# NAG Toolbox

# nag_nonpar_randtest_runs (g08ea)

## 1 Purpose

nag_nonpar_randtest_runs (g08ea) performs a runs up (or a runs down) test on a sequence of observations.

## 2 Syntax

```
[nruns, ncount, ex, cov, chi, df, prob, ifail] = nag_nonpar_randtest_runs(cl, x, m, nruns, ncount, 'n', n, 'maxr', maxr)
[nruns, ncount, ex, cov, chi, df, prob, ifail] = g08ea(cl, x, m, nruns, ncount, 'n', n, 'maxr', maxr)
```

## 3 Description

Runs tests may be used to investigate for trends in a sequence of observations. nag_nonpar_randtest_runs (g08ea) computes statistics for the runs up test. If the runs down test is desired then each observation must be multiplied by $-1$ before nag_nonpar_randtest_runs (g08ea) is called with the modified vector of observations. nag_nonpar_randtest_runs (g08ea) may be used in two different modes:

(i) a single call to nag_nonpar_randtest_runs (g08ea) which computes all test statistics after counting the runs;

(ii) multiple calls to nag_nonpar_randtest_runs (g08ea) 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 argument **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 $x_{k+1}$. nag_nonpar_randtest_runs (g08ea) counts the number of runs up of different lengths. Let $c_i$ denote the number of runs of length $i$, for $i = 1, 2, \ldots, r-1$. 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 nag_nonpar_randtest_runs (g08ea) (i.e., unless there is another call to nag_nonpar_randtest_runs (g08ea) to follow) in which case the unfinished run is used together with the beginning of the next sequence of numbers input to nag_nonpar_randtest_runs (g08ea) in the next call. The following is a trivial example.

Suppose we called nag_nonpar_randtest_runs (g08ea) 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 nag_nonpar_randtest_runs (g08ea) would have counted the runs up of the following lengths:

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

When the counting of runs is complete nag_nonpar_randtest_runs (g08ea) computes the expected values and covariances of the counts, $c_i$. For the details of the method used see Knuth (1981). An approximate $\chi^2$ statistic with $r$ degrees of freedom is computed, where

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

where

$c$ is the vector of counts, $c_i$, for $i = 1, 2, \ldots, r$,

$\mu_c$ is the vector of expected values,

$e_i$, for $i = 1, 2, \ldots, r$, where $e_i$ is the expected value for $c_i$ under the null hypothesis of randomness, and

$\Sigma_c$ is the covariance matrix of $c$ under the null hypothesis.

The use of the $\chi^2$-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.

You may specify the total number of runs to be found. If the specified number of runs is found before the end of a sequence nag_nonpar_randtest_runs (g08ea) 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

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 Parameters

## 5.1 Compulsory Input Parameters

1: **cl** – CHARACTER(1)

Must specify the type of call to nag_nonpar_randtest_runs (g08ea).

**cl** = 'S'
: This is the one and only call to nag_nonpar_randtest_runs (g08ea) (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'
: This is the first call to the function. 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 nag_nonpar_randtest_runs (g08ea).

**cl** = '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 nag_nonpar_randtest_runs (g08ea).

**cl** = 'L'
: This is the last call to nag_nonpar_randtest_runs (g08ea). The test statistics are computed after the final counting of runs is completed.

*Constraint*: **cl** = 'S', 'F', 'I' or 'L'.

2: **x**(**n**) – REAL (KIND=nag_wp) array

The sequence of observations.

3: **m** – INTEGER

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 nag_nonpar_randtest_runs (g08ea).

*Constraint*: if $\mathbf{m} \leq \mathbf{n}$, $\mathbf{cl} = \text{'S'}$.

4: **nruns** – INTEGER

If $\mathbf{cl} = \text{'S'}$ or 'F', **nruns** need not be set.

If $\mathbf{cl} = \text{'I'}$ or 'L', **nruns** must contain the value returned by the previous call to nag_nonpar_randtest_runs (g08ea).

5: **ncount**(**maxr**) – INTEGER array

If $\mathbf{cl} = \text{'S'}$ or 'F', **ncount** need not be set.

If $\mathbf{cl} = \text{'I'}$ or 'L', **ncount** must contain the values returned by the previous call to nag_nonpar_randtest_runs (g08ea).

## 5.2 Optional Input Parameters

1: **n** – INTEGER

*Default*: the dimension of the array **x**.

$n$, the length of the current sequence of observations.

*Constraints*:

if $\mathbf{cl} = \text{'S'}$, $\mathbf{n} \geq 3$;
otherwise $\mathbf{n} \geq 1$.

2: **maxr** – INTEGER

*Default*: the dimension of the array **ncount**.

$r$, the length of the longest run for which tabulation is desired. That is, all runs with length greater than or equal to $r$ are counted together.

**maxr** must not be changed between calls to nag_nonpar_randtest_runs (g08ea).

*Constraint*: $\mathbf{maxr} \geq 1$ and if $\mathbf{cl} = \text{'S'}$, $\mathbf{maxr} < \mathbf{n}$.

## 5.3 Output Parameters

1: **nruns** – INTEGER

The number of runs actually found.

2: **ncount**(**maxr**) – INTEGER array

The counts of runs of the different lengths, $c_i$, for $i = 1, 2, \ldots, r$.

3: **ex**(**maxr**) – REAL (KIND=nag_wp) array

If $\mathbf{cl} = \text{'S'}$ or 'L', (i.e., if it is the final exit) then **ex** contains the expected values of the counts, $e_i$, for $i = 1, 2, \ldots, r$.

