# nag_opt_lsq_covariance (e04ycc)

## 1. Purpose

**nag_opt_lsq_covariance (e04ycc)** returns estimates of elements of the variance-covariance matrix of the estimated regression coefficients for a nonlinear least-squares problem. The estimates are derived from the Jacobian of the function $f(x)$ at the solution.

The function nag_opt_lsq_covariance may be used following either of the NAG C Library nonlinear least-squares functions nag_opt_lsq_no_deriv (e04fcc) and nag_opt_lsq_deriv (e04gbc).

## 2. Specification

```
#include <nag.h>
#include <nage04.h>

void nag_opt_lsq_covariance(Integer job, Integer m, Integer n,
            double fsumsq, double cj[], Nag_E04_Opt *options, NagError *fail)
```

## 3. Description

nag_opt_lsq_covariance is intended for use when the nonlinear least-squares function, $F(x) = f^T(x)f(x)$, represents the goodness of fit of a nonlinear model to observed data. It assumes that the Hessian of $F(x)$, at the solution, can be adequately approximated by $2J^TJ$, where $J$ is the Jacobian of $f(x)$ at the solution. The estimated variance-covariance matrix $C$ is then given by

$$C = \sigma^2(J^TJ)^{-1} \qquad J^TJ \text{ non-singular,}$$

where $\sigma^2$ is the estimated variance of the residual at the solution, $\bar{x}$, given by

$$\sigma^2 = \frac{F(\bar{x})}{m-n},$$

$m$ being the number of observations and $n$ the number of variables.

The diagonal elements of $C$ are estimates of the variances of the estimated regression coefficients. See Chapter Introduction, Bard (1974) and Wolberg (1967) for further information on the use of the matrix $C$.

When $J^TJ$ is singular then $C$ is taken to be

$$C = \sigma^2(J^TJ)^{\dagger},$$

where $(J^TJ)^{\dagger}$ is the pseudo-inverse of $J^TJ$, and $\sigma^2 = \frac{F(\bar{x})}{m-k}, k = \mathrm{rank}(J)$ but in this case the parameter **fail** is returned with **fail.code** = **NW_LIN_DEPEND** as a warning to the user that $J$ has linear dependencies in its columns. The assumed rank of $J$ can be obtained from **fail.errnum**.

The function can be used to find either the diagonal elements of $C$, or the elements of the $j$th column of $C$, or the whole of $C$.

nag_opt_lsq_covariance must be preceded by one of the nonlinear least-squares functions mentioned in Section 1, and requires the parameters **fsumsq** and **options** to be supplied by those functions. **fsumsq** is the residual sum of squares $F(\bar{x})$ while the structure **options** contains the members **s** and **v** which give the singular values and right singular vectors respectively in the singular value decomposition of $J$.

## 4. Parameters

**job**

Input: indicates which elements of $C$ are returned as follows:

**job** $= -1$
The $n$ by $n$ symmetric matrix $C$ is returned.

**job** $= 0$
The diagonal elements of $C$ are returned.

**job** $> 0$
The elements of column **job** of $C$ are returned.

Constraint: $-1 \le$ **job** $\le$ **n**.

**m**

Input: the number $m$ of observations (residuals $f_i(x)$).
Constraint: **m** $\ge$ **n**.

**n**

Input: the number $n$ of variables ($x_j$).
Constraint: $1 \le$ **n** $\le$ **m**.

**fsumsq**

Input: the sum of squares of the residuals, $F(\bar{x})$, at the solution $\bar{x}$, as returned by the nonlinear least-squares routine.
Constraint: **fsumsq** $\ge 0.0$.

**cj[n]**

Output: with **job** $= 0$, **cj** returns the $n$ diagonal elements of $C$.
With **job** $= j > 0$, **cj** returns the $n$ elements of the $j$th column of $C$.
When **job** $= -1$, **cj** is not referenced.

