(z) = \operatorname{logit} z = \log\left(\frac{z}{1-z}\right)$ in (1) and reparametrise as follows

$$\alpha_i = \alpha_{i1}$$

$$\pi_i = \operatorname{logit}^{-1} \alpha_{i0}$$

(1) reduces to the logit model (see Bartholomew [1])

$$\pi_i(\theta) = \frac{\pi_i}{\pi_i + (1-\pi_i)\exp(-\alpha_i\theta)}$$

If we take $G(z) = \Phi^{-1}(z)$ (where Φ is the cumulative distribution function of a standard Normal random variable) and reparametrise as follows

$$\alpha_i = \frac{\alpha_{i1}}{\sqrt{(1+\alpha_{i1}^2)}}$$

$$\gamma_i = \frac{-\alpha_{i0}}{\sqrt{(1+\alpha_{i1}^2)}}$$

(1) reduces to the probit model (see Bock and Aitkin [3])

$$\pi_i(\theta) = \Phi\left(\frac{\alpha_i\theta - \gamma_i}{\sqrt{(1-\alpha_i^2)}}\right)$$

An E-M algorithm (see Bock and Aitkin [3]) is used to maximize the log likelihood function. The number of quadrature points used is set initially to 10 and once convergence is attained increased to 20.

The theta score of an individual responding in score pattern $\underset{\sim}{x}_l$ is computed as the posterior mean, i.e. $\mathrm{E}(\theta|\underset{\sim}{x}_l)$. For the logit model the component score $X_l = \sum_{j=1}^{p} \alpha_j x_{lj}$ is also calculated. (Note that in calculating the theta scores and measures of goodness of fit G11SAF automatically reverses the coding on item j if $\alpha_j < 0$ – it is assumed in the model that a response at the one level is showing a higher measure of latent ability than a response at the zero level.)

The frequency distribution of score patterns is required as input data. If the user's data is in the form of individual score patterns (uncounted), then G11SBF may be used to calculate the frequency distribution.

4. References

[1] BARTHOLOMEW, D.J.
Factor analysis for categorical data (with discussion).
J. Roy. Statist. Soc. Ser. B, 42, pp. 293-321, 1980.

[2] BARTHOLOMEW, D.J.
Latent Variable Models and Factor Analysis.
Griffin, London, 1987.

[3] BOCK, R.D. and AITKIN, M.
Marginal maximum likelihood estimation of item parameters: Application of an E-M algorithm.
Psychometrika, 46, pp. 443-459, 1981.

5. Parameters

1: IP – INTEGER. *Input*

On entry: the number of dichotomous variables, p.

Constraint: IP $\geq$ 3.

2: N – INTEGER. *Input*

On entry: the number of individuals in the sample, n.

Constraint: N $\geq$ 7.

3: GPROB – LOGICAL. *Input*

On entry: GPROB must be set equal to .TRUE. if $G(z) = \Phi^{-1}(z)$ and .FALSE. if $G(z) = \text{logit}\, z$.

4: IS – INTEGER. *Input*

On entry: IS must be set equal to the number of different score patterns in the sample, s.

Constraint: $2\times\text{IP} < \text{IS} \leq \min(2^{\text{IP}},\text{N})$.

5: X(NRX,IP) – LOGICAL array. *Input/Output*

On entry: the first s rows of X must contain the s different score patterns. The lth row of X must contain the lth score pattern with X(l,j) set equal to .TRUE. if $x_{lj} = 1$ and .FALSE. if $x_{lj} = 0$. All rows of X must be distinct.

On exit: if IFAIL = 0 or IFAIL $\geq$ 3 then the first s rows of X still contain the s different score patterns. However, the following points should be noted:

(1) If the estimated factor loading for the jth item is negative then that item is recoded, i.e. 0's and 1's (or .TRUE. and .FALSE.) in the jth column of X are interchanged.

(2) The rows of X will be re-ordered so that the theta scores corresponding to rows of X are in increasing order of magnitude.

6: NRX – INTEGER. *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G11SAF is called.

Constraint: NRX $\geq$ IS.

7: IRL(IS) – INTEGER array. *Input/Output*

On entry: the ith component of IRL must be set equal to the frequency with which the ith row of X occurs.

Constraints: IRL$(i) \geq 0$ for $i = 1,2,\ldots,s$.

$$\sum_{i=1}^{s} \text{IRL}(i) = n.$$

On exit: if IFAIL = 0 or IFAIL $\geq$ 3 then the first s components of IRL are re-ordered as are the rows of X.

8: A(IP) – ***real*** array. *Input/Output*

On entry: A(j) must be set equal to an initial estimate of α_{j1}. **In order to avoid divergence problems with the E-M algorithm the user is strongly advised to set all the A(j) to 0.5.**

On exit: A(j) contains the latest estimate of α_{j1}, for $j = 1,2,\ldots,p$. (Because of possible re-coding all elements of A will be positive.)

9: C(IP) – ***real*** array. *Input/Output*

On entry: C(j) must be set equal to an initial estimate of α_{j0}. **In order to avoid divergence problems with the E-M algorithm the user is strongly advised to set all the C(j)'s to 0.0.**

On exit: C(j) contains the latest estimate of α_{j0}, for $j = 1,2,...,p$.

10: IPRINT – INTEGER. *Input*

On entry: the frequency with which the maximum likelihood search routine is to be monitored.

If IPRINT > 0, the search is monitored once every IPRINT iterations, and when the number of quadrature points is increased, and again at the final solution point.

If IPRINT = 0, the search is monitored once at the final point.

If IPRINT < 0, the search is not monitored at all.

IPRINT should normally be set to a small positive number. If in doubt, try setting IPRINT to 1.

11: CGETOL – ***real.*** *Input*

On entry: the accuracy to which the solution is required. If CGETOL is set to 10^{-l} then, on exit, if IFAIL = 0 or 7 all elements of the gradient vector will be smaller than 10^{-l} in absolute value. For most practical purposes the value 10^{-4} should suffice. The user should be wary of setting CGETOL too small since the convergence criterion may then have become too strict for the machine to handle. If on entry, CGETOL has been set to a value which is less than the square root of the ***machine precision***, ε, then G11SAF will use the value $\sqrt{\varepsilon}$ instead.

12: MAXIT – INTEGER. *Input*

On entry: the maximum number of iterations to be made in the maximum likelihood search. There will be an error exit (see Section 6) if the search routine has not converged in MAXIT iterations. A reasonable setting for MAXIT may be 1000.

Constraint: MAXIT $\geq$ 1.

13: CHISQR – LOGICAL. *Input*

On entry: if CHISQR is set equal to .TRUE., then a likelihood ratio statistic will be calculated (see CHI).

If CHISQR is set equal to .FALSE., no such statistic will be calculated.

14: ISHOW – INTEGER. *Input*

On entry: indicates which of the following three quantities are to be printed before exit from the routine (except when IFAIL = 1 or 2):

(a) Table of maximum likelihood estimates and standard errors (as returned in the output arrays A, C, ALPHA, PIGAM and CM).

(b) Table of observed and expected first and second order margins (as returned in the output arrays EXPP and OBS).

(c) Table of observed and expected frequencies of score patterns along with theta scores (as returned in the output arrays IRL, EXF, Y, XL and IOB) and the likelihood ratio statistic (if required).

ISHOW = 0 none of the above are printed
ISHOW = 1 (a) only is printed
ISHOW = 2 (b) only is printed
ISHOW = 3 (c) only is printed
ISHOW = 4 (a) and (b) are printed
ISHOW = 5 (a) and (c) are printed
ISHOW = 6 (b) and (c) are printed
ISHOW = 7 (a), (b) and (c) are printed

Constraint: $0 \leq$ ISHOW ≤ 7.

15: NITER – INTEGER. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then NITER contains the number of iterations performed by the maximum likelihood search routine.

16: ALPHA(IP) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then ALPHA(j) contains the latest estimate of α_j. (Because of possible re-coding all elements of ALPHA will be positive.)

17: PIGAM(IP) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then PIGAM(j) contains the latest estimate of either π_j if GPROB = .FALSE. (logit model) or γ_j if GPROB = .TRUE. (probit model).

18: CM(ICM,2*IP) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then the strict lower triangle of CM contains the correlation matrix of the parameter estimates held in ALPHA and PIGAM on exit. The diagonal elements of CM contain the standard errors. Thus:

For $i = 1,2,...,p$

CM($2\times i-1$,$2\times i-1$)	= standard error (ALPHA(i))
CM($2\times i$,$2\times i$)	= standard error (PIGAM(i))
CM($2\times i$,$2\times i-1$)	= correlation (PIGAM(i),ALPHA(i))

For $j = 1,2,...,i-1$

CM($2\times i-1$,$2\times j-1$)	= correlation (ALPHA(i),ALPHA(j))
CM($2\times i$,$2\times j$)	= correlation (PIGAM(i),PIGAM(j))
CM($2\times i-1$,$2\times j$)	= correlation (ALPHA(i),PIGAM(j))
CM($2\times i$, $2\times j-1$)	= correlation (ALPHA(j),PIGAM(i))

If the second derivative matrix cannot be computed then all the elements of CM are returned as zero.

19: ICM – INTEGER. *Input*

On entry: the first dimension of the array CM as declared in the (sub)program from which G11SAF is called.

Constraint: ICM $\geq$ IP + IP.

20: G(2*IP) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then G contains the estimated gradient vector corresponding to the final point held in the arrays ALPHA and PIGAM. G($2\times j-1$) contains the derivative of the log likelihood with respect to ALPHA(j) for $j = 1,2,...,p$. G($2\times j$) contains the derivative of the log likelihood with respect to PIGAM(j) for $j = 1,2,...,p$.

21: EXPP(NRE,IP) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then EXPP(i,j) contains the expected percentage of individuals in the sample who respond positively to items i and j, corresponding to the final point held in the arrays ALPHA and PIGAM.

22: NRE – INTEGER. *Input*

On entry: the first dimension of the array EXPP and OBS as declared in the (sub)program from which G11SAF is called.

Constraint: NRE ≥ IP.

23: OBS(NRE,IP) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then OBS(i,j) contains the observed percentage of individuals in the sample who responded positively to items i and j.

24: EXF(IS) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then EXF(l) contains the expected frequency of the lth score pattern (lth row of X), corresponding to the final point held in the arrays ALPHA and PIGAM.

25: Y(IS) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then Y(l) contains the estimated theta score corresponding to the lth row of X, for the final point held in the arrays ALPHA and PIGAM.

26: XL(IS) – ***real*** array. *Workspace*

If GPROB has been set equal to .FALSE. (logit model) then on exit if IFAIL = 0 or IFAIL ≥ 3, XL(l) contains the estimated component score corresponding to the lth row of X for the final point held in the arrays ALPHA and PIGAM. If GPROB is set equal to .TRUE. (probit model) this array is not used.

27: IOB(IS) – INTEGER array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then IOB(l) contains the number of items in the lth row of X for which the response was positive (.TRUE.).

28: RLOGL – ***real***. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 3 then RLOGL contains the value of the log likelihood kernel corresponding to the final point held in the arrays ALPHA and PIGAM, viz

$$\sum_{l=1}^{s} \mathrm{IRL}(l)\times\log(\mathrm{EXF}(l)/n).$$

29: CHI – ***real***. *Output*

On exit: if CHISQR was set equal to .TRUE. on entry, then if IFAIL = 0 or IFAIL ≥ 3 CHI will contain the value of the likelihood ratio statistic corresponding to the final parameter estimates held in the arrays ALPHA and PIGAM, viz

$$2\times\sum_{l=1}^{s} \mathrm{IRL}(l)\times\log(\mathrm{EXF}(l)/\mathrm{IRL}(l)).$$

The summation is over those elements of IRL which are positive. If EXF(l) is less than 5.0, then adjacent score patterns are pooled (the score patterns in X being first put in order of increasing theta score).

If CHISQR has been set equal to .FALSE., then CHI is not used.

30: IDF – INTEGER. *Output*

On exit: if CHISQR was set equal to .TRUE. on entry, then if IFAIL = 0 or IFAIL ≥ 3 IDF will contain the degrees of freedom associated with the likelihood ratio statistic, CHI.

$$\mathrm{IDF} = s_0 - 2\times p \qquad \text{if } s_0 < 2^p$$
$$\mathrm{IDF} = s_0 - 2\times p - 1 \qquad \text{if } s_0 = 2^p$$

where s_0 denotes the number of terms summed to calculate CHI ($s_0 = s$ only if there is no pooling).

If CHISQR has been set equal to .FALSE., then IDF is not used.

31: SIGLEV – ***real***. *Output*

On exit: if CHISQR was set equal to .TRUE. on entry, then if IFAIL = 0 or IFAIL ≥ 3 SIGLEV will contain the significance level of CHI based on IDF degrees of freedom. If IDF is zero or negative then SIGLEV is set to zero. If CHISQR was set equal to .FALSE., then SIGLEV is not used.

32: W(LW) – ***real*** array. *Workspace*

33: LW – INTEGER. *Input*

On entry: the dimension of the array W as declared in the (sub)program from which G11SAF is called.

Constraint: LW ≥ 4×IP×(IP+16).

34: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, IP < 3,
or N < 7,
or IS ≤ 2×IP,
or IS > N,
or IS > 2^{IP},
or two or more rows of X are identical,
or NRX < IS,
or $\sum_{l=1}^{IS}$ IRL(l) ≠ N,
or at least one of IRL(l) < 0, for (l = 1,2,...,IS),
or MAXIT < 1,
or ISHOW < 0,
or ISHOW > 7,
or ICM < IP + IP,
or NRE < IP,
or LW < 4×IP×(IP+16).

IFAIL = 2

For at least one of the IP items the responses are all at the same level. To proceed, the user must delete this item from his data set.

IFAIL = 3

There have been MAXIT iterations of the maximum likelihood search routine. If steady increases in the log likelihood kernel were monitored up to the point where this exit occurred, then the exit probably occurred simply because MAXIT was set too small, so the calculations should be restarted from the final point held in A and C. This type of exit may also indicate that there is no maximum to the likelihood surface.

IFAIL = 4

One of the elements of A has exceeded 10.0 in absolute value (see Section 8.3). If steady increases in the log likelihood kernel were monitored up to the point where this exit occurred then this exit may indicate that there is no maximum to the likelihood surface. The user is advised to restart the calculations from a different point to see whether the E-M algorithm moves off in the same direction.

IFAIL = 5

This indicates a failure in F01ABF which is used to invert the second derivative matrix for calculating the variance-covariance matrix of parameter estimates. It was also found that MAXIT iterations had been performed (see IFAIL = 3). The elements of CM will then have been set to zero on exit. The user is advised to restart the calculations with a larger value for MAXIT.

IFAIL = 6

This indicates a failure in F01ABF which is used to invert the second derivative matrix for calculating the variance-covariance matrix of parameter estimates. It was also found that one of the elements of A had exceeded 10.0 in absolute value (see IFAIL = 4). The elements of CM will have then been set to zero on exit. The user is advised to restart the calculations from a different point to see whether the E-M algorithm moves off in the same direction.

IFAIL = 7

If CHISQR was set equal to .TRUE. on entry, so that a likelihood ratio statistic was calculated, then IFAIL = 7 merely indicates that the value of IDF on exit is $\leq$ 0, i.e. the χ^2 statistic is meaningless. In this case SIGLEV is returned as zero. **All other output information should be correct, i.e. can be treated as if IFAIL were 0 on exit.**

7. Accuracy

On exit from G11SAF if IFAIL = 0 or 7 then the following condition will be satisfied:

$$\max_{1\leq i\leq 2\times p} \{|G(i)|\} < \text{CGETOL}.$$

If IFAIL = 3 or 5 on exit (i.e. MAXIT iterations have been performed but the above condition doesn't hold), then the elements in A, C, ALPHA and PIGAM may still be good approximations to the maximum likelihood estimates. The user is advised to inspect the elements of G to see whether this is confirmed.

8. Further Comments

8.1. Timing

The number of iterations required in the maximum likelihood search depends upon the number of observed variables, p, and the distance of the user-supplied starting point from the solution. The number of multiplications and divisions performed in an iteration is proportional to p.

8.2. Initial Estimates

The user is strongly advised to use the recommended starting values for the elements of A and C. Divergence may result from user-supplied values even if they are very close to the solution. Divergence may also occur when an item has nearly all its responses at one level.

8.3. Heywood Cases

As in normal factor analysis, Heywood cases can often occur, particularly when p is small and n not very big. To overcome this difficulty the maximum likelihood search routine is terminated when the absolute value of one of the α_{j1} exceeds 10.0. The user has the option of deciding whether to exit from G11SAF (by setting IFAIL = 0 on entry) or to permit G11SAF to proceed onwards as if it had exited normally from the maximum likelihood search routine (setting IFAIL = −1 on entry). The elements in A, C, ALPHA and PIGAM may still be good approximations to the maximum likelihood estimates. The user is advised to inspect the elements G to see whether this is confirmed.

8.4. Goodness of Fit Statistic

When n is not very large compared to s a goodness of fit statistic should not be calculated as many of the expected frequencies will then be less than 5.

8.5. First and Second Order Margins

The observed and expected **percentages** of sample members responding to individual and pairs of items held in the arrays OBS and EXPP on exit can be converted to observed and expected **numbers** by multiplying all elements of these two arrays by $n/100.0$.

9. Example

A program to fit the logit latent variable model to the following data:

Index	Score Pattern	Observed Frequency
1	0000	154
2	1000	11
3	0001	42
4	0100	49
5	1001	2
6	1100	10
7	0101	27
8	0010	84
9	1101	10
10	1010	25
11	0011	75
12	0110	129
13	1011	30
14	1110	50
15	0111	181
16	1111	121
Total		1000

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G11SAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          IPMAX, ICM, ISMAX, NRE, LW, NRX
      PARAMETER        (IPMAX=10,ICM=2*IPMAX,ISMAX=1024,NRE=IPMAX,
     +                 LW=4*IPMAX*(IPMAX+16),NRX=ISMAX)
*     .. Local Scalars ..
      real             CGETOL, CHI, RLOGL, SIGLEV
      INTEGER          I, IDF, IFAIL, IP, IPRINT, IS, ISHOW, J, MAXIT,
     +                 N, NITER
      LOGICAL          CHISQR, GPROB
```

```
*     .. Local Arrays ..
      real              A(IPMAX), ALPHA(IPMAX), C(IPMAX),
     +                  CM(ICM,2*IPMAX), EXF(ISMAX), EXPP(NRE,IPMAX),
     +                  G(2*IPMAX), OBS(NRE,IPMAX), PIGAM(IPMAX), W(LW),
     +                  XL(ISMAX), Y(ISMAX)
      INTEGER           IOB(ISMAX), IRL(ISMAX)
      LOGICAL           X(NRX,IPMAX)
*     .. External Subroutines ..
      EXTERNAL          G11SAF, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G11SAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) IP, N, IS
      CALL X04ABF(1,NOUT)
      IF (IP.GT.0 .AND. IP.LE.IPMAX .AND. IS.GE.0 .AND. IS.LE.ISMAX)
     +    THEN
         DO 20 I = 1, IS
            READ (NIN,*) IRL(I), (X(I,J),J=1,IP)
   20    CONTINUE
         GPROB = .FALSE.
         DO 40 I = 1, IP
            A(I) = 0.5e0
            C(I) = 0.0e0
   40    CONTINUE
*        ** Set IPRINT > 0 to obtain intermediate output **
         IPRINT = -1
         CGETOL = 0.0001e0
         MAXIT = 1000
         CHISQR = .TRUE.
         ISHOW = 7
         IFAIL = -1
*
         CALL G11SAF(IP,N,GPROB,IS,X,NRX,IRL,A,C,IPRINT,CGETOL,MAXIT,
     +               CHISQR,ISHOW,NITER,ALPHA,PIGAM,CM,ICM,G,EXPP,NRE,
     +               OBS,EXF,Y,XL,IOB,RLOGL,CHI,IDF,SIGLEV,W,LW,IFAIL)
*
      END IF
      STOP
      END
```

9.2. Program Data

```
G11SAF Example Program Data
 4 1000 16
 154 F F F F
  11 T F F F
  42 F F F T
  49 F T F F
   2 T F F T
  10 T T F F
  27 F T F T
  84 F F T F
  10 T T F T
  25 T F T F
  75 F F T T
 129 F T T F
  30 T F T T
  50 T T T F
 181 F T T T
 121 T T T T
```

9.3. Program Results

```
 G11SAF Example Program Results

 LOG LIKELIHOOD KERNEL ON EXIT =   -0.24039E+04

 MAXIMUM LIKELIHOOD ESTIMATES OF ITEM PARAMETERS ARE AS FOLLOWS
 --------------------------------------------------------------

 ITEM J     ALPHA(J)         S.E.      ALPHA(J,0)        PI(J)       S.E.
 ------     --------         ----      ----------        -----       ----

     1        1.045         0.148        -1.276          0.218       0.017

     2        1.409         0.179         0.424          0.604       0.022

     3        2.659         0.525         1.615          0.834       0.036

     4        1.122         0.140        -0.062          0.485       0.020

 EXPECTED (AND OBSERVED) PERCENTAGE OF CASES PRODUCING
 POSITIVE RESPONSES FOR INDIVIDUAL AND PAIRS OF ITEMS
 -----------------------------------------------------

                                    ITEM
                                    ----

 ITEM      1     2     3     4
 ----      -     -     -     -
   1     25.9
        (25.9)
   2     19.1  57.7
        (19.1)(57.7)
   3     22.5  48.0  69.4
        (22.6)(48.1)(69.5)
   4     16.4  33.9  40.6  48.8
        (16.3)(33.9)(40.7)(48.8)

 OBSERVED  EXPECTED  THETA   COMPONENT  RAW   SCORE
 FREQUENCY FREQUENCY SCORE   SCORE      SCORE PATTERN
 --------- --------- -----   ---------  ----- -------

    154     147.061  -1.273    0.000      0   FFFF
     11      13.444  -0.873    1.045      1   TFFF
     42      42.420  -0.846    1.122      1   FFFT
     49      54.818  -0.747    1.409      1   FTFF
      2       5.886  -0.494    2.167      2   TFFT
     10       8.410  -0.399    2.455      2   TTFF
     27      27.511  -0.374    2.531      2   FTFT
     84      92.062  -0.332    2.659      1   FFTF
     10       6.237  -0.019    3.577      3   TTFT
     25      21.847   0.027    3.705      2   TFTF
     75      73.835   0.055    3.781      2   FFTT
    129     123.766   0.162    4.069      2   FTTF
     30      26.899   0.466    4.826      3   TFTT
     50      50.881   0.591    5.114      3   TTTF
    181     179.564   0.626    5.190      3   FTTT
    121     125.360   1.144    6.236      4   TTTT
 --------- ---------
   1000    1000.000

 LIKELIHOOD RATIO GOODNESS OF FIT STATISTIC =      9.027
                          SIGNIFICANCE LEVEL =      0.251
 (BASED ON   7 DEGREES OF FREEDOM)

 VALUE OF IFAIL PARAMETER ON EXIT FROM G11SAF =   0
```

With IPRINT set to 1 in the example program, intermediate output similar to the following is obtained:

```
G11SAF Example Program Results

ITERATION NUMBER =     0

THE NUMBER OF QUADRATURE POINTS HAS BEEN SET TO 10

CURRENT ESTIMATES OF ALPHA(J,1)'S
---------------------------------

         0.500         0.500         0.500         0.500

CURRENT ESTIMATES OF ALPHA(J,0)'S
---------------------------------

         0.000         0.000         0.000         0.000

************************************************************************

ITERATION NUMBER =     1

VALUE OF LOG LIKELIHOOD KERNEL =   -0.27245E+04

MAGNITUDE OF LARGEST COMPONENT OF GRADIENT VECTOR =      0.242E+03


CURRENT ESTIMATES OF ALPHA(J,1)'S
---------------------------------

         0.512         0.706         0.675         0.685

COMPONENTS OF GRADIENT VECTOR
-----------------------------

     0.769E+00     0.464E+02     0.404E+02     0.412E+02

CURRENT ESTIMATES OF ALPHA(J,0)'S
---------------------------------

        -1.029         0.322         0.824        -0.056

COMPONENTS OF GRADIENT VECTOR
-----------------------------

    -0.242E+03     0.761E+02     0.194E+03    -0.129E+02

************************************************************************
```

.

.

intermediate results omitted

.

.

```
************************************************************************
ITERATION NUMBER =   147

VALUE OF LOG LIKELIHOOD KERNEL =   -0.24039E+04

MAGNITUDE OF LARGEST COMPONENT OF GRADIENT VECTOR =      0.106E-03


CURRENT ESTIMATES OF ALPHA(J,1)'S
---------------------------------

         1.045          1.409          2.659          1.122

COMPONENTS OF GRADIENT VECTOR
-----------------------------

    -0.164E-04    -0.559E-04     0.106E-03    -0.300E-04

CURRENT ESTIMATES OF ALPHA(J,0)'S
---------------------------------

        -1.276          0.424          1.615         -0.062

COMPONENTS OF GRADIENT VECTOR
-----------------------------

     0.100E-05     0.823E-06    -0.351E-06     0.963E-06
************************************************************************
ITERATION NUMBER =   148

VALUE OF LOG LIKELIHOOD KERNEL =   -0.24039E+04

MAGNITUDE OF LARGEST COMPONENT OF GRADIENT VECTOR =      0.953E-04


CURRENT ESTIMATES OF ALPHA(J,1)'S
---------------------------------

         1.045          1.409          2.659          1.122

COMPONENTS OF GRADIENT VECTOR
-----------------------------

    -0.147E-04    -0.501E-04     0.953E-04    -0.269E-04

CURRENT ESTIMATES OF ALPHA(J,0)'S
---------------------------------

        -1.276          0.424          1.615         -0.062

COMPONENTS OF GRADIENT VECTOR
-----------------------------

     0.900E-06     0.737E-06    -0.315E-06     0.863E-06
************************************************************************
```

G11SBF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G11SBF is a service routine which may be used prior to calling G11SAF to calculate the frequency distribution of a set of dichotomous score patterns.

2. Specification

```
SUBROUTINE G11SBF (IP, N, IS, X, NRX, IRL, IFAIL)
INTEGER          IP, N, IS, NRX, IRL(N), IFAIL
LOGICAL          X(NRX,IP)
```

3. Description

When each of n individuals responds to each of p dichotomous variables the data assumes the form of the matrix X defined below

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ . & . & & . \\ . & . & & . \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} \underset{\sim}{x}_1' \\ \underset{\sim}{x}_2' \\ . \\ . \\ \underset{\sim}{x}_n' \end{bmatrix}$$

where the x's take the value of 0 or 1 and $\underset{\sim}{x}_l = (x_{l1}, x_{l2}, \ldots, x_{lp})'$, for $l = 1,2,\ldots,n$ denotes the score pattern of the lth individual ($'$ denoting the transpose of a vector). G11SBF calculates the number of different score patterns, s, and the frequency with which each occurs. This information can then be passed to G11SAF.

4. References

None.

5. Parameters

1: IP – INTEGER. *Input*

On entry: the number of dichotomous variables, p.

Constraint: IP ≥ 3.

2: N – INTEGER. *Input*

On entry: the number of individuals in the sample, n.

Constraint: N ≥ 7.

3: IS – INTEGER. *Output*

On exit: the number of different score patterns, s.

4: X(NRX,IP) – LOGICAL array. *Input/Output*

On entry: X(i,j) must be set equal to .TRUE. if $x_{ij} = 1$, and .FALSE. if $x_{ij} = 0$, for $i = 1,2,\ldots,n$; $j = 1,2,\ldots,p$.

On exit: the first s rows of X contain the s different score patterns.

5: NRX – INTEGER. *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which G11SBF is called.

Constraint: NRX ≥ N.

6: IRL(N) – INTEGER array. *Output*

On exit: the frequency with which the lth row of X occurs, for $l = 1,2,...,s$.

7: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, IP < 3,
or N < 7,
or NRX < N.

7. Accuracy

Exact.

8. Further Comments

The time taken by the routine is small and increases with n.

9. Example

A program to count the frequencies of different score patterns in the following list:

Score Patterns

000
010
111
000
001
000
000
110
001
011

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G11SBF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NRX, IPMAX
      PARAMETER        (NRX=100,IPMAX=5)
```

```
*     .. Local Scalars ..
      INTEGER          I, IFAIL, IP, IS, J, N
*     .. Local Arrays ..
      INTEGER          IRL(NRX)
      LOGICAL          X(NRX,IPMAX)
*     .. External Subroutines ..
      EXTERNAL         G11SBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G11SBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, IP
      IF (N.GT.0 .AND. N.LE.NRX .AND. IP.GT.0 .AND. IP.LE.IPMAX) THEN
         DO 20 I = 1, N
            READ (NIN,*) (X(I,J),J=1,IP)
   20    CONTINUE
         IFAIL = 0
*
         CALL G11SBF(IP,N,IS,X,NRX,IRL,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Frequency     Score pattern'
         WRITE (NOUT,*)
         DO 40 I = 1, IS
            WRITE (NOUT,99999) IRL(I), (X(I,J),J=1,IP)
   40    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,I5,12X,5L2)
      END
```

9.2. Program Data

```
G11SBF Example Program Data
10 3
F F F
F T F
T T T
F F F
F F T
F F F
F F F
T T F
F F T
F T T
```

9.3. Program Results

```
G11SBF Example Program Results

Frequency     Score pattern

    4             F F F
    1             F T F
    1             T T T
    2             F F T
    1             T T F
    1             F T T
```

Chapter G12 – Survival Analysis

Note. Please refer to the Users' Note for your implementation to check that a routine is available.

Routine Name	**Mark of Introduction**	**Purpose**
G12AAF	15	Computes Kaplan–Meier (product-limit) estimates of survival probabilities
G12BAF	17	Fits Cox's proportional hazard model
G12ZAF	19	Creates the risk sets associated with the Cox proportional hazards model for fixed covariates

Chapter G12

Survival Analysis

Contents

1 Scope of the Chapter

This chapter is concerned with statistical techniques used in the analysis of survival/reliability/failure time data.

Other chapters contain routines which are also used to analyse this type of data. Chapter G02 contains generalized linear models, Chapter G07 contains routines to fit distribution models, and Chapter G08 contains rank based methods.

2 Background to the Problems

2.1 Introduction to Terminology

This chapter is concerned with the analysis on the time, t, to a single event. This type of analysis occurs commonly in two areas. In medical research it is known as survival analysis and is often the time from the start of treatment to the occurrence of a particular condition or of death. In engineering it is concerned with reliability and the analysis of failure times, that is how long a component can be used until it fails. In this chapter the time t will be referred to as the **failure time**.

Let the probability density function of the failure time be $f(t)$, then the **survivor function**, $S(t)$, which is the probability of surviving to at least time t, is given by

$$S(t) = \int_t^\infty f(\tau)d\tau = 1 - F(t)$$

where $F(t)$ is the cumulative density function. The **hazard function**, $\lambda(t)$, is the probability that failure occurs at time t given that the individual survived up to time t, and is given by

$$\lambda(t) = f(t)/S(t).$$

The **cumulative hazard** rate is defined as

$$\Lambda(t) = \int_0^t \lambda(\tau)d\tau,$$

hence $S(t) = e^{-\Lambda(t)}$.

It is common in survival analysis for some of the data to be **right censored**. That is, the exact failure time is not known, only that failure occurred after a known time. This may be due to the experiment being terminated before all the individuals have failed, or an individual being removed from the experiment for a reason not connected with effects being tested in the experiment. The presence of censored data leads to complications in the analysis.

2.2 Estimating the Survivor Function and Hazard Plotting

The most common estimate of the survivor function for censored data is the **Kaplan–Meier** or **product-limit** estimate,

$$\hat{S}(t) = \prod_{j=1}^{i} \left(\frac{n_j - d_j}{n_j} \right), \quad t_i \leq t < t_{i+1}$$

where d_j is the number of failures occurring at time t_j out of n_j surviving to t_j. This is a step function with steps at each failure time but not at censored times.

As $S(t) = e^{-\Lambda(t)}$ the cumulative hazard rate can be estimated by

$$\hat{\Lambda}(t) = -\log(\hat{S}(t)).$$

A plot of $\hat{\Lambda}(t)$ or $\log(\hat{\Lambda}(t))$ against t or $\log t$ is often useful in identifying a suitable parametric model for the survivor times. The following relationships can be used in the identification.

(a) Exponential distribution: $\Lambda(t) = \lambda t$.
(b) Weibull distribution: $\log(\Lambda(t)) = \log\lambda + \gamma\log t$.
(c) Gompertz distribution: $\log(\lambda(t)) = \log\lambda + \gamma t$.
(d) Extreme value (smallest) distribution: $\log(\Lambda(t)) = \lambda(t - \gamma)$.

2.3 Proportional Hazard Models

Often in the analysis of survival data the relationship between the hazard function and the a number of explanatory variables or covariates is modelled. The covariates may be, for example, group or treatment indicators or measures of the state of the individual at the start of the observational period. There are two types of covariate time independent covariates such as those described above which do not change value during the observational period and time dependent covariates. The latter can be classified as either external covariates, in which case they are not directly involved with the failure mechanism, or as internal covariates which are time dependent measurements taken on the individual.

The most common function relating the covariates to the hazard function is the proportional hazard function

$$\lambda(t,z) = \lambda_0(t)\exp(\beta^T z)$$

where $\lambda_0(t)$ is a baseline hazard function, z is a vector of covariates and β is a vector of unknown parameters. The assumption is that the covariates have a multiplicative effect on the hazard.

The form of $\lambda_0(t)$ can be one of the distributions considered above or a non-parametric function. In the case of the exponential, Weibull and extreme value distributions the proportional hazard model can be fitted to censored data using the method described by Aitkin and Clayton [1] which uses a generalized linear model with Poisson errors. Other possible models are the gamma distribution and the lognormal distribution.

2.4 Cox's Proportional Hazard Model

Rather than using a specified form for the hazard function, Cox [2] considered the case when $\lambda_0(t)$ was an unspecified function of time. To fit such a model assuming fixed covariates a marginal likelihood is used. For each of the times at which a failure occurred, t_i, the set of those who were still in the study is considered, this includes any that were censored at t_i. This set is known as the risk set for time t_i and denoted by $R(t_{(i)})$. Given the risk set the probability that out of all possible sets of d_i subjects that could have failed the actual observed d_i cases failed can be written as

$$\frac{\exp(s_i^T\beta)}{\sum\exp(z_l^T\beta)} \tag{1}$$

where s_i is the sum of the covariates of the d_i individuals observed to fail at $t_{(i)}$ and the summation is over all distinct sets of n_i individuals drawn from $R(t_{(i)})$. This leads to a complex likelihood. If there are no ties in failure times the likelihood reduces to

$$L = \prod_{i=1}^{n_d} \frac{\exp(z_i^T\beta)}{[\sum_{l\in R(t_{(i)})}\exp(z_l^T\beta)]} \tag{2}$$

where n_d is the number of distinct failure times. For cases where there are ties the following approximation, due to Peto [2], can be used:

$$L = \prod_{i=1}^{n_d} \frac{\exp(s_i^T\beta)}{[\sum_{l\in R(t_{(i)})}\exp(z_l^T\beta)]^{d_i}}. \tag{3}$$

Having fitted the model an estimate of the base-line survivor function (derived from $\lambda_0(t)$ and the residuals) can be computed to examine the suitability of the model, in particular the proportional hazard assumption.

3 Recommendations on Choice and Use of Available Routines

The following routines are available.

G12AAF	computes Kaplan–Meier estimates of the survivor function and their standard deviations.
G12BAF	fits the Cox proportional hazards model for fixed covariates.
G12ZAF	creates the risk sets associated with the Cox proportional hazards model for fixed covariates.

The following routines from other chapters may also be useful in the analysis of survival data.

G01MBF	the reciprocal of Mills' Ratio, that is the hazard rate for the Normal distribution.
G02GCF	fits generalized linear model with Poisson errors (see Aitkin and Clayton [1]).
G02GDF	fits generalized linear model with gamma errors.
G07BBF	fits Normal distribution to censored data.
G07BEF	fits Weibull distribution to censored data.
G08RBF	fits linear model using likelihood based on ranks to censored data (see Kabfleisch and Prentice [4]).
G11CAF	fits a conditional logistic model. When applied to the risk sets generated by G12ZAF the Cox proportional hazards model is fitted by exact marginal likelihood in the presence of tied observations.

4 Routines Withdrawn or Scheduled for Withdrawal

None since Mark 13.

5 References

[1] Aitkin M and Clayton D (1980) The fitting of exponential, Weibull and extreme value distributions to complex censored survival data using GLIM *Appl. Statist.* **29** 156–163

[2] Cox D R (1972) Regression models in life tables (with discussion) *J. Roy. Statist. Soc. Ser. B* **34** 187–220

[3] Gross A J and Clark V A (1975) *Survival Distributions: Reliability Applications in the Biomedical Sciences* Wiley

[4] Kalbfleisch J D and Prentice R L (1980) *The Statistical Analysis of Failure Time Data* Wiley

G12AAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G12AAF computes the Kaplan-Meier, (or product-limit), estimates of survival probabilities for a sample of failure times.

2. Specification

```
      SUBROUTINE G12AAF (N, T, IC, FREQ, IFREQ, ND, TP, P, PSIG,
     1                   IWK, IFAIL)
      INTEGER          N, IC(N), IFREQ(*), ND, IWK(N), IFAIL
      real             T(N), TP(N), P(N), PSIG(N)
      CHARACTER*1      FREQ
```

3. Description

A survivor function, $S(t)$, is the probability of surviving to at least time t with $S(t) = 1 - F(t)$, where $F(t)$ is the cumulative distribution function of the failure times. The Kaplan-Meier or product limit estimator provides an estimate of $S(t)$, $\hat{S}(t)$, from sample of failure times which may be progressively right-censored.

Let t_i, $i = 1,2,...,n_d$, be the ordered distinct failure times for the sample of observed failure/censored times, and let the number of observations in the sample that have not failed by time t_i be n_i. If a failure and a loss (censored observation) occur at the same time t_i, then the failure is treated as if it had occured slightly before time t_i and the loss as if it had occured slightly after t_i.

The Kaplan-Meier estimate of the survival probabilities is a step function which in the interval t_i to t_{i+1} is given by

$$\hat{S}(t) = \prod_{j=1}^{i}\left(\frac{n_j-d_j}{n_j}\right)$$

where d_j is the number of failures occuring at time t_j.

G12AAF computes the Kaplan-Meier estimates and the corresponding estimates of the variances, $\hat{\mathrm{var}}(\hat{S}(t))$, using Greenwood's formula,

$$\hat{\mathrm{var}}(\hat{S}(t)) = \hat{S}(t)^2\sum_{j=1}^{i}\frac{d_j}{n_j(n_j-d_j)}.$$

4. References

[1] GROSS, A.J. and CLARK, V.A.
Survival Distributions: Reliability Applications in the Biomedical Sciences.
Wiley, 1975.

[2] KALBFLEISCH, J.D. and PRENTICE, R.L.
The Statistical Analysis of Failure Time Data.
Wiley, 1980.

5. Parameters

1: N – INTEGER. *Input*

On entry: the number of failure and censored times given in T.

Constraint: N $\geq$ 2.

2: T(N) – ***real*** array. *Input*

On entry: the failure and censored times; these need not be ordered.

3: IC(N) – INTEGER array. *Input*

On entry: IC(i) contains the censoring code of the ith observation, for $i = 1,2,...,$N.

If IC(i) = 0 the ith observation is a failure time.

If IC(i) = 1 the ith observation is right-censored.

Constraint: IC(i) = 0 or 1 for $i = 1,2,...,$N.

4: FREQ – CHARACTER*1. *Input*

On entry: indicates whether frequencies are provided for each time point.

If FREQ = 'F', then frequencies are provided for each failure and censored time.

If FREQ = 'S', then the failure and censored times are considered as single observations, i.e. a frequency of 1 is assumed.

Constraint: FREQ = 'F' or 'S'.

5: IFREQ(*) – INTEGER array. *Input*

Note: if FREQ = 'F', then IFREQ must be dimensioned at least N, otherwise it can be dimensioned 1.

On entry: if FREQ = 'F', then IFREQ(i) contains the frequency of the ith observation, otherwise a frequency of 1 is assumed and IFREQ is not referenced.

Constraint: if FREQ = 'F', IFREQ(i) $\geq$ 0, for $i = 1,2,...,$N.

6: ND – INTEGER. *Output*

On exit: the number of distinct failure times, n_d.

7: TP(N) – ***real*** array. *Output*

On exit: TP(i) contains the ith ordered distinct failure time, t_i, for $i = 1,2,...,n_d$.

8: P(N) – ***real*** array. *Output*

On exit: P(i) contains the Kaplan-Meier estimate of the survival probability, $\hat{S}(t)$, for time TP(i), for $i = 1,2,...,n_d$.

9: PSIG(N) – ***real*** array. *Output*

On exit: PSIG(i) contains an estimate of the standard deviation of P(i), for $i = 1,2,...,n_d$.

10: IWK(N) – INTEGER array. *Workspace*

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N < 2.

IFAIL = 2

On entry, FREQ ≠ 'F' or 'S'.

IFAIL = 3

On entry, IC(i) ≠ 0 or 1, for some i = 1,2,...,N.

IFAIL = 4

On entry, FREQ = 'F' and IFREQ(i) < 0, for some i = 1,2,...,N.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

If there are no censored observations, $\hat{S}(t)$, reduces to the ordinary binomial estimate of the probability of survival at time t.

9. Example

The remission times for a set of 21 leukemia patients at 18 distinct time points are read in and the Kaplan-Meier estimate computed and printed. For further details see Gross and Clark [1], page 242.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G12AAF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX
      PARAMETER        (NMAX=18)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, N, ND
*     .. Local Arrays ..
      real             P(NMAX), PSIG(NMAX), T(NMAX), TP(NMAX)
      INTEGER          IC(NMAX), IFREQ(NMAX), IWK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         G12AAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G12AAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
         READ (NIN,*) (T(I),IC(I),IFREQ(I),I=1,N)
         IFAIL = 0
*
         CALL G12AAF(N,T,IC,'Frequencies',IFREQ,ND,TP,P,PSIG,IWK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Time   Survival    Standard'
         WRITE (NOUT,*) '        probability deviation'
         WRITE (NOUT,*)
         DO 20 I = 1, ND
            WRITE (NOUT,99999) TP(I), P(I), PSIG(I)
   20    CONTINUE
```

```
      END IF
      STOP
*
99999 FORMAT (1X,F6.1,F10.3,2X,F10.3)
      END
```

9.2. Program Data

```
G12AAF Example Program Data
18
6.0 1 1 6.0 0 3 7.0 0 1 9.0 1 1 10.0 0 1 10.0 1 1
11.0 1 1 13.0 0 1 16.0 0 1 17.0 1 1 19.0 1 1 20.0 1 1
22.0 0 1 23.0 0 1 25.0 1 1 32.0 1 2 34.0 1 1 35.0 1 1
```

9.3. Program Results

```
 G12AAF Example Program Results

   Time   Survival    Standard
         probability  deviation

    6.0     0.857       0.076
    7.0     0.807       0.087
   10.0     0.753       0.096
   13.0     0.690       0.107
   16.0     0.627       0.114
   22.0     0.538       0.128
   23.0     0.448       0.135
```

G12BAF – NAG Fortran Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G12BAF returns parameter estimates and other statistics that are associated with the Cox proportional hazards model for fixed covariates.

2 Specification

```
      SUBROUTINE G12BAF(OFFSET, N, M, NS, Z, LDZ, ISZ, IP, T, IC, OMEGA,
     1                  ISI, DEV, B, SE, SC, COV, RES, ND, TP, SUR,
     2                  NDMAX, TOL, MAXIT, IPRINT, WK, IWK, IFAIL)
      INTEGER           N, M, NS, LDZ, ISZ(M), IP, IC(N), ISI(*), ND,
     1                  NDMAX, MAXIT, IPRINT, IWK(2*N), IFAIL
      real              Z(LDZ,M), T(N), OMEGA(*), DEV, B(IP), SE(IP),
     1                  SC(IP), COV(IP*(IP+1)/2), RES(N), TP(NDMAX),
     2                  SUR(NDMAX,*), TOL, WK(IP*(IP+9)/2+N)
      CHARACTER*1       OFFSET
```

3 Description

The proportional hazard model relates the time to an event, usually death or failure, to a number of explanatory variables known as covariates. Some of the observations may be right censored, that is the exact time to failure is not known, only that it is greater than a known time.

Let $t_i, i = 1, \ldots, n$ be the the failure time or censored time for the ith observation with the vector of p covariates z_i. It is assumed that censoring and failure mechanisms are independent. The hazard function, $\lambda(t, z)$, is the probability that an individual with covariates z fails at time t given that the individual survived up to time t. In the Cox proportional hazards model (Cox [1]) $\lambda(t, z)$ is of the form:

$$\lambda(t,z) = \lambda_0(t)\exp(z^T\beta + \omega)$$

where λ_0 is the base-line hazard function, an unspecified function of time, β is a vector of unknown parameters and ω is a known offset.

Assuming there are ties in the failure times giving $n_d < n$ distinct failure times, $t_{(1)} < \ldots < t_{(n_d)}$ such that d_i individuals fail at $t_{(i)}$, it follows that the marginal likelihood for β is well approximated (see Kalbfleisch and Prentice, [2]) by:

$$L = \prod_{i=1}^{n_d} \frac{\exp\left(s_i^T\beta + \omega_i\right)}{\left[\sum_{l \in R(t_{(i)})} \exp\left(z_l^T\beta + \omega_l\right)\right]^{d_i}} \tag{1}$$

where s_i is the sum of the covariates of individuals observed to fail at $t_{(i)}$ and $R(t_{(i)})$ is the set of individuals at risk just prior to $t_{(i)}$, that is it is all individuals that fail or are censored at time $t_{(i)}$ along with all individuals that survive beyond time $t_{(i)}$. The maximum likelihood estimates (MLEs) of β, given by $\hat{\beta}$, are obtained by maximizing (1) using a Newton-Raphson iteration technique that includes step halving and utilizes the first and second partial derivatives of (1) which are given by equations (2) and (3) below:

$$U_j(\beta) = \frac{\partial \ln L}{\partial \beta_j} = \sum_{i=1}^{n_d}\left[s_{ji} - d_i\alpha_{ji}(\beta)\right] = 0 \tag{2}$$

for $j = 1, \ldots, p$, where s_{ji} is the jth element in the vector s_i and

$$\alpha_{ji}(\beta) = \frac{\sum_{l \in R(t_{(i)})} z_{jl}\exp\left(z_l^T\beta + \omega_l\right)}{\sum_{l \in R(t_{(i)})} \exp\left(z_l^T\beta + \omega_l\right)}.$$

Similarly,

$$I_{hj}(\beta) = -\frac{\partial^2 \ln L}{\partial\beta_h \partial\beta_j} = \sum_{i=1}^{n_d} d_i \gamma_{hji} \tag{3}$$

where

$$\gamma_{hji} = \frac{\sum_{l\in R(t_{(i)})} z_{hl} z_{jl} \exp(z_l^T \beta + \omega_l)}{\sum_{l\in R(t_{(i)})} \exp(z_l^T \beta + \omega_l)} - \alpha_{hi}(\beta)\alpha_{ji}(\beta) \qquad h,j = 1,\ldots,p.$$

$U_j(\beta)$ is the jth component of a score vector and $I_{hj}(\beta)$ is the (h,j) element of the observed information matrix $I(\beta)$ whose inverse $I(\beta)^{-1} = [I_{hj}(\beta)]^{-1}$ gives the variance-covariance matrix of β.

It should be noted that if a covariate or a linear combination of covariates is monotonically increasing or decreasing with time then one or more of the β_j's will be infinite.

If $\lambda_0(t)$ varies across ν strata, where the number of individuals in the kth stratum is n_k $k = 1,\ldots,\nu$ with $n = \sum_{k=1}^{\nu} n_k$, then rather than maximizing (1) to obtain $\hat{\beta}$, the following marginal likelihood is maximized:

$$L = \prod_{k=1}^{\nu} L_k, \tag{4}$$

where L_k is the contribution to likelihood for the n_k observations in the kth stratum treated as a single sample in (1). When strata are included the covariate coefficients are constant across strata but there is a different base-line hazard function λ_0.

The base-line survivor function associated with a failure time $t_{(i)}$, is estimated as $\exp(-\hat{H}(t_{(i)}))$, where

$$\hat{H}(t_{(i)}) = \sum_{t_{(j)} \leq t_{(i)}} \left(\frac{d_i}{\sum_{l\in R(t_{(j)})} \exp\left(z_l^T \beta + \omega_l\right)} \right), \tag{5}$$

where d_i is the number of failures at time $t_{(i)}$. The residual for the lth observation is computed as:

$$r(t_l) = \hat{H}(t_l) \exp(-z_l^T \hat{\beta} + \omega_l)$$

where $\hat{H}(t_l) = \hat{H}(t_{(i)}), t_{(i)} \leq t_l < t_{(i+1)}$. The deviance is defined as $-2\times$(logarithm of marginal likelihood). There are two ways to test whether individual covariates are significant: the differences between the deviances of nested models can be compared with the appropiate χ^2-distribution; or, the asymptotic normality of the parameter estimates can be used to form z tests by dividing the estimates by their standard errors or the score function for the model under the null hypothesis can be used to form z tests.

4 References

[1] Cox D R (1972) Regression models in life tables (with discussion) *J. Roy. Statist. Soc. Ser. B* **34** 187–220

[2] Kalbfleisch J D and Prentice R L (1980) *The Statistical Analysis of Failure Time Data* Wiley

[3] Gross A J and Clark V A (1975) *Survival Distributions: Reliability Applications in the Biomedical Sciences* Wiley

5 Parameters

1: OFFSET — CHARACTER*1 *Input*

On entry: indicates if an offset is to be used.

If OFFSET = 'Y' then an offset must be included in OMEGA.

If OFFSET = 'N' no offset is included in the model.

Constraint: OFFSET = 'Y' or 'N'.

2: N — INTEGER *Input*

On entry: the number of data points, n.

Constraint: N $\geq$ 2.

3: M — INTEGER *Input*

On entry: the number of covariates in array Z.

Constraint: M $\geq$ 1.

4: NS — INTEGER *Input*

On entry: the number of strata. If NS > 0 then the stratum for each observation must be supplied in ISI.

Constraint: NS $\geq$ 0.

5: Z(LDZ,N) — ***real*** array *Input*

On entry: the ith row must contain the covariates which are associated with the ith failure time given in T.

6: LDZ — INTEGER *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which G12BAF is called.

Constraint: LDZ $\geq$ N.

7: ISZ(M) — INTEGER array *Input*

On entry: indicates which subset of covariates is to be included in the model.

If ISZ(j) $\geq$ 1 the jth covariate is included in the model.
If ISZ(j) = 0 the jth covariate is excluded from the model and not referenced.

Constraints: ISZ(j) $\geq$ 0 and at least one and at most $n_0 - 1$ elements of ISZ must be non-zero where n_0 is the number of observations excluding any with zero value of ISI.

8: IP — INTEGER *Input*

On entry: the number of covariates included in the model as indicated by ISZ.

Constraint: IP = number of non-zero values of ISZ.

9: T(N) — ***real*** array *Input*

On entry: the vector of n failure censoring times.

10: IC(N) — INTEGER array *Input*

On entry: the status of the individual at time t given in T.

IC(i) = 0 indicates that the ith individual has failed at time T(i).
IC(i) = 1 indicates that the ith individual has been censored at time T(i).

Constraint: IC(i) = 0 or 1 for $i = 1, 2, \ldots,$ N.

11: OMEGA(*) — ***real*** array *Input*

Note: the dimension of the array OMEGA must be at least N if OFFSET = 'Y' and 1 otherwise.

On entry: if OFFSET = 'Y' the offset, $\omega_i, i = 1, 2, \ldots,$ N. Otherwise OMEGA is not referenced.

12: ISI(*) — INTEGER array *Input*

Note: the dimension of the array ISI must be at least N if NS > 0 and 1 otherwise.

On entry: if NS > 0 the stratum indicators which also allow data points to be excluded from the analysis. If NS = 0 ISI is not referenced.

ISI(i) = k indicates that the ith data point is in the kth stratum, where $k = 1,2,\ldots,$NS.
ISI(i) = 0 indicates that the ith data point is omitted from the analysis.

Constraints: if NS > 0, 0 ≤ ISI(i) ≤ NS for $i = 1,2,\ldots,$N, and more than IP values of ISI(i) > 0.

13: DEV — ***real*** *Output*

On exit: the deviance, that is $-2\times$(maximized log marginal likelihood).

14: B(IP) — ***real*** array *Input/Output*

On entry: initial estimates of the covariate coefficient parameters β. B(j) must contain the initial estimate of the coefficent of the covariate in Z corresponding to the jth non-zero value of ISZ.

Suggested values: In many cases an initial value of zero for B(j) may be used. For other suggestions see Section 8.

On exit: B(j) contains the estimate $\hat{\beta}_i$, the coefficient of the covariate stored in the ith column of Z where i is the jth non-zero value in the array ISZ.

15: SE(IP) — ***real*** array *Output*

On exit: SE(j) is the asymptotic standard error of the estimate contained in B(j) and score function in SC(j) for $j = 1,2,\ldots,$IP.

16: SC(IP) — ***real*** array *Output*

On exit: SC(j) is the value of the score function, $U_j(\beta)$, for the estimate contained in B(j).

17: COV(IP*(IP+1)/2) — ***real*** array *Output*

On exit: the variance-covariance matrix of the parameter estimates in B stored in packed form by column, i.e. the covariance between the parameter estimates given in B(i) and B(j), $j \geq i$, is stored in COV($j(j-1)/2+i$).

18: RES(N) — ***real*** array *Output*

On exit: the residuals, $r(t_l), l = 1,2,\ldots,$N.

19: ND — INTEGER *Output*

On exit: the number of distinct failure times.

20: TP(NDMAX) — ***real*** array *Output*

On exit: TP(i) contains the ith distinct failure time for $i = 1,2,\ldots,$ND.

21: SUR(NDMAX,*) — ***real*** array *Output*

Note: the second dimension of the array SUR must be at least max(NS,1) .

On exit:

if NS = 0
  SUR(i, 1) contains the estimated survival function for the ith distinct failure time

if NS > 0
  SUR(i, k) contains the estimated survival function for the ith distinct failure time in the kth stratum.

22: NDMAX — INTEGER *Input*

On entry: the first dimension of the array SUR as declared in the (sub)program from which G12BAF is called.

Constraint: NDMAX $\geq$ the number of distinct failure times. This is returned in ND.

23: TOL — ***real*** *Input*

On entry: indicates the accuracy required for the estimation. Convergence is assumed when the decrease in deviance is less than TOL $\times$ (1.0 + CurrentDeviance). This corresponds approximately to an absolute precision if the deviance is small and a relative precision if the deviance is large.

Constraint: TOL $\geq$ 10$\times$ ***machine precision***.

24: MAXIT — INTEGER *Input*

On entry: the maximum number of iterations to be used for computing the estimates. If MAXIT is set to 0 then the standard errors, score functions, variance-covariance matrix and the survival function are computed for the input value of β in B but β is not updated.

Constraint: MAXIT $\geq$ 0.

25: IPRINT — INTEGER *Input*

On entry: indicates if the printing of information on the iterations is required. If IPRINT $\leq$ 0, there is no printing, if IPRINT $\geq$ 1 then the deviance and the current estimates are printed every IPRINT iterations.

When printing occurs the output is directed to the current advisory message unit (see X04ABF).

26: WK(IP*(IP+9)/2+N) — ***real*** array *Workspace*

27: IWK(2*N) — INTEGER array *Workspace*

28: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, OFFSET $\neq$ 'Y' or 'N',
or M $<$ 1,
or N $<$ 2,
or NS $<$ 0,
or LDZ $<$ N,
or TOL $<$ 10 $\times$ ***machine precision***,
or MAXIT $<$ 0.

IFAIL = 2

On entry, ISZ(i) $<$ 0 for some i,
or the value of IP is incompatible with ISZ,
or IC(i) $\neq$ 1 or 0.

or ISI(i) < 0 or ISI(i) > NS,

or number of values of ISZ(i) > 0 is greater or equal to n_0, the number of observations excluding any with ISI(i) = 0,

or all observations are censored, i.e., IC(i) = 1 for all i,

or NDMAX is too small.

IFAIL = 3

The matrix of second partial derivatives is singular. Try different starting values or include fewer covariates.

IFAIL = 4

Overflow has been detected. Try using different starting values.

IFAIL = 5

Convergence has not been achieved in MAXIT iterations. The progress toward convergence can be examined by using a non-zero value of IPRINT. Any non-convergence may be due to a linear combination of covariates being monotonic with time.

Full results are returned.

IFAIL = 6

In the current iteration 10 step halvings have been performed without decreasing the deviance from the previous iteration. Convergence is assumed.

7 Accuracy

The accuracy is specified by TOL.

8 Further Comments

The routine uses mean centering which involves subtracting the means from the covariables prior to computation of any statistics. This helps to minimize the effect of outlying observations and accelerates convergence.

If the initial estimates are poor then there may be a problem with overflow in calculating $\exp(\beta^T z_i)$ or there may be non-convergence. Reasonable estimates can often be obtained by fitting an exponential model using G02GCF.

9 Example

The data are the remission times for two groups of leukemia patients (see Gross and Clark [3] p242). A dummy variable indicates which group they come from. An initial estimate is computed using the exponential model and then the Cox proportional hazard model is fitted and parameter estimates and the survival function are printed.

9.1 Example Text

Note: the listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G12BAF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*
*     .. Parameters ..
      INTEGER          NMAX, NDMAX, MMAX, SMAX, NIN, NOUT
      PARAMETER        (NMAX=42,NDMAX=42,MMAX=2,SMAX=1,NIN=5,NOUT=6)
```

```
*     .. Local Scalars ..
      real             DEV, TOL
      INTEGER          I, IDF, IFAIL, IP, IPRINT, IRANK, J, LDZ, M,
     +                 MAXIT, N, ND, NS
*     .. Local Arrays ..
      real             B(MMAX), COV(MMAX*(MMAX+1)/2), OMEGA(NMAX),
     +                 RES(NMAX), SC(MMAX), SE(MMAX), SUR(NDMAX,SMAX),
     +                 T(NMAX), TP(NDMAX), V(NMAX,MMAX+7),
     +                 WK(MMAX*(MMAX+9)/2+NMAX), Y(NMAX), Z(NMAX,MMAX)
      INTEGER          IC(NMAX), ISI(NMAX), ISZ(MMAX), IWK(2*NMAX)
*     .. External Subroutines ..
      EXTERNAL         G02GCF, G12BAF
*     .. Intrinsic Functions ..
      INTRINSIC        real, LOG, MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G12BAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, M, NS, MAXIT, IPRINT
      IF (N.LE.NMAX .AND. M.LE.MMAX) THEN
         IF (NS.GT.0) THEN
            DO 20 I = 1, N
               READ (NIN,*) T(I), (Z(I,J),J=1,M), IC(I), ISI(I)
   20       CONTINUE
         ELSE
            DO 40 I = 1, N
               READ (NIN,*) T(I), (Z(I,J),J=1,M), IC(I)
   40       CONTINUE
         END IF
         READ (NIN,*) (ISZ(I),I=1,M), IP
         LDZ = NMAX
         TOL = 0.00005e0
         DO 60 I = 1, N
            Y(I) = 1.0e0 - real(IC(I))
            V(I,7) = LOG(T(I))
   60    CONTINUE
         IFAIL = -1
         CALL G02GCF('L','M','Y','U',N,Z,LDZ,M,ISZ,IP+1,Y,RES,0.0e0,DEV,
     +               IDF,B,IRANK,SE,COV,V,NMAX,TOL,MAXIT,0,0.0e0,WK,
     +               IFAIL)
         DO 80 I = 1, IP
            B(I) = B(I+1)
   80    CONTINUE
         IF (IRANK.NE.IP+1) THEN
            WRITE (NOUT,*) ' WARNING: covariates not of full rank'
         END IF
         IFAIL = 0
*
         CALL G12BAF('No-offset',N,M,NS,Z,LDZ,ISZ,IP,T,IC,OMEGA,ISI,DEV,
     +               B,SE,SC,COV,RES,ND,TP,SUR,NDMAX,TOL,MAXIT,IPRINT,
     +               WK,IWK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Parameter      Estimate',
     +     '        Standard Error'
         WRITE (NOUT,*)
         DO 100 I = 1, IP
            WRITE (NOUT,99999) I, B(I), SE(I)
```

```
  100    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,99998) ' Deviance = ', DEV
         WRITE (NOUT,*)
         WRITE (NOUT,*) '    Time     Survivor Function'
         WRITE (NOUT,*)
         NS = MAX(NS,1)
         DO 120 I = 1, ND
            WRITE (NOUT,99997) TP(I), (SUR(I,J),J=1,NS)
  120    CONTINUE
      END IF
      STOP
*
99999 FORMAT (I6,2(10X,F8.4))
99998 FORMAT (A,e13.4)
99997 FORMAT (F10.0,3(5X,F8.4))
      END
```

9.2 Example Data

```
G12BAF Example Program Data

42 1 0 20 0

 1 0 0
 1 0 0
 2 0 0
 2 0 0
 3 0 0
 4 0 0
 4 0 0
 5 0 0
 5 0 0
 8 0 0
 8 0 0
 8 0 0
 8 0 0
11 0 0
11 0 0
12 0 0
12 0 0
15 0 0
17 0 0
22 0 0
23 0 0
 6 1 0
 6 1 0
 6 1 0
 7 1 0
10 1 0
13 1 0
16 1 0
22 1 0
23 1 0
 6 1 1
 9 1 1
10 1 1
11 1 1
```

```
17 1 1
19 1 1
20 1 1
25 1 1
32 1 1
32 1 1
34 1 1
35 1 1
  1   1
```

9.3 Example Results

```
G12BAF Example Program Results

 Parameter      Estimate       Standard Error

    1           -1.5091            0.4096

Deviance =    0.1728E+03

    Time     Survivor Function

      1.         0.9640
      2.         0.9264
      3.         0.9065
      4.         0.8661
      5.         0.8235
      6.         0.7566
      7.         0.7343
      8.         0.6506
     10.         0.6241
     11.         0.5724
     12.         0.5135
     13.         0.4784
     15.         0.4447
     16.         0.4078
     17.         0.3727
     22.         0.2859
     23.         0.1908
```

G12ZAF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G12ZAF creates the risk sets associated with the Cox proportional hazards model for fixed covariates.

2 Specification

```
      SUBROUTINE G12ZAF(N, M, NS, Z, LDZ, ISZ, IP, T, IC, ISI, NUM, IXS,
     1                  NXS, X, MXN, ID, ND, TP, IRS, IFAIL)
      real              Z(LDZ,M), T(N), X(MXN,IP), TP(N)
      INTEGER           N, M, NS, LDZ, ISZ(M), IP, IC(N), ISI(*), NUM,
     1                  IXS(MXN), NXS, MXN, ID(MXN), ND, IRS(N), IFAIL
```

3 Description

The Cox proportional hazards model [2] relates the time to an event, usually death or failure, to a number of explanatory variables known as covariates. Some of the observations may be right censored, that is, the exact time to failure is not known, only that it is greater than a known time.

Let t_i for $i = 1, 2, \ldots, n$ be the the failure time or censored time for the ith observation with the vector of p covariates z_i. It is assumed that censoring and failure mechanisms are independent. The hazard function, $\lambda(t, z)$, is the probability that an individual with covariates z fails at time t given that the individual survived up to time t. In the Cox proportional hazards model [2], $\lambda(t, z)$ is of the form:

$$\lambda(t, z) = \lambda_0(t) \exp(z^T \beta)$$

where λ_0 is the base-line hazard function, an unspecified function of time and β is a vector of unknown parameters. As λ_0 is unknown, the parameters β are estimated using the conditional or marginal likelihood. This involves considering the covariate values of all subjects that are at risk at the time when a failure occurs. The probability that the subject that failed had their observed set of covariate values is computed.

The risk set at a failure time consists of those subjects that fail or are censored at that time and those who survive beyond that time. As risk sets are computed for every distinct failure time, it should be noted that the combined risk sets may be considerably larger than the original data. If the data can be considered as coming from different strata such that λ_0 varies from strata to strata but β remains constant; then G12ZAF will return a factor that indicates to which risk set/strata each member of the risk sets belongs rather than just to which risk set.

Given the risk sets the Cox proportional hazards model can then be fitted using a Poisson generalised linear model (G02GCF with G04EAF to compute dummy variables) using Breslow's approximation for ties [1]. This will give the same fit as G12BAF. If the exact treatment of ties in discrete time is required, as given by Cox [2], then the model is fitted as a conditional logistic model using G11CAF.

4 References

[1] Breslow N E (1974) Covariate analysis of censored survival data *Biometrics* **30** 89–99

[2] Cox D R (1972) Regression models in life tables (with discussion) *J. Roy. Statist. Soc. Ser. B* **34** 187–220

[3] Gross A J and Clark V A (1975) *Survival Distributions: Reliability Applications in the Biomedical Sciences* Wiley

5 Parameters

1: N — INTEGER *Input*

On entry: the number of data points, n.

Constraint: $N \geq 2$.

2: M — INTEGER *Input*

On entry: the number of covariates in array Z.

Constraint: $M \geq 1$.

3: NS — INTEGER *Input*

On entry: the number of strata. If $NS > 0$ then the stratum for each observation must be supplied in ISI.

Constraint: $NS \geq 0$.

4: Z(LDZ,N) — ***real*** array *Input*

On entry: the ith row must contain the covariates which are associated with the ith failure time given in T.

5: LDZ — INTEGER *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which G12ZAF is called.

Constraint: $LDZ \geq N$.

6: ISZ(M) — INTEGER array *Input*

On entry: indicates which subset of covariates are to be included in the model.

If $ISZ(j) \geq 1$ the jth covariate is included in the model.
If $ISZ(j) = 0$ the jth covariate is excluded from the model and not referenced.

Constraints: $ISZ(j) \geq 0$ and at least one value must be non-zero.

7: IP — INTEGER *Input*

On entry: the number of covariates included in the model, p, as indicated by ISZ.

Constraint: IP = number of non-zero values of ISZ.

8: T(N) — ***real*** array *Input*

On entry: the vector of n failure censoring times.

9: IC(N) — INTEGER array *Input*

On entry: the status of the individual at time t given in T.

$IC(i) = 0$ indicates that the ith individual has failed at time $T(i)$.
$IC(i) = 1$ indicates that the ith individual has been censored at time $T(i)$.

Constraint: $IC(i) = 0$ or 1 for $i = 1, 2, \ldots, N$.

10: ISI(*) — INTEGER array *Input*

Note: the dimension of the array ISI must be at least N if $NS > 0$ and 1 otherwise .

On entry: if $NS > 0$, the stratum indicators which also allow data points to be excluded from the analysis. If $NS = 0$, ISI is not referenced.

$ISI(i) = k$ indicates that the ith data point is in the kth stratum, where $k = 1, 2, \ldots, NS$.
$ISI(i) = 0$ indicates that the ith data point is omitted from the analysis.

Constraints: if $NS > 0$, $0 \leq ISI(i) \leq NS$ for $i = 1, 2, \ldots, N$.

11: NUM — INTEGER *Output*

On exit: the number of values in the combined risk sets.

12: IXS(MXN) — INTEGER array *Output*

On exit: the factor giving the risk sets/strata for the data in X and ID. If NS = 0 or 1, IXS(i) = l for members of the lth risk set. If NS > 1, IXS(i) = (j − 1)∗ND + l for the observations in the lth risk set for the jth strata.

13: NXS — INTEGER *Output*

On exit: the number of levels for the risk sets/strata factor given in IXS.

14: X(MXN,IP) — ***real*** array *Output*

On exit: the first NUM rows contain the values of the covariates for the members of the risk sets.

15: MXN — INTEGER *Input*

On entry: the first dimension of the array X and the dimension of the arrays IXS and ID as declared in the (sub)program from which G12CAF is called.

Constraint: MXN must be sufficiently large for the arrays to contain the expanded risk sets. The size will depend on the pattern of failures times and censored times. The minimum value will be returned in NUM unless the routine exits with IFAIL = 1 or 2.

16: ID(MXN) — INTEGER array *Output*

On exit: indicates if the member of the risk set given in X failed. ID(i) = 1 if the member of the risk set failed at the time defining the risk set and ID(i) = 0 otherwise.

17: ND — INTEGER *Output*

On exit: the number of distinct failure times, i.e., the number of risk sets.

18: TP(N) — ***real*** array *Output*

On exit: TP(i) contains the ith disitinct failure time for $i = 1,2,\ldots,$ND.

19: IRS(N) — INTEGER array *Output*

On exit: indicates rows in X and elements in IXS and ID corresponding to the risk sets. The first risk set corresponding to failure time TP(1) is given by rows 1 to IRS(1). The lth risk set is given by rows ID($l-1$) + 1 to ID(l) for $l = 1,2,\ldots,$ND.

20: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, M < 1,
or N < 2,
or NS < 0,
or LDZ < N.

IFAIL = 2

On entry, ISZ(i) < 0 for some i,
or the value of IP is incompatible with ISZ,
or IC(i) $\neq$ 1 or 0.
or NS > 0 and ISI(i) < 0,
or NS > 1 and ISI(i) > NS.

IFAIL = 3

MXN is too small, the minimum value is returned in NUM.

7 Accuracy

Not applicable.

8 Further Comments

When there are strata present, i.e., NS > 1, not all the NXS groups may be present.

9 Example

The data are the remission times for two groups of leukemia patients (see Gross and Clark [3] p242). A dummy variable indicates which group they come from. The risk sets are computed using G12ZAF and the Cox's proportional hazard model is fitted using G11CAF.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G12ZAF Example Program Text.
*     Mark 19 Release. NAG Copyright 1999.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX, MLWK, MNRS
      PARAMETER        (NMAX=500,MMAX=20,MLWK=10000,MNRS=1000)
*     .. Local Scalars ..
      real             DEV, TOL
      INTEGER          I, IFAIL, IP, IPRINT, J, LDZ, LWK, M, MAXIT, MXN,
     +                 N, ND, NS, NUM, NXS
*     .. Local Arrays ..
      real             B(MMAX), COV(MMAX*(MMAX+1)/2), SC(MMAX),
     +                 SE(MMAX), T(NMAX), TP(NMAX), WK(MLWK),
     +                 X(MNRS,MMAX), Z(NMAX,MMAX)
      INTEGER          IC(NMAX), ID(MNRS), IRS(NMAX), ISI(NMAX),
     +                 ISZ(MMAX), IXS(NMAX), NCA(NMAX), NCT(NMAX)
*     .. External Subroutines ..
      EXTERNAL         G11CAF, G12ZAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G12ZAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, M, NS, MAXIT, IPRINT
      IF (N.LE.NMAX .AND. M.LE.MMAX) THEN
```

```
         IF (NS.GT.0) THEN
            DO 20 I = 1, N
               READ (NIN,*) T(I), (Z(I,J),J=1,M), IC(I), ISI(I)
   20       CONTINUE
         ELSE
            DO 40 I = 1, N
               READ (NIN,*) T(I), (Z(I,J),J=1,M), IC(I)
   40       CONTINUE
         END IF
         READ (NIN,*) (ISZ(I),I=1,M), IP
         LDZ = NMAX
         MXN = MNRS
*
         IFAIL = 0
*
         CALL G12ZAF(N,M,NS,Z,LDZ,ISZ,IP,T,IC,ISI,NUM,IXS,NXS,X,MXN,ID,
     +               ND,TP,IRS,IFAIL)
*
         TOL = 1.0e-5
         LWK = MLWK
         READ (NIN,*) (B(I),I=1,IP)
*
         IFAIL = 0
*
         CALL G11CAF(NUM,IP,NXS,X,MXN,ISZ,IP,ID,IXS,DEV,B,SE,SC,COV,NCA,
     +               NCT,TOL,MAXIT,IPRINT,WK,LWK,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Parameter      Estimate',
     +     '       Standard Error'
         WRITE (NOUT,*)
         DO 60 I = 1, IP
            WRITE (NOUT,99999) I, B(I), SE(I)
   60    CONTINUE
      END IF
*
99999 FORMAT (I6,2(10X,F8.4))
      END
```

9.2 Program Data

```
G12ZAF Example Program Data

42 1 0 20 0

 1 0 0
 1 0 0
 2 0 0
 2 0 0
 3 0 0
 4 0 0
 4 0 0
 5 0 0
 5 0 0
 8 0 0
 8 0 0
 8 0 0
 8 0 0
11 0 0
11 0 0
12 0 0
12 0 0
15 0 0
17 0 0
22 0 0
23 0 0
 6 1 0
 6 1 0
 6 1 0
 7 1 0
10 1 0
13 1 0
16 1 0
22 1 0
23 1 0
 6 1 1
 9 1 1
10 1 1
11 1 1
17 1 1
19 1 1
20 1 1
25 1 1
32 1 1
32 1 1
34 1 1
35 1 1

