

## NAG Library Function Document

### nag\_multi\_normal (g01hbc)

#### 1 Purpose

nag\_multi\_normal (g01hbc) returns the upper tail, lower tail or central probability associated with a multivariate Normal distribution of up to ten dimensions.

#### 2 Specification

```
#include <nag.h>
#include <nagg01.h>
```

```
double nag_multi_normal (Nag_TailProbability tail, Integer n, const double a[],
    const double b[], const double mean[], const double sigma[], Integer tdsig,
    double tol, Integer maxpts, NagError *fail)
```

#### 3 Description

Let the vector random variable  $X = (X_1, X_2, \dots, X_n)^T$  follow an  $n$ -dimensional multivariate Normal distribution with mean vector  $\mu$  and  $n$  by  $n$  variance-covariance matrix  $\Sigma$ , then the probability density function,  $f(X : \mu, \Sigma)$ , is given by

$$f(X : \mu, \Sigma) = (2\pi)^{-(1/2)n} |\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(X - \mu)^T \Sigma^{-1} (X - \mu)\right).$$

The lower tail probability is defined by:

$$P(X_1 \leq b_1, \dots, X_n \leq b_n : \mu, \Sigma) = \int_{-\infty}^{b_1} \cdots \int_{-\infty}^{b_n} f(X : \mu, \Sigma) dX_n \cdots dX_1.$$

The upper tail probability is defined by:

$$P(X_1 \geq a_1, \dots, X_n \geq a_n : \mu, \Sigma) = \int_{a_1}^{\infty} \cdots \int_{a_n}^{\infty} f(X : \mu, \Sigma) dX_n \cdots dX_1.$$

The central probability is defined by:

$$P(a_1 \leq X_1 \leq b_1, \dots, a_n \leq X_n \leq b_n : \mu, \Sigma) = \int_{a_1}^{b_1} \cdots \int_{a_n}^{b_n} f(X : \mu, \Sigma) dX_n \cdots dX_1.$$

To evaluate the probability for  $n \geq 3$ , the probability density function of  $X_1, X_2, \dots, X_n$  is considered as the product of the conditional probability of  $X_1, X_2, \dots, X_{n-2}$  given  $X_{n-1}$  and  $X_n$  and the marginal bivariate Normal distribution of  $X_{n-1}$  and  $X_n$ . The bivariate Normal probability can be evaluated as described in nag\_bivariate\_normal\_dist (g01hac) and numerical integration is then used over the remaining  $n - 2$  dimensions. In the case of  $n = 3$ , nag\_1d\_quad\_gen\_1 (d01sjc) is used and for  $n > 3$  nag\_multid\_quad\_adapt\_1 (d01wcc) is used.

To evaluate the probability for  $n = 1$  a direct call to nag\_prob\_normal (g01eac) is made and for  $n = 2$  calls to nag\_bivariate\_normal\_dist (g01hac) are made.

## 4 References

Kendall M G and Stuart A (1969) *The Advanced Theory of Statistics (Volume 1)* (3rd Edition) Griffin

## 5 Arguments

- 1: **tail** – Nag\_TailProbability *Input*  
*On entry:* indicates which probability is to be returned.  
**tail** = Nag\_LowerTail  
 The lower tail probability is returned.  
**tail** = Nag\_UpperTail  
 The upper tail probability is returned.  
**tail** = Nag\_Central  
 The central probability is returned.  
*Constraint:* **tail** = Nag\_LowerTail, Nag\_UpperTail or Nag\_Central.
- 2: **n** – Integer *Input*  
*On entry:*  $n$ , the number of dimensions.  
*Constraint:*  $1 \leq \mathbf{n} \leq 10$ .
- 3: **a[n]** – const double *Input*  
*On entry:* if **tail** = Nag\_Central or Nag\_UpperTail, the lower bounds,  $a_i$ , for  $i = 1, 2, \dots, n$ .  
 If **tail** = Nag\_LowerTail, **a** is not referenced.
- 4: **b[n]** – const double *Input*  
*On entry:* if **tail** = Nag\_Central or Nag\_LowerTail, the upper bounds,  $b_i$ , for  $i = 1, 2, \dots, n$ .  
 If **tail** = Nag\_UpperTail **b**, is not referenced.  
*Constraint:* if **tail** = Nag\_Central,  $\mathbf{a}[i - 1] < \mathbf{b}[i - 1]$  for  $i = 1, 2, \dots, n$ .
- 5: **mean[n]** – const double *Input*  
*On entry:*  $\mu$ , the mean vector of the multivariate Normal distribution.
- 6: **sigma[n × tdsig]** – const double *Input*  
**Note:** the  $(i, j)$ th element of the matrix is stored in **sigma** $[(i - 1) \times \mathbf{tdsig} + j - 1]$ .  
*On entry:*  $\Sigma$ , the variance-covariance matrix of the multivariate Normal distribution. Only the lower triangle is referenced.  
*Constraint:*  $\Sigma$  must be positive-definite
- 7: **tdsig** – Integer *Input*  
*On entry:* the stride separating matrix row elements in **sigma**.  
*Constraint:* **tdsig**  $\geq \mathbf{n}$ .
- 8: **tol** – double *Input*  
*On entry:* if  $n > 2$  the relative accuracy required for the probability, and if the upper or the lower tail probability is requested then **tol** is also used to determine the cut-off points, see Section 7.  
 If  $n = 1$ , **tol** is not referenced.  
*Suggested value:* **tol** = 0.0001.  
*Constraint:* if  $\mathbf{n} > 1$ , **tol**  $> 0.0$ .