**options**

Input/Output: the structure used in the call to the nonlinear least-squares routine. The following members are relevant to nag_opt_lsq_covariance, their values should not be altered between the call to the least-squares routine and the call to nag_opt_lsq_covariance.

**s** – double *
Input: pointer to the $n$ singular values of the Jacobian as returned by the nonlinear least-squares routine.

**v** – double *
Input: pointer to the $n$ by $n$ right-hand orthogonal matrix (the right singular vectors) of $J$ as returned by the nonlinear least-squares routine.
Output: when **job** $\ge 0$ then **v** is unchanged.
When **job** $= -1$ then the leading $n$ by $n$ part of **v** is overwritten by the $n$ by $n$ matrix $C$. Matrix element $i, j$ is held in **options.v**$[(i-1)$***options.tdv** $+ j-1]$ for $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, n$.

**tdv** – Integer
Input: the trailing dimension used by **options.v**.

**fail**

The NAG error parameter, see the Essential Introduction to the NAG C Library.
Users are recommended to declare and initialise **fail** and set **fail.print** = **TRUE** for this function. See Sections 3 and 5 and the discussion of the warning exit **NW_LIN_DEPEND**.

## 5. Error Indications and Warnings

**NE_INT_ARG_LT**

On entry, **n** must not be less than 1: **n** $= \langle$*value*$\rangle$.
On entry, **job** must not be less than $-1$: **job** $= \langle$*value*$\rangle$.

**NE_2_INT_ARG_LT**

On entry, **m** $= \langle$*value*$\rangle$ while **n** $= \langle$*value*$\rangle$. These parameters must satisfy **m** $\ge$ **n**.

**NE_2_INT_ARG_GT**

On entry, **job** = ⟨*value*⟩ while **n** = ⟨*value*⟩. These parameters must satisfy **job** ≤ **n**.

**NE_REAL_ARG_LT**

On entry, **fsumq** must not be less than 0.0: **fsumsq** = ⟨*value*⟩.

**NE_SINGULAR_VALUES**

The singular values are all zero, so that at the solution the Jacobian matrix has rank 0.

**NW_LIN_DEPEND**

At the solution the Jacobian matrix contains linear, or near linear, dependencies amongst its columns. J assumed to have rank ⟨*value*⟩.

In this case the required elements of $C$ have still been computed based upon $J$ having an assumed rank given by **fail.errnum**. The rank is computed by regarding singular values **options.sv**$[j]$ that are not larger than $10\epsilon \times$ **options.sv**$[0]$ as zero, where $\epsilon$ is the ***machine precision*** (see nag_machine_precision (X02AJC)). Users who expect near linear dependencies at the solution and are happy with this tolerance in determining rank should not call nag_opt_lsq_covariance with the null pointer `NAGERR_DEFAULT` as the argument **fail** but should specifically declare and initialise a NagError structure for the parameter **fail**.

**Overflow**

If overflow occurs then either an element of $C$ is very large, or the singular values or singular vectors have been incorrectly supplied.

## 6. Further Comments

When **job** $= -1$ the time taken by the function is approximately proportional to $n^3$. When **job** $\geq 0$ the time taken by the function is approximately proportional to $n^2$.

### 6.1. Accuracy

The computed elements of $C$ will be the exact covariances corresponding to a closely neighbouring Jacobian matrix $J$.

### 6.2. References

Bard Y (1974) *Nonlinear Parameter Estimation* Academic Press, London.
Wolberg J R (1967) *Prediction Analysis* Van Nostrand, New York.