  1   1

0.0 0.0
```

9.3 Program Results

```
G12ZAF Example Program Results

 Parameter      Estimate       Standard Error

     1           1.6282            0.4331
```

Chapter G13 – Time Series Analysis

Note. Please refer to the Users' Note for your implementation to check that a routine is available.

Routine Name	Mark of Introduction	Purpose
G13AAF	9	Univariate time series, seasonal and non-seasonal differencing
G13ABF	9	Univariate time series, sample autocorrelation function
G13ACF	9	Univariate time series, partial autocorrelations from autocorrelations
G13ADF	9	Univariate time series, preliminary estimation, seasonal ARIMA model
G13AEF	9	Univariate time series, estimation, seasonal ARIMA model (comprehensive)
G13AFF	9	Univariate time series, estimation, seasonal ARIMA model (easy-to-use)
G13AGF	9	Univariate time series, update state set for forecasting
G13AHF	9	Univariate time series, forecasting from state set
G13AJF	10	Univariate time series, state set and forecasts, from fully specified seasonal ARIMA model
G13ASF	13	Univariate time series, diagnostic checking of residuals, following G13AEF or G13AFF
G13AUF	14	Computes quantities needed for range-mean or standard deviation-mean plot
G13BAF	10	Multivariate time series, filtering (pre-whitening) by an ARIMA model
G13BBF	11	Multivariate time series, filtering by a transfer function model
G13BCF	10	Multivariate time series, cross-correlations
G13BDF	11	Multivariate time series, preliminary estimation of transfer function model
G13BEF	11	Multivariate time series, estimation of multi-input model
G13BGF	11	Multivariate time series, update state set for forecasting from multi-input model
G13BHF	11	Multivariate time series, forecasting from state set of multi-input model
G13BJF	11	Multivariate time series, state set and forecasts from fully specified multi-input model
G13CAF	10	Univariate time series, smoothed sample spectrum using rectangular, Bartlett, Tukey or Parzen lag window
G13CBF	10	Univariate time series, smoothed sample spectrum using spectral smoothing by the trapezium frequency (Daniell) window
G13CCF	10	Multivariate time series, smoothed sample cross spectrum using rectangular, Bartlett, Tukey or Parzen lag window
G13CDF	10	Multivariate time series, smoothed sample cross spectrum using spectral smoothing by the trapezium frequency (Daniell) window
G13CEF	10	Multivariate time series, cross amplitude spectrum, squared coherency, bounds, univariate and bivariate (cross) spectra
G13CFF	10	Multivariate time series, gain, phase, bounds, univariate and bivariate (cross) spectra
G13CGF	10	Multivariate time series, noise spectrum, bounds, impulse response function and its standard error
G13DBF	11	Multivariate time series, multiple squared partial autocorrelations
G13DCF	12	Multivariate time series, estimation of VARMA model
G13DJF	15	Multivariate time series, forecasts and their standard errors
G13DKF	15	Multivariate time series, updates forecasts and their standard errors
G13DLF	15	Multivariate time series, differences and/or transforms (for use before G13DCF)
G13DMF	15	Multivariate time series, sample cross-correlation or cross-covariance matrices

G13DNF	15	Multivariate time series, sample partial lag correlation matrices, χ^2 statistics and significance levels
G13DPF	16	Multivariate time series, partial autoregression matrices
G13DSF	13	Multivariate time series, diagnostic checking of residuals, following G13DCF
G13DXF	15	Calculates the zeros of a vector autoregressive (or moving average) operator
G13EAF	17	Combined measurement and time update, one iteration of Kalman filter, time-varying, square root covariance filter
G13EBF	17	Combined measurement and time update, one iteration of Kalman filter, time-invariant, square root covariance filter

Chapter G13

Time Series Analysis

Contents

1 Scope of the Chapter

This chapter provides facilities for investigating and modelling the statistical structure of series of observations collected at equally spaced points in time. The models may then be used to forecast the series.

The chapter is divided into methods for

(a) univariate analysis, where consideration is given to the structure within a single series, and
(b) multivariate analysis in which the interdependence of two or more series may be investigated. This includes models with a single output series depending on a number of input series as well as multivariate models with each series considered on an equal footing.

For both univariate and multivariate methods, there is a further division into

(i) time domain methods, which model relations between series values separated by various time lags, and
(ii) frequency domain or spectral methods which interpret time series in terms of component sine waves of various frequencies.

2 Background to the Problems

2.1 Univariate Time Domain Analysis

Let the given time series be $x_1, x_2, \ldots, x_n$ where n is its length. The structure which is intended to be investigated, and which may be most evident to the eye in a graph of the series, can be broadly described as

(a) trends – linear or possibly higher-order polynomial;
(b) seasonal patterns, associated with fixed integer seasonal periods. The presence of such seasonality and the period will normally be known *a priori*. The pattern may be fixed, or slowly varying from one season to another;
(c) cycles, or waves of stable amplitude and period p (from peak to peak). The period is not necessarily integer, the corresponding absolute frequency (cycles/time unit) being $f = 1/p$ and angular frequency $\omega = 2\pi f$. The cycle may be of pure sinusoidal form like $\sin(\omega t)$, or the presence of higher harmonic terms may be indicated, e.g. by asymmetry in the wave form;
(d) quasi-cycles, i.e., waves of fluctuating period and amplitude; and
(e) irregular statistical fluctuations and swings about the overall mean or trend.

Trends, seasonal patterns, and cycles might be regarded as **deterministic** components following fixed mathematical equations, and the quasi-cycles and other statistical fluctuations as **stochastic** and describable by short-term correlation structure. For a finite data set it is not always easy to discriminate between these two types, and a common description using the class of autoregressive integrated moving-average (ARIMA) models is now widely used. The form of these models is that of difference equations (or recurrence relations) relating present and past values of the series. The user is referred to Box and Jenkins [2] for a thorough account of these models and how to use them. We follow their notation and outline the recommended steps in ARIMA model building for which routines are available.

2.1.1 Transformations

If the variance of the observations in the series is not constant across the range of observations it may be useful to apply a variance-stabilizing transformation to the series. A common situation is for the variance to increase with the magnitude of the observations and in this case typical transformations used are the log or square root transformation. A range–mean or standard deviation–mean plot provides a quick and easy way of detecting non-constant variance and of choosing, if required, a suitable transformation. This is a plot of the range or standard deviation of successive groups of observations against their means.

2.1.2 Differencing operations

These may be used to simplify the structure of a time series.

First-order differencing, i.e., forming the new series

$$\nabla x_t = x_t - x_{t-1}$$

will remove a linear trend. First-order seasonal differencing

$$\nabla_s x_t = x_t - x_{t-s}$$

eliminates a fixed seasonal pattern.

These operations reflect the fact that it is often appropriate to model a time series in terms of changes from one value to another. Differencing is also therefore appropriate when the series has something of the nature of a random walk, which is by definition the accumulation of independent changes.

Differencing may be applied repeatedly to a series giving

$$w_t = \nabla^d \nabla_s^D x_t$$

where d and D are the orders of differencing. The derived series w_t will be shorter, of length $N = n - d - s{\times}D$ and extend for $t = 1 + d + s{\times}D, \ldots, n$.

2.1.3 Sample autocorrelations

Given that a series has (possibly as a result of simplifying by differencing operations) a homogeneous appearance throughout its length, fluctuating with approximately constant variance about an overall mean level, it is appropriate to assume that its statistical properties are **stationary**. For most purposes the correlations ρ_k between terms x_t, x_{t+k} or w_t, w_{t+k} separated by lag k give an adequate description of the statistical structure and are estimated by the sample autocorrelation function (acf) r_k, for $k = 1, 2, \ldots$.

As described by Box and Jenkins [2], these may be used to indicate which particular ARIMA model may be appropriate.

2.1.4 Partial autocorrelations

The information in the autocorrelations ρ_k may be presented in a different light by deriving from them the coefficients of the partial autocorrelation function (pacf) $\phi_{k,k}$, for $k = 1, 2, \ldots$. $\phi_{k,k}$ measures the correlation between x_t and x_{t+k} conditional upon the intermediate values $x_{t+1}, x_{t+2}, \ldots, x_{t+k-1}$. The corresponding sample values $\hat{\phi}_{k,k}$ give further assistance in the selection of ARIMA models.

Both acf and pacf may be rapidly computed, particularly in comparison with the time taken to estimate ARIMA models.

2.1.5 Finite lag predictor coefficients and error variances

The partial autocorrelation coefficient $\phi_{k,k}$ is determined as the final parameter in the minimum variance predictor of x_t in terms of $x_{t-1}, x_{t-2}, \ldots, x_{t-k}$,

$$x_t = \phi_{k,1} x_{t-1} + \phi_{k,2} x_{t-2} + \cdots + \phi_{k,k} x_{t-k} + e_{k,t}$$

where $e_{k,t}$ is the prediction error, and the first subscript k of $\phi_{k,i}$ and $e_{k,t}$ emphasises the fact that the parameters will alter as k increases. Moderately good estimates $\hat{\phi}_{k,i}$ of $\phi_{k,i}$ are obtained from the sample acf, and after calculating the pacf up to lag L, the successive values $v_1, v_2, \ldots, v_L$ of the prediction error variance estimates, $v_k = \mathrm{var}(e_{k,t})$, are available, together with the final values of the coefficients $\hat{\phi}_{k,1}, \hat{\phi}_{k,2}, \ldots, \hat{\phi}_{k,L}$. If x_t has non-zero mean, $\bar{x}$, it is adequate to use $x_t - \bar{x}$, in place of x_t in the prediction equation.

Although Box and Jenkins [2] do not place great emphasis on these prediction coefficients, their use is advocated for example by Akaike [1], who recommends selecting an optimal order of the predictor as the lag for which the final prediction error (FPE) criterion $(1 + k/n)(1 - k/n)^{-1} v_k$ is a minimum.

2.1.6 ARIMA models

The correlation structure in stationary time series may often be represented by a model with a small number of parameters belonging to the autoregressive moving-average (ARMA) class. If the stationary series w_t has been derived by differencing from the original series x_t, then x_t is said to follow an ARIMA model. Taking $w_t = \nabla^d x_t$, the (non-seasonal) ARIMA (p, d, q) model with p autoregressive parameters $\phi_1, \phi_2, \ldots, \phi_p$ and q moving-average parameters $\theta_1, \theta_2, \ldots, \theta_q$, represents the structure of w_t by the equation

$$w_t = \phi_1 w_{t-1} + \cdots + \phi_p w_{t-p} + a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q}, \tag{1}$$

where a_t is an uncorrelated series (white noise) with mean 0 and constant variance σ_a^2. If w_t has a non-zero mean c, then this is allowed for by replacing $w_t, w_{t-1}, \ldots$ by $w_t - c, w_{t-1} - c, \ldots$ in the model. Although c is often estimated by the sample mean of w_t this is not always optimal.

A series generated by this model will only be stationary provided restrictions are placed on $\phi_1, \phi_2, \ldots, \phi_p$ to avoid unstable growth of w_t. These are called **stationarity** constraints. The series a_t may also be usefully interpreted as the linear **innovations** in x_t (and in w_t), i.e., the error if x_t were to be predicted using the information in all past values $x_{t-1}, x_{t-2}, \ldots$, provided also that $\theta_1, \theta_2, \ldots, \theta_q$ satisfy **invertibility** constraints. This allows the series a_t to be regenerated by rewriting the model equation as

$$a_t = w_t - \phi_1 w_{t-1} - \ldots - \phi_p w_{t-p} + \theta_1 a_{t-1} + \ldots + \theta_q a_{t-q}. \tag{2}$$

For a series with short-term correlation only, i.e., r_k is not significant beyond some low lag q (see Box and Jenkins [2] for the statistical test), then the pure moving-average model MA(q) is appropriate, with no autoregressive parameters, i.e., $p = 0$.

Autoregressive parameters are appropriate when the acf pattern decays geometrically, or with a damped sinusoidal pattern which is associated with quasi-periodic behaviour in the series. If the sample pacf $\hat{\phi}_{k,k}$ is significant only up to some low lag p, then a pure autoregressive model AR(p) is appropriate, with $q = 0$. Otherwise moving-average terms will need to be introduced, as well as autoregressive terms.

The seasonal ARIMA (p, d, q, P, D, Q, s) model allows for correlation at lags which are multiples of the seasonal period s. Taking $w_t = \nabla^d \nabla_s^D x_t$, the series is represented in a two-stage manner via an intermediate series e_t

$$w_t = \Phi_1 w_{t-s} + \cdots + \Phi_P w_{t-s\times P} + e_t - \Theta_1 e_{t-s} - \cdots - \Theta_Q e_{t-s\times Q} \tag{3}$$

$$e_t = \phi_1 e_{t-1} + \cdots + \phi_p e_{t-p} + a_t - \theta_1 a_{t-1} - \cdots - \theta_q a_{t-q} \tag{4}$$

where Φ_i, Θ_i are the seasonal parameters and P, Q are the corresponding orders. Again, w_t may be replaced by $w_t - c$.

2.1.7 ARIMA model estimation

In theory, the parameters of an ARIMA model are determined by a sufficient number of autocorrelations $\rho_1, \rho_2, \ldots$. Using the sample values $r_1, r_2, \ldots$ in their place it is usually (but not always) possible to solve for the corresponding ARIMA parameters.

These are rapidly computed but are not fully efficient estimates, particularly if moving-average parameters are present. They do provide useful **preliminary** values for an efficient but relatively slow iterative method of estimation. This is based on the least-squares principle by which parameters are chosen to minimize the sum of squares of the innovations a_t, which are regenerated from the data using (2), or the reverse of (3) and (4) in the case of seasonal models.

Lack of knowledge of terms on the right-hand side of (2), when $t = 1, 2, \ldots, \max(p, q)$, is overcome by introducing q unknown series values $w_0, w_1, \ldots, w_{1-q}$ which are estimated as nuisance parameters, and using correction for transient errors due to the autoregressive terms. If the data $w_1, w_2, \ldots, w_N = w$ is viewed as a single sample from a multivariate Normal density whose covariance matrix V is a function of the ARIMA model parameters, then the exact likelihood of the parameters is

$$-\tfrac{1}{2}\log|V| - \tfrac{1}{2}w^T V^{-1} w.$$

The least-squares criterion as outlined above is equivalent to using the quadratic form

$$QF = w^T V^{-1} w$$

as an objective function to be minimized. Neglecting the term $-\frac{1}{2}\log|V|$ yields estimates which differ very little from the exact likelihood except in small samples, or in seasonal models with a small number of whole seasons contained in the data. In these cases bias in moving-average parameters may cause them to stick at the boundary of their constraint region, resulting in failure of the estimation method.

Approximate standard errors of the parameter estimates and the correlations between them are available after estimation.

The model residuals, $\hat{a}_t$, are the innovations resulting from the estimation and are usually examined for the presence of autocorrelation as a check on the adequacy of the model.

2.1.8 ARIMA model forecasting

An ARIMA model is particularly suited to extrapolation of a time series. The model equations are simply used for $t = n+1, n+2, \ldots$ replacing the unknown future values of a_t by zero. This produces future values of w_t, and if differencing has been used this process is reversed (the so-called integration part of ARIMA models) to construct future values of x_t.

Forecast error limits are easily deduced.

This process requires knowledge only of the model orders and parameters together with a limited set of the terms $a_{t-i}, e_{t-i}, w_{t-i}, x_{t-i}$ which appear on the right-hand side of the models (3) and (4) (and the differencing equations) when $t = n$. It does not require knowledge of the whole series.

We call this the state set. It is conveniently constituted after model estimation. Moreover, if new observations $x_{n+1}, x_{n+2}, \ldots$ come to hand, then the model equations can easily be used to update the state set before constructing forecasts from the end of the new observations. This is particularly useful when forecasts are constructed on a regular basis. The new innovations $a_{n+1}, a_{n+2}, \ldots$ may be compared with the residual standard deviation, σ_a, of the model used for forecasting, as a check that the model is continuing to forecast adequately.

2.2 Univariate Spectral Analysis

In describing a time series using spectral analysis the fundamental components are taken to be sinusoidal waves of the form $R\cos(\omega t + \phi)$, which for a given angular frequency ω, $0 \leq \omega \leq \pi$, is specified by its amplitude $R > 0$ and phase ϕ, $0 \leq \phi < 2\pi$. Thus in a time series of n observations it is not possible to distinguish more than $n/2$ independent sinusoidal components. The frequency range $0 \leq \omega \leq \pi$ is limited to a shortest wavelength of two sampling units because any wave of higher frequency is indistinguishable upon sampling (or is aliased with) a wave within this range. Spectral analysis follows the idea that for a series made up of a **finite** number of sine waves the amplitude of any component at frequency ω is given to order $1/n$ by

$$R^2 = \left(\frac{1}{n^2}\right)\left|\sum_{t=1}^{n} x_t e^{i\omega t}\right|^2 .$$

2.2.1 The sample spectrum

For a series $x_1, x_2, \ldots, x_n$ this is defined as

$$f^*(\omega) = \left(\frac{1}{2n\pi}\right)\left|\sum_{t=1}^{n} x_t e^{i\omega t}\right|^2 ,$$

the scaling factor now being chosen in order that

$$2\int_0^{\pi} f^*(\omega)d\omega = \sigma_x^2,$$

i.e., the spectrum indicates how the sample variance (σ_x^2) of the series is distributed over components in the frequency range $0 \leq \omega \leq \pi$.

It may be demonstrated that $f^*(\omega)$ is equivalently defined in terms of the sample autocorrelation function (acf) r_k of the series as

$$f^*(\omega) = \left(\frac{1}{2\pi}\right)\left(c_0 + 2\sum_{k=1}^{n-1} c_k \cos k\omega\right) ,$$

where $c_k = \sigma_x^2 r_k$ are the sample autocovariance coefficients.

If the series x_t does contain a **deterministic** sinusoidal component of amplitude R, this will be revealed in the sample spectrum as a sharp peak of approximate width π/n and height $(n/2\pi)R^2$. This is called the discrete part of the spectrum, the variance R^2 associated with this component being in effect concentrated at a single frequency.

If the series x_t has no deterministic components, i.e., is purely **stochastic** being stationary with acf r_k, then with increasing sample size the expected value of $f^*(\omega)$ converges to the theoretical spectrum – the **continuous** part

$$f(\omega) = \left(\frac{1}{2\pi}\right)\left(\gamma_0 + 2\sum_{k=1}^{\infty}\gamma_k \cos(\omega k)\right),$$

where γ_k are the theoretical autocovariances.

The sample spectrum does **not** however converge to this value but at each frequency point fluctuates about the theoretical spectrum with an exponential distribution, being independent at frequencies separated by an interval of $2\pi/n$ or more. Various devices are therefore employed to smooth the sample spectrum and reduce its variability. Much of the strength of spectral analysis derives from the fact that the error limits are multiplicative so that features may still show up as significant in a part of the spectrum which has a generally low level, whereas they are completely masked by other components in the original series. The spectrum can help to distinguish deterministic cyclical components from the stochastic quasi-cycle components which produce a broader peak in the spectrum. (The deterministic components can be removed by regression and the remaining part represented by an ARIMA model).

A large discrete component in a spectrum can distort the continuous part over a large frequency range surrounding the corresponding peak. This may be alleviated at the cost of slightly broadening the peak by tapering a portion of the data at each end of the series with weights which decay smoothly to zero. It is usual to correct for the mean of the series and for any linear trend by simple regression, since they would similarly distort the spectrum.

2.2.2 Spectral smoothing by lag window

The estimate is calculated directly from the sample covariances c_k as

$$f(\omega) = \left(\frac{1}{2\pi}\right)\left(c_0 + 2\sum_{k=1}^{M-1} w_k c_k \cos k\omega\right),$$

the smoothing being induced by the lag window weights w_k which extend up to a **truncation lag** M which is generally much less than n. The smaller the value of M, the greater the degree of smoothing, the spectrum estimates being independent only at a wider frequency separation indicated by the **bandwidth** b which is proportional to $1/M$. It is wise, however, to calculate the spectrum at intervals appreciably less than this. Although greater smoothing narrows the error limits, it can also distort the spectrum, particularly by flattening peaks and filling in troughs.

2.2.3 Direct spectral smoothing

The unsmoothed sample spectrum is calculated for a fine division of frequencies, then averaged over intervals centred on each frequency point for which the smoothed spectrum is required. This is usually at a coarser frequency division. The bandwidth corresponds to the width of the averaging interval.

2.3 Linear Lagged Relationships Between Time Series

We now consider the context in which one time series, called the dependent or output series $y_1, y_2, \ldots, y_n$, is believed to depend on one or more explanatory or input series, e.g. $x_1, x_2, \ldots, x_n$. This dependency may follow a simple linear regression, e.g.

$$y_t = v x_t + n_t$$

or more generally may involve lagged values of the input

$$y_t = v_0 x_t + v_1 x_{t-1} + v_2 x_{t-2} + \cdots + n_t.$$

The sequence $v_0, v_1, v_2, \ldots$ is called the **impulse response function** (IRF) of the relationship. The term n_t represents that part of y_t which cannot be explained by the input, and it is assumed to follow a univariate ARIMA model. We call n_t the (output) noise component of y_t, and it includes any constant term in the relationship. It is assumed that the input series, x_t, and the noise component, n_t, are independent.

The part of y_t which is explained by the input is called the input component z_t:

$$z_t = v_0 x_t + v_1 x_{t-1} + v_2 x_{t-2} + \cdots$$

so $y_t = z_t + n_t$.

The eventual aim is to model both these components of y_t on the basis of observations of $y_1, y_2, \ldots, y_n$ and $x_1, x_2, \ldots, x_n$. In applications to forecasting or control both components are important. In general there may be more than one input series, e.g. $x_{1,t}$ and $x_{2,t}$, which are assumed to be independent and corresponding components $z_{1,t}$ and $z_{2,t}$, so

$$y_t = z_{1,t} + z_{2,t} + n_t.$$

2.3.1 Transfer function models

In a similar manner to that in which the structure of a univariate series may be represented by a finite-parameter ARIMA model, the structure of an input component may be represented by a **transfer function** (TF) model with delay time b, p autoregressive-like parameters $\delta_1, \delta_2, \ldots, \delta_p$ and $q+1$ moving-average-like parameters $\omega_0, \omega_1, \ldots, \omega_q$:

$$z_t = \delta_1 z_{t-1} + \delta_2 z_{t-2} + \cdots + \delta_p z_{t-p} + \omega_0 x_{t-b} - \omega_1 x_{t-b-1} - \cdots - \omega_q x_{t-b-q}.$$

If $p > 0$ this represents an IRF which is infinite in extent and decays with geometric and/or sinusoidal behaviour. The parameters $\delta_1, \delta_2, \ldots, \delta_p$ are constrained to satisfy a stability condition identical to the stationarity condition of autoregressive models. There is no constraint on $\omega_0, \omega_1, \ldots, \omega_q$.

2.3.2 Cross-correlations

An important tool for investigating how an input series x_t affects an output series y_t is the sample **cross-correlation function** (CCF) $r_{xy}(k)$, for $k = 0, 1, 2, \ldots$ between the series. If x_t and y_t are (jointly) stationary time series this is an estimator of the theoretical quantity

$$\rho_{xy}(k) = \operatorname{corr}(x_t, y_{t+k}).$$

The sequence $r_{yx}(k)$, for $k = 0, 1, 2, \ldots$ is distinct from $r_{xy}(k)$, though it is possible to interpret

$$r_{yx}(k) = r_{xy}(-k).$$

When the series y_t and x_t are believed to be related by a transfer function model, the CCF is determined by the IRF $v_0, v_1, v_2, \ldots$ and the autocorrelation function of the input x_t.

In the **particular** case when x_t is an uncorrelated series or **white noise** (and is uncorrelated with any other inputs)

$$\rho_{xy}(k) \propto v_k$$

and the sample CCF can provide an estimate of v_k:

$$\tilde{v}_k = (s_y / s_x) r_{xy}(k)$$

where s_y and s_x are the sample standard deviations of y_t and x_t, respectively.

In theory the IRF coefficients $v_b, \ldots, v_{b+p+q}$ determine the parameters in the TF model, and using $\tilde{v}_k$ to estimate $\tilde{v}_k$ it is possible to solve for **preliminary** estimates of $\delta_1, \delta_2, \ldots, \delta_p$, $\omega_0, \omega_1, \ldots, \omega_q$.

2.3.3 Prewhitening or filtering by an ARIMA model

In general an input series x_t is not white noise, but may be represented by an ARIMA model with innovations or residuals a_t which are white noise. If precisely the same operations by which a_t is generated from x_t are applied to the output y_t to produce a series b_t, then the transfer function relationship between y_t and x_t is preserved between b_t and a_t. It is then possible to estimate

$$\tilde{v}_k = (s_b / s_a) r_{ab}(k).$$

The procedure of generating a_t from x_t (and b_t from y_t) is called prewhitening or filtering by an ARIMA model. Although a_t is necesarily white noise, this is not generally true of b_t.

2.3.4 Multi-input model estimation

The term multi-input model is used for the situation when one output series y_t is related to one or more input series $x_{j,t}$, as described in Section 2.3. If for a given input the relationship is a simple linear regression, it is called a simple input; otherwise it is a transfer function input. The error or noise term follows an ARIMA model.

Given that the orders of all the transfer function models and the ARIMA model of a multi-input model have been specified, the various parameters in those models may be (simultaneously) estimated.

The procedure used is closely related to the least-squares principle applied to the innovations in the ARIMA noise model.

The innovations are derived for any proposed set of parameter values by calculating the response of each input to the transfer functions and then evaluating the noise n_t as the difference between this response (combined for all the inputs) and the output. The innovations are derived from the noise using the ARIMA model in the same manner as for a univariate series, and as described in Section 2.1.5.

In estimating the parameters, consideration has to be given to the lagged terms in the various model equations which are associated with times prior to the observation period, and are therefore unknown. The subroutine descriptions provide the necessary detail as to how this problem is treated.

Also, as described in Section 2.1.6 the sum of squares criterion

$$S = \sum a_t^2$$

is related to the quadratic form in the exact log-likelihood of the parameters:

$$-\tfrac{1}{2}\log|V| - \tfrac{1}{2}w^T V^{-1} w.$$

Here w is the vector of appropriately differenced noise terms, and

$$w^T V^{-1} w = S/\sigma_a^2,$$

where σ_a^2 is the innovation variance parameter.

The least-squares criterion is therefore identical to minimization of the quadratic form, but is not identical to exact likelihood. Because V may be expressed as $M\sigma_a^2$, where M is a function of the ARIMA model parameters, substitution of σ_a^2 by its maximum likelihood estimator yields a concentrated (or profile) likelihood which is a function of

$$|M|^{1/N} S.$$

N is the length of the differenced noise series w, and $|M| = \det M$.

Use of the above quantity, called the deviance, D, as an objective function is preferable to the use of S alone, on the grounds that it is equivalent to exact likelihood, and yields estimates with better properties. However, there is an appreciable computational penalty in calculating D, and in large samples it differs very little from S, except in the important case of seasonal ARIMA models where the number of whole seasons within the data length must also be large.

The user is given the option of taking the objective function to be either S or D, or a third possibility, the marginal likelihood. This is similar to exact likelihood but can counteract bias in the ARIMA model due to the fitting of a large number of simple inputs.

Approximate standard errors of the parameter estimates and the correlations between them are available after estimation.

The model residuals $\hat{a}_t$ are the innovations resulting from the estimation, and they are usually examined for the presence of either autocorrelation or cross-correlation with the inputs. Absence of such correlation provides some confirmation of the adequacy of the model.

2.3.5 Multi-input model forecasting

A multi-input model may be used to forecast the output series provided future values (possibly forecasts) of the input series are supplied.

Construction of the forecasts requires knowledge only of the model orders and parameters, together with a limited set of the most recent variables which appear in the model equations. This is called the state set. It is conveniently constituted after model estimation. Moreover, if new observations $y_{n+1}, y_{n+2}, \ldots$ of the output series and $x_{n+1}, x_{n+2}, \ldots$ of (all) the independent input series become available, then the model equations can easily be used to update the state set before constructing forecasts from the end of the new observations. The new innovations $a_{n+1}, a_{n+2}, \ldots$ generated in this updating may be used to monitor the continuing adequacy of the model.

2.3.6 Transfer function model filtering

In many time series applications it is desired to calculate the response (or output) of a transfer function model for a given input series.

Smoothing, detrending, and seasonal adjustment are typical applications. The user must specify the orders and parameters of a transfer function model for the purpose being considered. This may then be applied to the input series.

Again, problems may arise due to ignorance of the input series values prior to the observation period. The transient errors which can arise from this cause may be substantially reduced by using 'backforecasts' of these unknown observations.

2.4 Multivariate Time Series

Multi-input modelling represents one output time series in terms of one or more input series. Although there are circumstances in which it may be more appropriate to analyse a set of time series by modelling each one in turn as the output series with the remainder as inputs, there is a more symmetric approach in such a context. These models are known as vector autoregressive moving-average (VARMA) models.

2.4.1 Differencing and transforming a multivariate time series

As in the case of a univariate time series, it may be useful to simplify the series by differencing operations which may be used to remove linear or seasonal trend, thus ensuring that the resulting series to be used in the model estimation is stationary. It may also be neccessary to apply transformations to the individual components of the multivariate series in order to stabilize the variance. Commonly used transformations are the log and square root transformations.

2.4.2 Model identification for a multivariate time series

Multivariate analogues of the autocorrelation and partial autocorrelation functions are available for analysing a set of k time series, $x_{i,1}, x_{i,2}, \ldots, x_{i,n}$, for $i = 1, 2, \ldots, k$, thereby making it possible to obtain some understanding of a suitable VARMA model for the observed series.

It is assumed that the time series have been differenced if necessary, and that they are jointly stationary. The lagged correlations between all possible pairs of series, i.e.,

$$\rho_{ijl} = \operatorname{corr}(x_{i,t}, x_{j,t+l})$$

are then taken to provide an adequate description of the statistical relationships between the series. These quantities are estimated by sample auto- and cross-correlations r_{ijl}. For each l these may be viewed as elements of a (lagged) autocorrelation matrix.

Thus consider the **vector process** x_t (with elements x_{it}) and lagged autocovariance matrices Γ_l with elements of $\sigma_i \sigma_j \rho_{ijl}$ where $\sigma_i^2 = \operatorname{var}(x_{i,t})$. Correspondingly, Γ_l is estimated by the matrix C_l with elements $s_i s_j r_{ijl}$ where s_i^2 is the sample variance of x_{it}.

For a series with short-term cross-correlation only, i.e., r_{ijl} is not significant beyond some low lag q, then the pure vector MA(q) model, with no autoregressive parameters, i.e., $p = 0$, is appropriate.

The correlation matrices provide a description of the joint statistical properties of the series. It is also possible to calculate matrix quantities which are closely analogous to the partial autocorrelations of univariate series (see Section 2.1.3). Wei [6] discusses both the partial autoregression matrices proposed by Tiao and Box [5] and partial lag correlation matrices.

In the univariate case the partial autocorrelation function (pacf) between x_t and x_{t+l} is the correlation coefficient between the two after removing the linear dependence on each of the intervening variables $x_{t+1}, x_{t+2}, \ldots, x_{t+l-1}$. This partial autocorrelation may also be obtained as the last regression coefficient associated with x_t when regressing x_{t+l} on its l lagged variables $x_{t+l-1}, x_{t+l-2}, \ldots, x_t$. Tiao and Box [5] extended this method to the multivariate case to define the partial autoregression matrix. Heyse and Wei [4] also extended the univariate definition of the pacf to derive the correlation matrix between the vectors x_t and x_{t+l} after removing the linear dependence on each of the intervening vectors $x_{t+1}, x_{t+2}, \ldots, x_{t+l-1}$, the partial lag correlation matrix.

Note that the partial lag correlation matrix is a correlation coefficient matrix since each of its elements is a properly normalised correlation coefficient. This is not true of the partial autoregression matrices (except in the univariate case for which the two types of matrix are the same). The partial lag correlation matrix at lag 1 also reduces to the regular correlation matrix at lag 1; this is not true of the partial autoregression matrices (again except in the univariate case).

Both the above share the same cut-off property for autoregressive processes; that is for an autoregressive process of order p, the terms of the matrix at lags $p+1$ and greater are zero. Thus if the sample partial cross-correlations are significant only up to some low lag p then a pure vector AR(p) model is appropriate with $q = 0$. Otherwise moving-average terms will need to be introduced as well as autoregressive terms.

Under the hypothesis that x_t is an autoregressive process of order $l-1$, n times the sum of the squared elements of the partial lag correlation matrix at lag l is asymptotically distributed as a χ^2 variable with k^2 degrees of freedom where k is the dimension of the multivariate time series. This provides a diagnostic aid for determining the order of an autoregressive model.

The partial autoregression matrices may be found by solving a multivariate version of the Yule–Walker equations to find the autoregression matrices, using the final regression matrix coefficient as the partial autoregression matrix at that particular lag.

The basis of these calculations is a multivariate autoregressive model:

$$x_t = \phi_{l,1} x_{t-1} + \cdots + \phi_{l,l} x_{t-l} + e_{l,t}$$

where $\phi_{l,1}, \phi_{l,2}, \ldots, \phi_{l,l}$ are **matrix coefficients**, and $e_{l,t}$ is the vector of errors in the prediction. These coefficients may be rapidly computed using a recursive technique which requires, and simultaneously furnishes, a backward prediction equation:

$$x_{t-l-1} = \psi_{l,1} x_{t-l} + \psi_{l,2} x_{t-l+1} + \cdots + \psi_{l,l} x_{t-1} + f_{l,t}$$

(in the univariate case $\psi_{l,i} = \phi_{l,i}$).

The forward prediction equation coefficients, $\phi_{l,i}$, are of direct interest, together with the covariance matrix D_l of the prediction errors $e_{l,t}$. The calculation of these quantities for a particular maximum equation lag $l = L$ involves calculation of the same quantities for increasing values of $l = 1, 2, \ldots, L$.

The quantities $v_l = \det D_l / \det \Gamma_0$ may be viewed as generalized variance ratios, and provide a measure of the efficiency of prediction (the smaller the better). The reduction from v_{l-1} to v_l which occurs on extending the order of the predictor to l may be represented as

$$v_l = v_{l-1}(1 - \rho_l^2)$$

where ρ_l^2 is a multiple squared partial autocorrelation coefficient associated with k^2 degrees of freedom.

Sample estimates of all the above quantities may be derived by using the series covariance matrices C_l, for $l = 1, 2, \ldots, L$, in place of Γ_l. The best lag for prediction purposes may be chosen as that which yields the minimum final prediction error (FPE) criterion:

$$\mathrm{FPE}(l) = v_l \times \frac{(1 + lk^2/n)}{(1 - lk^2/n)}.$$

An alternative method of estimating the sample partial autoregression matrices is by using multivariate least-squares to fit a series of multivariate autoregressive models of increasing order.

2.4.3 VARMA model estimation

The cross-correlation structure of a stationary multivariate time series may often be represented by a model with a small number of parameters belonging to the vector autoregressive moving-average (VARMA) class. If the stationary series w_t has been derived by transforming and/or differencing the original series x_t, then w_t is said to follow the VARMA model:

$$w_t = \phi_1 w_{t-1} + \cdots + \phi_p w_{t-p} + \epsilon_t - \theta_1 \epsilon_{t-1} - \cdots - \theta_q \epsilon_{t-q},$$

where ϵ_t is a vector of uncorrelated residual series (white noise) with zero mean and constant covariance matrix Σ, $\phi_1, \phi_2, \ldots, \phi_p$ are the p autoregressive (AR) **parameter matrices** and $\theta_1, \theta_2, \ldots, \theta_q$ are the q moving-average (MA) **parameter matrices**. If w_t has a non-zero mean μ, then this can be allowed for by replacing $w_t, w_{t-1}, \ldots$ by $w_t - \mu, w_{t-1} - \mu, \ldots$ in the model.

A series generated by this model will only be stationary provided restrictions are placed on $\phi_1, \phi_2, \ldots, \phi_p$ to avoid unstable growth of w_t. These are **stationarity** constraints. The series ϵ_t may also be usefully interpreted as the linear **innovations** in w_t, i.e., the error if w_t were to be predicted using the information in all past values $w_{t-1}, w_{t-2}, \ldots$, provided also that $\theta_1, \theta_2, \ldots, \theta_q$ satisfy what are known as **invertibility** constraints. This allows the series ϵ_t to be generated by rewriting the model equation as

$$\epsilon_t = w_t - \phi_1 w_{t-1} - \cdots - \phi_p w_{t-p} + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q}.$$

The method of maximum likelihood may be used to estimate the parameters of a specified VARMA model from the observed multivariate time series together with their standard errors and correlations.

The residuals from the model may be examined for the presence of autocorrelations as a check on the adequacy of the fitted model.

2.4.4 VARMA model forecasting

Forecasts of the series may be constructed using a multivariate version of the univariate method. Efficient methods are available for updating the forecasts each time new observations become available.

2.5 Cross-spectral Analysis

The relationship between two time series may be investigated in terms of their sinusoidal components at different frequencies. At frequency ω a component of y_t of the form

$$R_y(\omega) \cos \omega t - \phi_y(\omega)$$

has its amplitude $R_y(\omega)$ and phase lag $\phi_y(\omega)$ estimated by

$$R_y(\omega) e^{i\phi_y(\omega)} = \frac{1}{n} \sum_{t=1}^{n} y_t e^{i\omega t}$$

and similarly for x_t. In the univariate analysis only the amplitude was important – in the cross analysis the phase is important.

2.5.1 The sample cross-spectrum

This is defined by

$$f_{xy}^*(\omega) = \frac{1}{2\pi n} \left(\sum_{t=1}^{n} y_t e^{i\omega t} \right) \left(\sum_{t=1}^{n} x_t e^{-i\omega t} \right).$$

It may be demonstrated that this is equivalently defined in terms of the sample CCF, $r_{xy}(k)$, of the series as

$$f_{xy}^*(\omega) = \frac{1}{2\pi} \sum_{-(n-1)}^{(n-1)} c_{xy}(k) e^{i\omega k}$$

where $c_{xy}(k) = s_x s_y r_{xy}(k)$ is the cross-covariance function.

2.5.2 The amplitude and phase spectrum

The cross-spectrum is specified by its real part or cospectrum $cf^*(\omega)$ and imaginary part or quadrature spectrum $qf^*(\omega)$, but for the purpose of interpretation the cross-amplitude spectrum and phase spectrum are useful:

$$A^*(\omega) = |f^*_{xy}(\omega)|, \; \phi^*(\omega) = \arg(f^*_{xy}(\omega)).$$

If the series x_t and y_t contain deterministic sinusoidal components of amplitudes R_y, R_x and phases ϕ_y, ϕ_x at frequency ω, then $A^*(\omega)$ will have a peak of approximate width π/n and height $(n/2\pi)R_yR_x$ at that frequency, with corresponding phase $\phi^*(\omega) = \phi_y - \phi_x$. This supplies no information that cannot be obtained from the two series separately. The statistical relationship between the series is better revealed when the series are purely stochastic and jointly stationary, in which case the expected value of $f^*_{xy}(\omega)$ converges with increasing sample size to the theoretical cross-spectrum

$$f_{xy}(\omega) = \frac{1}{2\pi}\sum_{-\infty}^{\infty}\gamma_{xy}(k)e^{i\omega k}$$

where $\gamma_{xy}(k) = \operatorname{cov}(x_t, y_{t+k})$. The sample spectrum, as in the univariate case, does not, however, converge to the theoretical spectrum without some form of smoothing which either implicitly (using a lag window) or explicitly (using a frequency window) averages the sample spectrum $f^*_{xy(\omega)}$ over wider bands of frequency to obtain a smoothed estimate $\hat{f}_{xy}(\omega)$.

2.5.3 The coherency spectrum

If there is no statistical relationship between the series at a given frequency, then $f_{xy}(\omega) = 0$, and the smoothed estimate $\hat{f}_{xy}(\omega)$, will be close to 0. This is assessed by the squared coherency between the series:

$$\hat{W}(\omega) = \frac{|\hat{f}_{xy}(\omega)|^2}{\hat{f}_{xx}(\omega)\hat{f}_{yy}(\omega)}$$

where $\hat{f}_{xx}(\omega)$ is the corresponding smoothed univariate spectrum estimate for x_t, and similarly for y_t. The coherency can be treated as a squared multiple correlation. It is similarly invariant in theory not only to simple scaling of x_t and y_t, but also to filtering of the two series, and provides a useful test statistic for the relationship between autocorrelated series. Note that without smoothing,

$$|f^*_{xy}(\omega)|^2 = f^*_{xx}(\omega)f^*_{yy}(\omega),$$

so the coherency is 1 at all frequencies, just as a correlation is 1 for a sample of size 1. Thus smoothing is essential for cross-spectrum analysis.

2.5.4 The gain and noise spectrum

If y_t is believed to be related to x_t by a linear lagged relationship as in Section 2.3, i.e.,

$$y_t = v_0x_t + v_1x_{t-1} + v_2x_{t-2} + \cdots + n_t,$$

then the theoretical cross-spectrum is

$$f_{xy}(\omega) = V(\omega)f_{xx}(\omega)$$

where

$$V(\omega) = G(\omega)e^{i\phi(\omega)} = \sum_{k=0}^{\infty}v_ke^{ik\omega}$$

is called the frequency response of the relationship.

Thus if x_t were a sinusoidal wave at frequency ω (and n_t were absent), y_t would be similar but multiplied in amplitude by $G(\omega)$ and shifted in phase by $\phi(\omega)$. Furthermore, the theoretical univariate spectrum

$$f_{yy}(\omega) = G(\omega)^2f_{xx}(\omega) + f_n(\omega)$$

where n_t, with spectrum $f_n(\omega)$, is assumed independent of the input x_t.

Cross-spectral analysis thus furnishes estimates of the gain

$$\hat{G}(\omega) = |\hat{f}_{xy}(\omega)|/\hat{f}_{xx}(\omega)$$

and the phase

$$\hat{\phi}(\omega) = \arg\left(\hat{f}_{xy}(\omega)\right)$$

From these representations of the estimated frequency response $\hat{V}(\omega)$, parametric TF models may be recognised and selected. The noise spectrum may also be estimated as

$$\hat{f}_{y|x}(\omega) = \hat{f}_{yy}(\omega)\left(1 - \hat{W}(\omega)\right)$$

– a formula which reflects the fact that in essence a regression is being performed of the sinusoidal components of y_t on those of x_t over each frequency band.

Interpretation of the frequency response may be aided by extracting from $\hat{V}(\omega)$ estimates of the IRF $\hat{v}_k$. It is assumed that there is no anticipatory response between y_t and x_t, i.e., no coefficients v_k with $k = -1, -2$ are needed (their presence might indicate feedback between the series).

2.5.5 Cross-spectrum smoothing by lag window

The estimate of the cross-spectrum is calculated from the sample cross-variances as

$$\hat{f}_{xy}(\omega) = \frac{1}{2\pi}\sum_{-M+S}^{M+S} w_{k-S}c_{xy}(k)e^{i\omega k}.$$

The lag window w_k extends up to a truncation lag M as in the univariate case, but its centre is shifted by an alignment lag S usually chosen to coincide with the peak cross-correlation. This is equivalent to an alignment of the series for peak cross-correlation at lag 0, and reduces bias in the phase estimation.

The selection of the truncation lag M, which fixes the bandwidth of the estimate, is based on the same criteria as for univariate series, and the same choice of M and window shape should be used as in univariate spectrum estimation to obtain valid estimates of the coherency, gain etc., and test statistics.

2.5.6 Direct smoothing of the cross-spectrum

The computations are exactly as for smoothing of the univariate spectrum except that allowance is made for an implicit alignment shift S between the series.

2.6 Kalman Filters

Kalman filtering provides a method for the analysis of multi-dimensional time series. The underlying model is:

$$\begin{aligned} X_{t+1} &= A_tX_t + B_tW_t \\ Y_t &= C_tX_t + V_t \end{aligned}$$

where X_t is the unobserved state vector, Y_t is the observed measurement vector, W_t is the state noise, V_t is the measurement noise, A_t is the state transition matrix, B_t is the noise coefficient matrix and C_t is the measurement coefficient matrix at time t. The state noise and the measurement noise are assumed to be uncorrelated with zero mean and covariance matrices:

$$E\{W_tW_t^T\} = Q_t \text{ and } E\{V_tV_t^T\} = R_t$$

If the system matrices A_t, B_t, C_t and the covariance matrices Q_t,R_t are known then Kalman filtering can be used to compute the minimum variance estimate of the stochastic variable X_t.

The estimate of X_t given observations Y_1 to Y_{t-1} is denoted by $\hat{X}_{t|t-1}$ with state covariance matrix $E\{\hat{X}_{t|t-1}\hat{X}_{t|t-1}^T\} = P_{t|t-1}$ while the estimate of X_t given observations Y_1 to Y_t is denoted by $\hat{X}_{t|t}$ with covariance matrix $E\{\hat{X}_{t|t}\hat{X}_{t|t}^T\} = P_{t|t}$.

The update of the estimate, $\hat{X}_{t+1|t}$, from time t to time $t+1$, is computed in two stages.

First, the update equations are:

$$\hat{X}_{t|t} = \hat{X}_{t|t-1} + K_t r_t, \quad P_{t|t} = (I - K_t C_t) P_{t|t-1}$$

where the residual $r_t = Y_t - C_t X_{t|t-1}$ has an associated covariance matrix $H_t = C_t P_{t|t-1} C_t^T + R_t$, and K_t is the Kalman gain matrix with

$$K_t = P_{t|t-1} C_t^T H_t^{-1}.$$

The second stage is the one-step-ahead prediction equations given by:

$$\hat{X}_{t+1|t} = A_t \hat{X}_{t|t}, \quad P_{t+1|t} = A_t P_{t|t} A_t^T + B_t Q_t B_t^T.$$

These two stages can be combined to give the one-step-ahead update-prediction equations:

$$\hat{X}_{t+1|t} = A_t \hat{X}_{t|t-1} + A_t K_t r_t.$$

The above equations thus provide a method for recursively calculating the estimates of the state vectors $\hat{X}_{t|t}$ and $\hat{X}_{t+1|t}$ and their covariance matrices $P_{t|t}$ and $P_{t+1|t}$ from their previous values. This recursive procedure can be viewed in a Bayesian framework as being the updating of the prior by the data Y_t.

The initial values $\hat{X}_{1|0}$ and $P_{1|0}$ are required to start the recursion. For stationary systems, $P_{1|0}$ can be computed from the following equation:

$$P_{1|0} = A_1 P_{1|0} A_1^T + B_1 Q_1 B_1^T,$$

which can be solved by iterating on the equation. For $\hat{X}_{1|0}$ the value $E\{X\}$ can be used if it is available.

2.6.1 Computational methods

To improve the stability of the computations the square root algorithm is used One recursion of the square root covariance filter algorithm which can be summarized as follows:

$$\begin{pmatrix} R_t^{1/2} & C_t S_t & 0 \\ 0 & A_t S_t & B_t Q_t^{1/2} \end{pmatrix} U = \begin{pmatrix} H_t^{1/2} & 0 & 0 \\ G_t & S_{t+1} & 0 \end{pmatrix}$$

where U is an orthogonal transformation triangularizing the left-hand pre-array to produce the right-hand post-array, S_t is the lower triangular Cholesky factor of the state covariance matrix $P_{t+1|t}$, $Q_t^{1/2}$ and $R_t^{1/2}$ are the lower triangular Cholesky factor of the covariance matrices Q and R and $H^{1/2}$ is the lower triangular Cholesky factor of the covariance matrix of the residuals. The relationship between the Kalman gain matrix, K_t, and G_t is given by

$$A_t K_t = G_t \left(H_t^{1/2}\right)^{-1}.$$

To improve the efficiency of the computations when the matrices A_t, B_t and C_t do not vary with time the system can be transformed to give a simpler structure, the transformed state vector is $U^* X$ where U^* is the transformation that reduces the matrix pair (A, C) to lower observer Hessenberg form. That is, the matrix U^* is computed such that the compound matrix,

$$\begin{pmatrix} CU^{*T} \\ U^* A U^{*T} \end{pmatrix}$$

is a lower trapezoidal matrix. The transformations need only be computed once at the start of a series, and the covariance matrices Q_t and R_t can still be time-varying.

2.6.2 Model fitting and forecasting

If the state space model contains unknown parameters, θ, these can be estimated using maximum likelihood. Assuming that W_t and V_t are normal variates the log-likelihood for observations $Y_t, t = 1, 2, \ldots, n$ is given by

$$\text{constant} - \frac{1}{2} \sum_{t=1}^{n} \ln(\det(H_t)) - \frac{1}{2} \sum_{t=1}^{t} r_t^T H_t^{-1} r_t$$

Optimal estimates for the unknown model parameters θ can then be obtained by using a suitable optimizer routine to maximize the likelihood function.

Once the model has been fitted forecasting can be performed by using the one-step-ahead prediction equations. The one-step-ahead prediction equations can also be used to 'jump over' any missing values in the series.

2.6.3 Kalman filter and time series models

Many commonly used time series models can be written as state space models. A univariate ARMA(p,q) model can be cast into the following state space form

$$\begin{aligned} x_t &= Ax_{t-1} + B\epsilon_t \\ w_t &= Cx_t \end{aligned}$$

where $r = \max(p, q+1)$, the first element of the state vector x_t is w_t,

$$A = \begin{pmatrix} \phi_1 & 1 & & & & \\ \phi_2 & & 1 & & & \\ . & & & . & & \\ . & & & & . & \\ \phi_{r-1} & & & & & 1 \\ \phi_r & 0 & 0 & . & . & 0 \end{pmatrix}, \; B = \begin{pmatrix} 1 \\ -\theta_1 \\ -\theta_2 \\ . \\ . \\ . \\ -\theta_{r-1} \end{pmatrix} \text{ and } C^T = \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \\ . \\ . \\ 0 \end{pmatrix}.$$

The representation for a k-variate ARMA(p,q) series (VARMA) is very similar to that given above, except now the state vector is of length kr and the ϕ's and θ's are now $k \times k$ matrices and the 1's in A, B and C are now the identity matrix of order k. If $p < r$ or $q + 1 < r$ then the appropriate ϕ or θ matrices are set to zero, respectively.

Since the compound matrix

$$\begin{pmatrix} C \\ A \end{pmatrix}$$

is already in lower observer Hessenberg form (i.e., it is lower trapezoidal with zeros in the top right-hand triangle) the invariant Kalman filter algorithm can be used directly without the need to generate a transformation matrix U^*.

3 Recommendations on Choice and Use of Available Routines

Note. Refer to the Users' Note for your implementation to check that a routine is available.

3.1 Time Domain Techniques – ARMA type models

This section is divided into routines for univariate, input-output and multivariate time-series modelling. The term input-output refers to time-series modelling of a single-output series dependent on one or more input series; this is also referred to as transfer function or multi-input modelling. These areas are discussed in relation to the process of model identification, estimation, checking and forecasting.

3.1.1 Univariate Series

(a) Model identification

The routine G13AUF may be used in obtaining either a range–mean or standard deviation–mean plot for a series of observations, which may be useful in detecting the need for a variance-stabilising transformation. G13AUF computes the range or standard deviation and the mean for successive groups of observations and G01AGF may then be used to produce a scatter plot of range against mean or of standard deviation against mean.

The routine G13AAF may be used to difference a time series. The $N = n - d - s \times D$ values of the differenced time series which extends for $t = 1 + d + s \times D, \ldots, n$ are stored in the first N elements of the output array.

The routine G13ABF may be used for direct computation of the autocorrelations. It requires the time series as input, after optional differencing by G13AAF.

An alternative is to use G13CAF, which uses the FFT to carry out the convolution for computing the autocovariances. Circumstances in which this is recommended are

(i) if the main aim is to calculate the smoothed sample spectrum,

(ii) if the series length and maximum lag for the autocorrelations are both very large, in which case appreciable computing time may be saved.

For more precise recommendations, see Gentleman and Sande [3]. In this case the autocorrelations r_k need to be obtained from the autocovariances c_k by $r_k = c_k/c_0$.

The routine G13ACF computes the partial autocorrelation function and prediction error variance estimates from an input autocorrelation function. Note that G13DNF, which is designed for multivariate time series, may also be used to compute the partial autocorrelation function together with χ^2 statistics and their significance levels.

Finite lag predictor coefficients are also computed by the routine G13ACF. It may have to be used twice, firstly with a large value for the maximum lag L in order to locate the optimum FPE lag, then again with L reset to this lag.

The routine G13DXF may be used to check that the autoregressive part of the model is stationary and that the moving-average part is invertible.

(b) Model estimation

The routine G13ADF is used to compute preliminary estimates of the ARIMA model parameters, the sample autocorrelations of the appropriately differenced series being input. The model orders are required.

The main routine for parameter estimation for ARIMA models is G13AEF, and an easy-to-use version is G13AFF. Both these routines use the least-squares criterion of estimation.

In some circumstances the use of G13BEF or G13DCF, which use maximum likelihood, is recommended.

The routines require the time series values to be input, together with the ARIMA orders. Any differencing implied by the model is carried out internally. They also require the maximum number of iterations to be specified, and return the state set for use in forecasting.

G13AEF should be preferred to G13AFF for:

(i) more information about the differenced series, its backforecasts and the intermediate series;

(ii) greater control over the output at successive iterations;

(iii) more detailed control over the search policy of the non-linear least-squares algorithm;

(iv) more information about the first and second derivatives of the objective function during and upon completion of the iterations.

G13BEF is primarily designed for estimating relationships between time series. It is, however, easily used in a univariate mode for ARIMA model estimation. The advantage is that it allows (optional) use of the exact likelihood estimation criterion, which is not available in G13AEF or G13AFF. This is particularly recommended for models which have seasonal parameters, because it reduces the tendency of parameter estimates to become stuck at points on the parameter space boundary. The model parameters estimated in this routine should be passed over to G13AJF for use in univariate forecasting.

The routine G13DCF is primarily designed for fitting vector ARMA models to multivariate time series but may also be used in a univariate mode. It allows the use of either the exact or conditional likelihood estimation criterion, and allows the user to fit non-multiplicative seasonal models which are not available in G13AEF, G13AFF or G13BEF.

(c) Model checking

G13ASF calculates the correlations in the residuals from a model fitted by either G13AEF or G13AFF. In addition the standard errors and correlations of the residual autocorrelations are computed along with a portmanteau test for model adequacy. G13ASF can be used after a univariate model has been fitted by G13BEF, but care must be taken in selecting the correct inputs to G13ASF. Note that if G13DCF has been used to fit a non-multiplicative seasonal model to a univariate series then G13DSF may be used to check the adequacy of the model.

(d) Forecasting using an ARIMA model

Given that the state set produced on estimation of the ARIMA model by either G13AEF or G13AFF has been retained, G13AHF can be used directly to construct forecasts for $x_{n+1}, x_{n+2}, \ldots$, together with probability limits. If some further observations $x_{n+1}, x_{n+2}, \ldots$ have come to hand since model estimation (and there is no desire to re-estimate the model using the extended series), then G13AGF can be used to update the state set using the new observations, prior to forecasting from the end of the extended series. The original series is not required.

The routine G13AJF is provided for forecasting when the ARIMA model is known but the state set is unknown. For example, the model may have been estimated by a procedure other than the use of G13AEF or G13AFF, such as G13BEF. G13AJF constructs the state set and optionally constructs forecasts with probability limits. It is equivalent to a call to G13AEF with zero iterations requested, followed by an optional call to G13AHF, but it is much more efficient.

3.1.2 Input-output/transfer function modelling

(a) Model identification

Normally use G13BCF for direct computation of cross-correlations, from which cross-covariances may be obtained by multiplying by $s_y s_x$, and impulse response estimates (after prewhitening) by multiplying by s_y/s_x, where s_y, s_x are the sample standard deviations of the series.

An alternative is to use G13CCF, which exploits the FFT to carry out the convolution for computing cross-covariances. The criteria for this are the same as given in Section 3.1.1 for calculation of autocorrelations. The impulse response function may also be computed by spectral methods without prewhitening using G13CGF.

G13BAF may be used to prewhiten or filter a series by an ARIMA model.

G13BBF may be used to filter a time series using a transfer function model.

(b) Estimation of input-output model parameters

The routine G13BDF is used to obtain preliminary estimates of transfer function model parameters. The model orders and an estimate of the impulse response function (see Section 3.2.1) are required.

The simultaneous estimation of the transfer function model parameters for the inputs, and ARIMA model parameters for the output, is carried out by G13BEF.

This routine requires values of the output and input series, and the orders of all the models. Any differencing implied by the model is carried out internally.

The routine also requires the maximum number of iterations to be specified, and returns the state set for use in forecasting.

(c) Input-output model checking

The routine G13ASF, primarily designed for univariate time series, can be used to test the residuals from an input-output model.

(d) Forecasting using an input-output model

Given that the state set produced on estimation of the model by G13BEF has been retained, the routine G13BHF can be used directly to construct forecasts of the output series. Future values of the input series (possibly forecasts previously obtained using G13AHF) are required.

If further observations of the output and input series have become available since model estimation (and there is no desire to re-estimate the model using the extended series) then G13BGF can be used to update the state set using the new observations prior to forecasting from the end of the extended series. The original series are not required.

The routine G13BJF is provided for forecasting when the multi-input model is known, but the state set is unknown. The set of output and input series must be supplied to the routine which then constructs the state set (for future use with G13BGF and/or G13BHF) and also optionally constructs forecasts of the output series in a similar manner to G13BHF.

In constructing probability limits for the forecasts, it is possible to allow for the fact that future input series values may themselves have been calculated as forecasts using ARIMA models. Use of this option requires that these ARIMA models be supplied to the routine.

(e) Filtering a time series using a transfer function model

The routine for this purpose is G13BBF.

3.1.3 Multivariate series

(a) Model identification

The routine G13DLF may be used to difference the series. The user must supply the differencing parameters for each component of the multivariate series. The order of differencing for each individual component does not have to be the same. The routine may also be used to apply a log or square root transformation to the components of the series.

The routine G13DMF may be used to calculate the sample cross-correlation or cross-covariance matrices. It requires a set of time series as input. The user may request either the cross-covariances or cross-correlations.

The routine G13DNF computes the partial lag correlation matrices from the sample cross-correlation matrices computed by G13DMF, and the routine G13DPF computes the least-squares estimates of the partial autoregression matrices and their standard errors. Both routines compute a series of χ^2 statistic that aid the determination of the order of a suitable autoregressive model. G13DBF may also be used in the identification of the order of an autoregressive model. The routine computes multiple squared partial autocorrelations and predicitve error variance ratios from the sample cross-correlations or cross-covariances computed by G13DMF.

The routine G13DXF may be used to check that the autoregressive part of the model is stationary and that the moving-average part is invertible.

(b) Estimation of VARMA model parameters

The routine for this purpose is G13DCF. This routine requires a set of time series to be input, together with values for p and q. The user must also specify the maximum number of likelihood evaluations to be permitted and which parameters (if any) are to be held at their initial (user-supplied) values. The fitting criterion is either **exact** maximum likelihood or **conditional** maximum likelihood.

G13DCF is primarily designed for estimating relationships between time series. It may, however, easily be used in univariate mode for non-seasonal and non-multiplicative seasonal ARIMA model estimation. The advantage is that it allows (optional) use of the exact maximum likelihood estimation criterion, which is not available in either G13AEF or G13AFF. The conditional likelihood option is recommended for those models in which the parameter estimates display a tendency to become stuck at points on the boundary of the parameter space. When one of the series is known to be influenced by all the others, but the others in turn are mutually independent and do not influence the output series, then G13BEF (the transfer function model fitting routine) may be more appropriate to use.

(c) VARMA model checking

G13DSF calculates the cross-correlation matrices of residuals for a model fitted by G13DCF. In addition the standard errors and correlations of the residual correlation matrices are computed along with a portmanteau test for model adequacy.

(d) Forecasting using a VARMA model

The routine G13DJF may be used to construct a chosen number of forecasts using the model estimated by G13DCF. The standard errors of the forecasts are also computed. A reference vector is set up by G13DJF so that should any further observations become available the existing forecasts can be efficiently updated using G13DKF. On a call to G13DKF the reference vector itself is also updated so that G13DKF may be called again each time new observations are available.

3.2 Frequency Domain Techniques

3.2.1 Univariate spectral estimation

Two routines are available, G13CAF carrying out smoothing using a lag window and G13CBF carrying out direct frequency domain smoothing. Both can take as input the original series, but G13CAF alone can use the sample autocovariances as alternative input. This has some computational advantage if a variety of spectral estimates needs to be examined for the same series using different amounts of smoothing.

However, the real choice in most cases will be which of the four shapes of lag window in G13CAF to use, or whether to use the trapezium frequency window of G13CBF. The references may be consulted for advice on this, but the two most recommended lag windows are the Tukey and Parzen. The Tukey window has a very small risk of supplying negative spectrum estimates; otherwise, for the same bandwidth, both give very similar results, though the Parzen window requires a higher truncation lag (more acf values).

The frequency window smoothing procedure of G13CBF with a trapezium shape parameter $p \simeq \frac{1}{2}$ generally gives similar results for the same bandwidth as lag window methods with a slight advantage of somewhat less distortion around sharp peaks, but suffering a rather less smooth appearance in fine detail.

3.2.2 Cross-spectrum estimation

Two routines are available for the main step in cross-spectral analysis. To compute the cospectrum and quadrature spectrum estimates using smoothing by a lag window, G13CCF should be used. It takes as input either the original series or cross-covariances which may be computed in a previous call of the same routine or possibly using results from G13BCF. As in the univariate case, this gives some advantage if estimates for the same series are to be computed with different amounts of smoothing.

The choice of window shape will be determined as the same as that which has already been used in univariate spectrum estimation for the series.

For direct frequency domain smoothing, G13CDF should be used, with similar consideration for the univariate estimation in choice of degree of smoothing.

The cross-amplitude and squared coherency spectrum estimates are calculated, together with upper and lower confidence bounds, using G13CEF. For input the cross-spectral estimates from either G13CCF or G13CDF and corresponding univariate spectra from either G13CAF or G13CBF are required.

The gain and phase spectrum estimates are calculated together with upper and lower confidence bounds using G13CFF. The required input is as for G13CEF above.

The noise spectrum estimates and impulse response function estimates are calculated together with multiplying factors for confidence limits on the former, and the standard error for the latter, using G13CGF. The required input is again the same as for G13CEF above.

3.3 Kalman filtering

Two routines are available for Kalman filtering: G13EAF for time varying systems and G13ABF for time invariant systems. The latter will optionally compute the required transformation to lower observer Hessenberg form. Both these routines return the Cholesky factor of the residual covariance matrix, H_t, with the Cholesky factor of the state covariance matrix S_{t+1} and the Kalman gain matrix, K_t pre-multiplied by A_t, in the case of G13EBF these may be for the transformed system. To compute the updated state vector and the residual vector the required matrix-vector multiplications can be performed by F06PAF (SGEMV/DGEMV).

3.4 Time Series Simulation

There are routines available in the G05 chapter for generating a realisation of a time series from a specified model: G05EGF and G05EWF for univariate time series and G05HDF for multivariate time series.

4 Routines Withdrawn or Scheduled for Withdrawal

Since Mark 13 the following routines have been withdrawn. Advice on replacing calls to these routines is given in the document 'Advice on Replacement Calls for Withdrawn/Superseded Routines'.

G13DAF

5 References

[1] Akaike H (1971) Autoregressive model fitting for control *Ann. Inst. Statist. Math.* **23** 163–180

[2] Box G E P and Jenkins G M (1976) *Time Series Analysis: Forecasting and Control* Holden-Day (Revised Edition)

[3] Gentleman W S and Sande G (1966) Fast Fourier transforms for fun and profit *Proc. Joint Computer Conference, AFIPS* **29** 563–578

[4] Heyse J F and Wei W W S (1985) The partial lag autocorrelation function *Technical Report No. 32* Department of Statistics, Temple University, Philadelphia

[5] Tiao G C and Box G E P (1981) Modelling multiple time series with applications *J. Am. Stat. Assoc.* **76** 802–816

[6] Wei W W S (1990) *Time Series Analysis: Univariate and Multivariate Methods* Addison-Wesley

G13AAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13AAF carries out non-seasonal and seasonal differencing on a time series. Information which allows the original series to be reconstituted from the differenced series is also produced. This information is required in time series forecasting.

2. Specification

```
SUBROUTINE G13AAF (X, NX, ND, NDS, NS, XD, NXD, IFAIL)
INTEGER          NX, ND, NDS, NS, NXD, IFAIL
real             X(NX), XD(NX)
```

3. Description

Let $\nabla^d \nabla_s^D x_i$ be the ith value of a time series x_i, for $i = 1,2,...,n$ after non-seasonal differencing of order d and seasonal differencing of order D (with period or seasonality s). In general,

$$\nabla^d \nabla_s^D x_i = \nabla^{d-1} \nabla_s^D x_{i+1} - \nabla^{d-1} \nabla_s^D x_i \qquad d > 0$$
$$\nabla^d \nabla_s^D x_i = \nabla^d \nabla_s^{D-1} x_{i+s} - \nabla^d \nabla_s^{D-1} x_i \qquad D > 0$$

Non-seasonal differencing up to the required order d is obtained using

$$\nabla^1 x_i = x_{i+1} - x_i \qquad \text{for } i = 1,2,...,(n-1)$$
$$\nabla^2 x_i = \nabla^1 x_{i+1} - \nabla^1 x_i \qquad \text{for } i = 1,2,...,(n-2)$$
$$\vdots$$
$$\nabla^d x_i = \nabla^{d-1} x_{i+1} - \nabla^{d-1} x_i \qquad \text{for } i = 1,2,...,(n-d)$$

Seasonal differencing up to the required order D is then obtained using

$$\nabla^d \nabla_s^1 x_i = \nabla^d x_{i+s} - \nabla^d x_i \qquad \text{for } i = 1,2,...,(n-d-s)$$
$$\nabla^d \nabla_s^2 x_i = \nabla^d \nabla_s^1 x_{i+s} - \nabla^d \nabla_s^1 x_i \qquad \text{for } i = 1,2,...,(n-d-2s)$$
$$\vdots$$
$$\nabla^d \nabla_s^D x_i = \nabla^d \nabla_s^{D+1} x_{i+s} - \nabla^d \nabla_s^{D+1} x_i \qquad \text{for } i = 1,2,...,(n-d-D\times s)$$

Mathematically, the sequence in which the differencing operations are performed does not affect the final resulting series of $m = n - d - D\times s$ values.

4. References

None.

5. Parameters

1: X(NX) – ***real*** array. *Input*

On entry: the undifferenced time series, x_i, for $i = 1,2,...,n$.

2: NX – INTEGER. *Input*

On entry: the number of values, n, in the undifferenced time series.

Constraint: NX > ND + (NDS×NS).

3: ND – INTEGER. *Input*

On entry: the order of non-seasonal differencing, d.

Constraint: ND $\geq$ 0.

4: NDS – INTEGER. *Input*

On entry: the order of seasonal differencing, D.

Constraint: NDS $\geq$ 0.

5: NS – INTEGER. *Input*

On entry: the seasonality, s.

If NDS > 0 then NS > 0.
If NDS = 0 then NS $\geq$ 0.

6: XD(NX) – ***real*** array. *Output*

On exit: the differenced values in elements 1 to NXD, and reconstitution data in the remainder of the array.

7: NXD – INTEGER. *Output*

On exit: the number of differenced values in the array XD.

8: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, ND < 0,
or NDS < 0,
or NS < 0,
or NS = 0 when NDS > 0.

IFAIL = 2

On entry, NX $\leq$ ND + (NDS$\times$NS).

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to (ND+NDS)$\times$NX.

9. Example

The following program reads in a set of data consisting of 20 observations from a time series. Non-seasonal differencing of order 2 and seasonal differencing of order 1 (with seasonality of 4) are applied to the input data, giving an output array holding 14 differenced values and 6 values which can be used to reconstitute the output array.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13AAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXMAX
      PARAMETER        (NXMAX=20)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, ND, NDS, NS, NX, NXD
*     .. Local Arrays ..
      real             X(NXMAX), XD(NXMAX)
*     .. External Subroutines ..
      EXTERNAL         G13AAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13AAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX, ND, NDS, NS
      IF (NX.GT.0 .AND. NX.LE.NXMAX) THEN
         READ (NIN,*) (X(I),I=1,NX)
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'Non-seasonal differencing of order ', ND,
     +     ' and seasonal differencing'
         WRITE (NOUT,99999) 'of order ', NDS, ' with seasonality ', NS,
     +     ' are applied'
         IFAIL = 0
*
         CALL G13AAF(X,NX,ND,NDS,NS,XD,NXD,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'The output array holds ', NX,
     +     ' values, of which the first ', NXD,
     +     ' are differenced values'
         WRITE (NOUT,*)
         WRITE (NOUT,99997) (XD(I),I=1,NX)
      END IF
      STOP
*
99999 FORMAT (1X,A,I1,A,I1,A)
99998 FORMAT (1X,A,I2,A,I2,A)
99997 FORMAT (1X,5F9.1)
      END
```

9.2. Program Data

```
G13AAF Example Program Data
 20  2  1  4
 120.0 108.0  98.0 118.0 135.0
 131.0 118.0 125.0 121.0 100.0
  82.0  82.0  89.0  88.0  86.0
  96.0 108.0 110.0  99.0 105.0
```

9.3. Program Results

```
G13AAF Example Program Results

Non-seasonal differencing of order 2 and seasonal differencing
of order 1 with seasonality 4 are applied

The output array holds 20 values, of which the first 14 are differenced values

    -11.0     -10.0      -8.0       4.0      12.0
     -2.0      18.0       9.0      -4.0      -6.0
     -5.0      -2.0     -12.0       5.0       2.0
    -10.0     -13.0      17.0       6.0     105.0
```

G13ABF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13ABF computes the sample autocorrelation function of a time series. It also computes the sample mean, the sample variance and a statistic which may be used to test the hypothesis that the true autocorrelation function is zero.

2. Specification

```
SUBROUTINE G13ABF (X, NX, NK, XM, XV, R, STAT, IFAIL)
INTEGER        NX, NK, IFAIL
real           X(NX), XM, XV, R(NK), STAT
```

3. Description

The data consist of n observations x_i, for $i = 1,2,...,n$ from a time series.

The quantities calculated are defined by

(a) The sample mean

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

(b) The sample variance (for $n \geq 2$)

$$s^2 = \frac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{(n-1)}$$

(c) The sample autocorrelation coefficients of lags $k = 1,2,...,K$, where K is a user-specified maximum lag, and $K < n$, $n > 1$.

The coefficient of lag k is defined as

$$r_k = \frac{\sum_{i=1}^{n-k}(x_i-\bar{x})(x_{i+k}-\bar{x})}{\sum_{i=1}^{n}(x_i-\bar{x})^2}$$

See page 496 *et seq.* of Box and Jenkins [1] for further details.

(d) A test statistic defined as

$$\text{STAT} = n\sum_{k=1}^{K} r_k^2,$$

which can be used to test the hypothesis that the true autocorrelation function is identically zero.

If n is large and K is much smaller than n, STAT has a χ_K^2 distribution under the hypothesis of a zero autocorrelation function. Values of STAT in the upper tail of the distribution provide evidence against the hypothesis; G01ECF can be used to compute the tail probability.

Section 8.2.2 of [1] provides further details of the use of STAT.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

5. Parameters

1: X(NX) – ***real*** array. *Input*

On entry: the time series, x_i, for $i = 1,2,...,n$.

2: NX – INTEGER. *Input*

On entry: the number of values, n, in the time series.

Constraint: NX > 1.

3: NK – INTEGER. *Input*

On entry: the number of lags, K, for which the autocorrelations are required. The lags range from 1 to K and do not include zero.

Constraint: 0 < NK < NX.

4: XM – ***real***. *Output*

On exit: the sample mean of the input time series.

5: XV – ***real***. *Output*

On exit: the sample variance of the input time series.

6: R(NK) – ***real*** array. *Output*

On exit: the sample autocorrelation coefficient relating to lag k, for $k = 1,2,...,K$.

7: STAT – ***real***. *Output*

On exit: the statistic used to test the hypothesis that the true autocorrelation function of the time series is identically zero.

8: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NX ≤ NK,
or NX ≤ 1,
or NK ≤ 0.

IFAIL = 2

On entry, all values of X are practically identical, giving zero variance. In this case R and STAT are undefined on exit.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to NX×NK.

If the input series for G13ABF was generated by differencing using G13AAF, ensure that only the differenced values are input to G13ABF, and not the reconstituting information.

9. Example

In the example below, a set of 50 values of sunspot counts is used as input. The first 10 autocorrelations are computed.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13ABF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXMAX, NKMAX
      PARAMETER        (NXMAX=50,NKMAX=10)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             STAT, XM, XV
      INTEGER          I, IFAIL, NK, NX
*     .. Local Arrays ..
      real             R(NKMAX), X(NXMAX)
*     .. External Subroutines ..
      EXTERNAL         G13ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13ABF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX, NK
      WRITE (NOUT,*)
      IF (NK.GT.0 .AND. NK.LE.NKMAX .AND. NX.GT.0 .AND. NX.LE.NXMAX)
     +    THEN
         READ (NIN,*) (X(I),I=1,NX)
         WRITE (NOUT,99999) 'The first ', NK,
     +     ' coefficients are required'
         IFAIL = 0
*
         CALL G13ABF(X,NX,NK,XM,XV,R,STAT,IFAIL)
*
         WRITE (NOUT,99998) 'The input array has sample mean ', XM
         WRITE (NOUT,99998) 'The input array has sample variance ', XV
         WRITE (NOUT,*) 'The sample autocorrelation coefficients are'
         WRITE (NOUT,*)
         WRITE (NOUT,*) '   Lag    Coeff      Lag    Coeff'
         WRITE (NOUT,99997) (I,R(I),I=1,10)
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'The value of STAT is ', STAT
      END IF
      STOP
*
99999 FORMAT (1X,A,I2,A)
99998 FORMAT (1X,A,F12.4)
99997 FORMAT (1X,I6,F10.4,I8,F10.4)
      END
```

9.2. Program Data

```
G13ABF Example Program Data
 50 10
   5.0  11.0  16.0  23.0  36.0
  58.0  29.0  20.0  10.0   8.0
   3.0   0.0   0.0   2.0  11.0
  27.0  47.0  63.0  60.0  39.0
  28.0  26.0  22.0  11.0  21.0
  40.0  78.0 122.0 103.0  73.0
  47.0  35.0  11.0   5.0  16.0
  34.0  70.0  81.0 111.0 101.0
  73.0  40.0  20.0  16.0   5.0
  11.0  22.0  40.0  60.0  80.9
```

9.3. Program Results

```
 G13ABF Example Program Results

 The first 10 coefficients are required
 The input array has sample mean       37.4180
 The input array has sample variance    1002.0301
 The sample autocorrelation coefficients are

    Lag    Coeff       Lag    Coeff
      1    0.8004        2    0.4355
      3    0.0328        4   -0.2835
      5   -0.4505        6   -0.4242
      7   -0.2419        8    0.0550
      9    0.3783       10    0.5857

 The value of STAT is      92.1231
```

G13ACF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13ACF calculates partial autocorrelation coefficients given a set of autocorrelation coefficients. It also calculates the predictor error variance ratios for increasing order of finite lag autoregressive predictor, and the autoregressive parameters associated with the predictor of maximum order.

2. Specification

```
SUBROUTINE G13ACF (R, NK, NL, P, V, AR, NVL, IFAIL)
INTEGER        NK, NL, NVL, IFAIL
real           R(NK), P(NL), V(NL), AR(NL)
```

3. Description

The data consist of values of autocorrelation coefficients $r_1,r_2,\ldots,r_K$, relating to lags $1,2,\ldots,K$. These will generally (but not necessarily) be sample values such as may be obtained from a time series x_t using G13ABF.

The partial autocorrelation coefficient at lag l may be identified with the parameter $p_{l,l}$ in the autoregression

$$x_t = c_l + p_{l,1}x_{t-1} + p_{l,2}x_{t-2} + \ldots + p_{l,l}x_{t-l} + e_{l,t}$$

where $e_{l,t}$ is the predictor error.

The first subscript l of $p_{l,l}$ and $e_{l,t}$ emphasises the fact that the parameters will in general alter as further terms are introduced into the equation (i.e. as l is increased).

The parameters are determined from the autocorrelation coefficients by the Yule-Walker equations

$$r_i = p_{l,1}r_{i-1} + p_{l,2}r_{i-2} + \ldots + p_{l,l}r_{i-l}, \qquad i = 1,2,\ldots,l$$

taking $r_j = r_{|j|}$ when $j < 0$, and $r_0 = 1$.

The predictor error variance ratio $v_l = \mathrm{var}(e_{l,t})/\mathrm{var}(x_t)$ is defined by
$v_l = 1 - p_{l,1}r_1 - p_{l,2}r_2 - \ldots - p_{l,l}r_l$.

The above sets of equations are solved by a recursive method (the Durbin-Levinson algorithm). The recursive cycle applied for $l = 1,2,\ldots,(L-1)$, where L is the number of partial autocorrelation coefficients required, is initialised by setting $p_{1,1} = r_1$ and $v_1 = 1 - r_1^2$.

Then

$$\begin{aligned}
p_{l+1,l+1} &= (r_{l+1} - p_{l,1}r_l - p_{l,2}r_{l-1} - \ldots - p_{l,l}r_1)/v_l \\
p_{l+1,j} &= p_{l,j} - p_{l+1,l+1}p_{l,l+1-j}, \quad j = 1,2,\ldots,l \\
v_{l+1} &= v_l(1-p_{l+1,l+1})(1+p_{l+1,l+1}).
\end{aligned}$$

If the condition $|p_{l,l}| \geq 1$ occurs, say when $l = l_0$, it indicates that the supplied autocorrelation coefficients do not form a positive-definite sequence (see Hannan [3]), and the recursion is not continued. The autoregressive parameters are overwritten at each recursive step, so that upon completion the only available values are p_{Lj}, for $j = 1,2,\ldots,L$ or $p_{l_0-1,j}$ if the recursion has been prematurely halted.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

[2] DURBIN, J.
The Fitting of Time Series Models.
Rev. Inst. Internat. Stat. 28, p. 233, 1960.

[3] HANNAN, E.J.
Time Series Analysis.
Methuen, p. 11, 1960.

5. Parameters

1: R(NK) – ***real*** array. *Input*

On entry: the autocorrelation coefficient relating to lag k, for $k = 1,2,...,K$.

2: NK – INTEGER. *Input*

On entry: the number of lags, K. The lags range from 1 to K and do not include zero.

Constraint: NK > 0.

3: NL – INTEGER. *Input*

On entry: the number of partial autocorrelation coefficients required, L.

Constraint: 0 < NL ≤ NK.

4: P(NL) – ***real*** array. *Output*

On exit: P(l) contains the partial autocorrelation coefficient at lag l, $p_{l,l}$, for $l = 1,2,...,$NVL.

5: V(NL) – ***real*** array. *Output*

On exit: V(l) contains the predictor error variance ratio v_l, for $l = 1,2,...,$NVL.

6: AR(NL) – ***real*** array. *Output*

On exit: the autoregressive parameters of maximum order, i.e. p_{Lj} if IFAIL = 0, or $p_{l_0-1,j}$ if IFAIL = 3, for $j = 1,2,...,$NVL.

7: NVL – INTEGER. *Output*

On exit: the number of valid values in each of P, V and AR. Thus in the case of premature termination at iteration l_0 (see Section 3), NVL is returned as l_0-1.

8: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NK ≤ 0,
or NL ≤ 0,
or NK < NL.

IFAIL = 2

On entry, the autocorrelation coefficient of lag 1 has an absolute value greater than or equal to 1.0; no recursions could be performed.

IFAIL = 3

Recursion has been prematurely terminated; the supplied autocorrelation coefficients do not form a positive-definite sequence (see Section 3). Parameter NVL returns the number of valid values computed.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is proportional to $(\text{NVL})^2$.

9. Example

In the example below the input series is the set of 10 sample autocorrelation coefficients derived from the original series of sunspot numbers by G13ABF example program. The results show 5 values of each of the three output arrays – partial autocorrelation coefficients, predictor error variance ratios and autoregressive parameters. All of these were valid.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13ACF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NLMAX, NKMAX
      PARAMETER        (NLMAX=5,NKMAX=10)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, NK, NL, NVL
*     .. Local Arrays ..
      real             AR(NLMAX), P(NLMAX), R(NKMAX), V(NLMAX)
*     .. External Subroutines ..
      EXTERNAL         G13ACF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13ACF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NK, NL
      WRITE (NOUT,*)
      IF (NL.GT.0 .AND. NL.LE.NLMAX .AND. NK.GT.0 .AND. NK.LE.NKMAX)
     +    THEN
         READ (NIN,*) (R(I),I=1,NK)
         IFAIL = 1
*
         CALL G13ACF(R,NK,NL,P,V,AR,NVL,IFAIL)
*
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99999) 'G13ACF fails. IFAIL = ', IFAIL
            WRITE (NOUT,*)
         END IF
         IF (IFAIL.EQ.3) THEN
            WRITE (NOUT,99998) '   Only', NVL,
     +        'valid sets were generated'
            WRITE (NOUT,*)
         END IF
```

```
            IF (IFAIL.EQ.0 .OR. IFAIL.EQ.3) THEN
               WRITE (NOUT,*)
     +           'Lag  Partial    Predictor error  Autoregressive'
               WRITE (NOUT,*) '     autocorrn  variance ratio   parameter'
               WRITE (NOUT,*)
               WRITE (NOUT,99997) (I,P(I),V(I),AR(I),I=1,NVL)
            END IF
         END IF
         STOP
*
99999 FORMAT (1X,A,I1)
99998 FORMAT (1X,A,I2,A)
99997 FORMAT (1X,I2,F9.3,F16.3,F14.3)
      END
```

9.2. Program Data

```
G13ACF Example Program Data
 10  5
    0.8004    0.4355    0.0328   -0.2835   -0.4505
   -0.4242   -0.2419   -0.0550    0.3783    0.5857
```

9.3. Program Results

```
 G13ACF Example Program Results

 Lag  Partial    Predictor error  Autoregressive
      autocorrn  variance ratio   parameter

  1     0.800           0.359          1.108
  2    -0.571           0.242         -0.290
  3    -0.239           0.228         -0.193
  4    -0.049           0.228         -0.014
  5    -0.032           0.228         -0.032
```

G13ADF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13ADF calculates preliminary estimates of the parameters of an autoregressive integrated moving average (ARIMA) model from the autocorrelation function of the appropriately differenced times series.

2. Specification

```
      SUBROUTINE G13ADF (MR, R, NK, XV, NPAR, WA, NWA, PAR, RV, ISF,
     1                   IFAIL)
      INTEGER            MR(7), NK, NPAR, NWA, ISF(4), IFAIL
      real               R(NK), XV, WA(NWA), PAR(NPAR), RV
```

3. Description

Preliminary estimates of the p non-seasonal autoregressive parameters $\phi_1,\phi_2,...,\phi_p$ and the q non-seasonal moving average parameters $\theta_1,\theta_2,...,\theta_q$ may be obtained from the sample autocorrelations relating to lags 1 to $p + q$, i.e. $r_1,...,r_{p+q}$, of the differenced $\nabla^d \nabla_s^D x_t$ where x_t is assumed to follow a (possibly) seasonal ARIMA model (see Section 3 of G13AEF for the specification of an ARIMA model).

Taking $r_0 = 1$ and $r_{-k} = r_k$, the ϕ_i, for $i = 1,2,...,p$ are the solutions to the equations

$$r_{q+i-1}\phi_1 + r_{q+i+2}\phi_2 + ... + r_{q+i-p}\phi_p = r_{q+i}, \qquad \text{for } i = 1,2,...,p.$$

The θ_j, for $j = 1,2,...,q$ are obtained from the solutions to the equations

$$c_j = \tau_0\tau_j + \tau_1\tau_{j+1} + ... + \tau_{q+j}\tau_q, \qquad \text{for } j = 0,1,...,q$$

(Cramer Wold-factorization) by setting

$$\theta_j = -\frac{\tau_j}{\tau_0}$$

where c_j are the 'covariances' modified in a 2-stage process by the autoregressive parameters.

Stage 1:

$$d_j = r_j - \phi_1 r_{j-1} - ... - \phi_p r_{j-p}, \qquad \text{for } j = 0,1,...,q;$$
$$d_j = 0, \qquad \text{for } j = q+1,q+2,...,p+q.$$

Stage 2:

$$c_j = d_j - \phi_1 d_{j+1} - \phi_2 d_{j+2} - ... - \phi_p d_{j+p}, \qquad \text{for } j = 0,1,...,q.$$

The P seasonal autoregressive parameters $\Phi_1,\Phi_2,...,\Phi_P$ and the Q seasonal moving average parameters $\Theta_1,\Theta_2,...,\Theta_Q$ are estimated in the same way as the non-seasonal parameters, but each r_j is replaced in the calculation by $r_{s\times j}$, where s is the seasonal period.

An estimate of the residual variance is obtained by successively reducing the sample variance, first for non-seasonal, and then for seasonal, parameter estimates. If moving average parameters are estimated, the variance is reduced by a multiplying factor of τ_0^2, but otherwise by c_0.

4. References

[1] BOX, G.E.P., and JENKINS, G.H.
Time Series Analysis: Forecasting and Control (Revised Edition)
Holden-Day, p. 498, 1976.

5. Parameters

1: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s) of the ARIMA model whose parameters are to be estimated. p, q, P and Q refer respectively to the number of autoregressive (ϕ), moving average (θ), seasonal autoregressive (Φ) and seasonal moving average (Θ) parameters. d, D and s refer respectively to the order of non-seasonal differencing, the order of seasonal differencing and the seasonal period.

Constraints: $p,d,q,P,D,Q,s \geq 0$,
$p + q + P + Q > 0$,
$s \neq 1$,
if $s = 0$, then $P + D + Q = 0$,
if $s > 1$, then $P + D + Q > 0$.

2: R(NK) – ***real*** array. *Input*

On entry: the autocorrelations (starting at lag 1), which must have been calculated after the time series has been appropriately differenced.

Constraint: $-1.0 \leq \mathrm{R}(i) \leq 1.0$, for $i = 1,2,...,\mathrm{NK}$.

3: NK – INTEGER. *Input*

On entry: the maximum lag of the autocorrelations in array R.

Constraint: $\mathrm{NK} \geq \max(p+q,s\times(P+Q))$.

4: XV – ***real.*** *Input*

On entry: the series sample variance, calculated after appropriate differencing has been applied to the series.

Constraint: $\mathrm{XV} > 0.0$.

5: NPAR – INTEGER. *Input*

On entry: the exact number of parameters specified in the model by array MR.

Constraint: $\mathrm{NPAR} = p + q + P + Q$.

6: WA(NWA) – ***real*** array. *Workspace*

7: NWA – INTEGER. *Input*

On entry: the amount of workspace available.

Constraint: if $\mathrm{MR} = (p,d,q,P,D,Q,s)$ and $p' = \max(p,P)$ and $q' = \max(q,Q)$ then $\mathrm{NWA} \geq \max(p'^2+p',4(q'+1))$.

8: PAR(NPAR) – ***real*** array. *Output*

On exit: the first NPAR elements of PAR contain the preliminary estimates of the ARIMA model parameters, in standard order.

9: RV – ***real.*** *Output*

On exit: an estimate of the residual variance of the preliminarily estimated model.

10: ISF(4) – INTEGER array. *Output*

On exit: the first 4 elements of ISF contain success/failure indicators, one for each of the 4 types of parameter (autoregressive, moving average, seasonal autoregressive, seasonal moving average).

The indicator has the interpretation:

0	no parameter of this type is in the model.
1	parameters of this type appear in the model and satisfactory preliminary estimates of this type were obtained.
–1	parameters of this type appear in the model but satisfactory preliminary estimates of this type were not obtainable. The estimates of this type of parameter were set to 0.0 in array PAR.

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, the orders vector MR is invalid. One of the constraints in Section 5 has been violated.

IFAIL = 2

On entry, NK $< \max(p+q,s\times(P+Q))$. There are not enough autocorrelations to enable the required model to be estimated.

IFAIL = 3

On entry, at least one element of R lies outside the range [–1.0,1.0].

IFAIL = 4

On entry, XV $\leq$ 0.0.

IFAIL = 5

On entry, NPAR $\neq p + q + P + Q$.

IFAIL = 6

On entry, the workspace array WA is too small. See Section 5 for the minimum size formula.

IFAIL = 7

Satisfactory parameter estimates could not be obtained for all parameter types in the model. Inspect array ISF for indicators of the parameter type(s) which could not be estimated.

7. Accuracy

The performance of the algorithm is conditioned by the roots of the autoregressive and moving average operators. If these are not close to unity in modulus, the errors, e, should satisfy $e < 100\varepsilon$ where ε is ***machine precision.***

8. Further Comments

The time taken by the routine is approximately proportional to $(p^3+q^2+P^3+Q^2)$.

9. Example

This example reads the sample autocorrelations to lag 40 and the sample variance of the lagged and doubly differenced series of airline passenger totals (Box and Jenkins example series G [1]). Preliminary estimates of the parameters of the (0,1,1,0,1,1,12) model are obtained by a call to G13ADF.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13ADF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NPMAX, NWA, NLMAX
      PARAMETER        (NPMAX=10,NWA=200,NLMAX=50)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             RV, YV
      INTEGER          I, IFAIL, NL, NPAR
*     .. Local Arrays ..
      real             PAR(NPMAX), R(NLMAX), WA(NWA)
      INTEGER          ISF(4), MR(7)
*     .. External Subroutines ..
      EXTERNAL         G13ADF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13ADF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NL
      READ (NIN,*) YV
      WRITE (NOUT,*)
      IF (NL.GT.0 .AND. NL.LE.NLMAX) THEN
         READ (NIN,*) (R(I),I=1,NL)
         READ (NIN,*) MR
         NPAR = MR(1) + MR(3) + MR(4) + MR(6)
         IF (NL.GT.0 .AND. NPAR.LE.NPMAX) THEN
            IFAIL = 1
*
            CALL G13ADF(MR,R,NL,YV,NPAR,WA,NWA,PAR,RV,ISF,IFAIL)
*
            IF (IFAIL.NE.0) THEN
               WRITE (NOUT,99999) 'G13ADF fails. IFAIL = ', IFAIL
               WRITE (NOUT,*)
            END IF
            IF (IFAIL.EQ.0 .OR. IFAIL.GE.7) THEN
               WRITE (NOUT,99998)
     +           'Parameter estimation success/failure indicator',
     +           (ISF(I),I=1,4)
               WRITE (NOUT,*)
               WRITE (NOUT,99997) 'ARIMA model parameter values ',
     +           (PAR(I),I=1,NPAR)
               WRITE (NOUT,*)
               WRITE (NOUT,99997) 'Residual variance', RV
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I1)
99998 FORMAT (1X,A,4I4)
99997 FORMAT (1X,A,5F10.5)
      END
```

9.2. Program Data

```
G13ADF Example Program Data
  40 200
  0.00213
 -0.32804   0.09850  -0.21854   0.05585   0.04679   0.04135
 -0.07989   0.00335   0.13973  -0.04022   0.07618  -0.40583
  0.18239  -0.05057   0.16094  -0.15900   0.09152  -0.03474
  0.05195  -0.14417   0.04264  -0.08170   0.23389  -0.02828
 -0.09001   0.03050  -0.02046   0.05522  -0.02048  -0.06651
 -0.02940   0.20204  -0.13953   0.10098  -0.20849   0.03338
  0.00829   0.07082  -0.04457  -0.01216
  0  1  1  0  1  1 12
```

9.3. Program Results

```
G13ADF Example Program Results

Parameter estimation success/failure indicator   0   1   0   1

ARIMA model parameter values    0.37390   0.51237

Residual variance   0.00148
```

G13AEF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13AEF fits a seasonal autoregressive integrated moving average (ARIMA) model to an observed time series, using a nonlinear least-squares procedure incorporating backforecasting. Parameter estimates are obtained, together with appropriate standard errors. The residual series is returned, and information for use in forecasting the time series is produced for use by the routines G13AGF and G13AHF.

The estimation procedure is iterative, starting with initial parameter values such as may be obtained using G13ADF. It continues until a specified convergence criterion is satisfied, or until a specified number of iterations has been carried out. The progress of the procedure can be monitored by means of a user-supplied routine.

2. Specification

```
SUBROUTINE G13AEF (MR, PAR, NPAR, C, KFC, X, NX, ICOUNT, EX, EXR,
1                  AL, IEX, S, G, IGH, SD, H, IH, ST, IST, NST, PIV,
2                  KPIV, NIT, ITC, ZSP, KZSP, ISF, WA, IWA, HC,
3                  IFAIL)
 INTEGER         MR(7), NPAR, KFC, NX, ICOUNT(6), IEX, IGH, IH,
1                IST, NST, KPIV, NIT, ITC, KZSP, ISF(4), IWA,
2                IFAIL
 real            PAR(NPAR), C, X(NX), EX(IEX), EXR(IEX), AL(IEX),
1                S, G(IGH), SD(IGH), H(IH,IGH), ST(IST), ZSP(4),
2                WA(IWA), HC(IH,IGH)
 EXTERNAL        PIV
```

3. Description

The time series $x_1,x_2,...,x_n$ supplied to the routine is assumed to follow a seasonal autoregressive integrated moving average (ARIMA) model defined as follows:

$$\nabla^d \nabla_s^D x_t - c = w_t$$

where $\nabla^d \nabla_s^D x_t$ is the result of applying non-seasonal differencing of order d and seasonal differencing of seasonality s and order D to the series x_t, as outlined in the description of G13AAF. The differenced series is then of length $N = n - d'$, where $d' = d + (D\times s)$ is the generalised order of differencing. The scalar c is the expected value of the differenced series, and the series $w_1,w_2,...,w_N$ follows a zero-mean stationary autoregressive moving average (ARMA) model defined by a pair of recurrence equations. These express w_t in terms of an uncorrelated series a_t, via an intermediate series e_t. The first equation describes the seasonal structure:

$$w_t = \Phi_1 \ w_{t-s} + \Phi_2 \ w_{t-2\times s} + ... + \Phi_P \ w_{t-P\times s} + e_t - \Theta_1 \ e_{t-s} - \Theta_2 \ e_{t-2\times s}$$
$$- ... - \Theta_Q \ e_{t-Q\times s}.$$

The second equation describes the non-seasonal structure. If the model is purely non-seasonal the first equation is redundant and e_t above is equated with w_t:

$$e_t = \phi_1 \ e_{t-1} + \phi_2 \ e_{t-2} + ... + \phi_p \ e_{t-p} + a_t - \theta_1 \ a_{t-1} - \theta_2 \ a_{t-2} - ... - \theta_q \ a_{t-q}.$$

Estimates of the model parameters defined by

$$\phi_1,\phi_2,...,\phi_p,\theta_1,\theta_2,...,\theta_q,$$
$$\Phi_1,\Phi_2,...,\Phi_P,\Theta_1,\Theta_2,...,\Theta_Q$$

and (optionally) c are obtained by minimizing a quadratic form in the vector $w = (w_1,w_2,...,w_N)'$.

This is $QF = w' \ V^{-1} \ w$, where V is the covariance matrix of w, and is a function of the model parameters. This matrix is not explicitly evaluated, since QF may be expressed as a 'sum of squares' function. When moving average parameters θ_i or Θ_i are present, so that the generalized moving average order $q' = q + s \times Q$ is positive, backforecasts $w_{1-q'}, w_{2-q'}, \ldots, w_0$ are introduced as nuisance parameters. The 'sum of squares' function may then be written as

$$S(pm) = \sum_{t=1-q'}^{N} a_t^2 - \sum_{t=1-q'-p'}^{-q'} b_t^2$$

where pm is a combined vector of parameters, consisting of the backforecasts followed by the ARMA model parameters.

The terms a_t correspond to the ARMA model residual series a_t, and $p' = p + s \times P$ is the generalised autoregressive order. The terms b_t are only present if autoregressive parameters are in the model, and serve to correct for transient errors introduced at the start of the autoregression.

The equations defining a_t and b_t are precisely:

$$e_t = w_t - \Phi_1 w_{t-s} - \Phi_2 w_{t-2\times s} - \ldots - \Phi_P w_{t-P\times s} + \Theta_1 e_{t-s} + \Theta_2 e_{t-2\times s} + \ldots + \Theta_Q e_{t-Q\times s}$$

for $t = 1-q', 2-q', \ldots, n$.

$$a_t = e_t - \phi_1 e_{t-1} - \phi_2 e_{t-2} - \ldots - \phi_p e_{t-p} + \theta_1 a_{t-1} + \theta_2 a_{t-2} + \ldots + \theta_q a_{t-q}$$

for $t = 1-q', 2-q', \ldots, n$.

$$f_t = w_t - \Phi_1 w_{t+s} - \Phi_2 w_{t+2\times s} - \ldots - \Phi_P w_{t+P\times s} + \Theta_1 f_{t-s} + \Theta_2 f_{t-2\times s} + \ldots + \Theta_Q f_{t-Q\times s}$$

for $t = (1-q'-s\times P), (2-q'-s\times P), \ldots, (-q'+P)$

$$b_t = f_t - \phi_1 f_{t+1} - \phi_2 f_{t+2} - \ldots - \phi_p f_{t+p} + \theta_1 b_{t-1} + \theta_2 b_{t-2} + \ldots + \theta_q b_{t-q}$$

for $t = (1-q'-p'), (2-q'-p'), \ldots, (-q')$.

For all four of these equations, the following conditions hold:

$w_i = 0$ if $i < 1 - q'$
$e_i = 0$ if $i < 1 - q'$
$a_i = 0$ if $i < 1 - q'$
$f_i = 0$ if $i < 1 - q' - s\times P$
$b_i = 0$ if $i < 1 - q' - p'$

Minimization of S with respect to pm uses an extension of the algorithm of Marquardt [2].

The first derivatives of S with respect to the parameters are calculated as

$$2\times\sum a_t \times a_{t,i} - 2\sum b_t \times b_{t,i} = 2\times G_i$$

where $a_{t,i}$ and $b_{t,i}$ are derivatives of a_t and b_t with respect to the ith parameter.

The second derivative of S is approximated by

$$2\times\sum a_{t,i} \times a_{t,j} - 2\times\sum b_{t,i} \times b_{t,j} = 2\times H_{ij}.$$

Successive parameter iterates are obtained by calculating a vector of corrections dpm by solving the equations

$$(H + \alpha \times D) \times dpm = -G$$

where G is a vector with elements G_i, H is a matrix with elements H_{ij}, α is a scalar used to control the search and D is the diagonal matrix of H.

The new parameter values are then $pm + dpm$.

The scalar α controls the step size, to which it is inversely related.

If a step results in new parameter values which give a reduced value of S, then α is reduced by a factor β. If a step results in new parameter values which give an increased value of S, or in ARMA model parameters which in any way contravene the stationarity and invertibility conditions, then the new parameters are rejected, α is increased by the factor β, and the revised equations are solved for a new parameter correction.

This action is repeated until either a reduced value of S is obtained, or α reaches the limit of 10^9, which is used to indicate a failure of the search procedure.

This failure may be due to a badly conditioned sum of squares function or to too strict a convergence criterion. Convergence is deemed to have occurred if the fractional reduction in the residual sum of squares in successive iterations is less than a value γ, while $\alpha < 1.0$.

The stationarity and invertibility conditions are tested to within a specified tolerance multiple δ of machine accuracy. Upon convergence, or completion of the specified maximum number of iterations without convergence, statistical properties of the estimates are derived. In the latter case the sequence of iterates should be checked to ensure that convergence is adequate for practical purposes, otherwise these properties are not reliable.

The estimated residual variance is

$$erv = S_{\min} \,/\, df$$

where $S_{\min}$ is the final value of S, and the residual number of degrees of freedom is given by

$$df = N - p - q - P - Q \;(-1 \text{ if } c \text{ is estimated}).$$

The covariance matrix of the vector of estimates pm is given by

$$erv \times H^{-1}$$

where H is evaluated at the final parameter values.

From this expression are derived the vector of standard deviations, and the correlation matrix for the whole parameter set. These are asymptotic approximations.

The differenced series w_t (now uncorrected for the constant), intermediate series e_t and residual series a_t are all available upon completion of the iterations over the range (extended by backforecasts)

$$t = 1-q',2-q',\ldots,N.$$

The values a_t can only properly be interpreted as residuals for $t \geq 1 + p' - q'$, as the earlier values are corrupted by transients if $p' > 0$.

In consequence of the manner in which differencing is implemented, the residual a_t is the one step ahead forecast error for $x_{t+d'}$.

For convenient application in forecasting, the following quantities constitute the 'state set', which contains the minimum amount of time series information needed to construct forecasts:

(i) the differenced series w_t, for $(N-s\times P) < t \leq N$

(ii) the d' values required to reconstitute the original series x_t from the differenced series w_t

(iii) the intermediate series e_t, for $(N-\max(p,Q\times s)) < t \leq N$,

(iv) the residual series a_t, for $(N-q) < t \leq N$.

This state set is available upon completion of the iterations. The routine may be used purely for the construction of this state set, given a previously estimated model and time series x_t, by requesting zero iterations. Backforecasts are estimated, but the model parameter values are unchanged. If later observations become available and it is desired to update the state set, G13AGF can be used.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

[2] MARQUARDT, D.W.
An Algorithm for Least-squares Estimation of Nonlinear Parameters,
J. Soc. Indust. Appl. Math., 11, 431, 1963.

5. Parameters

1: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s) of the ARIMA model whose parameters are to be estimated. p, q, P and Q refer respectively to the number of autoregressive (ϕ), moving average (θ), seasonal autoregressive (Φ) and seasonal moving average (Θ) parameters. d, D and s refer respectively to the order of non-seasonal differencing, the order of seasonal differencing and the seasonal period.

Constraints: $p, d, q, P, D, Q, s \geq 0$,
$p + q + P + Q > 0$,
$s \neq 1$,
if $s = 0$, then $P + D + Q = 0$,
if $s > 1$, then $P + D + Q > 0$.

2: PAR(NPAR) – ***real*** array. *Input/Output*

On entry: the initial estimates of the p values of the ϕ parameters, the q values of the θ parameters, the P values of the Φ parameters and the Q values of the Θ parameters, in that order.

On exit: the latest values of the estimates of these parameters.

3: NPAR – INTEGER. *Input*

On entry: the total number of ϕ, θ, Φ, and Θ parameters to be estimated.

Constraint: NPAR $= p + q + P + Q$.

4: C – ***real***. *Input/Output*

On entry: if KFC $= 0$, C must contain the expected value, c, of the differenced series; if KFC $= 1$, C must contain an initial estimate of c.

On exit: if KFC $= 0$, C is unchanged; if KFC $= 1$, C contains the latest estimate of c. Therefore, if C and KFC are both zero on entry, there is no constant correction.

5: KFC – INTEGER. *Input*

On entry: the value 0 if the constant is to remain fixed, and 1 if it is to be estimated.

6: X(NX) – ***real*** array. *Input*

On entry: the n values of the original undifferenced time series.

7: NX – INTEGER. *Input*

On entry: the length of the original undifferenced time series, n.

8: ICOUNT(6) – INTEGER array. *Output*

On exit:

ICOUNT(1), contains $q + (Q \times s)$, the number of backforecasts.

ICOUNT(2), contains $n - d - (D \times s)$, the number of differenced values.

ICOUNT(3), contains $d + (D \times s)$, the number of values of reconstitution information.

ICOUNT(4), contains $n + q + (Q \times s)$, the number of values held in each of the series EX, EXR and AL.

ICOUNT(5), contains $(n-d-(D \times s)-p-q-P-Q-\text{KFC})$, the number of degrees of freedom associated with S.

ICOUNT(6), contains ICOUNT(1)+NPAR+KFC, the number of parameters being estimated.

These values are always computed regardless of the exit value of IFAIL.

9: EX(IEX) – ***real*** array. *Output*

On exit: the extended differenced series which is made up of:

ICOUNT(1), backforecast values of the differenced series

ICOUNT(2), actual values of the differenced series

ICOUNT(3), values of reconstitution information

in that order.

The total number of these values held in EX is ICOUNT(4).

If the routine exits because of a faulty input parameter, the contents of EX will be indeterminate.

10: EXR(IEX) – ***real*** array. *Output*

On exit: the values of the model residuals which are made up of:

ICOUNT(1), residuals corresponding to the backforecasts in the differenced series.

ICOUNT(2), residuals corresponding to the actual values in the differenced series.

The remaining ICOUNT(3) values contain zeros.

If the routine exits with IFAIL holding a value other than 0 or 9, the contents of EXR will be indeterminate.

11: AL(IEX) – ***real*** array. *Output*

On exit: the intermediate series which is made up of:

ICOUNT(1), intermediate series values corresponding to the backforecasts in the differenced series.

ICOUNT(2), intermediate series values corresponding to the actual values in the differenced series.

The remaining ICOUNT(3) values contain zeros.

If the routine exits with IFAIL $\neq$ 0, the contents of AL will be indeterminate.

12: IEX – INTEGER. *Input*

On entry: the dimension of the arrays EX, EXR and AL as declared in the (sub)program from which G13AEF is called.

Constraint: IEX $\geq q + (Q \times s) + n$, which is equivalent to the exit value of ICOUNT(4).

13: S – ***real***. *Output*

On exit: the residual sum of squares after the latest series of parameter estimates has been incorporated into the model. If the routine exits with a faulty input parameter, S contains zero.

14: G(IGH) – ***real*** array. *Output*

On exit: the latest value of the derivatives of S with respect to each of the parameters being estimated (backforecasts, PAR parameters, and where relevant the constant – in that order). The contents of G will be indeterminate if the routine exits with a faulty input parameter.

15: IGH – INTEGER. *Input*

On entry: the dimension of each of the arrays G and SD, and the second dimension of each of the arrays H and HC.

Constraint: IGH $\geq q + (Q \times s)$ + NPAR + KFC which is equivalent to the exit value of ICOUNT(6).

16: SD(IGH) – ***real*** array. *Output*

On exit: the standard deviations corresponding to each of the parameters being estimated (backforecasts, PAR parameters, and where relevant the constant – in that order).

If the routine exits with IFAIL containing a value other than 0 or 9, or if the required number of iterations is zero, the contents of SD will be indeterminate.

17: H(IH,IGH) – ***real*** array. *Output*

On exit: (a) the latest values of an approximation to the second derivative of S with respect to each of the (q+Q×s+NPAR+KFC) parameters being estimated (backforecasts, PAR parameters, and where relevant the constant – in that order), and (b) the correlation coefficients relating to each pair of these parameters. These are held in a matrix defined by the first (q+Q×s+NPAR+KFC) rows and the first (q+Q×s+NPAR+KFC) columns of H. (Note that ICOUNT(6) contains the value of this expression.) The values of (a) are contained in the upper triangle, and the values of (b) in the strictly lower triangle.

These correlation coefficients are zero during intermediate print out using PIV, and indeterminate if IFAIL contains on exit a value other than 0 or 9.

All the contents of H are indeterminate if the required number of iterations is zero.

18: IH – INTEGER. *Input*

On entry: the first dimension of the arrays H and HC as declared in the (sub)program from which G13AEF is called.

Constraint: IH > q + (Q×s) + NPAR + KFC, which is equivalent to the exit value of ICOUNT(6).

19: ST(IST) – ***real*** array. *Output*

On exit: the NST values of the state set array. If the routine exits with IFAIL containing a value other than 0 or 9, the contents of ST will be indeterminate.

20: IST – INTEGER. *Input*

On entry: the dimension of the array ST as declared in the (sub)program from which G13AEF is called.

Constraint: IST ≥ (P×s) + d + (D×s) + q + max(p,Q×s).

21: NST – INTEGER. *Output*

On exit: the number of values in the state set array ST.

22: PIV – SUBROUTINE, supplied by the user. *External Procedure*

PIV is used to monitor the progress of the optimization.

Its specification is:

```
SUBROUTINE PIV(MR, PAR, NPAR, C, KFC, ICOUNT, S, G, H, IH, IGH, ITC,
               ZSP)
INTEGER    MR(7), NPAR, KFC, ICOUNT(6), IH, IGH, ITC
real       PAR(NPAR), C, S, G(IGH), H(IH,IGH), ZSP(4)
```

All the parameters are defined as for G13AEF itself. PIV is called on each iteration by G13AEF when the input value of KPIV is non-zero and is bypassed when it is 0.

The routine G13AFZ may be used as PIV. It prints the heading