- 9: **maxpts** – Integer *Input*  
*On entry:* the maximum number of sub-intervals or integrand evaluations.  
 If  $n = 3$ , then the maximum number of sub-intervals used by nag\_1d\_quad\_gen\_1 (d01sjc) is **maxpts**/4. Note however increasing **maxpts** above 1000 will not increase the maximum number of sub-intervals above 250.  
 If  $n > 3$  the maximum number of integrand evaluations used by nag\_multid\_quad\_adapt\_1 (d01wcc) is  $\alpha(\mathbf{maxpts}/n - 1)$ , where  $\alpha = 2^{n-2} + 2(n-2)^2 + 2(n-2) + 1$ .  
 If  $n = 1$  or  $2$ , then **maxpts** will not be used.  
*Suggested value:* 2000 if  $n > 3$  and 1000 if  $n = 3$ .  
*Constraint:* if  $n \geq 3$ , **maxpts**  $\geq 4 \times n$ .
- 10: **fail** – NagError \* *Input/Output*  
 The NAG error argument (see Section 3.6 in the Essential Introduction).

## 6 Error Indicators and Warnings

### NE\_2\_INT\_ARG\_LT

On entry, **tdsig** =  $\langle value \rangle$ , **n** =  $\langle value \rangle$ .  
 Constraint: **tdsig**  $\geq n$ .

### NE\_2\_REAL\_ARRAYS\_CONS

On entry, the  $\langle value \rangle$  value in **b** is less than or equal to the corresponding value in **a**.

### NE\_ACC

Full accuracy not achieved, relative accuracy =  $\langle value \rangle$ .

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.

### NE\_BAD\_PARAM

On entry, argument  $\langle value \rangle$  had an illegal value.

### NE\_INT\_ARG\_CONS

On entry, **maxpts** =  $\langle value \rangle$ , **n** =  $\langle value \rangle$ .  
 Constraint: if  $n \geq 3$ , **maxpts**  $\geq 4 \times n$ .

On entry, **n** =  $\langle value \rangle$ .  
 Constraint:  $1 \leq n \leq 10$ .

### 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.

### NE\_POS\_DEF

On entry, **sigma** is not positive-definite.

### NE\_REAL\_ARG\_CONS

On entry, **tol** =  $\langle value \rangle$ .  
 Constraint: **tol**  $> 0.0$ .

**NE\_ROUND\_OFF**

Accuracy requested by **tol** is too strict: **tol** =  $\langle value \rangle$ .

**7 Accuracy**

The accuracy should be as specified by **tol**. When on exit **fail.code** = NE\_ACC the approximate accuracy achieved is given in the error message. For the upper and lower tail probabilities the infinite limits are approximated by cut-off points for the  $n - 2$  dimensions over which the numerical integration takes place; these cut-off points are given by  $\Phi^{-1}(\mathbf{tol}/(10 \times n))$ , where  $\Phi^{-1}$  is the inverse univariate Normal distribution function.