## 7. See Also

nag_opt_lsq_no_deriv (e04fcc)
nag_opt_lsq_deriv (e04gbc)
nag_opt_init (e04xxc)
nag_opt_free (e04xzc)

## 8. Example

To estimate the variance-covariance matrix $C$ for the least-squares estimates of $x_1$, $x_2$ and $x_3$ in the model

$$y = x_1 + \frac{t_1}{x_2 t_2 + x_3 t_3}$$

using the 15 sets of data given in the following table:

| $y$ | $t_1$ | $t_2$ | $t_3$ |
|---|---|---|---|
| 0.14 | 1.0 | 15.0 | 1.0 |
| 0.18 | 2.0 | 14.0 | 2.0 |
| 0.22 | 3.0 | 13.0 | 3.0 |
| 0.25 | 4.0 | 12.0 | 4.0 |
| 0.29 | 5.0 | 11.0 | 5.0 |
| 0.32 | 6.0 | 10.0 | 6.0 |
| 0.35 | 7.0 | 9.0 | 7.0 |
| 0.39 | 8.0 | 8.0 | 8.0 |
| 0.37 | 9.0 | 7.0 | 7.0 |
| 0.58 | 10.0 | 6.0 | 6.0 |
| 0.73 | 11.0 | 5.0 | 5.0 |
| 0.96 | 12.0 | 4.0 | 4.0 |
| 1.34 | 13.0 | 3.0 | 3.0 |
| 2.10 | 14.0 | 2.0 | 2.0 |
| 4.39 | 15.0 | 1.0 | 1.0 |

The program uses (0.5,1.0,1.5) as the initial guess at the position of the minimum and computes the least-squares solution using nag_opt_lsq_no_deriv (e04fcc). Note that the structure **options** is initialised by nag_opt_init (e04xxc) before calling nag_opt_lsq_no_deriv (e04fcc). See the function documents for nag_opt_lsq_no_deriv (e04fcc), nag_opt_init (e04xxc) and nag_opt_free (e04xzc) for further information.

## 8.1. Program Text

```
/* nag_opt_lsq_covariance (e04ycc) Example Program
 *
 * Copyright 1991 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 */

#include <nag.h>
#include <stdio.h>
#include <math.h>
#include <nag_stdlib.h>
#include <nage04.h>

#ifdef NAG_PROTO
static void lsqfun(Integer m, Integer n, double x[], double fvec[],
                   Nag_Comm *comm);
#else
static void lsqfun();
#endif

#define MMAX 15
#define NMAX 3
#define TMAX 3

/* Define a user structure template to store data in lsqfun */
struct user
{
  double y[MMAX];
  double t[MMAX][TMAX];
};

main()
{
  double fjac[MMAX][NMAX], fvec[MMAX], x[NMAX], cj[NMAX];
  Integer i, j, m, n, nt, tdj, job;
  double fsumsq;
  Nag_E04_Opt options;
  Nag_Comm comm;
  static NagError fail, fail2;
  struct user s;
```

```
  Vprintf("e04ycc Example Program Results.\n");
  Vscanf(" %*[^\n]"); /* Skip heading in data file */
  n = 3;
  m = 15;
  tdj = NMAX;
  nt = 3;

  /* Read data into structure.
   * Observations t (j = 0, 1, 2) are held in s->t[i][j]
   * (i = 0, 1, 2, . . . , 14)
   */
  nt = 3;
  for (i = 0; i < m; ++i)
    {
      Vscanf("%lf", &s.y[i]);
      for (j = 0; j < nt; ++j) Vscanf("%lf", &s.t[i][j]);
    }

  /* Set up the starting point */
  x[0] = 0.5;
  x[1] = 1.0;
  x[2] = 1.5;

  e04xxc(&options); /* Initialise options structure */

  /* Assign address of user defined structure to
   * comm.p for communication to lsqfun().
   */
  comm.p = (Pointer)&s;

  fail.print = TRUE;
  e04fcc(m, n, lsqfun, x, &fsumsq, fvec, (double *)fjac, tdj,
         &options, &comm, &fail);

  if (fail.code == NE_NOERROR || fail.code == NW_COND_MIN)
    {
      job = 0;
      e04ycc(job, m, n, fsumsq, cj, &options, &fail);

      if (fail.code == NE_NOERROR)
        {
          Vprintf("\nEstimates of the variances of the sample regression");
          Vprintf(" coefficients are:\n");
          for (i = 0; i < n; ++i)
            Vprintf(" %15.5e", cj[i]);
          Vprintf("\n");
        }
    }

  /* Free memory allocated to pointers s and v */
  fail2.print = TRUE;
  e04xzc(&options, "all", &fail2);
  if (fail.code != NE_NOERROR || fail2.code != NE_NOERROR) exit(EXIT_FAILURE);
  exit(EXIT_SUCCESS);
}                          /* main */

#ifdef NAG_PROTO
static void lsqfun(Integer m, Integer n, double x[], double fvec[],
                   Nag_Comm *comm)
#else
     static void lsqfun(m, n, x, fvec, comm)
     Integer m, n;
     double x[], fvec[];
     Nag_Comm *comm;
#endif
{
  /* Function to evaluate the residuals.
   *
   * The address of the user defined structure is recovered in each call
```

```
   * to lsqfun() from comm->p and the structure used in the calculation
   * of the residuals.
   */

  Integer i;
  struct user *s = (struct user *)comm->p;

  for (i = 0; i < m; ++i)
    fvec[i] = x[0] + s->t[i][0] / (x[1]*s->t[i][1] + x[2]*s->t[i][2]) - s->y[i];
}                                /* lsqfun */
```

