Module 25.1: nag_lin_reg
Regression Analysis

`nag_lin_reg` contains procedures that perform a simple or multiple linear regression analysis.

Contents

Introduction

This module provides procedures for performing linear regression analysis and calculating various parameter estimates associated with a regression model. It contains the following two procedures.

- `nag_simple_lin_reg` performs a simple linear regression analysis for a pair of related variables (Y and X) and returns parameter estimates associated with the simple model $y = \alpha + \beta x + e$.
- `nag_mult_lin_reg` performs a general multiple regression analysis and returns residuals, leverages, residual sum of squares (and its associated degrees of freedom), estimates of regression coefficients (and the regression constant if it is included in the model) and their standard errors. A facility is provided that allows only a subset of the predictor variables to be included in the model. Weighting of observations on the response (or dependent) variable is also allowed.

Procedure: nag_simple_lin_reg

1 Description

`nag_simple_lin_reg` returns various parameter estimates that are associated with a simple linear regression analysis of a set of paired data on variables X and Y. Y is called the response (or dependent) variable and X is the predictor (or independent) variable. The optional arguments `x_miss` and `y_miss` (which represent missing values on X and Y respectively) may be used to exclude paired data with missing value(s) from the regression analysis.

2 Usage

```
USE nag_lin_reg
CALL nag_simple_lin_reg(x, y, result  [, optional arguments])
```

3 Arguments

Note. All array arguments are assumed-shape arrays. The extent in each dimension must be exactly that required by the problem. Notation such as '`x`(n)' is used in the argument descriptions to specify that the array `x` must have exactly n elements.

This procedure derives the value of the following problem parameter from the shape of the supplied arrays.

$n \geq 2$ — the number of paired data observed on X and Y

3.1 Mandatory Arguments

x(n) — real(kind=wp), intent(in)

Input: the values of predictor variable.

y(n) — real(kind=wp), intent(in)

Input: the values of response variable.

result(21) — real(kind=wp), intent(out)

Output: the various statistics associated with the regression model.

`result(1)` = $\bar{x}$, the mean of `x`;
`result(2)` = $\bar{y}$, the mean of `y`;
`result(3)` = s_x, the standard deviation of `x`;
`result(4)` = s_y, the standard deviation of `y`;
`result(5)` = r, the Pearson product-moment correlation coefficient;
`result(6)` = $\hat{\beta}$, the regression coefficient (or slope);
`result(7)` = $\hat{\alpha}$, the regression constant (or intercept);
`result(8)` = $se(\hat{\beta})$, the standard error for $\hat{\beta}$;
`result(9)` = $se(\hat{\alpha})$, the standard error for $\hat{\alpha}$;
`result(10)` = $t(\hat{\beta})$, the t-value for $\hat{\beta}$;
`result(11)` = $t(\hat{\alpha})$, the t-value for $\hat{\alpha}$;
`result(12)` = SSR, the sum of squares associated with the regression line;
`result(13)` = DFR, the degrees of freedom associated with the regression line;
`result(14)` = MSR, the mean square associated with the regression;
`result(15)` = F, the F-value for the analysis of variance;

`result(16)` = SSD, the sum of squares of deviations about the regression line (i.e., the error);

`result(17)` = DFD, the degrees of freedom of deviations for the error;

`result(18)` = MSD, the mean square of deviations about the regression line;

`result(19)` = SST, the total sum of squares;

`result(20)` = DFT, the total degrees of freedom;

`result(21)` = m, the actual sample size used for the regression analysis after deleting all paired data with missing value, if any exist.

Note: any paired data with missing value is excluded from the analysis.

3.2 Optional Arguments

Note. Optional arguments must be supplied by keyword, not by position. The order in which they are described below may differ from the order in which they occur in the argument list.

add_alpha — logical, intent(in), optional

Input: indicates whether or not to include a constant term (α) in the regression model.

If `add_alpha` = `.true.`, the model includes the regression constant or intercept;

if `add_alpha` = `.false.`, the model does not include an intercept.

Default: `add_alpha` = `.true.`.

Note: if `add_alpha` = `.false.`, then `result(7)` = `result(9)` = `result(11)` = 0.0.

x_miss — real(kind=*wp*), intent(in), optional
y_miss — real(kind=*wp*), intent(in), optional

Input: the data values which are used to identify missing value(s) on variables `x` and `y` respectively.

Note: if `x_miss` or `y_miss` or both are supplied, any pair (`x`(i), `y`(i)) with `x`(i) = `x_miss` or `y`(i) = `y_miss` is not used in the analysis.

error — type(nag_error), intent(inout), optional

The NAG *fl*90 error-handling argument. See the Essential Introduction, or the module document `nag_error_handling` (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to `nag_set_error` before this procedure is called.

4 Error Codes

Fatal errors (error%level = 3):

error%code	Description
301	An input argument has an invalid value.
302	An array argument has an invalid shape.
303	Array arguments have inconsistent shapes.
320	The procedure was unable to allocate enough memory.