**8 Further Comments**

The time taken is related to the number of dimensions, the range over which the integration takes place ( $b_i - a_i$ , for  $i = 1, 2, \dots, n$ ) and the value of  $\Sigma$  as well as the accuracy required. As the numerical integration does not take place over the last two dimensions speed may be improved by arranging  $X$  so that the largest ranges of integration are for  $X_{n-1}$  and  $X_n$ .

**9 Example**

This example reads in the mean and covariance matrix for a multivariate Normal distribution and computes and prints the associated central probability.

**9.1 Program Text**

```

/* nag_multi_normal (g01hbc) Example Program.
 *
 * Copyright 2000 Numerical Algorithms Group.
 *
 * Mark 6, 2000.
 * Mark 7, revised.
 *
 */

#include <stdio.h>
#include <nag.h>
#include <nagx04.h>
#include <nag_stdlib.h>
#include <nagg01.h>

#define SIGMA(I, J) sigma[((I) - 1)*n + (J) - 1]
int main(int argc, char *argv[])
{
    FILE                *fpin, *fpout;
    Integer              exit_status = 0, i, j, maxpts, n;
    char                nag_enum_arg[40];
    double               *a = 0, *b = 0, *mean = 0, prob, *sigma = 0, tol;
    Nag_TailProbability  tail;
    NagError             fail;

    INIT_FAIL(fail);

    /* Check for command-line IO options */
    fpin = nag_example_file_io(argc, argv, "-data", NULL);
    fpout = nag_example_file_io(argc, argv, "-results", NULL);
    fprintf(fpout, "nag_multi_normal (g01hbc) Example Program Results\n");

    /* Skip heading in data file */
    fscanf(fpin, "%*[\n]");
    fscanf(fpin, "%ld %lf %s", &n, &tol, nag_enum_arg);
    /* nag_enum_name_to_value(x04nac).
     * Converts NAG enum member name to value
     */
    tail = (Nag_TailProbability) nag_enum_name_to_value(nag_enum_arg);

```

```

if (!(a = NAG_ALLOC(n, double))
    || !(b = NAG_ALLOC(n, double))
    || !(mean = NAG_ALLOC(n, double))
    || !(sigma = NAG_ALLOC(n*n, double)))
{
    fprintf(fpout, "Allocation failure\n");
    exit_status = -1;
    goto END;
}

for (j = 1; j <= n; ++j)
    fscanf(fpin, "%lf", &mean[j - 1]);
for (i = 1; i <= n; ++i)
    for (j = 1; j <= n; ++j)
        fscanf(fpin, "%lf", &SIGMA(i, j));
if (tail == Nag_Central || tail == Nag_UpperTail)
    for (j = 1; j <= n; ++j)
        fscanf(fpin, "%lf", &a[j - 1]);
if (tail == Nag_Central || tail == Nag_LowerTail)
    for (j = 1; j <= n; ++j)
        fscanf(fpin, "%lf", &b[j - 1]);
maxpts = 2000;
/* nag_multi_normal (g01hbc).
 * Computes probabilities for the multivariate Normal
 * distribution
 */
prob = nag_multi_normal(tail, n, a, b, mean, sigma, n, tol, maxpts,
                        &fail);
if (fail.code == NE_NOERROR || fail.code == NE_ACC
    || fail.code == NE_ROUND_OFF)
{
    fprintf(fpout, "\nMultivariate Normal probability = %6.4f\n", prob);
}
else
{
    fprintf(fpout, "Error from nag_multi_normal (g01hbc).\n%s\n",
            fail.message);
    exit_status = 1;
    goto END;
}
END:
if (fpin != stdin) fclose(fpin);
if (fpout != stdout) fclose(fpout);
if (a) NAG_FREE(a);
if (b) NAG_FREE(b);
if (mean) NAG_FREE(mean);
if (sigma) NAG_FREE(sigma);
return exit_status;
}

```

## 9.2 Program Data

nag\_multi\_normal (g01hbc) Example Program Data  
4 0.0001 Nag\_Central

```

0.0  0.0  0.0  0.0

1.0  0.9  0.9  0.9
0.9  1.0  0.9  0.9
0.9  0.9  1.0  0.9
0.9  0.9  0.9  1.0

-2.0 -2.0 -2.0 -2.0

2.0  2.0  2.0  2.0

```

### **9.3 Program Results**

nag\_multi\_normal (g01hbc) Example Program Results

Multivariate Normal probability = 0.9142

---