### 8.2. Program Data

```
e04ycc Example Program Data
 0.14  1.0 15.0  1.0
 0.18  2.0 14.0  2.0
 0.22  3.0 13.0  3.0
 0.25  4.0 12.0  4.0
 0.29  5.0 11.0  5.0
 0.32  6.0 10.0  6.0
 0.35  7.0  9.0  7.0
 0.39  8.0  8.0  8.0
 0.37  9.0  7.0  7.0
 0.58 10.0  6.0  6.0
 0.73 11.0  5.0  5.0
 0.96 12.0  4.0  4.0
 1.34 13.0  3.0  3.0
 2.10 14.0  2.0  2.0
 4.39 15.0  1.0  1.0
```

### 8.3. Program Results

```
e04ycc Example Program Results.

Parameters to e04fcc
--------------------

Number of residuals........... 15    Number of variables...........  3

optim_tol............... 1.05e-08    linesearch_tol.......... 5.00e-01
step_max................ 1.00e+05    max_iter................       50
print_level.........Nag_Soln_Iter    machine precision....... 1.11e-16
outfile.................   stdout

Memory allocation:
s.......................      Nag
v.......................      Nag    tdv.....................       3

Results from e04fcc:
-------------------

Iteration results:

  Itn  Nfun   Objective   Norm g   Norm x   Norm (x(k-1)-x(k))   Step
   0     4    1.0210e+01  3.2e+01  1.9e+00
   1     8    1.9873e-01  2.8e+00  2.4e+00       7.2e-01        1.0e+00
   2    12    9.2324e-03  1.9e-01  2.6e+00       2.5e-01        1.0e+00
   3    16    8.2149e-03  1.2e-03  2.6e+00       2.7e-02        1.0e+00
   4    25    8.2149e-03  1.2e-07  2.6e+00       3.8e-04        1.0e+00
   5    31    8.2149e-03  1.7e-10  2.6e+00       4.2e-06        1.0e+00

Final solution:

       x              g            Residuals
  8.24106e-02    -6.1762e-12      -5.8811e-03
  1.13304e+00     1.4264e-10      -2.6535e-04
  2.34370e+00     9.4150e-11       2.7469e-04
                                   6.5415e-03
                                  -8.2299e-04
                                  -1.2995e-03
                                  -4.4631e-03
```

```
                                    -1.9963e-02
                                     8.2216e-02
                                    -1.8212e-02
                                    -1.4811e-02
                                    -1.4710e-02
                                    -1.1208e-02
                                    -4.2040e-03
                                     6.8079e-03

The sum of squares is  8.2149e-03.

Estimates of the variances of the sample regression coefficients are:
     1.53120e-04     9.48024e-02     8.77806e-02
```