Warnings (error%level = 1):

error%code	Description
101	The results are meaningless. This is because all the data on `x`, `y` or both are identical.

5 Examples of Usage

A complete example of the use of this procedure appears in Example 1 of this module document.

6 Further Comments

6.1 Mathematical Background

Let $(x_i, y_i),\ i = 1, \ldots, n,$ be n paired data which are measured on variables X and Y. If any paired data has a missing value, either for x or for y, it is deleted and the remaining m completely observed paired data are used for the regression analysis. In this case, m takes the role of n in any of the expressions stated hereunder. The linear relationship between X and Y can be expressed as:

$$y_i = \alpha + \beta x_i + e_i, \quad i = 1, 2, \ldots, n \ (n > 2)$$

where α is the intercept (or regression constant) and β is the slope (or regression coefficient).

Estimates of α and β are usually calculated by minimizing

$$\sum_{i=1}^{n} e_i^2.$$

This yields the following.

(a) The regression coefficient (or slope) is

$$\hat{\beta} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}.$$

(b) The intercept or regression constant is

$$\hat{\alpha} = \bar{y} - \hat{\beta}\bar{x}$$

where

$$\bar{x} = \frac{1}{n}\sum_{i}^{n} x_i \ \text{ and } \ \bar{y} = \frac{1}{n}\sum_{i}^{n} y_i.$$

The procedure also calculates the following statistics.

(c) The standard deviations for `x` and `y` are

$$s_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \ \text{ and } \ s_y = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2}.$$

(d) The Pearson product-moment correlation coefficient is

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}.$$

(e) The sum of squares associated with the regression, SSR,the sum of squares of deviations about the regression, SSD,and the total sum of squares, SST,are

$$\text{SST} = \sum_{i=1}^{n}(y_i - \bar{y})^2, \qquad \text{SSD} = \sum_{i=1}^{n}(y_i - \hat{\alpha} - \hat{\beta}x_i)^2, \ \text{ and } \ \text{SSR} = \text{SST} - \text{SSD}.$$

(f) The degrees of freedom associated with the regression, DFR,the degrees of freedom of deviations about the regression, DFD,and the total degrees of freedom, DFT, are

$$\text{DFT} = n - 1, \qquad \text{DFD} = n - 2, \ \text{ and } \ \text{DFR} = 1.$$

(g) The mean square associated with the regression, MSR, and the mean square of deviations about the regression, MSD, are

$$\mathrm{MSR} = \mathrm{SSR}/\mathrm{DFR} \quad \text{and} \quad \mathrm{MSD} = \mathrm{SSD}/\mathrm{DFD}.$$

(h) The F-value for the analysis of variance is

$$\mathrm{F} = \mathrm{MSR}/\mathrm{MSD}.$$

(i) The standard errors for the regression coefficient, $se(\hat{\beta})$, and the regression constant, $se(\hat{\alpha})$, are

$$se(\hat{\beta}) = \sqrt{\frac{\mathrm{MSD}}{\sum_{i=1}^{n}(x_i - \bar{x})^2}} \quad \text{and} \quad se(\hat{\alpha}) = \sqrt{\mathrm{MSD}\left(\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}\right)}.$$

(j) The t-values for the regression coefficient (or slope), $t(\hat{\beta})$, and the regression constant (or intercept), $t(\hat{\alpha})$ are

$$t(\hat{\beta}) = \frac{\hat{\beta}}{se(\hat{\beta})} \quad \text{and} \quad t(\hat{\alpha}) = \frac{\hat{\alpha}}{se(\hat{\alpha})}.$$

When the model does not include an intercept, i.e., $\alpha = 0$, the regression coefficient and its standard error are

$$\hat{\beta} = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} {x_i}^2} \quad \text{and} \quad se(\hat{\beta}) = \sqrt{\frac{\mathrm{MSD}}{\sum_{\mathrm{i}=1}^{\mathrm{n}} {\mathrm{x_i}}^2}},$$

with the degrees of freedom of deviations about the regression DFD $= n - 1$.

6.2 Accuracy

There may be a loss of significant figures for large n.

6.3 Timing

The time taken by the procedure depends on n.

Procedure: nag_mult_lin_reg

1 Description

`nag_mult_lin_reg` returns various parameter estimates that are associated with a multiple linear regression analysis. Data on a response variable (Y) and a set of predictor variables ($X_1, \ldots, X_m$) must be supplied. If the data on Y are correlated or do not have equal variances, this procedure provides a facility for using weighted least-squares in the occurrence of either or both phenomena by including `wt` in the argument list and assigning appropriate weights to it. Also, if missing values are present on any rows of X or Y or both, the corresponding rows of `wt` must be assigned zero. In this case, weighted regression analysis is performed.

Since all predictor variables may not be relevant to the aims of a regression analysis, the procedure allows a subset of predictor variables to be selected and included in the multiple regression model; this is done by using the optional argument `var_in_model`.

