NAG C Library Function Document

nag_pairs_test (g08ebc)

1 Purpose

nag_pairs_test (g08ebc) performs a pairs test on a sequence of observations in the interval $[0, 1]$.

2 Specification

```
#include <nag.h>
#include <nagg08.h>

void nag_pairs_test (Integer n, const double x[], Integer max_count, Integer lag,
      double *chi, double *df, double *prob, NagError *fail)
```

3 Description

nag_pairs_test (g08ebc) 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 $(\mathbf{x}(i),\mathbf{x}(i+1))$ such that

$$\frac{j-1}{m} \leq \mathbf{x}(i) < \frac{j}{m}$$
$$\frac{k-1}{m} \leq \mathbf{x}(i+l) < \frac{k}{m}$$

where $i = 1, 3, 5, \ldots, n-1$ if $l = 1$, and $i = 1, 2, \ldots, l, 2l+1, 2l+2, \ldots, 3l, 4l+1, \ldots, n-l$ 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:

$$X^2 = \sum_{j,k=1}^{m} \frac{\left(c_{jk} - e\right)^2}{e}$$

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.

4 References

Dagpunar J (1988) *Principles of Random Variate Generation* Oxford University Press

Knuth D E (1981) *The Art of Computer Programming (Volume 2)* (2nd Edition) Addison–Wesley

Morgan B J T (1984) *Elements of Simulation* Chapman and Hall

Ripley B D (1987) *Stochastic Simulation* Wiley

5 Arguments

1: **n** – Integer *Input*

On entry: the number of observations, n.

Constraint: $\mathbf{n} \geq 2$.

2: **x**[**n**] – const double *Input*

On entry: the sequence of observations.

Constraint: $0.0 \leq \mathbf{x}[i-1] \leq 1.0$, for $i = 1, 2, \ldots, n$.

3: **max_count** – Integer *Input*

On entry: the size of the matrix of counts, m.

Constraint: **max_count** ≥ 2.

4: **lag** – Integer *Input*

On entry: the lag, l, to be used in choosing pairs.

lag $= 1$

We consider the pairs $(\mathbf{x}[i-1], \mathbf{x}[i])$, for $i = 1, 3, \ldots, n-1$ where n is the number of observations.

lag > 1

We consider the pairs $(\mathbf{x}[i-1], \mathbf{x}[x+l-1])$, for $i = 1, 2, \ldots, l, 2l+1, 2l+2, \ldots, 3l, 4l+1, \ldots, n-l$ where n is the number of observations.

Constraint: **lag** > 0, **lag** $<$ **n**.

5: **chi** – double * *Output*

On exit: contains the χ^2 test statistic, X^2, for testing the null hypothesis of randomness.

6: **df** – double * *Output*

On exit: contains the degrees of freedom for the χ^2 statistic.

7: **prob** – double * *Output*

On exit: contains the upper tail probability associated with the χ^2 test statistic, i.e., the significance level.

8: **fail** – NagError * *Input/Output*

The NAG error parameter, see the Essential Introduction.

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_G08EB_CELL

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 test statistic.

NE_G08EB_PAIRS

No pairs were found. This will occur if the value of **lag** is greater than or equal to the total number of observations.

NE_INT_2

On entry, **lag** $= \langle value \rangle$, **n** $= \langle value \rangle$.
Constraint: $1 \leq$ **lag** $<$ **n**.

NE_INT_ARG_LE

On entry, **max_count** must not be less than or equal to 1: **max_count** $= \langle value \rangle$.

NE_INT_ARG_LT

On entry, **n** must not be less than 2: **n** $= \langle value \rangle$.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

NE_REAL_ARRAY_CONS

On entry, $\mathbf{x}[0] = \langle value \rangle$.
Constraint: $0.0 \leq \mathbf{x}[i-1] \leq 1.0$, for $i = 1, 2, \ldots, n-1$.

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 five significant figures for most cases.

8 Further Comments

The time taken by nag_pairs_test (g08ebc) increases with the number of observations, n.

9 Example

The following program performs the pairs test on 10000 pseudo-random numbers from a uniform distribution $U(0,1)$ generated by nag_random_continuous_uniform (g05cac). nag_pairs_test (g08ebc) is called with **lag** $= 1$ and $m = 10$.

9.1 Program Text

```
/* nag_pairs_test (g08ebc) Example Program.
 *
 * Copyright 2000 Numerical Algorithms Group.
 *
 * Mark 6, 2000.
 *
 * Mark 8 revised, 2004
 *
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg05.h>
#include <nagg08.h>

int main(void)
{
  Integer exit_status=0, igen = 0, iseed[] = {0, 0, 0, 0}, lag, max_count, n;
  NagError fail;
  double chi, df, enda, endb, p, *x=0;

  INIT_FAIL(fail);
  Vprintf("nag_pairs_test (g08ebc) Example Program Results\n");

  /* nag_rngs_init_repeatable (g05kbc).
   * Initialize seeds of a given generator for random number
   * generating functions (that pass seeds explicitly) to give
   * a repeatable sequence
   */
  nag_rngs_init_repeatable(&igen, iseed);
  n = 10000;
  if (!(x = NAG_ALLOC(n, double)))
    {
      Vprintf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

  enda = 0.0;
  endb = 1.0;
  /* nag_rngs_uniform (g05lgc).
   * Generates a vector of random numbers from a uniform
   * distribution, seeds and generator number passed
   * explicitly
   */
  nag_rngs_uniform(enda, endb, n, x, igen, iseed, NAGERR_DEFAULT);
  max_count = 10;
  lag = 1;
  /* nag_pairs_test (g08ebc).
   * Performs the pairs (serial) test for randomness
   */
  nag_pairs_test(n, x, max_count, lag, &chi,
                 &df, &p, &fail);

  if (fail.code != NE_NOERROR && fail.code != NE_G08EB_CELL)
    {
      Vprintf("Error from nag_pairs_test (g08ebc).\n%s\n", fail.message);
      exit_status = 1;
      goto END;

    }
  Vprintf("\n");
  Vprintf("\n");
  Vprintf("%s%10.4f\n", "CHISQ          = ", chi);
  Vprintf("%s%8.2f\n", "DF             = ", df);
  Vprintf("%s%10.4f\n", "Probability    = ", p);
  if (fail.code == NE_G08EB_CELL)
    Vprintf("Error from nag_pairs_test (g08ebc).\n%s\n", fail.message);
 END:
  if (x) NAG_FREE(x);
  return exit_status;
}
```

9.2 Program Data

None.

9.3 Program Results

```
nag_pairs_test (g08ebc) Example Program Results


CHISQ          =    99.8000
DF             =    99.00
Probability    =     0.4586
```