Otherwise the elements of **ex** are not set.

4: **covar**($ldcov$, **maxr**) – REAL (KIND=nag_wp) array

If $\mathbf{cl} = \text{'S'}$ or 'L' (i.e., if it is the final exit) then **cov** contains the covariance matrix of the counts, $\Sigma_c$.

Otherwise the elements of **cov** are not set.

5: **chi** – REAL (KIND=nag_wp)

If **cl** = 'S' or 'L' (i.e., if it is the final exit) then **chi** contains the approximate $\chi^2$ test statistic, $X^2$.

Otherwise **chi** is not set.

6: **df** – REAL (KIND=nag_wp)

If **cl** = 'S' or 'L' (i.e., if it is the final exit) then **df** contains the degrees of freedom of the $\chi^2$ statistic.

Otherwise **df** is not set.

7: **prob** – REAL (KIND=nag_wp)

If **cl** = 'S' or 'L', (i.e., if it is the final exit) then **prob** contains the upper tail probability corresponding to the $\chi^2$ test statistic, i.e., the significance level.

Otherwise **prob** is not set.

8: **ifail** – INTEGER

**ifail** = 0 unless the function detects an error (see Section 5).

## 6 Error Indicators and Warnings

**Note**: nag_nonpar_randtest_runs (g08ea) may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected by the function:

**ifail** = 1

On entry, **cl** = ⟨*value*⟩.
Constraint: **cl** = 'S', 'F', 'I' or 'L'.

**ifail** = 2

Constraint: if **cl** = 'S', **n** ≥ 3, otherwise **n** ≥ 1.

**ifail** = 3

Constraint: if **cl** = 'S', **m** ≤ **n**.

**ifail** = 4

Constraint: if **cl** = 'S', **maxr** < **n**.

Constraint: **maxr** ≥ 1.

**ifail** = 5

Constraint: $ldcov \geq$ **maxr**.

**ifail** = 6

*lwrk***maxr** is too small.

**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 stored in **cov** is not positive definite, thus the approximate $\chi^2$ test statistic cannot be computed.
This may be because **maxr** is too large relative to the length of the full sequence.

**ifail** $= 10$ (*warning*)

The number of runs requested were not found, only ⟨*value*⟩ out of the requested ⟨*value*⟩ where found.
All statistics are returned and may still be of use.

**ifail** $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

**ifail** $= -399$

Your licence key may have expired or may not have been installed correctly.

**ifail** $= -999$

Dynamic memory allocation failed.

## 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_nonpar_randtest_runs (g08ea) increases with the number of observations $n$, and also depends to some extent on whether the call to nag_nonpar_randtest_runs (g08ea) is an only, first, intermediate or last call.

## 9 Example

The following program performs a runs up test on 500 pseudorandom numbers. nag_nonpar_randtest_runs (g08ea) is called 5 times with 100 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

```
function g08ea_example

fprintf('g08ea example results\n\n');

% Initialize the base generator to a repeatable sequence
seed = [nag_int(324213)];
genid = nag_int(1);
subid = nag_int(1);
[state, ifail] = g05kf( ...
                        genid, subid, seed);

m      = nag_int(0);
nruns  = nag_int(0);
ncount = [nag_int(0);0;0;0;0;0];
n      = nag_int(100);
nsampl = 5;
cl     = 'F';

for i=1:nsampl
  % Generate a sample from U(0,1)
  [state, x, ifail] = g05sq( ...
```

```
                            n, 0, 1, state);
  % Process the sample
  [nruns, ncount, ex, covar, chi, df, prob, ifail] = ...
  g08ea( ...
         cl, x, m, nruns, ncount);
  % Adjust CL
  cl = 'I';
  if i==nsampl-1
     cl = 'L';
  end
end

fprintf('Total number of runs found = %d\n', nruns);
fprintf('\n%33s\n', 'Count');
head = '        1        2        3        4        5       >5';
fprintf('%s\n', head);
fprintf('%9d', ncount);

fprintf('\n\n%34s\n', 'Expect');
fprintf('%s\n', head);
fprintf('%9.1f', ex);
fprintf('\n\n');

[ifail] = x04ca( ...
                'General', ' ', covar, 'Covariance matrix');

fprintf('\n\nChisq = %10.4f\n', chi);
fprintf('DF    = %7.1f\n', df);
fprintf('Prob  = %10.4f\n', prob);
```

## 9.2 Program Results

```
     g08ea example results

Total number of runs found = 251

                           Count
        1        2        3        4        5       >5
       77      120       39       12        1        2

                           Expect
        1        2        3        4        5       >5
     83.8    104.0     45.6     13.1      2.9      0.6

 Covariance matrix
             1          2          3          4          5          6
 1     64.2222    -9.8639    -7.4780    -3.5759    -1.1406    -0.3305
 2     -9.8639    70.2942   -24.4639    -9.8092    -2.7386    -0.7103
 3     -7.4780   -24.4639    29.9473    -5.8284    -1.5474    -0.3852
 4     -3.5759    -9.8092    -5.8284    11.0343    -0.5319    -0.1289
 5     -1.1406    -2.7386    -1.5474    -0.5319     2.7169    -0.0318
 6     -0.3305    -0.7103    -0.3852    -0.1289    -0.0318     0.5809

Chisq =     9.7559
DF    =     6.0
Prob  =     0.1353
```