```
G13AFZ MONITORING OUTPUT - ITERATION n
```

followed by the parameter values and the residual sum of squares. Output is directed to the advisory channel defined by X04ABF.

If KPIV = 0 a dummy routine PIV must be supplied. PIV must be declared as EXTERNAL in the (sub)program from which G13AEF is called. Parameters denoted as *Input* must **not** be changed by this procedure.

Note: in certain implementations where both single precision and double precision versions of the Library are available, the auxiliary routine G13AFZ is renamed as AFZG13; check the appropriate implementation document.

23: KPIV – INTEGER. *Input*

On entry: KPIV must be non-zero if the progress of the optimization is to be monitored using the user-supplied routine PIV. Otherwise KPIV must contain 0.

24: NIT – INTEGER. *Input*

On entry: the maximum number of iterations to be performed.

Constraint: NIT $\geq$ 0.

25: ITC – INTEGER. *Output*

On exit: the number of iterations performed.

26: ZSP(4) – ***real*** array. *Input/Output*

On entry: when KZSP = 1, the first 4 elements of ZSP must contain the four values used to guide the search procedure, as follows:

ZSP(1), contains α, the value used to constrain the magnitude of the search procedure steps.

Constraint: ZSP(1) > 0.0.

ZSP(2), contains β, the multiplier which regulates the value α.

Constraint: ZSP(2) > 1.0.

ZSP(3), contains δ, the value of the stationarity and invertibility test tolerance factor.

Constraint: ZSP(3) $\geq$ 1.0.

ZSP(4), contains γ, the value of the convergence criterion.

Constraint: 0.0 $\leq$ ZSP(4) < 1.0.

If KZSP $\neq$ 1 on entry, default values for ZSP are supplied by the routine.

These are 0.001, 10.0, 1000.0 and max($100\times\varepsilon$, 0.0000001), respectively, where ε is the value returned by X02AJF.

On exit: ZSP contains the values, default or otherwise, used by the routine.

27: KZSP – INTEGER. *Input*

On entry: the value 1 if the routine is to use the input values of ZSP, and any other value if the default values of ZSP are to be used.

28: ISF(4) – INTEGER array. *Output*

On exit: the first 4 elements of ISF contain success/failure indicators, one for each of the 4 types of parameter in the model (autoregressive, moving average, seasonal autoregressive, seasonal moving average), in that order.

Each indicator has the interpretation:

–2	On entry parameters of this type have initial estimates which do not satisfy the stationarity or invertibility test conditions.
–1	The search procedure has failed to converge because the latest set of parameter estimates of this type is invalid.
0	No parameter of this type is in the model.
1	Valid final estimates for parameters of this type have been obtained.

29: WA(IWA) – ***real*** array. *Workspace*

30: IWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13AEF is called.

Constraints: $IWA \geq (F_1 \times F_2) + (9 \times NPAR)$,

where $F_1 = NX + 1 + p + (P \times s) + q + (Q \times s)$

and $F_2 = 8$ if $KFC = 1$

$= 7$ if $KFC = 0, Q > 0$

$= 6$ if $KFC = 0, Q = 0, P > 0$

$= 5$ if $KFC = 0, Q = 0, P = 0, q > 0$

$= 4$ otherwise.

31: HC(IH,IGH) – ***real*** array. *Workspace*

32: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.** To suppress the output of an error message when soft failure occurs, set IFAIL to 1.

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

IFAIL = 1

On entry, $NPAR \neq p + q + P + Q$,
or the orders vector MR is invalid (check it against the constraints in Section 5),
or $KFC \neq 0$ or 1.

IFAIL = 2

On entry, $NX - d - D \times s \leq NPAR + KFC$ i.e. the number of terms in the differenced series is not greater than the number of parameters in the model. The model is over-parameterised.

IFAIL = 3

On entry, one or more of the user-supplied criteria for controlling the iterative process are invalid.

$NIT < 0$,
or if $KZSP = 1$, $ZSP(1) \leq 0.0$,
or $ZSP(2) \leq 1.0$,
or $ZSP(3) \leq 1.0$,
or $ZSP(4) < 0.0$,
or $ZSP(4) \geq 1.0$.

IFAIL = 4

On entry, the state set array ST is too small. The output value of NST contains the required value (see description of IST in Section 5 for the formula).

IFAIL = 5

On entry, the workspace array WA is too small. Check the value of IWA against the constraints in Section 5.

IFAIL = 6

On entry, IEX $< q + (Q \times s) +$ NX,
or IGH $< q + (Q \times s) +$ NPAR + KFC,
or IH $\le q + (Q \times s) +$ NPAR + KFC.

IFAIL = 7

This indicates a failure in the search procedure, with ZSP(1) $\ge$ 1.0E09.

Some output parameters may contain meaningful values – see Section 5 for details.

IFAIL = 8

This indicates a failure to invert H.

Some output parameters may contain meaningful values – see Section 5 for details.

IFAIL = 9

This indicates a failure in F04ASF which is used to solve the equations giving the latest estimates of the backforecasts.

IFAIL = 10

Satisfactory parameter estimates could not be obtained for all parameter types in the model. Inspect array ISF for further information on the parameter type(s) in error.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to
NX$\times$ITC$\times(q+Q\times s+$NPAR$+$KFC$)^2$

9. Example

The following program reads 30 observations from a time series relating to the rate of the earth's rotation about its polar axis. Differencing of order 1 is applied, and the number of non-seasonal parameters is 3, one autoregressive (ϕ), and two moving average (θ). No seasonal effects are taken into account.

The constant is estimated. Up to 25 iterations are allowed.

The initial estimates of ϕ_1, θ_1, θ_2 and c are zero.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13AEF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NPMAX, IEXMAX, NXMAX, IGHMAX, ISTMAX, IH, IWAMAX
      PARAMETER        (NPMAX=10,IEXMAX=50,NXMAX=50,IGHMAX=10,ISTMAX=10,
     +                 IH=IGHMAX,IWAMAX=350)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             C, S
      INTEGER          I, IEX, IFAIL, IGH, IST, ITC, IWA, J, KFC, KPIV,
     +                 KZSP, NDF, NEX, NGH, NIT, NPAR, NST, NX
```

```
*     .. Local Arrays ..
      real              AL(IEXMAX), EX(IEXMAX), EXR(IEXMAX), G(IGHMAX),
     +                  H(IH,IGHMAX), HC(IH,IGHMAX), PAR(NPMAX),
     +                  SD(IGHMAX), ST(ISTMAX), WA(IWAMAX), X(NXMAX),
     +                  ZSP(4)
      INTEGER           ICOUNT(6), ISF(4), MR(7)
*     .. External Subroutines ..
      EXTERNAL          G13AEF, PIV
*     .. Intrinsic Functions ..
      INTRINSIC         MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13AEF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX, (MR(I),I=1,7)
      IF (NX.GT.0 .AND. NX.LE.NXMAX) THEN
         READ (NIN,*) (X(I),I=1,NX)
         NPAR = MR(1) + MR(3) + MR(4) + MR(6)
         IF (NPAR.GT.0 .AND. NPAR.LE.NPMAX) THEN
            DO 20 I = 1, NPAR
               PAR(I) = 0.0e0
   20       CONTINUE
            KFC = 1
            C = 0.0e0
            IEX = MR(3) + (MR(6)*MR(7)) + NX
            IGH = MR(3) + (MR(6)*MR(7)) + NPAR + KFC
            IST = (MR(4)*MR(7)) + MR(2) + (MR(5)*MR(7)) + MR(3) +
     +            MAX(MR(1),(MR(6)*MR(7)))
*           * Set KPIV to 1 to obtain monitoring information *
            KPIV = 0
            NIT = 25
            ZSP(1) = 0.001e0
            ZSP(2) = 10.0e0
            ZSP(3) = 1000.0e0
            ZSP(4) = 0.0001e0
            KZSP = 1
            IWA = ((NX+1+MR(1)+(MR(4)*MR(7))+MR(3)+(MR(6)*MR(7)))*8) +
     +            (9*NPAR)
            IF (IEX.LE.IEXMAX .AND. IGH.LE.IGHMAX .AND. IST.LE.
     +          ISTMAX .AND. IWA.LE.IWAMAX) THEN
               IFAIL = 1
*
               CALL G13AEF(MR,PAR,NPAR,C,KFC,X,NX,ICOUNT,EX,EXR,AL,IEX,
     +                     S,G,IGH,SD,H,IH,ST,IST,NST,PIV,KPIV,NIT,ITC,
     +                     ZSP,KZSP,ISF,WA,IWA,HC,IFAIL)
*
               IF (IFAIL.NE.0) WRITE (NOUT,99999)
     +             'G13AEF fails. IFAIL = ', IFAIL
               IF (IFAIL.EQ.0 .OR. IFAIL.GE.7) THEN
                  NEX = ICOUNT(4)
                  NDF = ICOUNT(5)
                  NGH = ICOUNT(6)
                  WRITE (NOUT,*)
                  WRITE (NOUT,99998) 'Convergence was achieved after',
     +              ITC, ' cycles'
                  WRITE (NOUT,*)
                  WRITE (NOUT,*)
     +'Final values of the PAR parameters and the constant are as follow
     +s'
                  WRITE (NOUT,99997) (PAR(I),I=1,NPAR), C
                  WRITE (NOUT,*)
                  WRITE (NOUT,99996) 'Residual sum of squares is', S,
     +              '  with', NDF, ' degrees of freedom'
                  WRITE (NOUT,*)
                  WRITE (NOUT,*) 'The final values of ZSP were'
                  WRITE (NOUT,99995) (ZSP(I),I=1,4)
                  WRITE (NOUT,*)
                  WRITE (NOUT,99999)
     +              'The number of parameters estimated was', NGH
```

```
                  WRITE (NOUT,*)
     +              '( backward forecasts, PAR and C, in that order )'
                  WRITE (NOUT,*)
                  WRITE (NOUT,*) 'The corresponding G array holds'
                  WRITE (NOUT,99994) (G(I),I=1,NGH)
                  IF ((IFAIL.EQ.0 .OR. IFAIL.EQ.9) .AND. ITC.GT.0) THEN
                     WRITE (NOUT,*)
                     WRITE (NOUT,*) 'The corresponding SD array holds'
                     WRITE (NOUT,99994) (SD(I),I=1,NGH)
                     WRITE (NOUT,*)
                     WRITE (NOUT,*)
     +             'The corresponding H matrix holds second derivatives'
                     WRITE (NOUT,*)
     +                 'in the upper half (including the main diagonal)'
                     WRITE (NOUT,*)
     +              'and correlation coefficients in the lower triangle'
                     DO 40 I = 1, NGH
                        WRITE (NOUT,99993) (H(I,J),J=1,NGH)
   40                CONTINUE
                  END IF
                  WRITE (NOUT,*)
                  WRITE (NOUT,99992) 'EX, EXR, and AL each hold', NEX,
     +               ' values made up of', ICOUNT(1),
     +               ' back forecast(s),'
                  WRITE (NOUT,99991) ICOUNT(2),
     +               ' differenced values, and'
                  WRITE (NOUT,99991) ICOUNT(3),
     +               ' element(s) of reconstituted information'
                  WRITE (NOUT,*)
                  WRITE (NOUT,*) '  EX'
                  WRITE (NOUT,99990) (EX(I),I=1,NEX)
                  IF (IFAIL.EQ.0 .OR. IFAIL.EQ.9) THEN
                     WRITE (NOUT,*)
                     WRITE (NOUT,*) '  EXR'
                     WRITE (NOUT,99990) (EXR(I),I=1,NEX)
                  END IF
                  IF (IFAIL.EQ.0) THEN
                     WRITE (NOUT,*)
                     WRITE (NOUT,*) '  AL'
                     WRITE (NOUT,99990) (AL(I),I=1,NEX)
                  END IF
                  IF (IFAIL.EQ.0 .OR. IFAIL.EQ.9) THEN
                     WRITE (NOUT,*)
                     WRITE (NOUT,99998) 'The state set consists of',
     +                 NST, ' values'
                     WRITE (NOUT,99990) (ST(I),I=1,NST)
                  END IF
               END IF
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I2)
99998 FORMAT (1X,A,I3,A)
99997 FORMAT (1X,4F10.4)
99996 FORMAT (1X,A,F10.3,A,I4,A)
99995 FORMAT (1X,4e15.4)
99994 FORMAT (1X,10F9.4)
99993 FORMAT (1X,6F11.3)
99992 FORMAT (1X,A,I5,A,I5,A)
99991 FORMAT (1X,I5,A)
99990 FORMAT (1X,5F11.4)
      END
*
```

```
      SUBROUTINE PIV(MR,PAR,NPAR,C,KFC,ICOUNT,S,G,H,IH,NGH,ITC,ZSP)
*     .. Parameters ..
      INTEGER          NOUT
      PARAMETER        (NOUT=6)
*     .. Scalar Arguments ..
      real             C, S
      INTEGER          IH, ITC, KFC, NGH, NPAR
*     .. Array Arguments ..
      real             G(NGH), H(IH,NGH), PAR(NPAR), ZSP(4)
      INTEGER          ICOUNT(6), MR(7)
*     .. Executable Statements ..
      WRITE (NOUT,*)
      WRITE (NOUT,99999) 'Iteration', ITC,
     +  '  residual sum of squares = ', S
      RETURN
*
99999 FORMAT (1X,A,I3,A,e11.4)
      END
```

9.2. Program Data

```
G13AEF Example Program Data
 30  1  1  2  0  0  0  0
 -217 -177 -166 -136 -110  -95  -64  -37  -14  -25
  -51  -62  -73  -88 -113 -120  -83  -33  -19   21
   17   44   44   78   88  122  126  114   85   64
```

9.3. Program Results

```
 G13AEF Example Program Results

 Convergence was achieved after 16 cycles

 Final values of the PAR parameters and the constant are as follows
    -0.0547   -0.5568   -0.6636    9.9807

 Residual sum of squares is  9397.924  with  25 degrees of freedom

 The final values of ZSP were
      0.1000E-14    0.1000E+02    0.1000E+04    0.1000E-03

 The number of parameters estimated was 6
 ( backward forecasts, PAR and C, in that order )

 The corresponding G array holds
   -0.1512  -0.2343  -6.4097  13.5617 -72.6232  -0.1642

 The corresponding SD array holds
   14.8379  15.1887   0.3507   0.2709   0.1695   7.3893

 The corresponding H matrix holds second derivatives
 in the upper half (including the main diagonal)
 and correlation coefficients in the lower triangle
      1.942     -0.618      0.244      1.794     -0.836      0.241
      0.342      1.945     -0.165     -0.251      1.795      0.859
     -0.011      0.006   9041.606  -9682.485    546.264      0.818
     -0.012      0.006      0.813  17030.920  -5676.087      6.942
     -0.002     -0.001      0.367      0.479  17027.635      6.331
     -0.146     -0.260     -0.041     -0.048     -0.037      7.434
```

```
EX, EXR, and AL each hold   32 values made up of    2 back forecast(s),
   29 differenced values, and
    1 element(s) of reconstituted information

  EX
    19.5250      5.8753     40.0000     11.0000     30.0000
    26.0000     15.0000     31.0000     27.0000     23.0000
   -11.0000    -26.0000    -11.0000    -11.0000    -15.0000
   -25.0000     -7.0000     37.0000     50.0000     14.0000
    40.0000     -4.0000     27.0000      0.0000     34.0000
    10.0000     34.0000      4.0000    -12.0000    -29.0000
   -21.0000     64.0000

  EXR
    19.5250     -3.9279     19.5711     -5.6291     10.2221
    15.1582     -9.3276     16.4285     15.2115     -5.4211
   -27.3444    -18.3061      5.3890    -12.9812    -22.4767
   -15.2183      4.4944     33.6867     19.7586    -27.1470
    32.2426    -12.2765      1.6941     -1.8465     23.3772
   -10.4576     14.3302     -5.7061    -28.6401    -20.4502
    -2.7215      0.0000

  AL
    19.5250      5.8753     30.0193      1.0193     20.0193
    16.0193      5.0193     21.0193     17.0193     13.0193
   -20.9807    -35.9807    -20.9807    -20.9807    -24.9807
   -34.9807    -16.9807     27.0193     40.0193      4.0193
    30.0193    -13.9807     17.0193     -9.9807     24.0193
     0.0193     24.0193     -5.9807    -21.9807    -38.9807
   -30.9807      0.0000

The state set consists of  4 values
    64.0000    -30.9807    -20.4502     -2.7215
```

With KPIV set to 1 in the example program, maintaining information similar to that below is obtained:

```
G13AEF Example Program Results

Iteration  0  residual sum of squares =  0.1794E+05

Iteration  1  residual sum of squares =  0.1147E+05

Iteration  2  residual sum of squares =  0.9934E+04

Iteration  3  residual sum of squares =  0.9550E+04

Iteration  4  residual sum of squares =  0.9515E+04

Iteration  5  residual sum of squares =  0.9501E+04

Iteration  6  residual sum of squares =  0.9488E+04

Iteration  7  residual sum of squares =  0.9478E+04

Iteration  8  residual sum of squares =  0.9465E+04

Iteration  9  residual sum of squares =  0.9452E+04

Iteration 10  residual sum of squares =  0.9437E+04

Iteration 11  residual sum of squares =  0.9424E+04
```

```
    Iteration 12  residual sum of squares =  0.9413E+04

    Iteration 13  residual sum of squares =  0.9406E+04

    Iteration 14  residual sum of squares =  0.9402E+04

    Iteration 15  residual sum of squares =  0.9400E+04

    Iteration 16  residual sum of squares =  0.9399E+04
```

G13AFF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13AFF is an easy-to-use version of G13AEF. It fits a seasonal autoregressive integrated moving average (ARIMA) model to an observed time series, using a nonlinear least-squares procedure incorporating backforecasting. Parameter estimates are obtained, together with appropriate standard errors. The residual series is returned, and information for use in forecasting the time series is produced for use in the G13AGF and G13AHF.

The estimation procedure is iterative, starting with initial parameter values such as may be obtained using G13ADF. It continues until a specified convergence criterion is satisfied or until a specified number of iterations have been carried out. The progress of the iteration can be monitored by means of an optional printing facility.

2. Specification

```
SUBROUTINE G13AFF (MR, PAR, NPAR, C, KFC, X, NX, S, NDF, SD, NPPC,
1                  CM, ICM, ST, NST, KPIV, NIT, ITC, ISF, RES, IRES,
2                  NRES, IFAIL)
INTEGER        MR(7), NPAR, KFC, NX, NDF, NPPC, ICM, NST, KPIV,
1              NIT, ITC, ISF(4), IRES, NRES, IFAIL
real           PAR(NPAR), C, X(NX), S, SD(NPPC), CM(ICM,NPPC),
1              ST(NX), RES(IRES)
```

3. Description

The time series $x_1,x_2,...,x_n$ supplied to the routine is assumed to follow a seasonal autoregressive integrated moving average (ARIMA) model defined as follows:

$$\nabla^d \nabla_s^D x_t - c = w_t$$

where $\nabla^d \nabla_s^D x_t$ is the result of applying non-seasonal differencing of order d and seasonal differencing of seasonality s and order D to the series x_t, as outlined in the description of G13AAF. The differenced series is then of length $N = n - d'$, where $d' = d + (D\times s)$ is the generalized order of differencing. The scalar c is the expected value of the differenced series, and the series $w_1,w_2,...,w_N$ follows a zero-mean stationary autoregressive moving average (ARMA) model defined by a pair of recurrence equations. These express w_t in terms of an uncorrelated series a_t, via an intermediate series e_t. The first equation describes the seasonal structure:

$$w_t = \Phi_1 w_{t-s} + \Phi_2 w_{t-2\times s} + ... + \Phi_P w_{t-P\times s} + e_t - \Theta_1 e_{t-s} - \Theta_2 e_{t-2\times s} - ... - \Theta_Q e_{t-Q\times s}.$$

The second equation describes the non-seasonal structure. If the model is purely non-seasonal the first equation is redundant and e_t above is equated with w_t:

$$e_t = \phi_1 e_{t-1} + \phi_2 e_{t-2} + ... + \phi_p e_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - ... - \theta_q a_{t-q}.$$

Estimates of the model parameters defined by

$$\phi_1,\phi_2,...,\phi_p,\theta_1,\theta_2,...,\theta_q,$$
$$\Phi_1,\Phi_2,...,\Phi_P,\Theta_1,\Theta_2,...,\Theta_Q$$

and (optionally) c are obtained by minimizing a quadratic form in the vector $w = (w_1,w_2,...,w_N)^T$.

The minimization process is iterative, iterations being performed until convergence is achieved (see Section 3 of G13AEF for full details), or until the user-specified maximum number of iterations are completed.

The final values of the residual sum of squares and the parameter estimates are used to obtain asymptotic approximations to the standard deviations of the parameters, and the correlation matrix for the parameters. The 'state set' array of information required by forecasting is also returned.

Note: if the maximum number of iterations are performed without convergence, these quantities may not be reliable. In this case, the sequence of iterates should be checked, using the optional monitoring routine, to verify that convergence is adequate for practical purposes.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

[2] MARQUARDT, D.W.
An algorithm for least-squares estimation of nonlinear parameters,
J. Soc. Ind. Appl. Math. 11, p. 431, 1963.

5. Parameters

1: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s) of the ARIMA model whose parameters are to be estimated. p, q, P and Q refer respectively to the number of autoregressive (ϕ), moving average (θ), seasonal autoregressive (Φ) and seasonal moving average (Θ) parameters. d, D and s refer respectively to the order of non-seasonal differencing, the order of seasonal differencing and the seasonal period.

Constraints: $p,d,q,P,D,Q,s \geq 0$,
$p + q + P + Q > 0$,
$s \neq 1$,
if $s = 0$, then $P + D + Q = 0$,
if $s > 1$, then $P + D + Q > 0$.

2: PAR(NPAR) – ***real*** array. *Input/Output*

On entry: the initial estimates of the p values of the ϕ parameters, the q values of the θ parameters, the P values of the Φ parameters and the Q values of the Θ parameters, in that order.

On exit: PAR contains the latest values of the estimates of these parameters.

3: NPAR – INTEGER. *Input*

On entry: the total number of ϕ, θ, Φ, and Θ parameters to be estimated.

Constraint: NPAR $= p + q + P + Q$.

4: C – ***real.*** *Input/Output*

On entry: if KFC = 0, C must contain the expected value, c, of the differenced series if KFC = 1, C must contain an initial estimate of c.

Therefore, if C and KFC are both zero on entry, there is no constant correction.

On exit: if KFC = 0, C is unchanged if KFC = 1, C contains the latest estimate of c.

5: KFC – INTEGER. *Input*

On entry: the value of 0 if the constant is to remain fixed, and 1 if it is to be estimated.

Constraint: NFC = 0 or 1.

6: X(NX) – ***real*** array. *Input*

On entry: the n values of the original, undifferenced time series.

7: NX – INTEGER. *Input*

On entry: the length of the original, undifferenced time series, n.

8: S – ***real***. *Output*

On exit: the residual sum of squares after the latest series of parameter estimates has been incorporated into the model. If the routine exits with a faulty input parameter, S contains zero.

9: NDF – INTEGER. *Output*

On exit: the number of degrees of freedom associated with S.

Constraint: NDF $= n - d - D{\times}s - p - q - P - Q -$ KFC.

10: SD(NPPC) – ***real*** array. *Output*

On exit: the standard deviations corresponding to the parameters in the model (p autoregressive, q moving average, P seasonal autoregressive, Q seasonal moving average and c, if estimated, – in that order). If the routine exits with IFAIL containing a value other than 0 or 9, or if the required number of iterations is zero, the contents of SD will be indeterminate.

11: NPPC – INTEGER. *Input*

On entry: the number of ϕ, θ, Φ, Θ and c parameters to be estimated. NPPC $= p + q + P + Q + 1$ if the constant is being estimated and NPPC $= p + q + P + Q$ if not.

Constraint: NPPC = NPAR + KFC.

12: CM(ICM,NPPC) – ***real*** array. *Output*

On exit: the correlation coefficients associated with each pair of the NPPC parameters. These are held in the first NPPC rows and the first NPPC columns of CM. These correlation coefficients are indeterminate if IFAIL contains on exit a value other than 0 or 9, or if the required number of iterations is zero.

13: ICM – INTEGER. *Input*

On entry: the first dimension of the array CM as declared in the (sub)program from which G13AFF is called.

Constraint: ICM ≥ NPPC.

14: ST(NX) – ***real*** array. *Output*

On exit: the value of the state set in its first NST elements. If the routine exits with IFAIL containing a value other than 0 or 9, the contents of ST will be indeterminate.

15: NST – INTEGER. *Output*

On exit: the size of the state set. NST $= P{\times}s + D{\times}s + d + q + \max(p, Q{\times}s)$.

NST should be used subsequently in G13AGF and G13AHF as the DIMENSION of ST.

16: KPIV – INTEGER. *Input*

On entry: KPIV must be non-zero if the progress of the optimization is to be monitored using the built-in printing facility. Otherwise KPIV must contain zero. If selected, monitoring output will be sent to the current advisory message unit defined by X04ABF. For each iteration, the heading

```
G13AFZ MONITORING OUTPUT - ITERATION n
```

followed by the parameter values, and residual sum of squares, are printed. In certain implementations, G13AFZ may be renamed as AFZG13.

17: NIT – INTEGER. *Input*

On entry: the maximum number of iterations to be performed.

Constraint: NIT $\geq$ 0.

18: ITC – INTEGER. *Output*

On exit: the number of iterations performed.

19: ISF(4) – INTEGER array. *Output*

On exit: the first 4 elements of ISF contain success/failure indicators, one for each of the 4 types of parameter in the model (autoregressive, moving average, seasonal autoregressive, seasonal moving average), in that order.

Each indicator has the interpretation:

–2	On entry parameters of this type have initial estimates which do not satisfy the stationarity or invertibility test conditions.
–1	The search procedure has failed to converge because the latest set of parameter estimates of this type is invalid.
0	No parameter of this type is in the model.
1	Valid final estimates for parameters of this type have been obtained.

20: RES(IRES) – ***real*** array. *Output*

On exit: the first NRES elements of RES contain the model residuals derived from the differenced series. If the routine exits with IFAIL holding a value other than 0 or 9, these elements of RES will be indeterminate. The rest of the array RES is used as workspace.

21: IRES – INTEGER. *Input*

On entry: the dimension of the array RES as declared in the (sub)program from which G13AFF is called.

Constraint: IRES $\geq 15\times Q' + 11n + 13\times\text{NPPC} + 8\times P' + 12 + 2\times(Q'+\text{NPPC})^2$ where $P' = p + (P\times s)$ and $Q' = q + (Q\times s)$.

22: NRES – INTEGER. *Output*

On exit: the number of model residuals returned in RES.

23: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL $\neq$ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.** To suppress the output of an error message when soft failure occurs, set IFAIL to 1.

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

IFAIL = 1

On entry, NPAR $\neq p + q + P + Q$,
or the orders vector MR is invalid (check the constraints in Section 5),
or KFC $\neq$ 0 or 1,
or NPPC $\neq$ NPAR + KFC.

IFAIL = 2

On entry, NX $- d - D \times s \leq$ NPAR + KFC. i.e. the number of terms in the differenced series is not greater than the number of parameters in the model. The model is over-parameterised.

IFAIL = 3

On entry, NIT < 0.

IFAIL = 4

On entry, the required size of the state set array ST is greater than NX. This occurs only for very unusual models with long seasonal periods or large numbers of parameters. First check that the orders vector MR has been set up as intended. If it has, change to G13AEF with ST dimensioned at least (NST), where NST is the value returned by G13AFF, or computed using the formula in Section 5 of this document.

IFAIL = 5

On entry, the workspace array RES is too small. Check the value of IRES against the constraints in Section 5.

IFAIL = 6

On entry, ICM < NPPC.

IFAIL = 7

The search procedure in the algorithm has failed. This may be due to a badly conditioned sum of squares function, or the default convergence criterion may be too strict. Use G13AEF with a less strict convergence criterion.

Some output parameters may contain meaningful values – see Section 5 for details.

IFAIL = 8

The inversion of the Hessian matrix in the calculation of the covariance matrix of the parameter estimates has failed.

Some output parameters may contain meaningful values – see Section 5 for details.

IFAIL = 9

This indicates a failure in F03AFF which is used to solve the equations giving the latest estimates of the backforecasts.

Some output parameters may contain meaningful values – see Section 5 for details.

IFAIL = 10

Satisfactory parameter estimates could not be obtained for all parameter types in the model. Inspect array ISF for further information on the parameter type(s) in error.

IFAIL = 11

An internal error has arisen in partitioning RES for use by G13AEF. This error should not occur – report it to NAG via your site representative.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to NX×ITC×$(q+Q\times s+\text{NPPC})^2$.

9. Example

The following program reads 30 observations from a time series relating to the rate of the earth's rotation about its polar axis. Differencing of order 1 is applied, and the number of non-seasonal parameters is 3, one autoregressive (ϕ) and two moving average (θ). No seasonal effects are taken into account.

The constant is estimated. Up to 50 iterations are allowed.

The initial estimates of ϕ_1, θ_1, θ_2 and c are zero.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13AFF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NPMAX, NPC, ICM, NXMAX, IRSMAX
      PARAMETER        (NPMAX=10,NPC=NPMAX+1,ICM=NPC,NXMAX=50,
     +                 IRSMAX=550)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             C, S
      INTEGER          I, IFAIL, IPD, IQD, IRES, ITC, J, KFC, KPIV, NDF,
     +                 NIT, NPAR, NPPC, NRES, NST, NX
*     .. Local Arrays ..
      real             CM(ICM,NPC), PAR(NPMAX), RES(IRSMAX), SD(NPC),
     +                 ST(NXMAX), X(NXMAX)
      INTEGER          ISF(4), MR(7)
*     .. External Subroutines ..
      EXTERNAL         G13AFF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13AFF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX, (MR(I),I=1,7)
      WRITE (NOUT,*)
      IF (NX.GT.0 .AND. NX.LE.NXMAX) THEN
         READ (NIN,*) (X(I),I=1,NX)
         NPAR = MR(1) + MR(3) + MR(4) + MR(6)
         IF (NPAR.GT.0 .AND. NPAR.LE.NPMAX) THEN
            DO 20 I = 1, NPAR
               PAR(I) = 0.0e0
   20       CONTINUE
            KFC = 1
            C = 0.0e0
            NPPC = NPAR + KFC
*           * Set KPIV to 1 to obtain monitoring information *
            KPIV = 0
            NIT = 50
            IQD = MR(6)*MR(7) + MR(3)
            IPD = MR(4)*MR(7) + MR(1)
            IRES = 15*IQD + 11*NX + 13*NPPC + 8*IPD + 12 + 2*(IQD+NPPC)
     +             **2
            IF (IRES.LE.IRSMAX) THEN
               IFAIL = 1
*
               CALL G13AFF(MR,PAR,NPAR,C,KFC,X,NX,S,NDF,SD,NPPC,CM,ICM,
     +                     ST,NST,KPIV,NIT,ITC,ISF,RES,IRES,NRES,IFAIL)
*
               IF (IFAIL.NE.0) WRITE (NOUT,99997)
     +             'G13AFF fails. IFAIL = ', IFAIL
               IF (IFAIL.EQ.0 .OR. IFAIL.GE.7) THEN
                  WRITE (NOUT,99996) 'Convergence was achieved after',
     +              ITC, ' cycles'
                  WRITE (NOUT,*)
```

```
                 WRITE (NOUT,*)
     +'Final values of the PAR parameters and the constant are as follow
     +s'
                 WRITE (NOUT,99995) (PAR(I),I=1,NPAR), C
                 WRITE (NOUT,*)
                 WRITE (NOUT,99994) 'Residual sum of squares is', S,
     +             ' with', NDF, ' degrees of freedom'
                 IF ((IFAIL.EQ.0 .OR. IFAIL.EQ.9) .AND. ITC.GT.0) THEN
                    WRITE (NOUT,*)
                    WRITE (NOUT,*) 'The corresponding SD array holds'
                    WRITE (NOUT,99993) (SD(I),I=1,NPPC)
                    WRITE (NOUT,*)
                    WRITE (NOUT,*)
     +                'The correlation matrix is as follows'
                    DO 40 I = 1, NPPC
                       WRITE (NOUT,99992) (CM(I,J),J=1,NPPC)
   40               CONTINUE
                 END IF
                 IF (IFAIL.EQ.0 .OR. IFAIL.EQ.9) THEN
                    WRITE (NOUT,*)
                    WRITE (NOUT,99999) 'The residuals consist of',
     +                NRES, ' values'
                    WRITE (NOUT,99998) (RES(I),I=1,NRES)
                    WRITE (NOUT,*)
                    WRITE (NOUT,99996) 'The state set consists of',
     +                NST, ' values'
                    WRITE (NOUT,99992) (ST(I),I=1,NST)
                 END IF
              END IF
           END IF
        END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I4,A)
99998 FORMAT (1X,5F10.4)
99997 FORMAT (1X,A,I2)
99996 FORMAT (1X,A,I3,A)
99995 FORMAT (1X,4F10.4)
99994 FORMAT (1X,A,F10.3,A,I4,A)
99993 FORMAT (1X,10F9.4)
99992 FORMAT (1X,6F11.3)
      END
```

9.2. Program Data

```
G13AFF Example Program Data
 30  1  1  2  0  0  0  0
 -217 -177 -166 -136 -110  -95  -64  -37  -14  -25
  -51  -62  -73  -88 -113 -120  -83  -33  -19   21
   17   44   44   78   88  122  126  114   85   64
```

9.3. Program Results

```
G13AFF Example Program Results

Convergence was achieved after 25 cycles

Final values of the PAR parameters and the constant are as follows
   -0.0543   -0.5548   -0.6734    9.9848

Residual sum of squares is  9397.220  with  25 degrees of freedom

The corresponding SD array holds
   0.3457   0.2636   0.1665   7.4170
```

```
The correlation matrix is as follows
      1.000      0.807      0.355     -0.040
      0.807      1.000      0.468     -0.049
      0.355      0.468      1.000     -0.038
     -0.040     -0.049     -0.038      1.000

The residuals consist of  29 values
   19.6275   -5.3093    9.7983   15.2412   -9.1693
   16.1107   15.3929   -5.4500  -27.6205  -18.1306
    5.7202  -13.0881  -22.7151  -14.9256    4.6930
   33.5406   19.7138  -27.3360   32.1231  -11.7681
    1.1524   -1.7756   23.6821  -10.6238   13.9619
   -5.2727  -28.7868  -20.6573   -2.2555

The state set consists of  4 values
     64.000    -30.985    -20.657     -2.256
```

With KPIV set to 1 in the example program, monitoring information similar to that below is obtained:

```
G13AFF Example Program Results


G13AFZ MONITORING OUTPUT - ITERATION   0
AUTOREGRESSIVE PARAMETERS
   0.0000E+00
MOVING AVERAGE PARAMETERS
   0.0000E+00   0.0000E+00
CONSTANT TERM
   0.0000E+00
RESIDUAL SUM OF SQUARES
   0.1794E+05

G13AFZ MONITORING OUTPUT - ITERATION   1
AUTOREGRESSIVE PARAMETERS
   0.1814E+00
MOVING AVERAGE PARAMETERS
  -0.1814E+00  -0.2254E+00
CONSTANT TERM
   0.3335E+01
RESIDUAL SUM OF SQUARES
   0.1147E+05
   .
   .
```

intermediate results omitted

```
   .
   .
G13AFZ MONITORING OUTPUT - ITERATION  24
AUTOREGRESSIVE PARAMETERS
  -0.5456E-01
MOVING AVERAGE PARAMETERS
  -0.5551E+00  -0.6731E+00
CONSTANT TERM
   0.9985E+01
RESIDUAL SUM OF SQUARES
   0.9397E+04

G13AFZ MONITORING OUTPUT - ITERATION  25
AUTOREGRESSIVE PARAMETERS
  -0.4933E-01
MOVING AVERAGE PARAMETERS
  -0.5501E+00  -0.6705E+00
CONSTANT TERM
   0.9975E+01
RESIDUAL SUM OF SQUARES
   0.9397E+04
```

G13AGF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13AGF accepts a series of new observations of a time series, the model of which is already fully specified, and updates the 'state set' information for use in constructing further forecasts. The previous specifications of the time series model should have been obtained by using G13AEF or G13AFF to estimate the relevant parameters. The supplied state set will originally have been produced by G13AEF or G13AFF, but may since have been updated by earlier calls to G13AGF.

A set of residuals corresponding to the new observations is returned. These may be of use in checking that the new observations conform to the previously fitted model.

2. Specification

```
 SUBROUTINE G13AGF (ST, NST, MR, PAR, NPAR, C, ANX, NUV, ANEXR, WA,
1                   NWA, IFAIL)
 INTEGER        NST, MR(7), NPAR, NUV, NWA, IFAIL
 real           ST(NST), PAR(NPAR), C, ANX(NUV), ANEXR(NUV),
1               WA(NWA)
```

3. Description

The time series model is specified as outlined in Section 3 of G13AEF or G13AFF. This also describes how the state set, which contains the minimum amount of time series information needed to construct forecasts, is made up of

(i) the differenced series w_t (uncorrected for the constant c), for $(N-P\times s) < t \le N$,

(ii) the d' values required to reconstitute the original series x_t from the differenced series w_t,

(iii) the intermediate series e_t, for $(N-\max(p,Q\times s)) < t \le N$,

(iv) the residual series a_t, for $(N-q) < t \le N$.

If the number of original undifferenced observations was n, then $d' = d + (D\times s)$ and $N = n - d'$.

To update the state set, given a number of new undifferenced observations x_t, $t = n+1,n+2,...,n+k$, the four series above are first reconstituted.

Differencing and residual calculation operations are then applied to the new observations and k new values of w_t, e_t and a_t are derived.

The first k values in these three series are then discarded and a new state set is obtained.

The residuals in the a_t series corresponding to the k new observations are preserved in an output array. The parameters of the time series model are not changed in this routine.

4. References

None.

5. Parameters

1: ST(NST) – ***real*** array. *Input/Output*

On entry: the state set derived from G13AEF or G13AFF, or as modified using earlier calls of G13AGF.

On exit: the updated values of the state set.

2: NST – INTEGER. *Input*

On entry: the number of values in the state set array ST.

Constraint: NST = $P \times s + D \times s + d + q + \max(p, Q \times s)$. (As returned by G13AEF or G13AFF).

3: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s) of the ARIMA model, in the usual notation.

Constraints: $p, d, q, P, D, Q, s \geq 0$,
$p + q + P + Q > 0$,
$s \neq 1$,
if $s = 0$, then $P + D + Q = 0$,
if $s > 1$, then $P + D + Q > 0$.

4: PAR(NPAR) – ***real*** array. *Input*

On entry: the estimates of the p values of the ϕ parameters, the q values of the θ parameters, the P values of the Φ parameters and the Q values of the Θ parameters in the model – in that order, using the usual notation.

5: NPAR – INTEGER. *Input*

On entry: the number of ϕ, θ, Φ, Θ parameters in the model.

Constraint: NPAR = $p + q + P + Q$.

6: C – ***real.*** *Input*

On entry: the constant to be subtracted from the differenced data.

7: ANX(NUV) – ***real*** array. *Input*

On entry: the new undifferenced observations which are to be used to update ST.

8: NUV – INTEGER. *Input*

On entry: the number of new observations in ANX, k.

9: ANEXR(NUV) – ***real*** array. *Output*

On exit: the residuals corresponding to the new observations in ANX.

10: WA(NWA) – ***real*** array. *Workspace*
11: NWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13AGF is called.

Constraint: NWA $\geq$ (4×NPAR+3×NST).

12: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NPAR $\neq p + q + P + Q$,
or the orders vector MR is invalid (check the constraints in Section 5).

IFAIL = 2

On entry, NST $\neq P\times s + D\times s + d + q + \max(Q\times s,p)$.

IFAIL = 3

On entry, NUV ≤ 0.

IFAIL = 4

On entry, NWA < 4×NPAR + 3×NST.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to NUV×NPAR.

9. Example

The following program is based on data derived from a study of monthly airline passenger totals (in thousands) to which a logarithmic transformation had been applied. The time series model was based on seasonal and non-seasonal differencing both of order 1, with seasonal period 12. The number of parameters estimated was two; a non-seasonal moving average parameter θ_1 with value 0.327 and a seasonal moving average parameter Θ_1 with value 0.6270. There was no constant correction. These, together with the state set array, were obtained using G13AEF.

12 new observations are supplied. The routine updates the state set and outputs a set of residuals corresponding to the new observations.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13AGF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NPMAX, NSTMAX, NNVMAX, NWAMAX
      PARAMETER        (NPMAX=10,NSTMAX=50,NNVMAX=50,NWAMAX=150)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             C
      INTEGER          I, IFAIL, NNV, NPAR, NST, NWA
*     .. Local Arrays ..
      real             ANV(NNVMAX), ANVR(NNVMAX), PAR(NPMAX),
     +                 ST(NSTMAX), WA(NWAMAX)
      INTEGER          MR(7)
*     .. External Subroutines ..
      EXTERNAL         G13AGF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13AGF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NNV
      READ (NIN,*) (MR(I),I=1,7)
      NPAR = MR(1) + MR(3) + MR(4) + MR(6)
      NST = MR(4)*MR(7) + MR(5)*MR(7) + MR(2) + MR(3) + MAX(MR(1),MR(6)
     +      *MR(7))
      NWA = 4*NPAR + 3*NST
      IF (NNV.GT.0 .AND. NNV.LE.NNVMAX .AND. NPAR.GT.0 .AND. NPAR.LE.
     +    NPMAX .AND. NST.GT.0 .AND. NST.LE.NSTMAX) THEN
         READ (NIN,*) (PAR(I),I=1,NPAR), C
         READ (NIN,*) (ST(I),I=1,NST)
         READ (NIN,*) (ANV(I),I=1,NNV)
         IFAIL = 0
*
         CALL G13AGF(ST,NST,MR,PAR,NPAR,C,ANV,NNV,ANVR,WA,NWA,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     'The updated state set array now holds the values'
         WRITE (NOUT,99999) (ST(I),I=1,NST)
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'The residuals corresponding to the', NNV
         WRITE (NOUT,*) 'values used to update the system are'
         WRITE (NOUT,99999) (ANVR(I),I=1,NNV)
      END IF
      STOP
*
99999 FORMAT (1X,8F8.4)
99998 FORMAT (1X,A,I3,A)
      END
```

9.2. Program Data

```
G13AGF Example Program Data
 12
  0   1   1   0   1   1  12
 0.3270  0.6270  0.0000
 0.0118 -0.0669  0.1296 -0.0394  0.0422  0.1809  0.1211  0.0281
-0.2231 -0.1181 -0.1468  0.0835  5.8201 -0.0157 -0.0361 -0.0266
-0.0199  0.0298  0.0290  0.0147  0.0373 -0.0931  0.0223 -0.0172
-0.0353 -0.0413
 5.8861  5.8348  6.0064  5.9814  6.0403  6.1570  6.3063  6.3261
 6.1377  6.0088  5.8916  6.0039
```

9.3. Program Results

```
 G13AGF Example Program Results

 The updated state set array now holds the values
   0.0660 -0.0513  0.1716 -0.0250  0.0589  0.1167  0.1493  0.0198
  -0.1884 -0.1289 -0.1172  0.1123  6.0039  0.0444 -0.0070  0.0253
   0.0019  0.0354 -0.0460  0.0374  0.0151 -0.0237  0.0032  0.0188
   0.0067  0.0126

 The residuals corresponding to the 12
 values used to update the system are
   0.0309  0.0031  0.0263  0.0105  0.0388 -0.0333  0.0265  0.0238
  -0.0159 -0.0020  0.0182  0.0126
```

G13AHF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13AHF produces forecasts of a time series, given a time series model which has already been fitted to the time series using G13AEF or G13AFF. The original observations are not required, since G13AHF uses as input either the original state set produced by G13AEF or G13AFF or the state set updated by a series of new observations using G13AGF. Standard errors of the forecasts are also provided.

2. Specification

```
      SUBROUTINE G13AHF (ST, NST, MR, PAR, NPAR, C, RMS, NFV, FVA, FSD,
     1                   WA, NWA, IFAIL)
      INTEGER            NST, MR(7), NPAR, NFV, NWA, IFAIL
      real               ST(NST), PAR(NPAR), C, RMS, FVA(NFV), FSD(NFV),
     1                   WA(NWA)
```

3. Description

The original time series is x_t, for $t = 1,2,...,n$ and parameters have been fitted to the model of this time series using G13AEF or G13AFF.

Forecasts of x_t, for $t = n+1,n+2,...,n+L$, are calculated in five stages, as follows:

(i) set $a_t = 0$ for $t = N+1,N+2,...,N+L$,
where $N = n - d - (D\times s)$ is the number of differenced values in the series,

(ii) calculate the values of e_t for $t = N+1,N+2,...,N+L$,
where $e_t = \phi_1 \times e_{t-1} + ... + \phi_p \times e_{t-p} + a_t - \theta_1 \times a_{t-1} - ... - \theta_q \times a_{t-q}$,

(iii) calculate the values of w_t for $t = N+1,N+2,...,N+L$,
where
$w_t = \Phi_1 \times w_{t-s} + ... + \Phi_P \times w_{t-s\times P} + e_t - \Theta_1 \times e_{t-s} - ... - \Theta_Q \times e_{t-s\times Q}$,
where w_t for $t \leq N$ are the first $s\times P$ values in the state set, corrected for the constant,

(iv) add the constant term c to give the differenced series
$\nabla^d \nabla_s^D x_t = w_t + c$ for $t = N+1,N+2,...,N+L$,

(v) the differencing operations are reversed to reconstitute x_t for $t = n+1,n+2,...n+L$.

The standard errors of these forecasts are given by $s_t = \{V\times(\psi_0^2+\psi_1^2+...+\psi_{t-n-1}^2)\}^{\frac{1}{2}}$ for $t = n+1,n+2,...,n+L$, where $\psi_0 = 1$, V is the residual variance of a_t and ψ_j is the coefficient expressing the dependence of x_t on a_{t-j}.

To calculate ψ_j for $j = 1,2,...,(L-1)$ the following device is used.

A copy of the state set is initialised to zero throughout and the calculations outlined above for the construction of forecasts are carried out with the settings $a_{N+1} = 1$, and $a_t = 0$ for $t = N+2,N+3,...,N+L$.

The resulting quantities corresponding to the sequence $x_{N+1},x_{N+2},...,x_{N+L}$ are precisely 1, $\psi_1,\psi_2,...,\psi_{L-1}$.

The supplied time series model is used throughout these calculations, with the exception that the constant term c is taken to be zero.

4. References

None.

5. Parameters

1: ST(NST) – ***real*** array. *Input*

On entry: the state set derived from G13AEF or G13AFF originally, or as modified using earlier calls of G13AGF.

2: NST – INTEGER. *Input*

On entry: the number of values in the state set array ST.

Constraint: NST = $P \times s + D \times s + d + q + \max(p,Q \times s)$. (As returned by G13AEF or G13AFF).

3: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s) of the ARIMA model, in the usual notation.

Constraints: $p,d,q,P,D,Q,s \geq 0$,
$p + q + P + Q > 0$,
$s \neq 1$,
if $s = 0$, then $P + D + Q = 0$,
if $s > 1$, then $P + D + Q > 0$

4: PAR(NPAR) – ***real*** array. *Input*

On entry: the estimates of the p values of the ϕ parameters, the q values of the θ parameters, the P values of the Φ parameters and the Q values of the Θ parameters which specify the model and which were output originally by G13AEF or G13AFF.

5: NPAR – INTEGER. *Input*

On entry: the number of ϕ, θ, Φ and Θ parameters in the model.

Constraint: NPAR = $p + q + P + Q$.

6: C – ***real.*** *Input*

On entry: the value of the model constant, c. This will have been output by G13AEF or G13AFF.

7: RMS – ***real.*** *Input*

On entry: the residual variance, V, associated with the model. If G13AFF was used to estimate the model, RMS should be set to S/NDF, where S and NDF were output by G13AFF. If G13AEF was used to estimate the model, RMS should be set to S/ICOUNT(5), where S and ICOUNT(5) were output by G13AEF.

Constraint: RMS $\geq$ 0.0.

8: NFV – INTEGER. *Input*

On entry: the required number of forecasts, L.

Constraint: NFV > 0.

9: FVA(NFV) – ***real*** array. *Output*

On exit: NFV forecast values relating to the original undifferenced series.

10: FSD(NFV) – ***real*** array. *Output*

On exit: the standard errors associated with each of the NFV forecast values in FVA.

11: WA(NWA) – ***real*** array. *Workspace*

12: NWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13AHF is called.

Constraint: NWA ≥ (4×NPAR+3×NST).

13: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NPAR $\neq$ $p + q + P + Q$,
or the orders vector MR is invalid (check the constraints given in Section 5).

IFAIL = 2

On entry, NST $\neq$ $P{\times}s + D{\times}s + d + q + \max(Q{\times}s,p)$.

IFAIL = 3

On entry, NFV ≤ 0.

IFAIL = 4

On entry, NWA < 4×NPAR + 3×NST.

IFAIL = 5

On entry, RMS < 0.0.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to NFV×NPAR.

The storage required by internally declared arrays, including those of auxiliary routines, is 4 INTEGER elements.

9. Example

The following example is based on the data derived in the example used to illustrate G13AGF.

These consist of a set of orders indicating that there are two moving average parameters (one non-seasonal, and one seasonal with periodicity 12).

The model constant is zero.

The state set contains 26 values.

In addition the residual mean-square derived when the model was originally fitted is given.

Twelve forecasts and their associated errors are obtained.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13AHF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NPMAX, NSTMAX, NWAMAX, NFVMAX
      PARAMETER        (NPMAX=10,NSTMAX=40,NWAMAX=120,NFVMAX=25)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             C, RMS
      INTEGER          I, IFAIL, NFV, NPAR, NST, NWA
*     .. Local Arrays ..
      real             FSD(NFVMAX), FVA(NFVMAX), PAR(NPMAX), ST(NSTMAX),
     +                 WA(NWAMAX)
      INTEGER          MR(7)
*     .. External Subroutines ..
      EXTERNAL         G13AHF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13AHF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NFV
      READ (NIN,*) (MR(I),I=1,7)
      NPAR = MR(1) + MR(3) + MR(4) + MR(6)
      NST = MR(4)*MR(7) + MR(5)*MR(7) + MR(2) + MR(3) + MAX(MR(1),MR(6)
     +      *MR(7))
      NWA = 4*NPAR + 3*NST
      IF (NFV.GT.0 .AND. NFV.LE.NFVMAX .AND. NPAR.GT.0 .AND. NPAR.LE.
     +    NPMAX .AND. NST.GT.0 .AND. NST.LE.NSTMAX) THEN
         READ (NIN,*) (PAR(I),I=1,NPAR), C
         READ (NIN,*) (ST(I),I=1,NST)
         READ (NIN,*) RMS
         IFAIL = 0
*
         CALL G13AHF(ST,NST,MR,PAR,NPAR,C,RMS,NFV,FVA,FSD,WA,NWA,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'The required', NFV,
     +     ' forecast values are as follows'
         WRITE (NOUT,99999) (FVA(I),I=1,NFV)
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     'The standard deviations corresponding to the forecasts are'
         WRITE (NOUT,99999) (FSD(I),I=1,NFV)
      END IF
      STOP
*
99999 FORMAT (1X,8F8.4)
99998 FORMAT (1X,A,I3,A)
      END
```

9.2. Program Data

```
G13AHF Example Program Data
  12
  0   1   1   0   1   1  12
  0.3270  0.6262  0.0000
  0.0660 -0.0513  0.1715 -0.0249  0.0588  0.1167  0.1493  0.0199
 -0.1884 -0.1289 -0.1172  0.1122  6.0039  0.0443 -0.0070  0.0252
  0.0020  0.0353 -0.0460  0.0374  0.0151 -0.0237  0.0031  0.0188
  0.0066  0.0125
  0.0014
```

9.3. Program Results

```
G13AHF Example Program Results

The required 12 forecast values are as follows
  6.0381  5.9912  6.1469  6.1207  6.1574  6.3029  6.4288  6.4392
  6.2657  6.1348  6.0059  6.1139

The standard deviations corresponding to the forecasts are
  0.0374  0.0451  0.0517  0.0575  0.0627  0.0676  0.0721  0.0764
  0.0805  0.0843  0.0880  0.0915
```

G13AJF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13AJF applies a fully specified seasonal ARIMA model to an observed time series, generates the state set for forecasting and (optionally) derives a specified number of forecasts together with their standard deviations.

2. Specification

```
    SUBROUTINE G13AJF (MR, PAR, NPAR, C, KFC, X, NX, RMS, ST, IST, NST,
   1                   NFV, FVA, FSD, IFV, ISF, W, IW, IFAIL)
    INTEGER            MR(7), NPAR, KFC, NX, IST, NST, NFV, IFV, ISF(4),
   1                   IW, IFAIL
    real               PAR(NPAR), C, X(NX), RMS, ST(IST), FVA(IFV),
   1                   FSD(IFV), W(IW)
```

3. Description

The time series $x_1,x_2,...,x_n$ supplied to the routine is assumed to follow a seasonal autoregressive integrated moving average (ARIMA) model with known parameters.

The model is defined by the following relations:

(a) $\nabla^d \nabla_s^D x_t - c = w_t$

where $\nabla^d \nabla_s^D x_t$ is the result of applying non-seasonal differencing of order d and seasonal differencing of seasonality s and order D to the series x_t,

and c is a constant.

(b) $w_t = \Phi_1 w_{t-s} + \Phi_2 w_{t-2\times s} + ... + \Phi_P w_{t-P\times s} + e_t - \Theta_1 e_{t-s} - \Theta_2 e_{t-2\times s} - ... - \Theta_Q e_{t-Q\times s}$.

This equation describes the seasonal structure with seasonal period s; in the absence of seasonality it reduces to

$w_t = e_t$.

(c) $e_t = \phi_1 e_{t-1} + \phi_2 e_{t-2} + ... + \phi_p e_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - ... - \theta_q a_{t-q}$.

This equation describes the non-seasonal structure.

Given the series, the constant c, and the model parameters Φ, Θ, ϕ, θ, the routine computes:

(a) The state set required for forecasting. This contains the minimum amount of information required for forecasting and comprises:

(i) the differenced series w_t, for $(N-s\times P) \le t \le N$,

(ii) the $(d+D\times s)$ values required to reconstitute the original series x_t from the differenced series w_t,

(iii) the intermediate series e_t, for $N - \max(p,Q\times s) < t \le N$,

(iv) the residual series a_t, for $(N-q) < t \le N$ where $N = n - (d+D\times s)$.

(b) A set of L forecasts of x_t, for $t = n+1,n+2,...,n+L$ (L may be zero).

The forecasts are generated from the state set, and are identical to those that would be produced from the same state set by G13AHF.

Use of G13AJF should be confined to situations in which the state set for forecasting is unknown. Forecasting from the series requires recalculation of the state set and this is relatively expensive.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

5. Parameters

1: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s) of the ARIMA model, in the usual notation.

Constraints: $p,d,q,P,D,Q,s \geq 0$,
$p + q + P + Q > 0$,
$s \neq 1$,
if $s = 0$, then $P + D + Q = 0$,
if $s > 1$, then $P + D + Q > 0$.

2: PAR(NPAR) – ***real*** array. *Input*

On entry: the p values of the ϕ parameters, the q values of the θ parameters, the P values of the Φ parameters, and the Q values of the Θ parameters, in that order.

3: NPAR – INTEGER. *Input*

On entry: the exact number of ϕ, θ, Φ and Θ parameters.

Constraint: NPAR $= p + q + P + Q$.

4: C – ***real.*** *Input*

On entry: the expected value, c, of the differenced series (i.e. c is the constant correction). Where there is no constant term, C must be set to 0.0.

5: KFC – INTEGER. *Input*

On entry: KFC must be set to 0 if C was not estimated, and 1 if C was estimated. This is irrespective of whether or not C = 0.0. The only effect is that the residual degrees of freedom are one greater when KFC = 0. Assuming the supplied time series to be the same as that to which the model was originally fitted, this ensures an unbiased estimate of residual mean-square.

Constraint: KFC = 0 or 1.

6: X(NX) – ***real*** array. *Input*

On entry: the n values of the original, undifferenced time series.

7: NX – INTEGER. *Input*

On entry: the length of the original, undifferenced time series, n.

8: RMS – ***real.*** *Output*

On exit: the residual variance (mean square) associated with the model.

9: ST(IST) – ***real*** array. *Output*

On exit: the NST values of the state set.

10: IST – INTEGER. *Input*

On entry: the dimension of the array ST as declared in the (sub)program from which G13AJF is called.

Constraint: IST $\geq (P\times s) + d + (D\times s) + q + \max(p,Q\times s)$. The expression on the right-hand side of the inequality is returned in NST.

11: NST – INTEGER. *Output*

On exit: the number of values in the state set array ST.

12: NFV – INTEGER. *Input*

On entry: the required number of forecasts. If NFV $\leq$ 0, no forecasts will be computed.

13: FVA(IFV) – ***real*** array. *Output*

On exit: if NFV > 0, FVA contains the NFV forecast values relating to the original undifferenced time series.

14: FSD(IFV) – ***real*** array. *Output*

On exit: if NFV > 0, FSD contains the estimated standard errors of the NFV forecast values.

15: IFV – INTEGER. *Input*

On entry: the dimension of the array FVA and FSD as declared in the (sub)program from which G13AJF is called.

Constraint: IFV $\geq$ max(1,NFV).

16: ISF(4) – INTEGER array. *Output*

On exit: the first 4 elements of ISF contain validity indicators, one for each of the 4 possible parameter types in the model (autoregressive, moving average, seasonal autoregressive, seasonal moving average), in that order.

Each indicator has the interpretation:

–1 On entry the set of parameter values of this type does not satisfy the stationarity or invertibility test conditions.
0 No parameter of this type is in the model.
1 Valid parameter values of this type have been supplied.

17: W(IW) – ***real*** array. *Workspace*
18: IW – INTEGER. *Input*

On entry: the dimension of the array W as declared in the (sub)program from which G13AJF is called.

Constraint: IW $\geq 6\times n + 5\times(p+q+P+Q) + Q'^2 + 11\times Q' + 3\times P' + 7$
where $Q' = Q\times s + q$ and $P' = P\times s + p$.

19: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NPAR $\neq p + q + P + Q$,
or the orders vector MR is invalid (check the constraints in Section 5),
or KFC $\neq$ 0 or 1.

IFAIL = 2

On entry, NX $- d - D\times s \leq$ NPAR + KFC i.e. the number of terms in the differenced series is not greater than the number of parameters in the model. The model is over-parameterised.

IFAIL = 3

On entry, the workspace array W is too small.

IFAIL = 4

On entry, the state set array ST is too small. It must be at least as large as the exit value of NST.

IFAIL = 5

This indicates a failure in F04ASF which is used to solve the equations giving estimates of the back forecasts.

IFAIL = 6

On entry, valid values were not supplied for all parameter types in the model. Inspect array ISF for further information on the parameter type(s) in error.

IFAIL = 7

On entry, IFV < max(1,NFV).

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to n and the square of the number of backforecasts derived.

9. Example

The data are those used in the example program for G13AFF. Five forecast values and their standard errors, together with the state set, are computed and printed.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13AJF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NPMAX, ISTMAX, IFVMAX, NXMAX, IW
      PARAMETER        (NPMAX=10,ISTMAX=10,IFVMAX=10,NXMAX=30,IW=250)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             C, RMS
      INTEGER          I, IFAIL, IFV, IST, KFC, NFV, NPAR, NST, NX
*     .. Local Arrays ..
      real             FSD(IFVMAX), FVA(IFVMAX), PAR(NPMAX), ST(ISTMAX),
     +                 W(IW), X(NXMAX)
      INTEGER          ISF(4), MR(7)
*     .. External Subroutines ..
      EXTERNAL         G13AJF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13AJF Example Program Results'
```

```
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX, (MR(I),I=1,7), NFV
      NPAR = MR(1) + MR(3) + MR(4) + MR(6)
      IF (NX.GT.0 .AND. NX.LE.NXMAX .AND. NPAR.GT.0 .AND. NPAR.LE.NPMAX)
     +    THEN
         READ (NIN,*) (X(I),I=1,NX)
         READ (NIN,*) (PAR(I),I=1,NPAR)
         READ (NIN,*) KFC, C
         IST = MR(4) + MR(7) + MR(2) + MR(5) + MR(3) + MAX(MR(1),MR(6)
     +         *MR(7))
         IFV = MAX(1,NFV)
         IFAIL = 0
*
         CALL G13AJF(MR,PAR,NPAR,C,KFC,X,NX,RMS,ST,IST,NST,NFV,FVA,FSD,
     +               IFV,ISF,W,IW,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'The residual mean square is ', RMS
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'The state set consists of ', NST, ' values'
         WRITE (NOUT,99997) (ST(I),I=1,NST)
         WRITE (NOUT,*)
         WRITE (NOUT,99996) 'The ', NFV,
     +     ' forecast values and standard errors are -'
         WRITE (NOUT,99995) (FVA(I),FSD(I),I=1,NFV)
      END IF
      STOP
*
99999 FORMAT (1X,A,F9.2)
99998 FORMAT (1X,A,I1,A)
99997 FORMAT (1X,4F11.4)
99996 FORMAT (1X,A,I2,A)
99995 FORMAT (10X,2F10.2)
      END
```

9.2. Program Data

```
G13AJF Example Program Data
 30  1  1  2  0  0  0  0  5
 -217 -177 -166 -136 -110  -95  -64  -37  -14  -25
  -51  -62  -73  -88 -113 -120  -83  -33  -19   21
   17   44   44   78   88  122  126  114   85   64
 -0.0547 -0.5568 -0.6636
 1 9.9807
```

9.3. Program Results

```
 G13AJF Example Program Results

 The residual mean square is    375.91

 The state set consists of 4 values
     64.0000   -30.9807   -20.4495    -2.7212

 The  5 forecast values and standard errors are -
           60.59     19.39
           69.50     34.99
           79.54     54.25
           89.51     67.87
           99.50     79.20
```

G13ASF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13ASF is a diagnostic checking routine suitable for use after fitting a Box-Jenkins ARMA model to a univariate time series using G13AEF or G13AFF. The residual autocorrelation function is returned along with an estimate of its asymptotic standard errors and correlations. Also, G13ASF calculates the Box-Ljung portmanteau statistic and its significance level for testing model adequacy.

2. Specification

```
SUBROUTINE G13ASF (N, V, MR, M, PAR, NPAR, ISHOW, R, RCM, IRCM,
1                  CHI, IDF, SIGLEV, IW, LIW, WORK, LWORK,
2                  IFAIL)
 INTEGER        N, MR(7), M, NPAR, ISHOW, IRCM, IDF, IW(LIW),
1               LIW, LWORK, IFAIL
 real           V(N), PAR(NPAR), R(M), RCM(IRCM,M), CHI, SIGLEV,
1               WORK(LWORK)
```

3. Description

Consider the univariate multiplicative autoregressive-moving average model

$$\phi(B)\Phi(B^s)(W_t-\mu) = \theta(B)\Theta(B^s)\varepsilon_t \qquad (1)$$

where W_t, for $t = 1,2,...,n$ denotes a time series and ε_t, for $t = 1,2,...,n$ is a residual series assumed to be normally distributed with zero mean and variance σ^2 (> 0). The ε_t's are also assumed to be uncorrelated. Here μ is the overall mean term, s is the seasonal period and B is the backward shift operator such that $B^r W_t = W_{t-r}$. The polynomials in (1) are defined as follows:

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p$$

is the non-seasonal autoregressive (AR) operator;

$$\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q$$

is the non-seasonal moving average (MA) operator;

$$\Phi(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - ... - \Phi_P B^{Ps}$$

is the seasonal AR operator; and

$$\Theta(B^s) = 1 - \Theta_1 B^s - \Theta_2 B^{2s} - ... - \Theta_Q B^{Qs}$$

is the seasonal MA operator. The model (1) is assumed to be stationary, that is the zeros of $\phi(B)$ and $\Phi(B^s)$ are assumed to lie outside the unit circle. The model (1) is also assumed to be invertible, that is the zeros of $\theta(B)$ and $\Theta(B^s)$ are assumed to lie outside the unit circle. When both $\Phi(B^s)$ and $\Theta(B^s)$ are absent from the model, that is when $P = Q = 0$, then the model is said to be non-seasonal.

The estimated residual autocorrelation coefficient at lag l, $\hat{r}_l$, is computed as:

$$\hat{r}_l = \frac{\sum_{t=l+1}^{n} (\hat{\varepsilon}_{t-l}-\bar{\varepsilon})(\hat{\varepsilon}_t-\bar{\varepsilon})}{\sum_{t=1}^{n} (\hat{\varepsilon}_t-\bar{\varepsilon})^2}, \qquad l = 1,2,...$$

where $\hat{\varepsilon}_t$ denotes an estimate of the tth residual, ε_t, and $\bar{\varepsilon} = \sum_{t=1}^{n} \hat{\varepsilon}_t/n$. A portmanteau statistic, $Q_{(m)}$, is calculated from the formula (see Box and Ljung [1]):

$$Q_{(m)} = n(n+2)\sum_{l=1}^{m} \hat{r}_l^2/(n-l)$$

where m denotes the number of residual autocorrelations computed. (Advice on the choice of m is given in Section 8.2.) Under the hypothesis of model adequacy, $Q_{(m)}$ has an asymptotic χ^2 distribution on $m - p - q - P - Q$ degrees of freedom. Let $\hat{r}^T = (\hat{r}_1,\hat{r}_2,\ldots,\hat{r}_m)$ then the variance-covariance matrix of $\hat{r}$ is given by:

$$Var(\hat{r}) = [I_m - X(X^TX)^{-1}X^T]/n.$$

The construction of the matrix X is discussed in McLeod [2]. (Note that the mean, μ, and the residual variance, σ^2, play no part in calculating $Var(\hat{r})$ and therefore are not required as input to G13ASF.)

Note: for additive models with fixed parameter values (i.e. fitted by G13DCF) G13DSF should be used instead of G13ASF.

4. References

[1] BOX, G. E. P. and LJUNG, G. M.
On a Measure of Lack of Fit in Time Series Models.
Biometrika, 65, pp. 297-303, 1978.

[2] McLEOD, A. I.
On the Distribution of the Residual Autocorrelations in Box-Jenkins Models.
J. Roy. Statist. Soc. Ser. B, 40, pp. 296-302, 1978.

5. Parameters

1: N – INTEGER. *Input*

On entry: the number of observations in the residual series, n.

If G13ASF is used following a call to G13AEF then N must be the value ICOUNT(2) returned by G13AEF.

If G13ASF is used following a call to G13AFF then N must be the value NRES returned by G13AFF.

Constraint: N $\geq$ 3.

2: V(N) – ***real*** array. *Input*

On entry: V(t) must contain an estimate of ε_t, for $t = 1,2,\ldots,n$.

If G13ASF is used following a call to G13AEF then the actual argument V must be EXR(ICOUNT(1)+1) as returned by G13AEF.

If G13ASF is used following a call to G13AFF then the actual argument V must be RES as returned by G13AFF.

Constraint: V must contain at least two distinct elements.

3: MR(7) – INTEGER array. *Input*

On entry: the orders vector (p, d, q, P, D, Q, s) as supplied to G13AEF or G13AFF.

Constraints: $p, q, P, Q, S \geq 0$,
$p + q + P + Q > 0$,
if $s = 0$, then $P = 0$ and $Q = 0$.

4: M – INTEGER. *Input*

On entry: the value of m, the number of residual autocorrelations to be computed. See Section 8.2 for advice on the value of M.

Constraint: NPAR < M < N.

5: PAR(NPAR) – ***real*** array. *Input*

On entry: the parameter estimates in the order $\phi_1, \phi_2, ..., \phi_p$, $\theta_1, \theta_2, ..., \theta_q$, $\Phi_1, \Phi_2, ..., \Phi_P$, $\Theta_1, \Theta_2, ..., \Theta_Q$.

Constraint: the elements in PAR must satisfy the stationarity and invertibility conditions.

6: NPAR – INTEGER. *Input*

On entry: the total number of ϕ, θ, Φ and Θ parameters, i.e. NPAR $= p + q + P + Q$.

Constraint: NPAR = MR(1) + MR(3) + MR(4) + MR(6).

7: ISHOW – INTEGER. *Input*

On entry: ISHOW must be non-zero if the residual autocorrelations, their standard errors and the portmanteau statistics are to be printed and zero otherwise.

These quantities are available also as output variables in R, RCM, CHI, IDF and SIGLEV.

8: R(M) – ***real*** array. *Output*

On exit: an estimate of the residual autocorrelation coefficient at lag l, for $l = 1,2,...,m$. If IFAIL = 3 on exit then all elements of R are set to zero.

9: RCM(IRCM,M) – ***real*** array. *Output*

On exit: the estimated standard errors and correlations of the elements in the array R. The correlation between R(i) and R(j) is returned as RCM(i,j) except that if $i = j$ then RCM(i,j) contains the standard error of R(i). If on exit, IFAIL ≥ 5, then all off-diagonal elements of RCM are set to zero and all diagonal elements are set to $1/\sqrt{n}$.

10: IRCM – INTEGER. *Input*

On entry: the first dimension of the array RCM as declared in the (sub)program from which G13ASF is called.

Constraint: IRCM ≥ M.

11: CHI – ***real***. *Output*

On exit: the value of the portmanteau statistic, $Q_{(m)}$. If IFAIL = 3 on exit then CHI is returned as zero.

12: IDF – INTEGER. *Output*

On exit: the number of degrees of freedom of CHI.

13: SIGLEV – ***real***. *Output*

On exit: the significance level of CHI based on IDF degrees of freedom. If IFAIL = 3 on exit then SIGLEV is returned as one.

14: IW(LIW) – INTEGER array. *Workspace*

15: LIW – INTEGER. *Input*

On entry: the dimension of the array IW as declared in the (sub)program from which G13ASF is called.

Constraint: LIW ≥ max(MR(1), MR(3), MR(4), MR(6)).

16: WORK(LWORK) – ***real*** array. *Workspace*

17: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G13ASF is called.

Constraint: LWORK ≥ NPAR×(M+NPAR+1) + max(MR(1), MR(3), MR(4), MR(6))×max(MR(7),1) + M.

18: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, MR(1) < 0,
or MR(3) < 0,
or MR(4) < 0,
or MR(6) < 0,
or MR(7) < 0,
or MR(7) = 0 and either MR(4) > 0 or MR(6) > 0,
or MR(1) = MR(3) = MR(4) = MR(6) = 0,
or M ≤ NPAR,
or M ≥ N,
or NPAR ≠ MR(1) + MR(3) + MR(4) + MR(6),
or IRCM < M,
or LIW is too small,
or LWORK is too small.

IFAIL = 2

On entry, the autoregressive (or moving average) parameters are extremely close to or outside the stationarity (or invertibility) region. To proceed, the user must supply different parameter estimates in the array PAR.

IFAIL = 3

On entry, the residuals are practically identical giving zero (or near zero) variance. In this case CHI is set to zero, SIGLEV to one and all the elements of R set to zero.

IFAIL = 4

This is an unlikely exit brought about by an excessive number of iterations being needed to evaluate the zeros of the AR or MA polynomials. All output parameters are undefined.

IFAIL = 5

On entry, one or more of the AR operators has a factor in common with one or more of the MA operators. To proceed, this common factor must be deleted from the model. In this case, the off-diagonal elements of RCM are returned as zero and the diagonal elements set to $1/\sqrt{n}$. All other output quantities will be correct.

IFAIL = 6

This is an unlikely exit. At least one of the diagonal elements of RCM was found to be either negative or zero. In this case all off-diagonal elements of RCM are returned as zero and all diagonal elements of RCM set to $1/\sqrt{n}$.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

8.1. Timing

The time taken by the routine depends upon the number of residual autocorrelations to be computed, m.

8.2. Choice of m

The number of residual autocorrelations to be computed, m should be chosen to ensure that when the ARMA model (1) is written as either an infinite order autoregressive process:

$$\text{i.e.}\quad W_t - \mu = \sum_{j=1}^{\infty} \pi_j (W_{t-j} - \mu) + \varepsilon_t$$

or as an infinite order moving average process:

$$\text{i.e.}\quad W_t - \mu = \sum_{j=1}^{\infty} \psi_j \varepsilon_{t-j} + \varepsilon_t$$

then the two sequences $\{\pi_1, \pi_2, ...\}$ and $\{\psi_1, \psi_2, ...\}$ are such that π_j and ψ_j are approximately zero for $j > m$. An over-estimate of m is therefore preferable to an under-estimate of m. In many instances the choice $m = 10$ will suffice. In practice, to be on the safe side, the user should try setting $m = 20$.

8.3. Approximate Standard Errors

When IFAIL is returned as 5 or 6 all the standard errors in RCM are set to $1/\sqrt{n}$. This is the asymptotic standard error of $\hat{r}_l$ when all the autoregressive and moving average parameters are assumed to be known rather than estimated.

8.4. Alternative Applications

G13ASF may be used for diagnostic checking of suitable univariate ARMA model, as described in Section 3, fitted by G13BEF or G13DCF. Great care must be taken in obtaining the input values for G13ASF from the output values from G13BEF or G13DCF.

9. Example

A program to fit an ARIMA(1,1,2) model to a series of 30 observations. 10 residual autocorrelations are computed.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13ASF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NPARMX, NPPCMX, NXMAX, IRES, LIWORK, LWORK, MMAX,
     +                 ICM, IRCM
      PARAMETER        (NPARMX=10,NPPCMX=NPARMX+1,NXMAX=50,IRES=550,
     +                 LIWORK=10,LWORK=500,MMAX=10,ICM=NPARMX+1,
     +                 IRCM=MMAX)
*     .. Local Scalars ..
      real             C, CHI, S, SIGLEV
      INTEGER          I, IDF, IFAIL, ISHOW, ITC, KFC, KPIV, M, NDF,
     +                 NIT, NPAR, NPPC, NRES, NST, NX
```

```
*     .. Local Arrays ..
      real              CM(ICM,ICM), PAR(NPARMX), R(MMAX),
     +                  RCM(IRCM,MMAX), RES(IRES), SD(NPPCMX), ST(NXMAX),
     +                  WORK(LWORK), X(NXMAX)
      INTEGER           ISF(4), IWORK(LIWORK), MR(7)
*     .. External Subroutines ..
      EXTERNAL          G13AFF, G13ASF, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13ASF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX
      IF (NX.GT.0 .AND. NX.LE.NXMAX) THEN
         READ (NIN,*) (X(I),I=1,NX)
         READ (NIN,*) (MR(I),I=1,7)
         NPAR = MR(1) + MR(3) + MR(4) + MR(6)
         IF (NPAR.GT.0 .AND. NPAR.LE.NPARMX) THEN
            DO 20 I = 1, NPAR
               PAR(I) = 0.0e0
   20       CONTINUE
            KFC = 1
            C = 0.0e0
            NPPC = NPAR + KFC
*           * Set KPIV to 1 to obtain monitoring information *
            KPIV = 0
            NIT = 50
            IFAIL = 1
*
            CALL G13AFF(MR,PAR,NPAR,C,KFC,X,NX,S,NDF,SD,NPPC,CM,ICM,ST,
     +                  NST,KPIV,NIT,ITC,ISF,RES,IRES,NRES,IFAIL)
*
            IF (IFAIL.NE.0) WRITE (NOUT,99999) 'G13AFF fails. IFAIL =',
     +          IFAIL
*
            IF (IFAIL.EQ.0 .OR. IFAIL.EQ.9) THEN
               CALL X04ABF(1,NOUT)
               M = 10
               ISHOW = 1
               IFAIL = -1
*
               CALL G13ASF(NRES,RES,MR,M,PAR,NPAR,ISHOW,R,RCM,IRCM,CHI,
     +                     IDF,SIGLEV,IWORK,LIWORK,WORK,LWORK,IFAIL)
*
               IF (IFAIL.NE.0) WRITE (NOUT,99999)
     +             'G13ASF fails. IFAIL =', IFAIL
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I2)
      END
```

9.2. Program Data

```
G13ASF Example Program Data
30                      : NX, length of the time series
 -217 -177 -166 -136 -110  -95  -64  -37
  -14  -25  -51  -62  -73  -88 -113 -120
  -83  -33  -19   21   17   44   44   78
   88  122  126  114   85   64   : End of time series
1  1  2  0  0  0  0  : MR, orders vector of the model
```

9.3. Program Results

```
G13ASF Example Program Results

RESIDUAL AUTOCORRELATION FUNCTION
---------------------------------

LAG  K        1      2      3      4      5      6      7
R(K)      0.020 -0.040 -0.019  0.068 -0.143 -0.046 -0.205
ST.ERROR  0.007  0.125  0.128  0.150  0.168  0.168  0.178
----------------------------------------------------------
LAG  K        8      9     10
R(K)     -0.108 -0.001 -0.058
ST.ERROR  0.179  0.181  0.183
----------------------------------------------------------

BOX - LJUNG PORTMANTEAU STATISTIC =       3.465
               SIGNIFICANCE LEVEL =       0.839
(BASED ON   7 DEGREES OF FREEDOM)

VALUE OF IFAIL PARAMETER ON EXIT FROM G13ASF =   0
```

G13AUF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13AUF calculates the range (or standard deviation) and the mean for groups of successive time series values. It is intended for use in the construction of range-mean plots.

2. Specification

```
SUBROUTINE G13AUF (N, Z, M, NGRPS, RS, Y, MEAN, IFAIL)
INTEGER        N, M, NGRPS, IFAIL
real           Z(N), Y(NGRPS), MEAN(NGRPS)
CHARACTER*1    RS
```

3. Description

Let $Z_1,Z_2,\ldots,Z_n$ denote n successive observations in a time series. The series may be divided into groups of m successive values and for each group the range or standard deviation (depending on a user-supplied option) and the mean are calculated. If n is not a multiple of m then groups of equal size m are found starting from the end of the series of observations provided and any remaining observations at the start of the series are ignored. The number of groups used, k, is the integer part of n/m. If the user wishes to ensure that no observations are ignored then the number of observations, n, should be chosen so that n is divisible by m.

The mean, M_i, the range, R_i, and the standard deviation, S_i, for the ith group are defined as

$$M_i = \frac{1}{m}\sum_{j=1}^{m} Z_{l+m(i-1)+j}$$

$$R_i = \max_{1\le j\le m}\{Z_{l+m(i-1)+j}\} - \min_{1\le j\le m}\{Z_{l+m(i-1)+j}\}$$

and

$$S_i = \sqrt{\left(\frac{1}{m-1}\right)\sum_{j=1}^{m}(Z_{l+m(i-1)+j}-M_i)^2}$$

where $l = n - km$, the number of observations ignored.

For seasonal data it is recommended that m should be equal to the seasonal period. For nonseasonal data the recommended group size is 8.

A plot of range against mean or of standard deviation against mean is useful for finding a transformation of the series which makes the variance constant. If the plot appears random or the range (or standard deviation) seems to be constant irrespective of the mean level then this suggests that no transformation of the time series is called for. On the other hand an approximate linear relationship between range (or standard deviation) and mean would indicate that a log transformation is appropriate. Further details may be found in either Jenkins [1] or McLeod [2].

The user has the choice of whether to use the range or the standard deviation as a measure of variability. If the group size is small they are both equally good but if the group size is fairly large (e.g. m = 12 for monthly data) then the range may not be as good an estimate of variability as the standard deviation.

4. References

[1] JENKINS, G.M.
Practical experiences with modelling and forecasting time series.
GJP Publications: Lancaster, 1979.

[2] McLEOD, G.
Box-Jenkins in practice.
Volume 1: Univariate stochastic and single output transfer function / Noise analysis.
GJP Publications: Lancaster, 1982.

5. Parameters

1: N – INTEGER. *Input*

On entry: the number of observations in the time series, n.

Constraint: N $\geq$ M.

2: Z(N) – ***real*** array. *Input*

On entry: Z(t) must contain the tth observation Z_t, for $t = 1,2,...,n$.

3: M – INTEGER. *Input*

On entry: the group size, m.

Constraint: M $\geq$ 2.

4: NGRPS – INTEGER. *Input*

On entry: set equal to the number of groups, k.

Constraint: NGRPS = INT(N/M).

5: RS – CHARACTER*1. *Input*

On entry: indicates whether ranges or standard deviations are to be calculated.

If RS = 'R', then ranges are calculated.
If RS = 'S', then standard deviations are calculated.

Constraint: RS = 'R' or 'S'.

6: Y(NGRPS) – ***real*** array. *Output*

On exit: Y(i) contains the range or standard deviation, as determined by RS, of the ith group of observations, for $i = 1,2,...,k$.

7: MEAN(NGRPS) – ***real*** array. *Output*

On exit: MEAN(i) contains the mean of the ith group of observations for $i = 1,2,...,k$.

8: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, N < M,
or M < 2,
or NGRPS $\neq$ integer part of N/M.

IFAIL = 2

On entry, RS is not equal to 'R' or 'S'.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to n.

If the user wishes to obtain a plot of the group ranges or standard deviations against the group means then G01AGF may be used. The plot is output to the unit defined by X04ABF. The user should note that G01AGF sorts the data to be plotted on the y axis (in this case the ranges or standard deviations). If required the user may use M01EAF to re-arrange the data into their original order.

9. Example

The following program produces a range-mean plot for a series of 100 observations divided into groups of 8.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13AUF Example Program Text
*     Mark 14 Release.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, KMAX
      PARAMETER        (NMAX=100,KMAX=NMAX/2)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, K, M, N, NSTEPX, NSTEPY
*     .. Local Arrays ..
      real             MEAN(KMAX), RANGE(KMAX), Z(NMAX)
      INTEGER          ISORT(KMAX)
*     .. External Subroutines ..
      EXTERNAL         G01AGF, G13AUF, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13AUF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      CALL X04ABF(1,NOUT)
      READ (NIN,*) N, M
      IF (N.GE.M .AND. N.LE.NMAX .AND. M.GE.1) THEN
         READ (NIN,*) (Z(I),I=1,N)
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     '                                  Range-mean plot'
         WRITE (NOUT,*)
         K = N/M
         IFAIL = 0
*
         CALL G13AUF(N,Z,M,K,'RANGE',RANGE,MEAN,IFAIL)
*
*        Produce a scatterplot of range against mean or standard
*        deviation against mean.
         NSTEPX = 60
         NSTEPY = 35
*
         CALL G01AGF(MEAN,RANGE,K,ISORT,NSTEPX,NSTEPY,IFAIL)
*
      END IF
      STOP
      END
```

9.2. Program Data

```
G13AUF Example Program Data
100 8   : N, no. of obs in time series, M, no. of obs in each group
 101  82  66  35  31   6  20  90 154 125
  85  68  38  23  10  24  83 133 131 118
  90  67  60  47  41  21  16   6   4   7
  14  34  45  43  49  42  28  10   5   2
   0   1   3  12  14  35  47  41  30  24
  16   7   4   2   8  13  36  50  62  67
  72  48  29   8  13  57 122 139 103  86
  63  37  26  11  15  40  62  98 124  96
  65  64  54  39  21   7   4  23  53  94
  96  77  59  44  47  30  16   7  37  74 : End of time series
```

9.3. Program Results

```
 G13AUF Example Program Results

                                 Range-mean plot

     +....+....+....+....+....+....+....+....+....+....+....+....+.
160.0+                                                           +
     .                                                           .
     .                                                           .
     .                                               1           .
     .                                                           .
140.0+                                                           +
     .                                                           .
     .                                        1                  .
     .                                                           .
     .                                              1            .
120.0+                                                           +
     .                                                           .
     .                                                           .
     .                                                           .
     .                                                           .
100.0+                                                           +
     .                                                           .
     .                              11                           .
     .                                                           .
     .                            1                    1         .
 80.0+                                                           +
     .                                                           .
     .                                                           .
     .                         1                                 .
     .                   1                                       .
 60.0+                                                           +
     .                                                           .
     .                                                           .
     .                                                           .
     .                   1                                       .
 40.0+                 1                                         +
     .                                                           .
     .                                                           .
     .    1                                                      .
     .                                                           .
 20.0+....+....+....+....+....+....+....+....+....+....+....+....+.
    0.00    15.00     30.00     45.00     60.00     75.00     90.00
         7.50    22.50     37.50     52.50     67.50     82.50
```

G13BAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13BAF filters a time series by an ARIMA model.

2. Specification

```
      SUBROUTINE G13BAF (Y, NY, MR, NMR, PAR, NPAR, CY, WA, NWA, B, NB,
     1                   IFAIL)
      INTEGER            NY, MR(NMR), NMR, NPAR, NWA, NB, IFAIL
      real               Y(NY), PAR(NPAR), CY, WA(NWA), B(NB)
```

3. Description

From a given series $y_1,y_2,...,y_n$, a new series $b_1,b_2,...,b_n$ is calculated using a supplied (filtering) ARIMA model. This model will be one which has previously been fitted to a series x_t with residuals a_t. The equations defining b_t in terms of y_t are very similar to those by which a_t is obtained from x_t. The only dissimilarity is that no constant correction is applied after differencing. This is because the series y_t is generally distinct from the series x_t with which the model is associated, though y_t may be related to x_t. Whilst it is appropriate to apply the ARIMA model to y_t so as to preserve the same relationship between b_t and a_t as exists between y_t and x_t, the constant term in the ARIMA model is inappropriate for y_t. The consequence is that b_t will not necessarily have zero mean.

The equations are precisely:

$$w_t = \nabla^d \nabla_s^D y_t, \tag{1}$$

the appropriate differencing of y_t; both the seasonal and non-seasonal inverted autoregressive operations are then applied,

$$u_t = w_t - \Phi_1 w_{t-s} - ... - \Phi_P w_{t-s\times P} \tag{2}$$

$$v_t = u_t - \phi_1 u_{t-1} - ... - \phi_p u_{t-p} \tag{3}$$

followed by the inverted moving average operations

$$z_t = v_t + \Theta_1 z_{t-s} + ... + \Theta_Q z_{t-s\times Q} \tag{4}$$

$$b_t = z_t + \theta_1 b_{t-1} + ... + \theta_q b_{t-q}. \tag{5}$$

Because the filtered series value b_t depends on present and past values $y_t,y_{t-1},...$ there is a problem arising from ignorance of $y_0,y_{-1},...$ which particularly affects calculation of the early values $b_1,b_2,...$, causing 'transient errors'. The routine allows two possibilities.

(i) The equations (1), (2) and (3) are applied from successively later time points so that all terms on their right-hand sides are known, with v_t being defined for $t = (1+d+s\times D+s\times P),...,n$. Equations (4) and (5) are then applied over the same range, taking any values on the right-hand side associated with previous time points, to be zero.

This procedure may still however result in unacceptably large transient errors in early values of b_t.

(ii) The unknown values $y_0,y_{-1},...$ are estimated by backforecasting. This requires that an ARIMA model distinct from that which has been supplied for filtering, should have been previously fitted to y_t.

For efficiency, the user is asked to supply both this ARIMA model for y_t, and a limited number of backforecasts which are prefixed to the known values of y_t. Within the routine further backforecasts of y_t, and the series w_t, u_t, v_t in (1), (2), (3) are then easily calculated, and a set of linear equations solved for backforecasts of z_t,b_t for use in (4) and (5) in the case that $q + Q > 0$.

Even if the best model for y_t is not available, a very approximate guess such as

$$y_t = c + e_t$$

or

$$\nabla y_t = e_t$$

can help to reduce the transients substantially.

The backforecasts which need to be prefixed to y_t are of length $Q'_y = q_y + s_y \times Q_y$ where q_y and Q_y are the non-seasonal and seasonal moving average orders and s_y the seasonal period, for the ARIMA model of y_t. Thus the user need not carry out the backforecasting exercise if $Q'_y = 0$. Otherwise, the series $y_1, y_2, \ldots, y_n$ should be reversed to obtain $y_n, y_{n-1}, \ldots, y_1$ and G13AJF should be used to forecast Q'_y values, $\hat{y}_0, \ldots, \hat{y}_{1-Q'_y}$. The ARIMA model used is that fitted to y_t (as a forward series) except that if $d_y + D_y$ is odd, the constant should be changed in sign (to allow e.g. for the fact that a forward upward trend is a reversed downward trend). The ARIMA model for y_t supplied to the filtering routine must however have the appropriate constant for the forward series.

The series $\hat{y}_{1-Q'_y}, \ldots, \hat{y}_0, y_1, \ldots, y_n$ is then supplied to the routine, and a corresponding set of values returned for b_t.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

5. Parameters

1: Y(NY) – ***real*** array. *Input*

On entry: the Q'_y back forecasts, starting with backforecast at time $1 - Q'_y$ to backforecast at time 0, followed by the time series starting at time 1, where Q'_y = MR(10) + MR(13)×MR(14). If there are no backforecasts, either because the ARIMA model for the time series is not known, or because it is known but has no moving average terms, then the time series starts at the beginning of Y.

2: NY – INTEGER. *Input*

On entry: the total number of backforecasts and time series data points in array Y.

Constraint: NY $\geq$ max$(1+Q'_y$,NPAR).

3: MR(NMR) – INTEGER array. *Input*

On entry: the orders vector for the filtering model, followed by the orders vector for the ARIMA model for the time series if the latter is known. The orders appear in the standard sequence (p,d,q,P,D,Q,s) as given in the Chapter Introduction. If the ARIMA model for the time series is supplied, then the routine will assume that the first Q'_y values of the array Y are backforecasts.

Constraints: the filtering model is restricted in the following ways:

MR(1) + MR(3) + MR(4) + MR(6) > 0 i.e. filtering by a model which contains only differencing terms is not permitted;
MR(k) $\geq$ 0; $k = 1,2,\ldots,7$;
MR(4) + MR(5) + MR(6) = 0 if and only if MR(7) = 0;
MR(7) $\neq$ 1

the ARIMA model for the time series is restricted in the following ways:

MR(k) $\geq$ 0; $k = 8,\ldots,14$;
MR(11) + MR(12) + MR(13) = 0 if and only if MR(14) = 0;
MR(14) $\neq$ 1

4: NMR – INTEGER. *Input*

On entry: the number of values specified in the array MR. It takes the value 7 if no ARIMA model for the time series is supplied but otherwise it takes the value 14. Thus NMR acts as an indicator as to whether backforecasting can be carried out.

Constraint: NMR = 7 or 14.

5: PAR(NPAR) – ***real*** array. *Input*

On entry: the parameters of the filtering model, followed by the parameters of the ARIMA model for the time series, if supplied. Within each model the parameters are in the standard order of non-seasonal AR and MA followed by seasonal AR and MA.

6: NPAR – INTEGER. *Input*

On entry: the total number of parameters held in array PAR.

Constraints: if NMR = 7,
NPAR = MR(1) + MR(3) + MR(4) + MR(6),
if NMR = 14,
NPAR = MR(1) + MR(3) + MR(4) + MR(6)
+ MR(8) + MR(10) + MR(11) + MR(13).

Note: the first constraint (i.e. MR(1) + MR(3) + MR(4) + MR(6) > 0) on the orders of the filtering model, in parameter MR, ensures that NPAR > 0.

7: CY – ***real.*** *Input*

On entry: if the ARIMA model is known (i.e. NMR = 14), CY must specify the constant term of the ARIMA model for the time series. If this model is not known (i.e. NMR = 7) then CY is not used.

8: WA(NWA) – ***real*** array. *Workspace*
9: NWA – INTEGER. *Input*

On entry: the length of the array WA as declared in the (sub)program from which G13BAF is called. Workspace is only required if the ARIMA model for the time series is known.

Constraints: let K = MR(3) + MR(6)×MR(7) + MR(8) + MR(9) +
(MR(11) + MR(12))×MR(14), then
if NMR = 14, NWA ≥ K×(K+2)
if NMR = 7, NWA ≥ 1.

10: B(NB) – ***real*** array. *Output*

On exit: the filtered output series. If the ARIMA model for the time series was known, and hence Q_y' backforecasts were supplied in Y, then B contains Q_y' 'filtered' backforecasts followed by the filtered series. Otherwise, the filtered series begins at the start of B just as the original series began at the start of Y. In either case, if the value of the series at time t is held in Y(t), then the filtered value at time t is held in B(t).

11: NB – INTEGER. *Input*

On entry: the length of the array B as declared in the (sub)program from which G13BAF is called. In addition to holding the returned filtered series, B is also used as an intermediate work array if the ARIMA model for the time series was known.

Constraints: let K_1 = MR(1) + MR(4)×MR(7),
K_2 = MR(2) + MR(5)×MR(7),
K_3 = MR(3) + MR(6)×MR(7), then
if NMR = 14, NB ≥ NY + max(K_3,K_1+K_2),
if NMR = 7, NB ≥ NY.

12: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NMR $\neq$ 7 and NMR $\neq$ 14.

IFAIL = 2

On entry, the orders vector MR does not satisfy the constraints given in Section 5.

IFAIL = 3

On entry, NPAR is inconsistent with the contents of MR (see Section 5).

IFAIL = 4

On entry, NY is too small to carry out successfully the requested filtering, (see Section 5).

IFAIL = 5

On entry, the work array WA is too small.

IFAIL = 6

On entry, the array B is too small.

IFAIL = 7

The orders vector for the filtering model is invalid.

IFAIL = 8

The orders vector for the ARIMA model is invalid. (Only occurs if NMR = 14.)

IFAIL = 9

The initial values of the filtered series are indeterminate for the given models.

7. Accuracy

Accuracy and stability are high except when the MA parameters are close to the invertibility boundary.

8. Further Comments

The time taken by the routine is approximately proportional to

NY×(MR(1) + MR(3) + MR(4) + MR(6)),

with an appreciable fixed increase if an ARIMA model is supplied for the time series.

