

NAG C Library Function Document

nag_ref_vec_multi_normal (g05eac)

1 Purpose

nag_ref_vec_multi_normal (g05eac) sets up a reference vector for a multivariate Normal distribution with mean vector a and variance-covariance matrix C , so that nag_ref_vec_multi_normal (g05eac) may be used to generate pseudo-random vectors.

2 Specification

```
#include <nag.h>
#include <nagg05.h>

void nag_ref_vec_multi_normal(double a[], Integer n, double c[], Integer tdc,
    double eps, double **r, NagError *fail)
```

3 Description

When the variance-covariance matrix is non-singular (i.e., strictly positive-definite), the distribution has probability density function

$$f(x) = \sqrt{\frac{|C^{-1}|}{(2\pi)^n}} \exp\{-(x-a)^T C^{-1}(x-a)\}$$

where n is the number of dimensions, C is the variance-covariance matrix, a is the vector of means and x is the vector of positions.

Variance-covariance matrices are symmetric and positive semi-definite. Given such a matrix C , there exists a lower triangular matrix L such that $LL^T = C$. L is not unique, if C is singular.

nag_ref_vec_multi_normal decomposes C to find such an L . It then stores n , a and L in the reference vector r for later use by nag_return_multi_normal (g05ezc). nag_return_multi_normal (g05ezc) generates a vector x of independent standard Normal pseudo-random numbers. It then returns the vector $a + Lx$, which has the required multivariate Normal distribution.

It should be noted that this routine will work with a singular variance-covariance matrix C , provided C is positive semi-definite, despite the fact that the above formula for the probability density function is not valid in that case. Wilkinson (1965) should be consulted if further information is required.

4 Parameters

- | | | |
|----|--|--------------|
| 1: | a[n] – double | <i>Input</i> |
| | <i>On entry:</i> the vector of means, a , of the distribution. | |
| 2: | n – Integer | <i>Input</i> |
| | <i>On entry:</i> the number of dimensions, n , of the distribution. | |
| | <i>Constraint:</i> $n > 0$. | |
| 3: | c[n][tdc] – double | <i>Input</i> |
| | <i>On entry:</i> the variance-covariance matrix of the distribution. Only the upper triangle need be set. | |
| 4: | tdc – Integer | <i>Input</i> |
| | <i>On entry:</i> the second dimension of the array c as declared in the function from which nag_ref_vec_multi_normal is called. | |

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

5: **eps** – double *Input*

On entry: the maximum error in any element of C , relative to the largest element of C .

Constraint: $0.0 \leq \mathbf{eps} \leq 0.1/\mathbf{n}$.

6: **r** – double ** *Output*

On exit: reference vector for which memory will be allocated internally. This reference vector will subsequently be used by `nag_return_multi_normal` (g05ezc). If no memory is allocated to **r** (e.g., when an input error is detected) then **r** will be NULL on return, otherwise the user should use the NAG macro `NAG_FREE` to free the storage allocated by **r** when it is no longer of use.

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

The NAG error parameter (see the Essential Introduction).

5 Error Indicators and Warnings

NE_INT_ARG_LT

On entry, **n** must not be less than 1: $\mathbf{n} = \langle \text{value} \rangle$.

NE_2_INT_ARG_LT

On entry, $\mathbf{tdc} = \langle \text{value} \rangle$ while $\mathbf{n} = \langle \text{value} \rangle$. These parameters must satisfy $\mathbf{tdc} \geq \mathbf{n}$.

NE_REAL_ARG_LT

On entry, **eps** must not be less than 0.0: $\mathbf{eps} = \langle \text{value} \rangle$.

NE_2_REAL_ARG_GT

On entry, $\mathbf{eps} = \langle \text{value} \rangle$ while $0.1/\mathbf{n} = \langle \text{value} \rangle$. These parameters must satisfy $\mathbf{eps} \leq 0.1/\mathbf{n}$.

NE_ALLOC_FAIL

Memory allocation failed.

NE_NOT_POS_SEM_DEF

Matrix C is not positive semi-definite.

6 Further Comments

The time taken by the routine is of order n^3 .

It is recommended that the diagonal elements of C should not differ too widely in order of magnitude. This may be achieved by scaling the variables if necessary. The actual matrix decomposed is $C + E = LL^T$, where E is a diagonal matrix with small positive diagonal elements. This ensures that, even when C is singular, or nearly singular, the Cholesky Factor L corresponds to a positive-definite variance-covariance matrix that agrees with C within a tolerance determined by **eps**.

6.1 Accuracy

The maximum absolute error in LL^T , and hence in the variance-covariance matrix of the resulting vectors, is less than $(n \times \max(\mathbf{eps}, \varepsilon) + (n + 3)\varepsilon/2)$ times the maximum element of C , where ε is the *machine precision*. Under normal circumstances, the above will be small compared to sampling error.

6.2 References

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

Wilkinson J H (1965) *The Algebraic Eigenvalue Problem* Oxford University Press, London

7 See Also

nag_random_init_repeatable (g05cbc)

nag_random_init_nonrepeatable (g05ccc)

nag_random_normal (g05ddc)

nag_return_multi_normal (g05ezc)

8 Example

The example program prints five pseudo-random observations from a bivariate Normal distribution with means vector

$$\begin{bmatrix} 1.0 \\ 2.0 \end{bmatrix}$$

and variance-covariance matrix

$$\begin{bmatrix} 2.0 & 1.0 \\ 1.0 & 3.0 \end{bmatrix},$$

generated by nag_ref_vec_multi_normal and nag_return_multi_normal (g05ezc) after initialisation by nag_random_init_repeatable (g05cbc).

8.1 Program Text

```

/* nag_ref_vec_multi_normal(g05eac) Example Program
 *
 * Copyright 1991 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 *
 * Mark 3 revised, 1994.
 */

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

#define N 2
#define TDC N

main()
{
    Integer i, j;
    double a[N], c[N][TDC], z[N];
    double *r = (double *)0;
    double eps = 0.01;

    Vprintf("g05eac Example Program Results\n");
    a[0] = 1.0;
    a[1] = 2.0;
    c[0][0] = 2.0;
    c[1][1] = 3.0;

```

```
c[0][1] = 1.0;
c[1][0] = 1.0;
g05cbc((Integer)0);
g05eac(a, (Integer)N, (double *)c, (Integer)TDC,
      eps, &r, NAGERR_DEFAULT);
for (i=1; i<=5; i++)
  {
    g05ezc(z, r);
    for (j=0; j<2; j++)
      Vprintf("%10.4f",z[j]);
    Vprintf("\n");
  }
NAG_FREE(r);
exit(EXIT_SUCCESS);
}
```

8.2 Program Data

None.

8.3 Program Results

g05eac Example Program Results

1.7697	4.4481
3.2678	3.0583
3.1769	2.3651
-0.1055	1.8395
1.2933	-0.1850
