

NAG Fortran Library Routine Document

F05AAF

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

F05AAF applies the Schmidt orthogonalisation process to n vectors in m -dimensional space, $n \leq m$.

2 Specification

```
SUBROUTINE F05AAF(A, IA, M, N1, N2, S, CC, ICOL, IFAIL)
INTEGER          IA, M, N1, N2, ICOL, IFAIL
real           A(IA,N2), S(N2), CC
```

3 Description

The routine applies the Schmidt orthogonalisation process to n linearly independent vectors in m -dimensional space, $n \leq m$. The effect of this process is to replace the original n vectors by n orthonormal vectors which have the property that the r th vector is linearly dependent on the first r of the original vectors, and that the sum of squares of the elements of the r th vector is equal to 1, for $r = 1, 2, \dots, n$. Inner-products are accumulated using *additional precision*.

4 References

None.

5 Parameters

- 1: $A(IA,N2)$ – *real* array *Input/Output*
On entry: columns N1 to N2 contain the vectors to be orthogonalised. The vectors are stored by columns in elements 1 to m .
On exit: these vectors are overwritten by the orthonormal vectors.
- 2: IA – INTEGER *Input*
On entry: the first dimension of the array A as declared in the (sub)program from which F05AAF is called.
Constraint: $IA \geq M$.
- 3: M – INTEGER *Input*
On entry: m , the number of elements in each vector.
- 4: N1 – INTEGER *Input*
 5: N2 – INTEGER *Input*
On entry: the indices of the first and last columns of A to be orthogonalised.
Constraint: $N1 \leq N2$.
- 6: S(N2) – *real* array *Workspace*

- 7: CC – *real* *Output*
On exit: CC is used to indicate linear dependence of the original vectors. The nearer CC is to 1.0, the more likely vector ICOL is dependent on vectors N1 to ICOL – 1. See Section 8.
- 8: ICOL – *INTEGER* *Output*
On exit: the column number corresponding to CC. See Section 8.
- 9: 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 (see Section 6).
 For environments where it might be inappropriate to halt program execution when an error is detected, the value –1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, for users not familiar with this parameter the recommended value is 0. **When the value –1 or 1 is used it is essential to test the value of IFAIL on exit.**

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 or warnings detected by the routine:

IFAIL = 1

On entry, $N1 > N2$.

7 Accuracy

Innerproducts are accumulated using *additional precision* arithmetic and full machine accuracy should be obtained except when $CC > 0.99999$. (See Section 8.)

8 Further Comments

The time taken by the routine is approximately proportional to nm^2 , where $n = N2 - N1 + 1$.

Parameters CC and ICOL have been included to give some indication of whether or not the vectors are nearly linearly independent, and their values should always be tested on exit from the routine. CC will be in the range [0.0, 1.0] and the closer CC is to 1.0, the more likely the vector ICOL is to be linearly dependent on vectors N1 to ICOL – 1. Theoretically, when the vectors are linearly dependent, CC should be exactly 1.0. In practice, because of rounding errors, it may be difficult to decide whether or not a value of CC close to 1.0 indicates linear dependence. As a general guide a value of $CC > 0.99999$ usually indicates linear dependence, but examples exist which give $CC > 0.99999$ for linearly independent vectors. If one of the original vectors is zero or if, possibly due to rounding errors, an exactly zero vector is produced by the Gram–Schmidt process, then CC is set exactly to 1.0 and the vector is not, of course, normalised. If more than one such vector occurs then ICOL references the last of these vectors.

A user who is concerned with testing for near linear dependence in a set of vectors may wish to consider using routine F02WEF.

9 Example

To orthonormalise columns 2, 3 and 4 of the matrix:

$$\begin{pmatrix} 1 & -2 & 3 & 1 \\ -2 & 1 & -2 & -1 \\ 3 & -2 & 1 & 5 \\ 4 & 1 & 5 & 3 \end{pmatrix}.$$

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.

```
*      F05AAF Example Program Text
*      Mark 14 Revised.  NAG Copyright 1989.
*      .. Parameters ..
INTEGER      MMAX, IA, N2MAX
PARAMETER    (MMAX=5,IA=MMAX,N2MAX=5)
INTEGER      NIN, NOUT
PARAMETER    (NIN=5,NOUT=6)
*      .. Local Scalars ..
real       CC
INTEGER      I, ICOL, IFAIL, J, M, N1, N2
*      .. Local Arrays ..
real       A(IA,N2MAX), S(N2MAX)
*      .. External Subroutines ..
EXTERNAL     F05AAF
*      .. Executable Statements ..
WRITE (NOUT,*) 'F05AAF Example Program Results'
*      Skip heading in data file
READ (NIN,*)
READ (NIN,*) M, N1, N2
WRITE (NOUT,*)
WRITE (NOUT,99999) 'N1 = ', N1, ' N2 = ', N2
IF (M.GT.0 .AND. M.LE.MMAX .AND. N2.GT.0 .AND. N2.LE.N2MAX) THEN
  READ (NIN,*) ((A(I,J),J=1,M),I=1,M)
  IFAIL = 1
*
  CALL F05AAF(A,IA,M,N1,N2,S,CC,ICOL,IFAIL)
*
  WRITE (NOUT,*)
  IF (IFAIL.NE.0) THEN
    WRITE (NOUT,99999) 'Error in F05AAF. IFAIL =', IFAIL
  ELSE
    WRITE (NOUT,99998) 'CC = ', CC, ' ICOL = ', ICOL
    WRITE (NOUT,*)
    WRITE (NOUT,*) 'Final matrix'
    WRITE (NOUT,99997) ((A(I,J),J=1,M),I=1,M)
  END IF
ELSE
  WRITE (NOUT,*) 'M or N2 is out of range'
  WRITE (NOUT,99996) 'M = ', M, ' N2 = ', N2
END IF
STOP
*
99999 FORMAT (1X,A,I2,A,I2)
99998 FORMAT (1X,A,F7.4,A,I2)
99997 FORMAT (1X,4F9.4)
99996 FORMAT (1X,A,I5,A,I5)
END
```

9.2 Program Data

F05AAF Example Program Data

```
4 2 4
1 -2 3 1
-2 1 -2 -1
3 -2 1 5
4 1 5 3
```

9.3 Program Results

F05AAF Example Program Results

N1 = 2 N2 = 4

CC = 0.5822 ICOL = 4

Final matrix

```
1.0000 -0.6325 0.3310 -0.5404
-2.0000 0.3162 -0.2483 0.2119
3.0000 -0.6325 -0.0000 0.7735
4.0000 0.3162 0.9104 0.2543
```