9. Example

The example program reads a time series of length 296. It reads the univariate ARIMA (4,0,2,0,0,0,0) model and the ARIMA filtering (3,0,0,0,0,0,0) model for the series. Two initial backforecasts are required and these are calculated by a call to G13AJF. The backforecasts are inserted at the start of the series and G13BAF is called to perform the calculations.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13BAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXMAX, NPMAX, ISTMAX, IFVMAX, IW, IQXDMX, NYMAX,
     +                 NBMAX
      PARAMETER        (NXMAX=300,NPMAX=10,ISTMAX=20,IFVMAX=2,IW=2200,
     +                 IQXDMX=5,NYMAX=NXMAX+IQXDMX,NBMAX=320)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             A1, A2, CX, CY, RMS
      INTEGER          I, IDD, IDW, IFAIL, II, IJ, IQXD, J, K, N, NB,
     +                 NI, NMR, NPAR, NPARX, NST, NWA, NX, NY
*     .. Local Arrays ..
      real             B(NBMAX), FSD(IFVMAX), FVA(IFVMAX), PAR(NPMAX),
     +                 PARX(NPMAX), ST(ISTMAX), WA(IW), X(NXMAX),
     +                 Y(NYMAX)
      INTEGER          ISF(4), MR(14), MRX(7)
*     .. External Subroutines ..
      EXTERNAL         G13AJF, G13BAF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX, MIN, MOD
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX
      IF (NX.GT.0 .AND. NX.LE.NXMAX) THEN
         READ (NIN,*) (X(I),I=1,NX)
*        Read univariate ARIMA for series
         READ (NIN,*) (MRX(I),I=1,7)
         READ (NIN,*) CX
         NPARX = MRX(1) + MRX(3) + MRX(4) + MRX(6)
         IF (NPARX.GT.0 .AND. NPARX.LE.NPMAX) THEN
            READ (NIN,*) (PARX(I),I=1,NPARX)
*           Read model by which to filter series
            READ (NIN,*) (MR(I),I=1,7)
            NPAR = MR(1) + MR(3) + MR(4) + MR(6)
            IF (NPAR.GT.0 .AND. NPAR+NPARX.LE.NPMAX) THEN
               READ (NIN,*) (PAR(I),I=1,NPAR)
*              Initially backforecast QY values
*              (1) Reverse series in situ
               N = NX/2
               NI = NX
               DO 20 I = 1, N
                  A1 = X(I)
                  A2 = X(NI)
                  X(I) = A2
                  X(NI) = A1
                  NI = NI - 1
   20          CONTINUE
               IDD = MRX(2) + MRX(5)
*              (2) Possible sign reversal for ARIMA constant
               IF (MOD(IDD,2).NE.0) CX = -CX
*              (3) Calculate number of backforecasts required
               IQXD = MRX(3) + MRX(6)*MRX(7)
               IF (IQXD.NE.0) THEN
*                 (4) Set up parameter list for call to forecast
*                 routine G13AJF
                  IDW = IW
                  IFAIL = 0
*
```

```
            CALL G13AJF(MRX,PARX,NPARX,CX,1,X,NX,RMS,ST,ISTMAX,
     +                  NST,IQXD,FVA,FSD,IFVMAX,ISF,WA,IW,IFAIL)
*
            J = IQXD
            DO 40 I = 1, IQXD
               Y(I) = FVA(J)
               J = J - 1
   40       CONTINUE
*           Move series into Y
            J = IQXD + 1
            K = NX
            DO 60 I = 1, NX
               IF (J.GT.NYMAX) STOP
               Y(J) = X(K)
               J = J + 1
               K = K - 1
   60       CONTINUE
         END IF
*        Calculate series length
         NY = NX + IQXD
*        Move ARIMA for series into MR
         DO 80 I = 1, 7
            MR(7+I) = MRX(I)
   80    CONTINUE
*        Move parameters of ARIMA for Y into PAR
         DO 100 I = 1, NPARX
            PAR(NPAR+I) = PARX(I)
  100    CONTINUE
         NPAR = NPAR + NPARX
*        Move constant and reset sign reversal
         CY = CX
         IF (MOD(IDD,2).NE.0) CY = -CY
*        Set parameters for call to filter routine G13BAF
         NMR = 14
         NWA = MR(3) + MR(6)*MR(7) + MR(8) + MR(9) + (MR(11)
     +         +MR(12))*MR(14)
         NWA = NWA*(NWA+2)
         NB = NY + MAX(MR(3)+MR(6)*MR(7),MR(1)+MR(2)+(MR(4)+MR(5))
     +        *MR(7))
         IF (NWA.LE.IW .AND. NB.LE.NBMAX) THEN
            IFAIL = 0
*           Filter series by call to G13BAF
            CALL G13BAF(Y,NY,MR,NMR,PAR,NPAR,CY,WA,NWA,B,NB,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +        '                    Original       Filtered'
            WRITE (NOUT,*)
     +        'Backforecasts    y-series         series'
            IF (IQXD.NE.0) THEN
               IJ = -IQXD
               DO 120 I = 1, IQXD
                  WRITE (NOUT,99999) IJ, Y(I), B(I)
                  IJ = IJ + 1
  120          CONTINUE
            END IF
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     + '     Filtered        Filtered        Filtered        Filtered'
            WRITE (NOUT,*)
     + '      series          series          series          series'
            DO 140 I = IQXD + 1, NY, 4
               WRITE (NOUT,99998) (II-IQXD,B(II),II=I,MIN(NY,I+3))
  140             CONTINUE
               END IF
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,I8,F17.4,F15.4)
99998 FORMAT (1X,I5,F9.4,I7,F9.4,I7,F9.4,I7,F9.4)
      END
```

9.2. Program Data

```
G13BAF Example Program Data
  296
 53.8 53.6 53.5 53.5 53.4 53.1 52.7 52.4 52.2 52.0 52.0 52.4 53.0 54.0 54.9 56.0
 56.8 56.8 56.4 55.7 55.0 54.3 53.2 52.3 51.6 51.2 50.8 50.5 50.0 49.2 48.4 47.9
 47.6 47.5 47.5 47.6 48.1 49.0 50.0 51.1 51.8 51.9 51.7 51.2 50.0 48.3 47.0 45.8
 45.6 46.0 46.9 47.8 48.2 48.3 47.9 47.2 47.2 48.1 49.4 50.6 51.5 51.6 51.2 50.5
 50.1 49.8 49.6 49.4 49.3 49.2 49.3 49.7 50.3 51.3 52.8 54.4 56.0 56.9 57.5 57.3
 56.6 56.0 55.4 55.4 56.4 57.2 58.0 58.4 58.4 58.1 57.7 57.0 56.0 54.7 53.2 52.1
 51.6 51.0 50.5 50.4 51.0 51.8 52.4 53.0 53.4 53.6 53.7 53.8 53.8 53.8 53.3 53.0
 52.9 53.4 54.6 56.4 58.0 59.4 60.2 60.0 59.4 58.4 57.6 56.9 56.4 56.0 55.7 55.3
 55.0 54.4 53.7 52.8 51.6 50.6 49.4 48.8 48.5 48.7 49.2 49.8 50.4 50.7 50.9 50.7
 50.5 50.4 50.2 50.4 51.2 52.3 53.2 53.9 54.1 54.0 53.6 53.2 53.0 52.8 52.3 51.9
 51.6 51.6 51.4 51.2 50.7 50.0 49.4 49.3 49.7 50.6 51.8 53.0 54.0 55.3 55.9 55.9
 54.6 53.5 52.4 52.1 52.3 53.0 53.8 54.6 55.4 55.9 55.9 55.2 54.4 53.7 53.6 53.6
 53.2 52.5 52.0 51.4 51.0 50.9 52.4 53.5 55.6 58.0 59.5 60.0 60.4 60.5 60.2 59.7
 59.0 57.6 56.4 55.2 54.5 54.1 54.1 54.4 55.5 56.2 57.0 57.3 57.4 57.0 56.4 55.9
 55.5 55.3 55.2 55.4 56.0 56.5 57.1 57.3 56.8 55.6 55.0 54.1 54.3 55.3 56.4 57.2
 57.8 58.3 58.6 58.8 58.8 58.6 58.0 57.4 57.0 56.4 56.3 56.4 56.4 56.0 55.2 54.0
 53.0 52.0 51.6 51.6 51.1 50.4 50.0 50.0 52.0 54.0 55.1 54.5 52.8 51.4 50.8 51.2
 52.0 52.8 53.8 54.5 54.9 54.9 54.8 54.4 53.7 53.3 52.8 52.6 52.6 53.0 54.3 56.0
 57.0 58.0 58.6 58.5 58.3 57.8 57.3 57.0
     4    0    2    0    0    0    0
     0.000
     2.420    -2.380     1.160    -0.230     0.310    -0.470
     3    0    0    0    0    0    0
     1.970    -1.370     0.340
```

9.3. Program Results

```
 G13BAF Example Program Results

                  Original         Filtered
 Backforecasts    y-series          series
       -2          49.9807         3.4222
       -1          52.6714         3.0809

        Filtered         Filtered         Filtered         Filtered
         series           series           series           series
     1   2.9813      2   2.7803      3   3.7057      4   3.2450
     5   3.0760      6   3.0070      7   3.0610      8   3.1720
     9   3.1170     10   3.0360     11   3.2580     12   3.4520
    13   3.3320     14   3.6980     15   3.3140     16   3.8070
    17   3.3330     18   2.9580     19   3.2800     20   3.0960
    21   3.2270     22   3.0830     23   2.6410     24   3.1870
    25   2.9910     26   3.1110     27   2.8460     28   3.0240
    29   2.7030     30   2.6130     31   2.8060     32   2.9560
    33   2.8170     34   2.8950     35   2.8510     36   2.9160
    37   3.2530     38   3.3050     39   3.1830     40   3.3760
    41   2.9730     42   2.8610     43   3.0490     44   2.8420
    45   2.3190     46   2.3660     47   2.9410     48   2.3810
    49   3.3420     50   2.9340     51   3.1800     52   2.9230
    53   2.6470     54   2.8860     55   2.5310     56   2.6200
    57   3.4170     58   3.4940     59   3.2590     60   3.1310
    61   3.1420     62   2.6710     63   2.8990     64   2.8180
    65   3.2150     66   2.8800     67   2.9610     68   2.8800
    69   3.0020     70   2.8930     71   3.1210     72   3.2210
    73   3.2040     74   3.5360     75   3.7520     76   3.5630
    77   3.7260     78   3.1560     79   3.6310     80   2.9380
    81   3.1480     82   3.4490     83   3.1400     84   3.7380
    85   4.1200    86   3.1540    87   3.7480    88   3.3280
    89   3.3640    90   3.3400    91   3.3950    92   3.0720
    93   3.0050    94   2.8520    95   2.7810    96   3.1950
    97   3.2490    98   2.6370    99   3.0080   100   3.2410
   101   3.5570   102   3.2080   103   3.0880   104   3.3980
   105   3.1660   106   3.1960   107   3.2460   108   3.2870
   109   3.1590   110   3.2620   111   2.7280   112   3.4130
   113   3.2190   114   3.6750   115   3.8550   116   4.0100
   117   3.5380   118   3.8440   119   3.4660   120   3.0640
   121   3.4780   122   3.1140   123   3.5300   124   3.2400
   125   3.3630   126   3.2610   127   3.3020   128   3.1150
   129   3.3280   130   2.8730   131   3.0800   132   2.8390
   133   2.6570   134   3.0260   135   2.4580   136   3.2600
   137   2.8380   138   3.2150   139   3.1140   140   3.1050
   141   3.1400   142   2.9100   143   3.1370   144   2.7500
   145   3.1160   146   3.0680   147   2.8590   148   3.3840
   149   3.5500   150   3.4160   151   3.1770   152   3.3390
   153   3.0190   154   3.1780   155   3.0110   156   3.1940
   157   3.2680   158   3.0500   159   2.8060   160   3.1850
   161   3.0560   162   3.2690   163   2.7940   164   3.0900
   165   2.7100   166   2.7890   167   2.9510   168   3.2440
   169   3.2570   170   3.4360   171   3.4450   172   3.3780
   173   3.3520   174   3.9180   175   2.9190   176   3.1780
   177   2.2580   178   3.5150   179   2.8010   180   3.6030
   181   3.2610   182   3.5300   183   3.3270   184   3.4420
   185   3.5240   186   3.2720   187   3.1110   188   2.8240
   189   3.2330   190   3.1500   191   3.5710   192   3.0810
   193   2.7820   194   2.9040   195   3.2350   196   2.7970
   197   3.1320   198   3.1680   199   4.5210   200   2.6650
   201   4.6870   202   3.9470   203   3.2220   204   3.3410
   205   3.9950   206   3.4820   207   3.3630   208   3.4550
   209   3.2950   210   2.6910   211   3.4600   212   2.9440
   213   3.4400   214   3.1830   215   3.4200   216   3.4100
   217   4.0550   218   2.9990   219   3.8250   220   3.1340
   221   3.5010   222   3.0430   223   3.2660   224   3.3660
   225   3.2650   226   3.3720   227   3.2880   228   3.5470
   229   3.6840   230   3.3100   231   3.6790   232   3.1780
   233   2.9360   234   2.7910   235   3.8020   236   2.6100
   237   4.1690   238   3.7460   239   3.4560   240   3.3910
   241   3.5820   242   3.6220   243   3.4870   244   3.5770
   245   3.4240   246   3.3960   247   3.1220   248   3.4300
   249   3.4580   250   3.0280   251   3.7660   252   3.3770
   253   3.2470   254   3.0180   255   2.9720   256   2.8000
   257   3.2040   258   2.8020   259   3.4100   260   3.1680
   261   2.4600   262   2.8810   263   3.1750   264   3.1740
   265   4.8640   266   3.0600   267   2.9600   268   2.2530
   269   2.5620   270   3.3150   271   3.3480   272   3.5900
   273   3.2560   274   3.2320   275   3.6160   276   3.1700
   277   3.2890   278   3.1200   279   3.3300   280   2.9910
   281   2.9420   282   3.4070   283   2.8720   284   3.3470
   285   3.1920   286   3.4880   287   4.0680   288   3.7550
   289   3.0510   290   3.9680   291   3.3900   292   3.1380
   293   3.6170   294   3.1700   295   3.4150   296   3.4830
```

G13BBF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13BBF filters a time series by a transfer function model.

2. Specification

```
      SUBROUTINE G13BBF (Y, NY, MR, NMR, PAR, NPAR, CY, WA, IWA, B, NB,
     1                   IFAIL)
      INTEGER            NY, MR(NMR), NMR, NPAR, IWA, NB, IFAIL
      real               Y(NY), PAR(NPAR), CY, WA(IWA), B(NB)
```

3. Description

From a given series $y_1,y_2,...,y_n$ a new series $b_1,b_2,...,b_n$ is calculated using a supplied (filtering) transfer function model according to the equation

$$b_t = \delta_1 b_{t-1} + \delta_2 b_{t-2} + ... + \delta_p b_{t-p} + \omega_0 y_{t-b} - \omega_1 y_{t-b-1} - ... - \omega_q y_{t-b-q} \qquad (1)$$

As in the use of G13BAF large transient errors may arise in the early values of b_t due to ignorance of y_t for $t < 0$, and two possibilities are allowed:

(i) The equation (1) is applied from $t = 1+b+q,...,n$ so all terms in y_t on the right hand side of (1) are known, the unknown set of values b_t for $t = b+q,...,b+q+1-p$ being taken as zero.

(ii) The unknown values of y_t for $t \le 0$ are estimated by backforecasting exactly as for G13BAF.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition)
Holden-Day, 1976.

5. Parameters

1: Y(NY) – ***real*** array. *Input*

On entry: the Q_y' backforecasts starting with backforecast at time $1 - Q_y'$ to backforecast at time 0 followed by the time series starting at time 1, where Q_y' = MR(6) + MR(9)×MR(10). If there are no backforecasts either because the ARIMA model for the time series is not known or because it is known but has no moving average terms, then the time series starts at the beginning of Y.

2: NY – INTEGER. *Input*

On entry: the total number of backforecasts and time series data points in array Y.

Constraint: NY ≥ max($1+Q_y'$,NPAR).

3: MR(NMR) – INTEGER array. *Input*

On entry: the orders vector for the filtering TF model followed by the orders vector for the ARIMA model for the time series if the latter is known. The TF model orders appear in the standard form (b,q,p) as given in the Chapter Introduction. Note that if the ARIMA model for the time series is supplied then the routine will assume that the first Q_y' values of the array Y are backforecasts.

Constraints: the filtering model is restricted in the following way:

MR(1),MR(2),MR(3) ≥ 0.

the ARIMA model for the time series is restricted in the following ways:
MR(k) ≥ 0; k = 4,...,10;
MR(7) + MR(8) + MR(9) = 0 if and only if MR(10) = 0;
MR(10) ≠ 1.

4: NMR – INTEGER. *Input*

On entry: the number of values supplied in the array MR. It takes the value 3 if no ARIMA model for the time series is supplied but otherwise it takes the value 10. Thus NMR acts as an indicator as to whether backforecasting can be carried out.

Constraint: NMR = 3 or 10.

5: PAR(NPAR) – ***real*** array. *Input*

On entry: the parameters of the filtering TF model followed by the parameters of the ARIMA model for the time series. In the TF model the parameters are in the standard order of MA-like followed by AR-like operator parameters. In the ARIMA model the parameters are in the standard order of non-seasonal AR and MA followed by seasonal AR and MA.

6: NPAR – INTEGER. *Input*

On entry: the total number of parameters held in array PAR.

Constraints: if NMR = 3,
NPAR = MR(2) + MR(3) + 1,
if NMR = 10,
NPAR = MR(2) + MR(3) + 1 + MR(4) + MR(6)
+ MR(7) + MR(9).

7: CY – ***real.*** *Input*

On entry: if the ARIMA model is known (i.e. NMR = 10), CY must specify the constant term of the ARIMA model for the time series. If this model is not known (i.e. NMR = 3) then CY is not used.

8: WA(IWA) – ***real*** array. *Workspace*
9: IWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13BBF is called.

Constraints: let K = MR(3) + MR(4) + MR(5) + (MR(7)+MR(8))×MR(10) then,
if NMR = 3, IWA ≥ MR(1) + NPAR,
if NMR = 10, IWA ≥ MR(1) + NPAR + K×(K+2).

10: B(NB) – ***real*** array. *Output*

On exit: the filtered output series. If the ARIMA model for the time series was known and hence Q_y' backforecasts were supplied in Y then B contains Q_y' 'filtered' backforecasts followed by the filtered series. Otherwise the filtered series begins at the start of B just as the original series began at the start of Y. In either case if the value of the series at time t is held in Y(t) then the filtered value at time t is held in B(t).

11: NB – INTEGER. *Input*

On entry: the dimension of the array B as declared in the (sub)program from which G13BBF is called.

In addition to holding the returned filtered series, B is also used as an intermediate work array if the ARIMA model for the time series is known.

Constraints: if NMR = 3, NB ≥ NY,
if NMR = 10, NB ≥ NY + max(MR(1)+MR(2),MR(3))

12: IFAIL – INTEGER. ***Input/Output***

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NMR $\neq$ 3 and NMR $\neq$ 10,
or MR(i) < 0, i = 1,2,...,NMR,
or NMR = 10 and MR(10) = 1,
or NMR = 10 and MR(10) = 0 and MR(7) + MR(8) + MR(9) $\neq$ 0,
or NMR = 10 and MR(10) $\neq$ 0, and MR(7) + MR(8) + MR(9) = 0,
or NPAR is inconsistent with the contents of MR,
or WA is too small,
or B is too small.

IFAIL = 2

A supplied model has parameter values which have failed the validity test.

IFAIL = 3

The supplied time series is too short to carry out the requested filtering successfully.

IFAIL = 4

This only occurs when an ARIMA model for the time series has been supplied. The matrix which is used to solve for the starting values for MA filtering is singular.

7. Accuracy

Accuracy and stability are high except when the AR-like parameters are close to the invertibility boundary. All calculations are performed in ***basic precision*** except for one inner product type calculation which on machines of low precision is performed in ***additional precision.***

8. Further Comments

The time taken by the routine is roughly proportional to the product of the length of the series and number of parameters in the filtering model with appreciable increase if an ARIMA model is supplied for the time series.

9. Example

The example program reads a time series of length 296. It reads one univariate ARIMA (1,1,0,0,1,1,12) model for the series and the (0,13,12) transfer function filtering model. 12 initial backforecasts are required and these are calculated by a call to G13AJF. The backforecasts are inserted at the start of the series and G13BBF is called to perform the filtering.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13BBF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXMAX, NPMAX, ISTMAX, IFVMAX, IW, IQXDAX, NYMAX,
     +                 NBMAX
      PARAMETER        (NXMAX=200,NPMAX=30,ISTMAX=30,IFVMAX=12,IW=2000,
     +                 IQXDAX=15,NYMAX=NXMAX+IQXDAX,NBMAX=NYMAX+20)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             A1, A2, CX, CY, RMS
      INTEGER          I, IDD, IFAIL, II, IJ, IQXD, IWA, J, K, N, NB,
     +                 NI, NMR, NPAR, NPARX, NST, NX, NY
*     .. Local Arrays ..
      real             B(NXMAX), FSD(IFVMAX), FVA(IFVMAX), PAR(NPMAX),
     +                 PARX(NPMAX), ST(NPMAX), WA(IW), X(NXMAX),
     +                 Y(NYMAX)
      INTEGER          ISF(4), MR(10), MRX(7)
*     .. External Subroutines ..
      EXTERNAL         G13AJF, G13BBF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX, MIN, MOD
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX
      WRITE (NOUT,*)
      IF (NX.GT.0 .AND. NX.LE.NXMAX) THEN
         READ (NIN,*) (X(I),I=1,NX)
*        Read univariate ARIMA for series
         READ (NIN,*) (MRX(I),I=1,7)
         READ (NIN,*) CX
         NPARX = MRX(1) + MRX(3) + MRX(4) + MRX(6)
         IF (NPARX.GT.0 .AND. NPARX.LE.NPMAX) THEN
            READ (NIN,*) (PARX(I),I=1,NPARX)
*           Read model by which to filter series
            READ (NIN,*) (MR(I),I=1,3)
            NPAR = MR(2) + MR(3) + 1
            IF (NPAR.GT.0 .AND. NPAR+NPARX.LE.NPMAX) THEN
               READ (NIN,*) (PAR(I),I=1,NPAR)
*              Initially backforecast QY values
*              (1) Reverse series in situ
               N = NX/2
               NI = NX
               DO 20 I = 1, N
                  A1 = X(I)
                  A2 = X(NI)
                  X(I) = A2
                  X(NI) = A1
                  NI = NI - 1
   20          CONTINUE
               IDD = MRX(2) + MRX(5)
*              (2) Possible sign reversal for ARIMA constant
               IF (MOD(IDD,2).NE.0) CX = -CX
*              (3) Calculate number of backforecasts required
               IQXD = MRX(3) + MRX(6)*MRX(7)
               IFAIL = 0
               IF (IQXD.NE.0) CALL G13AJF(MRX,PARX,NPARX,CX,0,X,NX,RMS,
     +                                   ST,ISTMAX,NST,IQXD,FVA,FSD,
     +                                   IFVMAX,ISF,WA,IW,IFAIL)
```

```
*             Move backforecasts to start of Y array
              J = IQXD
              DO 40 I = 1, IQXD
                 Y(I) = FVA(J)
                 J = J - 1
   40         CONTINUE
*             Move series into Y
              J = IQXD + 1
              K = NX
              DO 60 I = 1, NX
                 IF (J.GT.NYMAX) STOP
                 Y(J) = X(K)
                 J = J + 1
                 K = K - 1
   60         CONTINUE
           END IF
*          Calculate series length
           NY = NX + IQXD
*          Move ARIMA for series into MR
           DO 80 I = 1, 7
              MR(3+I) = MRX(I)
   80      CONTINUE
*          Move parameters of ARIMA for Y into PAR
           DO 100 I = 1, NPARX
              PAR(NPAR+I) = PARX(I)
  100      CONTINUE
           NPAR = NPAR + NPARX
*          Move constant and reset sign reversal
           CY = CX
           IF (MOD(IDD,2).NE.0) CY = -CY
*          Set parameters for call to filter routine G13BBF
           NMR = 10
           IWA = MR(3) + MR(4) + MR(5) + (MR(7)+MR(8))*MR(10)
           IWA = NPAR + IWA*(IWA+2)
           NB = NY + MAX(MR(1)+MR(2),MR(3))
           IF (IWA.LE.IW .AND. NB.LE.NBMAX) THEN
              IFAIL = 0
*
*             Filter series by call to G13BBF
              CALL G13BBF(Y,NY,MR,NMR,PAR,NPAR,CY,WA,IWA,B,NB,IFAIL)
*
              WRITE (NOUT,*)
     +          '                   Original        Filtered'
              WRITE (NOUT,*)
     +          ' Backforecasts     y-series         series'
              IF (IQXD.NE.0) THEN
                 IJ = -IQXD
                 DO 120 I = 1, IQXD
                    WRITE (NOUT,99999) IJ, Y(I), B(I)
                    IJ = IJ + 1
  120            CONTINUE
                 WRITE (NOUT,*)
                 WRITE (NOUT,*)
     +'      Filtered        Filtered        Filtered        Filtered'
                 WRITE (NOUT,*)
     + '       series          series          series          series'
                 DO 140 I = IQXD + 1, NY, 4
                    WRITE (NOUT,99998) (II-IQXD,B(II),II=I,MIN(NY,I+3))
  140            CONTINUE
```

```
                 END IF
              END IF
           END IF
        END IF
        STOP
*
99999 FORMAT (1X,I8,F17.1,F16.1)
99998 FORMAT (1X,I5,F10.1,I6,F10.1,I6,F10.1,I6,F10.1)
      END
```

9.2. Program Data

```
G13BBF Example Program Data
  158
5312. 5402. 4960. 4717. 4383. 3828. 3665. 3718.
3744. 3994. 4150. 4064. 4324. 4256. 3986. 3670.
3292. 2952. 2765. 2813. 2850. 3085. 3256. 3213.
3514. 3386. 3205. 3124. 2804. 2536. 2445. 2649.
2761. 3183. 3456. 3529. 4067. 4079. 4082. 4029.
3887. 3684. 3707. 3923. 4068. 4557. 4975. 5197.
6054. 6471. 6277. 5529. 5059. 4539. 4236. 4305.
4299. 4478. 4561. 4470. 4712. 4512. 4129. 3942.
3572. 3149. 3026. 3141. 3145. 3322. 3384. 3373.
3630. 3555. 3413. 3127. 2966. 2685. 2642. 2789.
2867. 3032. 3125. 3176. 3359. 3265. 3053. 2915.
2690. 2518. 2523. 2737. 3074. 3671. 4355. 4648.
5232. 5349. 5228. 5172. 4932. 4637. 4642. 4930.
5033. 5223. 5482. 5560. 5960. 5929. 5697. 5583.
5316. 5039. 4972. 5169. 5138. 5316. 5409. 5375.
5803. 5736. 5643. 5416. 5059. 4810. 4937. 5166.
5187. 5348. 5483. 5626. 6077. 6033. 5996. 5860.
5499. 5210. 5421. 5609. 5586. 3663. 5829. 6005.
6693. 6792. 6966. 7227. 7089. 6823. 7286. 7621.
7758. 8000. 8393. 8592. 9186. 9175.
    1     1     0     0     1     1    12
  0.000
  0.620  0.820
    0    13    12
 1.0131  0.0806 -0.0150 -0.0150 -0.0150 -0.0150
-0.0150 -0.0150 -0.0150 -0.0150 -0.0150 -0.0150
 0.9981 -0.0956  0.0000  0.0000  0.0000  0.0000
 0.0000  0.0000  0.0000  0.0000  0.0000  0.0000
 0.0000  0.8200
```

9.3. Program Results

```
G13BBF Example Program Results

                   Original        Filtered
 Backforecasts     y-series         series
      -12           5159.0          4549.2
      -11           5165.9          4550.9
      -10           4947.5          4552.8
       -9           4729.8          4554.9
       -8           4424.5          4557.4
       -7           4072.5          4560.7
       -6           3995.5          4565.0
       -5           4142.7          4571.1
       -4           4219.7          4580.0
       -3           4452.1          4593.5
       -2           4758.0          4614.3
       -1           4834.6          4647.1
```

```
          Filtered          Filtered          Filtered          Filtered
           series            series            series            series
     1     4699.2      2     4782.2      3     4552.8      4     4550.4
     5     4525.7      6     4324.8      7     4256.9      8     4169.7
     9     4127.9     10     4154.6     11     4011.3     12     3878.7
    13     3705.1     14     3619.1     15     3603.1     16     3496.1
    17     3422.6     18     3463.5     19     3349.8     20     3262.1
    21     3225.9     22     3218.1     23     3103.6     24     3023.5
    25     2905.9     26     2758.5     27     2828.2     28     2958.4
    29     2926.2     30     3019.8     31     3010.7     32     3082.8
    33     3111.7     34     3286.3     35     3279.3     36     3324.4
    37     3461.7     38     3468.3     39     3709.0     40     3839.6
    41     4004.4     42     4146.3     43     4265.3     44     4344.6
    45     4419.8     46     4647.2     47     4802.6     48     4999.5
    49     5446.0     50     5861.0     51     5855.9     52     5310.7
    53     5202.5     54     5046.6     55     4857.1     56     4812.3
    57     4740.7     58     4631.1     59     4447.5     60     4317.7
    61     4079.8     62     3833.7     63     3667.7     64     3774.8
    65     3709.9     66     3648.5     67     3645.3     68     3619.8
    69     3549.4     70     3439.2     71     3250.3     72     3209.2
    73     3005.2     74     2912.4     75     2994.1     76     2947.9
    77     3103.7     78     3168.1     79     3226.0     80     3224.1
    81     3233.0     82     3119.2     83     2992.5     84     3014.8
    85     2763.7     86     2671.3     87     2664.9     88     2778.2
    89     2823.8     90     2989.0     91     3072.2     92     3132.1
    93     3394.6     94     3717.4     95     4180.5     96     4405.9
    97     4605.2     98     4733.0     99     4830.9    100     5030.8
   101     5079.0    102     5125.0    103     5236.7    104     5392.7
   105     5396.7    106     5300.7    107     5312.1    108     5336.6
   109     5347.9    110     5331.2    111     5322.0    112     5444.8
   113     5468.7    114     5532.9    115     5555.9    116     5603.4
   117     5483.2    118     5406.8    119     5250.5    120     5171.9
   121     5217.4    122     5162.3    123     5296.1    124     5268.2
   125     5204.9    126     5290.7    127     5500.0    128     5552.3
   129     5503.3    130     5419.2    131     5335.6    132     5447.6
   133     5495.1    134     5475.1    135     5643.8    136     5713.1
   137     5655.1    138     5691.9    139     5958.4    140     5959.0
   141     5884.8    142     3714.7    143     5877.8    144     5814.1
   145     6095.6    146     6210.7    147     6560.5    148     7013.9
   149     7174.8    150     7230.8    151     7726.7    152     7880.0
   153     7997.4    154     8428.5    155     8264.1    156     8443.1
   157     8615.4    158     8644.6
```

G13BCF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13BCF calculates cross correlations between two time series.

2. Specification

```
SUBROUTINE G13BCF (X, Y, NXY, NL, S, R0, R, STAT, IFAIL)
INTEGER          NXY, NL, IFAIL
real             X(NXY), Y(NXY), S, R0, R(NL), STAT
```

3. Description

Given two series $x_1,x_2,...,x_n$ and $y_1,y_2,...,y_n$ the routine calculates the cross correlations between x_t and lagged values of y_t:

$$r_{xy}(l) = \frac{\sum_{t=1}^{n-l}(x_t-\bar{x})(y_{t+l}-\bar{y})}{ns_xs_y}, \qquad l = 0,1,...,L$$

where

$$\bar{x} = \frac{\sum_{t=1}^{n} x_t}{n}$$

$$s_x^2 = \frac{\sum_{t=1}^{n}(x_t-\bar{x})^2}{n}$$

and similarly for y.

The ratio of standard deviations s_y/s_x is also returned, and a portmanteau statistic is calculated:

$$\mathrm{STAT} = n\sum_{l=1}^{L} r_{xy}(l)^2.$$

Provided n is large, L much less than n, and both x_t,y_t are samples of series whose true autocorrelation functions are zero, then, under the null hypothesis that the true cross correlations between the series are zero, STAT has a χ^2 distribution with L degrees of freedom. Values of STAT in the upper tail of this distribution provide evidence against the null hypothesis.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

5. Parameters

1: X(NXY) – ***real*** array. *Input*

On entry: the n values of the x series.

2: Y(NXY) – ***real*** array. *Input*

On entry: the n values of the y series.

3: NXY – INTEGER. *Input*

On entry: the length of the time series, n.

Constraint: NXY $\geq$ 2.

4: NL – INTEGER. *Input*

On entry: the maximum lag for calculating cross correlations. L.

Constraint: 1 $\leq$ NL $<$ NXY.

5: S – ***real***. *Output*

On exit: the ratio of the standard deviation of the y series to the standard deviation of the x series, s_y/s_x.

6: R0 – ***real***. *Output*

On exit: the cross correlation between the x and y series at lag zero.

7: R(NL) – ***real*** array. *Output*

On exit: the cross correlations between the x and y series at lags 1 to L, $r_{xy}(l)$ for $l = 1,2,...,L$.

8: STAT – ***real***. *Output*

On exit: the statistic for testing for absence of cross correlation.

9: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NXY $\leq$ 1,
or NL $<$ 1,
or NL $\geq$ NXY.

IFAIL = 2

One or both of the x and y series have zero variance and hence cross correlations cannot be calculated.

7. Accuracy

All computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to nL.

9. Example

The example program reads two time series of length 20. It calculates and prints the cross correlations up to lag 15 for the first series leading the second series and then for the second series leading the first series.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13BCF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXYMAX, NLMAX
      PARAMETER        (NXYMAX=20,NLMAX=15)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             R0XY, R0YX, STATXY, STATYX, SXY, SYX
      INTEGER          I, IFAIL, NL, NXY
*     .. Local Arrays ..
      real             RXY(NLMAX), RYX(NLMAX), X(NXYMAX), Y(NXYMAX)
*     .. External Subroutines ..
      EXTERNAL         G13BCF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BCF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*     Read series length and number of lags
      READ (NIN,*) NXY, NL
      IF (NXY.GT.2 .AND. NXY.LE.NXYMAX .AND. NL.GT.0 .AND. NL.LE.NLMAX)
     +    THEN
*        Read series
         READ (NIN,*) (X(I),I=1,NXY)
         READ (NIN,*) (Y(I),I=1,NXY)
*        Call routine to calculate cross correlations between X and Y
         IFAIL = 0
*
         CALL G13BCF(X,Y,NXY,NL,SXY,R0XY,RXY,STATXY,IFAIL)
*
         IFAIL = 0
*
*        Call routine to calculate cross correlations between Y and X
         CALL G13BCF(Y,X,NXY,NL,SYX,R0YX,RYX,STATYX,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +     '                            Between        Between'
         WRITE (NOUT,*)
     +     '                            X and Y        Y and X'
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'Standard deviation ratio', SXY, SYX
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Cross correlation at lag'
         WRITE (NOUT,99999) '                        0', R0XY, R0YX
         WRITE (NOUT,99998) (I,RXY(I),RYX(I),I=1,NL)
         WRITE (NOUT,*)
         WRITE (NOUT,99997) 'Test statistic           ', STATXY, STATYX
      END IF
      STOP
*
99999 FORMAT (1X,A,F10.4,F15.4)
99998 FORMAT (21X,I4,F10.4,F15.4)
99997 FORMAT (1X,A,F10.4,F15.4)
      END
```

9.2. Program Data

```
G13BCF Example Program Data
   20    15
  0.02  0.05  0.08  0.03 -0.05  0.11 -0.01 -0.08 -0.08 -0.11
 -0.18 -0.19 -0.09  0.03  0.10  0.15 -0.14  0.07  0.09  0.16
  3.18  3.21  3.26  3.25  3.08  3.01  3.06  3.17  3.12  3.04
  3.26  3.45  3.33  3.70  3.31  3.81  3.33  2.96  3.28  3.10
```

9.3. Program Results

```
G13BCF Example Program Results

                            Between       Between
                            X and Y       Y and X

Standard deviation ratio     2.0053        0.4987

Cross correlation at lag
                       0     0.0568        0.0568
                       1     0.0438       -0.0151
                       2    -0.3762        0.3955
                       3    -0.4864        0.3417
                       4    -0.6294        0.5486
                       5    -0.3871        0.2291
                       6    -0.1690        0.3190
                       7    -0.0678        0.1980
                       8     0.0962        0.0438
                       9     0.0788       -0.1428
                      10     0.2910       -0.1376
                      11     0.0950       -0.0387
                      12     0.0547       -0.0380
                      13     0.1855       -0.1551
                      14     0.0243       -0.1536
                      15     0.0034       -0.0696

Test statistic              22.1269       17.2917
```

G13BDF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13BDF calculates preliminary estimates of the parameters of a transfer function model.

2. Specification

```
      SUBROUTINE G13BDF (R0, R, NL, NNA, S, NWDS, WA, IWA, WDS, ISF,
     1                   IFAIL)
      INTEGER            NL, NNA(3), NWDS, IWA, ISF(2), IFAIL
      real               R0, R(NL), S, WA(IWA), WDS(NWDS)
```

3. Description

The routine calculates estimates of parameters $\delta_1,\delta_2,...,\delta_p$, $\omega_0,\omega_1,...,\omega_q$ in the transfer function model

$$y_t = \delta_1 y_{t-1} + \delta_2 y_{t-2} + ... + \delta_p y_{t-p} + \omega_0 x_{t-b} - \omega_1 x_{t-b-1} - ... - \omega_q x_{t-b-q}$$

given cross correlations between the series x_t and lagged values of y_t:

$$r_{xy}(l), \quad l = 0,1,...,L$$

and the ratio of standard deviations s_y/s_x, as supplied by G13BCF.

It is assumed that the series x_t used to calculate the cross correlations is a sample from a time series with true autocorrelations of zero. Otherwise the cross correlations between the series b_t and a_t, as defined in the description of G13BAF, should be used in place of those between y_t and x_t.

The estimates are obtained by solving for $\delta_1,\delta_2,...,\delta_p$ the equations

$$r_{xy}(b+q+j) = \delta_1 r_{xy}(b+q+j-1) + ... + \delta_p r_{xy}(b+q+j-p), \qquad j = 1,2,...,p$$

then calculating

$$\omega_i = \pm(s_y/s_x)\{r_{xy}(b+i) - \delta_1 r_{xy}(b+i-1) - ... - \delta_p r_{xy}(b+i-p)\}, \qquad i = 0,1,...,q$$

where the '+' is used for ω_0 and '–' for ω_i, $i > 0$.

Any value of $r_{xy}(l)$ arising in these equations for $l < b$ is taken as zero. The parameters $\delta_1,\delta_2,...,\delta_p$ are checked as to whether they satisfy the stability criterion.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control (Revised Edition).
Holden-Day, 1976.

5. Parameters

1: R0 – ***real***. *Input*

On entry: the cross correlation between the two series at lag 0, $r_{xy}(0)$.

Constraint: $-1.0 \le \text{R0} \le 1.0$

2: R(NL) – ***real*** array. *Input*

On entry: the cross correlations between the two series at lags 1 to L, $r_{xy}(l)$, for $l = 1,2,...,L$.

Constraint: $-1.0 \le \text{R}(i) \le 1.0$, for $i = 1,2,...,\text{NL}$.

3: NL – INTEGER. *Input*

On entry: the number of lagged cross correlations in the array R, L.

Constraint: NL $\geq$ max(NNA(1)+NNA(2)+NNA(3),1)

4: NNA(3) – INTEGER array. *Input*

On entry: the transfer function model orders in the standard form b,q,p (i.e. delay time, number of moving-average MA-like followed by number of autoregressive AR-like parameters).

Constraint: NNA(i) $\geq$ 0, for i = 1,2,3.

5: S – ***real***. *Input*

On entry: the ratio of the standard deviation of the y series to that of the x series, s_y/s_x.

Constraint: S > 0.0.

6: NWDS – INTEGER. *Input*

On entry: the exact number of parameters in the transfer function model.

Constraint: NWDS = NNA(2) + NNA(3) + 1.

7: WA(IWA) – ***real*** array. *Workspace*

8: IWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13BDF is called.

Constraint: IWA $\geq$ NNA(3)$\times$(NNA(3)+1).

9: WDS(NWDS) – ***real*** array. *Output*

On exit: the preliminary estimates of the parameters of the transfer function model in the order of NNA(2)+1 MA-like parameters followed by the NNA(3) AR-like parameters. If the estimation of either type of parameter fails then these parameters are set to 0.0.

10: ISF(2) – INTEGER array. *Output*

On exit: indicators of the success of the estimation of MA-like and AR-like parameters respectively. A value 0 indicates that there are no parameters of that type to be estimated. A value of 1 or −1 indicates that there are parameters of that type in the model and the estimation of that type has been successful or unsuccessful respectively. Note that there is always at least one MA-like parameter in the model.

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NNA(i) < 0, for i = 1,2,3,
or NL < max(NNA(1)+NNA(2)+NNA(3),1),
or R0 < −1.0 or R0 > 1.0,
or R(i) < −1.0 or R(i) > 1.0, for some i = 1,2,...,NL,
or S $\leq$ 0.0,
or NWDS $\neq$ NNA(2) + NNA(3) + 1,
or IWA < NNA(3)$\times$(NNA(3)+1).

7. Accuracy

Equations used in the computations may become unstable, in which case results are reset to zero with array ISF values set accordingly.

8. Further Comments

The time taken by the routine is roughly proportional to NWDS^3.

9. Example

The example program reads the cross correlations between 2 series at lags 0 to 6. It then reads a (3,2,1) transfer function model and calculates and prints the preliminary estimates of the parameters of the model.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13BDF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NLMAX, NWDSMX, IWAMAX
      PARAMETER        (NLMAX=10,NWDSMX=5,IWAMAX=20)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             R0, S
      INTEGER          I, IFAIL, IWA, NL, NWDS
*     .. Local Arrays ..
      real             R(NLMAX), WA(IWAMAX), WDS(NWDSMX)
      INTEGER          ISF(2), NNA(3)
*     .. External Subroutines ..
      EXTERNAL         G13BDF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BDF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NL
      READ (NIN,*) R0
      IF (NL.GT.0 .AND. NL.LE.NLMAX) THEN
         READ (NIN,*) (R(I),I=1,NL)
         READ (NIN,*) (NNA(I),I=1,3)
         READ (NIN,*) S
         NWDS = NNA(2) + NNA(3) + 1
         IWA = NNA(3)*(NNA(3)+1)
         IF (NWDS.LE.NWDSMX .AND. IWA.LE.IWAMAX) THEN
            IFAIL = 0
*
            CALL G13BDF(R0,R,NL,NNA,S,NWDS,WA,IWA,WDS,ISF,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Success/failure indicator', ISF(1),
     +        ISF(2)
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Transfer function model B, Q, P =',
     +        (NNA(I),I=1,3)
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Parameter initial estimates'
            WRITE (NOUT,99998) (WDS(I),I=1,NWDS)
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,3I4)
99998 FORMAT (1X,4F10.4)
      END
```

9.2. Program Data

```
G13BDF Example Program Data
     6
 -0.0155
  0.0339 -0.0374 -0.2895 -0.3430 -0.4518 -0.2787
     3     2     1
  1.9256
```

9.3. Program Results

```
G13BDF Example Program Results

Success/failure indicator   1   1

Transfer function model B, Q, P =   3   2   1

Parameter initial estimates
   -0.5575    0.3166    0.4626    0.6169
```

G13BEF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13BEF fits a multi-input model relating one output series to the input series with a choice of three different estimation criteria – nonlinear least-squares, exact likelihood and marginal likelihood. When no input series are present, G13BEF fits a univariate ARIMA model.

2. Specification

```
      SUBROUTINE G13BEF (MR, NSER, MT, PARA, NPARA, KFC, NXXY, XXY, IXXY,
     1                   KEF, NIT, KZSP, ZSP, ITC, SD, CM, ICM, S, D, NDF,
     2                   KZEF, RES, STTF, ISTTF, NSTTF, WA, IWA, MWA,
     3                   IMWA, KPRIV, IFAIL)
      INTEGER            MR(7), NSER, MT(4,NSER), NPARA, KFC, NXXY, IXXY,
     1                   KEF, NIT, KZSP, ITC, ICM, NDF, KZEF, ISTTF, NSTTF,
     2                   IWA, MWA(IMWA), IMWA, KPRIV, IFAIL
      real               PARA(NPARA), XXY(IXXY,NSER), ZSP(4), SD(NPARA),
     1                   CM(ICM,NPARA), S, D, RES(NXXY), STTF(ISTTF),
     2                   WA(IWA)
```

3. Description

3.1. The Multi-input Model

The output series y_t, for $t = 1,2,...,n$, is assumed to be the sum of (unobserved) components $z_{i,t}$ which are due respectively to the inputs $x_{i,t}$, for $i = 1,2,...,m$.

Thus $y_t = z_{1,t} + ... + z_{m,t} + n_t$ where n_t is the error, or output noise component.

A typical component z_t may be either:

(a) A simple regression component, $z_t = \omega x_t$ (here x_t is called a simple input) or

(b) A transfer function model component which allows for the effect of lagged values of the variable, related to x_t by

$$z_t = \delta_1 z_{t-1} + \delta_2 z_{t-2} + ... + \delta_p z_{t-p} + \omega_0 x_{t-b} - \omega_1 x_{t-b-1} - ... - \omega_q x_{t-b-q}.$$

The noise n_t is assumed to follow a (possibly seasonal) ARIMA model, i.e. may be represented in terms of an uncorrelated series, a_t, by the hierarchy of equations:

(c) $\nabla^d \nabla_s^D n_t = c + w_t$

(d) $w_t = \Phi_1 w_{t-s} + \Phi_2 w_{t-2\times s} + ... + \Phi_P w_{t-P\times s} + e_t - \Theta_1 e_{t-s} - \Theta_2 e_{t-2\times s} - ... - \Theta_Q e_{t-Q\times s}$

(e) $e_t = \phi_1 e_{t-1} + \phi_2 e_{t-2} + ... + \phi_p e_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - ... - \theta_q a_{t-q}$

as outlined in Section 3 of document G13AEF.

Note: the orders p,q appearing in each of the transfer function models and the ARIMA model are not necessarily the same; $\nabla^d \nabla_s^D n_t$ is the result of applying non-seasonal differencing of order d and seasonal differencing of seasonality s and order D to the series n_t, the differenced series is then of length $N = n - d - s\times D$; the constant term parameter c may optionally be held fixed at its initial value (usually, but not necessarily zero) rather than being estimated.

For the purpose of defining an estimation criterion it is assumed that the series a_t is a sequence of independent Normal variates having mean 0 and variance σ_a^2. An allowance has to be made for the effects of unobserved data prior to the observation period. For the noise component an allowance is always made using a form of backforecasting.

For each transfer function input, the user has to decide what values are to be assumed for the pre-period terms $z_0,z_1,...,z_{1-p}$ and $x_0,x_1,...,x_{1-b-q}$ which are in theory necessary to re-create the component series $z_1,z_2,...,z_n$, during the estimation procedure.

The first choice is to assume that all these values are zero. In this case in order to avoid undesirable transient distortion of the early values $z_1, z_2, \ldots,$ the user is advised first to correct the input series x_t by subtracting from all the terms a suitable constant to make the early values $x_1, x_2, \ldots,$ close to zero. The series mean $\bar{x}$ is one possibility, but for a series with strong trend, the constant might be simply x_1.

The second choice is to treat the unknown pre-period terms as nuisance parameters and estimate them along with the other parameters. This choice should be used with caution. For example, if $p = 1$ and $b = q = 0$, it is equivalent to fitting to the data a decaying geometric curve of the form $A\delta^t$, for $t = 1,2,3,\ldots,$ along with the other inputs, this being the form of the transient. If the output y_t contains a strong trend of this form, which is not otherwise represented in the model, it will have a tendency to influence the estimate of δ away from the value appropriate to the transfer function model.

In most applications the first choice should be adequate, with the option possibly being used as a refinement at the end of the modelling process. The number of nuisance parameters is then $\max(p,b+q)$, with a corresponding loss of degrees of freedom in the residuals. If the user aligns the input x_t with the output by using in its place the shifted series x_{t-b}, then setting $b = 0$ in the transfer function model, there is some improvement in efficiency. On some occasions when the model contains two or more inputs, each with estimation of pre-period nuisance parameters, these parameters may be co-linear and lead to failure of the routine. The option must then be 'switched off' for one or more inputs.

3.2. The Estimation Criterion

This is a measure of how well a proposed set of parameters in the transfer function and noise ARIMA models, matches the data. The estimation routine searches for parameter values which minimize this criterion. For a proposed set of parameter values it is derived by calculating:

(i) the components $z_{1,t}, z_{2,t}, \ldots, z_{m,t}$ as the responses to the input series $x_{1,t}, x_{2,t}, \ldots, x_{m,t}$ using the equations (a) or (b) above,

(ii) the discrepancy between the output and the sum of these components, as the noise

$$n_t = y_t - (z_{1,t} + z_{2,t} + \ldots + z_{m,t}),$$

(iii) the residual series a_t from n_t by reversing the recursive equations (c), (d) and (e) above.

This last step again requires treatment of the effect of unknown pre-period values of n_t and other terms in the equations regenerating a_t. This is identical to the treatment given in Section 3 of document G13AEF, and leads to a criterion which is a sum of squares function S, of the residuals a_t. It may be shown that the finite algorithm presented there is equivalent to taking the infinite set of past values $n_0, n_{-1}, n_{-2}, \ldots,$ as (linear) nuisance parameters. There is no loss of degrees of freedom however, because the sum of squares function S may be expressed as including the corresponding set of past residuals – see Box and Jenkins [1] page 273, who prove that

$$S = \sum_{-\infty}^{n} a_t^2.$$

The function $D = S$ is the first of the three possible criteria, and is quite adequate for moderate to long series with no seasonal parameters. The second is the exact likelihood criterion which considers the past set n_0, n_{-1}, n_{-2} not as simple nuisance parameters, but as unobserved random variables with known distribution. Calculation of the likelihood of the observed set $n_1, n_2, \ldots, n_n$ requires theoretical integration over the range of the past set. Fortunately this yields a criterion of the form $D = M \times S$ (whose minimization is equivalent to maximizing the exact likelihood of the data), where S is exactly as before, and the multiplier M is a function calculated from the ARIMA model parameters. The value of M is always ≥ 1, and M tends to 1 for any fixed parameter set as the sample size n tends to ∞. There is a moderate computational overhead in using this option, but its use avoids appreciable bias in the ARIMA model parameters and yields a better conditioned estimation problem.

The third criterion of marginal likelihood treats the coefficients of the simple inputs in a manner analogous to that given to the past set $n_0,n_{-1},n_{-2},\ldots$. These coefficients, together with the constant term c used to represent the mean of w_t, are in effect treated as random variables with highly dispersed distributions. This leads to the criterion $D = M{\times}S$ again, but with a different value of M which now depends on the simple input series values x_t. In the presence of a moderate to large number of simple inputs, the marginal likelihood criterion can counteract bias in the ARIMA model parameters which is caused by estimation of the simple inputs. This is particularly important in relatively short series.

G13BEF can be used with no input series present, to estimate a univariate ARIMA model for the ouput alone. The marginal likelihood criterion is then distinct from exact likelihood only if a constant term is being estimated in the model, because this is treated as an implicit simple input.

3.3. The Estimation Procedure

This is the minimization of the estimation criterion or objective function D (for deviance). The routine uses an extension of the algorithm of Marquardt [2]. The step size in the minimization is inversely related to a parameter α, which is increased or decreased by a factor β at successive iterations, depending on the progress of the minimization. Convergence is deemed to have occurred if the fractional reduction of D in successive iterations is less than a value γ, while $\alpha < 1$.

Certain model parameters (in fact all excluding the ω's) are subject to stability constraints which are checked throughout to within a specified tolerance multiple δ of machine accuracy. Using the least-squares criterion, the minimization may halt prematurely when some parameters 'stick' at a constraint boundary. This can happen particularly with short seasonal series (with a small number of whole seasons). It will not happen using the exact likelihood criterion, although convergence to a point on the boundary may sometimes be rather slow, because the criterion function may be very flat in such a region. There is also a smaller risk of a premature halt at a constraint boundary when marginal likelihood is used.

A positive, or zero number of iterations can be specified. In either case, the value D of the objective function at iteration zero is presented at the initial parameter values, except for estimation of any pre-period terms for the input series, backforecasts for the noise series, and the coefficients of any simple inputs, and the constant term (unless this is held fixed).

At any later iteration, the value of D is supplied after re-estimation of the backforecasts to their optimal values, corresponding to the model parameters presented at that iteration. This is not true for any pre-period terms for the input series which, although they are updated from the previous iteration, may not be precisely optimal for the parameter values presented, unless convergence of those parameters has occurred. However, in the case of marginal likelihood being specified, the coefficients of the simple inputs and the constant term are also re-estimated together with the backforecasts at each iteration, to values which are optimal for the other parameter values presented.

3.4. Further Results

The residual variance is taken as $erv = \dfrac{S}{df}$ where $df = N -$ (total number of parameters estimated), is the residual degrees of freedom. The pre-period nuisance parameters for the input series are included in the reduction of df, as is the constant if it is estimated.

The covariance matrix of the vector of model parameter estimates is given by

$$erv{\times}H^{-1}$$

where H is the linearised least-squares matrix taken from the final iteration of the algorithm of Marquardt. From this expression are derived the vector of standard deviations, and the correlation matrix of parameter estimates. These are approximations which are only valid asymptotically, and must be treated with great caution when the parameter estimates are close to their constraint boundaries.

The residual series a_t is available upon completion of the iterations over the range $t = 1{+}d{+}s{+}D,\ldots,n$ corresponding to the differenced noise series w_t.

Because of the algorithm used for backforecasting, these are only true residuals for $t \geq 1 + q + s{\times}Q - p - s{\times}P - d - S{\times}D$, provided this is positive. Estimation of pre-period terms for the inputs will also tend to reduce the magnitude of the early residuals, sometimes severely.

The model component series $z_{1,t},\ldots,z_{m,t}$ and n_t may optionally be returned in place of the supplied series values, in order to assess the effects of the various inputs on the output.

3.5. Forecasting Information

For the purpose of constructing forecasts of the ouput series at future time points $t = n+1,n+2,\ldots$ using G13BHF, it is not necessary to use the whole set of observations y_t and $x_{1,t},x_{2,t},\ldots,x_{m,t}$ for $t = 1,2,\ldots,m$. It is sufficient to retain a limited set of quantities constituting the 'state set' as follows: for each series which appears with lagged subscripts in equations (a), (b), (c), (d) and (e) above, include the values at times $n + 1 - k$ for $k = 1$ up to the maximum lag associated with that series in the equations. Note that (c) implicitly includes past values of n_t and intermediate differences of n_t such as $\nabla^{d-1}\nabla_s^D$.

If later observations of the series become available, it is possible to update the state set (without re-estimating the model) using G13BGF. If time series data is supplied with a previously estimated model, it is possible to construct the state set (and forecasts) using G13BJF.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis. Forecasting and Control (Revised Edition).
Holden-Day, 1976.

[2] MARQUARDT, D.W.
An Algorithm for Least-squares Estimation of Nonlinear Parameters.
J. Soc. Indust. Appl. Math., 11, pp. 431, 1963.

5. Parameters

1: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s) of the ARIMA model for the output noise component.

p, q, P and Q refer respectively to the number of autoregressive (ϕ), moving average (θ), seasonal autoregressive (Φ) and seasonal moving average (Θ) parameters.

d, D and s refer respectively to the order of non-seasonal differencing, the order of seasonal differencing and the seasonal period.

2: NSER – INTEGER. *Input*

On entry: the total number of input and output series. There may be any number of input series (including none), but always one output series.

Constraints: NSER > 1 if there are no parameters in the model (that is $p = q = P = Q = 0$ and KFC = 0),
NSER ≥ 1 otherwise.

3: MT(4,NSER) – INTEGER array. *Input*

On entry: the transfer function model orders b, p and q of each of the input series. The order parameters for input series i are held in column i. Row 1 holds the value b_i, row 2 holds the value q_i and row 3 holds the value p_i. For a simple input, $b_i = q_i = p_i = 0$.

Row 4 holds the value r_i, where $r_i = 1$ for a simple input, $r_i = 2$ for a transfer function input for which no allowance is to be made for pre-observation period effects, and $r_i = 3$ for a transfer function input for which pre-observation period effects will be treated by estimation of appropriate nuisance parameters.

When $r_i = 1$, any non-zero contents of rows 1, 2, and 3 of column i are ignored.

4: PARA(NPARA) – ***real*** array. *Input/Output*

On entry: initial values of the multi-input model parameters. These are in order, firstly the ARIMA model parameters: p values of ϕ parameters, q values of θ parameters, P values of Φ parameters and Q values of Θ parameters. These are followed by initial values of the transfer function model parameters $\omega_0,\omega_1,...,\omega_{q_1}$, $\delta_1,\delta_2,...,\delta_{p_1}$ for the first of any input series and similarly for each subsequent input series. The final component of PARA is the initial value of the constant c, whether it is fixed or is to be estimated.

On exit: the latest values of the estimates of these parameters.

5: NPARA – INTEGER. *Input*

On entry: the exact number of ϕ, θ, Φ, Θ, ω, δ and c parameters.

Constraint: NPARA $= p + q + P + Q +$ NSER $+ \sum(p_i+q_i)$, the summation being over all the input series. (c must be included, whether fixed or estimated.)

6: KFC – INTEGER. *Input*

On entry: KFC must be set to 0 if the constant c is to remain fixed at its initial value, and 1 if it is to be estimated.

Constraint: KFC = 0 or 1.

7: NXXY – INTEGER. *Input*

On entry: the (common) length of the original, undifferenced input and output time series.

8: XXY(IXXY,NSER) – ***real*** array. *Input/Output*

On entry: the columns of XXY must contain the NXXY original, undifferenced values of each of the input series and the output series in that order.

On exit: if KZEF = 0, XXY remains unchanged on exit.

If KZEF $\neq$ 0, the columns of XXY hold the corresponding values of the input component series z_t in place of x_t and the output noise component n_t in place of y_t, in that order.

9: IXXY – INTEGER. *Input*

On entry: the first dimension of the array XXY as declared in the (sub)program from which G13BEF is called.

Constraint: IXXY $\geq$ NXXY.

10: KEF – INTEGER. *Input*

On entry: indicates the likelihood option. KEF = 1 gives least-squares: KEF = 2 gives exact likelihood: KEF = 3 gives marginal likelihood.

Constraint: KEF = 1, 2 or 3.

11: NIT – INTEGER. *Input*

On entry: the maximum required number of iterations. If NIT = 0, no change is made to any of the model parameters in array PARA except that the constant c (if KFC = 1) and any ω relating to simple input series are estimated. (Apart from these, estimates are always derived for the nuisance parameters relating to any backforecasts and any pre-observation period effects for transfer function inputs.)

Constraint: NIT $\geq$ 0.

12: KZSP – INTEGER. *Input*

On entry: KZSP must be set to 1 if the routine is to use the input values of ZSP in the minimization procedure, and to any other value if the default values of ZSP are to be used.

13: ZSP(4) – ***real*** array. *Input/Output*

On entry: if KZSP = 1, then ZSP must contain the four values used to control the strategy of the search procedure. These are, as follows:

ZSP(1), contains α, the value used to constrain the magnitude of the search procedure steps.

ZSP(2), contains β, the multiplier which regulates the value of α.

ZSP(3), contains δ, the value of the stationarity and invertibility test tolerance factor.

ZSP(4), contains γ, the value of the convergence criterion.

If KZSP $\neq$ 1 before entry, default values of ZSP are supplied by the routine. These are 0.01, 10.0, 1000.0 and max(100$\times\varepsilon$, 0.0000001), respectively, where ε is the value returned by X02AJF.

On exit: contains the values, default or otherwise, used by the routine.

Constraints: if KZSP = 1, then
$$\text{ZSP}(1) > 0.0,$$
$$\text{ZSP}(2) > 1.0,$$
$$\text{ZSP}(3) \geq 1.0,$$
$$0 \leq \text{ZSP}(4) < 1.0.$$

14: ITC – INTEGER. *Output*

On exit: the number of iterations carried out.

A value of ITC = −1 on exit indicates that the only estimates obtained up to this point have been for the nuisance parameters relating to backforecasts, unless the marginal likelihood option is used in which case estimates have also been obtained for simple input coefficients ω and for the constant c (if KFC = 1). This value of ITC usually indicates a failure in a consequent step of estimating transfer function input pre-observation period nuisance parameters.

A value of ITC = 0 on exit indicates that estimates have been obtained up to this point for the constant c (if KFC = 1), for simple input coefficients ω and for the nuisance parameters relating to the backforecasts and to transfer function input pre-observation period effects.

15: SD(NPARA) – ***real*** array. *Output*

On exit: the NPARA values of the standard deviations corresponding to each of the parameters in PARA. When the constant is fixed its standard deviation is returned as zero. When the values of PARA are valid, the values of SD are usually also valid. However, if an exit value of IFAIL = 3, 8 or 10 is accompanied by a failure to invert the second derivative matrix (which would normally give IFAIL = 9), then the contents of SD will be indeterminate.

16: CM(ICM,NPARA) – ***real*** array. *Output*

On exit: the first NPARA rows and columns of CM contain the correlation coefficients relating to each pair of parameters in PARA. All coefficients relating to the constant will be zero if the constant is fixed. The contents of CM will be indeterminate under the same conditions as SD.

17: ICM – INTEGER. *Input*

On entry: the first dimension of the array CM as declared in the (sub)program from which G13BEF is called.

Constraint: ICM $\geq$ NPARA.

18: S – ***real***. *Output*

On exit: the residual sum of squares, S, at the latest set of valid parameter estimates.

19: D – ***real***. *Output*

On exit: the objective function, D, at the latest set of valid parameter estimates.

20: NDF – INTEGER. *Output*

On exit: the number of degrees of freedom associated with S.

21: KZEF – INTEGER. *Input*

On entry: KZEF must not be set to 0, if the values of the input component series z_t and the values of the output noise component n_t are to overwrite the contents of XXY on exit, and must be set to 0 if XXY is to remain unchanged.

22: RES(NXXY) – ***real*** array. *Output*

On exit: the values of the residuals relating to the differenced values of the output series. The remainder of the first NXXY terms in the array will be zero.

23: STTF(ISTTF) – ***real*** array. *Output*

On exit: the NSTTF values of the state set array.

24: ISTTF – INTEGER. *Input*

On entry: the dimension of the array STTF as declared in the (sub)program from which G13BEF is called.

Constraint: ISTTF $\geq (P\times s) + d + (D\times s) + q + \max(p,Q\times s) + ncg$
where $ncg = \sum(b_i+q_i+p_i)$ over all input series for which $r_i > 1$.

25: NSTTF – INTEGER. *Output*

On exit: the number of values in the state set array STTF.

26: WA(IWA) – ***real*** array. *Workspace*
27: IWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13BEF is called.

It is not practical to outline a method for deriving the exact minimum permissible value of IWA, but the following gives a reasonably good conservative approximation. (It should be noted that if IWA is too small (but not grossly so) then the exact minimum is returned in MWA(i) and is also printed if KPRIV $\neq$ 0).

Let $q' = q + (Q\times s)$
$d' = d + (D\times s)$

where the orders of the output noise model are p, d, q, P, D, Q, s.

Let there be l input series, where l = NSER – 1.

Let $mx_i = \max(b_i+q_i\ ,p_i)$, if $r_i = 3$ for $i = 1,2...,l$
$mx_i = 0$, if $r_i \neq 3$ for $i = 1,2...,l$

where the transfer function model orders for input i are given by b_i, q_i, p_i, r_i.

Let $qx = \max(q', mx_1, mx_2,..., mx_l)$.

Let ncd = NPARA + KFC + $qx + \sum_{i=1}^{l} mx_i$ and nce = NXXY + $d' + (6qx)$.

Finally, let ncf = NSER, and then increment ncf by 1 every time any of the following conditions is satisfied. (The last six conditions should be applied separately to each input series, so that, for example, if we have two input series and if $p_1 > 0$ and $p_2 > 0$ then ncf is incremented by 2.)

The conditions are:

$$p > 0$$
$$q > 0$$
$$P > 0$$
$$Q \geq 0$$
$$qx > 0$$
$$\text{KFC} > 0$$

$$\left.\begin{array}{r} p > 0 \\ q > 0 \\ P > 0 \\ Q > 0 \end{array}\right\} \text{ and } q > 0 \text{ and KEF} > 1.$$

$$\left.\begin{array}{r} p > 0 \\ q > 0 \\ P > 0 \\ Q > 0 \end{array}\right\} \text{ and KFC} > 0 \text{ and KEF} = 3.$$

$$\left.\begin{array}{r} mx_i > 0 \\ p_i > 0 \\ p > 0 \\ q > 0 \\ P > 0 \\ Q > 0 \end{array}\right\} \text{ and } r_i = 1 \text{ and KEF} = 3 \text{ separately for } i = 1,2,\ldots,l.$$

Then IWA $\geq 2\times(ncd)^2 + (nce)\times(ncf+4)$.

28: MWA(IMWA) – INTEGER array. *Workspace*

29: IMWA – INTEGER. *Input*

On entry: the dimension of the array MWA as declared in the (sub)program from which G13BEF is called.

Constraint: IMWA $\geq$ (16×NSER) + (7×ncd) + (3×NPARA) + (3×KFC) + 27, where the derivation of ncd is shown under IWA above.

If IMWA is too small then the exact minimum needed is returned in IMWA and if KPRIV $\neq$ 0 it is also printed.

30: KPRIV – INTEGER. *Input*

On entry: KPRIV must not be set to 0, if it is required to monitor the course of the optimization or to print out the requisite minimum values of IWA or IMWA in the event of an error of the type IFAIL = 6 or 7. The course of the optimization is monitored by printing out at each iteration the iteration count (ITC), the residual sum of squares (S), the objective function (D) and a description and value for each of the parameters in the PARA array. The descriptions are PHI for ϕ, THETA for θ, SPHI for Φ, STHETA for Θ, OMEGA/SI for ω in a simple input, OMEGA for ω in a transfer function input, DELTA for δ and CONSTANT for c. In addition SERIES 1, SERIES 2, etc, indicate the input series relevant to the OMEGA and DELTA parameters.

KPRIV must be set to 0 if the print-out of the above information is not required.

31: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL $\neq$ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.** To suppress the output of an error message when soft failure occurs, set IFAIL to 1.

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

IFAIL = 1

On entry,	KFC < 0,
or	KFC > 1,
or	IXXY < NXXY,
or	ICM < NPARA,
or	KEF < 1,
or	KEF > 3,
or	NIT < 0,
or	NSER < 1,
or	NSER = 1 and there are no parameters in the model ($p = q = P = Q = 0$ and KFC = 0).

IFAIL = 2

On entry, there is inconsistency between NPARA and KFC on the one hand and the orders arrays MR and MT on the other,
or one of the r_i, stored in MT$(4,i)$, is not equal to 1, 2 or 3.

IFAIL = 3

On entry, or during execution, one or more sets of δ parameters do not satisfy the stationarity or invertibility test conditions.

IFAIL = 4

On entry, when KZSP = 1,	ZSP(1) ≤ 0.0,
or	ZSP(2) ≤ 1.0,
or	ZSP(3) < 1.0,
or	ZSP(4) < 0.0,
or	ZSP(4) ≥ 1.0.

IFAIL = 5

On entry, IWA is too small by a considerable margin. No information is supplied about the requisite minimum size.

IFAIL = 6

On entry, IWA is too small, but the requisite minimum size is returned in MWA(1), this is also printed if KPRIV ≠ 0.

IFAIL = 7

On entry, IMWA is too small, but the requisite minimum size is returned in MWA(1), this is also printed if KPRIV ≠ 0.

IFAIL = 8

This indicates a failure in F04ASF which is used to solve the equations giving the latest estimates of the parameters.

IFAIL = 9

This indicates a failure in the inversion of the second derivative matrix. This is needed in the calculation of the correlation matrix and the standard deviations of the parameter estimates.

IFAIL = 10

On entry, or during execution, one or more sets of the ARIMA (ϕ, θ, Φ or Θ) parameters do not satisfy the stationarity or invertibility test conditions.

IFAIL = 11

On entry, ISTTF is too small. The state set information will not be produced and if KZEF $\neq$ 0 array XXY will remain unchanged. All other parameters will be produced correctly.

IFAIL = 12

The routine has failed to converge after NIT iterations. If steady decreases in the objective function, D, were monitored up to the point where this exit occured, then the exit probably occured because NIT was set too small, so the calculations should be restarted from the final point held in PARA.

IFAIL = 13

On entry, ISTTF is too small (see IFAIL = 11) and NIT iterations were carried out without the convergence conditions being satisfied (see IFAIL = 12).

7. Accuracy

The computation used is believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to NXXY$\times$ITC$\times$NPARA^2.

9. Example

The data in the example relate to 40 observations of an output time series and of a single input time series. The noise series has one autoregressive (φ) and one seasonal moving average (Θ) parameter (both of which are initially set to zero) for which the seasonal period is 4. The input series is defined by orders $b_1 = 1, q_1 = 0, p_1 = 1, r_1 = 3$, so that it has one ω (initially set to 2.0) and one δ (initially set to 0.5), and allows for pre-observation period effects. The constant (initially set to zero) is to be estimated. Default values of ZSP are used. Up to 20 iterations are allowed, and the progress of these is not monitored. Marginal likelihood is the chosen option.

After the full 11 iterations, the following are computed and printed out: the final values of the PARA parameters and their standard errors, the correlation matrix, the residuals for the 36 differenced values, the values of z_t and n_t, the values of the state set and the number of degrees of freedom.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13BEF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NSERMX, NPMAX, NXXYMX, ISTTF, IXXY, ICM, IWA,
     +                 IMWA
      PARAMETER        (NSERMX=2,NPMAX=10,NXXYMX=50,ISTTF=20,
     +                 IXXY=NXXYMX,ICM=NPMAX,IWA=1500,IMWA=200)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             D, S
      INTEGER          I, IFAIL, ITC, J, KEF, KFC, KPRIV, KZEF, KZSP,
     +                 NDF, NDV, NIT, NPARA, NSER, NSTTF, NXXY
*     .. Local Arrays ..
      real             CM(ICM,NPMAX), PARA(NPMAX), RES(NXXYMX),
     +                 SD(NPMAX), STTF(ISTTF), WA(IWA),
     +                 XXY(IXXY,NSERMX), ZSP(4)
      INTEGER          MR(7), MT(4,NSERMX), MWA(IMWA)
*     .. External Subroutines ..
      EXTERNAL         G13BEF, X04ABF
```

```
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BEF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) KZEF, KFC, NXXY, NSER, KEF, NIT, KZSP
      IF (NXXY.GT.0 .AND. NXXY.LE.NXXYMX .AND. NSER.GT.0 .AND. NSER.LE.
     +    NSERMX) THEN
         IF (KZSP.NE.0) READ (NIN,*) ZSP
         CALL X04ABF(1,NOUT)
         READ (NIN,*) (MR(I),I=1,7)
         DO 20 I = 1, 4
            READ (NIN,*) (MT(I,J),J=1,NSER)
   20    CONTINUE
         NPARA = 0
         DO 40 I = 1, NSER
            NPARA = NPARA + MT(2,I) + MT(3,I)
   40    CONTINUE
         NPARA = NPARA + MR(1) + MR(3) + MR(4) + MR(6) + NSER
         IF (NPARA.LE.NPMAX) THEN
            READ (NIN,*) (PARA(I),I=1,NPARA)
            DO 60 I = 1, NXXY
               READ (NIN,*) (XXY(I,J),J=1,NSER)
   60       CONTINUE
*           * Set KPRIV to 1 to obtain monitoring information *
            KPRIV = 0
            IFAIL = 1
*
            CALL G13BEF(MR,NSER,MT,PARA,NPARA,KFC,NXXY,XXY,IXXY,KEF,NIT,
     +                  KZSP,ZSP,ITC,SD,CM,ICM,S,D,NDF,KZEF,RES,STTF,
     +                  ISTTF,NSTTF,WA,IWA,MWA,IMWA,KPRIV,IFAIL)
*
            IF (IFAIL.NE.0) THEN
               WRITE (NOUT,*)
               WRITE (NOUT,99999) 'G13BEF fails. IFAIL =', IFAIL
            END IF
            IF (IFAIL.EQ.0 .OR. IFAIL.EQ.8 .OR. IFAIL.EQ.9 .OR.
     +          IFAIL.EQ.11) THEN
               WRITE (NOUT,*)
               WRITE (NOUT,99999)
     +           'The number of iterations carried out is', ITC
               WRITE (NOUT,*)
               WRITE (NOUT,*)
     +'The final values of the parameters and their standard deviations
     +are'
               WRITE (NOUT,*)
               WRITE (NOUT,*)
     +           '  I           PARA(I)                  SD'
               WRITE (NOUT,*)
               DO 80 I = 1, NPARA
                  WRITE (NOUT,99998) I, PARA(I), SD(I)
   80          CONTINUE
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'The correlation matrix is'
               WRITE (NOUT,*)
               WRITE (NOUT,99997) ((CM(I,J),J=1,NPARA),I=1,NPARA)
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'The residuals and the z and n values are'
               WRITE (NOUT,*)
               WRITE (NOUT,*)
     +           '  I         RES(I)          z(t)           n(t)'
               WRITE (NOUT,*)
               NDV = NXXY - MR(2) - MR(5)*MR(7)
               DO 100 I = 1, NXXY
                  IF (I.LE.NDV) THEN
                     WRITE (NOUT,99996) I, RES(I), (XXY(I,J),J=1,NSER)
                  ELSE
                     WRITE (NOUT,99995) I, (XXY(I,J),J=1,NSER)
                  END IF
  100          CONTINUE
```

```
               IF (MR(2).NE.0 .OR. MR(5).NE.0) THEN
                  WRITE (NOUT,*)
                  WRITE (NOUT,*)
     +      '** Note that the residuals relate to differenced values **'
               END IF
               WRITE (NOUT,*)
               WRITE (NOUT,99994) 'The state set consists of', NSTTF,
     +           ' values'
               WRITE (NOUT,*)
               WRITE (NOUT,99993) (STTF(I),I=1,NSTTF)
               WRITE (NOUT,*)
               WRITE (NOUT,99999) 'The number of degrees of freedom is',
     +           NDF
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I4)
99998 FORMAT (1X,I4,2F20.6)
99997 FORMAT (1X,5F10.4)
99996 FORMAT (1X,I4,3F15.3)
99995 FORMAT (1X,I4,F30.3,F15.3)
99994 FORMAT (1X,A,I4,A)
99993 FORMAT (1X,6F10.4)
      END
```

9.2. Program Data

```
G13BEF Example Program Data
    1    1   40    2    3   20    0
    1    0    0    0    0    1    4
    1    0
    0    0
    1    0
    3    0
0.0        0.0        2.0        0.5        0.0
     8.075     105.0
     7.819     119.0
     7.366     119.0
     8.113     109.0
     7.380     117.0
     7.134     135.0
     7.222     126.0
     7.768     112.0
     7.386     116.0
     6.965     122.0
     6.478     115.0
     8.105     115.0
     8.060     122.0
     7.684     138.0
     7.580     135.0
     7.093     125.0
     6.129     115.0
     6.026     108.0
     6.679     100.0
     7.414      96.0
     7.112     107.0
     7.762     115.0
     7.645     123.0
     8.639     122.0
     7.667     128.0
     8.080     136.0
     6.678     140.0
     6.739     122.0
     5.569     102.0
     5.049     103.0
     5.642      89.0
     6.808      77.0
     6.636      89.0
     8.241      94.0
     7.968     104.0
     8.044     108.0
     7.791     119.0
     7.024     126.0
     6.102     119.0
     6.053     103.0
```

9.3. Program Results

```
G13BEF Example Program Results

The number of iterations carried out is  11

The final values of the parameters and their standard deviations are

   I             PARA(I)                 SD

   1             0.380924             0.166379
   2            -0.257786             0.178178
   3             8.956084             0.948061
   4             0.659641             0.060239
   5           -75.435521            33.505341

The correlation matrix is

   1.0000    -0.1839    -0.1775    -0.0340     0.1394
  -0.1839     1.0000     0.0518     0.2547    -0.2860
  -0.1775     0.0518     1.0000    -0.3070    -0.2926
  -0.0340     0.2547    -0.3070     1.0000    -0.8185
   0.1394    -0.2860    -0.2926    -0.8185     1.0000

The residuals and the z and n values are

   I          RES(I)          z(t)            n(t)

   1           0.397         180.567         -75.567
   2           3.086         191.430         -72.430
   3          -2.818         196.302         -77.302
   4          -9.941         195.460         -86.460
   5          -5.061         201.594         -84.594
   6          14.053         199.076         -64.076
   7           2.624         195.211         -69.211
   8          -5.823         193.450         -81.450
   9          -2.147         197.179         -81.179
  10          -0.216         196.217         -74.217
  11          -2.517         191.812         -76.812
  12           7.916         184.544         -69.544
  13           1.423         194.322         -72.322
  14          11.936         200.369         -62.369
  15           5.117         200.990         -65.990
  16          -5.672         200.468         -75.468
  17          -5.681         195.763         -80.763
  18          -1.637         184.025         -76.025
  19          -1.019         175.360         -75.360
  20          -2.623         175.492         -79.492
  21           3.283         182.162         -75.162
  22           6.896         183.857         -68.857
  23           5.395         190.797         -67.797
  24           0.875         194.327         -72.327
  25          -4.153         205.558         -77.558
  26           6.206         204.261         -68.261
  27           4.208         207.104         -67.104
  28          -2.387         196.423         -74.423
  29         -11.803         189.924         -87.924
  30           6.435         175.158         -72.158
  31           1.342         160.761         -71.761
  32          -4.924         156.575         -79.575
  33           4.799         164.256         -75.256
  34          -0.074         167.783         -73.783
```

```
 35         -6.023          184.483         -80.483
 36         -6.427          193.055         -85.055
 37         -2.527          199.390         -80.390
 38          2.039          201.302         -75.302
 39          0.243          195.695         -76.695
 40         -3.166          183.738         -80.738

The state set consists of   6 values

    6.0530  183.7384   -5.7855   -0.1645    0.1800   -3.0977

The number of degrees of freedom is  34
```

With KPRIV set to 1 in the example program, monitoring information similar to that below is obtained:

```
G13BEF Example Program Results


ITC=   -1            S=   0.645665E+04           D=   0.709718E+04

PHI                     0.000000E+00
STHETA                  0.000000E+00
OMEGA       SERIES  1   0.200000E+01
DELTA       SERIES  1   0.500000E+00
CONSTANT                0.868840E+02


ITC=    0            S=   0.580277E+04           D=   0.637844E+04

PHI                     0.000000E+00
STHETA                  0.000000E+00
OMEGA       SERIES  1   0.200000E+01
DELTA       SERIES  1   0.500000E+00
CONSTANT                0.857327E+02


ITC=    1            S=   0.235466E+04           D=   0.249865E+04

PHI                     0.658915E+00
STHETA                  0.657139E-01
OMEGA       SERIES  1   0.372118E+01
DELTA       SERIES  1   0.523797E+00
CONSTANT                0.573913E+02


ITC=    2            S=   0.192234E+04           D=   0.203237E+04

PHI                     0.641769E+00
STHETA                 -0.236119E+00
OMEGA       SERIES  1   0.452313E+01
DELTA       SERIES  1   0.574282E+00
CONSTANT                0.381486E+02


ITC=    3            S=   0.153080E+04           D=   0.163060E+04

PHI                     0.555080E+00
STHETA                 -0.309733E+00
OMEGA       SERIES  1   0.769730E+01
DELTA       SERIES  1   0.735837E+00
CONSTANT               -0.932220E+02
```

```
ITC=    4           S=   0.123293E+04            D=    0.132412E+04

PHI                        0.369833E+00
STHETA                    -0.214529E+00
OMEGA        SERIES  1     0.911652E+01
DELTA        SERIES  1     0.692374E+00
CONSTANT                  -0.998555E+02


ITC=    5           S=   0.120081E+04            D=    0.128927E+04

PHI                        0.388928E+00
STHETA                    -0.264965E+00
OMEGA        SERIES  1     0.890675E+01
DELTA        SERIES  1     0.665990E+00
CONSTANT                  -0.778251E+02


ITC=    6           S=   0.119792E+04            D=    0.128673E+04

PHI                        0.375273E+00
STHETA                    -0.249996E+00
OMEGA        SERIES  1     0.895717E+01
DELTA        SERIES  1     0.661614E+00
CONSTANT                  -0.765626E+02


ITC=    7           S=   0.119793E+04            D=    0.128662E+04

PHI                        0.380405E+00
STHETA                    -0.259453E+00
OMEGA        SERIES  1     0.895418E+01
DELTA        SERIES  1     0.659901E+00
CONSTANT                  -0.755343E+02


ITC=    8           S=   0.119801E+04            D=    0.128661E+04

PHI                        0.380708E+00
STHETA                    -0.256745E+00
OMEGA        SERIES  1     0.895606E+01
DELTA        SERIES  1     0.659744E+00
CONSTANT                  -0.754919E+02


ITC=    9           S=   0.119799E+04            D=    0.128661E+04

PHI                        0.380877E+00
STHETA                    -0.258056E+00
OMEGA        SERIES  1     0.895598E+01
DELTA        SERIES  1     0.659651E+00
CONSTANT                  -0.754385E+02


ITC=   10           S=   0.119800E+04            D=    0.128661E+04

PHI                        0.380922E+00
STHETA                    -0.257583E+00
OMEGA        SERIES  1     0.895611E+01
DELTA        SERIES  1     0.659648E+00
CONSTANT                  -0.754401E+02
```

```
ITC=   11          S=   0.119800E+04            D=   0.128661E+04

PHI                         0.380924E+00
STHETA                     -0.257786E+00
OMEGA       SERIES  1       0.895608E+01
DELTA       SERIES  1       0.659641E+00
CONSTANT                   -0.754355E+02
```

G13BGF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G13BGF accepts a series of new observations of an output time series and any associated input time series, for which a multi-input model is already fully specified, and updates the 'state set' information for use in constructing further forecasts.

The previous specification of the multi-input model will normally have been obtained by using G13BEF to estimate the relevant transfer function and ARIMA parameters. The supplied state set will originally have been produced by G13BEF (or possibly G13BJF), but may since have been updated by G13BGF.

2 Specification

```
      SUBROUTINE G13BGF(STTF, NSTTF, MR, NSER, MT, PARA, NPARA, NNV,
     1                  XXYN, IXXYN, KZEF, RES, WA, IWA, IFAIL)
      INTEGER           NSTTF, MR(7), NSER, MT(4,NSER), NPARA, NNV,
     1                  IXXYN, KZEF, IWA, IFAIL
      real              STTF(NSTTF), PARA(NPARA), XXYN(IXXYN,NSER),
     1                  RES(NNV), WA(IWA)
```

3 Description

The multi-input model is specified in Section 3 of the document for G13BEF. The form of these equations required to update the state set is as follows:

$$z_t = \delta_1 z_{t-1} + \delta_2 z_{t-2} + \ldots + \delta_p z_{t-p} + \omega_0 x_{t-b} - \omega_1 x_{t-b-1} - \ldots - \omega_q x_{t-b-q}$$

the transfer models which generate input component values $z_{i,t}$ from one or more inputs $x_{i,t}$,

$$n_t = y_t - z_{1,t} - z_{2,t} - \ldots - z_{m,t}$$

which generates the output noise component from the output y_t and the input components, and

$$\begin{aligned} w_t &= \nabla^d \nabla_s^D n_t - c \\ e_t &= w_t - \Phi_1 w_{t-s} - \Phi_2 w_{t-2\times s} - \ldots - \Phi_P w_{t-P\times s} + \Theta_1 e_{t-s} + \Theta_2 e_{t-2\times s} + \ldots + \Theta_Q e_{t-Q\times s} \\ a_t &= e_t - \phi_1 e_{t-1} - \phi_2 e_{t-2} - \ldots - \phi_p e_{t-p} + \theta_1 a_{t-1} + \theta_2 a_{t-2} + \ldots + \theta_q a_{t-q} \end{aligned}$$

the ARIMA model for the output noise which generates the residuals a_t.

The state set (as also given in Section 3 of the document for G13BEF) is the collection of terms

$$z_{n+1-k}, x_{n+1-k}, n_{n+1-k}, w_{n+1-k}, e_{n+1-k} \text{ and } a_{n+1-k}$$

for $k = 1$ up to the maximum lag associated with each of these series respectively, in the above model equations. n is the latest time point of the series from which the state set has been generated.

The routine accepts further values of the series y_t, $x_{1,t}, x_{2,t}, \ldots, x_{m,t}$ for $t = n+1, n+2, \ldots, n+l$, and applies the above model equations over this time range, to generate new values of the various model components, noise series and residuals. The state set is reconstructed, corresponding to the latest time point $n+l$, the earlier values being discarded.

The set of residuals corresponding to the new observations may be of use in checking that the new observations conform to the previously fitted model. The components of the new observations of the output series which are due to the various inputs, and the noise component, are also optionally returned.

The parameters of the model are not changed in this routine.

4 References

[1] Box G E P and Jenkins G M (1976) *Time Series Analysis: Forecasting and Control* Holden-Day (Revised Edition)

5 Parameters

1: STTF(NSTTF) — ***real*** array *Input/Output*

On entry: the NSTTF values in the state set before updating as returned by G13BEF, G13BJF or a previous call to G13BGF.

On exit: the state set values after updating.

2: NSTTF — INTEGER *Input*

On entry: the exact number of values in the state set array STTF as returned by G13BEF or G13BJF.

3: MR(7) — INTEGER array *Input*

On entry: the orders vector (p, d, q, P, D, Q, s) of the ARIMA model for the output noise component.

p, q, P and Q refer respectively to the number of autoregressive (ϕ), moving average (θ), seasonal autoregressive (Φ) and seasonal moving average (Θ) parameters.

d, D and s refer respectively to the order of non-seasonal differencing, the order of seasonal differencing, and the seasonal period.

4: NSER — INTEGER *Input*

On entry: the total number of input and output series. There may be any number of input series (including none), but only one output series.

5: MT(4,NSER) — INTEGER array *Input*

On entry: the transfer function model orders b, p and q of each of the input series. The data for input series i are held in column i. Row 1 holds the value b_i, row 2 holds the value q_i and row 3 holds the value p_i. For a simple input, $b_i = q_i = p_i = 0$.

Row 4 holds the value r_i, where $r_i = 1$ for a simple input and $r_i = 2$ or 3 for a transfer function input. When $r_i = 1$ any non-zero contents of rows 1, 2 and 3 of column i are ignored. The choice of $r_i = 2$ or $r_i = 3$ is an option for use in model estimation and does not affect the operation of this routine.

Constraint: MT(4,i) = 1,2 or 3 for $i = 1, 2, \ldots$,NSER−1.

6: PARA(NPARA) — ***real*** array *Input*

On entry: estimates of the multi-input model parameters as returned by G13BEF. These are in order, firstly the ARIMA model parameters: p values of ϕ parameters, q values of θ parameters, P values of Φ parameters and Q values of Θ parameters. These are followed by the transfer function model parameter values $\omega_0, \omega_1, \ldots, \omega_{q_1}$, $\delta_1, \delta_2, \ldots, \delta_{p_1}$ for the first of any input series and similarly for each subsequent input series. The final component of PARA is the value of the constant c.

7: NPARA — INTEGER *Input*

On entry: the exact number of ϕ, θ, Φ, Θ, ω, δ and c parameters. (c must be included whether its value was previously estimated or was set fixed.)

8: NNV — INTEGER *Input*

On entry: the number of new observation sets being used to update the state set, each observation set consisting of a value of the output series and the associated values of each of the input series at a particular time point.

9: XXYN(IXXYN,NSER) — ***real*** array *Input/Output*

On entry: the NNV new observation sets being used to update the state set. Column i contains the values of input series i. Column NSER contains the values of the output series. Consecutive rows correspond to increasing time sequence.

On exit: if KZEF = 0, XXYN remains unchanged.

If KZEF $\neq$ 0, the columns of XXYN hold the corresponding values of the input component series z_t and the output noise component n_t, in that order.

10: IXXYN — INTEGER *Input*

On entry: the first dimension of the array XXYN as declared in the (sub)program from which G13BGF is called.

Constraint: IXXYN $\geq$ NNV.

11: KZEF — INTEGER *Input*

On entry: KZEF must not be set to 0, if the values of the input component series z_t and the values of the output noise component n_t are to overwrite the contents of XXYN on exit, and must be set to 0 if XXYN is to remain unchanged on exit.

12: RES(NNV) — ***real*** array *Output*

On exit: the values of the residual series a_t corresponding to the new observations of the output series.

13: WA(IWA) — ***real*** array *Workspace*

14: IWA — INTEGER *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13BGF is called.

Constraint: IWA $\geq$ NNV + 2 $\times$ NSTTF + max(NNV,NSTTF) + max(NNV,*ncc*), where *ncc* = $4 \times (p+q+P+Q)$.

15: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NSTTF is not consistent with the orders in arrays MR and MT.

IFAIL = 2

On entry, NPARA is not consistent with the orders in arrays MR and MT.

IFAIL = 3

On entry, IXXYN is too small.

IFAIL = 4

On entry, IWA is too small.

IFAIL = 5

On entry, one of the r_i, stored in MT(4,i), for $i = 1,2,\ldots,$NSER−1 does not equal 1, 2 or 3.

7 Accuracy

The computations are believed to be stable.

8 Further Comments

The time taken by the routine is approximately proportional to NNV $\times$ NPARA.

9 Example

The example uses the data described in G13BEF in which 40 observations of an output time series and a single input series were processed. In this example a model which included seasonal differencing of order 1 was used. The 10 values of the state set and the 5 final values of PARA then obtained are used as input to this program, together with the values of 4 new observations and the transfer function orders of the input series. The model used is $\phi_1 = 0.5158$, $\Theta_1 = 0.9994$, $\omega_0 = 8.6343$, $\delta_1 = 0.6726$, $c = -0.3172$.

The following are computed and printed out: the updated state set, the residuals a_t and the values of the components z_t and the output noise component n_t corresponding to the new observations.

9.1 Program Text

