NAG Library Function Document

nag_triplets_test (g08ecc)

1 Purpose

nag_triplets_test (g08ecc) performs the triplets test on a sequence of observations from the interval $[0, 1]$.

2 Specification

```
#include <nag.h>
#include <nagg08.h>
void nag_triplets_test (Integer n, const double x[], Integer max_count,
     double *chi, double *df, double *prob, NagError *fail)
```

3 Description

nag_triplets_test (g08ecc) computes the statistics for performing a triplets test which may be used to investigate deviations from randomness in a sequence, $x = \{x_i : i = 1, 2, \ldots, n\}$, 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 $i = 1, 4, 7, \ldots, n-2$, such that

$$\frac{j-1}{m} \leq x_i < \frac{j}{m}$$
$$\frac{k-1}{m} \leq x_{i+1} < \frac{k}{m}$$
$$\frac{l-1}{m} \leq x_{i+2} < \frac{l}{m}.$$

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, $\sum_{j,k,l=1}^{m} c_{\mathrm{jkl}}$, divided by the number of classes, m^3.

The χ^2 test statistic used to test the hypothesis of randomness is defined as

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

where $e = \sum_{j,k,l=1}^{m} c_{\mathrm{jkl}}/m^3 =$ 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 length of the sequence relative to m increases and hence 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: n, the number of observations.

Constraint: $\mathbf{n} \geq 3$.

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: m, the size of the count matrix to be formed.

Constraint: **max_count** ≥ 2.

4: **chi** – double * *Output*

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

5: **df** – double * *Output*

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

6: **prob** – double * *Output*

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

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

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 3.2.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_G08EC_CELL

max_count is too large relative to the number of triplets, therefore the expected value for at least one 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.
max_count = ⟨*value*⟩ , number of triplets = ⟨*value*⟩ and expected value = ⟨*value*⟩.
All statistics are returned and may still be of use.

NE_G08EC_TRIPLETS

No triplets were found because less than 3 observations were provided in total.

NE_INT

On entry, **max_count** = ⟨*value*⟩.
Constraint: **max_count** ≥ 2.

NE_INT_ARG_LE

On entry, **max_count** = ⟨*value*⟩.
Constraint: **max_count** ≥ 2.

NE_INT_ARG_LT

On entry, **n** = ⟨*value*⟩.
Constraint: **n** ≥ 3.

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 contact NAG for assistance.

An unexpected error has been triggered by this function. Please contact NAG.
See Section 3.6.6 in How to Use the NAG Library and its Documentation for further information.

NE_NO_LICENCE

Your licence key may have expired or may not have been installed correctly.
See Section 3.6.5 in How to Use the NAG Library and its Documentation for further information.

NE_REAL_ARRAY_CONS

On entry, at least one element of **x** is out of range.
Constraint: $0 \leq \mathbf{x}[\mathrm{i}-1] \leq 1$, for $i = 1, 2, \ldots,$ **n**.

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 five significant figures for most cases.

8 Parallelism and Performance

nag_triplets_test (g08ecc) is not threaded in any implementation.

9 Further Comments

The time taken by the function increases with the number of observations n.

10 Example

The following program performs the triplets test on 10000 pseudorandom numbers taken from a uniform distribution $U(0, 1)$, generated by nag_rand_basic (g05sac). nag_triplets_test (g08ecc) is called with **max_count** = 5.

10.1 Program Text

```
/* nag_triplets_test (g08ecc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 *
 *
 */

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

int main(void)
{
  /* Integer scalar and array declarations */
  Integer exit_status = 0;
  Integer lstate;
  Integer *state = 0;

  /* NAG structures */
  NagError fail;

  /* Double scalar and array declarations */
  double chi, df, p;
  double *x = 0;

  /* Choose the base generator */
  Nag_BaseRNG genid = Nag_Basic;
  Integer subid = 0;

  /* Set the seed */
  Integer seed[] = { 32423 };
  Integer lseed = 1;

  /* Set the size of the (randomly generated) dataset */
  Integer n = 10000;

  /* Set the size of the count matrix */
  Integer max_count = 5;

  /* Initialize the error structure */
  INIT_FAIL(fail);

  printf("nag_triplets_test (g08ecc) Example Program Results\n");

  /* Get the length of the state array */
  lstate = -1;
  nag_rand_init_repeatable(genid, subid, seed, lseed, state, &lstate, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_rand_init_repeatable (g05kfc).\n%s\n",
           fail.message);
    exit_status = 1;
    goto END;
  }

  /* Allocate arrays */
  if (!(x = NAG_ALLOC(n, double)) || !(state = NAG_ALLOC(lstate, Integer)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

  /* Initialize the generator to a repeatable sequence */
```

```
  nag_rand_init_repeatable(genid, subid, seed, lseed, state, &lstate, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_rand_init_repeatable (g05kfc).\n%s\n",
           fail.message);
    exit_status = 1;
    goto END;
  }

  /* Generate vector of n uniform variates between 0.0 and 1.0 */
  nag_rand_basic(n, state, x, &fail);
  if (fail.code != NE_NOERROR) {
    printf("Error from nag_rand_basic (g05sac).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }

  /* nag_triplets_test (g08ecc).
   * Performs the triplets test for randomness
   */
  nag_triplets_test(n, x, max_count, &chi, &df, &p, &fail);

  /* Display the results */
  if (fail.code != NE_NOERROR && fail.code != NE_G08EC_CELL) {
    printf("Error from nag_triplets_test (g08ecc).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
  }
  printf("\n");
  printf("Chisq          = %10.4f\n", chi);
  printf("DF             = %8.2f\n", df);
  printf("Prob           = %10.4f\n", p);
  if (fail.code == NE_G08EC_CELL)
    printf("Error from nag_triplets_test (g08ecc).\n%s\n", fail.message);

END:
  NAG_FREE(x);
  NAG_FREE(state);

  return exit_status;
}
```

10.2 Program Data

None.

10.3 Program Results

```
nag_triplets_test (g08ecc) Example Program Results

Chisq          =   120.1578
DF             =   124.00
Prob           =     0.5809
```