```
*     G13BGF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NSERMX, NPMAX, NSTTFM, NNVMAX, IXXYN, IWA
      PARAMETER        (NSERMX=2,NPMAX=10,NSTTFM=20,NNVMAX=40,
     +                 IXXYN=NNVMAX,IWA=160)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, KZEF, NNV, NPARA, NSER, NSTTF
*     .. Local Arrays ..
      real             PARA(NPMAX), RES(NNVMAX), STTF(NSTTFM), WA(IWA),
     +                 XXYN(IXXYN,NSERMX)
      INTEGER          MR(7), MT(4,NSERMX)
*     .. External Subroutines ..
      EXTERNAL         G13BGF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BGF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NSTTF, NSER, NNV, KZEF
      IF (NSTTF.GT.0 .AND. NSTTF.LE.NSTTFM .AND. NSER.GT.0 .AND.
     +    NSER.LE.NSERMX .AND. NNV.GT.0 .AND. NNV.LE.NNVMAX) THEN
         READ (NIN,*) (MR(I),I=1,7)
         DO 20 I = 1, 4
            READ (NIN,*) (MT(I,J),J=1,NSER)
   20    CONTINUE
         NPARA = 0
         DO 40 I = 1, NSER
            NPARA = NPARA + MT(2,I) + MT(3,I)
   40    CONTINUE
         NPARA = NPARA + MR(1) + MR(3) + MR(4) + MR(6) + NSER
         READ (NIN,*) (STTF(I),I=1,NSTTF)
         IF (NPARA.LE.NPMAX) THEN
            READ (NIN,*) (PARA(I),I=1,NPARA)
            DO 60 I = 1, NNV
               READ (NIN,*) (XXYN(I,J),J=1,NSER)
   60        CONTINUE
             IFAIL = 0
*
             CALL G13BGF(STTF,NSTTF,MR,NSER,MT,PARA,NPARA,NNV,XXYN,IXXYN,
     +                   KZEF,RES,WA,IWA,IFAIL)
*
             WRITE (NOUT,*)
             WRITE (NOUT,*) 'The updated state set'
             WRITE (NOUT,99999) (STTF(I),I=1,NSTTF)
             WRITE (NOUT,*)
             WRITE (NOUT,*) 'The residuals (after differencing)'
             DO 80 I = 1, NNV
                WRITE (NOUT,99998) I, RES(I)
   80        CONTINUE
             WRITE (NOUT,*)
             WRITE (NOUT,*) 'The values of z(t) and n(t)'
             DO 100 I = 1, NNV
                WRITE (NOUT,99998) I, (XXYN(I,J),J=1,NSER)
  100        CONTINUE
          END IF
       END IF
       STOP
*
99999 FORMAT (1X,6F10.4)
99998 FORMAT (1X,I4,2F10.4)
       END
```

9.2 Program Data

```
G13BGF Example Program Data
  10    2    4    1
   1    0    0    0    1    1    4
   1    0
   0    0
   1    0
   3    0
 6.0530 184.4749 -80.0885 -75.1704 -76.9481 -81.4749   0.7776  -2.6190
-2.3054  -1.1963
 0.5158   0.9994   8.6343   0.6726  -0.3172
 5.9410  96.0000
 5.3860  95.0000
 5.8110  80.0000
 6.7160  88.0000
```

9.3 Program Results

```
G13BGF Example Program Results

The updated state set
    6.7160  158.3155  -80.3412  -74.9035  -80.7814  -70.3155
    0.8416   -2.0333   -5.8201   10.2810

The residuals (after differencing)
   1    1.4586
   2   -2.4674
   3   -4.7714
   4   13.2830
```

```
The values of z(t) and n(t)
   1  176.3412  -80.3412
   2  169.9035  -74.9035
   3  160.7814  -80.7814
   4  158.3155  -70.3155
```

G13BHF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G13BHF produces forecasts of a time series (the output series) which depends on one or more other (input) series via a multi-input model which will usually have been fitted using G13BEF. The future values of the input series must be supplied. The original observations are not required. G13BHF uses as input either the original state set obtained from G13BEF, or the state set updated by a series of new observations from G13BGF. Standard errors of the forecasts are produced. If future values of some of the input series have been obtained as forecasts using ARIMA models for those series, this may be allowed for in the calculation of the standard errors.

2 Specification

```
SUBROUTINE G13BHF(STTF, NSTTF, MR, NSER, MT, PARA, NPARA, NFV,
1                 XXYN, IXXYN, MRX, PARX, IPARX, RMSXY, KZEF, FVA,
2                 FSD, WA, IWA, IFAIL)
INTEGER           NSTTF, MR(7), NSER, MT(4,NSER), NPARA, NFV,
1                 IXXYN, MRX(7,NSER), IPARX, KZEF, IWA, IFAIL
real              STTF(NSTTF), PARA(NPARA), XXYN(IXXYN,NSER),
1                 PARX(IPARX,NSER), RMSXY(NSER), FVA(NFV),
2                 FSD(NFV), WA(IWA)
```

3 Description

The forecasts of the output series y_t are calculated for $t = n+1, n+2, \ldots, n+L$ where n is the latest time point of the observations used to produce the state set and L is the maximum lead time of the forecasts.

First the new input series values x_t are used to form the input components z_t for $t = n+1, n+2, \ldots, n+L$ using the transfer function models:

(a) $z_t = \delta_1 z_{t-1} + \delta_2 z_{t-2} + \ldots + \delta_p z_{t-p} + \omega_0 x_{t-b} - \omega_1 x_{t-b-1} - \cdots - \omega_q x_{t-b-q}$.

The output noise component n_t for $t = n+1, n+2, \ldots, n+L$ is then forecast by setting $a_t = 0$ for $t = n+1, n+2, \ldots, n+L$ and using the ARIMA model equations:

(b) $e_t = \phi_1 e_{t-1} + \phi_2 e_{t-2} + \ldots + \phi_p e_{t-p} + a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \ldots - \theta_1 a_{t-q}$

(c) $w_t = \Phi_1 w_{t-s} + \Phi_2 w_{t-2\times s} + \ldots + \Phi_P w_{t-P\times s} + e_t - \Theta_1 e_{t-s} - \Theta_2 e_{t-2\times s} - \cdots - \Theta_Q e_{t-Q\times s}$

(d) $n_t = (\nabla^d \nabla_s^D)^{-1}(w_t + c)$.

This last step of 'integration' reverses the process of differencing. Finally the output forecasts are calculated as

$$y_t = z_{1,t} + z_{2,t} + \ldots + z_{m,t} + n_t.$$

The forecast error variance of y_{t+l} (i.e., at lead time l) is S_l^2, which is the sum of parts which arise from the various input series, and the output noise component. That part due to the output noise is

$$sn_l^2 = V_n \times (\psi_0^2 + \psi_1^2 + \ldots + \psi_{l-1}^2)$$

V_n is the estimated residual variance of the output noise ARIMA model, and $\psi_0, \psi_1, \ldots$ are the 'psi-weights' of this model as defined in Box and Jenkins [1]. They are calculated by applying the equations (b), (c) and (d) above for $t = 0, 1, \ldots, L$, but with artificial values for the various series and with the constant c set to 0. Thus all values of a_t, e_t, w_t and n_t are taken as zero for $t < 0$; a_t is taken to be 1 for $t = 0$ and 0 for $t > 0$. The resulting values of n_t for $t = 0, 1, \ldots, L$ are precisely $\psi_0, \psi_1, \ldots, \psi_L$ as required.

Further contributions to S_l^2 come only from those input series, for which future values are forecasts which have been obtained by applying input series ARIMA models. For such a series the contribution is

$$sz_l^2 = V_x \times (\nu_0^2 + \nu_1^2 + \ldots + \nu_{l-1}^2)$$

V_x is the estimated residual variance of the input series ARIMA model. The coefficients $\nu_0, \nu_1, \ldots$ are calculated by applying the transfer function model equation (a) above for $t = 0, 1, \ldots, L$, but again with artificial values of the series. Thus all values of z_t and x_t for $t < 0$ are taken to be zero, and $x_0, x_1, \ldots$ are taken to be the psi-weight sequence $\psi_0, \psi_1, \ldots$ for the **input series** ARIMA model. The resulting values of z_t for $t = 0, 1, \ldots, L$ are precisely $\nu_0, \nu_1, \ldots, \nu_L$ as required.

In adding such contributions sz_l^2 to sn_l^2 to make up the total forecast error variance S_l^2, it is assumed that the various input series with which these contributions are associated, are statistically independent of each other.

When using the routine in practice an ARIMA model is required for all the input series. In the case of those inputs for which no such ARIMA model is available (or its effects are to be excluded), the corresponding orders and parameters and the estimated residual variance should be set to zero.

4 References

[1] Box G E P and Jenkins G M (1976) *Time Series Analysis: Forecasting and Control* Holden-Day (Revised Edition)

5 Parameters

1: STTF(NSTTF) — ***real*** array *Input*

On entry: the NSTTF values in the state set as returned by G13BEF or G13BGF.

2: NSTTF — INTEGER *Input*

On entry: the exact number of values in the state set array STTF as returned by G13BEF or G13BGF.

3: MR(7) — INTEGER array *Input*

On entry: the orders vector (p, d, q, P, D, Q, s) of the ARIMA model for the output noise component.

p, q, P and Q give respectively the number of autoregressive (ϕ), moving average (θ), seasonal autoregressive (Φ) and seasonal moving average (Θ) parameters.

d, D and s refer respectively to the order of non-seasonal differencing, the order of seasonal differencing, and the seasonal period.

4: NSER — INTEGER *Input*

On entry: the total number of input and output series. There may be any number of input series (including none), but only one output series.

5: MT(4,NSER) — INTEGER array *Input*

On entry: the transfer function orders b, p and q of each of the input series. The data for input series i are held in column i. Row 1 holds the value b_i, row 2 holds the value q_i and row 3 holds the value p_i. For a simple input, $b_i = q_i = p_i = 0$.

Row 4 holds the value r_i, where $r_i = 1$ for a simple input, $r_i = 2$ or 3 for a transfer function input. When $r_i = 1$, any non-zero contents of rows 1, 2 and 3 of column i are ignored. The choice of $r_i = 2$ or $r_i = 3$ is an option for use in model estimation and does not affect the operation of this routine.

Constraint: MT(4,i) = 1, 2 or 3, for $i = 1, 2, \ldots$,NSER−1.

6: PARA(NPARA) — ***real*** array *Input*

On entry: estimates of the multi-input model parameters as returned by G13BEF. These are in order firstly the ARIMA model parameters: p values of ϕ parameters, q values of θ parameters, P values of Φ parameters and Q values of Θ parameters. These are followed by the transfer function model parameter values $\omega_0, \omega_1, \ldots, \omega_{q_1}$, $\delta_1, \delta_2, \ldots, \delta_{p_1}$ for the first of any input series and similar sets of values for any subsequent input series. The final component of PARA is the constant c.

7: NPARA — INTEGER *Input*

On entry: the exact number of ϕ, θ, Φ, Θ, ω, δ and c parameters. (c must be included, whether its value was previously estimated or was set fixed).

8: NFV — INTEGER *Input*

On entry: the number of forecast values required.

9: XXYN(IXXYN,NSER) — ***real*** array *Input/Output*

On entry: the supplied NFV values for each of the input series required to produce the NFV output series forecasts. Column i contains the values for input series i. Column NSER need not be supplied.

On exit: if KZEF $= 0$, then column NSER of XXYN contains the output series forecast values (as does FVA), but XXYN is otherwise unchanged.

If KZEF $\neq 0$, then the columns of XXYN hold the corresponding values of the forecast components z_t for each of the input series and the values of the output noise component n_t in that order.

10: IXXYN — INTEGER *Input*

On entry: the first dimension of the array XXYN as declared in the (sub)program from which G13BHF is called.

Constraint: IXXYN $\geq$ NFV.

11: MRX(7,NSER) — INTEGER array *Input/Output*

On entry: the orders array for each of the input series ARIMA models. Thus, column i contains values of p, d, q, P, D, Q, s for input series i. In the case of those inputs for which no ARIMA model is available, the corresponding orders should be set to 0. (The model for any input series only affects the standard errors of the forecast values.)

On exit: unchanged, apart from column NSER which is used for workspace.

12: PARX(IPARX,NSER) — ***real*** array *Input*

On entry: values of the parameters (ϕ, θ, Φ and Θ) for each of the input series ARIMA models. Thus column i contains MRX(1,i) values of ϕ parameters, MRX(3,i) values of θ parameters, MRX(4,i) values of Φ parameters and MRX(6,i) values of Θ parameters – in that order.

Values in the columns relating to those input series for which no ARIMA model is available are ignored. (The model for any input series only affects the standard errors of the forecast values.)

13: IPARX — INTEGER *Input*

On entry: the dimension of the array PARX as declared in the (sub)program from which G13BHF is called.

Constraint: IPARX $\geq$ *ncd*, where *ncd* is the maximum number of parameters in any of the input series ARIMA models. If there are no input series, IPARX ≥ 1.

14: RMSXY(NSER) — ***real*** array *Input*

On entry: the estimated residual variances for each input series ARIMA model followed by that for the output noise ARIMA model. In the case of those inputs for which no ARIMA model is available, or when its effects are to be excluded in the calculation of forecast standard errors, the corresponding entry of RMSXY should be set to 0.

15: KZEF — INTEGER *Input*

On entry: KZEF must not be set to 0, if the values of the input component series z_t and the values of the output noise component n_t are to overwrite the contents of XXYN on exit, and must be set to 0 if XXYN is to remain unchanged on exit, apart from the appearance of the forecast values in column NSER.

16: FVA(NFV) — ***real*** array *Output*

On exit: the required forecast values for the output series.

17: FSD(NFV) — ***real*** array *Output*

On exit: the standard errors for each of the forecast values.

18: WA(IWA) — ***real*** array *Workspace*

19: IWA — INTEGER *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13BHF is called.

A good, slightly conservative approximation to the required size of IWA is given by

$$\mathrm{IWA} \geq 4 \times (\mathrm{NSTTF} + \mathrm{NFV} + ncf)$$

where ncf is the largest number of ARIMA parameters in any one of the input or output series.

An exact value for the required size of IWA can be calculated as follows:

Let $ncg = \max(p_i)$,
$nch = \max(b_i + q_i)$,
$nci = \max(b_i + q_i + p_i)$,

over each of the transfer function input series for which $r_i > 1$, where b_i, q_i, p_i are the orders held in rows 1 to 3 of array MT.

Let $ncj = 1 + nci$
$nck = \mathrm{NFV} + \max(ncg, nch)$,
$ncl = \max(\mathrm{NSTTF}, ncf, ncj, nck)$,
$ncm = \max(\mathrm{NSTTF}{+}4{\times}ncf, ncl)$.

Then IWA $\geq ncm + 3 \times ncl + \mathrm{NFV}$.

20: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NSTTF is not consistent with the orders in arrays MR and MT.

IFAIL = 2

On entry, NPARA is not consistent with the orders in arrays MR and MT.

IFAIL = 3

On entry, IXXYN is too small.

IFAIL = 4

On entry, IWA is too small.

IFAIL = 5

On entry, IPARX is too small.

IFAIL = 6

On entry, one of the r_i, stored in MT(4,i), for $i = 1, 2, \ldots$,NSER-1, does not equal 1, 2 or 3.

7 Accuracy

The computations are believed to be stable.

8 Further Comments

The time taken by the routine is approximately proportional to NFV $\times$ NPARA.

9 Example

The example follows up that described in G13BGF and makes use of its data. These consist of output series orders and parameter values, input series transfer function orders and the updated state set.

Four new values of the input series are supplied, as are the orders and parameter values for the single input series ARIMA model (which has 2 values of ϕ, 2 values of θ, 1 value of Θ, single seasonal differencing and a seasonal period of 4), and the estimated residual variances for the input series ARIMA model and the output noise ARIMA model.

Four forecast values and their standard errors are computed and printed; also the values of the components z_t and the output noise component n_t corresponding to the forecasts.

9.1 Program Text

```
*     G13BHF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NSERMX, NFVMAX, NPMAX, IPARX, NSTTFM, IXXYN, IWA
      PARAMETER        (NSERMX=2,NFVMAX=40,NPMAX=10,IPARX=NPMAX,
     +                 NSTTFM=20,IXXYN=NFVMAX,IWA=100)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, KZEF, NFV, NPARA, NSER, NSTTF
*     .. Local Arrays ..
      real             FSD(NFVMAX), FVA(NFVMAX), PARA(NPMAX),
     +                 PARX(IPARX,NSERMX), RMSXY(NSERMX), STTF(NSTTFM),
     +                 WA(IWA), XXYN(IXXYN,NSERMX)
      INTEGER          MR(7), MRX(7,NSERMX), MT(4,NSERMX)
*     .. External Subroutines ..
      EXTERNAL         G13BHF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BHF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NSTTF, NSER, NFV, KZEF
      IF (NSTTF.GT.0 .AND. NSTTF.LE.NSTTFM .AND. NSER.GT.0 .AND.
     +    NSER.LE.NSERMX .AND. NFV.GT.0 .AND. NFV.LE.NFVMAX) THEN
```

```
         READ (NIN,*) (MR(I),I=1,7)
         DO 20 I = 1, 4
            READ (NIN,*) (MT(I,J),J=1,NSER)
   20    CONTINUE
         NPARA = 0
         DO 40 I = 1, NSER
            NPARA = NPARA + MT(2,I) + MT(3,I)
   40    CONTINUE
         NPARA = NPARA + MR(1) + MR(3) + MR(4) + MR(6) + NSER
         READ (NIN,*) (STTF(I),I=1,NSTTF)
         IF (NPARA.LE.NPMAX) THEN
            READ (NIN,*) (PARA(I),I=1,NPARA)
            DO 60 I = 1, NFV
               READ (NIN,*) (XXYN(I,J),J=1,NSER)
   60       CONTINUE
            DO 80 I = 1, 7
               READ (NIN,*) (MRX(I,J),J=1,NSER)
   80       CONTINUE
            DO 100 I = 1, NPARA
               READ (NIN,*) (PARX(I,J),J=1,NSER)
  100       CONTINUE
            READ (NIN,*) (RMSXY(I),I=1,NSER)
            IFAIL = 0
*
            CALL G13BHF(STTF,NSTTF,MR,NSER,MT,PARA,NPARA,NFV,XXYN,IXXYN,
     +                  MRX,PARX,IPARX,RMSXY,KZEF,FVA,FSD,WA,IWA,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +        'The forecast values and their standard errors'
            WRITE (NOUT,*)
            WRITE (NOUT,*) '   I      FVA        FSD'
            WRITE (NOUT,*)
            DO 120 I = 1, NFV
               WRITE (NOUT,99999) I, FVA(I), FSD(I)
  120       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'The values of z(t) and n(t)'
            DO 140 I = 1, NFV
               WRITE (NOUT,99999) I, (XXYN(I,J),J=1,NSER)
  140       CONTINUE
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,I4,2F10.4)
      END
```

9.2 Program Data

```
G13BHF Example Program Data
   10    2    4    1
    1    0    0    0    1    1    4
    1    0
    0    0
    1    0
    3    0
  6.7160 158.3022 -80.3352 -74.8937 -80.7694 -70.3022   0.8476  -2.0234
```

```
-5.8080  10.2943
 0.5158   0.9994   8.6343   0.6726  -0.3172
  6.923   0.0000
  6.939   0.0000
  6.705   0.0000
  6.914   0.0000
   2    0
   0    0
   2    0
   0    0
   1    0
   1    0
   4    0
 1.6743  0.0
-0.9505  0.0
 1.4605  0.0
-0.4862  0.0
 0.8993  0.0
 0.1720 22.9256
```

9.3 Program Results

```
G13BHF Example Program Results

The forecast values and their standard errors

   I     FVA       FSD

   1   88.2723    4.7881
   2   99.9425    6.4690
   3  100.6499    7.3175
   4   95.0958    7.5534

The values of z(t) and n(t)
   1  164.4620  -76.1897
   2  170.3924  -70.4499
   3  174.5193  -73.8694
   4  175.2747  -80.1789
```

G13BJF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13BJF produces forecasts of a time series (the output series) which depends on one or more other (input) series via a previously estimated multi-input model for which the state set information is not available. The future values of the input series must be supplied. In contrast with G13BHF the original past values of the input and output series are required. Standard errors of the forecasts are produced. If future values of some of the input series have been obtained as forecasts using ARIMA models for those series, this may be allowed for in the calculation of the standard errors.

2. Specification

```
      SUBROUTINE G13BJF (MR, NSER, MT, PARA, NPARA, KFC, NEV, NFV, XXY,
     1                   IXXY, KZEF, RMSXY, MRX, PARX, IPARX, FVA, FSD,
     2                   STTF, ISTTF, NSTTF, WA, IWA, MWA, IMWA, IFAIL)
      INTEGER            MR(7), NSER, MT(4,NSER), NPARA, KFC, NEV, NFV,
     1                   IXXY, KZEF, MRX(7,NSER), IPARX, ISTTF, NSTTF, IWA,
     2                   MWA(IMWA), IMWA, IFAIL
      real               PARA(NPARA), XXY(IXXY,NSER), RMSXY(NSER),
     1                   PARX(IPARX,NSER), FVA(NFV), FSD(NFV), STTF(ISTTF),
     2                   WA(IWA)
```

3. Description

The routine has two stages. The first stage is essentially the same as a call to the model estimation routine G13BEF, with zero iterations. In particular, all the parameters remain unchanged in the supplied input series transfer function models and output noise series ARIMA model. The internal nuisance parameters associated with the pre-observation period effects of the input series are estimated where requested, and so are any backforecasts of the output noise series. The output components z_t and n_t, and residuals a_t are calculated exactly as in Section 3 of G13BEF, and the state set for forecasting is constituted.

The second stage is essentially the same as a call to the forecasting routine G13BHF. The same information is required, and the same information is returned.

Use of G13BJF should be confined to situations in which the state set for forecasting is unknown. Forecasting from the original data is relatively expensive because it requires recalculation of the state set. G13BJF returns the state set for use in producing further forecasts using G13BHF, or for updating the state set using G13BGF.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control.
Holden-Day, (Revised Edition) 1976.

5. Parameters

1: MR(7) – INTEGER array. *Input*

On entry: the first 7 elements of MR must specify the orders vector (p,d,q,P,D,Q,s), of the ARIMA model for the output noise component.

p, q, P, Q refer respectively to the number of autoregressive (ϕ), moving average (θ), seasonal autoregressive (Φ) and seasonal moving average (Θ) parameters.

d, D, s refer respectively to the order of non-seasonal differencing, the order of seasonal differencing and the seasonal period.

2: NSER – INTEGER. *Input*

On entry: the number of input and output series. There may be any number of input series (including none), but only one output series.

3: MT(4,NSER) – INTEGER array. *Input*

On entry: the transfer function model orders b, p and q of each of the input series. The data for input series i are held in column i. Row 1 holds the value b_i, row 2 holds the value q_i and row 3 holds the value p_i.

For a simple input, $b_i = q_i = p_i = 0$.

Row 4 holds the value r_i, where $r_i = 1$ for a simple input, and $r_i = 2$ or 3 for a transfer function input.

The choice $r_i = 3$ leads to estimation of the pre-period input effects as nuisance parameters, and $r_i = 2$ suppresses this estimation. This choice may affect the returned forecasts and the state set.

When $r_i = 1$, any non-zero contents of rows 1,2 and 3 of column i are ignored.

Constraint: MT$(4,i) = 1$, 2 or 3, for $i = 1,2,...,\text{NSER}-1$.

4: PARA(NPARA) – ***real*** array. *Input*

On entry: estimates of the multi-input model parameters. These are in order firstly the ARIMA model parameters: p values of ϕ parameters, q values of θ parameters, P values of Φ parameters, Q values of Θ parameters.

These are followed by the transfer function model parameter values $\omega_0,\omega_1,...,\omega_{q_1}$, $\delta_1,...,\delta_{p_1}$ for the first of any input series and similarly for each subsequent input series. The final component of PARA is the value of the constant c.

5: NPARA – INTEGER. *Input*

On entry: the exact number of ϕ, θ, Φ, Θ, ω, δ, c parameters, so that NPARA $= p + q + P + Q + \text{NSER} + \sum(p_i+q_i)$, the summation being over all the input series. (c must be included whether its value was previously estimated or was set fixed.)

6: KFC – INTEGER. *Input*

On entry: KFC must be set to 1 if the constant was estimated when the model was fitted, and 0 if it was held at a fixed value. This only affects the degrees of freedom used in calculating the estimated residual variance.

Constraint: KFC = 0 or 1.

7: NEV – INTEGER. *Input*

On entry: the number of original (undifferenced) values in each of the input and output time-series.

8: NFV – INTEGER. *Input*

On entry: the number of forecast values of the output series required.

Constraint: NFV > 0.

9: XXY(IXXY,NSER) – ***real*** array. *Input/Output*

On entry: the columns of XXY must contain in the first NEV places, the past values of each of the input and output series, in that order. In the next NFV places, the columns relating to the input series (i.e. columns 1 to NSER – 1) contain the future values of the input series which are necessary for construction of the forecasts of the output series y.

On exit: if KZEF = 0 then XXY is unchanged except that the relevant NFV values in the column relating to the output series (column NSER) contain the forecast values (FVA), but if KZEF ≠ 0 then the columns of XXY contain the corresponding values of the input component series z_t and the values of the output noise component n_t in that order.

10: IXXY – INTEGER. *Input*

On entry: the first dimension of the array XXY as declared in the (sub)program from which G13BJF is called.

Constraint: IXXY ≥ (NEV+NFV).

11: KZEF – INTEGER. *Input*

On entry: KZEF must be set to 0 if the relevant NFV values of the forecasts (FVA) are to be held in the output series column (NSER) of XXY (which is otherwise unchanged) on exit, and must not be set to 0, if the values of the input component series z_t and the values of the output noise component n_t are to overwrite the contents of XXY on exit.

12: RMSXY(NSER) – ***real*** array. *Input/Output*

On entry: the first (NSER−1) elements of RMSXY must contain the estimated residual variance of the input series ARIMA models. In the case of those inputs for which no ARIMA model is available or its effects are to be excluded in the calculation of forecast standard errors, the corresponding entry of RMSXY should be set to 0.

On exit: RMSXY(NSER) contains the estimated residual variance of the output noise ARIMA model which is calculated from the supplied series. Otherwise RMSXY is unchanged.

13: MRX(7,NSER) – INTEGER array. *Input/Output*

On entry: the orders array for each of the input series ARIMA models. Thus, column i contains values of p, d, q, P, D, Q, s for input series i. In the case of those inputs for which no ARIMA model is available, the corresponding orders should be set to 0.

On exit: unchanged, except for column NSER which is used as workspace.

14: PARX(IPARX,NSER) – ***real*** array. *Input*

On entry: values of the parameters (ϕ, θ, Φ, and Θ) for each of the input series ARIMA models.

Thus column i contains MRX(1,i) values of ϕ, MRX(3,i) values of θ, MRX(4,i) values of Φ and MRX(6,i) values of Θ – in that order.

Values in the columns relating to those input series for which no ARIMA model is available are ignored.

15: IPARX – INTEGER. *Input*

On entry: the first dimension of the array PARX as declared in the (sub)program from which G13BJF is called.

Constraint: IPARX ≥ *nce*, where *nce* is the maximum number of parameters in any of the input series ARIMA models. If there are no input series, then IPARX ≥ 1.

16: FVA(NFV) – ***real*** array. *Output*

On exit: the required forecast values for the output series. (Note that these are also output in column NSER of XXY if KZEF = 0.)

17: FSD(NFV) – ***real*** array. *Output*

On exit: the standard errors for each of the forecast values.

18: STTF(ISTTF) – ***real*** array. *Output*

On exit: the NSTTF values of the state set based on the first NEV sets of (past) values of the input and output series.

19: ISTTF – INTEGER. *Input*

On entry: the dimension of the array STTF as declared in the (sub)program from which G13BJF is called.

Constraint: ISTTF $\geq$ $(p \times s) + d + (D \times s) + q + \max(p, Q \times s) + ncf$, where $ncf = \Sigma(b_i + q_i + p_i)$, where the summation is over all input series for which $r_i > 1$.

20: NSTTF – INTEGER. *Output*

On exit: the number of values in the state set array STTF.

21: WA(IWA) – ***real*** array. *Workspace*

22: IWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13BJF is called.

It is not practical to outline a method for deriving the exact minimum permissible value of IWA, but the following gives a reasonably good approximation which tends to be on the conservative side.

Note: there are three error indicators associated with IWA. These are IFAIL = 4, 5 and 6. The first of these probably indicates an abnormal entry value of NFV, while the second indicates that IWA is much too small and needs to be increased by a substantial amount. The last of these indicates that IWA is too small but returns the necessary minimum value in MWA(1).

Let $q' = q + (Q \times s)$
$d' = d + (D \times s)$

where the output noise ARIMA model orders are p, d, q, P, D, Q, s.

Let there be l input series, where l = NSER – 1.

Let $mx_i = \max(b_i + q_i, p_i)$, if $r_i = 3$ for $i = 1,2,...,l$ if $l > 0$
$mx_i = 0$, if $r_i \neq 3$ for $i = 1,2,...,l$ if $l > 0$

where the transfer function model orders of input series i are given by b_i, q_i, p_i, r_i.

Let $qx = \max(q', mx_1, mx_2, ..., mx_l)$

Let $ncg = \text{NPARA} + qx + \sum_{i=1}^{l} mx_i$

and $nch = N + d + (6 \times qx)$.

Finally, let nci = NSER, and then increment nci by 1 every time any of the following conditions are satisfied. (The last two conditions should be applied separately to each input series, so that for example if we have two input series and $p_1 > 0$ and $p_2 > 0$, then nci is incremented by 2 in respect of these.)

The conditions are:

$p > 0$
$q > 0$
$P > 0$
$Q > 0$
$qx > 0$
$mx_i > 0$ separately for $i = 1,2,...,l$ if $l > 0$
$p_i > 0$ separately for $i = 1,2,...,l$ if $l > 0$.

Then IWA $> 2 \times (ncg)^2 + nch \times (nci + 4)$.

23: MWA(IMWA) – INTEGER array. *Workspace*

24: IMWA – INTEGER. *Input*

On entry: the dimension of the array MWA as declared in the (sub)program from which G13BJF is called.

Constraint: IMWA $\geq$ (16$\times$NSER) + (7$\times ncg$) + (3$\times$NPARA) + 27, where the derivation of ncg is described under IWA above.

When IMWA is too small, as indicated by IFAIL = 7, the requisite minimum value of IMWA is returned in MWA(1).

25: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, KFC < 0,
or KFC > 1,
or IXXY < (NEV+NFV),
or NFV $\leq$ 0.

IFAIL = 2

On entry, IPARX is too small or NPARA is inconsistent with the orders specified in arrays MR and MT; or one of the r_i, stored in MT(4,i), is not equal to 1, 2 or 3.

IFAIL = 3

On entry, or during execution, one or more sets of δ parameters do not satisfy the stationarity or invertibility test conditions.

IFAIL = 4

On entry, IWA is too small for the final forecasting calculations. This is a highly unlikely error, and would probably indicate that NFV was abnormally large.

IFAIL = 5

On entry, IWA is too small by a very considerable margin. No information is supplied about the requisite minimum size.

IFAIL = 6

On entry, IWA is too small, but the requisite minimum size is returned in MWA(1).

IFAIL = 7

On entry, IMWA is too small, but the requisite minimum size is returned in MWA(1).

IFAIL = 8

This indicates a failure in F04ASF, which is used to solve the equations giving the latest estimates of the parameters.

IFAIL = 9

This indicates a failure in the inversion of the second derivative matrix associated with parameter estimation.

IFAIL = 10

On entry, or during execution, one or more sets of the ARIMA (ϕ, θ, Φ or Θ) parameters do not satisfy the stationarity or invertibility test conditions.

IFAIL = 11

On entry, ISTTF is too small.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken by the routine is approximately proportional to the product of the length of each series and the square of the number of parameters in the multi-input model.

9. Example

The data in the example relate to 40 observations of an output time series and 5 input time series. This differs from the example in G13BEF in that there are now 4 simple input series. The output series has one autoregressive (ϕ) parameter and one seasonal moving average (Θ) parameter. The seasonal period is 4. The transfer function input (the fifth in the set) is defined by orders $b_5 = 1, q_5 = 0, p_5 = 1, r_5 = 3$, so that it allows for pre-observation period effects. The initial values of the specified model are:

$$\phi = 0.495,\ \Theta = 0.238,\ \omega_1 = -0.367 \quad \omega_2 = -3.876 \quad \omega_3 = 4.516$$
$$\omega_4 = 2.474 \quad \omega_{5,1} = 8.629 \quad \delta_{5,1} = 0.688,\ c = -82.858.$$

A further 8 values of the input series are supplied, and it is assumed that the values for the fifth series have themselves been forecast from an ARIMA model with orders 2 0 2 0 1 1 4 , in which $\phi_1 = 1.6743$, $\phi_2 = -0.9505$, $\theta_1 = 1.4605$, $\theta_2 = -0.4862$ and $\Theta_1 = 0.8993$, and for which the residual mean square is 0.1720.

The following are computed and printed out: the state set after initial processing of the original 40 sets of values, the estimated residual variance for the output noise series, the 8 forecast values and their standard errors, and the values of the components z_t and the output noise component n_t.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13BJF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NSERMX, NPMAX, IPARX, NFVMAX, ISTTF, NEVMAX,
     +                 IXXY, IWA, IMWA
      PARAMETER        (NSERMX=6,NPMAX=10,IPARX=8,NFVMAX=10,ISTTF=20,
     +                 NEVMAX=40,IXXY=NEVMAX+NFVMAX,IWA=1500,IMWA=250)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, KFC, KZEF, N, NEV, NFV, NPARA, NSER,
     +                 NSTTF
*     .. Local Arrays ..
      real             FSD(NFVMAX), FVA(NFVMAX), PARA(NFVMAX),
     +                 PARX(IPARX,NSERMX), RMSXY(NSERMX), STTF(ISTTF),
     +                 WA(IWA), XXY(IXXY,NSERMX)
      INTEGER          MR(7), MRX(7,NSERMX), MT(4,NSERMX), MWA(IMWA)
*     .. External Subroutines ..
      EXTERNAL         G13BJF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13BJF Example Program Results'
```

```
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) KFC, NEV, NFV, NSER, KZEF
      IF (NSER.GT.0 .AND. NSER.LE.NSERMX .AND. NFV.GT.0 .AND. NFV.LE.
     +    NFVMAX .AND. NEV.GT.0 .AND. NEV.LE.NEVMAX) THEN
         READ (NIN,*) (MR(I),I=1,7)
         DO 20 I = 1, 4
            READ (NIN,*) (MT(I,J),J=1,NSER)
   20    CONTINUE
         NPARA = 0
         DO 40 I = 1, NSER
            NPARA = NPARA + MT(2,I) + MT(3,I)
   40    CONTINUE
         NPARA = NPARA + MR(1) + MR(3) + MR(4) + MR(6) + NSER
         IF (NPARA.LE.NPMAX) THEN
            READ (NIN,*) (PARA(I),I=1,NPARA)
            N = NEV + NFV
            DO 60 I = 1, N
               READ (NIN,*) (XXY(I,J),J=1,NSER)
   60       CONTINUE
            READ (NIN,*) (RMSXY(I),I=1,NSER)
            DO 80 I = 1, 7
               READ (NIN,*) (MRX(I,J),J=1,NSER)
   80       CONTINUE
            DO 100 I = 1, 5
               READ (NIN,*) (PARX(I,J),J=1,NSER)
  100       CONTINUE
            IFAIL = 1
*
            CALL G13BJF(MR,NSER,MT,PARA,NPARA,KFC,NEV,NFV,XXY,IXXY,KZEF,
     +                  RMSXY,MRX,PARX,IPARX,FVA,FSD,STTF,ISTTF,NSTTF,
     +                  WA,IWA,MWA,IMWA,IFAIL)
*
            WRITE (NOUT,*)
            IF (IFAIL.NE.0) WRITE (NOUT,99999) 'G13BJF fails. IFAIL =',
     +          IFAIL
            IF (IFAIL.EQ.0 .OR. IFAIL.EQ.8 .OR. IFAIL.EQ.9 .OR.
     +          IFAIL.EQ.11) THEN
               WRITE (NOUT,99999) 'After processing', NEV,
     +           ' sets of observations'
               WRITE (NOUT,99998) NSTTF,
     +           ' values of the state set are derived'
               WRITE (NOUT,*)
               WRITE (NOUT,99997) (STTF(I),I=1,NSTTF)
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'The residual mean square for the output'
               WRITE (NOUT,99996)
     +           'series is also derived and its value is', RMSXY(NSER)
               WRITE (NOUT,*)
               WRITE (NOUT,*)
     +           'The forecast values and their standard errors are'
               WRITE (NOUT,*)
               WRITE (NOUT,*) '   I       FVA       FSD'
               WRITE (NOUT,*)
               DO 120 I = 1, NFV
                  WRITE (NOUT,99995) I, FVA(I), FSD(I)
  120          CONTINUE
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'The values of z(t) and n(t) are'
               WRITE (NOUT,*)
               WRITE (NOUT,*)
     +  '   I       z1        z2        z3        z4        z5        n'
               WRITE (NOUT,*)
               DO 140 I = 1, N
                  WRITE (NOUT,99994) I, (XXY(I,J),J=1,NSER)
  140          CONTINUE
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I3,A)
99998 FORMAT (1X,I3,A)
99997 FORMAT (1X,6F10.4)
99996 FORMAT (1X,A,F10.4)
99995 FORMAT (1X,I4,F10.3,F10.4)
99994 FORMAT (1X,I4,6F10.3)
      END
```

9.2. Program Data

```
G13BJF Example Program Data
    1   40    8    6    1
    1    0    0    0    0    1    4
    0    0    0    0    1    0
    0    0    0    0    0    0
    0    0    0    0    1    0
    1    1    1    1    3    0
  0.4950  0.2380 -0.3670 -3.8760  4.5160  2.4740  8.6290  0.6880
-82.8580
   1.0      1.0      0.0      0.0      8.075 105.0
   1.0      0.0      1.0      0.0      7.819 119.0
   1.0      0.0      0.0      1.0      7.366 119.0
   1.0     -1.0     -1.0     -1.0      8.113 109.0
   2.0      1.0      0.0      0.0      7.380 117.0
   2.0      0.0      1.0      0.0      7.134 135.0
   2.0      0.0      0.0      1.0      7.222 126.0
   2.0     -1.0     -1.0     -1.0      7.768 112.0
   3.0      1.0      0.0      0.0      7.386 116.0
   3.0      0.0      1.0      0.0      6.965 122.0
   3.0      0.0      0.0      1.0      6.478 115.0
   3.0     -1.0     -1.0     -1.0      8.105 115.0
   4.0      1.0      0.0      0.0      8.060 122.0
   4.0      0.0      1.0      0.0      7.684 138.0
   4.0      0.0      0.0      1.0      7.580 135.0
   4.0     -1.0     -1.0     -1.0      7.093 125.0
   5.0      1.0      0.0      0.0      6.129 115.0
   5.0      0.0      1.0      0.0      6.026 108.0
   5.0      0.0      0.0      1.0      6.679 100.0
   5.0     -1.0     -1.0     -1.0      7.414  96.0
   6.0      1.0      0.0      0.0      7.112 107.0
   6.0      0.0      1.0      0.0      7.762 115.0
   6.0      0.0      0.0      1.0      7.645 123.0
   6.0     -1.0     -1.0     -1.0      8.639 122.0
   7.0      1.0      0.0      0.0      7.667 128.0
   7.0      0.0      1.0      0.0      8.080 136.0
   7.0      0.0      0.0      1.0      6.678 140.0
   7.0     -1.0     -1.0     -1.0      6.739 122.0
   8.0      1.0      0.0      0.0      5.569 102.0
   8.0      0.0      1.0      0.0      5.049 103.0
   8.0      0.0      0.0      1.0      5.642  89.0
   8.0     -1.0     -1.0     -1.0      6.808  77.0
   9.0      1.0      0.0      0.0      6.636  89.0
   9.0      0.0      1.0      0.0      8.241  94.0
   9.0      0.0      0.0      1.0      7.968 104.0
   9.0     -1.0     -1.0     -1.0      8.044 108.0
  10.0      1.0      0.0      0.0      7.791 119.0
  10.0      0.0      1.0      0.0      7.024 126.0
  10.0      0.0      0.0      1.0      6.102 119.0
  10.0     -1.0     -1.0     -1.0      6.053 103.0
  11.0      1.0      0.0      0.0      5.941   0.0
  11.0      0.0      1.0      0.0      5.386   0.0
  11.0      0.0      0.0      1.0      5.811   0.0
  11.0     -1.0     -1.0     -1.0      6.716   0.0
  12.0      1.0      0.0      0.0      6.923   0.0
```

```
12.0     0.0     1.0     0.0     6.939   0.0
12.0     0.0     0.0     1.0     6.705   0.0
12.0    -1.0    -1.0    -1.0     6.914   0.0
 0.0     0.0     0.0     0.0     0.1720  0.0
 0    0    0    0    2    0
 0    0    0    0    0    0
 0    0    0    0    2    0
 0    0    0    0    0    0
 0    0    0    0    1    0
 0    0    0    0    1    0
 0    0    0    0    4    0
0.0      0.0     0.0     0.0     1.6743  0.0
0.0      0.0     0.0     0.0    -0.9505  0.0
0.0      0.0     0.0     0.0     1.4605  0.0
0.0      0.0     0.0     0.0    -0.4862  0.0
0.0      0.0     0.0     0.0     0.8993  0.0
```

9.3. Program Results

```
G13BJF Example Program Results

After processing 40 sets of observations
  6 values of the state set are derived

    6.0530  193.8741    2.0790   -2.8580   -3.5906   -2.5203

The residual mean square for the output
series is also derived and its value is   20.7599

The forecast values and their standard errors are

    I        FVA        FSD

    1     93.398     4.5563
    2     96.958     6.2172
    3     86.046     7.0933
    4     77.589     7.3489
    5     82.139     7.3941
    6     96.276     7.5823
    7     98.345     8.1445
    8     93.577     8.8536

The values of z(t) and n(t) are

    I       z1         z2         z3         z4         z5          n

    1     -0.339     -3.889      0.000      0.000    188.603    -79.375
    2     -0.339      0.000      4.514      0.000    199.438    -84.613
    3     -0.339      0.000      0.000      2.479    204.683    -87.823
    4     -0.339      3.889     -4.514     -2.479    204.383    -91.940
    5     -0.678     -3.889      0.000      0.000    210.623    -89.056
    6     -0.678      0.000      4.514      0.000    208.591    -77.426
    7     -0.678      0.000      0.000      2.479    205.070    -80.870
    8     -0.678      3.889     -4.514     -2.479    203.407    -87.624
    9     -1.017     -3.889      0.000      0.000    206.974    -86.068
   10     -1.017      0.000      4.514      0.000    206.132    -87.628
   11     -1.017      0.000      0.000      2.479    201.920    -88.381
   12     -1.017      3.889     -4.514     -2.479    194.819    -75.698
   13     -1.356     -3.889      0.000      0.000    203.974    -76.729
   14     -1.356      0.000      4.514      0.000    209.884    -75.041
   15     -1.356      0.000      0.000      2.479    210.705    -76.828
   16     -1.356      3.889     -4.514     -2.479    210.373    -80.912
   17     -1.695     -3.889      0.000      0.000    205.942    -85.358
   18     -1.695      0.000      4.514      0.000    194.575    -89.394
   19     -1.695      0.000      0.000      2.479    185.866    -86.650
   20     -1.695      3.889     -4.514     -2.479    185.509    -84.709
   21     -2.035     -3.889      0.000      0.000    191.606    -78.682
   22     -2.035      0.000      4.514      0.000    193.194    -80.673
   23     -2.035      0.000      0.000      2.479    199.896    -77.340
   24     -2.035      3.889     -4.514     -2.479    203.497    -76.358
  25    -2.374    -3.889     0.000     0.000   214.552   -80.290
  26    -2.374     0.000     4.514     0.000   213.770   -79.910
  27    -2.374     0.000     0.000     2.479   216.796   -76.901
  28    -2.374     3.889    -4.514    -2.479   206.780   -79.302
  29    -2.713    -3.889     0.000     0.000   200.416   -91.814
  30    -2.713     0.000     4.514     0.000   185.941   -84.742
  31    -2.713     0.000     0.000     2.479   171.495   -82.261
  32    -2.713     3.889    -4.514    -2.479   166.673   -83.857
  33    -3.052    -3.889     0.000     0.000   173.418   -77.477
  34    -3.052     0.000     4.514     0.000   176.573   -84.035
  35    -3.052     0.000     0.000     2.479   192.594   -88.021
  36    -3.052     3.889    -4.514    -2.479   201.261   -87.105
  37    -3.391    -3.889     0.000     0.000   207.879   -81.599
  38    -3.391     0.000     4.514     0.000   210.249   -85.372
  39    -3.391     0.000     0.000     2.479   205.262   -85.350
  40    -3.391     3.889    -4.514    -2.479   193.874   -84.379
  41    -3.730    -3.889     0.000     0.000   185.617   -84.600
  42    -3.730     0.000     4.514     0.000   178.969   -82.795
  43    -3.730     0.000     0.000     2.479   169.607   -82.309
  44    -3.730     3.889    -4.514    -2.479   166.832   -82.409
  45    -4.069    -3.889     0.000     0.000   172.733   -82.636
  46    -4.069     0.000     4.514     0.000   178.579   -82.748
  47    -4.069     0.000     0.000     2.479   182.739   -82.804
  48    -4.069     3.889    -4.514    -2.479   183.582   -82.831
```

G13CAF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13CAF calculates the smoothed sample spectrum of a univariate time series using one of four lag windows – rectangular, Bartlett, Tukey or Parzen window.

2. Specification

```
      SUBROUTINE G13CAF (NX, MTX, PX, IW, MW, IC, NC, C, KC, L, LG, NXG,
     1                   XG, NG, STATS, IFAIL)
      INTEGER            NX, MTX, IW, MW, IC, NC, KC, L, LG, NXG, NG,
     1                   IFAIL
      real               PX, C(NC), XG(NXG), STATS(4)
```

3. Description

The smoothed sample spectrum is defined as

$$\hat{f}(\omega) = \frac{1}{2\pi}\left(C_0+2\sum_{k=1}^{M-1} w_k C_k \cos(\omega k)\right)$$

where M is the window width, and is calculated for frequency values

$$\omega_i = \frac{2\pi i}{L}, \quad i = 0,1,\ldots,[L/2]$$

where [ ] denotes the integer part.

The autocovariances C_k may be supplied by the user, or constructed from a time series $x_1,x_2,\ldots,x_n$, as

$$C_k = \frac{1}{n}\sum_{t=1}^{n-k} x_t x_{t+k}$$

the fast Fourier transform (FFT) being used to carry out the convolution in this formula.

The time series may be mean or trend corrected (by classical least squares), and tapered before calculation of the covariances, the tapering factors being those of the split cosine bell:

$$\begin{array}{ll} \frac{1}{2}(1-\cos(\pi(t-\frac{1}{2})/T)), & 1 \le t \le T \\ \frac{1}{2}(1-\cos(\pi(n-t+\frac{1}{2})/T)), & n+1-T \le t \le n \\ 1, & \text{otherwise} \end{array}$$

where $T = \left[\frac{np}{2}\right]$ and p is the tapering proportion.

The smoothing window is defined by

$$w_k = W\left(\frac{k}{M}\right), \quad k \le M-1$$

which for the various windows is defined over $0 \le \alpha < 1$ by

rectangular:

$$W(\alpha) = 1$$

Bartlett:

$$W(\alpha) = 1 - \alpha$$

Tukey:

$$W(\alpha) = \tfrac{1}{2}(1+\cos(\pi\alpha))$$

Parzen:

$$W(\alpha) = 1 - 6\alpha^2 + 6\alpha^3, \qquad 0 \leq \alpha \leq \tfrac{1}{2}$$
$$W(\alpha) = 2(1-\alpha)^3, \qquad \tfrac{1}{2} < \alpha < 1.$$

The sampling distribution of $\hat{f}(\omega)$ is approximately that of a scaled χ_d^2 variate, whose degrees of freedom d is provided by the routine, together with multiplying limits mu, ml from which approximate 95% confidence intervals for the true spectrum $f(\omega)$ may be constructed as $[ml \times \hat{f}(\omega),\ mu \times \hat{f}(\omega)]$. Alternatively, $\log \hat{f}(\omega)$ may be returned, with additive limits.

The bandwidth b of the corresponding smoothing window in the frequency domain is also provided. Spectrum estimates separated by (angular) frequencies much greater than b may be assumed to be independent.

4. References

[1] JENKINS, G.M. and WATTS, D.G.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[2] BLOOMFIELD, P.
Fourier Analysis of Time Series: An Introduction.
Wiley, 1976.

5. Parameters

1: NX – INTEGER. *Input*

On entry: the length of the time series, n.

Constraint: NX $\geq$ 1.

2: MTX – INTEGER. *Input*

On entry: if covariances are to be calculated by the routine (IC = 0), MTX must specify whether the data are to be initially mean or trend corrected.

MTX = 0 for no correction,
MTX = 1 for mean correction,
MTX = 2 for trend correction.

Constraint: 0 $\leq$ MTX $\leq$ 2, if IC = 0.

If covariances are supplied (IC $\neq$ 0), MTX is not used.

3: PX – ***real***. *Input*

On entry: if covariances are to be calculated by the routine (IC = 0), PX must specify the proportion of the data (totalled over both ends) to be initially tapered by the split cosine bell taper. If covariances are supplied (IC $\neq$ 0), then PX must specify the proportion of data tapered before the supplied covariances were calculated and after any mean or trend correction. PX is required for the calculation of output statistics. A value of 0.0 implies no tapering.

Constraint: 0.0 $\leq$ PX $\leq$ 1.0.

4: IW – INTEGER. *Input*

On entry: the choice of lag window. IW = 1 for rectangular, 2 for Bartlett, 3 for Tukey or 4 for Parzen.

Constraint: 1 $\leq$ IW $\leq$ 4.

5: MW – INTEGER. *Input*

On entry: the 'cut-off' point, M, of the lag window. Windowed covariances at lag M or greater are zero.

Constraint: 1 $\leq$ MW $\leq$ NX.

6: IC – INTEGER. *Input*

On entry: indicates whether covariances are to be calculated in the routine or supplied in the call to the routine.

IC = 0 if covariances are to be calculated,
IC ≠ 0 if covariances are to be supplied.

7: NC – INTEGER. *Input*

On entry: the number of covariances to be calculated in the routine or supplied in the call to the routine.

Constraint: MW ≤ NC ≤ NX.

8: C(NC) – ***real*** array. *Input/Output*

On entry: if IC ≠ 0, then C must contain the NC covariances for lags from 0 to (NC−1), otherwise C need not be set.

On exit: if IC = 0, C will contain the NC calculated covariances.

If IC ≠ 0, the contents of C will be unchanged.

9: KC – INTEGER. *Input*

On entry: if IC = 0, KC must specify the order of the fast Fourier transform (FFT) used to calculate the covariances. KC should be a product of small primes such as 2^m where m is the smallest integer such that 2^m ≥ NX + NC, provided $m \leq 20$.

If IC ≠ 0, that is covariances are supplied, then KC is not used.

Constraints: KC ≥ NX + NC.

The largest prime factor of KC must not exceed 19, and the total number of prime factors of KC, counting repetitions, must not exceed 20. These two restrictions are imposed by C06EAF which performs the FFT.

10: L – INTEGER. *Input*

On entry: the frequency division, L, of the spectral estimates as $\frac{2\pi}{L}$. Therefore it is also the order of the FFT used to construct the sample spectrum from the covariances. L should be a product of small primes such as 2^m where m is the smallest integer such that $2^m \geq 2M - 1$, provided $m \leq 20$.

Constraints: L ≥ 2×MW − 1.

The largest prime factor of L must not exceed 19, and the total number of prime factors of L, counting repetitions, must not exceed 20. These two restrictions are imposed by C06EAF which performs the FFT.

11: LG – INTEGER. *Input*

On entry: indicates whether unlogged or logged spectral estimates and confidence limits are required.

LG = 0 for unlogged, LG ≠ 0 for logged.

12: NXG – INTEGER. *Input*

On entry: the length of the array XG, as declared in the (sub)program from which G13CAF is called.

Constraints: if covariances are to be calculated, i.e. IC = 0, NXG ≥ max(KC,L),
if covariances are supplied, i.e. if IC ≠ 0, NXG ≥ L.

13: XG(NXG) – ***real*** array. *Input/Output*

On entry: if the covariances are to be calculated, then XG must contain the NX data points. If covariances are supplied, XG may contain any values.

On exit: contains the NG spectral estimates, $\hat{f}(\omega_i)$, for $i = 0,1,...,[L/2]$ in XG(1) to XG(NG) respectively (logged if LG $\neq$ 0). The elements XG(i), for i = NG+1,...,NXG contain 0.0

14: NG – INTEGER. *Output*

On exit: the number of spectral estimates, [L/2] + 1, in XG.

15: STATS(4) – ***real*** array. *Output*

On exit: four associated statistics. These are the degrees of freedom in STATS(1), the lower and upper 95% confidence limit factors in STATS(2) and STATS(3) respectively (logged if LG $\neq$ 0), and the bandwidth in STATS(4).

16: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NX < 1,
or MTX < 0 and IC = 0,
or MTX > 2 and IC = 0,
or PX < 0.0,
or PX > 1.0,
or IW < 1,
or IW > 4,
or MW < 1,
or MW > NX,
or NC < MW,
or NC > NX,
or NXG < max(KC,L) and IC = 0,
or NXG < L and IC $\neq$ 0.

IFAIL = 2

On entry, KC < NX + NC,
or KC has a prime factor exceeding 19,
or KC has more than 20 prime factors, counting repetitions.

This error only occurs when IC = 0.

IFAIL = 3

On entry, L < 2×MW – 1,
or L has a prime factor exceeding 19,
or L has more than 20 prime factors, counting repetitions.

IFAIL = 4

One or more spectral estimates are negative. Unlogged spectral estimates are returned in XG, and the degrees of freedom, unlogged confidence limit factors and bandwidth in STATS.

IFAIL = 5

The calculation of confidence limit factors has failed. This error will not normally occur. Spectral estimates (logged if requested) are returned in XG, and degrees of freedom and bandwidth in STATS.

7. Accuracy

The FFT is a numerically stable process, and any errors introduced during the computation will normally be insignificant compared with uncertainty in the data.

8. Further Comments

G13CAF carries out two FFTs of length KC to calculate the covariances and one FFT of length L to calculate the sample spectrum. The time taken by the routine for an FFT of length n is approximately proportional to $n\log(n)$ (but see Section 8 of the document C06EAF for further details).

9. Example

The example program reads a time series of length 256. It selects the mean correction option, a tapering proportion of 0.1, the Parzen smoothing window and a cut-off point for the window at lag 100. It chooses to have 100 auto-covariances calculated and unlogged spectral estimates at a frequency division of $2\pi/200$. It then calls G13CAF to calculate the univariate spectrum and statistics and prints the autocovariances and the spectrum together with its 95% confidence multiplying limits.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13CAF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXG, NCMAX
      PARAMETER        (NXG=500,NCMAX=200)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             PX
      INTEGER          I, IC, IFAIL, IW, KC, L, LG, MTX, MW, NC, NG, NX
*     .. Local Arrays ..
      real             C(NCMAX), STATS(4), XG(NXG)
*     .. External Subroutines ..
      EXTERNAL         G13CAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13CAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX, NC
      IF (NX.GT.0 .AND. NX.LE.NXG .AND. NC.GT.0 .AND. NC.LE.NCMAX) THEN
         READ (NIN,*) (XG(I),I=1,NX)
         MTX = 1
         PX = 0.1e0
         IW = 4
         MW = 100
         IC = 0
         KC = 360
         L = 200
         LG = 0
         IFAIL = 1
*
```

```
         CALL G13CAF(NX,MTX,PX,IW,MW,IC,NC,C,KC,L,LG,NXG,XG,NG,STATS,
     +               IFAIL)
*
         WRITE (NOUT,*)
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99999) 'G13CAF fails. IFAIL =', IFAIL
            WRITE (NOUT,*)
         END IF
         IF (IFAIL.EQ.0 .OR. IFAIL.GE.4) THEN
            WRITE (NOUT,*) 'Covariances'
            WRITE (NOUT,99998) (C(I),I=1,NC)
            WRITE (NOUT,*)
            WRITE (NOUT,99997) 'Degrees of freedom =', STATS(1),
     +        '      Bandwidth =', STATS(4)
            WRITE (NOUT,*)
            WRITE (NOUT,99996)
     +        '95 percent confidence limits -     Lower =', STATS(2),
     +        '  Upper =', STATS(3)
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +       '      Spectrum       Spectrum       Spectrum       Spectrum'
            WRITE (NOUT,*)
     +       '      estimate       estimate       estimate       estimate'
            WRITE (NOUT,99995) (I,XG(I),I=1,NG)
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I3)
99998 FORMAT (1X,6F11.4)
99997 FORMAT (1X,A,F4.1,A,F7.4)
99996 FORMAT (1X,A,F7.4,A,F7.4)
99995 FORMAT (1X,I4,F10.4,I5,F10.4,I5,F10.4,I5,F10.4)
      END
```

9.2. Program Data

```
G13CAF Example Program Data
 256 100
  5.0  11.0  16.0  23.0  36.0  58.0  29.0  20.0  10.0   8.0   3.0   0.0
  0.0   2.0  11.0  27.0  47.0  63.0  60.0  39.0  28.0  26.0  22.0  11.0
 21.0  40.0  78.0 122.0 103.0  73.0  47.0  35.0  11.0   5.0  16.0  34.0
 70.0  81.0 111.0 101.0  73.0  40.0  20.0  16.0   5.0  11.0  22.0  40.0
 60.0  80.9  83.4  47.7  47.8  30.7  12.2   9.6  10.2  32.4  47.6  54.0
 62.9  85.9  61.2  45.1  36.4  20.9  11.4  37.8  69.8 106.1 100.8  81.6
 66.5  34.8  30.6   7.0  19.8  92.5 154.4 125.9  84.8  68.1  38.5  22.8
 10.2  24.1  82.9 132.0 130.9 118.1  89.9  66.6  60.0  46.9  41.0  21.3
 16.0   6.4   4.1   6.8  14.5  34.0  45.0  43.1  47.5  42.2  28.1  10.1
  8.1   2.5   0.0   1.4   5.0  12.2  13.9  35.4  45.8  41.1  30.1  23.9
 15.6   6.6   4.0   1.8   8.5  16.6  36.3  49.6  64.2  67.0  70.9  47.8
 27.5   8.5  13.2  56.9 121.5 138.3 103.2  85.7  64.6  36.7  24.2  10.7
 15.0  40.1  61.5  98.5 124.7  96.3  66.6  64.5  54.1  39.0  20.6   6.7
  4.3  22.7  54.8  93.8  95.8  77.2  59.1  44.0  47.0  30.5  16.3   7.3
 37.6  74.0 139.0 111.2 101.6  66.2  44.7  17.0  11.3  12.4   3.4   6.0
 32.3  54.3  59.7  63.7  63.5  52.2  25.4  13.1   6.8   6.3   7.1  35.6
 73.0  85.1  78.0  64.0  41.8  26.2  26.7  12.1   9.5   2.7   5.0  24.4
 42.0  63.5  53.8  62.0  48.5  43.9  18.6   5.7   3.6   1.4   9.6  47.4
 57.1 103.9  80.6  63.6  37.6  26.1  14.2   5.8  16.7  44.3  63.9  69.0
 77.8  64.9  35.7  21.2  11.1   5.7   8.7  36.1  79.7 114.4 109.6  88.8
 67.8  47.5  30.6  16.3   9.6  33.2  92.6 151.6 136.3 134.7  83.9  69.4
 31.5  13.9   4.4  38.0
```

9.3. Program Results

```
G13CAF Example Program Results

Covariances
  1152.9733   937.3289   494.9243    14.8648  -342.8548  -514.6479
  -469.2733  -236.6896   109.0608   441.3498   637.4571   641.9954
   454.0505   154.5960  -136.8016  -343.3911  -421.8441  -374.4095
  -241.1943   -55.6140   129.4067   267.4248   311.8293   230.2807
    56.4402  -146.4689  -320.9948  -406.4077  -375.6384  -273.5936
  -132.6214    11.0791   126.4843   171.3391   122.6284   -11.5482
  -169.2623  -285.2358  -331.4567  -302.2945  -215.4832  -107.8732
    -3.4126    73.2521    98.0831    71.8949    17.0985   -27.5632
   -76.7900  -110.5354  -126.1383  -121.1043  -103.9362   -67.4619
   -10.8678    58.5009   116.4587   140.0961   129.5928    66.3211
   -35.5487  -135.3894  -203.7149  -216.2161  -152.7723   -30.4361
    99.3397   188.9594   204.9047   148.4056    34.4975  -103.7840
  -208.5982  -252.4128  -223.7600  -120.8640    23.3565   156.0956
   227.7642   228.5123   172.3820    87.4911   -21.2170  -117.5282
  -176.3634  -165.1218   -75.1308    67.1634   195.7290   279.3039
   290.8258   225.3811   104.0784   -44.4731  -162.7355  -207.7480
  -165.2444   -48.5473   118.8872   265.0045

Degrees of freedom = 9.0      Bandwidth = 0.1165

95 percent confidence limits -      Lower = 0.4731  Upper = 3.3329

        Spectrum          Spectrum          Spectrum          Spectrum
        estimate          estimate          estimate          estimate
   1    210.4696     2    428.2020     3    810.1419     4    922.5900
   5    706.1605     6    393.4052     7    207.6481     8    179.0657
   9    170.1320    10    133.0442    11    103.6752    12    103.0644
  13    141.5173    14    194.3041    15    266.5730    16    437.0181
  17    985.3130    18   2023.1574    19   2681.8980    20   2363.7439
  21   1669.9001    22   1012.1320    23    561.4822    24    467.2741
  25    441.9977    26    300.1985    27    172.0184    28    114.7823
  29     79.1533    30     49.4882    31     27.0902    32     16.8081
  33     27.5111    34     59.4429    35     97.0145    36    119.3664
  37    116.6737    38     87.3142    39     54.9570    40     42.9781
  41     46.6097    42     53.6206    43     50.6050    44     36.7780
  45     25.6285    46     24.8555    47     30.2626    48     31.5642
  49     27.3351    50     22.4443    51     18.5418    52     15.2425
  53     12.0207    54     12.6846    55     18.3975    56     19.3058
  57     12.6103    58      7.9511    59      7.1333    60      5.4996
  61      3.4182    62      3.2359    63      5.3836    64      8.5225
  65     10.0610    66      7.9483    67      4.2261    68      3.2631
  69      5.5751    70      7.8491    71      9.3694    72     11.0791
  73     10.1386    74      6.3158    75      3.6375    76      2.6561
  77      1.8026    78      1.0103    79      1.0693    80      2.3950
  81      4.0822    82      4.6221    83      4.0672    84      3.8460
  85      4.8489    86      6.3964    87      6.4762    88      4.9457
  89      4.4444    90      5.2131    91      5.0389    92      4.6141
  93      5.8722    94      7.9268    95      7.9486    96      5.7854
  97      4.5495    98      5.2696    99      6.3893   100      6.5216
 101      6.2129
```

G13CBF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13CBF calculates the smoothed sample spectrum of a univariate time series using spectral smoothing by the trapezium frequency (Daniell) window.

2. Specification

```
      SUBROUTINE G13CBF (NX, MTX, PX, MW, PW, L, KC, LG, XG, NG, STATS,
     1                   IFAIL)
      INTEGER          NX, MTX, MW, L, KC, LG, NG, IFAIL
      real             PX, PW, XG(KC), STATS(4)
```

3. Description

The supplied time series may be mean or trend corrected (by least-squares), and tapered, the tapering factors being those of the split cosine bell:

$$\begin{array}{ll} \frac{1}{2}\left(1-\cos\left(\frac{\pi(t-\frac{1}{2})}{T}\right)\right), & 1 \le t \le T \\ \frac{1}{2}\left(1-\cos\left(\frac{\pi(n-t+\frac{1}{2})}{T}\right)\right), & n+1-T \le t \le n \\ 1, & \text{otherwise} \end{array}$$

where $T = \left[\frac{np}{2}\right]$ and p is the tapering proportion.

The unsmoothed sample spectrum

$$f^*(\omega) = \frac{1}{2\pi}\left|\sum_{t=1}^{n} x_t \exp(i\omega t)\right|^2$$

is then calculated for frequency values

$$\omega_k = \frac{2\pi k}{K}, \quad k = 0,1,\ldots,[K/2]$$

where [ ] denotes the integer part.

The smoothed spectrum is returned at a subset of these frequencies for which k is a multiple of a chosen value r, i.e.

$$\omega_{rl} = \nu_l = \frac{2\pi l}{L}, \quad l = 0,1,\ldots,[L/2]$$

where $K = r\times L$. The user will normally fix L first, then choose r so that K is sufficiently large to provide an adequate representation for the unsmoothed spectrum, i.e. $K \ge 2\times n$. It is possible to take $L = K$, i.e. $r = 1$.

The smoothing is defined by a trapezium window whose shape is supplied by the function

$$W(\alpha) = 1, \quad |\alpha| \le p$$

$$W(\alpha) = \frac{1-|\alpha|}{1-p}, \quad p < |\alpha| \le 1$$

the proportion p being supplied by the user.

The width of the window is fixed as $\frac{2\pi}{M}$ by the user supplying M. A set of averaging weights are constructed:

$$W_k = g\times W\left(\frac{\omega_k M}{\pi}\right), \quad 0 \le \omega_k \le \frac{\pi}{M}$$

where g is a normalising constant, and the smoothed spectrum obtained is

$$\hat{f}(\nu_l) = \sum_{|\omega_k|<\frac{\pi}{M}} W_k f^*(\nu_l+\omega_k).$$

If no smoothing is required M should be set to n, in which case the values returned are $\hat{f}(\nu_l) = f^*(\nu_l)$. Otherwise, in order that the smoothing approximates well to an integration, it is essential that $K \gg M$, and preferable, but not essential, that K be a multiple of M. A choice of $L > M$ would normally be required to supply an adequate description of the smoothed spectrum. Typical choices of $L \simeq n$ and $K \simeq 4n$ should be adequate for usual smoothing situations when $M < n/5$.

The sampling distribution of $\hat{f}(\omega)$ is approximately that of a scaled χ^2_d variate, whose degrees of freedom d is provided by the routine, together with multiplying limits mu, ml from which approximate 95% confidence intervals for the true spectrum $f(\omega)$ may be constructed as $[ml{\times}\hat{f}(\omega), mu{\times}\hat{f}(\omega)]$. Alternatively, $\log \hat{f}(\omega)$ may be returned, with additive limits.

The bandwidth b of the corresponding smoothing window in the frequency domain is also provided. Spectrum estimates separated by (angular) frequencies much greater than b may be assumed to be independent.

4. References

[1] JENKINS, G.M. and WATTS, D.G.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[2] BLOOMFIELD, P.
Fourier Analysis of Time Series: An Introduction.
Wiley, 1976.

5. Parameters

1: NX – INTEGER. *Input*

On entry: the length of the time series, n.

Constraint: NX $\geq$ 1.

2: MTX – INTEGER. *Input*

On entry: whether the data are to be initially mean or trend corrected.

MTX = 0 for no correction,
MTX = 1 for mean correction,
MTX = 2 for trend correction.

Constraint: 0 $\leq$ MTX $\leq$ 2.

3: PX – ***real.*** *Input*

On entry: the proportion of the data (totalled over both ends) to be initially tapered by the split cosine bell taper. (A value of 0.0 implies no tapering).

Constraint: 0.0 $\leq$ PX $\leq$ 1.0.

4: MW – INTEGER. *Input*

On entry: the value of M which determines the frequency width of the smoothing window as $\frac{2\pi}{M}$. A value of n implies no smoothing is to be carried out.

Constraint: 1 $\leq$ MW $\leq$ NX.

5: PW – ***real.*** *Input*

On entry: the shape parameter, p, of the trapezium frequency window.

A value of 0.0 gives a triangular window, and a value of 1.0 a rectangular window.

If MW = NX (i.e. no smoothing is carried out), then PW is not used.

Constraint: $0.0 \le \text{PW} \le 1.0$.

6: L – INTEGER. *Input*

On entry: the frequency division, L, of smoothed spectral estimates as $\dfrac{2\pi}{L}$.

Constraints: L ≥ 1.
L must be a factor of KC (see below).

7: KC – INTEGER. *Input*

On entry: the order of the fast Fourier transform (FFT), K, used to calculate the spectral estimates. KC should be a multiple of small primes such as 2^m where m is the smallest integer such that $2^m \ge 2n$, provided $m \le 20$.

Constraints: KC ≥ 2×NX.

KC must be a multiple of L.

The largest prime factor of KC must not exceed 19, and the total number of prime factors of KC, counting repetitions, must not exceed 20. These two restrictions are imposed by C06EAF which performs the FFT.

8: LG – INTEGER. *Input*

On entry: indicates whether unlogged or logged spectral estimates and confidence limits are required.

LG = 0 for unlogged.
LG ≠ 0 for logged.

9: XG(KC) – ***real*** array. *Input/Output*

On entry: the n data points.

On exit: contains the NG spectral estimates $\hat{f}(\omega_i)$, for $i = 0,1,...,[L/2]$, in XG(1) to XG(NG) (logged if LG ≠ 0). The elements XG(i), for i = NG+1,...,KC contain 0.0.

10: NG – INTEGER. *Output*

On exit: the number of spectral estimates, [L/2] + 1, in XG.

11: STATS(4) – ***real*** array. *Output*

On exit: four associated statistics. These are the degrees of freedom in STATS(1), the lower and upper 95% confidence limit factors in STATS(2) and STATS(3) respectively (logged if LG ≠ 0), and the bandwidth in STATS(4).

12: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.** To suppress the output of an error message when soft failure occurs, set IFAIL to 1.

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

IFAIL = 1

On entry,	$NX < 1$,
or	$MTX < 0$,
or	$MTX > 2$,
or	$PX < 0.0$,
or	$PX > 1.0$,
or	$MW < 1$,
or	$MW > NX$,
or	$PW < 0.0$ and $MW \neq NX$,
or	$PW > 1.0$ and $MW \neq NX$,
or	$L < 1$.

IFAIL = 2

On entry,	$KC < 2 \times NX$,
or	KC is not a multiple of L,
or	KC has a prime factor exceeding 19,
or	KC has more than 20 prime factors, counting repetitions.

IFAIL = 3

This indicates that a serious error has occurred. Check all array subscripts and subroutine parameter lists in calls to G13CBF. Seek expert help.

IFAIL = 4

One or more spectral estimates are negative. Unlogged spectral estimates are returned in XG, and the degrees of freedom, unlogged confidence limit factors and bandwidth in STATS.

IFAIL = 5

The calculation of confidence limit factors has failed. This error will not normally occur. Spectral estimates (logged if requested) are returned in XG, and degrees of freedom and bandwidth in STATS.

7. Accuracy

The FFT is a numerically stable process, and any errors introduced during the computation will normally be insignificant compared with uncertainty in the data.

8. Further Comments

G13CBF carries out a FFT of length KC to calculate the sample spectrum. The time taken by the routine for this is approximately proportional to KC×log (KC) (but see Section 8 of C06EAF routine document for further details).

9. Example

The example program reads a time series of length 131. It selects the mean correction option, a tapering proportion of 0.2, the option of no smoothing and a frequency division for logged spectral estimates of $\frac{2\pi}{100}$. It then calls G13CBF to calculate the univariate spectrum and prints the logged spectrum together with 95% confidence limits. The program then selects a smoothing window with frequency width $\frac{2\pi}{30}$ and shape parameter 0.5 and recalculates and prints the logged spectrum and 95% confidence limits.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13CBF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          KCMAX, NXMAX
      PARAMETER        (KCMAX=400,NXMAX=KCMAX/2)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             PW, PX
      INTEGER          I, IFAIL, KC, L, LG, MTX, MW, NG, NX
*     .. Local Arrays ..
      real             STATS(4), XG(KCMAX), XH(NXMAX)
*     .. External Subroutines ..
      EXTERNAL         G13CBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13CBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX
      IF (NX.GT.0 .AND. NX.LE.NXMAX) THEN
         READ (NIN,*) (XH(I),I=1,NX)
         MTX = 1
         PX = 0.2e0
         MW = NX
         PW = 0.5e0
         KC = 400
         L = 100
         LG = 1
   20    READ (NIN,*,END=60) MW
         IF (MW.GT.0 .AND. MW.LE.NX) THEN
            DO 40 I = 1, NX
               XG(I) = XH(I)
   40       CONTINUE
            IFAIL = 1
*
            CALL G13CBF(NX,MTX,PX,MW,PW,L,KC,LG,XG,NG,STATS,IFAIL)
*
            WRITE (NOUT,*)
            IF (MW.EQ.NX) THEN
               WRITE (NOUT,*) 'No smoothing'
            ELSE
               WRITE (NOUT,99999)
     +           'Frequency width of smoothing window = 1/', MW
            END IF
            WRITE (NOUT,*)
            IF (IFAIL.NE.0) THEN
               WRITE (NOUT,99999) 'G13CBF fails. IFAIL =', IFAIL
               WRITE (NOUT,*)
            END IF
            IF (IFAIL.EQ.0 .OR. IFAIL.GE.4) THEN
               WRITE (NOUT,99998) 'Degrees of freedom =', STATS(1),
     +           '      Bandwidth =', STATS(4)
               WRITE (NOUT,*)
               WRITE (NOUT,99997)
     +           '95 percent confidence limits -     Lower =', STATS(2),
     +           '  Upper =', STATS(3)
               WRITE (NOUT,*)
               WRITE (NOUT,*)
     +     '       Spectrum       Spectrum       Spectrum       Spectrum'
```

```
                  WRITE (NOUT,*)
     +     '        estimate        estimate        estimate        estimate'
                  WRITE (NOUT,99996) (I,XG(I),I=1,NG)
               END IF
               GO TO 20
            END IF
         END IF
   60 STOP
*
99999 FORMAT (1X,A,I3)
99998 FORMAT (1X,A,F4.1,A,F7.4)
99997 FORMAT (1X,A,F7.4,A,F7.4)
99996 FORMAT (1X,I4,F10.4,I5,F10.4,I5,F10.4,I5,F10.4)
      END
```

9.2. Program Data

```
G13CBF Example Program Data
 131
 11.500  9.890  8.728  8.400  8.230  8.365  8.383  8.243
  8.080  8.244  8.490  8.867  9.469  9.786 10.100 10.714
 11.320 11.900 12.390 12.095 11.800 12.400 11.833 12.200
 12.242 11.687 10.883 10.138  8.952  8.443  8.231  8.067
  7.871  7.962  8.217  8.689  8.989  9.450  9.883 10.150
 10.787 11.000 11.133 11.100 11.800 12.250 11.350 11.575
 11.800 11.100 10.300  9.725  9.025  8.048  7.294  7.070
  6.933  7.208  7.617  7.867  8.309  8.640  9.179  9.570
 10.063 10.803 11.547 11.550 11.800 12.200 12.400 12.367
 12.350 12.400 12.270 12.300 11.800 10.794  9.675  8.900
  8.208  8.087  7.763  7.917  8.030  8.212  8.669  9.175
  9.683 10.290 10.400 10.850 11.700 11.900 12.500 12.500
 12.800 12.950 13.050 12.800 12.800 12.800 12.600 11.917
 10.805  9.240  8.777  8.683  8.649  8.547  8.625  8.750
  9.110  9.392  9.787 10.340 10.500 11.233 12.033 12.200
 12.300 12.600 12.800 12.650 12.733 12.700 12.259 11.817
 10.767  9.825  9.150
 131
  30
```

9.3. Program Results

```
 G13CBF Example Program Results

 No smoothing

 Degrees of freedom = 2.0      Bandwidth = 0.0480

 95 percent confidence limits -     Lower =-1.3053  Upper = 3.6762

       Spectrum        Spectrum        Spectrum        Spectrum
       estimate        estimate        estimate        estimate
    1   -5.9354    2   -0.1662    3   -0.8250    4   -0.9452
    5    3.2137    6    0.2738    7   -1.0690    8   -1.0401
    9   -1.2388   10   -3.5434   11   -5.2568   12   -3.2450
   13   -2.4294   14   -3.9987   15   -2.9853   16   -4.6631
   17   -4.3317   18   -4.6982   19   -4.6335   20   -3.6732
   21   -5.8411   22   -4.7727   23   -3.9747   24   -4.8351
   25   -5.9979   26   -6.1169   27   -5.5245   28   -4.4774
   29   -5.6331   30   -4.0707   31   -4.6921   32   -5.6515
   33   -9.2919   34   -4.6302   35   -4.1700   36   -4.7829
   37   -6.6058   38   -5.8145   39   -5.2714   40   -5.8736
   41  -10.2188   42   -5.7887   43   -7.0751   44   -7.4055
   45   -8.2774   46   -7.8966   47   -6.4435   48   -5.7844
   49   -5.4690   50   -6.8709   51   -8.7123

 Frequency width of smoothing window = 1/ 30
```

```
Degrees of freedom = 7.0       Bandwidth = 0.1767

95 percent confidence limits -     Lower =-0.8275  Upper = 1.4213

      Spectrum         Spectrum         Spectrum         Spectrum
      estimate         estimate         estimate         estimate
   1   -0.1776    2   -0.4561    3   -0.1784    4    1.9042
   5    2.1094    6    1.7061    7   -0.7659    8   -1.4734
   9   -1.5939   10   -2.1157   11   -2.9151   12   -2.7055
  13   -2.8200   14   -3.4077   15   -3.8813   16   -3.6607
  17   -4.0601   18   -4.4756   19   -4.2700   20   -4.3092
  21   -4.5711   22   -4.8111   23   -4.5658   24   -4.7285
  25   -5.4386   26   -5.5081   27   -5.2325   28   -5.0262
  29   -4.4539   30   -4.4764   31   -4.9152   32   -5.8492
  33   -5.5872   34   -4.9804   35   -4.8904   36   -5.2666
  37   -5.7643   38   -5.8620   39   -5.5011   40   -5.7129
  41   -6.3894   42   -6.4027   43   -6.1352   44   -6.5766
  45   -7.3676   46   -7.1405   47   -6.1674   48   -5.8600
  49   -6.1036   50   -6.2673   51   -6.4321
```

G13CCF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13CCF calculates the smoothed sample cross spectrum of a bivariate time series using one of four lag windows – rectangular, Bartlett, Tukey or Parzen window.

2. Specification

```
 SUBROUTINE G13CCF (NXY, MTXY, PXY, IW, MW, IS, IC, NC, CXY, CYX, KC,
1                   L, NXYG, XG, YG, NG, IFAIL)
 INTEGER           NXY, MTXY, IW, MW, IS, IC, NC, KC, L, NXYG, NG,
1                  IFAIL
 real              PXY, CXY(NC), CYX(NC), XG(NXYG), YG(NXYG)
```

3. Description

The smoothed sample cross spectrum is a complex valued function of frequency ω, $f_{xy}(\omega) = cf(\omega) + iqf(\omega)$, defined by its real part or co-spectrum

$$cf(\omega) = \frac{1}{2\pi}\sum_{k=-M+1}^{M-1} w_k C_{xy}(k+S)\cos(\omega k)$$

and imaginary part or quadrature spectrum

$$qf(\omega) = \frac{1}{2\pi}\sum_{k=-M+1}^{M-1} w_k C_{xy}(k+S)\sin(\omega k)$$

where $w_k = w_{-k}$, $k = 0,1,...,M-1$ is the smoothing lag window as defined in the description of G13CAF. The alignment shift S is recommended to be chosen as the lag k at which the cross covariances $c_{xy}(k)$ peak, so as to minimize bias.

The results are calculated for frequency values

$$\omega_j = \frac{2\pi j}{L}, \qquad j = 0,1,...,[L/2]$$

where [ ] denotes the integer part.

The cross covariances $c_{xy}(k)$ may be supplied by the user, or constructed from supplied series $x_1,x_2,...,x_n$; $y_1,y_2,...,y_n$ as

$$c_{xy}(k) = \frac{\sum_{t=1}^{n-k} x_t y_{t+k}}{n}, \qquad k \geq 0$$

$$c_{xy}(k) = \frac{\sum_{t=1-k}^{n} x_t y_{t+k}}{n} = c_{yx}(-k), \qquad k < 0$$

this convolution being carried out using the finite Fourier transform.

The supplied series may be mean and trend corrected and tapered before calculation of the cross covariances, in exactly the manner described in G13CAF for univariate spectrum estimation. The results are corrected for any bias due to tapering.

The bandwidth associated with the estimates is not returned. It will normally already have been calculated in previous calls of G13CAF for estimating the univariate spectra of y_t and x_t.

4. References

[1] JENKINS, G.M. and WATTS, D.G.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[2] BLOOMFIELD, P.
Fourier Analysis of Time Series: An Introduction.
Wiley, 1976.

5. Parameters

1: NXY – INTEGER. *Input*

On entry: the length, n, of the time series x and y.

Constraint: NXY $\geq$ 1.

2: MTXY – INTEGER. *Input*

On entry: if cross covariances are to be calculated by the routine (IC = 0), MTXY must specify whether the data is to be initially mean or trend corrected.

MTXY = 0 for no correction,
MTXY = 1 for mean correction,
MTXY = 2 for trend correction,

if cross covariances are supplied (IC $\neq$ 0), MTXY is not used.

Constraint: 0 $\leq$ MTXY $\leq$ 2 if IC = 0.

3: PXY – ***real***. *Input*

On entry: if cross covariances are to be calculated by the routine (IC = 0), PXY must specify the proportion of the data (totalled over both ends) to be initially tapered by the split cosine bell taper. A value of 0.0 implies no tapering. If cross covariances are supplied (IC $\neq$ 0), PXY is not used.

Constraint: 0.0 $\leq$ PXY $\leq$ 1.0, if IC = 0.

4: IW – INTEGER. *Input*

On entry: the choice of lag window. IW = 1 for rectangular, 2 for Bartlett, 3 for Tukey or 4 for Parzen.

Constraint: 1 $\leq$ IW $\leq$ 4.

5: MW – INTEGER. *Input*

On entry: the 'cut-off' point, M, of the lag window, relative to any alignment shift that has been applied. Windowed cross covariances at lags (–MW+IS) or less, and at lags (MW+IS) or greater are zero.

Constraints: MW $\geq$ 1,
MW + |IS| $\leq$ NXY.

6: IS – INTEGER. *Input*

On entry: the alignment shift, S, between the x and y series. If x leads y, the shift is positive.

Constraint: –MW < IS < MW.

7: IC – INTEGER. *Input*

On entry: indicates whether cross covariances are to be calculated in the routine or supplied in the call to the routine.

IC = 0 if cross covariances are to be calculated,
IC $\neq$ 0 if cross covariances are to be supplied.

8: NC – INTEGER. *Input*

On entry: the number of cross covariances to be calculated in the routine or supplied in the call to the routine.

Constraint: MW + |IS| ≤ NC ≤ NXY.

9: CXY(NC) – ***real*** array. *Input/Output*

On entry: if IC ≠ 0, then CXY must contain the NC cross covariances between values in the y series and earlier values in time in the x series, for lags from 0 to (NC–1). If IC = 0 CXY need not be set.

On exit: if IC = 0, CYX will contain the NC calculated cross covariances.

If IC ≠ 0, the contents of CXY will be unchanged.

10: CYX(NC) – ***real*** array. *Input/Output*

On entry: if IC ≠ 0, then CYX must contain the NC cross covariances between values in the y series and later values in time in the x series, for lags from 0 to (NC–1). If IC = 0, CYX need not be set.

On exit: if IC = 0, CYX will contain the NC calculated cross covariances.

If IC ≠ 0, the contents of CYX will be unchanged.

11: KC – INTEGER. *Input*

On entry: if IC = 0, KC must specify the order of the fast Fourier transform (FFT) used to calculate the cross covariances. KC should be a product of small primes such as 2^m where m is the smallest integer such that $2^m \geq n$ + NC.

If IC ≠ 0, that is if covariances are supplied, then KC is not used.

Constraints: KC ≥ NXY + NC.

The largest prime factor of KC must not exceed 19, and the total number of prime factors of KC, counting repetitions, must not exceed 20. These two restrictions are imposed by C06EAF and C06EBF which perform the FFT.

12: L – INTEGER. *Input*

On entry: the frequency division, L of the spectral estimates as $\frac{2\pi}{L}$. Therefore it is also the order of the FFT used to construct the sample spectrum from the cross covariances. L should be a product of small primes such as 2^m where m is the smallest integer such that $2^m \geq 2M - 1$.

Constraints: L ≥ 2×MW – 1.

The largest prime factor of L must not exceed 19, and the total number of prime factors of L, counting repetitions, must not exceed 20. These two restrictions are imposed by C06EAF which performs the FFT.

13: NXYG – INTEGER. *Input*

On entry: the length of the smaller of the arrays XG and YG, as declared in the (sub)program from which G13CCF is called.

Constraints: if IC = 0, that is cross covariances are to be calculated, NXYG ≥ max(KC,L),
if IC ≠ 0, that is cross covariances are to be supplied, NXYG ≥ L.

14: XG(NXYG) – ***real*** array. *Input/Output*

On entry: if the cross covariances are to be calculated, then XG must contain the NXY data points of the x series. If covariances are supplied, XG need not be set.

On exit: contains the real parts of the NG complex spectral estimates in elements XG(1) to XG(NG), and XG(NG+1) to XG(NXYG) contain 0.0. The y series leads the x series.

15: YG(NXYG) – ***real*** array. *Input/Output*

On entry: if cross covariances are to be calculated, then YG must contain the NXY data points of the y series. If covariances are supplied, YG need not be set.

On exit: contains the imaginary parts of the NG complex spectral estimates in elements YG(1) to YG(NG), and YG(NG+1) to YG(NXYG) contain 0.0. The y series leads the x series.

16: NG – INTEGER. *Output*

On exit: the number, $[L/2] + 1$, of complex spectral estimates, whose separate parts are held in XG and YG.

17: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry,	NXY < 1,
or	MTXY < 0 and IC = 0,
or	MTXY > 2 and IC = 0,
or	PXY < 0.0 and IC = 0,
or	PXY > 1.0 and IC = 0,
or	IW ≤ 0,
or	IW > 4,
or	MW < 1,
or	MW + |IS| > NXY,
or	|IS| ≥ MW,
or	NC < MW + |IS|,
or	NC > NXY,
or	NXYG < max(KC,L) and IC = 0,
or	NXYG < L and IC ≠ 0.

IFAIL = 2

On entry,	KC < NXY + NC,
or	KC has a prime factor exceeding 19,
or	KC has more than 20 prime factors, counting repetitions.

This error only occurs when IC = 0.

IFAIL = 3

On entry,	L < 2×MW – 1,
or	L has a prime factor exceeding 19,
or	L has more than 20 prime factors, counting repetitions.

7. Accuracy

The FFT is a numerically stable process, and any errors introduced during the computation will normally be insignificant compared with uncertainty in the data.

8. Further Comments

G13CCF carries out two FFTs of length KC by calls to C06EAF and C06EBF to calculate the sample cross covariances and one FFT of length L to calculate the sample spectrum. The timing of G13CCF is therefore dependent on the choice of these values. The time taken for an FFT of length n is approximately proportional to $n \log n$ (but see Section 8 of the document C06EAF for further details).

9. Example

The example program reads 2 time series of length 296. It then selects mean correction, a 10% tapering proportion, the Parzen smoothing window and a cut-off point of 35 for the lag window. The alignment shift is set to 3 and 50 cross covariances are chosen to be calculated. The program then calls G13CCF to calculate the cross spectrum and then prints the cross covariances and cross spectrum.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13CCF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXYG, NCMAX
      PARAMETER        (NXYG=350,NCMAX=50)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             PXY
      INTEGER          I, IC, IFAIL, II, IS, IW, KC, L, MTXY, MW, NC,
     +                 NG, NXY
*     .. Local Arrays ..
      real             CXY(NCMAX), CYX(NCMAX), XG(NXYG), YG(NXYG)
*     .. External Subroutines ..
      EXTERNAL         G13CCF
*     .. Intrinsic Functions ..
      INTRINSIC        MIN
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13CCF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NXY, NC, IC
      IF (NXY.GT.0 .AND. NXY.LE.NXYG .AND. NC.GT.0 .AND. NC.LE.NCMAX)
     +    THEN
         IF (IC.EQ.0) THEN
            READ (NIN,*) (XG(I),I=1,NXY)
            READ (NIN,*) (YG(I),I=1,NXY)
         ELSE
            READ (NIN,*) (CXY(I),I=1,NC)
            READ (NIN,*) (CYX(I),I=1,NC)
         END IF
*        Set parameters for call to G13CCF
*        Mean correction and 10 percent taper
         MTXY = 1
         PXY = 0.1e0
*        Parzen window and zero covariance at lag 35
         IW = 4
         MW = 35
*        Alignment shift of 3, 50 covariances to be calculated
         IS = 3
         KC = 350
         L = 80
         IFAIL = 0
*
         CALL G13CCF(NXY,MTXY,PXY,IW,MW,IS,IC,NC,CXY,CYX,KC,L,NXYG,XG,
     +               YG,NG,IFAIL)
*
```

```
            WRITE (NOUT,*)
            WRITE (NOUT,*) '                  Returned cross covariances'
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     + 'Lag      XY        YX   Lag      XY        YX   Lag      XY        YX'
            DO 20 I = 1, NC, 3
               WRITE (NOUT,99999) (II-1,CXY(II),CYX(II),II=I,MIN(I+2,NC))
   20       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,*) '                     Returned sample spectrum'
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +'         Real  Imaginary       Real  Imaginary       Real  Imaginar
     +y'
            WRITE (NOUT,*)
     +'Lag     part      part  Lag     part      part  Lag     part      part'
            DO 40 I = 1, NG, 3
               WRITE (NOUT,99999) (II-1,XG(II),YG(II),II=I,MIN(I+2,NG))
   40       CONTINUE
         END IF
         STOP
*
99999 FORMAT (1X,I3,2F9.4,I4,2F9.4,I4,2F9.4)
         END
```

9.2. Program Data

```
G13CCF Example Program Data
   296 50 0
 -0.109  0.000  0.178  0.339  0.373  0.441  0.461  0.348
  0.127 -0.180 -0.588 -1.055 -1.421 -1.520 -1.302 -0.814
 -0.475 -0.193  0.088  0.435  0.771  0.866  0.875  0.891
  0.987  1.263  1.775  1.976  1.934  1.866  1.832  1.767
  1.608  1.265  0.790  0.360  0.115  0.088  0.331  0.645
  0.960  1.409  2.670  2.834  2.812  2.483  1.929  1.485
  1.214  1.239  1.608  1.905  2.023  1.815  0.535  0.122
  0.009  0.164  0.671  1.019  1.146  1.155  1.112  1.121
  1.223  1.257  1.157  0.913  0.620  0.255 -0.280 -1.080
 -1.551 -1.799 -1.825 -1.456 -0.944 -0.570 -0.431 -0.577
 -0.960 -1.616 -1.875 -1.891 -1.746 -1.474 -1.201 -0.927
 -0.524  0.040  0.788  0.943  0.930  1.006  1.137  1.198
  1.054  0.595 -0.080 -0.314 -0.288 -0.153 -0.109 -0.187
 -0.255 -0.299 -0.007  0.254  0.330  0.102 -0.423 -1.139
 -2.275 -2.594 -2.716 -2.510 -1.790 -1.346 -1.081 -0.910
 -0.876 -0.885 -0.800 -0.544 -0.416 -0.271  0.000  0.403
  0.841  1.285  1.607  1.746  1.683  1.485  0.993  0.648
  0.577  0.577  0.632  0.747  0.999  0.993  0.968  0.790
  0.399 -0.161 -0.553 -0.603 -0.424 -0.194 -0.049  0.060
  0.161  0.301  0.517  0.566  0.560  0.573  0.592  0.671
  0.933  1.337  1.460  1.353  0.772  0.218 -0.237 -0.714
 -1.099 -1.269 -1.175 -0.676  0.033  0.556  0.643  0.484
  0.109 -0.310 -0.697 -1.047 -1.218 -1.183 -0.873 -0.336
  0.063  0.084  0.000  0.001  0.209  0.556  0.782  0.858
  0.918  0.862  0.416 -0.336 -0.959 -1.813 -2.378 -2.499
 -2.473 -2.330 -2.053 -1.739 -1.261 -0.569 -0.137 -0.024
 -0.050 -0.135 -0.276 -0.534 -0.871 -1.243 -1.439 -1.422
 -1.175 -0.813 -0.634 -0.582 -0.625 -0.713 -0.848 -1.039
 -1.346 -1.628 -1.619 -1.149 -0.488 -0.160 -0.007 -0.092
 -0.620 -1.086 -1.525 -1.858 -2.029 -2.024 -1.961 -1.952
 -1.794 -1.302 -1.030 -0.918 -0.798 -0.867 -1.047 -1.123
 -0.876 -0.395  0.185  0.662  0.709  0.605  0.501  0.603
  0.943  1.223  1.249  0.824  0.102  0.025  0.382  0.922
  1.032  0.866  0.527  0.093 -0.458 -0.748 -0.947 -1.029
 -0.928 -0.645 -0.424 -0.276 -0.158 -0.033  0.102  0.251
  0.280  0.000 -0.493 -0.759 -0.824 -0.740 -0.528 -0.204
  0.034  0.204  0.253  0.195  0.131  0.017 -0.182 -0.262
 53.8 53.6 53.5 53.5 53.4 53.1 52.7 52.4 52.2 52.0 52.0 52.4 53.0 54.0 54.9 56.0
 56.8 56.8 56.4 55.7 55.0 54.3 53.2 52.3 51.6 51.2 50.8 50.5 50.0 49.2 48.4 47.9
 47.6 47.5 47.5 47.6 48.1 49.0 50.0 51.1 51.8 51.9 51.7 51.2 50.0 48.3 47.0 45.8
 45.6 46.0 46.9 47.8 48.2 48.3 47.9 47.2 47.2 48.1 49.4 50.6 51.5 51.6 51.2 50.5
 50.1 49.8 49.6 49.4 49.3 49.2 49.3 49.7 50.3 51.3 52.8 54.4 56.0 56.9 57.5 57.3
 56.6 56.0 55.4 55.4 56.4 57.2 58.0 58.4 58.4 58.1 57.7 57.0 56.0 54.7 53.2 52.1
 51.6 51.0 50.5 50.4 51.0 51.8 52.4 53.0 53.4 53.6 53.7 53.8 53.8 53.8 53.3 53.0
 52.9 53.4 54.6 56.4 58.0 59.4 60.2 60.0 59.4 58.4 57.6 56.9 56.4 56.0 55.7 55.3
```

```
55.0 54.4 53.7 52.8 51.6 50.6 49.4 48.8 48.5 48.7 49.2 49.8 50.4 50.7 50.9 50.7
50.5 50.4 50.2 50.4 51.2 52.3 53.2 53.9 54.1 54.0 53.6 53.2 53.0 52.8 52.3 51.9
51.6 51.6 51.4 51.2 50.7 50.0 49.4 49.3 49.7 50.6 51.8 53.0 54.0 55.3 55.9 55.9
54.6 53.5 52.4 52.1 52.3 53.0 53.8 54.6 55.4 55.9 55.9 55.2 54.4 53.7 53.6 53.6
53.2 52.5 52.0 51.4 51.0 50.9 52.4 53.5 55.6 58.0 59.5 60.0 60.4 60.5 60.2 59.7
59.0 57.6 56.4 55.2 54.5 54.1 54.1 54.4 55.5 56.2 57.0 57.3 57.4 57.0 56.4 55.9
55.5 55.3 55.2 55.4 56.0 56.5 57.1 57.3 56.8 55.6 55.0 54.1 54.3 55.3 56.4 57.2
57.8 58.3 58.6 58.8 58.8 58.6 58.0 57.4 57.0 56.4 56.3 56.4 56.4 56.0 55.2 54.0
53.0 52.0 51.6 51.6 51.1 50.4 50.0 50.0 52.0 54.0 55.1 54.5 52.8 51.4 50.8 51.2
52.0 52.8 53.8 54.5 54.9 54.9 54.8 54.4 53.7 53.3 52.8 52.6 52.6 53.0 54.3 56.0
57.0 58.0 58.6 58.5 58.3 57.8 57.3 57.0
```

9.3. Program Results

```
G13CCF Example Program Results

                    Returned cross covariances

Lag      XY        YX     Lag      XY        YX     Lag      XY        YX
  0  -1.6700   -1.6700     1  -2.0581   -1.3606     2  -2.4859   -1.1383
  3  -2.8793   -0.9926     4  -3.1473   -0.9009     5  -3.2239   -0.8382
  6  -3.0929   -0.7804     7  -2.7974   -0.7074     8  -2.4145   -0.6147
  9  -2.0237   -0.5080    10  -1.6802   -0.4032    11  -1.4065   -0.3159
 12  -1.2049   -0.2554    13  -1.0655   -0.2250    14  -0.9726   -0.2238
 15  -0.9117   -0.2454    16  -0.8658   -0.2784    17  -0.8180   -0.3081
 18  -0.7563   -0.3257    19  -0.6750   -0.3315    20  -0.5754   -0.3321
 21  -0.4701   -0.3308    22  -0.3738   -0.3312    23  -0.3023   -0.3332
 24  -0.2665   -0.3384    25  -0.2645   -0.3506    26  -0.2847   -0.3727
 27  -0.3103   -0.3992    28  -0.3263   -0.4152    29  -0.3271   -0.4044
 30  -0.3119   -0.3621    31  -0.2837   -0.2919    32  -0.2568   -0.2054
 33  -0.2427   -0.1185    34  -0.2490   -0.0414    35  -0.2774    0.0227
 36  -0.3218    0.0697    37  -0.3705    0.1039    38  -0.4083    0.1356
 39  -0.4197    0.1805    40  -0.3920    0.2460    41  -0.3241    0.3319
 42  -0.2273    0.4325    43  -0.1216    0.5331    44  -0.0245    0.6199
 45   0.0528    0.6875    46   0.1074    0.7329    47   0.1448    0.7550
 48   0.1713    0.7544    49   0.1943    0.7349

                       Returned sample spectrum

        Real  Imaginary           Real  Imaginary           Real  Imaginary
Lag     part       part  Lag      part       part  Lag      part       part
  0  -6.5500     0.0000    1  -5.4267    -1.9842    2  -3.1323    -2.7307
  3  -1.2649    -2.3998    4  -0.2102    -1.7520    5   0.3411    -1.1903
  6   0.6063    -0.7420    7   0.6178    -0.3586    8   0.4391    -0.1008
  9   0.2422     0.0061   10   0.1233     0.0409   11   0.0574     0.0529
 12   0.0174     0.0452   13  -0.0008     0.0289   14  -0.0058     0.0161
 15  -0.0051     0.0084   16  -0.0027     0.0040   17  -0.0010     0.0015
 18  -0.0006     0.0006   19  -0.0005     0.0003   20  -0.0003     0.0003
 21  -0.0003     0.0004   22  -0.0003     0.0003   23  -0.0003     0.0002
 24  -0.0004     0.0001   25  -0.0004     0.0000   26  -0.0003    -0.0001
 27  -0.0002    -0.0001   28  -0.0001     0.0001   29  -0.0002     0.0003
 30  -0.0003     0.0002   31  -0.0002     0.0001   32  -0.0001     0.0000
 33   0.0000     0.0000   34   0.0001    -0.0001   35   0.0001    -0.0002
 36   0.0001    -0.0001   37   0.0001    -0.0001   38   0.0001    -0.0001
 39   0.0001    -0.0001   40   0.0001     0.0000
```

G13CDF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13CDF calculates the smoothed sample cross spectrum of a bivariate time series using spectral smoothing by the trapezium frequency (Daniell) window.

2. Specification

```
      SUBROUTINE G13CDF (NXY, MTXY, PXY, MW, IS, PW, L, KC, XG, YG, NG,
     1                   IFAIL)
      INTEGER            NXY, MTXY, MW, IS, L, KC, NG, IFAIL
      real               PXY, PW, XG(KC), YG(KC)
```

3. Description

The supplied time series may be mean and trend corrected and tapered as in the description of G13CBF before calculation of the unsmoothed sample cross-spectrum

$$f_{xy}^{*}(\omega) = \frac{1}{2\pi n}\left\{\sum_{t=1}^{n} y_t \exp(i\omega t)\right\}\times\left\{\sum_{t=1}^{n} x_t \exp(-i\omega t)\right\}$$

for frequency values $\omega_j = \dfrac{2\pi j}{K}$, $\quad 0 \le \omega_j \le \pi$.

A correction is made for bias due to any tapering.

As in the descripton of G13CBF for univariate frequency window smoothing, the smoothed spectrum is returned at a subset of these frequencies,

$$\nu_l = \frac{2\pi l}{L}, \quad l = 0,1,\ldots,[L/2]$$

where [ ] denotes the integer part.

Its real part or co-spectrum $cf(\nu_l)$, and imaginary part or quadrature spectrum $qf(\nu_l)$ are defined by

$$f_{xy}(\nu_l) = cf(\nu_l) + iqf(\nu_l) = \sum_{|\omega_k|<\frac{\pi}{M}} \tilde{w}_k f_{xy}^{*}(\nu_l+\omega_k)$$

where the weights $\tilde{w}_k$ are similar to the weights w_k defined for G13CBF, but allow for an implicit alignment shift S between the series:

$$\tilde{w}_k = w_k \exp(-2\pi i S k/L).$$

It is recommended that S is chosen as the lag k at which the cross covariances $c_{xy}(k)$ peak, so as to minimize bias.

If no smoothing is required, the integer M which determines the frequency window width $\dfrac{2\pi}{M}$, should be set to n.

The bandwidth of the estimates will normally have been calculated in a previous call of G13CBF for estimating the univariate spectra of y_t and x_t.

4. References

[1] JENKINS, G.M. and WATTS, D.G.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[2] BLOOMFIELD, P.
Fourier Analysis of Time Series: An Introduction.
Wiley, 1976.

5. Parameters

1: NXY – INTEGER. *Input*

On entry: the length of the time series x and y, n.

Constraint: NXY $\geq$ 1.

2: MTXY – INTEGER. *Input*

On entry: whether the data is to be initially mean or trend corrected.

MTXY = 0 for no correction,
MTXY = 1 for mean correction,
MTXY = 2 for trend correction.

Constraint: 0 $\leq$ MTXY $\leq$ 2.

3: PXY – ***real***. *Input*

On entry: the proportion of the data (totalled over both ends) to be initially tapered by the split cosine bell taper.

A value of 0.0 implies no tapering.

Constraint: 0.0 $\leq$ PXY $\leq$ 1.0.

4: MW – INTEGER. *Input*

On entry: the frequency width, M, of the smoothing window as $\frac{2\pi}{M}$.

A value of n implies that no smoothing is to be carried out.

Constraint: 1 $\leq$ MW $\leq$ NXY.

5: IS – INTEGER. *Input*

On entry: the alignment shift, S, between the x and y series. If x leads y, the shift is positive.

Constraint: –L < IS < L.

6: PW – ***real***. *Input*

On entry: the shape parameter, p, of the trapezium frequency window.

A value of 0.0 gives a triangular window, and a value of 1.0 a rectangular window.

If MW = NXY (i.e. no smoothing is carried out) then PW is not used.

Constraint: 0.0 $\leq$ PW $\leq$ 1.0 if MX $\neq$ NXY.

7: L – INTEGER. *Input*

On entry: the frequency division, L, of smoothed cross spectral estimates as $\frac{2\pi}{L}$.

Constraints: L $\geq$ 1.
L must be a factor of KC (see below).

8: KC – INTEGER. *Input*

On entry: the order of the fast Fourier transform (FFT) used to calculate the spectral estimates. KC should be a product of small primes such as 2^m where m is the smallest integer such that $2^m \geq 2n$, provided $m \leq 20$.

Constraints: KC $\geq$ 2×NXY.

KC must be a multiple of L. The largest prime factor of KC must not exceed 19, and the total number of prime factors of KC, counting repetitions, must not exceed 20. These two restrictions are imposed by C06EAF and C06EBF which perform the FFT.

9: XG(KC) – ***real*** array. ***Input/Output***

On entry: the NXY data points of the x series.

On exit: the real parts of the NG cross spectral estimates in elements XG(1) to XG(NG), and XG(NG+1) to XG(KC) contain 0.0. The y series leads the x series.

10: YG(KC) – ***real*** array. ***Input/Output***

On entry: the NXY data points of the y series.

On exit: the imaginary parts of the NG cross spectral estimates in elements YG(1) to YG(NG), and YG(NG+1) to YG(KC) contain 0.0. The y series leads the x series.

11: NG – INTEGER. ***Output***

On exit: the number of spectral estimates, $[L/2] + 1$, whose separate parts are held in XG and YG.

12: IFAIL – INTEGER. ***Input/Output***

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry,	NXY < 1,
or	MTXY < 0,
or	MTXY > 2,
or	PXY < 0.0,
or	PXY > 1.0,
or	MW < 1,
or	MW > NXY,
or	PW < 0.0 and MW ≠ NXY,
or	PW > 1.0 and MW ≠ NXY,
or	L < 1,
or	|IS| ≥ L.

IFAIL = 2

On entry,	KC < 2×NXY,
or	KC is not a multiple of L,
or	KC has a prime factor exceeding 19,
or	KC has more than 20 prime factors, counting repetitions.

IFAIL = 3

This indicates that a serious error has occurred. Check all array subscripts in calls to G13CDF. Seek expert help.

7. Accuracy

The FFT is a numerically stable process, and any errors introduced during the computation will normally be insignificant compared with uncertainty in the data.

8. Further Comments

G13CDF carries out an FFT of length KC to calculate the sample cross spectrum. The time taken by the routine for this is approximately proportional to KC×log(KC) (but see routine document C06EAF for further details).

9. Example

The example program reads 2 time series of length 296. It selects mean correction and a 10% tapering proportion. It selects a $\frac{2\pi}{16}$ frequency width of smoothing window, a window shape parameter of 0.5 and an alignment shift of 3. It then calls G13CDF to calculate the smoothed sample cross spectrum and prints the results.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13CDF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NXYMAX, L, KC
      PARAMETER        (NXYMAX=300,L=80,KC=8*L)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             PW, PXY
      INTEGER          I, IFAIL, IS, J, MTXY, MW, NG, NXY
*     .. Local Arrays ..
      real             XG(KC), YG(KC)
*     .. External Subroutines ..
      EXTERNAL         G13CDF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13CDF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NXY
      IF (NXY.GT.0 .AND. NXY.LE.NXYMAX) THEN
         READ (NIN,*) (XG(I),I=1,NXY)
         READ (NIN,*) (YG(I),I=1,NXY)
*        Set parameters for call to G13CDF
*        Mean correction and 10 percent taper
         MTXY = 1
         PXY = 0.1e0
*        Window shape parameter and zero covariance at lag 16
         PW = 0.5e0
         MW = 16
*        Alignment shift of 3
         IS = 3
         IFAIL = 0
*
         CALL G13CDF(NXY,MTXY,PXY,MW,IS,PW,L,KC,XG,YG,NG,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) '                      Returned sample spectrum'
         WRITE (NOUT,*)
         WRITE (NOUT,*)
     +'       Real  Imaginary      Real  Imaginary      Real  Imaginary
     +'
         WRITE (NOUT,*)
     + '        part      part         part      part         part      part'
         WRITE (NOUT,99999) (J,XG(J),YG(J),J=1,NG)
      END IF
      STOP
*
99999 FORMAT (1X,I3,F8.4,F9.4,I5,F8.4,F9.4,I5,F8.4,F9.4)
      END
```

9.2. Program Data

```
G13CDF Example Program Data
  296
-0.109  0.000  0.178  0.339  0.373  0.441  0.461  0.348
 0.127 -0.180 -0.588 -1.055 -1.421 -1.520 -1.302 -0.814
-0.475 -0.193  0.088  0.435  0.771  0.866  0.875  0.891
 0.987  1.263  1.775  1.976  1.934  1.866  1.832  1.767
 1.608  1.265  0.790  0.360  0.115  0.088  0.331  0.645
 0.960  1.409  2.670  2.834  2.812  2.483  1.929  1.485
 1.214  1.239  1.608  1.905  2.023  1.815  0.535  0.122
 0.009  0.164  0.671  1.019  1.146  1.155  1.112  1.121
 1.223  1.257  1.157  0.913  0.620  0.255 -0.280 -1.080
-1.551 -1.799 -1.825 -1.456 -0.944 -0.570 -0.431 -0.577
-0.960 -1.616 -1.875 -1.891 -1.746 -1.474 -1.201 -0.927
-0.524  0.040  0.788  0.943  0.930  1.006  1.137  1.198
 1.054  0.595 -0.080 -0.314 -0.288 -0.153 -0.109 -0.187
-0.255 -0.299 -0.007  0.254  0.330  0.102 -0.423 -1.139
-2.275 -2.594 -2.716 -2.510 -1.790 -1.346 -1.081 -0.910
-0.876 -0.885 -0.800 -0.544 -0.416 -0.271  0.000  0.403
 0.841  1.285  1.607  1.746  1.683  1.485  0.993  0.648
 0.577  0.577  0.632  0.747  0.999  0.993  0.968  0.790
 0.399 -0.161 -0.553 -0.603 -0.424 -0.194 -0.049  0.060
 0.161  0.301  0.517  0.566  0.560  0.573  0.592  0.671
 0.933  1.337  1.460  1.353  0.772  0.218 -0.237 -0.714
-1.099 -1.269 -1.175 -0.676  0.033  0.556  0.643  0.484
 0.109 -0.310 -0.697 -1.047 -1.218 -1.183 -0.873 -0.336
 0.063  0.084  0.000  0.001  0.209  0.556  0.782  0.858
 0.918  0.862  0.416 -0.336 -0.959 -1.813 -2.378 -2.499
-2.473 -2.330 -2.053 -1.739 -1.261 -0.569 -0.137 -0.024
-0.050 -0.135 -0.276 -0.534 -0.871 -1.243 -1.439 -1.422
-1.175 -0.813 -0.634 -0.582 -0.625 -0.713 -0.848 -1.039
-1.346 -1.628 -1.619 -1.149 -0.488 -0.160 -0.007 -0.092
-0.620 -1.086 -1.525 -1.858 -2.029 -2.024 -1.961 -1.952
-1.794 -1.302 -1.030 -0.918 -0.798 -0.867 -1.047 -1.123
-0.876 -0.395  0.185  0.662  0.709  0.605  0.501  0.603
 0.943  1.223  1.249  0.824  0.102  0.025  0.382  0.922
 1.032  0.866  0.527  0.093 -0.458 -0.748 -0.947 -1.029
-0.928 -0.645 -0.424 -0.276 -0.158 -0.033  0.102  0.251
 0.280  0.000 -0.493 -0.759 -0.824 -0.740 -0.528 -0.204
 0.034  0.204  0.253  0.195  0.131  0.017 -0.182 -0.262
53.8 53.6 53.5 53.5 53.4 53.1 52.7 52.4 52.2 52.0 52.0 52.4 53.0 54.0 54.9 56.0
56.8 56.8 56.4 55.7 55.0 54.3 53.2 52.3 51.6 51.2 50.8 50.5 50.0 49.2 48.4 47.9
47.6 47.5 47.5 47.6 48.1 49.0 50.0 51.1 51.8 51.9 51.7 51.2 50.0 48.3 47.0 45.8
45.6 46.0 46.9 47.8 48.2 48.3 47.9 47.2 47.2 48.1 49.4 50.6 51.5 51.6 51.2 50.5
50.1 49.8 49.6 49.4 49.3 49.2 49.3 49.7 50.3 51.3 52.8 54.4 56.0 56.9 57.5 57.3
56.6 56.0 55.4 55.4 56.4 57.2 58.0 58.4 58.4 58.1 57.7 57.0 56.0 54.7 53.2 52.1
51.6 51.0 50.5 50.4 51.0 51.8 52.4 53.0 53.4 53.6 53.7 53.8 53.8 53.8 53.3 53.0
52.9 53.4 54.6 56.4 58.0 59.4 60.2 60.0 59.4 58.4 57.6 56.9 56.4 56.0 55.7 55.3
55.0 54.4 53.7 52.8 51.6 50.6 49.4 48.8 48.5 48.7 49.2 49.8 50.4 50.7 50.9 50.7
50.5 50.4 50.2 50.4 51.2 52.3 53.2 53.9 54.1 54.0 53.6 53.2 53.0 52.8 52.3 51.9
51.6 51.6 51.4 51.2 50.7 50.0 49.4 49.3 49.7 50.6 51.8 53.0 54.0 55.3 55.9 55.9
54.6 53.5 52.4 52.1 52.3 53.0 53.8 54.6 55.4 55.9 55.9 55.2 54.4 53.7 53.6 53.6
53.2 52.5 52.0 51.4 51.0 50.9 52.4 53.5 55.6 58.0 59.5 60.0 60.4 60.5 60.2 59.7
59.0 57.6 56.4 55.2 54.5 54.1 54.1 54.4 55.5 56.2 57.0 57.3 57.4 57.0 56.4 55.9
55.5 55.3 55.2 55.4 56.0 56.5 57.1 57.3 56.8 55.6 55.0 54.1 54.3 55.3 56.4 57.2
57.8 58.3 58.6 58.8 58.8 58.6 58.0 57.4 57.0 56.4 56.3 56.4 56.4 56.0 55.2 54.0
53.0 52.0 51.6 51.6 51.1 50.4 50.0 50.0 52.0 54.0 55.1 54.5 52.8 51.4 50.8 51.2
52.0 52.8 53.8 54.5 54.9 54.9 54.8 54.4 53.7 53.3 52.8 52.6 52.6 53.0 54.3 56.0
57.0 58.0 58.6 58.5 58.3 57.8 57.3 57.0
```

9.3. Program Results

```
G13CDF Example Program Results

                          Returned sample spectrum

        Real  Imaginary         Real  Imaginary         Real  Imaginary
        part    part            part    part            part    part
  1 -6.1563   0.0000     2 -5.5905  -2.0119     3 -3.2711  -2.7963
  4 -1.1803  -2.3264     5 -0.2061  -1.8132     6  0.3434  -1.1357
  7  0.6200  -0.7351     8  0.5967  -0.3449     9  0.4523  -0.0984
 10  0.2391   0.0177    11  0.1129   0.0402    12  0.0564   0.0523
 13  0.0134   0.0443    14 -0.0032   0.0299    15 -0.0057   0.0148
 16 -0.0057   0.0069    17 -0.0033   0.0038    18 -0.0011   0.0012
 19 -0.0004   0.0001    20 -0.0004   0.0002    21 -0.0003   0.0001
 22 -0.0001   0.0002    23 -0.0002   0.0003    24 -0.0002   0.0002
 25 -0.0002   0.0000    26 -0.0004   0.0000    27 -0.0002  -0.0002
 28 -0.0001   0.0000    29 -0.0001   0.0002    30 -0.0001   0.0002
 31 -0.0002   0.0003    32 -0.0002   0.0001    33 -0.0001   0.0000
 34  0.0000   0.0000    35  0.0000  -0.0001    36  0.0001  -0.0001
 37  0.0001  -0.0001    38  0.0001  -0.0001    39  0.0000  -0.0001
 40  0.0000  -0.0001    41  0.0001   0.0000
```

G13CEF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

For a bivariate time series, G13CEF calculates the cross amplitude spectrum and squared coherency, together with lower and upper bounds from the univariate and bivariate (cross) spectra.

2. Specification

```
      SUBROUTINE G13CEF (XG, YG, XYRG, XYIG, NG, STATS, CA, CALW, CAUP, T,
     1                   SC, SCLW, SCUP, IFAIL)
      INTEGER            NG, IFAIL
      real               XG(NG), YG(NG), XYRG(NG), XYIG(NG), STATS(4),
     1                   CA(NG), CALW(NG), CAUP(NG), T, SC(NG), SCLW(NG),
     2                   SCUP(NG)
```

3. Description

Estimates of the cross amplitude spectrum $A(\omega)$ and squared coherency $W(\omega)$ are calculated for each frequency ω as

$$A(\omega) = |f_{xy}(\omega)| = \sqrt{cf(\omega)^2+qf(\omega)^2} \text{ and}$$

$$W(\omega) = \frac{|f_{xy}(\omega)|^2}{f_{xx}(\omega)f_{yy}(\omega)}$$

where:

$cf(\omega)$ and $qf(\omega)$ are the co-spectrum and quadrature spectrum estimates between the series, i.e. the real and imaginary parts of the cross spectrum $f_{xy}(\omega)$ as obtained using G13CCF or G13CDF.

$f_{xx}(\omega)$ and $f_{yy}(\omega)$ are the univariate spectrum estimates for the two series as obtained using G13CAF or G13CBF.

The same type and amount of smoothing should be used for these estimates, and this is specified by the degrees of freedom and bandwidth values which are passed from the calls of G13CAF or G13CBF.

Upper and lower 95% confidence limits for the cross amplitude are given approximately by

$$A(\omega)\left[1\pm(1.96/\sqrt{d})\sqrt{W(\omega)^{-1}+1}\right],$$

except that a negative lower limit is reset to 0.0, in which case the approximation is rather poor. The user is therefore particularly recommended to compare the coherency estimate $W(\omega)$ with the critical value T derived from the upper 5% point of the F-distribution on $(2,\ d-2)$ degrees of freedom:

$$T = \frac{2F}{d-2+2F}$$

where d is the degrees of freedom associated with the univariate spectrum estimates. The value of T is returned by the routine.

The hypothesis that the series are unrelated at frequency ω, i.e. that both the true cross amplitude and coherency are zero, may be rejected at the 5% level if $W(\omega) > T$. Tests at two frequencies separated by more than the bandwidth may be taken to be independent.

The confidence limits on $A(\omega)$ are strictly appropriate only at frequencies for which the coherency is significant. The same applies to the confidence limits on $W(\omega)$ which are however calculated at all frequencies using the approximation that $\operatorname{arctanh}\left(\sqrt{W(l)}\right)$ is Normal with variance $1/d$.

4. References

[1] JENKINS, G.M. and WATTS, D.G.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[2] BLOOMFIELD, P.
Fourier Analysis of Time Series: An Introduction.
Wiley, 1976.

5. Parameters

1: XG(NG) – ***real*** array. *Input*

On entry: the NG univariate spectral estimates, $f_{xx}(w)$, for the x series.

2: YG(NG) – ***real*** array. *Input*

On entry: the NG univariate spectral estimates, $f_{yy}(w)$, for the y series.

3: XYRG(NG) – ***real*** array. *Input*

On entry: the real parts, $cf(w)$, of the NG bivariate spectral estimates for the x and y series. The x series leads the y series.

4: XYIG(NG) – ***real*** array. *Input*

On entry: the imaginary parts, $qf(w)$, of the NG bivariate spectral estimates for the x and y series. The x series leads the y series.

Note: the two univariate and the bivariate spectra must each have been calculated using the same method of smoothing. For rectangular, Bartlett, Tukey or Parzen smoothing windows, the same cut off point of lag window and the same frequency division of the spectral estimates must be used. For the trapezium frequency smoothing window, the frequency width and the shape of the window and the frequency division of the spectral estimates must be the same. The spectral estimates and statistics must also be unlogged.

5: NG – INTEGER. *Input*

On entry: the number of spectral estimates in each of the arrays XG, YG, XYRG and XYIG. It is also the number of cross amplitude spectral and squared coherency estimates.

Constraint: NG $\geq$ 1.

6: STATS(4) – ***real*** array. *Input*

On entry: the 4 associated statistics for the univariate spectral estimates for the x and y series. STATS(1) contains the degrees of freedom, STATS(2) and STATS(3) contain the lower and upper bound multiplying factors respectively and STATS(4) contains the bandwidth.

Constraints: STATS(1) $\geq$ 3.0,
$0.0 <$ STATS(2) ≤ 1.0,
STATS(3) $\geq$ 1.0.

7: CA(NG) – ***real*** array. *Output*

On exit: the NG cross amplitude spectral estimates $\hat{A}(w)$ at each frquency of w.

8: CALW(NG) – ***real*** array. *Output*

On exit: the NG lower bounds for the NG cross amplitude spectral estimates.

9: CAUP(NG) – ***real*** array. *Output*

On exit: the NG upper bounds for the NG cross amplitude spectral estimates.

10: T – ***real.*** *Output*

On exit: the critical value for the significance of the squared coherency, T.

11: SC(NG) – ***real*** array. *Output*

On exit: the NG squared coherency estimates, $\hat{W}(w)$ at each frequency w.

12: SCLW(NG) – ***real*** array. *Output*

On exit: the NG lower bounds for the NG squared coherency estimates.

13: SCUP(NG) – ***real*** array. *Output*

On exit: the NG upper bounds for the NG squared coherency estimates.

14: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NG < 1,
or STATS(1) < 3.0,
or STATS(2) ≤ 0.0,
or STATS(2) > 1.0,
or STATS(3) < 1.0.

IFAIL = 2

A bivariate spectral estimate is zero. For this frequency the cross amplitude spectrum and squared coherency and their bounds are set to zero.

IFAIL = 3

A univariate spectral estimate is negative. For this frequency the cross amplitude spectrum and squared coherency and their bounds are set to zero.

IFAIL = 4

A univariate spectral estimate is zero. For this frequency the cross amplitude spectrum and squared coherency and their bounds are set to zero.

IFAIL = 5

A calculated value of the squared coherency exceeds 1.0. For this frequency the squared coherency is reset to 1.0 and this value for the squared coherency is used in the formulae for the calculation of bounds for both the cross amplitude spectrum and squared coherency. This has the consequence that both squared coherency bounds are 1.0.

If more than one failure of the types 2,3,4 and 5 occurs then the failure type which occurred at lowest frequency is returned in IFAIL. However the actions indicated above are also carried out for failures at higher frequencies.

7. Accuracy

All computations are very stable and yield good accuracy.

8. Further Comments

The time taken by the routine is approximately proportional to NG.

9. Example

The example program reads the set of univariate spectrum statistics, the 2 univariate spectra and the cross spectrum at a frequency division of $\frac{2\pi}{20}$ for a pair of time series. It calls G13CEF to calculate the cross amplitude spectrum and squared coherency and their bounds and prints the results.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13CEF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NGMAX
      PARAMETER        (NGMAX=9)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             T
      INTEGER          I, IFAIL, J, NG
*     .. Local Arrays ..
      real             CA(NGMAX), CALW(NGMAX), CAUP(NGMAX), SC(NGMAX),
     +                 SCLW(NGMAX), SCUP(NGMAX), STATS(4), XG(NGMAX),
     +                 XYIG(NGMAX), XYRG(NGMAX), YG(NGMAX)
*     .. External Subroutines ..
      EXTERNAL         G13CEF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13CEF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NG
      READ (NIN,*) (STATS(I),I=1,4)
      READ (NIN,*) (XG(I),YG(I),XYRG(I),XYIG(I),I=1,NG)
      IFAIL = 1
*
      CALL G13CEF(XG,YG,XYRG,XYIG,NG,STATS,CA,CALW,CAUP,T,SC,SCLW,SCUP,
     +            IFAIL)
*
      WRITE (NOUT,*)
      IF (IFAIL.NE.0) THEN
         WRITE (NOUT,99999) 'G13CEF fails. IFAIL =', IFAIL
         WRITE (NOUT,*)
      END IF
      IF (IFAIL.NE.1) THEN
         WRITE (NOUT,*) '      Cross amplitude spectrum'
         WRITE (NOUT,*)
         WRITE (NOUT,*) '                            Lower     Upper'
         WRITE (NOUT,*) '                  Value     bound     bound'
         DO 20 J = 1, NG
            WRITE (NOUT,99998) J - 1, CA(J), CALW(J), CAUP(J)
   20    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,99997) 'Squared coherency test statistic =', T
         WRITE (NOUT,*)
         WRITE (NOUT,*) '            Squared coherency'
         WRITE (NOUT,*)
         WRITE (NOUT,*) '                            Lower     Upper'
         WRITE (NOUT,*) '                  Value     bound     bound'
         DO 40 J = 1, NG
            WRITE (NOUT,99998) J - 1, SC(J), SCLW(J), SCUP(J)
   40    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,A,I3)
99998 FORMAT (1X,I5,3F10.4)
99997 FORMAT (1X,A,F12.4)
      END
```

9.2. Program Data

```
G13CEF Example Program Data
   9
 30.00000    .63858  1.78670    .33288
  2.03490  21.97712 -6.54995   0.00000
   .51554   3.29761   .34107  -1.19030
   .07640    .28782   .12335    .04087
   .01068    .02480  -.00514    .00842
   .00093    .00285  -.00033    .00032
   .00100    .00203  -.00039   -.00001
   .00076    .00125  -.00026    .00018
   .00037    .00107   .00011   -.00016
   .00021    .00191   .00007   0.00000
```

9.3. Program Results

```
 G13CEF Example Program Results

       Cross amplitude spectrum

                       Lower       Upper
           Value       bound       bound
     0    6.5499      3.9277     10.9228
     1    1.2382      0.7364      2.0820
     2    0.1299      0.0755      0.2236
     3    0.0099      0.0049      0.0197
     4    0.0005      0.0001      0.0017
     5    0.0004      0.0001      0.0015
     6    0.0003      0.0001      0.0010
     7    0.0002      0.0001      0.0007
     8    0.0001      0.0000      0.0018

 Squared coherency test statistic =       0.1926

          Squared coherency

                       Lower       Upper
           Value       bound       bound
     0    0.9593      0.9185      0.9799
     1    0.9018      0.8093      0.9507
     2    0.7679      0.5811      0.8790
     3    0.3674      0.1102      0.6177
     4    0.0797      0.0000      0.3253
     5    0.0750      0.0000      0.3182
     6    0.1053      0.0000      0.3610
     7    0.0952      0.0000      0.3475
     8    0.0122      0.0000      0.1912
```

G13CFF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

For a bivariate time series, G13CFF calculates the gain and phase together with lower and upper bounds from the univariate and bivariate spectra.

2. Specification

```
SUBROUTINE G13CFF (XG, YG, XYRG, XYIG, NG, STATS, GN, GNLW, GNUP,
1                  PH, PHLW, PHUP, IFAIL)
INTEGER          NG, IFAIL
real             XG(NG), YG(NG), XYRG(NG), XYIG(NG), STATS(4),
1                GN(NG), GNLW(NG), GNUP(NG), PH(NG), PHLW(NG),
2                PHUP(NG)
```

3. Description

Estimates of the gain $G(\omega)$ and phase $\phi(\omega)$ of the dependency of series y on series x at frequency ω are given by

$$\hat{G}(\omega) = \frac{A(\omega)}{f_{xx}(\omega)}$$

$$\hat{\phi}(\omega) = \cos^{-1}\left(\frac{cf(\omega)}{A(\omega)}\right), \qquad \text{if } qf(\omega) \geq 0$$

$$\hat{\phi}(\omega) = 2\pi - \cos^{-1}\left(\frac{cf(\omega)}{A(\omega)}\right), \qquad \text{if } qf(\omega) < 0.$$

The quantities used in these definitions are obtained as in Section 3 of G13CEF.

Confidence limits are returned for both gain and phase, but should again be taken as very approximate when the coherency $W(\omega)$, as calculated by G13CEF, is not significant. These are based on the assumption that both $(\hat{G}(\omega)/G(\omega))-1$ and $\hat{\phi}(\omega)$ are Normal with variance

$$\frac{1}{d}\left(\frac{1}{W(\omega)}-1\right).$$

Although the estimate of $\phi(\omega)$ is always given in the range $[0,2\pi)$, no attempt is made to restrict its confidence limits to this range.

4. References

[1] JENKINS, G.M. and WATTS, D.G.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[2] BLOOMFIELD, P.
Fourier Analysis of Time Series: An Introduction.
Wiley, 1976.

5. Parameters

1: XG(NG) – ***real*** array. *Input*

On entry: the NG univariate spectral estimates, $f_{xx}(w)$, for the x series.

2: YG(NG) – ***real*** array. *Input*

On entry: the NG univariate spectral estimates, $f_{yy}(w)$, for the y series.

3: XYRG(NG) – ***real*** array. *Input*

On entry: the real parts, $cf(w)$ of the NG bivariate spectral estimates for the x and y series. The x series leads the y series.

4: XYIG(NG) – ***real*** array. *Input*

On entry: the imaginary parts, $qf(w)$, of the NG bivariate spectral estimates for the x and y series. The x series leads the y series.

Note: the two univariate and the bivariate spectra must each have been calculated using the same method of smoothing. For rectangular, Bartlett, Tukey or Parzen smoothing windows, the same cut off point of lag window and the same frequency division of the spectral estimates must be used. For the trapezium frequency smoothing window, the frequency width and the shape of the window and the frequency division of the spectral estimates must be the same. The spectral estimates and statistics must also be unlogged.

5: NG – INTEGER. *Input*

On entry: the number of spectral estimates in each of the arrays XG, YG, XYRG and XYIG. It is also the number of gain and phase estimates.

Constraint: NG $\geq$ 1.

6: STATS(4) – ***real*** array. *Input*

On entry: the 4 associated statistics for the univariate spectral estimates for the x and y series. STATS(1) contains the degrees of freedom, STATS(2) and STATS(3) contain the lower and upper bound multiplying factors respectively and STATS(4) holds the bandwidth.

Constraint: STATS(1) $\geq$ 3.0.

7: GN(NG) – ***real*** array. *Output*

On exit: the NG gain estimates, $\hat{G}(w)$, at each frequency w.

8: GNLW(NG) – ***real*** array. *Output*

On exit: the NG lower bounds for the NG gain estimates.

9: GNUP(NG) – ***real*** array. *Output*

On exit: the NG upper bounds for the NG gain estimates.

10: PH(NG) – ***real*** array. *Output*

On exit: the NG phase estimates, $\hat{\phi}(w)$, at each fequency w.

11: PHLW(NG) – ***real*** array. *Output*

On exit: the NG lower bounds for the NG phase estimates.

12: PHUP(NG) – ***real*** array. *Output*

On exit: the NG upper bounds for the NG phase estimates.

13: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NG < 1,
or STATS(1) < 3.0.

IFAIL = 2

A bivariate spectral estimate is zero. For this frequency the gain and the phase and their bounds are set to zero.

IFAIL = 3

A univariate spectral estimate is negative. For this frequency the gain and the phase and their bounds are set to zero.

IFAIL = 4

A univariate spectral estimate is zero. For this frequency the gain and the phase and their bounds are set to zero.

IFAIL = 5

A calculated value of the squared coherency exceeds 1.0. For this frequency the squared coherency is reset to 1.0 in the formulae for the gain and phase bounds.

If more than one failure of types 2, 3, 4 and 5 occurs then the failure type which occurred at lowest frequency is returned in IFAIL. However the actions indicated above are also carried out for failures at higher frequencies.

7. Accuracy

All computations are very stable and yield good accuracy.

8. Further Comments

The time taken by the routine is approximately proportional to NG.

9. Example

The example program reads the set of univariate spectrum statistics, the 2 univariate spectra and the cross spectrum at a frequency division of $\frac{2\pi}{20}$ for a pair of time series. It calls G13CFF to calculate the gain and the phase and their bounds and prints the results.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13CFF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NGMAX
      PARAMETER        (NGMAX=9)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, NG
*     .. Local Arrays ..
      real             GN(NGMAX), GNLW(NGMAX), GNUP(NGMAX), PH(NGMAX),
     +                 PHLW(NGMAX), PHUP(NGMAX), STATS(4), XG(NGMAX),
     +                 XYIG(NGMAX), XYRG(NGMAX), YG(NGMAX)
*     .. External Subroutines ..
      EXTERNAL         G13CFF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13CFF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NG
      IF (NG.GT.0 .AND. NG.LE.NGMAX) THEN
         READ (NIN,*) (STATS(I),I=1,4)
         READ (NIN,*) (XG(I),YG(I),XYRG(I),XYIG(I),I=1,NG)
         IFAIL = 1
```

```
*
         CALL G13CFF(XG,YG,XYRG,XYIG,NG,STATS,GN,GNLW,GNUP,PH,PHLW,PHUP,
     +               IFAIL)
*
         WRITE (NOUT,*)
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99999) 'G13CFF fails. IFAIL =', IFAIL
            WRITE (NOUT,*)
         END IF
         IF (IFAIL.NE.1) THEN
            WRITE (NOUT,*) '                The gain'
            WRITE (NOUT,*)
            WRITE (NOUT,*) '                      Lower     Upper'
            WRITE (NOUT,*) '            Value     bound     bound'
            DO 20 J = 1, NG
               WRITE (NOUT,99998) J - 1, GN(J), GNLW(J), GNUP(J)
   20       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,*) '               The phase'
            WRITE (NOUT,*)
            WRITE (NOUT,*) '                      Lower     Upper'
            WRITE (NOUT,*) '            Value     bound     bound'
            DO 40 J = 1, NG
               WRITE (NOUT,99998) J - 1, PH(J), PHLW(J), PHUP(J)
   40       CONTINUE
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I3)
99998 FORMAT (1X,I5,3F10.4)
      END
```

9.2. Program Data

```
G13CFF Example Program Data
     9
 30.00000    .63858  1.78670    .33288
  2.03490  21.97712 -6.54995   0.00000
   .51554   3.29761   .34107  -1.19030
   .07640    .28782   .12335    .04087
   .01068    .02480  -.00514    .00842
   .00093    .00285  -.00033    .00032
   .00100    .00203  -.00039   -.00001
   .00076    .00125  -.00026    .00018
   .00037    .00107   .00011   -.00016
   .00021    .00191   .00007   0.00000
```

9.3. Program Results

```
 G13CFF Example Program Results

                The gain

                      Lower     Upper
            Value     bound     bound
     0     3.2188    2.9722    3.4859
     1     2.4018    2.1138    2.7290
     2     1.7008    1.3748    2.1042
     3     0.9237    0.5558    1.5350
     4     0.4943    0.1327    1.8415
     5     0.3901    0.1002    1.5196
     6     0.4161    0.1346    1.2863
     7     0.5248    0.1591    1.7306
     8     0.3333    0.0103   10.8301
```

```
                The phase

                           Lower      Upper
             Value         bound      bound
     0      3.1416        3.0619     3.2213
     1      4.9915        4.8637     5.1192
     2      0.3199        0.1071     0.5328
     3      2.1189        1.6109     2.6268
     4      2.3716        1.0563     3.6868
     5      3.1672        1.8075     4.5270
     6      2.5360        1.4074     3.6647
     7      5.3147        4.1214     6.5079
     8      0.0000       -3.4809     3.4809
```

G13CGF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

For a bivariate time series, G13CGF calculates the noise spectrum together with multiplying factors for the bounds and the impulse response function and its standard error, from the univariate and bivariate spectra.

2. Specification

```
      SUBROUTINE G13CGF (XG, YG, XYRG, XYIG, NG, STATS, L, N, ER, ERLW,
     1                   ERUP, RF, RFSE, IFAIL)
      INTEGER          NG, L, N, IFAIL
      real             XG(NG), YG(NG), XYRG(NG), XYIG(NG), STATS(4),
     1                 ER(NG), ERLW, ERUP, RF(L), RFSE
```

3. Description

An estimate of the noise spectrum in the dependence of series y on series x at frequency ω is given by

$$f_{y|x}(\omega) = f_{yy}(\omega)(1-W(\omega))$$

where $W(\omega)$ is the squared coherency described in G13GEF and $f_{yy}(\omega)$ is the univariate spectrum estimate for series y. Confidence limits on the true spectrum are obtained using multipliers as described for G13CAF, but based on $(d-2)$ degrees of freedom.

If the dependence of y_t on x_t can be assumed to be represented in the time domain by the one sided relationship

$$y_t = v_0x_t + v_1x_{t-1} + \ldots + n_t$$

where the noise n_t is independent of x_t, then it is the spectrum of this noise which is estimated by $f_{y|x}(\omega)$.

Estimates of the impulse response function $v_0,v_1,v_2,\ldots$ may also be obtained as

$$v_k = \frac{1}{\pi}\int_0^{\pi} \mathrm{Re}\left(\frac{\exp(ik\omega)f_{xy}(\omega)}{f_{xx}(\omega)}\right)$$

where Re indicates the real part of the expression. For this purpose it is essential that the univariate spectrum for $x, f_{xx}(\omega)$, and the cross spectrum, $f_{xy}(\omega)$ be supplied to this routine for a frequency range

$$\omega_l = \left[\frac{2\pi l}{L}\right], \quad 0 \le l \le [L/2],$$

where [ ] denotes the integer part, the integral being approximated by a finite Fourier transform.

An approximate standard error is calculated for the estimates v_k. Significant values of v_k in the locations described as anticipatory responses in the parameter array RF, indicate that feedback exists from y_t to x_t. This will bias the estimates of v_k in any causal dependence of y_t on $x_t,x_{t-1},\ldots$.

4. References

[1] JENKINS, G.M. and WATTS, D.G.
Spectral Analysis and its Applications,
Holden-Day, 1968.

[2] BLOOMFIELD, P.
Fourier Analysis of Time Series: An Introduction,
Wiley, 1976.

5. Parameters

1: XG(NG) – ***real*** array. *Input*

On entry: the NG univariate spectral estimates, $f_{xx}(w)$, for the x series.

2: YG(NG) – ***real*** array. *Input*

On entry: the NG univariate spectral estimates, $f_{yy}(w)$, for the y series.

3: XYRG(NG) – ***real*** array. *Input*

On entry: the real parts, $cf(w)$, of the NG bivariate spectral estimates for the x and y series. The x series leads the y series.

4: XYIG(NG) – ***real*** array. *Input*

On entry: the imaginary parts, $qf(w)$, of the NG bivariate spectral estimates for the x and y series. The x series leads the y series.

Note: the two univariate and bivariate spectra must each have been calculated using the same method of smoothing. For rectangular, Bartlett, Tukey or Parzen smoothing windows, the same cut off point of lag window and the same frequency division of the spectral estimates must be used. For the trapezium frequency smoothing window, the frequency width and the shape of the window and the frequency division of the spectral estimates must be the same. The spectral estimates and statistics must also be unlogged.

5: NG – INTEGER. *Input*

On entry: the number of spectral estimates in each of the arrays XG, YG, XYRG, XYIG. It is also the number of noise spectral estimates.

Constraint: NG $\geq$ 1.

6: STATS(4) – ***real*** array. *Input*

On entry: the 4 associated statistics for the univariate spectral estimates for the x and y series. STATS(1) contains the degree of freedom, STATS(2) and STATS(3) contain the lower and upper bound multiplying factors respectively and STATS(4) contains the bandwidth.

Constraints: STATS(1) $\geq$ 3.0,
0.0 < STATS(2) $\leq$ 1.0,
STATS(3) $\geq$ 1.0.

7: L – INTEGER. *Input*

On entry: the frequency division, L, of the spectral estimates as $\frac{2\pi}{L}$. It is also the order of the FFT used to calculate the impulse response function. L must relate to the parameter NG by the relationship.

The largest prime factor of L must not exceed 19, and the total number of prime factors of L, counting repetitions, must not exceed 20. These two restrictions are imposed by C06EBF which performs the FFT.

Constraint: NG $= \left[\frac{\mathrm{L}}{2}\right] + 1$.

8: N – INTEGER. *Input*

On entry: the number of points in each of the time series x and y. N should have the same value as NXY in the call of G13CCF or G13CDF which calculated the smoothed sample cross spectrum. N is used in calculating the impulse response function standard error (RFSE).

Constraint: N $\geq$ 1.

9: ER(NG) – ***real*** array. *Output*

On exit: the NG estimates of the noise spectrum, $\hat{f}_{y|x}(w)$ at each frequency.

10: ERLW – ***real.*** *Output*

On exit: the noise spectrum lower limit multiplying factor.

11: ERUP – ***real.*** *Output*

On exit: the noise spectrum upper limit multiplying factor.

12: RF(L) – ***real*** array. *Output*

On exit: the impulse response function. Causal responses are stored in ascending frequency in RF(1) to RF(NG) and anticipatory responses are stored in descending frequency in RF(NG+1) to RF(L).

13: RFSE – ***real.*** *Output*

On exit: the impulse response function standard error.

14: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL ≠ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.** To suppress the output of an error message when soft failure occurs, set IFAIL to 1.

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

IFAIL = 1

On entry, NG < 1,
or STATS(1) < 3.0,
or STATS(2) ≤ 0.0,
or STATS(2) > 1.0,
or STATS(3) < 1.0,
or N < 1.

IFAIL = 2

A bivariate spectral estimate is zero. For this frequency the noise spectrum is set to zero, and the contribution to the impulse response function and its standard error is set to zero.

IFAIL = 3

A univariate spectral estimate is negative. For this frequency the noise spectrum is set to zero, and the contributions to the impulse response function and its standard error are set to zero.

IFAIL = 4

A univariate spectral estimate is zero. For this frequency the noise spectrum is set to zero and the contributions to the impulse response function and its standard error are set to zero.

IFAIL = 5

A calculated value of the squared coherency exceeds 1.0. For this frequency the squared coherency is reset to 1.0 with the consequence that the noise spectrum is zero and the contribution to the impulse response function at this frequency is zero.

IFAIL = 6

On entry, $\left[\frac{\mathrm{L}}{2}\right] + 1 \neq \mathrm{NG}$,

or L has a prime factor exceeding 19,

or L has more than 20 prime factors, counting repetitions.

If more than one failure of types 2,3,4 and 5 occurs then the failure type which occurred at lowest frequency is returned in IFAIL. However the actions indicated above are also carried out for failures at higher frequencies.

7. Accuracy

The computation of the noise is stable and yields good accuracy. The FFT is a numerically stable process, and any errors introduced during the computation will normally be insignificant compared with uncertainty in the data.

8. Further Comments

The time taken by the routine is approximately proportional to NG.

9. Example

The example program reads the set of univariate spectrum statistics, the 2 univariate spectra and the cross spectrum at a frequency division of $\frac{2\pi}{20}$ for a pair of time series. It calls G13CGF to calculate the noise spectrum and its confidence limits multiplying factors, the impulse response function and its standard error. It then prints the results.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13CGF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NGMAX, LMAX
      PARAMETER        (NGMAX=9,LMAX=16)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             ERLW, ERUP, RFSE
      INTEGER          I, IFAIL, J, L, N, NG
*     .. Local Arrays ..
      real             ER(NGMAX), RF(LMAX), STATS(4), XG(NGMAX),
     +                 XYIG(NGMAX), XYRG(NGMAX), YG(NGMAX)
*     .. External Subroutines ..
      EXTERNAL         G13CGF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13CGF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NG, L, N
      IF (NG.GT.0 .AND. NG.LE.NGMAX .AND. L.GT.0 .AND. L.LE.LMAX) THEN
         READ (NIN,*) (STATS(I),I=1,4)
         READ (NIN,*) (XG(I),YG(I),XYRG(I),XYIG(I),I=1,NG)
         IFAIL = 1
*
```

```
         CALL G13CGF(XG,YG,XYRG,XYIG,NG,STATS,L,N,ER,ERLW,ERUP,RF,RFSE,
     +               IFAIL)
*
         WRITE (NOUT,*)
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99999) 'G13CGF fails. IFAIL =', IFAIL
            WRITE (NOUT,*)
         END IF
         IF (IFAIL.NE.1) THEN
            WRITE (NOUT,*) '           Noise spectrum'
            DO 20 J = 1, NG
               WRITE (NOUT,99998) J - 1, ER(J)
   20       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Noise spectrum bounds multiplying factors'
            WRITE (NOUT,99997) 'Lower =', ERLW, '     Upper =', ERUP
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Impulse response function'
            WRITE (NOUT,*)
            DO 40 J = 1, L
               WRITE (NOUT,99998) J - 1, RF(J)
   40       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,99997)
     +        'Impulse response function standard error =', RFSE
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I3)
99998 FORMAT (1X,I5,F16.4)
99997 FORMAT (1X,A,F10.4,A,F10.4)
      END
```

9.2. Program Data

```
G13CGF Example Program Data
     9    16   296
 30.00000   .63858  1.78670   .33288
  2.03490 21.97712 -6.54995  0.00000
   .51554  3.29761   .34107 -1.19030
   .07640   .28782   .12335   .04087
   .01068   .02480  -.00514   .00842
   .00093   .00285  -.00033   .00032
   .00100   .00203  -.00039  -.00001
   .00076   .00125  -.00026   .00018
   .00037   .00107   .00011  -.00016
   .00021   .00191   .00007  0.00000
```

9.3. Program Results

```
 G13CGF Example Program Results

           Noise spectrum
     0         0.8941
     1         0.3238
     2         0.0668
     3         0.0157
     4         0.0026
     5         0.0019
     6         0.0011
     7         0.0010
     8         0.0019

 Noise spectrum bounds multiplying factors
 Lower =    0.6298     Upper =    1.8291
```

```
Impulse response function

    0         -0.0547
    1          0.0586
    2         -0.0322
    3         -0.6956
    4         -0.7181
    5         -0.8019
    6         -0.4303
    7         -0.2392
    8         -0.0766
    9          0.0657
   10         -0.1652
   11         -0.0439
   12         -0.0494
   13         -0.0384
   14          0.0838
   15         -0.0814

Impulse response function standard error =    0.0863
```

G13DBF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DBF calculates the multivariate partial autocorrelation function of a multivariate time series.

2. Specification

```
      SUBROUTINE G13DBF (C0, C, NSM, NS, NL, NK, P, V0, V, D, DB, W, WB,
     1                   NVP, WA, IWA, IFAIL)
      INTEGER            NSM, NS, NL, NK, NVP, IWA, IFAIL
      real               C0(NSM,NS), C(NSM,NSM,NL), P(NK), V0, V(NK),
     1                   D(NSM,NSM,NK), DB(NSM,NS), W(NSM,NSM,NK),
     2                   WB(NSM,NSM,NK), WA(IWA)
```

3. Description

The input is a set of lagged autocovariance matrices $C_0,C_1,C_2,...,C_K$. These will generally be sample values such as are obtained from a multivariate time series using G13DMF. If sample autocorrelation matrices are used as input, then the output will be relevant to the original series scaled by their standard deviations. If these autocorrelation matrices are produced by G13DMF, the user must replace the diagonal elements of C_0 (otherwise used to hold the series variances) by 1.

The main calculation is the recursive determination of the coefficients in the finite lag (forward) prediction equation

$$x_t = \Phi_{k,1}x_{t-1} + ... + \Phi_{k,k}x_{t-k} + e_{k,t}$$

and the associated backward prediction equation

$$x_{t-k-1} = \Psi_{k,1}x_{t-k} + ... + \Psi_{k,k}x_{t-1} + f_{k,t}$$

together with the covariance matrices D_k of $e_{k,t}$ and G_k of $f_{k,t}$.

The recursive cycle by which the order of the prediction equation is extended from k to $k+1$, is to calculate

$$M_{k+1} = C'_{k+1} - \Phi_{k,1}C'_k - ... - \Phi_{k,k}C'_1 \qquad (1)$$

then $\Phi_{k+1,k+1} = M_{k+1}D_k^{-1}$; $\Psi_{k+1,k+1} = M'_{k+1}G_k^{-1}$

from which $\Phi_{k+1,j} = \Phi_{k,j} - \Phi_{k+1,k+1}\Psi_{k,k+1-j}$ for $j = 1,2,...,k$ (2)

and $\Psi_{k+1,j} = \Psi_{k,j} - \Psi_{k+1,k+1}\Phi_{k,k+1-j}$ for $j = 1,2,...,k$. (3)

Finally $D_{k+1} = D_k - M_{k+1}\Phi'_{k+1,k+1}$; $G_{k+1} = G_k - M'_{k+1}\Psi'_{k+1,k+1}$.

(Here $'$ denotes the transpose of a matrix.)

The cycle is initialised by taking (for $k = 0$)

$$D_0 = G_0 = C_0.$$

In the step from $k = 0$ to 1, the above equations contain redundant terms and simplify. Thus (1) becomes $M_1 = C'_1$ and neither (2) or (3) are needed.

Quantities useful in assessing the effectiveness of the prediction equation are generalized variance ratios

$$v_k = \det D_k / \det C_0, \qquad k = 1,2,...$$

and multiple squared partial autocorrelations

$$p_k^2 = 1 - v_k/v_{k-1}.$$

4. References

[1] WHITTLE, P.
On the fitting of multivariate autoregressions and the approximate canonical factorization of a spectral density matrix.
Biometrika, 50, pp. 129-134, 1963.

[2] AKAIKE, H.
Autoregressive Model Fitting for Control.
Ann. Inst. Statist. Math., 23, pp. 163-180, 1971.

5. Parameters

1: C0(NSM,NS) – ***real*** array. *Input*

On entry: contains the zero lag cross covariances between the NS series. C0 is assumed to be symmetric (upper triangle only is used).

2: C(NSM,NSM,NL) – ***real*** array. *Input*

On entry: contains the cross covariances at lags 1 to NL. C(i,j,k) must contain the cross covariance, c_{ijk}, of series i and series j at lag k. Series j leads series i.

3: NSM – INTEGER. *Input*

On entry: the first dimension of arrays C0 and DB and the first and second dimension of arrays C, D, W and WB as declared in the (sub)program from which G13DBF is called.

Constraint: NSM $\geq$ max(NS,1).

4: NS – INTEGER. *Input*

On entry: the number of time series whose cross covariances are supplied in C and C0.

Constraint: NS $\geq$ 1.

5: NL – INTEGER. *Input*

On entry: the maximum lag, K, for which cross covariances are supplied in C.

Constraint: NL $\geq$ 1.

6: NK – INTEGER. *Input*

On entry: the number of lags to which partial auto-correlations are to be calculated.

Constraint: 1 $\leq$ NK $\leq$ NL.

7: P(NK) – ***real*** array. *Output*

On exit: the multiple squared partial autocorrelations from lags 1 to NVP, that is P(k) contains p_k^2, for k = 1,2,...,NVP. For lags NVP + 1 to NK the elements of P are set to zero.

8: V0 – ***real***. *Output*

On exit: the lag zero prediction error variance (equal to the determinant of C0).

9: V(NK) – ***real*** array. *Output*

On exit: the prediction error variance ratios from lags 1 to NVP, that is V(k) contains v_k, for k = 1,2,...,NVP. For lags NVP + 1 to NK the elements of V are set to zero.

10: D(NSM,NSM,NK) – ***real*** array. *Output*

On exit: the prediction error variance matrices at lags 1 to NVP.

Element (i,j,k) of D contains the prediction error covariance of series i and series j at lag k, for k = 1,2,...,NVP. Series j leads series i, that is the (i,j)th element of D_k. For lags NVP + 1 to NK the elements of D are set to zero.

11: DB(NSM,NS) – ***real*** array. *Output*

On exit: the backward prediction error variance matrix at lag NVP.

DB(i,j) contains the backward prediction error covariance of series i and series j, that is the (i,j)th element of the G_k, where k = NVP.

12: W(NSM,NSM,NK) – ***real*** array. *Output*

On exit: the prediction coefficient matrices at lags 1 to NVP.

W(i,j,l) contains the jth prediction coefficient of series i at lag l, that is the (i,j)th element of Φ_{kl}, where k = NVP, for l = 1,2,...,NVP. For lags NVP + 1 to NK the elements of W are set to zero.

13: WB(NSM,NSM,NK) – ***real*** array. *Output*

On exit: the backward prediction coefficient matrices at lags 1 to NVP.

WB(i,j,l) contains the jth backward prediction coefficient of series i at lag l, that is the (i,j)th element of Ψ_{kl}, where k = NVP, for l = 1,2,...,NVP. For lags NVP + 1 to NK the elements of NB are set to zero.

14: NVP – INTEGER. *Output*

On exit: the maximum lag for which calculation of P, V, D, DB, W and WB was successful. If the routine completes successfully NVP will equal NK.

15: WA(IWA) – ***real*** array. *Workspace*

16: IWA – INTEGER. *Input*

On entry: the dimension of the array WA as declared in the (sub)program from which G13DBF is called.

Constraint: IWA $\geq$ (2×NS+1)×NS.

17: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

IFAIL = 1

On entry, NSM < 1,
or NS < 1,
or NS > NSM,
or NL < 1,
or NK < 1,
or NK > NL,
or IWA < (2×NS+1)×NS.

IFAIL = 2

C0 is not positive-definite.
V0, V, P, D, DB, W, WB and NVP are set to zero.

IFAIL = 3

At lag k = NVP + 1 $\leq$ NK, D_k was found not to be positive-definite. Up to lag NVP, V0, V, P, D, W and WB contain the values calculated so far and from lag NVP + 1 to lag NK the matrices contain zero. DB contains the backward prediction coefficients for lag NVP.

7. Accuracy

The conditioning of the problem depends on the prediction error variance ratios. Very small values of these may indicate loss of accuracy in the computations.

8. Further Comments

The time taken by the routine is roughly proportional to $NK^2 \times NS^3$.

9. Example

The example program reads the autocovariance matrices for 4 series from lag 0 to 5. It calls G13DBF to calculate the multivariate partial autocorrelation function and other related matrices of statistics up to lag 3. It prints the results.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DBF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NSMAX, NSM, NLMAX, NKMAX, IWA
      PARAMETER        (NSMAX=6,NSM=NSMAX,NLMAX=5,NKMAX=NLMAX,
     +                 IWA=(2*NSMAX+1)*NSMAX)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             V0
      INTEGER          I, I1, IFAIL, J, J1, K, NK, NL, NS, NVP
*     .. Local Arrays ..
      real             C(NSM,NSM,NLMAX), C0(NSM,NSMAX),
     +                 D(NSM,NSM,NKMAX), DB(NSM,NSMAX), P(NKMAX),
     +                 V(NKMAX), W(NSM,NSM,NKMAX), WA(IWA),
     +                 WB(NSM,NSM,NKMAX)
*     .. External Subroutines ..
      EXTERNAL         G13DBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*     Read series length, and numbers of lags
      READ (NIN,*) NS, NL, NK
      IF (NS.GT.0 .AND. NS.LE.NSMAX .AND. NL.GT.0 .AND. NL.LE.
     +    NLMAX .AND. NK.GT.0 .AND. NK.LE.NKMAX) THEN
*        Read autocovariances
         READ (NIN,*) ((C0(I,J),J=1,NS),I=1,NS)
         READ (NIN,*) (((C(I,J,K),J=1,NS),I=1,NS),K=1,NL)
*        Call routine to calculate multivariate partial autocorrelation
*        function
         IFAIL = 1
*
         CALL G13DBF(C0,C,NSM,NS,NL,NK,P,V0,V,D,DB,W,WB,NVP,WA,IWA,
     +               IFAIL)
*
         WRITE (NOUT,*)
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,99999) 'G13DBF fails. IFAIL =', IFAIL
            WRITE (NOUT,*)
         END IF
         IF (IFAIL.EQ.0 .OR. IFAIL.EQ.3) THEN
            WRITE (NOUT,99998) 'Number of valid parameters =', NVP
            WRITE (NOUT,*)
            WRITE (NOUT,*) 'Multivariate partial autocorrelations'
            WRITE (NOUT,99997) (P(I1),I1=1,NK)
            WRITE (NOUT,*)
            WRITE (NOUT,*)
     +          'Zero lag predictor error variance determinant'
               WRITE (NOUT,*) 'followed by error variance ratios'
               WRITE (NOUT,99997) V0, (V(I1),I1=1,NK)
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'Prediction error variances'
               DO 40 K = 1, NK
                  WRITE (NOUT,*)
                  WRITE (NOUT,99996) 'Lag =', K
                  DO 20 I = 1, NS
                     WRITE (NOUT,99997) (D(I,J1,K),J1=1,NS)
   20             CONTINUE
   40          CONTINUE
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'Last backward prediction error variances'
               WRITE (NOUT,*)
               WRITE (NOUT,99996) 'Lag =', NVP
               DO 60 I = 1, NS
                  WRITE (NOUT,99997) (DB(I,J1),J1=1,NS)
   60          CONTINUE
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'Prediction coefficients'
               DO 100 K = 1, NK
                  WRITE (NOUT,*)
                  WRITE (NOUT,99996) 'Lag =', K
                  DO 80 I = 1, NS
                     WRITE (NOUT,99997) (W(I,J1,K),J1=1,NS)
   80             CONTINUE
  100          CONTINUE
               WRITE (NOUT,*)
               WRITE (NOUT,*) 'Backward prediction coefficients'
               DO 140 K = 1, NK
                  WRITE (NOUT,*)
                  WRITE (NOUT,99996) 'Lag =', K
                  DO 120 I = 1, NS
                     WRITE (NOUT,99997) (WB(I,J1,K),J1=1,NS)
  120             CONTINUE
  140          CONTINUE
            END IF
         END IF
         STOP
*
99999 FORMAT (1X,A,I3)
99998 FORMAT (1X,A,I10)
99997 FORMAT (1X,5F12.5)
99996 FORMAT (1X,A,I5)
      END
```

9.2. Program Data

```
G13DBF Example Program Data
    4     5     3   500
  .10900E-01 -.77917E-02  .13004E-02  .12654E-02
 -.77917E-02  .57040E-01  .24180E-02  .14409E-01
  .13004E-02  .24180E-02  .43960E-01 -.21421E-01
  .12654E-02  .14409E-01 -.21421E-01  .72289E-01
  .45889E-02  .46510E-03 -.13275E-03  .77531E-02
 -.24419E-02 -.11667E-01 -.21956E-01 -.45803E-02
  .11080E-02 -.80479E-02  .13621E-01 -.85868E-02
 -.50614E-03  .14045E-01 -.10087E-02  .12269E-01
  .18652E-02 -.64389E-02  .88307E-02 -.24808E-02
 -.11865E-01  .72367E-02 -.19802E-01  .59069E-02
 -.80307E-02  .14306E-01  .14546E-01  .13510E-01
 -.21791E-02 -.29528E-01 -.15887E-01  .88308E-03
 -.80550E-04 -.37759E-02  .75463E-02 -.42276E-02
  .41447E-02 -.37987E-02  .19332E-02 -.17564E-01
 -.10582E-01  .67733E-02  .69832E-02  .61747E-02
  .41352E-02 -.16013E-01  .17043E-01 -.13412E-01
  .76079E-03 -.10134E-02  .11870E-01 -.41651E-02
  .36014E-02 -.36375E-02 -.25571E-01  .50218E-02
 -.13924E-01  .11718E-01 -.59088E-02  .59297E-02
```

```
  .10739E-01 -.14571E-01  .13816E-01 -.12588E-01
 -.64365E-03 -.44556E-02  .51334E-02  .71587E-03
  .63617E-02  .15217E-03  .27270E-02 -.22261E-02
 -.85855E-02  .14468E-02 -.28698E-02  .44384E-02
  .68339E-02 -.21790E-02  .13759E-01  .28217E-03
```

9.3. Program Results

```
G13DBF Example Program Results

Number of valid parameters =          3

Multivariate partial autocorrelations
     0.64498     0.92669     0.84300

Zero lag predictor error variance determinant
followed by error variance ratios
     0.00000     0.35502     0.02603     0.00409

Prediction error variances

Lag =      1
     0.00811    -0.00511     0.00159    -0.00029
    -0.00511     0.04089     0.00757     0.01843
     0.00159     0.00757     0.03834    -0.01894
    -0.00029     0.01843    -0.01894     0.06760

Lag =      2
     0.00354    -0.00087    -0.00075    -0.00105
    -0.00087     0.01946     0.00535     0.00566
    -0.00075     0.00535     0.01900    -0.01071
    -0.00105     0.00566    -0.01071     0.04058

Lag =      3
     0.00301    -0.00087    -0.00054     0.00065
    -0.00087     0.01824     0.00872     0.00247
    -0.00054     0.00872     0.00935    -0.00216
     0.00065     0.00247    -0.00216     0.02254

Last backward prediction error variances

Lag =      3
     0.00331    -0.00392    -0.00106     0.00592
    -0.00392     0.01890     0.00348    -0.00330
    -0.00106     0.00348     0.01003    -0.01054
     0.00592    -0.00330    -0.01054     0.03336

Prediction coefficients

Lag =      1
     0.81861     0.23399    -0.17097     0.09256
     0.06738    -0.48720    -0.14064     0.04295
     0.15036     0.11924    -0.36725    -0.42092
    -0.70971     0.02998     0.59779     0.34610

Lag =      2
    -0.34049    -0.13370     0.40610    -0.02183
    -1.27574    -0.13591    -0.65779    -0.11267
    -0.45439     0.19379     0.63420     0.33920
    -0.43237    -0.54848    -0.62897     0.16670

Lag =      3
     0.16437     0.13858     0.01290     0.03463
     0.39291     0.07407    -0.08802    -0.15361
    -1.29240    -0.24489     0.30235     0.39442
     0.89768    -0.39040     0.25151    -0.28304
```

```
Backward prediction coefficients

Lag =     1
     0.41541      0.06149      0.15319      0.05079
     0.12370     -0.26471     -0.22721      0.48503
    -0.86933     -0.47373      0.37924      0.13814
     1.30779     -0.09178     -1.45398     -0.21967

Lag =     2
    -0.06740     -0.12255     -0.13673     -0.09730
    -1.24801      0.03090      0.51706     -0.28925
     0.98045     -0.20194      0.16307     -0.10869
    -1.68389     -0.74589      0.52900      0.41580

Lag =     3
     0.03794      0.10491     -0.21635      0.08015
     0.75392      0.22603     -0.25661     -0.47450
    -0.00338      0.05636     -0.08818      0.12723
     0.55022     -0.41232      0.71649     -0.14565
```

G13DCF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DCF fits a vector autoregressive moving average (VARMA) model to an observed vector of time series using the method of maximum likelihood. Standard errors of parameter estimates are computed along with their appropriate correlation matrix. The routine also calculates estimates of the residual series.

2. Specification

```
SUBROUTINE G13DCF (K, N, IP, IQ, MEAN, PAR, NPAR, QQ, IK, W, PARHLD,
1                  EXACT, IPRINT, CGETOL, MAXCAL, ISHOW, NITER,
2                  RLOGL, V, G, CM, ICM, WORK, LWORK, IW, LIW, IFAIL)
INTEGER         K, N, IP, IQ, NPAR, IK, IPRINT, MAXCAL, ISHOW, NITER,
1               ICM, LWORK, IW(LIW), LIW, IFAIL
real            PAR(NPAR), QQ(IK,K), W(IK,N), CGETOL, RLOGL, V(IK,N),
1               G(NPAR), CM(ICM,NPAR), WORK(LWORK)
LOGICAL         MEAN, PARHLD(NPAR), EXACT
```

3. Description

Let $W_t = (w_{1t},w_{2t},...,w_{kt})^T$, for $t = 1,2,...,n$ denote a vector of k time series which is assumed to follow a multivariate ARMA model of the form:

$$\begin{aligned} W_t - \mu = {} & \phi_1(W_{t-1}-\mu) + \phi_2(W_{t-2}-\mu) + ... + \phi_p(W_{t-p}-\mu) \\ & + \varepsilon_t - \theta_1\ \varepsilon_{t-1} - \theta_2\ \varepsilon_{t-2} - ... - \theta_q\ \varepsilon_{t-q} \end{aligned} \tag{1}$$

where $\varepsilon_t = (\varepsilon_{1t},\varepsilon_{2t},...,\varepsilon_{kt})^T$, for $t = 1,2,...,n$ is a vector of k residual series assumed to be Normally distributed with zero mean and positive-definite covariance matrix Σ. The components of ε_t are assumed to be uncorrelated at non-simultaneous lags. The ϕ_i's and θ_j's are k by k matrices of parameters. $\{\phi_i\}$, for $i = 1,2,...,p$, are called the autoregressive (AR) parameter matrices, and $\{\theta_i\}$, for $i = 1,2,...,q$, the moving average (MA) parameter matrices. The parameters in the model are thus the p k by k ϕ-matrices, the q k by k θ-matrices, the mean vector, μ, and the residual error covariance matrix Σ. Let

$$A(\phi) = \begin{bmatrix} \phi_1 & I\ 0 & . & . & . & 0 \\ \phi_2 & 0\ I & 0 & . & . & 0 \\ . & & . & & & \\ . & & & & . & \\ . & & & & . & \\ \phi_{p-1} & 0 & . & . & . & 0\ I \\ \phi_p & 0 & . & . & . & 0\ 0 \end{bmatrix}_{pk\times pk} \quad \text{and} \quad B(\theta) = \begin{bmatrix} \theta_1 & I\ 0 & . & . & . & 0 \\ \theta_2 & 0\ I & 0 & . & . & 0 \\ . & & . & & & \\ . & & & & . & \\ . & & & & . & \\ \theta_{q-1} & 0 & . & . & . & . & I \\ \theta_q & 0 & . & . & . & . & 0 \end{bmatrix}_{qk\times qk}$$

where I denotes the k by k identity matrix.

The model (1) is said to be stationary if the eigenvalues of $A(\phi)$ lie inside the unit circle. Similarly, (1) is said to be invertible if the eigenvalues of $B(\theta)$ lie inside the unit circle.

The method of computing the exact likelihood function (using a Kalman filter algorithm) is discussed in Shea [1]. A quasi-Newton algorithm (see Gill and Murray [2]) is then used to search for the maximum of the log likelihood function. Stationarity and invertibility are enforced on the model using the reparameterisation discussed in Ansley and Kohn [3]. Conditional on the maximum likelihood estimates being equal to their true values the estimates of the residual series are uncorrelated with zero mean and constant variance Σ.

The user has the option of setting a parameter (EXACT to .FALSE.) so that G13DCF calculates conditional maximum likelihood estimates (conditional on

$W_0 = W_{-1} = ... = W_{1-p} = \varepsilon_0 = \varepsilon_{-1} = ... = \varepsilon_{1-q} = 0$).
This may be useful if the exact maximum likelihood estimates are close to the boundary of the invertibility region.

The user also has the option (see Section 5) of requesting G13DCF to constrain elements of the ϕ and θ matrices and μ vector to have pre-specified values.

4. References

[1] SHEA, B.L.
Estimation of multivariate time series.
J. Time Series Analysis, 8, pp. 95-110, 1987.

[2] GILL, P.E. and MURRAY, W.
Quasi-Newton methods for unconstrained optimization.
J. Inst. Maths. Applics., 9, pp. 91-108, 1972.

[3] ANSLEY, C.F. and KOHN, R.
A Note on Reparameterising a Vector Autoregressive Moving Average Model to Enforce Stationarity.
J. Stat. Comp. Sim., 24, pp. 99-106, 1986.

5. Parameters

1: K – INTEGER. *Input*

On entry: the number of observed time series, k.

Constraint: K $\geq$ 1.

2: N – INTEGER. *Input*

On entry: the number of observations in each time series, n.

Constraint: N $\geq$ 3.

3: IP – INTEGER. *Input*

On entry: the number of AR parameter matrices, p.

Constraint: IP $\geq$ 0.

4: IQ – INTEGER. *Input*

On entry: the number of MA parameter matrices, q.

Constraint: IQ $\geq$ 0.

Note: IP = IQ = 0 is **not permitted**.

5: MEAN – LOGICAL. *Input*

On entry: MEAN must be set to .TRUE. if components of μ are to be estimated and .FALSE. if all elements of μ are to be taken as zero.

6: PAR(NPAR) – ***real*** array. *Input/Output*

On entry: initial parameter estimates read in row by row in the order $\phi_1,\phi_2,...,\phi_p$, $\theta_1,\theta_2,...,\theta_q,\mu$. Thus, if IP > 0 then PAR($(l-1)\times k\times k+(i-1)\times k+j$) must be set equal to an initial estimate of the (i,j)th element of ϕ_l for $l = 1,2,...,p$; $i,j = 1,2,...,k$. If IQ > 0 then PAR($p\times k\times k+(l-1)\times k\times k+(i-1)\times k+j$) must be set equal to an initial estimate of the (i,j)th element of θ_l, $l = 1,2,...,q$; $i,j = 1,2,...,k$. If MEAN has been set equal to .TRUE., then PAR($(p+q)\times k\times k+i$) should be set equal to an initial estimate of the ith component of μ ($\mu(i)$). (If the user sets PAR($(p+q)\times k\times k+i$) to 0.0 then G13DCF will calculate the mean of the ith series and use this as an initial estimate of $\mu(i)$.)

The first $p\times k\times k$ elements of PAR must satisfy the stationarity condition and the next $q\times k\times k$ elements of PAR must satisfy the invertibility condition.

If in doubt set all elements of PAR to 0.0.

On exit: if IFAIL = 0 or IFAIL ≥ 4 then all the elements of PAR will be overwritten by the latest estimates of the corresponding ARMA parameters.

7: NPAR – INTEGER. *Input*

On entry: the number of initial parameter estimates. The total number of observations $(n \times k)$ must exceed the total number of parameters in the model $(\text{NPAR}+k(k+1)/2)$.

Constraint: if MEAN = .FALSE., NPAR must be set equal to $(p+q)\times k\times k$, if MEAN = .TRUE., $(p+q)\times k\times k + k$.

8: QQ(IK,K) – ***real*** array. *Input/Output*

On entry: QQ(i,j) must be set equal to an initial estimate of the (i,j)th element of Σ. The lower triangle only is needed. QQ must be positive-definite. It is strongly recommended that on entry, the elements of QQ are of the same order of magnitude as at the solution point. If the user sets QQ(i,j) = 0, for $i = 1,2,...,k$; $j = 1,2,...,i$, then G13DCF will calculate the covariance matrix between the k time series and use this as an initial estimate of Σ.

On exit: if IFAIL = 0 or IFAIL ≥ 4 then QQ(i,j) will contain the latest estimate of the (i,j)th element of Σ. The lower triangle only is returned.

9: IK – INTEGER. *Input*

On entry: the first dimension of the array QQ, W and V as declared in the (sub)program from which G13DCF is called.

Constraint: IK ≥ K.

10: W(IK,N) – ***real*** array. *Input*

On entry: W(i,t) must be set equal to the ith component of W_t, for $i = 1,2,...,k$; $t = 1,2,...,n$.

11: PARHLD(NPAR) – LOGICAL array. *Input*

On entry: PARHLD(i) must be set to .TRUE., if PAR(i) is to be held constant at its input value and .FALSE., if PAR(i) is a free parameter, for i = 1,2,...,NPAR.

If in doubt try setting all elements of PARHLD to .FALSE..

12: EXACT – LOGICAL. *Input*

On entry: EXACT must be set equal to .TRUE. if the user wishes the routine to compute exact maximum likelihood estimates. EXACT must be set equal to .FALSE. if only conditional likelihood estimates are required.

13: IPRINT – INTEGER. *Input*

On entry: the frequency with which the automatic monitoring routine is to be called. See Section 9 for an example of the printed output.

If IPRINT > 0, the ML search procedure is monitored once every IPRINT iterations and just before exit from the search routine.

If IPRINT = 0, the search routine is monitored once at the final point.

If IPRINT < 0 the search routine is not monitored at all.

14: CGETOL – ***real.*** *Input*

On entry: the accuracy to which the solution in PAR and QQ is required. If CGETOL is set to 10^{-l} and on exit IFAIL = 0 or IFAIL ≥ 6, then all the elements in PAR and QQ should be accurate to approximately l decimal places. For most practical purposes the value 10^{-4} should suffice. The user should be wary of setting CGETOL too small since the convergence criteria may then have become too strict for the machine to handle. If on entry, CGETOL has been set to a value which is less than the ***machine precision***, ε, then G13DCF will use the value $10.0\times\sqrt{\varepsilon}$ instead.

15: MAXCAL – INTEGER. *Input*

On entry: the maximum number of likelihood evaluations to be permitted by the search procedure. A reasonable setting for MAXCAL may be 40×NPAR×(NPAR+5).

Constraint: MAXCAL ≥ 1.

16: ISHOW – INTEGER. *Input*

On entry: which of the following two quantities are to be printed.

(a) Table of maximum likelihood estimates and their standard errors (as returned in the output arrays PAR, QQ and CM).

(b) Table of residual series (as returned in the output array V).

ISHOW = 0 none of the above are printed,
ISHOW = 1 (a) only is printed,
ISHOW = 2 (a) and (b) are printed.

17: NITER – INTEGER. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 4 then NITER contains the number of iterations performed by the search routine.

18: RLOGL – ***real***. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 4 then RLOGL contains the value of the log likelihood function corresponding to the final point held in PAR and QQ.

19: V(IK,N) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 4 then V(i,t) will contain an estimate of the ith component of ε_t, for $i = 1,2,...,k$; $t = 1,2,...,n$, corresponding to the final point held in PAR and QQ.

20: G(NPAR) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 4 then G(i) will contain the estimated first derivative of the log likelihood function with respect to the ith element in the array PAR. If the gradient cannot be computed then all the elements of G are returned as zero.

21: CM(ICM,NPAR) – ***real*** array. *Output*

On exit: if IFAIL = 0 or IFAIL ≥ 4 then CM(i,j) will contain an estimate of the correlation coefficient between the ith and jth elements in the PAR array for $1 \le i \le$ NPAR, $1 \le j \le$ NPAR. If $i = j$ then CM(i,j) will contain the estimated standard error of PAR(i). If the lth component of PAR has been held constant i.e. PARHLD(l) was set to .TRUE., then the lth row and column of CM will be set to zero. If the second derivative matrix cannot be computed then all the elements of CM are returned as zero.

22: ICM – INTEGER. *Input*

On entry: the first dimension of the array CM as declared in the (sub)program from which G13DCF is called.

Constraint: ICM ≥ NPAR.

23: WORK(LWORK) – ***real*** array. *Workspace*
24: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G13DCF is called.

Constraints: if $r = \max(p,q)$ and M = NPAR + $k(k+1)/2$, then

$$\text{LWORK} \ge k(5+3n+2r) + k^2(2p+q+2r(r+2)+9) + \text{M}(5\text{M}+29)/2 + k^4(r+1)^2.$$

25: IW(LIW) – INTEGER array. *Workspace*

26: LIW – INTEGER. *Input*

On entry: the dimension of the array IW as declared in the (sub)program from which G13DCF is called.

Constraint: LIW $\geq$ NPAR + kr + 2 + $k(k+1)/2$.

27: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL $\neq$ 0 on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, K < 1,
or IP < 0,
or IQ < 0,
or IP = IQ = 0,
or NPAR $\neq$ (IP+IQ)×K×K + Δ×K, where Δ = 1 if MEAN = .TRUE. or Δ = 0 if MEAN = .FALSE.,
or N×K $\leq$ NPAR + K×(K+1)/2,
or IK < K,
or MAXCAL < 1,
or ISHOW < 0,
or ISHOW > 2,
or ICM < NPAR,
or LWORK is too small,
or LIW is too small.

IFAIL = 2

On entry, either the initial estimate of Σ is not positive-definite, or the initial estimates of the AR parameters are such that the model is non-stationary, or the initial estimates of the MA parameters are such that the model is non-invertible. To proceed, the user must try a different starting point.

IFAIL = 3

The routine cannot compute a sufficiently accurate estimate of the gradient vector at the user-supplied starting point. This usually occurs if either the initial parameter estimates are very close to the ML parameter estimates, or the user has supplied a very poor estimate of Σ or the starting point is very close to the boundary of the stationarity or invertibility region. To proceed, the user must try a different starting point.

IFAIL = 4

There have been MAXCAL log likelihood evaluations made in the routine. If steady increases in the log likelihood function were monitored up to the point where this exit occurred, then the exit probably simply occurred because MAXCAL was set too small, so the calculations should be restarted from the final point held in PAR and QQ. This type of exit may also indicate that there is no maximum to the likelihood surface. Output quantities (as described in Section 5) are computed at the final point held in PAR and QQ, except that if G or CM cannot be computed, they are set to zero.

IFAIL = 5

The conditions for a solution have not all been met, but a point at which the log likelihood took a larger value could not be found.

Provided that the estimated first derivatives are sufficiently small, and that the estimated condition number of the second derivative (Hessian) matrix, as printed when IPRINT ≥ 0, is not too large, this error exit may simply mean that, although it has not been possible to satisfy the specified requirements, the algorithm has in fact found the solution as far as the accuracy of the machine permits.

Such a condition can arise, for instance, if CGETOL has been set so small that rounding error in evaluating the likelihood function makes attainment of the convergence conditions impossible.

If the estimated condition number at the final point is large, it could be that the final point is a solution but that the smallest eigenvalue of the Hessian matrix is so close to zero at the solution that it is not possible to recognise it as a solution. Output quantities (as described in Section 5) are computed at the final point held in PAR and QQ, except that if G or CM cannot be computed, they are set to zero.

IFAIL = 6

The ML solution is so close to the boundary of either the stationarity region or the invertibility region that G13DCF cannot evaluate the Hessian matrix. The elements of CM will then be set to zero on exit. The elements of G will also be set to zero. All other output quantities will be correct.

IFAIL = 7

This is an unlikely exit, which could occur in E04XAF, which computes an estimate of the second derivative matrix and the gradient vector at the solution point. Either the Hessian matrix was found to be too ill-conditioned to be evaluated accurately or the gradient vector could not be computed to an acceptable degree of accuracy. In this case the elements of CM will be set to zero on exit as will the elements of G. All other output quantities will be correct.

IFAIL = 8

The second derivative matrix at the solution point is not positive-definite. In this case the elements of CM will be set to zero on exit. All other output quantities will be correct.

7. Accuracy

On exit from G13DCF, if IFAIL = 0 or IFAIL ≥ 6 and CGETOL has been set to 10^{-l}, then all the parameters should be accurate to approximately l decimal places. If CGETOL was set equal to a value less than the ***machine precision***, ε, then all the parameters should be accurate to approximately $10.0\times\sqrt{\varepsilon}$.

If IFAIL = 4 on exit, (i.e. MAXCAL likelihood evaluations have been made but the convergence conditions of the search routine have not been satisfied) then the elements in PAR and QQ may still be good approximations to the ML estimates. The user is advised to inspect the elements of G to see whether this is likely to be so.

8. Further Comments

8.1. Timing

The number of iterations required depends upon the number of parameters in the model and the distance of the user-supplied starting point from the solution.

8.2. Constraining for Stationarity and Invertibility

If the solution lies on the boundary of the admissibility region (stationarity and invertibility region) then G13DCF may get into difficulty and exit with IFAIL = 5. If this exit occurs the user is advised to either try a different starting point or a different setting for EXACT. If this still continues to occur then the user is urged to try fitting a more parsimonious model.

8.3. Over-parameterisation

The user is advised to try and avoid fitting models with an excessive number of parameters since over-parameterisation can cause the maximization problem to become ill-conditioned.

8.4. Standardising the Residual Series

The standardised estimates of the residual series ε_t (denoted by $\hat{e}_t$) can easily be calculated by forming the Cholesky decomposition of Σ e.g. GG^T and setting $\hat{e}_t = G^{-1}\hat{\varepsilon}_t$. F07FDF (SPOTRF/DPOTRF) may be used to calculate the array G. The components of $\hat{e}_t$ which are now uncorrelated at **all** lags can sometimes be more easily interpreted.

8.5. Assessing the Fit of the Model

If the user's time series model provides a good fit to the data then the residual series should be approximately white noise, i.e. exhibit no serial cross-correlation. An examination of the residual cross-correlation matrices should confirm whether this is likely to be so. The user is advised to call G13DSF to provide information for diagnostic checking. G13DSF returns the residual cross-correlation matrices along with their asymptotic standard errors. The routine also computes a portmanteau statistic and its asymptotic significance level for testing model adequacy. If IFAIL = 0 or IFAIL ≥ 5 on exit from G13DCF then the output quantities K, N, V, IK, IP, IQ, PAR, PARHLD, and QQ will be suitable for input to G13DSF.

9. Example

A program to fit a bivariate AR(1) model to two series each of length 48. μ will be estimated and $\phi_1(2,1)$ will be constrained to be zero.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DCF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          KMAX, IK, IPMAX, IQMAX, NMAX, NPARMX, ICM, LWORK,
     +                 LIW
      PARAMETER        (KMAX=3,IK=KMAX,IPMAX=3,IQMAX=3,NMAX=100,
     +                 NPARMX=(IPMAX+IQMAX)*KMAX*KMAX+KMAX,ICM=NPARMX,
     +                 LWORK=2000,LIW=100)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             CGETOL, RLOGL
      INTEGER          I, IFAIL, IP, IPRINT, IQ, ISHOW, J, K, MAXCAL, N,
     +                 NITER, NPAR
      LOGICAL          EXACT, MEAN
*     .. Local Arrays ..
      real             CM(ICM,NPARMX), G(NPARMX), PAR(NPARMX),
     +                 QQ(IK,KMAX), V(IK,NMAX), W(IK,NMAX), WORK(LWORK)
      INTEGER          IW(LIW)
      LOGICAL          PARHLD(NPARMX)
*     .. External Subroutines ..
      EXTERNAL         G13DCF, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DCF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, IP, IQ, N, MEAN
      CALL X04ABF(1,NOUT)
      WRITE (NOUT,*)
      IF (K.GT.0 .AND. K.LE.KMAX .AND. IP.GE.0 .AND. IP.LE.IPMAX .AND.
     +    IQ.GE.0 .AND. IQ.LE.IQMAX) THEN
         NPAR = (IP+IQ)*K*K
         IF (MEAN) NPAR = NPAR + K
```

```
            IF ((N.LE.NMAX) .AND. (NPAR.LE.NPARMX)) THEN
               DO 20 I = 1, NPAR
                  PAR(I) = 0.0e0
                  PARHLD(I) = .FALSE.
   20          CONTINUE
*
*              Set all elements of Q to zero to use covariance matrix
*              between the K time series as the initial estimate of the
*              covariance matrix
*
               DO 60 J = 1, K
                  DO 40 I = J, K
                     QQ(I,J) = 0.0e0
   40             CONTINUE
   60          CONTINUE
               DO 80 I = 1, K
                  READ (NIN,*) (W(I,J),J=1,N)
   80          CONTINUE
               PARHLD(3) = .TRUE.
               EXACT = .TRUE.
*              ** Set IPRINT .GT. 0 to obtain intermediate output
               IPRINT = -1
               CGETOL = 0.0001e0
               MAXCAL = 40*NPAR*(NPAR+5)
               ISHOW = 2
               IFAIL = -1
*
               CALL G13DCF(K,N,IP,IQ,MEAN,PAR,NPAR,QQ,IK,W,PARHLD,EXACT,
     +                     IPRINT,CGETOL,MAXCAL,ISHOW,NITER,RLOGL,V,G,CM,
     +                     ICM,WORK,LWORK,IW,LIW,IFAIL)
*
            END IF
         END IF
         STOP
         END
```

9.2. Program Data

```
G13DCF Example Program Data
2 1 0 48 T
-1.490 -1.620  5.200  6.230  6.210  5.860  4.090  3.180
 2.620  1.490  1.170  0.850 -0.350  0.240  2.440  2.580
 2.040  0.400  2.260  3.340  5.090  5.000  4.780  4.110
 3.450  1.650  1.290  4.090  6.320  7.500  3.890  1.580
 5.210  5.250  4.930  7.380  5.870  5.810  9.680  9.070
 7.290  7.840  7.550  7.320  7.970  7.760  7.000  8.350
 7.340  6.350  6.960  8.540  6.620  4.970  4.550  4.810
 4.750  4.760 10.880 10.010 11.620 10.360  6.400  6.240
 7.930  4.040  3.730  5.600  5.350  6.810  8.270  7.680
 6.650  6.080 10.250  9.140 17.750 13.300  9.630  6.800
 4.080  5.060  4.940  6.650  7.940 10.760 11.890  5.850
 9.010  7.500 10.020 10.380  8.150  8.370 10.730 12.140
```

9.3. Program Results

```
 G13DCF Example Program Results


 VALUE OF IFAIL PARAMETER ON EXIT FROM G13DCF =    0

 VALUE OF LOG LIKELIHOOD FUNCTION ON EXIT = -0.20280E+03

 MAXIMUM LIKELIHOOD ESTIMATES OF AR PARAMETER MATRICES
 ----------------------------------------------------

 PHI(1)     ROW-WISE :     0.802    0.065
                          (0.091)  (0.102)

                           0.000    0.575
                          (0.000)  (0.121)
```

```
MAXIMUM LIKELIHOOD ESTIMATE OF PROCESS MEAN
-------------------------------------------

                    4.271   7.825
                   (1.219) (0.776)

MAXIMUM LIKELIHOOD ESTIMATE OF SIGMA MATRIX
-------------------------------------------

                    2.964

                    0.637   5.380

          RESIDUAL SERIES NUMBER  1
          -------------------------

   T     1      2      3      4      5      6      7      8
V(T)  -3.33  -1.24   5.75   1.27   0.32   0.11  -1.27  -0.73

   T     9     10     11     12     13     14     15     16
V(T)  -0.58  -1.26  -0.67  -1.13  -2.02  -0.57   1.24  -0.13

   T    17     18     19     20     21     22     23     24
V(T)  -0.77  -2.09   1.34   0.95   1.71   0.23  -0.01  -0.60

   T    25     26     27     28     29     30     31     32
V(T)  -0.68  -1.89  -0.77   2.05   2.11   0.94  -3.32  -2.50

   T    33     34     35     36     37     38     39     40
V(T)   3.16   0.47   0.05   2.77  -0.82   0.25   3.99   0.20

   T    41     42     43     44     45     46     47     48
V(T)  -0.70   1.07   0.44   0.28   1.09   0.50  -0.10   1.70

          RESIDUAL SERIES NUMBER  2
          -------------------------

   T     1      2      3      4      5      6      7      8
V(T)  -0.19  -1.20  -0.02   1.21  -1.62  -2.16  -1.63  -1.13

   T     9     10     11     12     13     14     15     16
V(T)  -1.34  -1.30   4.82   0.43   2.54   0.35  -2.88  -0.77

   T    17     18     19     20     21     22     23     24
V(T)   1.02  -3.85  -1.92   0.13  -1.20   0.41   1.03  -0.40

   T    25     26     27     28     29     30     31     32
V(T)  -1.09  -1.07   3.43  -0.08   9.17  -0.23  -1.34  -2.06

   T    33     34     35     36     37     38     39     40
V(T)  -3.16  -0.61  -1.30   0.48   0.79   2.87   2.38  -4.31

   T    41     42     43     44     45     46     47     48
V(T)   2.32  -1.01   2.38   1.29  -1.14   0.36   2.59   2.64
```

With IPRINT set to 1 in the example program, intermediate output similar to the following is obtained:

```
G13DCF Example Program Results


ITERATION NUMBER =     0

ESTIMATED CONDITION NUMBER OF HESSIAN MATRIX =   0.159E+02

NUMBER OF LIKELIHOOD EVALUATIONS MADE SO FAR =       0

VALUE OF LOG LIKELIHOOD FUNCTION = -0.23408E+03
```

```
AR PARAMETERS :   0.000E+00  0.000E+00  0.000E+00  0.000E+00
MEAN VECTOR   :   0.437E+01  0.787E+01
SIGMA MATRIX  :   0.794E+01
                  0.198E+01  0.792E+01

NORM OF GRADIENT VECTOR =  0.43931E+02

*************************************************************************

ITERATION NUMBER =     1

ESTIMATED CONDITION NUMBER OF HESSIAN MATRIX =    0.159E+02

NUMBER OF LIKELIHOOD EVALUATIONS MADE SO FAR =       11

VALUE OF LOG LIKELIHOOD FUNCTION = -0.21278E+03

AR PARAMETERS :   0.788E+00  0.838E-01  0.000E+00  0.522E+00
MEAN VECTOR   :   0.437E+01  0.787E+01
SIGMA MATRIX  :   0.794E+01
                  0.198E+01  0.792E+01

NORM OF GRADIENT VECTOR =  0.33305E+02

*************************************************************************
```

.

.

intermediate results omitted

.

.

```
ITERATION NUMBER =    16

ESTIMATED CONDITION NUMBER OF HESSIAN MATRIX =    0.323E+03

NUMBER OF LIKELIHOOD EVALUATIONS MADE SO FAR =      156

VALUE OF LOG LIKELIHOOD FUNCTION = -0.20280E+03

AR PARAMETERS :   0.802E+00  0.648E-01  0.000E+00  0.575E+00
MEAN VECTOR   :   0.427E+01  0.783E+01
SIGMA MATRIX  :   0.296E+01
                  0.637E+00  0.538E+01

NORM OF GRADIENT VECTOR =  0.28650E-03

*************************************************************************

ITERATION NUMBER =    16

ESTIMATED CONDITION NUMBER OF HESSIAN MATRIX =    0.323E+03

NUMBER OF LIKELIHOOD EVALUATIONS MADE SO FAR =      180

VALUE OF LOG LIKELIHOOD FUNCTION = -0.20280E+03

AR PARAMETERS :   0.802E+00  0.648E-01  0.000E+00  0.575E+00
MEAN VECTOR   :   0.427E+01  0.783E+01
SIGMA MATRIX  :   0.296E+01
                  0.637E+00  0.538E+01

NORM OF GRADIENT VECTOR =  0.28950E-03

*************************************************************************
```

G13DJF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DJF computes forecasts of a multivariate time series. It is assumed that a vector ARMA model has already been fitted to the appropriately differenced/transformed time series using G13DCF. The standard deviations of the forecast errors are also returned. A reference vector is set up so that should future series values become available the forecasts and their standard errors may be updated by calling G13DKF.

2. Specification

```
      SUBROUTINE G13DJF (K, N, Z, IK, TR, ID, DELTA, IP, IQ, MEAN,
     1                   PAR, LPAR, QQ, V, LMAX, PREDZ, SEFZ, REF,
     2                   LREF, WORK, LWORK, IWORK, LIWORK, IFAIL)
      INTEGER          K, N, IK, ID(K), IP, IQ, LPAR, LMAX, LREF,
     1                 LWORK, IWORK(LIWORK), LIWORK, IFAIL
      real             Z(IK,N), DELTA(IK,*), PAR(LPAR), QQ(IK,K),
     1                 V(IK,*), PREDZ(IK,LMAX), SEFZ(IK,LMAX),
     2                 REF(LREF), WORK(LWORK)
      CHARACTER*1      TR(K), MEAN
```

3. Description

Let the vector $Z_t = (z_{1t},z_{2t},\ldots,z_{kt})^T$, $(t = 1,2,\ldots,n)$, denote a k-dimensional time series for which forecasts of $Z_{n+1},Z_{n+2},\ldots,Z_{n+l_{max}}$ are required. Let $W_t = (w_{1t},w_{2t},\ldots,w_{kt})^T$ be defined as follows:

$$w_{it} = \delta_i(B)z_{it}^*, \quad \text{for } i = 1,2,\ldots,k$$

where $\delta_i(B)$ is the differencing operator applied to the ith series and where z_{it}^* is equal to either z_{it}, $\sqrt{z_{it}}$ or $\log_e z_{it}$ depending on whether or not a transformation was required to stabilize the variance before fitting the model.

If the order of differencing required for the ith series is d_i, then the differencing operator for the ith series is defined by $\delta_i(B) = 1 - \delta_{i1}B - \delta_{i2}B^2 - \ldots - \delta_{id_i}B^{d_i}$ where B is the backward shift operator, that is $BZ_t = Z_{t-1}$. The differencing parameters δ_{ij} for $i = 1,2,\ldots,k$; $j = 1,2,\ldots,d_i$ must be supplied by the user. If the ith series does not require differencing, then $d_i = 0$.

W_t is assumed to follow a multivariate ARMA model of the form:

$$\begin{aligned} W_t-\mu = {} & \phi_1(W_{t-1}-\mu) + \phi_2(W_{t-2}-\mu) + \ldots + \phi_p(W_{t-p}-\mu) \\ & + \varepsilon_t - \theta_1\varepsilon_{t-1} - \theta_2\varepsilon_{t-2} - \ldots - \theta_q\varepsilon_{t-q} \end{aligned} \qquad (1)$$

where $\varepsilon_t = (\varepsilon_{1t},\varepsilon_{2t},\ldots,\varepsilon_{kt})^T$, for $t = 1,2,\ldots,n$ is a vector of k residual series assumed to be Normally distributed with zero mean and positive-definite covariance matrix Σ. The components of ε_t are assumed to be uncorrelated at non-simultaneous lags. The ϕ_i's and θ_j's are k by k matrices of parameters. The matrices ϕ_i, for $i = 1,2,\ldots,p$, are the autoregressive (AR) parameter matrices, and the matrices θ_i, for $i = 1,2,\ldots,q$, the moving average (MA) parameter matrices. The parameters in the model are thus the p (k by k) ϕ-matrices, the q (k by k) θ-matrices, the mean vector μ and the residual error covariance matrix Σ. The ARMA model (1) must be both stationary and invertible; see G13DXF for a method of checking these conditions.

The ARMA model (1) may be rewritten as

$$\phi(B)(\delta(B)Z_t^*-\mu) = \theta(B)\varepsilon_t$$

where $\phi(B)$ and $\theta(B)$ are the autoregressive and moving average polynomials and $\delta(B)$ denotes the k by k diagonal matrix whose ith diagonal elements is $\delta_i(B)$ and $Z_t^* = (z_{1t}^*,z_{2t}^*,\ldots,z_{kt}^*)^T$.

This may be rewritten as

$$\phi(B)\delta(B)Z_t^* = \phi(B)\mu + \theta(B)\varepsilon_t$$

or

$$Z_t^* = \tau + \psi(B)\varepsilon_t = \tau + \varepsilon_t + \psi_1\varepsilon_{t-1} + \psi_2\varepsilon_{t-2} + \ldots$$

where $\psi(B) = \delta^{-1}(B)\phi^{-1}(B)\theta(B)$ and $\tau = \delta^{-1}(B)\mu$ is a vector of length k.

Forecasts are computed using a multivariate version of the procedure described in Box and Jenkins [1]. If $\hat{Z}_n^*(l)$ denotes the forecast of Z_{n+l}^*, then $\hat{Z}_n^*(l)$ is taken to be that linear function of $Z_n^*, Z_{n-1}^*, \ldots$ which minimises the elements of $E\{e_n(l)e_n'(l)\}$ where $e_n(l) = Z_{n+l}^* - \hat{Z}_n^*(l)$ is the forecast error. $\hat{Z}_n^*(l)$ is referred to as the linear minimum mean square error forecast of Z_{n+l}^*.

The linear predictor which minimises the mean square error may be expressed as

$$\hat{Z}_n^*(l) = \tau + \psi_l\varepsilon_n + \psi_{l+1}\varepsilon_{n-1} + \psi_{l+2}\varepsilon_{n-2} + \ldots$$

The forecast error at t for lead l is then

$$e_n(l) = Z_{n+l}^* - \hat{Z}_n^*(l) = \varepsilon_{n+l} + \psi_1\varepsilon_{n+l-1} + \psi_2\varepsilon_{n+l-2} + \ldots + \psi_{l-1}\varepsilon_{n+1}.$$

Let d = maximum (d_i), for $i = 1,2,\ldots,k$. Unless $q = 0$ the routine requires estimates of ε_t, for $t = d+1, d+2, \ldots, n$ which are obtainable from G13DCF. The terms ε_t are assumed to be zero, for $t = n+1, n+2, \ldots, n+l_{\mathrm{max}}$. The user may use G13DKF to update these l_{max} forecasts should further observations, $Z_{n+1}, Z_{n+2}, \ldots$ become available. Note that when l_{max} or more further observations are available then G13DJF must be used to produce new forecasts for $Z_{n+l_{\mathrm{max}}+1}, Z_{n+l_{\mathrm{max}}+2}, \ldots$ should they be required.

When a transformation has been used the forecasts and their standard errors are suitably modified to give results in terms of the original series, Z_t, see Granger and Newbold [2].

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control.
San Francisco, Holden Day, 1976.

[2] GRANGER, C.W.J. and NEWBOLD, P.
Forecasting Transformed Series.
J. R. Statist. Soc. Ser. B, 38, pp. 189-203, 1976.

[3] WEI, W.W.S.
Time Series Analysis: Univariate and Multivariate Methods.
Addison-Wesley, 1990.

5. Parameters

The output quantities, K, N, IK, IP, IQ, PAR, NPAR, QQ and V from G13DCF are suitable for input to G13DJF.

1: K – INTEGER. *Input*

On entry: the dimension, k, of the multivariate time series.

Constraint: K $\geq$ 1.

2: N – INTEGER. *Input*

On entry: the number, n, of observations in the series, Z_t, prior to differencing.

Constraint: N $\geq$ 3.

The total number of observations must exceed the total number of parameters in the model, that is:

if MEAN = 'Z', then N×K > (IP+IQ)×K×K + K×(K+1)/2;
if MEAN = 'M', then N×K > (IP+IQ)×K×K + K + K×(K+1)/2;

(see the parameters IP, IQ and MEAN).

3: Z(IK,N) – ***real*** array. *Input*

On entry: Z(i,t) must be set equal to the ith component of Z_t, for $i = 1,2,...,k$; $t = 1,2,...,n$.

Constraints: if TR(i) = 'L', then Z$(i,t) > 0.0$ and
if TR(i) = 'S', then Z$(i,t) \geq 0.0$, for $i = 1,2,...,k$; $t = 1,2,...,n$.

4: IK – INTEGER. *Input*

On entry: the first dimension of the arrays Z, QQ, DELTA, V, PREDZ and SEFZ as declared in the (sub)program from which G13DJF is called.

Constraint: IK ≥ K.

5: TR(K) – CHARACTER*1 array. *Input*

On entry: TR(i) indicates whether the ith time series is to be transformed, for $i = 1,2,...,k$.

If TR(i) = 'N', then no transformation is used;
If TR(i) = 'L', then a log transformation is used;
If TR(i) = 'S', then a square root transformation is used.

Constraint: TR(i) must equal either 'N', 'L' or 'S', for $i = 1,2,...,k$.

6: ID(K) – INTEGER. *Input*

On entry: ID(i) must specify, d_i, the order of differencing required for the ith series.

Constraint: $0 \leq$ ID(i) < N – max(IP,IQ), for $i = 1,2,...,k$.

7: DELTA(IK,*) – ***real*** array. *Input*

Note: the second dimension of the array DELTA must be at least max$(1,d)$, where d = max(ID(i)).

On entry: if ID$(i) > 0$, then DELTA(i,j) must be set equal to δ_{ij}, for $j = 1,2,...,d_i$; $i = 1,2,...,k$. If $d = 0$, then DELTA is not referenced.

8: IP – INTEGER. *Input*

On entry: the number, p, of AR parameter matrices.

Constraint: IP ≥ 0.

9: IQ – INTEGER. *Input*

On entry: the number, q, of MA parameter matrices.

Constraint: IQ ≥ 0.

10: MEAN – CHARACTER*1. *Input*

On entry: MEAN must be set to 'M' if components of μ have been estimated and 'Z' if all elements of μ are to be taken as zero.

Constraint: MEAN = 'M' or 'Z'.

11: PAR(LPAR) – ***real*** array. *Input*

On entry: PAR must contain the parameter estimates read in row by row in the order $\phi_1,\phi_2,...,\phi_p$, $\theta_1,\theta_2,...,\theta_q$, μ.

Thus, if IP > 0, then PAR$((l-1)\times k\times k+(i-1)\times k+j)$ must be set equal to an estimate of the (i,j)th element of ϕ_l, for $l = 1,2,...,p$; $i = 1,2,...,k$; $j = 1,2,...,k$.

If IQ > 0, then PAR($(p \times k \times k+(l-1) \times k \times k+(i-1) \times k+j)$) must be set equal to an estimate of the (i,j)th element of θ_l, for $l = 1,2,...,q$; $i = 1,2,...,k$; $j = 1,2,...,k$.

If MEAN has been set equal to 'M', then PAR($(p+q) \times k \times k+i$) must be set equal to an estimate of the ith component of μ, for $i = 1,2,...,k$.

Constraint: the first IP×K×K elements of PAR must satisfy the stationarity condition and the next IQ×K×K elements of PAR must satisfy the invertibility condition.

12: LPAR – INTEGER. *Input*

On entry: the dimension of the array PAR as declared in the (sub)program from which G13DJF is called.

Constraints: if MEAN = 'Z', then LPAR ≥ max(1,(IP+IQ)×K×K);
if MEAN = 'M', then LPAR ≥ (IP+IQ)×K×K + K.

13: QQ(IK,K) – ***real*** array. *Input/Output*

On entry: QQ(i,j) must contain an estimate of the (i,j)th element of Σ. The lower triangle only is needed.

Constraint: QQ must be positive-definite.

On exit: if IFAIL ≠ 1, then the upper triangle is set equal to the lower triangle.

14: V(IK,*) – ***real*** array. *Input*

Note: the second dimension of the array V must be at least max(1,N–max(ID(i))).

On entry: V(i,t) must contain an estimate of the ith component of ε_{t+d}, for $i = 1,2,...,k$; $t = 1,2,...,n-d$ where d is the maximum value of the elements of the array ID. If $q = 0$, then V is not used.

15: LMAX – INTEGER. *Input*

On entry: the number, $l_{\max}$, of forecasts required.

Constraint: LMAX ≥ 1.

16: PREDZ(IK,LMAX) – ***real*** array. *Output*

On exit: PREDZ(i,l) contains the forecast of $z_{i,n+l}$, for $i = 1,2,...,k$; $l = 1,2,...,l_{\max}$.

17: SEFZ(IK,LMAX) – ***real*** array. *Output*

On exit: SEFZ(i,l) contains an estimate of the standard error of the forecast of $z_{i,n+l}$, for $i = 1,2,...,k$; $l = 1,2,...,l_{\max}$.

18: REF(LREF) – ***real*** array. *Output*

On exit: the reference vector which may be used to update forecasts using G13DKF. The first (LMAX–1)×K×K elements contain the ψ weight matrices, $\psi_1,\psi_2,...,\psi_{l_{\max}-1}$, stored column wise. The next K×LMAX elements contain the forecasts of the transformed series $\hat{Z}^*_{n+1},\hat{Z}^*_{n+2},...,\hat{Z}^*_{n+l_{\max}}$ and the next K×LMAX contain the variances of the forecasts of the transformed variables. The last K elements are used to store the transformations for the series.

19: LREF – INTEGER. *Input*

On entry: the dimension of the array REF as declared in the (sub)program from which G13DJF is called.

Constraint: LREF ≥ (LMAX–1)×K×K + 2×K×LMAX + K.

20: WORK(LWORK) – ***real*** array. *Workspace*

21: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G13DJF is called.

Constraint: if r = maximum (IP,IQ) and d = max(ID(i)), for i = 1,2,...,k, then LWORK ≥ maximum {Kr(Kr+2), (IP+d+2)K^2 + (N+LMAX)K}.

22: IWORK(LIWORK) – INTEGER array. *Workspace*

23: LIWORK – INTEGER. *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which G13DJF is called.

Constraint: LIWORK ≥ K×max(IP,IQ).

24: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, K < 1,
or N < 3,
or IK < K,
or ID(i) < 0 for some i = 1,2,...,k,
or ID(i) ≥ N – max(IP,IQ) for some i = 1,2,...,k,
or IP < 0,
or IQ < 0,
or MEAN ≠ 'M' or 'Z',
or LPAR < (IP+IQ)×K×K + K, and MEAN = 'M',
or LPAR < (IP+IQ)×K×K and MEAN = 'Z',
or N×K ≤ (IP+IQ)×K×K + K + K(K+1)/2, and MEAN = 'M',
or N×K ≤ (IP+IQ)×K×K + K(K+1)/2 and MEAN 'Z',
or LMAX < 1,
or LREF < (LMAX–1)×K×K + 2×K×LMAX + K,
or LWORK is too small,
or LIWORK is too small.

IFAIL = 2

On entry, at least one of the first k elements of TR is not equal to 'N', 'L' or 'S'.

IFAIL = 3

On entry, one or more of the transformations requested cannot be computed; that is, the user may be trying to log or square-root a series, some of whose values are negative.

IFAIL = 4

On entry, either QQ is not positive-definite or the autoregressive parameter matrices are extremely close to or outside the stationarity region, or the moving average parameter matrices are extremely close to or outside the invertibility region. To proceed, the user must supply different parameter estimates in the arrays PAR and QQ.

IFAIL = 5

This is an unlikely exit brought about by an excessive number of iterations being needed to evaluate the eigenvalues of the matrices required to check for stationarity and invertibility, see G13DXF. All output parameters are undefined.

IFAIL = 6

This is an unlikely exit which could occur if QQ is nearly non positive-definite. In this case the standard deviations of the forecast errors may be non-positive. To proceed, the user must supply different parameter estimates in the array QQ.

IFAIL = 7

This is an unlikely exit. For one of the series, overflow will occur if the forecasts are computed. The user should check whether the transformations requested in the array TR are sensible. All output parameters are undefined.

7. Accuracy

The matrix computations are believed to be stable.

8. Further Comments

The same differencing operator does not have to be applied to all the series. For example, suppose we have $k = 2$, and wish to apply the second order differencing operator ∇^2 to the first series and the first order differencing operator ∇ to the second series:

$$w_{1t} = \nabla^2 z_{1t} = (1-B)^2 z_{1t} = (1-2B+B^2)Z_{1t} \text{ and}$$

$$w_{2t} = \nabla z_{2t} = (1-B)z_{2t}.$$

Then $d_1 = 2, d_2 = 1, d = \text{maximum } (d_1,d_2) = 2$, and

$$\text{DELTA} = \begin{bmatrix} \delta_{11} & \delta_{12} \\ \delta_{21} & \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ 1 & \end{bmatrix}.$$

Note: although differencing may already have been applied prior to the model fitting stage, the differencing parameters supplied in DELTA are part of the model definition and are still required by this routine to produce the forecasts.

This routine should not be used when the moving average parameters lie close to the boundary of the invertibility region. The routine does test for both invertibility and stationarity but if in doubt, the user may use G13DXF, before calling this routine, to check that the VARMA model being used is invertible.

On a successful exit, the output quantities K, LMAX, IK, REF and LREF will be suitable for input to G13DKF.

9. Example

A program to compute forecasts of the next five values in two series each of length 48. No transformation is to be used and no differencing is to be applied to either of the series. G13DCF is first called to fit an AR(1) model to the series. The mean vector μ is to be estimated and $\phi_1(2,1)$ constrained to be zero.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DJF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          KMAX, IK, IPMAX, IQMAX, NMAX, NPARMX, LWORK,
     +                 LIWORK, ICM, MMAX, IDMAXL, LREF
      PARAMETER        (KMAX=3,IK=KMAX,IPMAX=3,IQMAX=3,NMAX=100,
     +                 NPARMX=(IPMAX+IQMAX)*KMAX*KMAX+KMAX,LWORK=2000,
     +                 LIWORK=100,ICM=NPARMX,MMAX=10,IDMAXL=2,
     +                 LREF=MMAX*KMAX*(KMAX+2))
*     .. Local Scalars ..
      real             CGETOL, RLOGL
      INTEGER          I, I2, IDMAX, IDMIN, IFAIL, IP, IPRINT, IQ,
     +                 ISHOW, J, K, L, L2, LMAX, LOOP, MAXCAL, N, ND,
     +                 NITER, NPAR
      LOGICAL          EXACT, MEANL
      CHARACTER        MEAN
*     .. Local Arrays ..
      real             CM(ICM,NPARMX), DELTA(IK,IDMAXL), G(NPARMX),
     +                 PAR(NPARMX), PREDZ(IK,MMAX), QQ(IK,KMAX),
     +                 REF(LREF), SEFZ(IK,MMAX), V(IK,NMAX), W(IK,NMAX),
     +                 WORK(LWORK), Z(IK,NMAX)
      INTEGER          ID(KMAX), IWORK(LIWORK)
      LOGICAL          PARHLD(NPARMX)
      CHARACTER        TR(KMAX)
*     .. External Subroutines ..
      EXTERNAL         G13DCF, G13DJF, G13DLF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX, MIN, MOD
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DJF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, N, IP, IQ, MEAN, LMAX
      NPAR = (IP+IQ)*K*K
      MEANL = .FALSE.
      IF (MEAN.EQ.'M' .OR. MEAN.EQ.'m') THEN
         NPAR = NPAR + K
         MEANL = .TRUE.
      END IF
      IF (K.GT.0 .AND. K.LE.KMAX .AND. N.GE.1 .AND. N.LE.NMAX .AND.
     +    NPAR.GE.1 .AND. NPAR.LE.NPARMX .AND. LMAX.GE.1 .AND. LMAX.LE.
     +    MMAX) THEN
         READ (NIN,*) (ID(I),I=1,K)
         IDMIN = 0
         IDMAX = 0
         DO 20 I = 1, K
            IDMIN = MIN(ID(I),IDMIN)
            IDMAX = MAX(ID(I),IDMAX)
   20    CONTINUE
*
         IF (IDMIN.GE.0 .AND. IDMAX.LE.IDMAXL) THEN
            DO 40 I = 1, K
               READ (NIN,*) (Z(I,J),J=1,N)
   40       CONTINUE
            READ (NIN,*) (TR(I),I=1,K)
*
            IF (IDMAX.GT.0) THEN
               DO 60 I = 1, K
                  READ (NIN,*) (DELTA(I,J),J=1,ID(I))
   60          CONTINUE
            END IF
            IFAIL = 0
```

```
*
            CALL G13DLF(K,N,Z,IK,TR,ID,DELTA,W,ND,WORK,IFAIL)
*
            DO 80 I = 1, NPAR
               PAR(I) = 0.0e0
               PARHLD(I) = .FALSE.
   80       CONTINUE
            DO 120 I = 1, K
               DO 100 J = 1, I
                  QQ(I,J) = 0.0e0
  100          CONTINUE
  120       CONTINUE
            PARHLD(3) = .TRUE.
            EXACT = .TRUE.
*           ** Set IPRINT .lt. 0 for no monitoring
            IPRINT = -1
            CGETOL = 0.0001e0
            MAXCAL = 40*NPAR*(NPAR+5)
*           ** Set ISHOW = 0 for no printing of results from G13DCF
            ISHOW = 0
            IFAIL = 1
*
            CALL G13DCF(K,ND,IP,IQ,MEANL,PAR,NPAR,QQ,IK,W,PARHLD,EXACT,
     +                  IPRINT,CGETOL,MAXCAL,ISHOW,NITER,RLOGL,V,G,CM,
     +                  ICM,WORK,LWORK,IWORK,LIWORK,IFAIL)
*
            IF (IFAIL.EQ.0 .OR. IFAIL.GE.4) THEN
               IFAIL = 0
*
               CALL G13DJF(K,N,Z,IK,TR,ID,DELTA,IP,IQ,MEAN,PAR,NPAR,QQ,
     +                     V,LMAX,PREDZ,SEFZ,REF,LREF,WORK,LWORK,IWORK,
     +                     LIWORK,IFAIL)
*
               WRITE (NOUT,*)
               WRITE (NOUT,*) ' FORECAST SUMMARY TABLE'
               WRITE (NOUT,*) ' ----------------------'
               WRITE (NOUT,*)
               WRITE (NOUT,99998) ' Forecast origin is set at t = ', N
               WRITE (NOUT,*)
               LOOP = LMAX/5
               IF (MOD(LMAX,5).NE.0) LOOP = LOOP + 1
               DO 160 J = 1, LOOP
                  I2 = (J-1)*5
                  L2 = MIN(I2+5,LMAX)
                  WRITE (NOUT,99997) 'Lead Time ', (I,I=I2+1,L2)
                  WRITE (NOUT,*)
                  I = 1
                  WRITE (NOUT,99996) 'Series ', I, ' : Forecast    ',
     +              (PREDZ(1,L),L=I2+1,L2)
                  WRITE (NOUT,99995) ' : Standard Error ',
     +              (SEFZ(1,L),L=I2+1,L2)
                  DO 140 I = 2, K
                     WRITE (NOUT,99996) 'Series ', I, ' : Forecast    ',
     +                 (PREDZ(I,L),L=I2+1,L2)
                     WRITE (NOUT,99995) ' : Standard Error ',
     +                 (SEFZ(I,L),L=I2+1,L2)
  140             CONTINUE
                  WRITE (NOUT,*)
  160          CONTINUE
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I2,A)
99998 FORMAT (1X,A,I4)
99997 FORMAT (1X,A,12X,5I10)
99996 FORMAT (1X,A,I2,A,5F10.2)
99995 FORMAT (10X,A,4(F7.2,3X),F7.2)
      END
```

9.2. Program Data

```
G13DJF Example Program Data
2 48 1 0 'M' 5  :  K, N, IP, IQ, MEAN, M
0 0             :  ID(I),I=1,K
-1.490 -1.620  5.200  6.230  6.210  5.860  4.090  3.180
 2.620  1.490  1.170  0.850 -0.350  0.240  2.440  2.580
 2.040  0.400  2.260  3.340  5.090  5.000  4.780  4.110
 3.450  1.650  1.290  4.090  6.320  7.500  3.890  1.580
 5.210  5.250  4.930  7.380  5.870  5.810  9.680  9.070
 7.290  7.840  7.550  7.320  7.970  7.760  7.000  8.350
 7.340  6.350  6.960  8.540  6.620  4.970  4.550  4.810
 4.750  4.760 10.880 10.010 11.620 10.360  6.400  6.240
 7.930  4.040  3.730  5.600  5.350  6.810  8.270  7.680
 6.650  6.080 10.250  9.140 17.750 13.300  9.630  6.800
 4.080  5.060  4.940  6.650  7.940 10.760 11.890  5.850
 9.010  7.500 10.020 10.380  8.150  8.370 10.730 12.140 : End of time series
 'N'  'N'       :  TR(1), TR(2)
```

9.3. Program Results

```
 G13DJF Example Program Results

  FORECAST SUMMARY TABLE
  ----------------------

  Forecast origin is set at t =   48

 Lead Time                        1         2         3         4         5

 Series  1 : Forecast           7.82      7.28      6.77      6.33      5.95
           : Standard Error     1.72      2.23      2.51      2.68      2.79
 Series  2 : Forecast          10.31      9.25      8.65      8.30      8.10
           : Standard Error     2.32      2.68      2.78      2.82      2.83
```

G13DKF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DKF accepts a sequence of new observations in a multivariate time series and updates both the forecasts and the standard deviations of the forecast errors. A call to G13DJF must be made prior to calling this routine in order to calculate the elements of a reference vector together with a set of forecasts and their standard errors. On a successful exit from G13DKF the reference vector is updated so that should future series values become available these forecasts may be updated by recalling G13DKF.

2. Specification

```
      SUBROUTINE G13DKF (K, LMAX, M, MLAST, Z, IK, REF, LREF, V, PREDZ,
     1                   SEFZ, WORK, IFAIL)
      INTEGER        K, LMAX, M, MLAST, IK, LREF, IFAIL
      real           Z(IK,M), REF(LREF), V(IK,M), PREDZ(IK,LMAX),
     1               SEFZ(IK,LMAX), WORK(K*LMAX)
```

3. Description

Let $Z_t = (z_{1t},z_{2t},...,z_{kt})^T$, for $t = 1,2,...,n$ denote a k-dimensional time series for which forecasts of $\hat{Z}_{n+1},\hat{Z}_{n+2},...,\hat{Z}_{n+l_{\max}}$ have been computed using G13DJF. Given m further observations $Z_{n+1},Z_{n+2},...,Z_{n+m}$, where $m < l_{\max}$, the routine updates the forecasts of $Z_{n+m+1},Z_{n+m+2},...,Z_{n+l_{\max}}$ and their corresponding standard errors.

The routine uses a multivariate version of the procedure described in Box and Jenkins [1]. The forecasts are updated using the ψ weights, computed in G13DJF. If Z_t^* denotes the transformed value of Z_t and $\hat{Z}_t^*(l)$ denotes the forecast of Z_{t+l}^* from time t with a lead of l (that is the forecast of Z_{t+l}^* given observations $Z_t^*,Z_{t-1}^*,...$); then

$$\hat{Z}_{t+1}^*(l) = \tau + \psi_l\epsilon_{t+1} + \psi_{l+1}\epsilon_t + \psi_{l+2}\epsilon_{t-1} + ...$$

and

$$\hat{Z}_t^*(l+1) = \tau + \psi_{l+1}\epsilon_t + \psi_{l+2}\epsilon_{t-1} + ...$$

where τ is a constant vector of length k involving the differencing parameters and the mean vector μ. By subtraction we obtain

$$\hat{Z}_{t+1}^*(l) = \hat{Z}_t^*(l+1) + \psi_l\epsilon_{t+1}.$$

Estimates of the residuals corresponding to the new observations are also computed as $\epsilon_{n+l} = Z_{n+l}^* - \hat{Z}_n^*(l)$, for $l = 1,2,...,m$. These may be of use in checking that the new observations conform to the previously fitted model.

On a successful exit, the reference array is updated so that G13DKF may be called again should future series values become available, see Section 8.

When a transformation has been used the forecasts and their standard errors are suitably modified to give results in terms of the original series Z_t, see Granger and Newbold [2].

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control.
San Francisco, Holden Day, 1976.

[2] GRANGER, C.W.J. and NEWBOLD, P.
Forecasting Transformed Series.
J. R. Statist. Soc. Ser. B, 38, pp. 189-203, 1976.

[3] WEI, W.W.S.
Time Series Analysis: Univariate and Multivariate Methods.
Addison-Wesley, 1990.

5. Parameters

The quantities K, LMAX, IK, REF and LREF output from G13DJF are suitable for input to G13DKF.

1: K – INTEGER. *Input*

On entry: the dimension, k, of the multivariate time series.

Constraint: K $\geq$ 1.

2: LMAX – INTEGER. *Input*

On entry: the number, $l_{\max}$, of forecasts requested in the call to G13DJF.

Constraint: LMAX $\geq$ 2.

3: M – INTEGER. *Input*

On entry: the number, m, of new observations available since the last call to either G13DJF or G13DKF. The number of new observations since the last call to G13DJF is then M + MLAST.

Constraint: 0 < M < LMAX – MLAST.

4: MLAST – INTEGER. *Input/Output*

On entry: on the first call to G13DKF, since calling G13DJF, MLAST must be set to 0 to indicate that no new observations have yet been used to update the forecasts; on subsequent calls MLAST must contain the value of MLAST as output on the previous call to G13DKF.

On exit: MLAST is incremented by m to indicate that MLAST + M observations have now been used to update the forecasts since the last call to G13DJF.

MLAST must not be changed between calls to G13DKF, unless a call to G13DJF has been made between the calls in which case MLAST should be reset to 0.

Constraint: 0 $\leq$ MLAST < LMAX – M.

5: Z(IK,M) – ***real*** array. *Input*

On entry: Z(i,j) must contain the value of $z_{i,n+\mathrm{MLAST}+j}$, for $i = 1,2,\ldots,k$; $j = 1,2,\ldots,m$ where n is the number of observations in the time series in the last call made to G13DJF.

Constraint: if the transformation defined in TR in G13DJF for the ith series is the log transformation, then Z(i,j) > 0.0, and if it is the square-root transformation, then Z(i,j) $\geq$ 0.0, for $j = 1,2,\ldots,m$; $i = 1,2,\ldots,k$.

6: IK – INTEGER. *Input*

On entry: the first dimension of the arrays Z, PREDZ and SEFZ as declared in the (sub)program from which G13DKF is called.

Constraint: IK $\geq$ K.

7: REF(LREF) – ***real*** array. *Input/Output*

On entry: REF must contain the first (LMAX–1)×K×K + 2×K×LMAX + K elements of the reference vector as returned on a successful exit from G13DJF (or a previous call to G13DKF).

On exit: the elements of REF are updated. The first (LMAX–1)×K×K elements store the ψ weights $\psi_1,\psi_2,\ldots,\psi_{l_{\max}-1}$, stored column wise. The next K×LMAX elements contain the forecasts of the transformed series and the next K×LMAX elements contain the variances of the forecasts of the transformed variables, see G13DJF. The last K elements are not updated.

8: LREF – INTEGER. *Input*

On entry: the dimension of the array REF as declared in the (sub)program from which G13DKF is called.

Constraint: LREF $\geq$ (LMAX−1)×K×K + 2×K×LMAX + K.

9: V(IK,M) – ***real*** array. *Output*

On exit: V(i,j) contains an estimate of the ith component of $\varepsilon_{n+\mathrm{MLAST}+j}$, for $i = 1,2,...,k$; $j = 1,2,...,m$.

10: PREDZ(IK,LMAX) – ***real*** array. *Output*

On exit: PREDZ(i,j) contains the updated forecast of $z_{i,n+j}$, for $i = 1,2,...,k$; j = MLAST+M+1,MLAST+M+2,...,$l_{\max}$.

The columns of PREDZ corresponding to the new observations since the last call to either G13DJF or G13DKF are set equal to the corresponding columns of Z.

11: SEFZ(IK,LMAX) – ***real*** array. *Output*

On exit: SEFZ(i,j) contains an estimate of the standard error of the corresponding element of PREDZ, for $i = 1,2,...,k$; j = MLAST+M+1,MLAST+M+2,...,$l_{\max}$.

The columns of SEFZ corresponding to the new observations since the last call to either G13DJF or G13DKF are set equal to zero.

12: WORK(K*M) – ***real*** array. *Workspace*

13: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, K < 1,
or LMAX < 2,
or M $\leq$ 0,
or MLAST + M $\geq$ LMAX,
or MLAST < 0,
or IK < K,
or LREF < (LMAX−1)×K×K + 2×K×LMAX + K.

IFAIL = 2

On entry, some of the elements of the reference vector, REF, have been corrupted since the most recent call to G13DJF (or G13DKF).

IFAIL = 3

On entry, one or more of the elements of Z is invalid, for the transformation being used; that is the user may be trying to log or square root a series, some of whose values are negative.

IFAIL = 4

This is an unlikely exit. For one of the series, overflow will occur if the forecasts are updated. The user should check whether the elements of REF have been corrupted.

7. Accuracy

The matrix computations are believed to be stable.

8. Further Comments

If a further m^* obervations, $Z_{n+\mathrm{MLAST}+1},Z_{n+\mathrm{MLAST}+2},...,Z_{n+\mathrm{MLAST}+m^*}$, become available, then forecasts of $Z_{n+\mathrm{MLAST}+m^*+1},Z_{n+\mathrm{MLAST}+m^*+2},...,Z_{n+l_{\max}}$ may be updated by recalling G13DKF with M = m^*. Note that M and the contents of the array Z are the only quantities which need updating; MLAST is updated on exit from the previous call. On a successful exit, V contains estimates of $\varepsilon_{n+\mathrm{MLAST}+1},\varepsilon_{n+\mathrm{MLAST}+2},...,\varepsilon_{n+\mathrm{MLAST}+m^*}$; columns MLAST+1, MLAST+2,..., MLAST+m^* of PREDZ contain the new observed values $Z_{n+\mathrm{MLAST}+1},Z_{n+\mathrm{MLAST}+2},...,Z_{n+\mathrm{MLAST}+m^*}$ and columns MLAST+1, MLAST+2,..., MLAST+m^* of SEFZ are set to zero.

9. Example

A program to update the forecasts of two series each of length 48. No transformation has been used and no differencing applied to either of the series. G13DCF is first called to fit an AR(1) model to the series. μ is to be estimated and $\phi_1(2,1)$ constrained to be zero. A call to G13DJF is then made in order to compute forecasts of the next five series values. After one new observation becomes available the four forecasts are updated. A further observation becomes available and the three forecasts are updated.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DKF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          KMAX, IK, IPMAX, IQMAX, NMAX, NPARMX, ICM, LWORK,
     +                 LIWORK, IDMAXL, LLMAX, LREF
      PARAMETER        (KMAX=3,IK=KMAX,IPMAX=3,IQMAX=3,NMAX=100,
     +                 NPARMX=(IPMAX+IQMAX)*KMAX*KMAX+KMAX,ICM=NPARMX,
     +                 LWORK=2000,LIWORK=100,IDMAXL=2,LLMAX=10,
     +                 LREF=(LLMAX-1)*KMAX*KMAX+2*KMAX*LLMAX+KMAX)
*     .. Local Scalars ..
      real             CGETOL, RLOGL
      INTEGER          I, IDMAX, IDMIN, IFAIL, IP, IPRINT, IQ, ISHOW, J,
     +                 K, LMAX, LPAR, M, MAXCAL, MLAST, N, ND, NITER
      LOGICAL          EXACT, MEANL
      CHARACTER        MEAN
*     .. Local Arrays ..
      real             CM(ICM,NPARMX), DELTA(IK,IDMAXL), G(NPARMX),
     +                 PAR(NPARMX), PREDZ(IK,LLMAX), QQ(IK,KMAX),
     +                 REF(LREF), SEFZ(IK,LLMAX), V(IK,NMAX),
     +                 W(IK,NMAX), WORK(LWORK), Z(IK,NMAX)
      INTEGER          ID(KMAX), IWORK(LIWORK)
      LOGICAL          PARHLD(NPARMX)
      CHARACTER        TR(KMAX)
*     .. External Subroutines ..
      EXTERNAL         FPRINT, G13DCF, G13DJF, G13DKF, G13DLF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX, MIN
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DKF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, N, IP, IQ, MEAN, LMAX
      MEANL = .FALSE.
      LPAR = (IP+IQ)*K*K
      IF (MEAN.EQ.'M' .OR. MEAN.EQ.'m') THEN
         LPAR = LPAR + K
         MEANL = .TRUE.
      END IF
      IF (K.GT.0 .AND. K.LE.KMAX .AND. N.GE.1 .AND. N.LE.NMAX .AND.
     +    LPAR.GE.1 .AND. LPAR.LE.NPARMX .AND. LMAX.GE.1 .AND. LMAX.LE.
     +    LLMAX) THEN
         READ (NIN,*) (ID(I),I=1,K)
         IDMAX = 0
         IDMIN = 0
         DO 20 I = 1, K
            IDMIN = MIN(ID(I),IDMIN)
            IDMAX = MAX(ID(I),IDMAX)
   20    CONTINUE
         IF (IDMIN.GE.0 .AND. IDMAX.LE.IDMAXL) THEN
            DO 40 I = 1, K
               READ (NIN,*) (Z(I,J),J=1,N)
   40       CONTINUE
            READ (NIN,*) (TR(I),I=1,K)
            IF (IDMAX.GT.0) THEN
               DO 60 I = 1, K
                  READ (NIN,*) (DELTA(I,J),J=1,ID(I))
   60          CONTINUE
            END IF
            IFAIL = 0
*
            CALL G13DLF(K,N,Z,IK,TR,ID,DELTA,W,ND,WORK,IFAIL)
*
            DO 80 I = 1, LPAR
               PAR(I) = 0.0e0
               PARHLD(I) = .FALSE.
   80       CONTINUE
            DO 120 J = 1, K
               DO 100 I = J, K
                  QQ(I,J) = 0.0e0
  100          CONTINUE
  120       CONTINUE
            PARHLD(3) = .TRUE.
            EXACT = .TRUE.
*           ** Set IPRINT.gt.0 for no intermediate monitoring
            IPRINT = -1
            CGETOL = 0.0001e0
            MAXCAL = 40*LPAR*(LPAR+5)
*           ** Set ISHOW.eq.0 for no results from G13DCF
            ISHOW = 0
            IFAIL = 1
*
            CALL G13DCF(K,ND,IP,IQ,MEANL,PAR,LPAR,QQ,IK,W,PARHLD,EXACT,
     +                  IPRINT,CGETOL,MAXCAL,ISHOW,NITER,RLOGL,V,G,CM,
     +                  ICM,WORK,LWORK,IWORK,LIWORK,IFAIL)
*
            IF (IFAIL.EQ.0 .OR. IFAIL.GE.4) THEN
               IFAIL = 0
*
               CALL G13DJF(K,N,Z,IK,TR,ID,DELTA,IP,IQ,MEAN,PAR,LPAR,QQ,
     +                     V,LMAX,PREDZ,SEFZ,REF,LREF,WORK,LWORK,IWORK,
     +                     LIWORK,IFAIL)
*
               CALL FPRINT(K,N,LMAX,PREDZ,SEFZ,IK,NOUT)
               M = 1
               MLAST = 0
               Z(1,1) = 8.1e0
               Z(2,1) = 10.2e0
               IFAIL = 0
*
```

```
               CALL G13DKF(K,LMAX,M,MLAST,Z,IK,REF,LREF,V,PREDZ,SEFZ,
     +                     WORK,IFAIL)
*
               CALL FPRINT(K,N+MLAST,LMAX,PREDZ,SEFZ,IK,NOUT)
               M = 1
*              Leave MLAST unchanged from last call
               Z(1,1) = 8.5e0
               Z(2,1) = 10.0e0
               IFAIL = 0
*
               CALL G13DKF(K,LMAX,M,MLAST,Z,IK,REF,LREF,V,PREDZ,SEFZ,
     +                     WORK,IFAIL)
*
               CALL FPRINT(K,N+MLAST,LMAX,PREDZ,SEFZ,IK,NOUT)
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I2,A)
      END
*
      SUBROUTINE FPRINT(K,NM,LMAX,PREDZ,SEFZ,IK,NOUT)
*     .. Scalar Arguments ..
      INTEGER           IK, K, LMAX, NM, NOUT
*     .. Array Arguments ..
      real              PREDZ(IK,LMAX), SEFZ(IK,LMAX)
*     .. Local Scalars ..
      INTEGER           I, I2, J, L, L2, LOOP
*     .. Intrinsic Functions ..
      INTRINSIC         MIN, MOD
*     .. Executable Statements ..
      WRITE (NOUT,*)
      WRITE (NOUT,*) ' FORECAST SUMMARY TABLE'
      WRITE (NOUT,*) ' ----------------------'
      WRITE (NOUT,*)
      WRITE (NOUT,99999) ' Forecast origin is set at t = ', NM
      WRITE (NOUT,*)
      LOOP = LMAX/5
      IF (MOD(LMAX,5).NE.0) LOOP = LOOP + 1
      DO 40 J = 1, LOOP
         I2 = (J-1)*5
         L2 = MIN(I2+5,LMAX)
         WRITE (NOUT,99998) 'Lead Time ', (I,I=I2+1,L2)
         WRITE (NOUT,*)
         I = 1
         WRITE (NOUT,99997) 'Series ', I, ' : Forecast    ',
     +     (PREDZ(1,L),L=I2+1,L2)
         WRITE (NOUT,99996) ' : Standard Error ', (SEFZ(1,L),L=I2+1,L2)
         DO 20 I = 2, K
            WRITE (NOUT,99997) 'Series ', I, ' : Forecast    ',
     +        (PREDZ(I,L),L=I2+1,L2)
            WRITE (NOUT,99996) ' : Standard Error ',
     +        (SEFZ(I,L),L=I2+1,L2)
   20    CONTINUE
         WRITE (NOUT,*)
   40 CONTINUE
      RETURN
*
99999 FORMAT (1X,A,I4)
99998 FORMAT (1X,A,12X,5I10)
99997 FORMAT (1X,A,I2,A,5F10.2)
99996 FORMAT (10X,A,4(F7.2,3X),F7.2)
      END
```

9.2. Program Data

```
G13DKF Example Program Data
2 48 1 0 'M' 5   :  K, N, IP, IQ, MEAN, LMAX
0  0             :  ID(I),I=1,K
-1.490 -1.620  5.200  6.230  6.210  5.860  4.090  3.180
 2.620  1.490  1.170  0.850 -0.350  0.240  2.440  2.580
 2.040  0.400  2.260  3.340  5.090  5.000  4.780  4.110
 3.450  1.650  1.290  4.090  6.320  7.500  3.890  1.580
 5.210  5.250  4.930  7.380  5.870  5.810  9.680  9.070
 7.290  7.840  7.550  7.320  7.970  7.760  7.000  8.350
 7.340  6.350  6.960  8.540  6.620  4.970  4.550  4.810
 4.750  4.760 10.880 10.010 11.620 10.360  6.400  6.240
 7.930  4.040  3.730  5.600  5.350  6.810  8.270  7.680
 6.650  6.080 10.250  9.140 17.750 13.300  9.630  6.800
 4.080  5.060  4.940  6.650  7.940 10.760 11.890  5.850
 9.010  7.500 10.020 10.380  8.150  8.370 10.730 12.140 : End of time series
'N'  'N'         :  TR(1), TR(2)
```

9.3. Program Results

```
 G13DKF Example Program Results

  FORECAST SUMMARY TABLE
  ----------------------

  Forecast origin is set at t =   48

 Lead Time                    1         2         3         4         5

 Series  1 : Forecast        7.82      7.28      6.77      6.33      5.95
           : Standard Error  1.72      2.23      2.51      2.68      2.79
 Series  2 : Forecast       10.31      9.25      8.65      8.30      8.10
           : Standard Error  2.32      2.68      2.78      2.82      2.83

  FORECAST SUMMARY TABLE
  ----------------------

  Forecast origin is set at t =   49

 Lead Time                    1         2         3         4         5

 Series  1 : Forecast        8.10      7.49      6.94      6.46      6.06
           : Standard Error  0.00      1.72      2.23      2.51      2.68
 Series  2 : Forecast       10.20      9.19      8.61      8.28      8.08
           : Standard Error  0.00      2.32      2.68      2.78      2.82

  FORECAST SUMMARY TABLE
  ----------------------

  Forecast origin is set at t =   50

 Lead Time                    1         2         3         4         5

 Series  1 : Forecast        8.10      8.50      7.80      7.18      6.65
           : Standard Error  0.00      0.00      1.72      2.23      2.51
 Series  2 : Forecast       10.20     10.00      9.08      8.54      8.24
           : Standard Error  0.00      0.00      2.32      2.68      2.78
```

G13DLF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DLF differences and/or transforms a multivariate time series. It is intended to be used prior to calling NAG Fortran Library routine G13DCF to fit a vector autoregressive moving average (VARMA) model to the differenced/transformed series.

2. Specification

```
SUBROUTINE G13DLF (K, N, Z, IK, TR, ID, DELTA, W, ND, WORK, IFAIL)
INTEGER        K, N, IK, ID(K), ND, IFAIL
real           Z(IK,N), DELTA(IK,*), W(IK,*), WORK(K*N)
CHARACTER*1    TR(K)
```

3. Description

For certain time series it may first be necessary to difference the original data to obtain a stationary series before fitting a VARMA model. This routine also allows the user to apply either a square root or a log transformation to the original time series to stabalize the variance if required.

If the order of differencing required for the ith series is d_i, then the differencing operator is defined by $\delta_i(B) = 1 - \delta_{i1}B - \delta_{i2}B^2 - \ldots - \delta_{id_i}B^{d_i}$ where B is the backward shift operator, that is $BZ_t = Z_{t-1}$. Let d denote the maximum of the orders of differencing, d_i, over the k series. The routine computes values of the differenced/transformed series $W_t = (w_{1t},w_{2t},\ldots,w_{kt})^T$ for $t = d+1,d+2,\ldots,n$ as follows;

$$w_{it} = \delta_i(B)\; z_{it}^*, \quad \text{for } i = 1,2,\ldots,k$$

where z_{it}^* are the transformed values of the original k-dimensional time series $Z_t = (z_{1t},z_{2t},\ldots,z_{kt})^T$.

The differencing parameters δ_{ij}, for $i = 1,2,\ldots,k$; $j = 1,2,\ldots,d_i$ must be supplied by the user. If the ith series does not require differencing, then $d_i = 0$.

4. References

[1] BOX, G.E.P. and JENKINS, G.M.
Time Series Analysis: Forecasting and Control.
Holden-Day, 1976.

[2] WEI, W.W.S.
Time Series Analysis: Univariate and Multivariate Methods.
Addison-Wesley, 1990.

5. Parameters

1: K – INTEGER. *Input*

On entry: the dimension, k, of the multivariate time series.

Constraint: K $\geq$ 1.

2: N – INTEGER. *Input*

On entry: the number, n, of observations in the series, prior to differencing.

Constraint: N $\geq$ 1.

3: Z(IK,N) – ***real*** array. *Input*

On entry: Z(i,t) must contain, z_{it}, the ith component of Z_t, for $i = 1,2,...,k$; $t = 1,2,...,n$.

Constraints: if TR(i) = 'L', then Z$(i,t) > 0.0$ and
if TR(i) = 'S', then Z$(i,t) \geq 0.0$, for $i = 1,2,...,k$; $t = 1,2,...,n$.

4: IK – INTEGER. *Input*

On entry: the first dimension of the arrays Z, DELTA and W as declared in the (sub)program from which G13DLF is called.

Constraint: IK $\geq$ K.

5: TR(K) – CHARACTER*1 array. *Input*

On entry: TR(i) indicates whether the ith time series is to be transformed, for $i = 1,2,...,k$.

If TR(i) = 'N', then no transformation is used;
If TR(i) = 'L', then a log transformation is used;
If TR(i) = 'S', then a square root transformation is used.

Constraint: TR(i) must equal either 'N', 'L' or 'S', for $i = 1,2,...,k$.

6: ID(K) – INTEGER array. *Input*

On entry: the order of differencing for each series, $d_1,d_2,...,d_k$.

Constraint: $0 \leq$ ID$(i) <$ N, for $i = 1,2,...,k$.

7: DELTA(IK,*) – ***real*** array. *Input*

Note: the second dimension of the array DELTA must be at least max$(1,d)$ where d = max(ID(i)).

On entry: if ID$(i) > 0$, then DELTA(i,j) must be set equal to δ_{ij}, for $j = 1,2,...,d_i$; $i = 1,2,...,k$. If $d = 0$, then DELTA is not referenced.

8: W(IK,*) – ***real*** array. *Output*

Note: the second dimension of the array W must be at least N$-d$.

On exit: W(i,t) contains the value of $w_{i,t+d}$, for $i = 1,2,...,k$; $t = 1,2,...,n-d$.

9: ND – INTEGER. *Output*

On exit: the number of differenced values, $n-d$, in the series.

10: WORK(K*N) – ***real*** array. *Workspace*

11: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, K < 1,
or N < 1,
or IK < K.

IFAIL = 2

On entry, ID(i) < 0, for some i = 1,2,...,k,
or ID(i) ≥ N, for some i = 1,2,...,k.

IFAIL = 3

On entry, at least one of the first k elements of TR is not equal to 'N', 'L' or 'S'.

IFAIL = 4

On entry, one or more of the elements of Z is invalid, for the transformation requested; that is the user may be trying to log or square root a series, some of whose values are negative.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The same differencing operator does not have to be applied to all the series. For example, suppose we have $k = 2$, and wish to apply the second order differencing operator ∇^2 to the first series and the first order differencing operator ∇ to the second series:

$$w_{1t} = \nabla^2 z_{1t} = (1-B)^2 z_{1t} = (1-2B+B^2)z_{1t}, \text{ and}$$

$$w_{2t} = \nabla z_{2t} = (1-B)z_{2t}.$$

Then $d_1 = 2$, $d_2 = 1$, d = maximum $(d_1,d_2) = 2$, and

$$\text{DELTA} = \begin{bmatrix} \delta_{11} & \delta_{12} \\ \delta_{21} & \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ 1 & \end{bmatrix}.$$

9. Example

A program to difference (nonseasonally) each of two time series of length 48. No transformation is to be applied to either of the series.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DLF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          KMAX, NMAX, IK, IDMAX
      PARAMETER        (KMAX=3,NMAX=100,IK=KMAX,IDMAX=2)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, K, MAXD, MIND, N, ND
*     .. Local Arrays ..
      real             DELTA(IK,IDMAX), W(IK,NMAX), WORK(KMAX*NMAX),
     +                 Z(IK,NMAX)
      INTEGER          ID(KMAX)
      CHARACTER        TR(KMAX)
*     .. External Subroutines ..
      EXTERNAL         G13DLF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX, MIN
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DLF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, N
      IF (K.GT.0 .AND. K.LE.KMAX .AND. N.GT.0 .AND. N.LE.NMAX) THEN
         READ (NIN,*) (ID(I),I=1,K)
         MIND = 0
         MAXD = 0
         DO 20 I = 1, K
            MIND = MIN(MIND,ID(I))
            MAXD = MAX(MAXD,ID(I))
   20    CONTINUE
         IF (MIND.GE.0 .AND. MAXD.LE.IDMAX) THEN
            DO 40 I = 1, K
               READ (NIN,*) (Z(I,J),J=1,N)
   40       CONTINUE
            READ (NIN,*) (TR(I),I=1,K)
            IF (MAXD.GT.0) THEN
               DO 60 I = 1, K
                  READ (NIN,*) (DELTA(I,J),J=1,ID(I))
   60          CONTINUE
            END IF
            IFAIL = 0
*
            CALL G13DLF(K,N,Z,IK,TR,ID,DELTA,W,ND,WORK,IFAIL)
*
            WRITE (NOUT,*)
            WRITE (NOUT,*) ' Transformed/Differenced series'
            WRITE (NOUT,*) ' ------------------------------'
            DO 80 I = 1, K
               WRITE (NOUT,*)
               WRITE (NOUT,99999) ' Series ', I
               WRITE (NOUT,*) ' -----------'
               WRITE (NOUT,*)
               WRITE (NOUT,99998) ' Number of differenced values = ', ND
               WRITE (NOUT,*)
               WRITE (NOUT,99997) (W(I,J),J=1,ND)
   80       CONTINUE
*
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I2)
99998 FORMAT (1X,A,I6)
99997 FORMAT (1X,8F9.3)
      END
```

9.2. Program Data

```
G13DLF Example Program Data
2 48 1  :  K, N
 1  1   :  ID(1), ID(2)
-1.490 -1.620  5.200  6.230  6.210  5.860  4.090  3.180
 2.620  1.490  1.170  0.850 -0.350  0.240  2.440  2.580
 2.040  0.400  2.260  3.340  5.090  5.000  4.780  4.110
 3.450  1.650  1.290  4.090  6.320  7.500  3.890  1.580
 5.210  5.250  4.930  7.380  5.870  5.810  9.680  9.070
 7.290  7.840  7.550  7.320  7.970  7.760  7.000  8.350
 7.340  6.350  6.960  8.540  6.620  4.970  4.550  4.810
 4.750  4.760 10.880 10.010 11.620 10.360  6.400  6.240
 7.930  4.040  3.730  5.600  5.350  6.810  8.270  7.680
 6.650  6.080 10.250  9.140 17.750 13.300  9.630  6.800
 4.080  5.060  4.940  6.650  7.940 10.760 11.890  5.850
 9.010  7.500 10.020 10.380  8.150  8.370 10.730 12.140 : End of time series
 'N'  'N'      :  TR(1), TR(2)
 1.0   :  delta(1,1)
 1.0   :  delta(2,1)
```

9.3. Program Results

```
G13DLF Example Program Results

  Transformed/Differenced series
  ------------------------------

  Series  1
  -----------

  Number of differenced values =      47

    -0.130    6.820    1.030   -0.020   -0.350   -1.770   -0.910   -0.560
    -1.130   -0.320   -0.320   -1.200    0.590    2.200    0.140   -0.540
    -1.640    1.860    1.080    1.750   -0.090   -0.220   -0.670   -0.660
    -1.800   -0.360    2.800    2.230    1.180   -3.610   -2.310    3.630
     0.040   -0.320    2.450   -1.510   -0.060    3.870   -0.610   -1.780
     0.550   -0.290   -0.230    0.650   -0.210   -0.760    1.350

  Series  2
  -----------

  Number of differenced values =      47

    -0.990    0.610    1.580   -1.920   -1.650   -0.420    0.260   -0.060
     0.010    6.120   -0.870    1.610   -1.260   -3.960   -0.160    1.690
    -3.890   -0.310    1.870   -0.250    1.460    1.460   -0.590   -1.030
    -0.570    4.170   -1.110    8.610   -4.450   -3.670   -2.830   -2.720
     0.980   -0.120    1.710    1.290    2.820    1.130   -6.040    3.160
    -1.510    2.520    0.360   -2.230    0.220    2.360    1.410
```

G13DMF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DMF calculates the sample cross-correlation (or cross-covariance) matrices of a multivariate time series.

2. Specification

```
SUBROUTINE G13DMF (MATRIX, K, N, M, W, IK, WMEAN, R0, R, IFAIL)
INTEGER        K, N, M, IK, IFAIL
real           W(IK,N), WMEAN(K), R0(IK,K), R(IK,IK,M)
CHARACTER*1    MATRIX
```

3. Description

Let $W_t = (w_{1t},w_{2t},...,w_{kt})^T$, for $t = 1,2,...,n$ denote n observations of a vector of k time series. The sample cross-covariance matrix at lag l is defined to be the k by k matrix $\hat{C}(l)$, whose (i,j)th element is given by

$$\hat{C}_{ij}(l) = \frac{1}{n}\sum_{t=l+1}^{n}(w_{i(t-l)}-\bar{w}_i)(w_{jt}-\bar{w}_j), \qquad \text{for } l = 0,1,2,...,m;\ i = 1,2,...,k;\ j = 1,2,...,k,$$

where $\bar{w}_i$ and $\bar{w}_j$ denote the sample means for the ith and jth series respectively. The sample cross-correlation matrix at lag l is defined to be the k by k matrix $\hat{R}(l)$, whose (i,j)th element is given by

$$\hat{R}_{ij}(l) = \frac{\hat{C}_{ij}(l)}{\sqrt{\hat{C}_{ii}(0)\hat{C}_{jj}(0)}}, \qquad \text{for } l = 0,1,2,...,m;\ i = 1,2,...,k;\ j = 1,2,...,k.$$

The number of lags, m, is usually taken to be at most $n/4$.

If W_t follows a vector moving average model of order q, then it can be shown that the theoretical cross-correlation matrices $(R(l))$ are zero beyond lag q. In order to help spot a possible cut-off point, the elements of $\hat{R}(l)$ are usually compared to their approximate standard error of $1/\sqrt{n}$. For further details see, for example, Wei [1].

The routine uses a single pass through the data to compute the means and the cross-covariance matrix at lag zero. The cross-covariance matrices at further lags are then computed on a second pass through the data.

4. References

[1] WEI, W.W.S.
Time Series Analysis: Univariate and Multivariate Methods.
Addison-Wesley, 1990.

[2] WEST, D.H.D.
Updating Mean and Variance Estimates: An Improved Method.
Comm. ACM, 22, 9, pp. 532-535, 1979.

5. Parameters

1: MATRIX – CHARACTER*1. *Input*

On entry: indicates whether the cross-covariance or cross-correlation matrices are to be computed;

If MATRIX = 'V', then the cross-covariance matrices are computed, If MATRIX = 'R', then the cross-correlation matrices are computed.

Constraint: MATRIX = 'V' or 'R'.

2: K – INTEGER. *Input*

On entry: the dimension, k, of the multivariate time series.

Constraint: K $\geq$ 1.

3: N – INTEGER. *Input*

On entry: the number of observations in the series, n.

Constraint: N $\geq$ 2.

4: M – INTEGER. *Input*

On entry: the number, m, of cross-correlation (or cross-covariance) matrices to be computed. If in doubt set M = 10. However it should be noted that M is usually taken to be at most N/4.

Constraint: 1 $\leq$ M < N.

5: W(IK,N) – ***real*** array. *Input*

On entry: W(i,t) must contain the observation w_{it}, for $i = 1,2,...,k$; $t = 1,2,...,n$.

6: IK – INTEGER. *Input*

On entry: the first dimension of the arrays W and R0 and the first and second dimensions of the array R as declared in the (sub)program from which G13DMF is called.

Constraint: IK $\geq$ K.

7: WMEAN(K) – ***real*** array. *Output*

On exit: the means, $\bar{w}_i$, for $i = 1,2,...,k$.

8: R0(IK,K) – ***real*** array. *Output*

On exit: if $i \neq j$, then R0(i,j) contains an estimate of the (i,j)th element of the cross-correlation (or cross-covariance) matrix at lag zero, $\hat{R}_{ij}(0)$; if $i = j$, then if MATRIX = 'V', R0(i,i) contains the variance of the ith series, $\hat{C}_{ii}(0)$, and if MATRIX = 'R', R0(i,i) contains the standard deviation of the ith series, $\sqrt{\hat{C}_{ii}(0)}$.

If IFAIL = 2 and MATRIX = 'R', then on exit all the elements in R0 whose computation involves the zero variance are set to zero.

9: R(IK,IK,M) – ***real*** array. *Output*

On exit: R(i,j,l) contains an estimate of the (i,j)th element of the cross-correlation (or cross-covariance) at lag l, $\hat{R}_{ij}(l)$, for $l = 1,2,...,m$; $i = 1,2,...,k$; $j = 1,2,...,k$.

If IFAIL = 2 and MATRIX = 'R', then on exit all the elements in R whose computation involves the zero variance are set to zero.

10: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, MATRIX ≠ 'V' or 'R',
or K < 1,
or N < 2,
or M < 1,
or M ≥ N,
or IK < K.

IFAIL = 2

On entry, at least one of the k series is such that all its elements are practically equal giving zero (or near zero) variance. In this case if MATRIX = 'R' all the correlations in R0 and R involving this variance are set to zero.

7. Accuracy

For a discussion of the accuracy of the one-pass algorithm used to compute the sample cross-covariances at lag zero see West [2]. For the other lags a two-pass algorithm is used to compute the cross-covariances; the accuracy of this algorithm is also discussed in [2]. The accuracy of the cross-correlations will depend on the accuracy of the computed cross-covariances.

8. Further Comments

The time taken is roughly proportional to mnk^2.

9. Example

This program computes the sample cross-correlation matrices of two time series of length 48, up to lag 10. It also prints the cross-correlation matrices together with plots of symbols indicating which elements of the correlation matrices are significant. Three *'s represent significance at the 0.5% level, two *'s represent significance at the 1% level and a single * represents significance at the 5% level. The *'s are plotted above or below the line depending on whether the elements are significant in the positive or negative direction.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DMF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          IK, NMAX, MMAX
      PARAMETER        (IK=3,NMAX=100,MMAX=20)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, K, M, N
*     .. Local Arrays ..
      real             R(IK,IK,MMAX), R0(IK,IK), W(IK,NMAX), WMEAN(IK)
*     .. External Subroutines ..
      EXTERNAL         CPRINT, G13DMF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DMF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, N, M
```

```
         IF (K.GT.0 .AND. K.LE.IK .AND. N.GE.1 .AND. N.LE.NMAX .AND. M.GE.
     +       1 .AND. M.LE.MMAX) THEN
            DO 20 I = 1, K
               READ (NIN,*) (W(I,J),J=1,N)
   20       CONTINUE
            IFAIL = 0
*
            CALL G13DMF('R',K,N,M,W,IK,WMEAN,R0,R,IFAIL)
*
            CALL CPRINT(K,N,IK,M,WMEAN,R0,R,NOUT)
         END IF
         STOP
*
         END
*
         SUBROUTINE CPRINT(K,N,IK,M,WMEAN,R0,R,NOUT)
*        .. Scalar Arguments ..
         INTEGER           IK, K, M, N, NOUT
*        .. Array Arguments ..
         real              R(IK,IK,M), R0(IK,K), WMEAN(K)
*        .. Local Scalars ..
         real              C1, C2, C3, C5, C6, C7, CONST, SUM
         INTEGER           I, I2, IFAIL2, J, L, LL
*        .. Local Arrays ..
         CHARACTER*1       CLABS(1), RLABS(1)
         CHARACTER*80      REC(7)
*        .. External Subroutines ..
         EXTERNAL          X04CBF
*        .. Intrinsic Functions ..
         INTRINSIC         real, SQRT
*        .. Executable Statements ..
*
*        Print the correlation matrices and indicator symbols.
*
         CONST = 1.0e0/SQRT(real(N))
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' THE MEANS'
         WRITE (NOUT,*) ' ---------'
         WRITE (NOUT,99999) (WMEAN(I),I=1,K)
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' CROSS-CORRELATION MATRICES'
         WRITE (NOUT,*) ' --------------------------'
         DO 20 L = 1, M
            WRITE (NOUT,99998) ' Lag = ', L
            IFAIL2 = 0
            CALL X04CBF('G','N',K,K,R(1,1,L),IK,'F9.3',' ','N',RLABS,'N',
     +                  CLABS,80,5,IFAIL2)
   20    CONTINUE
*
*        Print indicator symbols to indicate significant elements.
*
         WRITE (NOUT,99997) ' Standard error = 1 / SQRT(N) = ', CONST
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' TABLES OF INDICATOR SYMBOLS'
         WRITE (NOUT,*) ' ---------------------------'
         WRITE (NOUT,99998) ' For Lags 1 to ', M
*
*        Set up annotation for the plots.
*
         WRITE (REC(1),99996) '                0.005  :'
         WRITE (REC(2),99996) '          +     0.01   :'
         WRITE (REC(3),99996) '                0.05   :'
         WRITE (REC(4)(1:23),99996) '   Sig. Level         :'
         WRITE (REC(4)(24:),99996) '- - - - - - - - - -  Lags'
         WRITE (REC(5),99996) '                0.05   :'
         WRITE (REC(6),99996) '          -     0.01   :'
         WRITE (REC(7),99996) '                0.005  :'
*
```

```
*     Set up the critical values
*
      C1 = 3.29e0*CONST
      C2 = 2.58e0*CONST
      C3 = 1.96e0*CONST
      C5 = -C3
      C6 = -C2
      C7 = -C1
*
      DO 120 I = 1, K
         DO 100 J = 1, K
            WRITE (NOUT,*)
            IF (I.EQ.J) THEN
               WRITE (NOUT,99995) ' Auto-correlation function for',
     +           ' series ', I
            ELSE
               WRITE (NOUT,99994) ' Cross-correlation function for',
     +           ' series ', I, ' and series', J
            END IF
            DO 60 L = 1, M
               LL = 23 + 2*L
               SUM = R(I,J,L)
*
*              Clear the last plot with blanks
*
               DO 40 I2 = 1, 7
                  IF (I2.NE.4) REC(I2) (LL:LL) = ' '
   40          CONTINUE
*
*              Check for significance
*
               IF (SUM.GT.C1) REC(1) (LL:LL) = '*'
               IF (SUM.GT.C2) REC(2) (LL:LL) = '*'
               IF (SUM.GT.C3) REC(3) (LL:LL) = '*'
               IF (SUM.LT.C5) REC(5) (LL:LL) = '*'
               IF (SUM.LT.C6) REC(6) (LL:LL) = '*'
               IF (SUM.LT.C7) REC(7) (LL:LL) = '*'
   60       CONTINUE
*
*           Print
*
            DO 80 I2 = 1, 7
               WRITE (NOUT,99996) REC(I2)
   80       CONTINUE
  100    CONTINUE
  120 CONTINUE
      RETURN
*
99999 FORMAT (/1X,2(2X,F9.3))
99998 FORMAT (/1X,A,I2)
99997 FORMAT (/1X,A,F5.3,A)
99996 FORMAT (1X,A)
99995 FORMAT (//1X,A,A,I2,/)
99994 FORMAT (//1X,A,A,I2,A,I2,/)
      END
```

9.2. Program Data

```
G13DMF Example Program Data
2 48 10 :  K, no. of series,  N, no. of obs in each series,  M, no. of lags
-1.490 -1.620  5.200  6.230  6.210  5.860  4.090  3.180
 2.620  1.490  1.170  0.850 -0.350  0.240  2.440  2.580
 2.040  0.400  2.260  3.340  5.090  5.000  4.780  4.110
 3.450  1.650  1.290  4.090  6.320  7.500  3.890  1.580
 5.210  5.250  4.930  7.380  5.870  5.810  9.680  9.070
 7.290  7.840  7.550  7.320  7.970  7.760  7.000  8.350
 7.340  6.350  6.960  8.540  6.620  4.970  4.550  4.810
 4.750  4.760 10.880 10.010 11.620 10.360  6.400  6.240
 7.930  4.040  3.730  5.600  5.350  6.810  8.270  7.680
 6.650  6.080 10.250  9.140 17.750 13.300  9.630  6.800
 4.080  5.060  4.940  6.650  7.940 10.760 11.890  5.850
 9.010  7.500 10.020 10.380  8.150  8.370 10.730 12.140 : End of time series
```

9.3. Program Results

```
 G13DMF Example Program Results

  THE MEANS
  ---------

       4.370      7.868

  CROSS-CORRELATION MATRICES
  --------------------------

  Lag =  1
            0.736    0.174
            0.211    0.555

  Lag =  2
            0.456    0.076
            0.069    0.260

  Lag =  3
            0.379    0.014
            0.026   -0.038

  Lag =  4
            0.322    0.110
            0.093   -0.236

  Lag =  5
            0.341    0.269
            0.087   -0.250

  Lag =  6
            0.363    0.344
            0.132   -0.227

  Lag =  7
            0.280    0.425
            0.207   -0.128

  Lag =  8
            0.248    0.522
            0.197   -0.085

  Lag =  9
            0.240    0.266
            0.254    0.075

  Lag = 10
            0.162   -0.020
            0.267    0.005

  Standard error = 1 / SQRT(N) = 0.144
```

```
TABLES OF INDICATOR SYMBOLS
---------------------------

For Lags 1 to 10


Auto-correlation function for series  1

              0.005  : *
        +     0.01   : * * *
              0.05   : * * * * * *
   Sig. Level        : - - - - - - - - - -   Lags
              0.05   :
        -     0.01   :
              0.005  :


Cross-correlation function for series  1 and series 2

              0.005  :               *
        +     0.01   :             * *
              0.05   :           * * *
   Sig. Level        : - - - - - - - - - -   Lags
              0.05   :
        -     0.01   :
              0.005  :


Cross-correlation function for series  2 and series 1

              0.005  :
        +     0.01   :
              0.05   :
   Sig. Level        : - - - - - - - - - -   Lags
              0.05   :
        -     0.01   :
              0.005  :


Auto-correlation function for series  2

              0.005  : *
        +     0.01   : *
              0.05   : *
   Sig. Level        : - - - - - - - - - -   Lags
              0.05   :
        -     0.01   :
              0.005  :
```

G13DNF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DNF calculates the sample partial lag correlation matrices of a multivariate time series. A set of χ^2 statistics and their sigificance levels are also returned. A call to G13DMF is usually made prior to calling this routine in order to calculate the sample cross-correlation matrices.

2. Specification

```
      SUBROUTINE G13DNF (K, N, M, IK, R0, R, MAXLAG, PARLAG, X,
     1                   PVALUE, WORK, LWORK, IFAIL)
      INTEGER            K, N, M, IK, MAXLAG, LWORK, IFAIL
      real               R0(IK,K), R(IK,IK,M), PARLAG(IK,IK,M), X(M),
     1                   PVALUE(M), WORK(LWORK)
```

3. Description

Let $W_t = (w_{1t},w_{2t},...,w_{kt})^T$, for $t = 1,2,...,n$ denote n observations of a vector of k time series. The partial lag correlation matrix at lag l, $P(l)$, is defined to be the correlation matrix between W_t and W_{t+l}, after removing the linear dependence on each of the intervening vectors $W_{t+1},W_{t+2},...,W_{t+l-1}$. It is the correlation matrix between the residual vectors resulting from the regression of W_{t+l} on the carriers $W_{t+l-1},...,W_{t+1}$ and the regression of W_t on the same set of carriers, see Heyse and Wei [1].

$P(l)$ has the following properties:

(i) If W_t follows a vector autoregressive model of order p, then $P(l) = 0$ for $l > p$;

(ii) When $k = 1$, $P(l)$ reduces to the univariate partial autocorrelation at lag l;

(iii) Each element of $P(l)$ is a properly normalized correlation coefficient;

(iv) When $l = 1$, $P(l)$ is equal to the cross-correlation matrix at lag 1 (a natural property which also holds for the univariate partial autocorrelation function).

Sample estimates of the partial lag correlation matrices may be obtained using the recursive algorithm described in Wei [2]. They are calculated up to lag m, which is usually taken to be at most $n/4$. Only the sample cross-correlation matrices ($\hat{R}(l)$, $l = 0,1,...,m$) and the standard deviations of the series are required as input to G13DNF. These may be computed by G13DMF. Under the hypothesis that W_t follows an autoregressive model of order $s-1$, the elements of the sample partial lag matrix $\hat{P}(s)$, denoted by $\hat{P}_{ij}(s)$, are asymptotically Normally distributed with mean zero and variance $1/n$. In addition the statistic

$$X(s) = n\sum_{i=1}^{k}\sum_{j=1}^{k}\hat{P}_{ij}(s)^2$$

has an asymptotic χ^2 distribution with k^2 degrees of freedom. These quantities, $X(l)$, are useful as a diagnostic aid for determining whether the series follows an autoregressive model and, if so, of what order.

4. References

[1] HEYSE, J.F. and WEI, W.W.S.
The partial lag autocorrelation function.
Technical Report No. 32, Department of Statistics, Temple University, Philadelphia, 1985.

[2] WEI, W.W.S.
Time Series Analysis: Univariate and Multivariate Methods.
Addison-Wesley, 1990.

5. Parameters

1: K – INTEGER. *Input*

On entry: the dimension, k, of the multivariate time series.

Constraint: K $\geq$ 1.

2: N – INTEGER. *Input*

On entry: the number of observations in each series, n.

Constraint: N $\geq$ 2.

3: M – INTEGER. *Input*

On entry: the number, m, of partial lag correlation matrices to be computed. Note this also specifies the number of sample cross-correlation matrices that must be contained in the array R.

Constraint: 1 $\leq$ M $<$ N.

4: IK – INTEGER. *Input*

On entry: the first dimension of the array R0 and the first and second dimension of the arrays R and PARLAG as declared in the (sub)program from which G13DNF is called.

Constraint: IK $\geq$ K.

5: R0(IK,K) – ***real*** array. *Input*

On entry: if $i \neq j$, then R0(i,j) must contain the (i,j)th element of the sample cross-correlation matrix at lag zero, $\hat{R}_{ij}(0)$. If $i = j$, then R0(i,i) must contain the standard deviation of the ith series.

6: R(IK,IK,M) – ***real*** array. *Input*

On entry: R(i,j,l) must contain the (i,j)th element of the sample cross-correlation at lag l, $\hat{R}_{ij}(l)$, for $l = 1,2,...,m$; $i = 1,2,...,k$; $j = 1,2,...,k$, where series j leads series i (see Section 8).

7: MAXLAG – INTEGER. *Output*

On exit: the maximum lag up to which partial lag correlation matrices (along with χ^2 statistics and their significance levels) have been successfully computed. On a successful exit MAXLAG will equal M. If IFAIL = 2 on exit, then MAXLAG will be less than M.

8: PARLAG(IK,IK,M) – ***real*** array. *Output*

On exit: PARLAG(i,j,l) contains the (i,j)th element of the sample partial lag correlation matrix at lag l, $\hat{P}_{ij}(l)$, for l = 1,2,...,MAXLAG; $i = 1,2,...,k$; $j = 1,2,...,k$.

9: X(M) – ***real*** array. *Output*

On exit: X(l) contains the χ^2 statistic at lag l, for l = 1,2,...,MAXLAG.

10: PVALUE(M) – ***real*** array. *Output*

On exit: PVALUE(l) contains the significance level of the corresponding χ^2 statistic in X for l = 1,2,...,MAXLAG.

11: WORK(LWORK) – ***real*** array. *Workspace*

12: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G13DNF is called.

Constraint: LWORK $\geq$ $(5M+6)K^2 + K$.

13: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, $K < 1$,
or $N < 2$,
or $M < 1$,
or $M \geq N$,
or $IK < K$,
or $LWORK < (5M+6)K^2 + K$.

IFAIL = 2

The recursive equations used to compute the sample partial lag correlation matrices have broken down at lag MAXLAG + 1. All output quantities in the arrays PARLAG, X and PVALUE up to and including lag MAXLAG will be correct.

7. Accuracy

The accuracy will depend upon the accuracy of the sample cross-correlations.

8. Further Comments

The time taken is roughly proportional to m^2k^3.

If the user has calculated the sample cross-correlation matrices in the arrays R0 and R, without calling G13DMF, then care must be taken to ensure they are supplied as described in Section 5. In particular, for $l \geq 1$, $\hat{R}_{ij}(l)$ must contain the sample cross-correlation coefficient between $w_{i(t-l)}$ and w_{jt}.

The routine G13DBF computes squared partial autocorrelations for a specified number of lags. It may also be used to estimate a sequence of partial autoregression matrices at lags 1,2,... by making repeated calls to the routine with the parameter NK set to 1,2,.... The (i,j)th element of the sample partial autoregression matrix at lag l is given by $W(i,j,l)$ when NK is set equal to l on entry to G13DBF. Note that this is the 'Yule-Walker' estimate. Unlike the partial lag correlation matrices computed by G13DNF, when W_t follows an autoregressive model of order $s-1$, the elements of the sample partial autoregressive matrix at lag s do not have variance $1/n$ making it very difficult to spot a possible cut-off point. The differences between these matrices are discussed further by Wei [2].

Note that G13DBF takes the sample cross-covariance matrices as input whereas this routine requires the sample cross-correlation matrices to be input.

9. Example

This program computes the sample partial lag correlation matrices of two time series of length 48, up to lag 10. The matrices, their χ^2 statistics and significance levels and a plot of symbols indicating which elements of the sample partial lag correlation matrices are significant are printed. Three *'s represent significance at the 0.5% level, 2 *'s represent significance at the 1% level and a single * represents significance at the 5% level. The *'s are plotted above or below the central line depending on whether the elements are significant in a positive or negative direction.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DNF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          KMAX, IK, NMAX, MMAX, LWORK
      PARAMETER        (KMAX=3,IK=KMAX,NMAX=100,MMAX=20,LWORK=(5*MMAX+6)
     +                 *KMAX*KMAX+KMAX)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, K, M, MAXLAG, N
*     .. Local Arrays ..
      real             PARLAG(IK,IK,MMAX), PVALUE(MMAX), R(IK,IK,MMAX),
     +                 R0(IK,KMAX), W(IK,NMAX), WMEAN(KMAX),
     +                 WORK(LWORK), X(MMAX)
*     .. External Subroutines ..
      EXTERNAL         G13DMF, G13DNF, ZPRINT
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DNF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, N, M
      IF (K.GT.0 .AND. K.LE.KMAX .AND. N.GE.1 .AND. N.LE.NMAX .AND.
     +    M.GE.1 .AND. M.LE.MMAX) THEN
         DO 20 I = 1, K
            READ (NIN,*) (W(I,J),J=1,N)
   20    CONTINUE
         IFAIL = 0
*
         CALL G13DMF('R-correlation',K,N,M,W,IK,WMEAN,R0,R,IFAIL)
*
         IFAIL = 0
*
         CALL G13DNF(K,N,M,IK,R0,R,MAXLAG,PARLAG,X,PVALUE,WORK,LWORK,
     +               IFAIL)
*
         CALL ZPRINT(K,N,M,IK,PARLAG,X,PVALUE,NOUT)
      END IF
      STOP
*
      END
*
      SUBROUTINE ZPRINT(K,N,M,IK,PARLAG,X,PVALUE,NOUT)
*     .. Scalar Arguments ..
      INTEGER           IK, K, M, N, NOUT
*     .. Array Arguments ..
      real              PARLAG(IK,IK,M), PVALUE(M), X(M)
*     .. Local Scalars ..
      real              C1, C2, C3, C5, C6, C7, CONST, SUM
      INTEGER           I, I2, IFAIL2, J, L, LL
*     .. Local Arrays ..
      CHARACTER*1       CLABS(1), RLABS(1)
      CHARACTER*80      REC(7)
```

```
*     .. External Subroutines ..
      EXTERNAL         X04CBF
*     .. Intrinsic Functions ..
      INTRINSIC        real, SQRT
*     .. Executable Statements ..
*
*     Print the partial lag correlation matrices.
*
      CONST = 1.0e0/SQRT(real(N))
      WRITE (NOUT,*)
      WRITE (NOUT,*) ' PARTIAL LAG CORRELATION MATRICES'
      WRITE (NOUT,*) ' --------------------------------'
      DO 20 L = 1, M
         WRITE (NOUT,99999) ' Lag = ', L
         IFAIL2 = 0
         CALL X04CBF('G','N',K,K,PARLAG(1,1,L),IK,'F9.3',' ','N',RLABS,
     +               'N',CLABS,80,5,IFAIL2)
   20 CONTINUE
      WRITE (NOUT,99998) ' Standard error = 1 / SQRT(N) = ', CONST
*
*     Print indicator symbols to indicate significant elements.
*
      WRITE (NOUT,*)
      WRITE (NOUT,*) ' TABLES OF INDICATOR SYMBOLS'
      WRITE (NOUT,*) ' ---------------------------'
      WRITE (NOUT,99999) ' For Lags 1 to ', M
*
*     Set up annotation for the plots.
*
      WRITE (REC(1),99997) '              0.005  :'
      WRITE (REC(2),99997) '         +    0.01   :'
      WRITE (REC(3),99997) '              0.05   :'
      WRITE (REC(4)(1:23),99997) '   Sig. Level        :'
      WRITE (REC(4)(24:),99997) '- - - - - - - - - -  Lags'
      WRITE (REC(5),99997) '              0.05   :'
      WRITE (REC(6),99997) '         -    0.01   :'
      WRITE (REC(7),99997) '              0.005  :'
*
*     Set up the critical values
*
      C1 = 3.29e0*CONST
      C2 = 2.58e0*CONST
      C3 = 1.96e0*CONST
      C5 = -C3
      C6 = -C2
      C7 = -C1
*
      DO 120 I = 1, K
         DO 100 J = 1, K
            WRITE (NOUT,*)
            IF (I.EQ.J) THEN
               WRITE (NOUT,99996) ' Auto-correlation function for',
     +           ' series ', I
            ELSE
               WRITE (NOUT,99995) ' Cross-correlation function for',
     +           ' series ', I, ' and series', J
            END IF
            DO 60 L = 1, M
               LL = 23 + 2*L
               SUM = PARLAG(I,J,L)
*
*              Clear the last plot with blanks
*
               DO 40 I2 = 1, 7
                  IF (I2.NE.4) REC(I2) (LL:LL) = ' '
   40          CONTINUE
*
```

```
*             Check for significance
*
              IF (SUM.GT.C1) REC(1) (LL:LL) = '*'
              IF (SUM.GT.C2) REC(2) (LL:LL) = '*'
              IF (SUM.GT.C3) REC(3) (LL:LL) = '*'
              IF (SUM.LT.C5) REC(5) (LL:LL) = '*'
              IF (SUM.LT.C6) REC(6) (LL:LL) = '*'
              IF (SUM.LT.C7) REC(7) (LL:LL) = '*'
   60       CONTINUE
*
*           Print
*
            DO 80 I2 = 1, 7
               WRITE (NOUT,99997) REC(I2)
   80       CONTINUE
  100    CONTINUE
  120 CONTINUE
*
*     Print the chi-square statistics and p-values.
*
      WRITE (NOUT,*)
      WRITE (NOUT,*)
      WRITE (NOUT,*) ' Lag     Chi-square statistic     P-value'
      WRITE (NOUT,*) ' ---     --------------------     -------'
      WRITE (NOUT,*)
      DO 140 L = 1, M
         WRITE (NOUT,99994) L, X(L), PVALUE(L)
  140 CONTINUE
      RETURN
*
99999 FORMAT (/1X,A,I2)
99998 FORMAT (/1X,A,F5.3,A)
99997 FORMAT (1X,A)
99996 FORMAT (//1X,A,A,I2,/)
99995 FORMAT (//1X,A,A,I2,A,I2,/)
99994 FORMAT (1X,I4,10X,F8.3,11X,F8.4)
      END
```

9.2. Program Data

```
G13DNF Example Program Data
2 48 10 :  K, no. of series,  N, no. of obs in each series,  M, no. of lags
-1.490 -1.620  5.200  6.230  6.210  5.860  4.090  3.180
 2.620  1.490  1.170  0.850 -0.350  0.240  2.440  2.580
 2.040  0.400  2.260  3.340  5.090  5.000  4.780  4.110
 3.450  1.650  1.290  4.090  6.320  7.500  3.890  1.580
 5.210  5.250  4.930  7.380  5.870  5.810  9.680  9.070
 7.290  7.840  7.550  7.320  7.970  7.760  7.000  8.350
 7.340  6.350  6.960  8.540  6.620  4.970  4.550  4.810
 4.750  4.760 10.880 10.010 11.620 10.360  6.400  6.240
 7.930  4.040  3.730  5.600  5.350  6.810  8.270  7.680
 6.650  6.080 10.250  9.140 17.750 13.300  9.630  6.800
 4.080  5.060  4.940  6.650  7.940 10.760 11.890  5.850
 9.010  7.500 10.020 10.380  8.150  8.370 10.730 12.145 : End of time series
```

9.3. Program Results

```
 G13DNF Example Program Results

  PARTIAL LAG CORRELATION MATRICES
  --------------------------------


  Lag =  1
           0.736    0.174
           0.211    0.555

  Lag =  2
          -0.187   -0.083
          -0.180   -0.072
```

```
Lag =  3
         0.278   -0.007
         0.084   -0.213

Lag =  4
        -0.084    0.227
         0.128   -0.176

Lag =  5
         0.236    0.238
        -0.047   -0.046

Lag =  6
        -0.016    0.087
         0.100   -0.081

Lag =  7
        -0.036    0.261
         0.126    0.012

Lag =  8
         0.077    0.381
         0.027   -0.149

Lag =  9
        -0.065   -0.387
         0.189    0.057

Lag = 10
        -0.026   -0.286
         0.028   -0.173

Standard error = 1 / SQRT(N) = 0.144

TABLES OF INDICATOR SYMBOLS
---------------------------

For Lags 1 to 10


Auto-correlation function for series  1

              0.005  : *
        +     0.01   : *
              0.05   : *
   Sig. Level        : - - - - - - - - - -  Lags
              0.05   :
        -     0.01   :
              0.005  :


Cross-correlation function for series  1 and series 2

              0.005  :
        +     0.01   :               *
              0.05   :               *
   Sig. Level        : - - - - - - - - - -  Lags
              0.05   :                 * *
        -     0.01   :                 *
              0.005  :
```

```
Cross-correlation function for series  2 and series 1

            0.005  :
      +     0.01   :
            0.05   :
  Sig. Level       : - - - - - - - - - - -  Lags
            0.05   :
      -     0.01   :
            0.005  :


Auto-correlation function for series  2

            0.005  : *
      +     0.01   : *
            0.05   : *
  Sig. Level       : - - - - - - - - - - -  Lags
            0.05   :
      -     0.01   :
            0.005  :

Lag     Chi-square statistic     P-value
---     --------------------     -------

  1           44.362              0.0000
  2            3.824              0.4304
  3            6.219              0.1834
  4            5.094              0.2778
  5            5.609              0.2303
  6            1.170              0.8830
  7            4.098              0.3929
  8            8.371              0.0789
  9            9.244              0.0553
 10            5.435              0.2455
```

G13DPF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DPF calculates the sample partial autoregression matrices of a multivariate time series. A set of likelihood ratio statistics and their significance levels are also returned. These quantities are useful for determining whether the series follows an autoregessive model and, if so, of what order.

2. Specification

```
      SUBROUTINE G13DPF (K, N, Z, IK, M, MAXLAG, PARLAG, SE, QQ, X, PVALUE,
     1                   LOGLHD, WORK, LWORK, IWORK, IFAIL)
      INTEGER           K, N, IK, M, MAXLAG, LWORK, IWORK(K*M), IFAIL
      real              Z(IK,N), PARLAG(IK,IK,M), SE(IK,IK,M), QQ(IK,IK,M),
     1                  X(M), PVALUE(M), LOGLHD(M), WORK(LWORK)
```

3. Description

Let $W_t = (w_{1t}, w_{2t}, \ldots, w_{kt})^T$, for $t = 1,2,\ldots,n$ denote a vector of k time series. The partial autoregression matrix at lag l, P_l, is defined to be the last matrix coefficient when a vector autoregressive model of order l is fitted to the series. P_l has the property that if W_t follows a vector autoregressive model of order p then $P_l = 0$ for $l > p$.

Sample estimates of the partial autoregression matrices may be obtained by fitting autoregressive models of successively higher orders by multivariate least squares, see Tiao and Box [1] and Wei [2]. These models are fitted using a *QR* algorithm based on the routines G02DCF and G02DFF. They are calculated up to lag m, which is usually taken to be at most $n/4$.

The routine also returns the asymptotic standard errors of the elements of $\hat{P}_l$ and an estimate of the corresponding variance-covariance matrix $\hat{\Sigma}_l$ for $l = 1,2,\ldots,m$. If S_l denotes the residual sum of squares and cross products matrix after fitting an AR(l) model to the series then under the null hypothesis $H_0 : P_l = 0$ the test statistic

$$X_l = -\{(n-m-1) - \tfrac{1}{2} - lk\} \log\left\{\frac{|S_l|}{|S_{l-1}|}\right\}$$

is asymptotically distributed as χ^2 with k^2 degrees of freedom. X_l provides a useful diagnostic aid in determining the order of an autoregressive model. (Note that $\hat{\Sigma}_l = S_l/(n-l)$.) The routine also returns an estimate of the maximum of the log-likelihood function for each AR model that has been fitted.

4. References

[1] TIAO, G.C. and BOX, G.E.P.
Modelling multiple time series with applications.
J. Am. Stat. Assoc., 76, pp. 802-816, 1981.

[2] WEI, W.W.S.
Time Series Analysis: Univariate and Multivariate Methods.
Addison-Wesley, 1990.

5. Parameters

1: K – INTEGER. *Input*

On entry: the number of time series, k.

Constraint: K $\geq$ 1.

2: N – INTEGER. *Input*

On entry: the number of observations in the time series, n.

Constraint: N $\geq$ 4.

3: Z(IK,N) – ***real*** array. *Input*

On entry: Z(i,t) must contain the observation w_{it} for $i = 1,2,...,k$; $t = 1,2,...,n$.

4: IK – INTEGER. *Input*

On entry: the first dimension of the array Z and the first and second dimensions of arrays PARLAG, SE, QQ as declared in the (sub)program from which G13DPF is called.

Constraint: IK $\geq$ K.

5: M – INTEGER. *Input*

On entry: the number, m, of partial autoregression matrices to be computed. If in doubt set M = 10.

Constraints: M $\geq$ 1 and N − M − (K×M+1) $\geq$ K.

6: MAXLAG – INTEGER. *Output*

On exit: the maximum lag up to which partial autoregression matrices (along with their likelihood ratio statistics and their significance levels) have been successfully computed. On a successful exit MAXLAG will equal M. If IFAIL = 2 on exit then MAXLAG will be less than M.

7: PARLAG(IK,IK,M) – ***real*** array. *Output*

On exit: PARLAG(i,j,l) contains an estimate of the (i,j)th element of the partial autoregression matrix at lag l, $\hat{P}_l(ij)$, for l = 1,2,...,MAXLAG; $i = 1,2,...,k$; $j = 1,2,...,k$.

8: SE(IK,IK,M) – ***real*** array. *Output*

On exit: SE(i,j,l) contains an estimate of the standard error of the corresponding element in the array PARLAG.

9: QQ(IK,IK,M) – ***real*** array. *Output*

On exit: QQ(i,j,l) contains an estimate of the (i,j)th element of the corresponding variance-covariance matrix $\hat{\Sigma}_l$ for l = 1,2,...,MAXLAG; $i = 1,2,...,k$; $j = 1,2,...,k$.

10: X(M) – ***real*** array. *Output*

On exit: X(l) contains X_l, the likelihood ratio statistic at lag l for l = 1,2,...,MAXLAG.

11: PVALUE(M) – ***real*** array. *Output*

On exit: PVALUE(l) contains the significance level of the statistic in the corresponding element of X.

12: LOGLHD(M) – ***real*** array. *Output*

On exit: LOGLHD(l) contains an estimate of the maximum of the log-likelihood function when an AR(l) model has been fitted to the series for l = 1,2,...,MAXLAG.

13: WORK(LWORK) – ***real*** array. *Workspace*

14: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G13DPF is called.

Constraint: LWORK $\geq$ $(k+1)k + l(4+k) + 2l^2$, where $l = mk+1$.

15: IWORK(K*M) – INTEGER array. *Workspace*

16: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, K < 1,
or N < 4,
or IK < K,
or M < 1,
or N – M – (K×M+1) < K,
or LWORK is too small.

IFAIL = 2

The recursive equations used to compute the sample partial autoregression matrices have broken down at lag MAXLAG + 1. This exit could occur if the regression model is overparameterised. For the users settings of k and n the value returned by MAXLAG is the largest permissible value of m for which the model is not overparameterised. All output quantities in the arrays PARLAG, SE, QQ, X, PVALUE and LOGHLD up to and including lag MAXLAG will be correct.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

The time taken is roughly proportional to nmk.

For each order of autoregressive model that has been estimated, G13DPF returns the maximum of the log-likelihood function. An alternative means of choosing the order of a vector AR process is to choose the order for which Akaike's information criterion is smallest. That is choose the value of l for which $-2\times\text{LOGHLD}(l) + 2lk^2$ is smallest. The user should be warned that this does not always lead to the same choice of l as indicated by the sample partial autoregression matrices and the likelihood ratio statistics.

9. Example

A program to compute the sample partial autoregression matrices of two time series of length 48 up to lag 10.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DPF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          KMAX, NMAX, MMAX, LWORK
      PARAMETER        (KMAX=4,NMAX=50,MMAX=10,LWORK=2081)
```

```
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, K, M, MAXLAG, N
*     .. Local Arrays ..
      real             LOGLHD(MMAX), PARLAG(KMAX,KMAX,MMAX),
     +                 PVALUE(MMAX), QQ(KMAX,KMAX,MMAX),
     +                 SE(KMAX,KMAX,MMAX), W(KMAX,NMAX), WORK(LWORK),
     +                 X(MMAX)
      INTEGER          IWORK(KMAX*MMAX)
*     .. External Subroutines ..
      EXTERNAL         G13DPF, ZPRINT
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DPF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, N, M
*
      IF (K.GT.0 .AND. K.LE.KMAX .AND. N.GE.1 .AND. N.LE.NMAX .AND.
     +    M.GE.1 .AND. M.LE.MMAX) THEN
*
         DO 20 I = 1, K
            READ (NIN,*) (W(I,J),J=1,N)
   20    CONTINUE
*
         IFAIL = 0
*
         CALL G13DPF(K,N,W,KMAX,M,MAXLAG,PARLAG,SE,QQ,X,PVALUE,LOGLHD,
     +               WORK,LWORK,IWORK,IFAIL)
*
         CALL ZPRINT(K,N,M,KMAX,MAXLAG,PARLAG,SE,QQ,X,PVALUE,NOUT,IFAIL)
*
      END IF
      STOP
      END
*
      SUBROUTINE ZPRINT(K,N,M,KMAX,MAXLAG,PARLAG,SE,QQ,X,PVALUE,NOUT,
     +                  IFAIL)
*
*     .. Scalar Arguments ..
      INTEGER           IFAIL, K, KMAX, M, MAXLAG, N, NOUT
*     .. Array Arguments ..
      real              PARLAG(KMAX,KMAX,M), PVALUE(M), QQ(KMAX,KMAX,M),
     +                  SE(KMAX,KMAX,M), X(M)
*     .. Local Scalars ..
      real              SUM
      INTEGER           I, I2, J, L
*     .. Local Arrays ..
      CHARACTER*6       ST(6)
*     .. Executable Statements ..
*
      IF (K.GT.1) WRITE (NOUT,99999)
      IF (K.EQ.1) WRITE (NOUT,99998)
      DO 80 L = 1, MAXLAG
         DO 20 J = 1, K
            SUM = PARLAG(1,J,L)
            ST(J) = '.'
            IF (SUM.GT.1.96e0*SE(1,J,L)) ST(J) = '+'
            IF (SUM.LT.-1.96e0*SE(1,J,L)) ST(J) = '-'
   20    CONTINUE
         IF (K.EQ.1) THEN
            WRITE (NOUT,99997) L, (PARLAG(1,J,L),J=1,K),
     +        (ST(I2),I2=1,K), QQ(1,1,L), X(L), PVALUE(L)
            WRITE (NOUT,99996) (SE(1,J,L),J=1,K)
         ELSE IF (K.EQ.2) THEN
            WRITE (NOUT,99995) L, (PARLAG(1,J,L),J=1,K),
     +        (ST(I2),I2=1,K), QQ(1,1,L), X(L), PVALUE(L)
            WRITE (NOUT,99994) (SE(1,J,L),J=1,K)
         ELSE IF (K.EQ.3) THEN
            WRITE (NOUT,99993) L, (PARLAG(1,J,L),J=1,K),
     +        (ST(I2),I2=1,K), QQ(1,1,L), X(L), PVALUE(L)
            WRITE (NOUT,99992) (SE(1,J,L),J=1,K)
            ELSE IF (K.EQ.4) THEN
               WRITE (NOUT,99991) L
               WRITE (NOUT,99984) (PARLAG(1,J,L),J=1,K), (ST(I2),I2=1,K),
     +            QQ(1,1,L), X(L), PVALUE(L)
               WRITE (NOUT,99990) (SE(1,J,L),J=1,K)
            END IF
*
            DO 60 I = 2, K
*
               DO 40 J = 1, K
                  SUM = PARLAG(I,J,L)
                  ST(J) = '.'
                  IF (SUM.GT.1.96e0*SE(I,J,L)) ST(J) = '+'
                  IF (SUM.LT.-1.96e0*SE(I,J,L)) ST(J) = '-'
   40          CONTINUE
               IF (K.EQ.2) THEN
                  WRITE (NOUT,99987) (PARLAG(I,J,L),J=1,K),
     +              (ST(I2),I2=1,K), QQ(I,I,L)
                  WRITE (NOUT,99994) (SE(I,J,L),J=1,K)
               ELSE IF (K.EQ.3) THEN
                  WRITE (NOUT,99986) (PARLAG(I,J,L),J=1,K),
     +              (ST(I2),I2=1,K), QQ(I,I,L)
                  WRITE (NOUT,99992) (SE(I,J,L),J=1,K)
               ELSE IF (K.EQ.4) THEN
                  WRITE (NOUT,99985) (PARLAG(I,J,L),J=1,K),
     +              (ST(I2),I2=1,K), QQ(I,I,L)
                  WRITE (NOUT,99990) (SE(I,J,L),J=1,K)
               END IF
*
   60       CONTINUE
   80    CONTINUE
*
         WRITE (NOUT,99983) IFAIL
*
         RETURN
*
99999 FORMAT (/' Partial Autoregression Matrices',4X,'Indicator',2X,
     +        'Residual',3X,'Chi-Square',2X,'Pvalue',/37X,'Symbols',3X,
     +        'Variances',3X,'Statistic',/' ----------------------------',
     +        '----',4X,'---------',2X,'---------',2X,'----------',1X,
     +        '------')
99998 FORMAT (/' Partial Autoregression Function',4X,'Indicator',2X,
     +        'Residual',3X,'Chi-Square',2X,'Pvalue',/37X,'Symbols',3X,
     +        'Variances',3X,'Statistic',/' ----------------------------',
     +        '----',4X,'---------',2X,'---------',2X,'----------',1X,
     +        '------')
99997 FORMAT (/' Lag',I3,1X,':',F7.3,22X,A1,F14.3,3X,F10.3,F9.3)
99996 FORMAT (10X,'(',F5.3,')')
99995 FORMAT (/' Lag',I3,1X,':',2F8.3,14X,2A1,F13.3,3X,F10.3,F9.3)
99994 FORMAT (11X,'(',F5.3,') (',F5.3,')')
99993 FORMAT (/' Lag',I3,1X,':',3F8.3,6X,3A1,F12.3,3X,F10.3,F9.3)
99992 FORMAT (11X,'(',F5.3,') (',F5.3,') (',F5.3,')')
99991 FORMAT (/' Lag',I3)
99990 FORMAT (3X,'(',F5.3,') (',F5.3,') (',F5.3,') (',F5.3,')')
99989 FORMAT (/' Lag',I3,1X,':',5F7.3,1X,5A1,F10.3,3X,F10.3,F9.3)
99988 FORMAT (10X,'(',F5.3,')(',F5.3,')(',F5.3,')(',F5.3,')(',F5.3,')')
99987 FORMAT (9X,2F8.3,14X,2A1,F13.3)
99986 FORMAT (9X,3F8.3,6X,3A1,F12.3)
99985 FORMAT (1X,4F8.3,5X,4A1,F12.3)
99984 FORMAT (1X,4F8.3,5X,4A1,F12.3,3X,F10.3,F9.3)
99983 FORMAT (/' Value of IFAIL parameter on exit from G13DPF = ',I2)
      END
```

9.2. Program Data

```
G13DPF Example Program Data
2 48 10 :  K, no. of series,  N, no. of obs in each series,  M, no. of lags
-1.490 -1.620  5.200  6.230  6.210  5.860  4.090  3.180
 2.620  1.490  1.170  0.850 -0.350  0.240  2.440  2.580
 2.040  0.400  2.260  3.340  5.090  5.000  4.780  4.110
```

```
3.450  1.650  1.290  4.090  6.320  7.500  3.890  1.580
5.210  5.250  4.930  7.380  5.870  5.810  9.680  9.070
7.290  7.840  7.550  7.320  7.970  7.760  7.000  8.350
7.340  6.350  6.960  8.540  6.620  4.970  4.550  4.810
4.750  4.760 10.880 10.010 11.620 10.360  6.400  6.240
7.930  4.040  3.730  5.600  5.350  6.810  8.270  7.680
6.650  6.080 10.250  9.140 17.750 13.300  9.630  6.800
4.080  5.060  4.940  6.650  7.940 10.760 11.890  5.850
9.010  7.500 10.020 10.380  8.150  8.370 10.730 12.140 : End of time series
```

9.3. Program Results

```
G13DPF Example Program Results

Partial Autoregression Matrices      Indicator  Residual   Chi-Square  Pvalue
                                      Symbols   Variances   Statistic
-------------------------------      ---------  ---------  ----------- ------

Lag  1 :   0.757   0.062                +.        2.731       49.884    0.000
          (0.092) (0.092)
           0.061   0.570                .+        5.440
          (0.129) (0.130)

Lag  2 :  -0.161  -0.135                ..        2.530        3.347    0.502
          (0.145) (0.109)
          -0.093  -0.065                ..        5.486
          (0.213) (0.160)

Lag  3 :   0.237   0.044                ..        1.755       13.962    0.007
          (0.128) (0.095)
           0.047  -0.248                ..        5.291
          (0.222) (0.165)

Lag  4 :  -0.098   0.152                ..        1.661        7.071    0.132
          (0.134) (0.099)
           0.402  -0.194                ..        4.786
          (0.228) (0.168)

Lag  5 :   0.257  -0.026                ..        1.504        5.184    0.269
          (0.141) (0.106)
           0.400  -0.021                ..        4.447
          (0.242) (0.183)

Lag  6 :  -0.075   0.112                ..        1.480        2.083    0.721
          (0.156) (0.111)
           0.196  -0.106                ..        4.425
          (0.269) (0.192)

Lag  7 :  -0.054   0.097                ..        1.478        5.074    0.280
          (0.166) (0.121)
           0.574  -0.080                +.        3.838
          (0.267) (0.195)

Lag  8 :   0.147   0.041                ..        1.415       10.991    0.027
          (0.188) (0.128)
           0.916  -0.242                +.        2.415
          (0.246) (0.167)

Lag  9 :  -0.039   0.099                ..        1.322        3.936    0.415
          (0.251) (0.140)
          -0.500   0.173                ..        2.196
          (0.324) (0.181)

Lag 10 :   0.189   0.131                ..        1.206        3.175    0.529
          (0.275) (0.157)
          -0.183  -0.040                ..        2.201
          (0.371) (0.212)

Value of IFAIL parameter on exit from G13DPF =  0
```

G13DSF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DSF is a diagnostic checking routine suitable for use after fitting a vector ARMA model to a multivariate time series using G13DCF. The residual cross-correlation matrices are returned along with an estimate of their asymptotic standard errors and correlations. Also, G13DSF calculates the modified Li-McLeod portmanteau statistic and its significance level for testing model adequacy.

2. Specification

```
SUBROUTINE G13DSF (K, N, V, IK, IP, IQ, M, PAR, PARHLD, QQ, ISHOW, R0,
1                  R, RCM, IRCM, CHI, IDF, SIGLEV, IW, LIW, WORK,
2                  LWORK, IFAIL)
INTEGER         K, N, IK, IP, IQ, M, ISHOW, IRCM, IDF, IW(LIW), LIW,
1               LWORK, IFAIL
real            V(IK,N), PAR((IP+IQ)*K*K), QQ(IK,K), R0(IK,K),
1               R(IK,IK,M), RCM(IRCM,M*K*K), CHI, SIGLEV,
2               WORK(LWORK)
LOGICAL         PARHLD((IP+IQ)*K*K)
```

3. Description

Let $W_t = (w_{1t},w_{2t},\ldots,w_{kt})^T$, for $t = 1,2,\ldots,n$ denote a vector of k time series which is assumed to follow a multivariate ARMA model of the form:

$$\begin{aligned} W_t - \mu = {} & \phi_1(W_{t-1}-\mu) + \phi_2(W_{t-2}-\mu) + \ldots + \phi_p(W_{t-p}-\mu) \\ & + \varepsilon_t - \theta_1\varepsilon_{t-1} - \theta_2\varepsilon_{t-2} - \ldots - \theta_q\varepsilon_{t-q} \end{aligned} \qquad (1)$$

where $\varepsilon_t = (\varepsilon_{1t},\varepsilon_{2t},\ldots,\varepsilon_{kt})^T$, for $t = 1,2,\ldots,n$ is a vector of k residual series assumed to be normally distributed with zero mean and positive-definite covariance matrix Σ. The components of ε_t are assumed to be uncorrelated at non-simultaneous lags. The ϕ_i's and θ_j's are k by k matrices of parameters. $\{\phi_i\}$, for $i = 1,2,\ldots,p$, are called the autoregressive (AR) parameter matrices, and $\{\theta_i\}$, for $i = 1,2,\ldots,q$, the moving average (MA) parameter matrices. The parameters in the model are thus the p k by k ϕ-matrices, the q k by k θ-matrices, the mean vector μ and the residual error covariance matrix Σ. Let

$$A(\phi) = \begin{bmatrix} \phi_1 & I & 0 & . & . & . & 0 \\ \phi_2 & 0 & I & 0 & . & . & 0 \\ . & & & . & & & \\ . & & & & & . & \\ . & & & & & . & \\ \phi_{p-1} & 0 & . & . & . & 0 & I \\ \phi_p & 0 & . & . & . & 0 & 0 \end{bmatrix}_{pk\times pk} \quad \text{and} \quad B(\theta) = \begin{bmatrix} \theta_1 & I & 0 & . & . & . & 0 \\ \theta_2 & 0 & I & 0 & . & . & 0 \\ . & & & . & & & \\ . & & & & & . & \\ . & & & & & . & \\ \theta_{q-1} & 0 & . & . & . & & I \\ \theta_q & 0 & . & . & . & . & 0 \end{bmatrix}_{qk\times qk}$$

where I denotes the k by k identity matrix.

The model (1) is said to be stationary if the eigenvalues of $A(\phi)$ lie inside the unit circle, and invertible if the eigenvalues of $B(\theta)$ lie inside the unit circle. The ARMA model is assumed to be both stationary and invertible. Note that some of the elements of the ϕ- and/or θ-matrices may have been fixed at pre-specified values (for example by calling G13DCF).

The estimated residual cross-correlation matrix at lag l is defined to the k by k matrix $\hat{R}_l$ whose (i,j)th element is computed as

$$\hat{r}_{ij}(l) = \frac{\sum_{t=l+1}^{n} (\hat{\varepsilon}_{i,t-l}-\bar{\varepsilon}_i)(\hat{\varepsilon}_{jt}-\bar{\varepsilon}_j)}{\sqrt{\sum_{t=1}^{n} (\hat{\varepsilon}_{it}-\bar{\varepsilon}_i)^2 \sum_{t=1}^{n} (\hat{\varepsilon}_{jt}-\bar{\varepsilon}_j)^2}} \qquad l = 0,1,...; \; i,j = 1,2,...,k$$

where $\hat{\varepsilon}_{it}$, denotes an estimate of the tth residual for the ith series, ε_{it}, and $\bar{\varepsilon}_i = \sum_{t=1}^{n} \hat{\varepsilon}_{it}/n$. (Note that $\hat{R}_l$ is an estimate of $E(\varepsilon_{t-l}\varepsilon_t^T)$, where E is the expected value.)

A modified portmanteau statistic, $Q^*_{(m)}$, is calculated from the formula (see Li and McLeod [1]):

$$Q^*_{(m)} = \frac{k^2 m(m+1)}{2n} + n \sum_{l=1}^{m} \hat{r}(l)^T \; (\hat{R}_0^{-1} \otimes \hat{R}_0^{-1}) \; \hat{r}(l)$$

where $\otimes$ denotes kronecker product, $\hat{R}_0$ is the estimated residual cross-correlation matrix at lag zero and $\hat{r}(l) = \text{vec } (\hat{R}_l^T)$ where vec of a k by k matrix is a vector with the (i,j)th element in position $(i-1)k+j$. m denotes the number of residual cross-correlation matrices computed. (Advice on the choice of m is given in Section 8.2.) Let l_C denote the total number of 'free' parameters in the ARMA model excluding the mean, μ, and the residual error covariance matrix Σ. Then, under the hypothesis of model adequacy, $Q^*_{(m)}$, has an asymptotic χ^2 distribution on $mk^2 - l_C$ degrees of freedom.

Let $\underset{\sim}{\hat{r}} = (\text{vec}(\hat{R}_1^T),\text{vec}(\hat{R}_2^T),...,\text{vec}(\hat{R}_m^T))$ then the covariance matrix of $\underset{\sim}{\hat{r}}$ is given by

$$\text{Var}(\underset{\sim}{\hat{r}}) = [Y - X(X^T G G^T X)^{-1} X^T]/n$$

where $Y = I_m \otimes (\Delta \otimes \Delta)$ and $G = I_m \otimes (G \otimes G^T)$. Δ is the dispersion matrix Σ in correlation form and G a non-singular k by k matrix such that $GG^T = \Delta^{-1}$ and $G\Delta G^T = I_k$. The construction of the matrix X is discussed in Li and McLeod [1]. (Note that the mean, μ, plays no part in calculating $\text{Var}(\underset{\sim}{\hat{r}})$ and therefore is not required as input to G13DSF.)

4. References

[1] LI, W. K. and McLEOD, A. I.
Distribution of the Residual Autocorrelations in Multivariate ARMA Time Series Models.
J. Roy. Statist. Soc. Ser. B, 43, pp. 231-239, 1981.

5. Parameters

The output quantities K, N, V, IK, IP, IQ, PAR, PARHLD and QQ from G13DCF are suitable for input to G13DSF.

1: K – INTEGER. *Input*

On entry: the number of residual time series, k.

Constraint: K $\geq$ 1.

2: N – INTEGER. *Input*

On entry: the number of observations in each residual series, n.

Constraint: N $\geq$ 3.

3: V(IK,N) – ***real*** array. *Input*

On entry: an estimate of the ith component of ε_t, for $i = 1,2,...,k$; $t = 1,2,...,n$.

Constraints: no two rows of V may be identical.
In each row there must be at least two distinct elements.

4: IK – INTEGER. *Input*

On entry: the first dimension of the array V, QQ and R0 as declared in the (sub)program from which G13DSF is called.

Constraint: IK $\geq$ K.

5: IP – INTEGER. *Input*

On entry: the number of AR parameter matrices, p.

Constraint: IP $\geq$ 0.

6: IQ – INTEGER. *Input*

On entry: the number of MA parameter matrices, q.

Constraint: IQ $\geq$ 0.

Note: IP = IQ = 0 is **not permitted.**

7: M – INTEGER. *Input*

On entry: the value of m, the number of residual cross-correlation matrices to be computed. See Section 8.2 for advice on the choice of M.

Constraint: IP + IQ < M < N.

8: PAR((IP+IQ)*K*K) – ***real*** array. *Input*

On entry: the parameter estimates read in row by row in the order $\phi_1,\phi_2,...,\phi_p$, $\theta_1,\theta_2,...,\theta_q$. Thus, if IP > 0 then PAR$((l-1)\times k\times k+(i-1)\times k+j)$ must be set equal to an estimate of the (i,j)th element of ϕ_l for $l = 1,2,...,p$; $i,j = 1,2,...,k$. If IQ $\geq$ 0 then PAR$(p\times k\times k+(l-1)\times k\times k+(i-1)\times k+j)$ must be set equal to an estimate of the (i,j)th element of θ_l for $l = 1,2,...,q$; $i,j = 1,2,...,k$.

The first $p\times k\times k$ elements of PAR must satisfy the stationarity condition and the next $q\times k\times k$ elements of PAR must satisfy the invertibility condition.

9: PARHLD((IP+IQ)*K*K) – LOGICAL array. *Input*

On entry: PARHLD(i) must be set to .TRUE. if PAR(i) has been held constant at a pre-specified value and .FALSE. if PAR(i) is a free parameter, for $i = 1,2,...,(p+q)\times k\times k$.

10: QQ(IK,K) – ***real*** array. *Input/Output*

On entry: an efficient estimate of the (i,j)th element of Σ. The lower triangle only is needed.

Constraint: QQ must be positive-definite.

On exit: if IFAIL $\neq$ 1, then the upper triangle is set equal to the lower triangle.

11: ISHOW – INTEGER. *Input*

On entry: ISHOW must be non-zero if the residual cross-correlation matrices $\{\hat{r}_{ij}(l)\}$ and their standard errors $\{se(\hat{r}_{ij}(l))\}$, the modified portmanteau statistic with its significance and a summary table are to be printed and zero otherwise. The summary table indicates which elements of the residual correlation matrices are significant at the 5% level in either a positive or negative direction; i.e. if $\hat{r}_{ij}(l) > 1.96\times se(\hat{r}_{ij}(l))$ then a '+' is printed, if $\hat{r}_{ij}(l) < -1.96\times se(\hat{r}_{ij}(l))$ then a '–' is printed, otherwise a '.' is printed. The summary table is only printed if $k \leq 6$ on entry.

The residual cross-correlation matrices, their standard errors and the modified portmanteau statistic with its significance are available also as output variables in R, RCM, CHI, IDF and SIGLEV.

12: RO(IK,K) – ***real*** array. *Output*

On exit: the matrix $\hat{R}_0$. If $i \neq j$, then RO(i,j) contains an estimate of the (i,j)th element of the residual cross-correlation matrix at lag zero. When $i = j$, RO(i,j) contains the standard deviation of the ith residual series. If IFAIL = 3 on exit then the first K rows and columns of R0 are set to zero.

13: R(IK,IK,M) – ***real*** array. *Output*

On exit: an estimate of the (i,j)th element of the residual cross-correlation matrix at lag l, for $l = 1,2,...,m$; $i,j = 1,2,...,k$. If IFAIL = 3 on exit then all elements of R are set to zero.

14: RCM(IRCM,M*K*K) – ***real*** array. *Output*

On exit: the estimated standard errors and correlations of the elements in the array R. The correlation between R(i,j,l) and R(i_2,j_2,l_2) is returned as RCM(s,t) where $s = (l-1)\times k\times k + (j-1)\times k + i$ and $t = (l_2-1)\times k\times k + (j_2-1)\times k + i_2$ except that if $s = t$, then RCM(s,t) contains the standard error of R(i,j,l). If on exit, IFAIL $\geq$ 5, then all off-diagonal elements of RCM are set to zero and all diagonal elements are set to $1/\sqrt{n}$.

15: IRCM – INTEGER. *Input*

On entry: the first dimension of the array RCM as declared in the (sub)program from which G13DSF is called.

Constraint: IRCM $\geq$ M×K×K.

16: CHI – ***real***. *Output*

On exit: the value of the modified portmanteau statistic, $Q^*_{(m)}$. If IFAIL = 3 on exit then CHI is returned as zero.

17: IDF – INTEGER. *Output*

On exit: the number of degrees of freedom of CHI.

18: SIGLEV – ***real***. *Output*

On exit: the significance level of CHI based on IDF degrees of freedom. If IFAIL = 3 on exit then SIGLEV is returned as one.

19: IW(LIW) – INTEGER array. *Workspace*
20: LIW – INTEGER. *Input*

On entry: the dimension of the array IW as declared in the (sub)program from which G13DSF is called.

Constraint: LIW $\geq$ K×max(IP,IQ).

21: WORK(LWORK) – ***real*** array. *Workspace*
22: LWORK – INTEGER. *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which G13DSF is called.

Constraint: if NPAR = $(p+q)k^2$ then,

$$\text{LWORK} \geq k(n+\text{IK}+2) + mk^2(\text{NPAR}+mk^2+1) + 3k^2 + (\text{NPAR}+1)\text{NPAR}.$$

23: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: IFAIL = 0 unless the routine detects an error or gives a warning (see Section 6).

For this routine, because the values of output parameters may be useful even if IFAIL $\neq$ 0

For this routine, because the values of output parameters may be useful even if IFAIL $\neq 0$ on exit, users are recommended to set IFAIL to –1 before entry. **It is then essential to test the value of IFAIL on exit.**

6. Error Indicators and Warnings

Errors or warnings specified by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, K < 1,
or IK < K,
or IP < 0,
or IQ < 0,
or IP = IQ = 0,
or M ≤ IP + IQ,
or M ≥ N,
or IRCM < M×K×K,
or LIW is too small,
or LWORK is too small.

IFAIL = 2

On entry, either QQ is not positive-definite or the autoregressive parameter matrices are extremely close to or outside the stationarity region, or the moving average parameter matrices are extremely close to or outside the invertibility region. To proceed, the user must supply different parameter estimates in the arrays PAR and QQ.

IFAIL = 3

On entry, at least one of the k residual series is such that all its elements are practically identical giving zero (or near zero) variance or at least two of the residual series are identical. In this case CHI is set to zero, SIGLEV to one and all the elements of R0 and R set to zero.

IFAIL = 4

This is an unlikely exit brought about by an excessive number of iterations being needed to evaluate the zeros of the determinantal polynomials $\det(A(\phi))$ and $\det(B(\theta))$. All output parameters are undefined.

IFAIL = 5

On entry, either the eigenvalues and eigenvectors of Δ (the matrix QQ in correlation form) could not be computed or the determinantal polynomials $\det(A(\phi))$ and $\det(B(\theta))$ have a factor in common. To proceed, the user must either supply different parameter estimates in the array QQ or delete this common factor from the model. In this case, the off-diagonal elements of RCM are returned as zero and the diagonal elements set to $1/\sqrt{n}$. All other output quantities will be correct.

IFAIL = 6

This is an unlikely exit. At least one of the diagonal elements of RCM was found to be either negative or zero. In this case all off-diagonal elements of RCM are returned as zero and all diagonal elements of RCM set to $1/\sqrt{n}$.

7. Accuracy

The computations are believed to be stable.

8. Further Comments

8.1. Timing

The time taken by the routine depends upon the number of residual cross-correlation matrices to be computed, m, and the number of time series, k.

8.2. Choice of m

The number of residual cross-correlation matrices to be computed, m should be chosen to ensure that when the ARMA model (1) is written as either an infinite order autoregressive process:

$$\text{i.e.} \quad W_t - \mu = \sum_{j=1}^{\infty} \pi_j (W_{t-j} - \mu) + \varepsilon_t$$

or as an infinite order moving average process:

$$\text{i.e.} \quad W_t - \mu = \sum_{j=1}^{\infty} \psi_j \varepsilon_{t-j} + \varepsilon_t$$

then the two sequences of k by k matrices $\{\pi_1, \pi_2, ...\}$ and $\{\psi_1, \psi_2, ...\}$ are such that π_j and ψ_j are approximately zero for $j > m$. An over-estimate of m is therefore preferable to an under-estimate of m. In many instances the choice $m = 10$ will suffice. In practice, to be on the safe side, the user should try setting $m = 20$.

8.3. Checking a 'White Noise' Model

If the user has fitted the 'white noise' model

$$W_t - \mu = \varepsilon_t$$

then G13DSF should be entered with $p = 1$, $q = 0$; and the first k^2 elements of PAR and PARHLD set to zero and .TRUE. respectively.

8.4. Approximate Standard Errors

When IFAIL is returned as 5 or 6 all the standard errors in RCM are set to $1/\sqrt{n}$. This is the asymptotic standard error of $\hat{r}_{ij}(l)$ when all the autoregressive and moving average parameters are assumed to be known rather than estimated.

8.5. Alternative Tests

$\hat{R}_0$ is useful in testing for instantaneous causality. If the user wishes to carry out a likelihood ratio test then the covariance matrix at lag zero $(\hat{C}_0)$ can be used. It can be recovered from $\hat{R}_0$ by setting:

$$\begin{aligned} \hat{C}_0(i,j) &= \hat{R}_0(i,j) \times \hat{R}_0(i,i) \times \hat{R}_0(j,j), && \text{for } i \neq j \\ &= \hat{R}_0(i,j) \times \hat{R}_0(i,j), && \text{for } i = j \end{aligned}$$

9. Example

A program to fit a bivariate AR(1) model to two series each of length 48. μ has been estimated but $\phi_1(2,1)$ has been constrained to be zero. 10 residual cross-correlation matrices are to be computed.

9.1. Program Text

Note: the listing of the example program presented below uses ***bold italicised*** terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DSF Example Program Text
*     Mark 15 Revised.  NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          KMAX, IK, IPMAX, IQMAX, NMAX, NPARMX, LWORK, LIW,
     +                 MMAX, IRCM, ICM
      PARAMETER        (KMAX=3,IK=KMAX,IPMAX=3,IQMAX=3,NMAX=100,
     +                 NPARMX=(IPMAX+IQMAX)*KMAX*KMAX+KMAX,LWORK=2000,
     +                 LIW=100,MMAX=20,IRCM=MMAX*KMAX*KMAX,ICM=NPARMX)
*     .. Local Scalars ..
      real             CGETOL, CHI, RLOGL, SIGLEV
      INTEGER          I, IDF, IFAIL, IP, IPRINT, IQ, ISHOW, J, K, M,
     +                 MAXCAL, N, NITER, NPAR
      LOGICAL          EXACT, MEAN
*     .. Local Arrays ..
      real             CM(ICM,NPARMX), G(NPARMX), PAR(NPARMX),
     +                 QQ(IK,KMAX), R(IK,IK,MMAX), R0(IK,KMAX),
     +                 RCM(IRCM,MMAX*KMAX*KMAX), V(IK,NMAX), W(IK,NMAX),
     +                 WORK(LWORK)
      INTEGER          IW(LIW)
      LOGICAL          PARHLD(NPARMX)
*     .. External Subroutines ..
      EXTERNAL         G13DCF, G13DSF, X04ABF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DSF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, N
*
      CALL X04ABF(1,NOUT)
*
      IF (K.GT.0 .AND. K.LE.KMAX .AND. N.GE.3 .AND. N.LE.NMAX) THEN
         DO 20 I = 1, K
            READ (NIN,*) (W(I,J),J=1,N)
   20    CONTINUE
         READ (NIN,*) IP, IQ, MEAN, M
         IF (IP.GE.0 .AND. IP.LE.IPMAX .AND. IQ.GE.0 .AND. IQ.LE.IQMAX)
     +       THEN
            NPAR = (IP+IQ)*K*K
            IF (MEAN) NPAR = NPAR + K
            IF (NPAR.LE.NPARMX) THEN
               DO 40 I = 1, NPAR
                  PAR(I) = 0.0e0
                  PARHLD(I) = .FALSE.
   40          CONTINUE
               DO 80 J = 1, K
                  DO 60 I = J, K
                     QQ(I,J) = 0.0e0
   60             CONTINUE
   80          CONTINUE
               PARHLD(3) = .TRUE.
               EXACT = .TRUE.
*              ** Set IPRINT > 0 to obtain intermediate output **
               IPRINT = -1
               CGETOL = 0.0001e0
               MAXCAL = 40*NPAR*(NPAR+5)
               ISHOW = 2
               IFAIL = 1
*
```

```
               CALL G13DCF(K,N,IP,IQ,MEAN,PAR,NPAR,QQ,IK,W,PARHLD,EXACT,
     +                     IPRINT,CGETOL,MAXCAL,ISHOW,NITER,RLOGL,V,G,
     +                     CM,ICM,WORK,LWORK,IW,LIW,IFAIL)
*
               WRITE (NOUT,*)
               IF (IFAIL.NE.0) THEN
                  WRITE (NOUT,99999) 'G13DCF fails. IFAIL =', IFAIL
                  WRITE (NOUT,*)
               END IF
               IF ((IFAIL.EQ.0 .OR. IFAIL.GE.4) .AND. M.LE.MMAX) THEN
                  WRITE (NOUT,*) 'Output from G13DSF'
                  WRITE (NOUT,*)
                  ISHOW = 1
                  IFAIL = -1
*
                  CALL G13DSF(K,N,V,IK,IP,IQ,M,PAR,PARHLD,QQ,ISHOW,R0,R,
     +                        RCM,IRCM,CHI,IDF,SIGLEV,IW,LIW,WORK,LWORK,
     +                        IFAIL)
*
                  IF (IFAIL.NE.0) WRITE (NOUT,99999)
     +                'G13DSF fails. IFAIL =', IFAIL
               END IF
            END IF
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I3)
      END
```

9.2. Program Data

```
G13DSF Example Program Data
2 48  : K, no. of time series, N, no. of obs in each time series
-1.490 -1.620  5.200  6.230  6.210  5.860
 4.090  3.180  2.620  1.490  1.170  0.850
-0.350  0.240  2.440  2.580  2.040  0.400
 2.260  3.340  5.090  5.000  4.780  4.110
 3.450  1.650  1.290  4.090  6.320  7.500
 3.890  1.580  5.210  5.250  4.930  7.380
 5.870  5.810  9.680  9.070  7.290  7.840
 7.550  7.320  7.970  7.760  7.000  8.350
 7.340  6.350  6.960  8.540  6.620  4.970
 4.550  4.810  4.750  4.760 10.880 10.010
11.620 10.360  6.400  6.240  7.930  4.040
 3.730  5.600  5.350  6.810  8.270  7.680
 6.650  6.080 10.250  9.140 17.750 13.300
 9.630  6.800  4.080  5.060  4.940  6.650
 7.940 10.760 11.890  5.850  9.010  7.500
10.020 10.380  8.150  8.370 10.730 12.140 : End of time series
1 0 T 10 : IP, IQ, MEAN and M
```

9.3. Program Results

```
G13DSF Example Program Results

VALUE OF IFAIL PARAMETER ON EXIT FROM G13DCF =   0

VALUE OF LOG LIKELIHOOD FUNCTION ON EXIT = -0.20280E+03

MAXIMUM LIKELIHOOD ESTIMATES OF AR PARAMETER MATRICES
-----------------------------------------------------

PHI(1)    ROW-WISE :    0.802   0.065
                       (0.091) (0.102)

                        0.000   0.575
                       (0.000) (0.121)

MAXIMUM LIKELIHOOD ESTIMATE OF PROCESS MEAN
-------------------------------------------

                        4.271   7.825
                       (1.219) (0.776)

MAXIMUM LIKELIHOOD ESTIMATE OF SIGMA MATRIX
-------------------------------------------

                        2.964

                        0.637   5.380

            RESIDUAL SERIES NUMBER  1
            -------------------------

   T     1      2      3      4      5      6      7      8
 V(T) -3.33  -1.24   5.75   1.27   0.32   0.11  -1.27  -0.73

   T     9     10     11     12     13     14     15     16
 V(T) -0.58  -1.26  -0.67  -1.13  -2.02  -0.57   1.24  -0.13

   T    17     18     19     20     21     22     23     24
 V(T) -0.77  -2.09   1.34   0.95   1.71   0.23  -0.01  -0.60

   T    25     26     27     28     29     30     31     32
 V(T) -0.68  -1.89  -0.77   2.05   2.11   0.94  -3.32  -2.50

   T    33     34     35     36     37     38     39     40
 V(T)  3.16   0.47   0.05   2.77  -0.82   0.25   3.99   0.20

   T    41     42     43     44     45     46     47     48
 V(T) -0.70   1.07   0.44   0.28   1.09   0.50  -0.10   1.70

            RESIDUAL SERIES NUMBER  2
            -------------------------

   T     1      2      3      4      5      6      7      8
 V(T) -0.19  -1.20  -0.02   1.21  -1.62  -2.16  -1.63  -1.13

   T     9     10     11     12     13     14     15     16
 V(T) -1.34  -1.30   4.82   0.43   2.54   0.35  -2.88  -0.77

   T    17     18     19     20     21     22     23     24
 V(T)  1.02  -3.85  -1.92   0.13  -1.20   0.41   1.03  -0.40

   T    25     26     27     28     29     30     31     32
 V(T) -1.09  -1.07   3.43  -0.08   9.17  -0.23  -1.34  -2.06
```

```
   T    33     34     35     36     37     38     39     40
V(T)  -3.16  -0.61  -1.30   0.48   0.79   2.87   2.38  -4.31

   T    41     42     43     44     45     46     47     48
V(T)   2.32  -1.01   2.38   1.29  -1.14   0.36   2.59   2.64
```

Output from G13DSF

```
RESIDUAL CROSS-CORRELATION MATRICES
-----------------------------------

LAG      1             :    0.130    0.112
                          (0.119)  (0.143)
                            0.094    0.043
                          (0.069)  (0.102)

LAG      2             :   -0.312    0.021
                          (0.128)  (0.144)
                           -0.162    0.098
                          (0.125)  (0.132)

LAG      3             :    0.004   -0.176
                          (0.134)  (0.144)
                           -0.168   -0.091
                          (0.139)  (0.140)

LAG      4             :   -0.090   -0.120
                          (0.137)  (0.144)
                            0.099   -0.232
                          (0.142)  (0.143)

LAG      5             :    0.041    0.093
                          (0.140)  (0.144)
                           -0.009   -0.089
                          (0.144)  (0.144)

LAG      6             :    0.234   -0.008
                          (0.141)  (0.144)
                            0.069   -0.103
                          (0.144)  (0.144)

LAG      7             :   -0.076    0.007
                          (0.142)  (0.144)
                            0.168    0.000
                          (0.144)  (0.144)

LAG      8             :   -0.074    0.559
                          (0.143)  (0.144)
                            0.008   -0.101
                          (0.144)  (0.144)

LAG      9             :    0.091    0.193
                          (0.144)  (0.144)
                            0.055    0.170
                          (0.144)  (0.144)

LAG     10             :   -0.060    0.061
                          (0.144)  (0.144)
                            0.191    0.089
                          (0.144)  (0.144)
```

```
SUMMARY TABLE
-------------

LAGS   1 -  10

***************************
*            *            *
* .-........ * .......+.. *
*            *            *
***************************
*            *            *
* .......... * .......... *
*            *            *
***************************

LI-MCLEOD PORTMANTEAU STATISTIC =      49.234
             SIGNIFICANCE LEVEL =       0.086
(BASED ON  37 DEGREES OF FREEDOM)

VALUE OF IFAIL PARAMETER ON EXIT FROM G13DSF =  0
```

G13DXF – NAG Fortran Library Routine Document

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details. The routine name may be precision-dependent.

1. Purpose

G13DXF calculates the zeros of a vector autoregressive (or moving average) operator. This routine is likely to be used in conjunction with G05HDF, G13ASF, G13DCF or G13DSF.

2. Specification

```
      SUBROUTINE G13DXF (K, IP, PAR, RR, RI, RMOD, WORK, IWORK, IFAIL)
      INTEGER          K, IP, IWORK(K*IP), IFAIL
      real             PAR(IP*K*K), RR(K*IP), RI(K*IP),
     1                 RMOD(K*IP), WORK(K*K*IP*IP)
```

3. Description

Consider the vector autoregressive moving average (VARMA) model

$$W_t-\mu = \phi_1(W_{t-1}-\mu) + \phi_2(W_{t-2}-\mu) + \dots + \phi_p(W_{t-p}-\mu) \\ + \epsilon_t - \theta_1\epsilon_{t-1} - \theta_2\epsilon_{t-2} - \dots - \theta_q\epsilon_{t-q} \qquad (1)$$

where W_t denotes a vector of k time series and ϵ_t is a vector of k residual series having zero mean and a constant variance-covariance matrix. The components of ϵ_t are also assumed to be uncorrelated at non-simultaneous lags. $\phi_1, \phi_2, \dots, \phi_p$, denotes a sequence of k by k matrices of autoregressive (AR) parameters and $\theta_1, \theta_2, \dots, \theta_q$, denotes a sequence of k by k matrices of moving average (MA) parameters. μ is a vector of length k containing the series means. Let

$$A(\phi) = \begin{bmatrix} \phi_1 & I & 0 & . & . & . & 0 \\ \phi_2 & 0 & I & 0 & . & . & 0 \\ . & & & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ \phi_{p-1} & 0 & . & . & . & 0 & I \\ \phi_p & 0 & . & . & . & 0 & 0 \end{bmatrix}_{pk\times pk}$$

where I denotes the k by k identity matrix.

The model (1) is said to be stationary if the eigenvalues of $A(\phi)$ lie inside the unit circle. Similarly let

$$B(\theta) = \begin{bmatrix} \theta_1 & I & 0 & . & . & . & 0 \\ \theta_2 & 0 & I & 0 & . & . & 0 \\ . & & & & & & . \\ . & & & & & & . \\ . & & & & & & . \\ \theta_{q-1} & 0 & . & . & . & 0 & I \\ \theta_q & 0 & . & . & . & 0 & 0 \end{bmatrix}_{qk\times qk}$$

Then the model is said to be invertible if the eigenvalues of $B(\theta)$ lie inside the unit circle.

G13DXF returns the pk eigenvalues of $A(\phi)$ (or the qk eigenvalues of $B(\theta)$) along with their moduli, in descending order of magnitude. Thus to check for stationarity or invertibility the user should check whether the modulus of the largest eigenvalue is less than one.

4. References

[1] WEI, W.W.S.
Time Series Analysis: Univariate and Multivariate Methods.
Addison-Wesley, 1990.

5. Parameters

1: K – INTEGER. *Input*

On entry: the dimension, k, of the multivariate time series.

Constraint: K $\geq$ 1.

2: IP – INTEGER. *Input*

On entry: the number of AR (or MA) parameter matrices, p (or q).

Constraint: IP $\geq$ 1.

3: PAR(IP*K*K) – ***real*** array. *Input*

On entry: the AR (or MA) parameter matrices read in row by row in the order $\phi_1, \phi_2, ..., \phi_p$ (or $\theta_1, \theta_2, ..., \theta_q$). That is PAR($(l-1)\times k\times k+(i-1)\times k+j$) must be set equal to the (i,j)th element of ϕ_l, for $l = 1,2,...,p$ (or the (i,j)th element of θ_l, for $l = 1,2,...,q$).

4: RR(K*IP) – ***real*** array. *Output*

On exit: the real parts of the eigenvalues.

5: RI(K*IP) – ***real*** array. *Output*

On exit: the imaginary parts of the eigenvalues.

6: RMOD(K*IP) – ***real*** array. *Output*

On exit: the moduli of the eigenvalues.

7: WORK(K*K*IP*IP) – ***real*** array. *Workspace*

8: IWORK(K*IP) – INTEGER array. *Workspace*

9: IFAIL – INTEGER. *Input/Output*

On entry: IFAIL must be set to 0, –1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6. Error Indicators and Warnings

Errors detected by the routine:

If on entry IFAIL = 0 or –1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

IFAIL = 1

On entry, K < 1,
or IP < 1.

IFAIL = 2

An excessive number of iterations are needed to evaluate the eigenvalues of $A(\phi)$ (or $B(\theta)$). This is an unlikely exit. All output parameters are undefined.

7. Accuracy

The accuracy of the results depends on the original matrix and the multiplicity of the roots.

8. Further Comments

The time taken is approximately proportional to kp^3 (or kq^3).

9. Example

This example program finds the eigenvalues of $A(\phi)$ where $k = 2$ and $p = 1$ and $\phi_1 = \begin{bmatrix} 0.802 & 0.065 \\ 0.000 & 0.575 \end{bmatrix}$.

9.1. Program Text

Note: the listing of the example program presented below uses *bold italicised* terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13DXF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          KMAX, IPMAX
      PARAMETER        (KMAX=6,IPMAX=3)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, IP, K, NPAR
*     .. Local Arrays ..
      real             PAR(KMAX*KMAX*IPMAX), RI(KMAX*IPMAX),
     +                 RMOD(KMAX*IPMAX), RR(KMAX*IPMAX),
     +                 WORK(KMAX*KMAX*IPMAX*IPMAX)
      INTEGER          IW(KMAX*IPMAX)
*     .. External Subroutines ..
      EXTERNAL         G13DXF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13DXF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) K, IP
      IF (K.GT.0 .AND. K.LE.KMAX .AND. IP.GT.0 .AND. IP.LE.IPMAX) THEN
*        Read the AR (or MA) parameters
         NPAR = IP*K*K
         READ (NIN,*) (PAR(I),I=1,NPAR)
         IFAIL = 0
*
         CALL G13DXF(K,IP,PAR,RR,RI,RMOD,WORK,IW,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) '           Eigenvalues        Moduli'
         WRITE (NOUT,*) '           -----------        ------'
*
         DO 20 I = 1, K*IP
            IF (RI(I).GE.0.0e0) THEN
               WRITE (NOUT,99999) RR(I), RI(I), RMOD(I)
            ELSE
               WRITE (NOUT,99998) RR(I), -RI(I), RMOD(I)
            END IF
   20    CONTINUE
      ELSE
         WRITE (NOUT,*) ' Either K or IP is out of range'
      END IF
      STOP
*
99999 FORMAT (' ',F10.3,'  + ',F6.3,' i  ',F8.3)
99998 FORMAT (' ',F10.3,'  - ',F6.3,' i  ',F8.3)
      END
```

9.2. Program Data

```
G13DXF Example Program Data
 2 1
 0.802 0.065
 0.000 0.575
```

9.3. Program Results

```
 G13DXF Example Program Results

        Eigenvalues        Moduli
        -----------        ------
     0.802  +  0.000 i      0.802
     0.575  +  0.000 i      0.575
```

G13EAF – NAG Fortran Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G13EAF performs a combined measurement and time update of one iteration of the time-varying Kalman filter using a square root covariance filter.

2 Specification

```
      SUBROUTINE G13EAF(N, M, L, A, LDS, B, STQ, Q, LDQ, C, LDM, R, S,
     1                  K, H, TOL, IWK, WK, IFAIL)
      INTEGER           N, M, L, LDS, LDQ, LDM, IWK(M), IFAIL
      real              A(LDS,N), B(LDS,L), Q(LDQ,*), C(LDM,N),
     1                  R(LDM,M), S(LDS,N), K(LDS,M), H(LDM,M), TOL,
     2                  WK((N+M)*(N+M+L))
      LOGICAL           STQ
```

3 Description

The Kalman filter arises from the state space model given by:

$$\begin{aligned} X_{i+1} &= A_iX_i + B_iW_i, \quad \mathrm{var}(W_i) = Q_i \\ Y_i &= C_iX_i + V_i, \quad \mathrm{var}(V_i) = R_i \end{aligned}$$

where X_i is the state vector of length n at time i, Y_i is the observation vector of length m at time i and W_i of length l and V_i of length m are the independent state noise and measurement noise respectively.

The estimate of X_i given observations Y_1 to Y_{i-1} is denoted by $\hat{X}_{i|i-1}$ with state covariance matrix $\mathrm{var}(\hat{X}_{i|i-1}) = P_{i|i-1} = S_iS_i^T$ while the estimate of X_i given observations Y_1 to Y_i is denoted by $\hat{X}_{i|i}$ with covariance matrix $\mathrm{var}(\hat{X}_{i|i}) = P_{i|i}$. The update of the estimate, $\hat{X}_{i|i-1}$, from time i to time $(i+1)$, is computed in two stages. First, the measurement-update is given by:

$$\hat{X}_{i|i} = \hat{X}_{i|i-1} + K_i[Y_i - C_i\hat{X}_{i|i-1}] \tag{1}$$

and

$$P_{i|i} = [I - K_iC_i]P_{i|i-1} \tag{2}$$

where $K_i = P_{i|i-1}C_i^T[C_iP_{i|i-1}C_i^T + R_i]^{-1}$ is the Kalman gain matrix. The second stage is the time-update for X which is given by:

$$\hat{X}_{i+1|i} = A_i\hat{X}_{i|i} + D_iU_i \tag{3}$$

and

$$P_{i+1|i} = A_iP_{i|i}A_i^T + B_iQ_iB_i^T \tag{4}$$

where D_iU_i represents any deterministic control used.

The square root covariance filter algorithm provides a stable method for computing the Kalman gain matrix and the state covariance matrix. The algorithm can be summarized as:

$$\begin{pmatrix} R_i^{1/2} & C_iS_i & 0 \\ 0 & A_iS_i & B_iQ_i^{1/2} \end{pmatrix} U = \begin{pmatrix} H_i^{1/2} & 0 & 0 \\ G_i & S_{i+1} & 0 \end{pmatrix} \tag{5}$$

where U is an orthogonal transformation triangularizing the the left-hand pre-array to produce the right-hand post-array. The relationship between the Kalman gain matrix, K_i, and G_i is given by

$$A_iK_i = G_i\left(H_i^{1/2}\right)^{-1}.$$

G13EAF requires the input of the lower triangular Cholesky factors of the noise covariance matrices, $R_i^{1/2}$ and, optionally, $Q_i^{1/2}$ and the lower triangular Cholesky factor of the current state covariance matrix, S_i, and returns the product of the matrices A_i and K_i, $A_i K_i$, the Cholesky factor of the updated state covariance matrix S_{i+1} and the matrix $H_i^{1/2}$ used in the computation of the likelihood for the model.

4 References

[1] Vanbegin M, Van Dooren P and Verhaegen M H G (1989) Algorithm 675: FORTRAN subroutines for computing the square root covariance filter and square root information filter in dense or Hesenberg forms *ACM Trans. Math. Software* **15** 243–256

[2] Verhaegen M H G and Van Dooren P (1986) Numerical aspects of different Kalman filter implementations *IEEE Trans. Auto. Contr.* **AC-31** 907–917

5 Parameters

1: N — INTEGER *Input*

On entry: the size of the state vector, n.

Constraint: N $\geq$ 1.

2: M — INTEGER *Input*

On entry: the size of the observation vector, m.

Constraint: M $\geq$ 1.

3: L — INTEGER *Input*

On entry: the dimension of the state noise, l.

Constraint: L $\geq$ 1.

4: A(LDS,N) — ***real*** array *Input*

On entry: the state transition matrix, A_i.

5: LDS — INTEGER *Input*

On entry: the first dimension of the arrays A, B, S and K as declared in the (sub)program from which G13EAF is called.

Constraint: LDS $\geq$ N.

6: B(LDS,L) — ***real*** array *Input*

On entry: the noise coefficient matrix B_i.

7: STQ — LOGICAL *Input*

On entry: if STQ = .TRUE. then the state noise covariance matrix Q_i is assumed to be the identity matrix. Otherwise the lower triangular Cholesky factor, $Q_i^{1/2}$, must be provided in Q.

8: Q(LDQ,*) — ***real*** array *Input*

Note: the second dimension of the array Q must be at least at least L if STQ = .FALSE. and 1 if STQ = .TRUE..

On entry: if STQ = .FALSE. Q must contain the lower triangular Cholesky factor of the state noise covariance matrix, $Q_i^{1/2}$. Otherwise Q is not referenced.

9: LDQ — INTEGER *Input*

On entry: the first dimension of the array Q as declared in the (sub)program from which G13EAF is called.

Constraint: if STQ = .FALSE., LDQ $\geq$ L otherwise LDQ $\geq$ 1.

10: C(LDM,N) — ***real*** array *Input*

On entry: the measurement coefficient matrix, C_i.

11: LDM — INTEGER *Input*

On entry: the first dimension of the arrays C, R and H as declared in the (sub)program from which G13EAF is called.

Constraint: LDM $\geq$ M.

12: R(LDM,M) — ***real*** array *Input*

On entry: the lower triangular Cholesky factor of the measurement noise covariance matrix, $R_i^{1/2}$.

13: S(LDS,N) — ***real*** array *Input/Output*

On entry: the lower triangular Cholesky factor of the state covariance matrix, S_i.

On exit: the lower triangular Cholesky factor of the state covariance matrix, S_{i+1}.

14: K(LDS,M) — ***real*** array *Output*

On exit: the Kalman gain matrix, K_i, premultiplied by the state transition matrix, A_i, A_iK_i.

15: H(LDM,M) — ***real*** array *Output*

On exit: the lower triangular matrix $H_i^{1/2}$.

16: TOL — ***real*** *Input*

On entry: the tolerance used to test for the singularity of $H_i^{1/2}$. If $0.0 \leq \text{TOL} < m^2 \times$ ***machine precision***, then $m^2\times$***machine precision*** is used instead. The inverse of the condition number of $H^{1/2}$ is estimated by a call to F07TGF(STRCON/DTRCON). If this estimate is less than TOL then $H^{1/2}$ is assumed to be singular.

Suggested value: TOL = 0.0.

Constraint: TOL $\geq$ 0.0.

17: IWK(M) — INTEGER array *Workspace*
18: WK((N+M)*(N+M+L)) — ***real*** array *Workspace*
19: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, N < 1,
or M < 1,
or L < 1,
or LDS < N,
or LDM < M,
or STQ = .TRUE. and LDQ < 1,
or STQ = .FALSE. and LDQ < L,
or TOL < 0.0.

IFAIL = 2

The matrix $H_i^{1/2}$ is singular.

7 Accuracy

The use of the square root algorithm improves the stability of the computations as compared with the direct coding of the Kalman filter. The accuracy will depend on the model.

8 Further Comments

For models with time-invariant A, B and C, G13EBF can be used.

The estimate of the state vector $\hat{X}_{i+1|i}$ can be computed from $\hat{X}_{i|i-1}$ by:

$$\hat{X}_{i+1|i} = A_i\hat{X}_{i|i-1} + AK_i r_i$$

where

$$r_i = Y_i - C_i\hat{X}_{i|i-1}$$

are the independent one step prediction residuals. The required matrix-vector multiplications can be performed by F06PAF (SGEMV/DGEMV).

If W_i and V_i are independent multivariate Normal variates then the log-likelihood for observations $i = 1, 2, \ldots, t$ is given by

$$l(\theta) = \kappa - \frac{1}{2}\sum_{i=1}^{t} ln(\det(H_i)) - \frac{1}{2}\sum_{i=1}^{t}(Y_i - C_iX_{i|i-1})^T H_i^{-1}(Y_i - C_iX_{i|i-1})$$

where κ is a constant.

The Cholesky factors of the covariance matrices can be computed using F07FDF (SPOTRF/DPOTRF).

Note that the model:

$$\begin{aligned} X_{i+1} &= A_iX_i + W_i, \quad \text{var}(W_i) = Q_i \\ Y_i &= C_iX_i + V_i, \quad \text{var}(V_i) = R_i \end{aligned}$$

can be specified either with B set to the identity matrix and STQ = .FALSE. and the matrix $Q^{1/2}$ input in Q or with STQ = .TRUE. and B set to $Q^{1/2}$.

The algorithm requires $\frac{7}{6}n^3 + n^2(\frac{5}{2}m + l) + n(\frac{1}{2}l^2 + m^2)$ operations and is backward stable (see Verhaegen and Van Dooren [2]).

9 Example

The example program first inputs the number of updates to be computed and the problem sizes. The initial state vector and state covariance matrix are input followed by the model matrices A_i, B_i, C_i, R_i and optionally Q_i. The Cholesky factors of the covariance matrices can be computed if required. The model matrices can be input at each update or only once at the first step. At each update the observed values are input and the residuals are computed and printed and the estimate of the state vector, $\hat{X}_{i|i-1}$, and the deviance are updated. The deviance is $-2\times$log-likelihood ignoring the constant. After the final update the state covariance matrix is computed from S and printed along with final estimate of the state vector and the value of the deviance.

The data is for a two dimensional time series to which a VARMA(1,1) has been fitted. For the specification of a VARMA model as a state space model see the Chapter Introduction. The initial value of P, P_0, is the solution to:

$$P_0 = A_1P_0A_1^T + B_1Q_1B_1^T.$$

For convenience, the mean of each series is input before the first update and subtracted from the observations before the measurement update is computed.

9.1 Example Text

Note: the listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13EAF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX, LMAX
      PARAMETER        (NMAX=4,MMAX=2,LMAX=2)
*     .. Local Scalars ..
      real             DEV, TOL
      INTEGER          I, IFAIL, INFO, ISTEP, J, L, LDM, LDQ, LDS, M, N,
     +                 NCALL
      LOGICAL          CONST, FULL, STQ
*     .. Local Arrays ..
      real             A(NMAX,NMAX), AX(NMAX), B(NMAX,LMAX),
     +                 C(MMAX,NMAX), H(MMAX,MMAX), K(NMAX,MMAX),
     +                 P(NMAX,NMAX), Q(LMAX,LMAX), R(MMAX,MMAX),
     +                 S(NMAX,NMAX), WK((NMAX+MMAX)*(NMAX+MMAX+LMAX)),
     +                 X(NMAX), Y(MMAX), YMEAN(MMAX)
      INTEGER          IWK(MMAX)
*     .. External Functions ..
      real             sdot
      EXTERNAL         sdot
*     .. External Subroutines ..
      EXTERNAL         saxpy, scopy, sgemv, spotrf, strmv, strsv, G13EAF
*     .. Intrinsic Functions ..
      INTRINSIC        LOG
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13EAF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NCALL, N, M, L, STQ, FULL, CONST
      IF (N.LE.NMAX .AND. M.LE.MMAX .AND. L.LE.LMAX) THEN
         LDS = NMAX
         LDM = MMAX
         LDQ = LMAX
         READ (NIN,*) ((S(I,J),J=1,N),I=1,N)
         IF (FULL) THEN
            CALL spotrf('L',N,S,LDS,INFO)
            IF (INFO.GT.0) THEN
               WRITE (NOUT,*) ' S not positive definite'
               GO TO 100
            END IF
         END IF
         READ (NIN,*) (X(I),I=1,N)
         READ (NIN,*) (YMEAN(I),I=1,M)
         TOL = 0.0e0
         DEV = 0.0e0
         WRITE (NOUT,*)
         WRITE (NOUT,*) '         Residuals'
         WRITE (NOUT,*)
*
*         Loop through data
*
         DO 40 ISTEP = 1, NCALL
```

```
            IF ( .NOT. CONST .OR. ISTEP.EQ.1) THEN
               READ (NIN,*) ((A(I,J),J=1,N),I=1,N)
               READ (NIN,*) ((B(I,J),J=1,L),I=1,N)
               READ (NIN,*) ((C(I,J),J=1,N),I=1,M)
               READ (NIN,*) ((R(I,J),J=1,M),I=1,M)
               IF (FULL .AND. R(1,1).NE.0.0e0) THEN
                  CALL spotrf('L',M,R,LDM,INFO)
                  IF (INFO.GT.0) THEN
                     WRITE (NOUT,*) ' R not positive definite'
                     GO TO 100
                  END IF
               END IF
               IF ( .NOT. STQ) THEN
                  READ (NIN,*) ((Q(I,J),J=1,L),I=1,L)
                  IF (FULL) THEN
                     CALL spotrf('L',L,Q,LDQ,INFO)
                     IF (INFO.GT.0) THEN
                        WRITE (NOUT,*) ' Q not positive definite'
                        GO TO 100
                     END IF
                  END IF
               END IF
            END IF
            IFAIL = 0
*
            CALL G13EAF(N,M,L,A,LDS,B,STQ,Q,LDQ,C,LDM,R,S,K,H,TOL,IWK,
     +                  WK,IFAIL)
*
            READ (NIN,*) (Y(I),I=1,M)
            CALL saxpy(M,-1.0e0,YMEAN,1,Y,1)
*
*        Perform time and measurement update
*
            CALL sgemv('N',M,N,-1.0e0,C,LDM,X,1,1.0e0,Y,1)
            WRITE (NOUT,99999) (Y(I),I=1,M)
            CALL sgemv('N',N,N,1.0e0,A,LDS,X,1,0.0e0,AX,1)
            CALL sgemv('N',N,M,1.0e0,K,LDS,Y,1,1.0e0,AX,1)
            CALL scopy(N,AX,1,X,1)
*
*        Update loglikelihood
*
            CALL strsv('L','N','N',M,H,LDM,Y,1)
            DEV = DEV + sdot(M,Y,1,Y,1)
            DO 20 I = 1, M
               DEV = DEV + 2.0e0*LOG(H(I,I))
   20       CONTINUE
   40    CONTINUE
*
*        Compute P from S
*
         DO 60 I = 1, N
            CALL scopy(I,S(I,1),LDS,P(1,I),1)
            CALL strmv('L','N','N',I,S,LDS,P(1,I),1)
            CALL scopy(I-1,P(1,I),1,P(I,1),LDS)
   60    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Final X(I+1:I) '
         WRITE (NOUT,*)
```

```
            WRITE (NOUT,99999) (X(J),J=1,N)
            WRITE (NOUT,*)
            WRITE (NOUT,*) ' Final Value of P'
            WRITE (NOUT,*)
            DO 80 I = 1, N
               WRITE (NOUT,99999) (P(I,J),J=1,I)
   80       CONTINUE
            WRITE (NOUT,*)
            WRITE (NOUT,99998) ' Deviance = ', DEV
         END IF
  100 CONTINUE
      STOP
*
99999 FORMAT (6F12.4)
99998 FORMAT (A,e13.4)
      END
```

9.2 Example Data

```
G13EAF Example Program Data

48 4 2 2 F T T

 8.2068  2.0599  1.4807  0.3627
 2.0599  7.9645  0.9703  0.2136
 1.4807  0.9703  0.9253  0.2236
 0.3627  0.2136  0.2236  0.0542

 0.000  0.000  0.000  0.000

 4.404  7.991

 0.607 -0.033  1.000  0.000
 0.000  0.543  0.000  1.000
 0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000

 1.000  0.000
 0.000  1.000
 0.543  0.125
 0.134  0.026

 1.000  0.000  0.000  0.000
 0.000  1.000  0.000  0.000

 0.000  0.000
 0.000  0.000

 2.598  0.560
 0.560  5.330

-1.490  7.340
-1.620  6.350
 5.200  6.960
 6.230  8.540
 6.210  6.620
 5.860  4.970
 4.090  4.550
 3.180  4.810
 2.620  4.750
 1.490  4.760
 1.170 10.880
 0.850 10.010
-0.350 11.620
 0.240 10.360
 2.440  6.400
 2.580  6.240
 2.040  7.930
 0.400  4.040
 2.260  3.730
 3.340  5.600
 5.090  5.350
 5.000  6.810
 4.780  8.270
 4.110  7.680
 3.450  6.650
 1.650  6.080
 1.290 10.250
 4.090  9.140
 6.320 17.750
 7.500 13.300
 3.890  9.630
 1.580  6.800
 5.210  4.080
 5.250  5.060
 4.930  4.940
 7.380  6.650
 5.870  7.940
 5.810 10.760
 9.680 11.890
 9.070  5.850
 7.290  9.010
 7.840  7.500
 7.550 10.020
 7.320 10.380
 7.970  8.150
 7.760  8.370
 7.000 10.730
 8.350 12.140
```

9.3 Example Results

```
G13EAF Example Program Results

          Residuals

     -5.8940       -0.6510
     -1.4710       -1.0407
      5.1658        0.0447
     -1.3280        0.4580
      1.3652       -1.5066
     -0.2337       -2.4192
     -0.8685       -1.7065
     -0.4624       -1.1519
     -0.7510       -1.4218
     -1.3526       -1.3335
```

```
    -0.6707      4.8593
    -1.7389      0.4138
    -1.6376      2.7549
    -0.6137      0.5463
     0.9067     -2.8093
    -0.8255     -0.9355
    -0.7494      1.0247
    -2.2922     -3.8441
     1.8812     -1.7085
    -0.7112     -0.2849
     1.6747     -1.2400
    -0.6619      0.0609
     0.3271      1.0074
    -0.8165     -0.5325
    -0.2759     -1.0489
    -1.9383     -1.1186
    -0.3131      3.5855
     1.3726     -0.1289
     1.4153      8.9545
     0.3672     -0.4126
    -2.3659     -1.2823
    -1.0130     -1.7306
     3.2472     -3.0836
    -1.1501     -1.1623
     0.6855     -1.2751
     2.3432      0.2570
    -1.6892      0.3565
     1.3871      3.0138
     3.3840      2.1312
    -0.5118     -4.7670
     0.8569      2.3741
     0.9558     -1.2209
     0.6778      2.1993
     0.4304      1.1393
     1.4987     -1.2255
     0.5361      0.1237
     0.2649      2.4582
     2.0095      2.5623

 Final X(I+1:I)

     3.6698      2.5888      0.0000      0.0000

 Final Value of P

     2.5980
     0.5600      5.3300
     1.4807      0.9703      0.9253
     0.3627      0.2136      0.2236      0.0542

Deviance =    0.2229E+03
```

G13EBF – NAG Fortran Library Routine Document

Note: Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

G13EBF performs a combined measurement and time update of one iteration of the time-invariant Kalman filter using a square root covariance filter.

2 Specification

```
      SUBROUTINE G13EBF(TRANSF, N, M, L, A, LDS, B, STQ, Q, LDQ, C, LDM,
     1                  R, S, K, H, U, TOL, IWK, WK, IFAIL)
      real              A(LDS,N), B(LDS,L), Q(LDQ,*), C(LDM,N),
     1                  R(LDM,M), S(LDS,N), K(LDS,M), H(LDM,M),
     2                  U(LDS,N), TOL, WK((N+M)*(N+M+L))
      INTEGER           N, M, L, LDS, LDQ, LDM, IWK(M) IFAIL
      CHARACTER*1       TRANSF
      LOGICAL           STQ
```

3 Description

The Kalman filter arises from the state space model given by:

$$\begin{aligned} X_{i+1} &= AX_i + BW_i, \quad \mathrm{var}(W_i) = Q_i \\ Y_i &= CX_i + V_i, \quad \mathrm{var}(V_i) = R_i \end{aligned}$$

where X_i is the state vector of length n at time i, Y_i is the observation vector of length m at time i and W_i of length l and V_i of length m are the independent state noise and measurement noise respectively. The matrices A, B and C are time invariant.

The estimate of X_i given observations Y_1 to Y_{i-1} is denoted by $\hat{X}_{i|i-1}$ with state covariance matrix $\mathrm{var}(\hat{X}_{i|i-1}) = P_{i|i-1} = S_i S_i^T$ while the estimate of X_i given observations Y_1 to Y_i is denoted by $\hat{X}_{i|i}$ with covariance matrix $\mathrm{var}(\hat{X}_{i|i}) = P_{i|i}$. The update of the estimate, $\hat{X}_{i|i-1}$, from time i to time $(i+1)$ is computed in two stages. First, the measurement-update is given by:

$$\hat{X}_{i|i} = \hat{X}_{i|i-1} + K_i[Y_i - C\hat{X}_{i|i-1}] \tag{1}$$

where $K_i = P_{i|i}C^T[CP_{i|i}C^T + R_i]^{-1}$ is the Kalman gain matrix. The second stage is the time-update for X which is given by:

$$\hat{X}_{i+1|i} = A\hat{X}_{i|i} + D_iU_i \tag{2}$$

where D_iU_i represents any deterministic control used.

The square root covariance filter algorithm provides a stable method for computing the Kalman gain matrix and the state covariance matrix. The algorithm can be summarized as:

$$\begin{pmatrix} R_i^{1/2} & 0 & CS_i \\ 0 & BQ_i^{1/2} & AS_i \end{pmatrix} U = \begin{pmatrix} H_i^{1/2} & 0 & 0 \\ G_i & S_{i+1} & 0 \end{pmatrix}$$

where U is an orthogonal transformation triangularizing the the left-hand pre-array to produce the right-hand post-array. The triangularization is carried out via Householder transformations exploiting the zero pattern of the pre-array. The relationship between the Kalman gain matrix K_i and G_i is given by

$$AK_i = G_i\left(H_i^{1/2}\right)^{-1}.$$

In order to exploit the invariant parts of the model to simplify the computation of U the results for the transformed state space U^*X are computed where U^* is the transformation that reduces the matrix pair (A, C) to lower observer Hessenberg form. That is, the matrix U^* is computed such that the compound matrix

$$\left[\begin{array}{c} CU^{*T} \\ U^*AU^{*T} \end{array}\right]$$

is a lower trapezoidal matrix. Further the matrix B is transformed to U^*B. These transformations need only be computed once at the start of a series, and G13EBF will, optionally, compute them. G13EBF returns transformed matrices U^*AU^{*T}, U^*B, CU^{*T} and U^*AK_i, the Cholesky factor of the updated transformed state covariance matrix S^*_{i+1} (where $U^*P_{i+1|i}U^{*T} = S^*_{i+1}S^{*T}_{i+1}$) and the matrix $H_i^{1/2}$, valid for both transformed and original models, which is used in the computation of the likelihood for the model. Note that the covariance matrices Q_i and R_i can be time-varying.

4 References

[1] Vanbegin M, Van Dooren P and Verhaegen M H G (1989) Algorithm 675: FORTRAN subroutines for computing the square root covariance filter and square root information filter in dense or Hesenberg forms *ACM Trans. Math. Software* **15** 243–256

[2] Verhaegen M H G and Van Dooren P (1986) Numerical aspects of different Kalman filter implementations *IEEE Trans. Auto. Contr.* **AC-31** 907–917

5 Parameters

1: TRANSF — CHARACTER*1 *Input*

On entry: indicates whether to transform the input matrix pair (A, C) to lower observer Hessenberg form. The transformation will only be required on the first call to G13EBF. If TRANSF = 'T' the matrices in arrays A and C are transformed to lower observer Hessenberg form and the matrices in B and S are transformed as described in Section 3. If TRANSF = 'H' the matrices in arrays A, C and B should be as returned from a previous call to G13EBF with TRANSF = 'T'.

Constraint: TRANSF = 'T' or 'H'.

2: N — INTEGER *Input*

On entry: the size of the state vector, n.

Constraint: N $\geq$ 1.

3: M — INTEGER *Input*

On entry: the size of the observation vector, m.

Constraint: M $\geq$ 1.

4: L — INTEGER *Input*

On entry: the dimension of the state noise, l.

Constraint: L $\geq$ 1.

5: A(LDS,N) — ***real*** array *Input/Output*

On entry: if TRANSF = 'T', the state transition matrix, A. If TRANSF = 'H', the transformed matrix as returned by a previous call to G13EBF with TRANSF = 'T'.

On exit: if TRANSF = 'T', the transformed matrix, U^*AU^{*T}, otherwise A is unchanged.

6: LDS — INTEGER *Input*

On entry: the first dimension of the arrays A, B, S, K and U as declared in the (sub)program from which G13EBF is called.

Constraint: LDS $\geq$ N.

7: B(LDS,L) — ***real*** array *Input/Output*

On entry: if TRANSF = 'T', the noise coefficient matrix B. If TRANSF = 'H', the transformed matrix as returned by a previous call to G13EBF with TRANSF = 'T'.

On exit: if TRANSF = 'T', the transformed matrix, U^*B, otherwise B is unchanged.

8: STQ — LOGICAL *Input*

On entry: if STQ = .TRUE. then the state noise covariance matrix Q_i is assumed to be the identity matrix. Otherwise the lower triangular Cholesky factor, $Q_i^{1/2}$, must be provided in Q.

9: Q(LDQ,*) — ***real*** array *Input*

Note: the second dimension of the array Q must be at least at least L if STQ = .FALSE. and 1 if STQ = .TRUE..

On entry: if STQ = .FALSE. Q must contain the lower triangular Cholesky factor of the state noise covariance matrix, $Q_i^{1/2}$. Otherwise Q is not referenced.

10: LDQ — INTEGER *Input*

On entry: the first dimension of the array Q as declared in the (sub)program from which G13EBF is called.

Constraint: if STQ = .FALSE., LDQ $\geq$ L otherwise LDQ $\geq$ 1.

11: C(LDM,N) — ***real*** array *Input/Output*

On entry: if TRANSF = 'T', the measurement coefficient matrix, C. If TRANSF = 'H', the transformed matrix as returned by a previous call to G13EBF with TRANSF = 'T'.

On exit: if TRANSF = 'T', the transformed matrix, CU^{*T}, otherwise C is unchanged.

12: LDM — INTEGER *Input*

On entry: the first dimension of the arrays C, R and H as declared in the (sub)program from which G13EBF is called.

Constraint: LDM $\geq$ M.

13: R(LDM,M) — ***real*** array *Input*

On entry: the lower triangular Cholesky factor of the measurement noise covariance matrix, $R_i^{1/2}$.

14: S(LDS,N) — ***real*** array *Input/Output*

On entry: if TRANSF = 'T' the lower triangular Cholesky factor of the state covariance matrix, S_i. If TRANSF = 'H' the lower triangular Cholesky factor of the covariance matrix of the transformed state vector S_i^* as returned from a previous call to G13EBF with TRANSF = 'T'.

On exit: the lower triangular Cholesky factor of the transformed state covariance matrix, S_{i+1}^*.

15: K(LDS,M) — ***real*** array *Output*

On exit: the Kalman gain matrix for the transformed state vector premultiplied by the state transformed transition matrix, U^*AK_i.

16: H(LDM,M) — ***real*** array *Output*

On exit: the lower triangular matrix $H_i^{1/2}$.

17: U(LDS,N) — ***real*** array *Output*

On exit: if TRANSF = 'T' the n by n transformation matrix U^*, otherwise U is not referenced.

18: TOL — ***real*** *Input*

On entry: the tolerance used to test for the singularity of $H_i^{1/2}$. If $0.0 \leq \text{TOL} < m^2 \times$ ***machine precision***, then $m^2 \times$***machine precision*** is used instead. The inverse of the condition number of $H^{1/2}$ is estimated by a call to F07TGF(STRCON/DTRCON). If this estimate is less than TOL then $H^{1/2}$ is assumed to be singular.

Suggested value: TOL = 0.0.

Constraint: TOL $\geq$ 0.0.

19: IWK(M) — INTEGER array *Workspace*

20: WK((N+M)*(N+M+L)) — ***real*** array *Workspace*

21: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Errors and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL=1

On entry, TRANSF $\neq$ 'T' or 'H',
or N < 1,
or M < 1,
or L < 1,
or LDS < N,
or LDM < M,
or STQ = .TRUE. and LDQ < 1,
or STQ = .FALSE. and LDQ < L,
or TOL < 0.0.

IFAIL = 2

The matrix $H_i^{1/2}$ is singular.

7 Accuracy

The use of the square root algorithm improves the stability of the computations as compared with the direct coding of the Kalman filter. The accuracy will depend on the model.

8 Further Comments

For models with time-varying A, B and C, G13EAF can be used.

The initial estimate of the transformed state vector can be computed from the estimate of the original state vector $\hat{X}_{1|0}$, say, by premultiplying it by U^* as returned by G13EBF with TRANSF = 'T', that is $\hat{X}^*_{1|0} = U^*\hat{X}_{1|0}$ The estimate of the transformed state vector $\hat{X}^*_{i+1|i}$ can be computed from the previous value $\hat{X}^*_{i|i-1}$ by:

$$\hat{X}^*_{i+1|i} = (U^*AU^{*T})\hat{X}^*_{i|i-1} + (U^*AK_i)r_i$$

where

$$r_i = Y_i - (CU^{*T})\hat{X}^*_{i|i-1}$$

are the independent one step prediction residuals for both the transformed and original model. The estimate of the original state vector can be computed from the transformed state vector as $U^{*T}\hat{X}^*_{1+1|i}$. The required matrix-vector multiplications can be performed by F06PAF (SGEMV/DGEMV).

If W_i and V_i are independent multivariate Normal variates then the log-likelihood for observations $i = 1, 2, \ldots, t$ is given by

$$l(\theta) = \kappa - \frac{1}{2}\sum_{i=1}^{t} ln(\det(H_i)) - \frac{1}{2}\sum_{i=1}^{t}(Y_i - C_i X_{i|i-1})^T H_i^{-1}(Y_i - C_i X_{i|i-1})$$

where κ is a constant.

The Cholesky factors of the covariance matrices can be computed using F07FDF (SPOTRF/DPOTRF).

Note that the model:

$$\begin{aligned} X_{i+1} &= AX_i + W_i, \quad \text{var}(W_i) = Q_i \\ Y_i &= CX_i + V_i, \quad \text{var}(V_i) = R_i \end{aligned}$$

can be specified either with B set to the identity matrix and STQ = .FALSE. and the matrix $Q^{1/2}$ input in Q or with STQ = .TRUE. and B set to $Q^{1/2}$.

The algorithm requires $\frac{1}{6}n^3 + n^2(\frac{3}{2}m + l) + 2nm^2 + \frac{2}{3}p^3$ operations and is backward stable (see Verhaegen and Van Dooren [2]). The transformation to lower observer Hessenberg form requires $O((n+m)n^2)$ operations.

9 Example

The example program first inputs the number of updates to be computed and the problem sizes. The initial state vector and the Cholesky factor of the state covariance matrix are input followed by the model matrices $A, B, C, R^{1/2}$ and optionally $Q^{1/2}$ (the Cholesky factors of the covariance matrices being input). At the first update the matrices are transformed using the TRANSF = 'T' option and the inital value of the state vector is transformed. At each update the observed values are input and the residuals are computed and printed and the estimate of the transformed state vector, $\hat{U}^* X_{i|i-1}$, and the deviance are updated. The deviance is $-2\times$log-likelihood ignoring the constant. After the final update the estimate of the state vector is computed from the transformed state vector and the state covariance matrix is computed from S and these are printed along with the value of the deviance.

The data is for a two dimensional time series to which a VARMA(1,1) has been fitted. For the specification of a VARMA model as a state space model see the Chapter Introduction. The means of the two series are included as addtional states that do not change over time. The initial value of P, P_0, is the solution to:

$$P_0 = AP_0A^T + BQB^T.$$

9.1 Example Text

Note: the listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     G13EBF Example Program Text
*     Mark 17 Release. NAG Copyright 1995.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX, LMAX
      PARAMETER        (NMAX=6,MMAX=2,LMAX=2)
*     .. Local Scalars ..
      real             DEV, TOL
      INTEGER          I, IFAIL, INFO, ISTEP, J, L, LDM, LDQ, LDS, M, N,
     +                 NCALL
      LOGICAL          FULL, STQ
      CHARACTER        TRANSF
*     .. Local Arrays ..
      real             A(NMAX,NMAX), AX(NMAX), B(NMAX,LMAX),
     +                 C(MMAX,NMAX), H(MMAX,MMAX), K(NMAX,MMAX),
     +                 P(NMAX,NMAX), Q(LMAX,LMAX), R(MMAX,MMAX),
     +                 S(NMAX,NMAX), U(NMAX,NMAX), US(NMAX,NMAX),
     +                 WK((NMAX+MMAX)*(NMAX+MMAX+LMAX)), X(NMAX),
     +                 Y(MMAX)
      INTEGER          IWK(MMAX)
*     .. External Functions ..
      real             sdot
      EXTERNAL         sdot
*     .. External Subroutines ..
      EXTERNAL         scopy, sgemv, spotrf, ssyrk, strsv, F06QHF,
     +                 G13EBF
*     .. Intrinsic Functions ..
      INTRINSIC        LOG
*     .. Executable Statements ..
      WRITE (NOUT,*) 'G13EBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NCALL, N, M, L, STQ, FULL
      IF (N.LE.NMAX .AND. M.LE.MMAX .AND. L.LE.LMAX) THEN
         LDS = NMAX
         LDM = MMAX
         LDQ = LMAX
         CALL F06QHF('G',N,N,0.0e0,0.0e0,S,LDS)
         READ (NIN,*) ((S(I,J),J=1,N),I=1,N)
         IF (FULL) THEN
            CALL spotrf('L',N,S,LDS,INFO)
            IF (INFO.GT.0) THEN
               WRITE (NOUT,*) ' S not positive definite'
               GO TO 100
            END IF
         END IF
         READ (NIN,*) (AX(I),I=1,N)
         READ (NIN,*) ((A(I,J),J=1,N),I=1,N)
         READ (NIN,*) ((B(I,J),J=1,L),I=1,N)
         READ (NIN,*) ((C(I,J),J=1,N),I=1,M)
         CALL F06QHF('G',M,M,0.0e0,0.0e0,R,LDM)
         READ (NIN,*) ((R(I,J),J=1,M),I=1,M)
         IF (FULL) THEN
            CALL spotrf('L',M,R,LDM,INFO)
            IF (INFO.GT.0) THEN
               WRITE (NOUT,*) ' R not positive definite'
               GO TO 100
            END IF
         END IF
         IF ( .NOT. STQ) THEN
            READ (NIN,*) ((Q(I,J),J=1,L),I=1,L)
            IF (FULL) THEN
               CALL spotrf('L',L,Q,LDQ,INFO)
               IF (INFO.GT.0) THEN
                  WRITE (NOUT,*) ' Q not positive definite'
                  GO TO 100
               END IF
            END IF
         END IF

         TOL = 0.0e0
         DEV = 0.0e0
         TRANSF = 'T'
         WRITE (NOUT,*)
         WRITE (NOUT,*) '        Residuals'
         WRITE (NOUT,*)
*
*         Loop through data
*
         DO 40 ISTEP = 1, NCALL
            IFAIL = 0

            IF (ISTEP.EQ.1) THEN
*
*         Make first call to G13EBF
*
               CALL G13EBF('T',N,M,L,A,LDS,B,STQ,Q,LDQ,C,LDM,R,S,K,H,U,
     +                     TOL,IWK,WK,IFAIL)
*
               CALL sgemv('N',N,N,1.0e0,U,LDS,AX,1,0.0e0,X,1)
            ELSE
*
               CALL G13EBF('H',N,M,L,A,LDS,B,STQ,Q,LDQ,C,LDM,R,S,K,H,U,
     +                     TOL,IWK,WK,IFAIL)
            END IF
*
            READ (NIN,*) (Y(I),I=1,M)
*
*        Perform time and measurement update
*
            CALL sgemv('N',M,N,-1.0e0,C,LDM,X,1,1.0e0,Y,1)
            WRITE (NOUT,99999) (Y(I),I=1,M)
            CALL sgemv('N',N,N,1.0e0,A,LDS,X,1,0.0e0,AX,1)
            CALL sgemv('N',N,M,1.0e0,K,LDS,Y,1,1.0e0,AX,1)
            CALL scopy(N,AX,1,X,1)
*
*        Update loglikelihood
*
            CALL strsv('L','N','N',M,H,LDM,Y,1)
            DEV = DEV + sdot(M,Y,1,Y,1)
            DO 20 I = 1, M
               DEV = DEV + 2.0e0*LOG(H(I,I))
   20       CONTINUE
            TRANSF = 'H'
   40    CONTINUE
*
*        Calculate back-transformed X
*
         CALL sgemv('T',N,N,1.0e0,U,LDS,AX,1,0.0e0,X,1)
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Final X(I+1:I) '
         WRITE (NOUT,*)
         WRITE (NOUT,99999) (X(J),J=1,N)
*
```

```
*        Compute back-transformed P from S
*
         DO 60 I = 1, N
            CALL sgemv('T',N-I+1,N,1.0e0,U(I,1),LDS,S(I,I),1,0.0e0,
     +                 US(1,I),1)
   60    CONTINUE
         CALL ssyrk('L','N',N,N,1.0e0,US,LDS,0.0e0,P,LDS)
         WRITE (NOUT,*)
         WRITE (NOUT,*) ' Final Value of P'
         WRITE (NOUT,*)
         DO 80 I = 1, N
            WRITE (NOUT,99999) (P(I,J),J=1,I)
   80    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,99998) ' Deviance = ', DEV
      END IF
  100 CONTINUE
      STOP
*
99999 FORMAT (6F12.4)
99998 FORMAT (A,e13.4)
      END
```

9.2 Example Data

```
G13EBF Example Program Data

48 6 2 2 F F

 2.8648  0.0000  0.0000  0.0000  0.0000  0.0000
 0.7191  2.7290  0.0000  0.0000  0.0000  0.0000
 0.5169  0.2194  0.7810  0.0000  0.0000  0.0000
 0.1266  0.0449  0.1899  0.0098  0.0000  0.0000
 0.0000  0.0000  0.0000  0.0000  0.0000  0.0000
 0.0000  0.0000  0.0000  0.0000  0.0000  0.0000

 0.000  0.000  0.000  0.000  4.404  7.991

 0.607 -0.033  1.000  0.000  0.000  0.000
 0.000  0.543  0.000  1.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  0.000  0.000
 0.000  0.000  0.000  0.000  1.000  0.000
 0.000  0.000  0.000  0.000  0.000  1.000


 1.000  0.000
 0.000  1.000
 0.543  0.125
 0.134  0.026
 0.000  0.000
 0.000  0.000

 1.000  0.000  0.000  0.000  1.000  0.000
 0.000  1.000  0.000  0.000  0.000  1.000

 0.000  0.000
 0.000  0.000
```

```
  1.612  0.000
  0.347  2.282

 -1.490  7.340
 -1.620  6.350
  5.200  6.960
  6.230  8.540
  6.210  6.620
  5.860  4.970
  4.090  4.550
  3.180  4.810
  2.620  4.750
  1.490  4.760
  1.170 10.880
  0.850 10.010
 -0.350 11.620
  0.240 10.360
  2.440  6.400
  2.580  6.240
  2.040  7.930
  0.400  4.040
  2.260  3.730
  3.340  5.600
  5.090  5.350
  5.000  6.810
  4.780  8.270
  4.110  7.680
  3.450  6.650
  1.650  6.080
  1.290 10.250
  4.090  9.140
  6.320 17.750
  7.500 13.300
  3.890  9.630
  1.580  6.800
  5.210  4.080
  5.250  5.060
  4.930  4.940
  7.380  6.650
  5.870  7.940
  5.810 10.760
  9.680 11.890
  9.070  5.850
  7.290  9.010
  7.840  7.500
  7.550 10.020
  7.320 10.380
  7.970  8.150
  7.760  8.370
  7.000 10.730
  8.350 12.140
```

9.3 Example Results

```
 G13EBF Example Program Results

           Residuals

        -5.8940      -0.6510
        -1.4710      -1.0407
         5.1658       0.0447
        -1.3281       0.4580
         1.3653      -1.5066
        -0.2337      -2.4192
        -0.8685      -1.7065
        -0.4624      -1.1519
        -0.7510      -1.4218
        -1.3526      -1.3335
        -0.6707       4.8593
        -1.7389       0.4138
        -1.6376       2.7549
        -0.6137       0.5463
         0.9067      -2.8093
        -0.8255      -0.9355
        -0.7494       1.0247
        -2.2922      -3.8441
         1.8812      -1.7085
        -0.7112      -0.2849
         1.6747      -1.2400
        -0.6619       0.0609
         0.3271       1.0074
        -0.8165      -0.5325
        -0.2759      -1.0489
        -1.9383      -1.1186
        -0.3131       3.5855
         1.3726      -0.1289
         1.4153       8.9545
         0.3672      -0.4126
        -2.3659      -1.2823
        -1.0130      -1.7306
         3.2472      -3.0836
        -1.1501      -1.1623
         0.6855      -1.2751
         2.3432       0.2570
        -1.6892       0.3565
         1.3871       3.0138
         3.3840       2.1312
        -0.5118      -4.7670
         0.8569       2.3741
         0.9558      -1.2209
         0.6778       2.1993
         0.4304       1.1393
         1.4987      -1.2255
         0.5361       0.1237
         0.2649       2.4582
         2.0095       2.5623

    Final X(I+1:I)

         3.6698       2.5888       0.0000       0.0000       4.4040       7.9910
 Final Value of P

    2.5985
    0.5594      5.3279
    1.4809      0.9697      0.9254
    0.3627      0.2135      0.2237      0.0542
    0.0000      0.0000      0.0000      0.0000      0.0000
    0.0000      0.0000      0.0000      0.0000      0.0000      0.0000

Deviance =    0.2229E+03
```