2 Usage

```
USE nag_lin_reg
CALL nag_mult_lin_reg(x, y, beta  [, optional arguments])
```

3 Arguments

Note. All array arguments are assumed-shape arrays. The extent in each dimension must be exactly that required by the problem. Notation such as '`x`(n)' is used in the argument descriptions to specify that the array `x` must have exactly n elements.

This procedure derives the values of the following problem parameters from the shape of the supplied arrays.

$n > 1$ — the number of observations

$m \geq 1$ — the number of predictor variables

3.1 Mandatory Arguments

x(n, m) — real(kind=*wp*), intent(in)

Input: the values of predictor variables.

y(n) — real(kind=*wp*), intent(in)

Input: the values of response variable.

beta($m + 1$) — real(kind=*wp*), intent(out)

Output: the estimates of the parameters of the regression model.

`beta`(i) for any i, such that $1 \leq i \leq m$, contains the estimate of the regression coefficient associated with the predictor variable `x`(i), if `x`(i) is included in the model. But if `x`(i) is excluded from the model (i.e., if `var_in_model`(i) = `.false.`), `beta`(i) is set to zero.

`beta`($m+1$) contains the estimate of the regression constant, if the model includes the constant term. If the model does not include a regression constant (i.e., if `add_alpha` = `.false.`), `beta`($m+1$) is set to zero.

3.2 Optional Arguments

Note. Optional arguments must be supplied by keyword, not by position. The order in which they are described below may differ from the order in which they occur in the argument list.

var_in_model(m) — logical, intent(in), optional

Input: the predictor variables to be included in the model.

If `var_in_model`(i) = `.true.`, the ith predictor variable is *included* in the model;

if `var_in_model`(i) = `.false.`, the ith predictor variable is *excluded* from the model.

Default: all predictor variables are included in the model.

wt(n) — real(kind=wp), intent(in), optional

Input: the weights that are associated with the data values on `y`.

Note: if `wt` is present, a weighted regression analysis is performed. However, if `wt`(i) is zero, the ith vector of data values on `x`'s and `y` is excluded from the weighted regression analysis.

Default: `wt` = 1.0.

Constraints: `wt` ≥ 0.

df — integer, intent(out), optional

Output: the degrees of freedom that is associated with the residual sum of squares.

add_alpha — logical, intent(in), optional

Input: indicates whether or not to include a constant term in the regression model.

If `add_alpha` = `.true.`, the model includes the regression constant term;

if `add_alpha` = `.false.`, the model does not include the constant term.

Default: `add_alpha` = `.false.`.

std_err($m+1$) — real(kind=wp), intent(out), optional

Output: the standard errors of the parameters of the regression model.

`std_err`(i) for any i, such that $1 \leq i \leq m$, contains the standard error of the regression coefficient which is associated with the predictor variable `x`(i), if `x`(i) is included in the model. If `x`(i) is excluded from the model (i.e., if `var_in_model`(i) = `.false.`), `std_err`(i) is set to zero.

`std_err`($m+1$) contains the standard error of the regression constant, if the model includes the constant term. If the model does not include a regression constant (i.e., if `add_alpha` = `.false.`), `std_err`($m+1$) is set to zero.

cov($((m+1)(m+2))/2$) — real(kind=wp), intent(out), optional

Output: the upper triangular part of the variance-covariance matrix of the parameters of the regression model in packed storage.

`cov`($i+(j(j-1))/2$) for any (i,j), such that $1 \leq i \leq j \leq m$, contains the variance-covariance between the parameters associated with the predictor variables `x`(i) and `x`(j), if the two variables are included in the model. If either `x`(i) or `x`(j) is excluded from the model (i.e., if `var_in_model`(i) = `.false.` or `var_in_model`(j) = `.false.`), `cov`($i+(j(j-1))/2$) is set to zero.

`cov`($i+(m(m+1))/2$) for any i, such that $1 \leq i \leq m$, contains the covariance between the parameter associated with the predictor variable `x`(i) and the regression constant, if the variable `x`(i) and the regression constant are included in the model. If `x`(i) or the constant term is excluded from the model (i.e., if `var_in_model`(i) = `.false.` or `add_alpha` = `.false.`), `cov`($i+(m(m+1))/2$) is set to zero.

`cov`($((m+2)(m+1))/2$) contains the variance of the regression constant, if it is included in the model. If the model does not include the constant term (i.e., if `add_alpha` = `.false.`), `cov`($((m+2)(m+1))/2$) is set to zero.

resid(*n*) — real(kind=*wp*), intent(out), optional

Output: the residuals from the regression model.

Note: if `wt` is present, `resid` contains the weighted residuals.

resid_sum_sq — real(kind=*wp*), intent(out), optional

Output: the sum of squares of the residuals from the regression model.

lev(*n*) — real(kind=*wp*), intent(out), optional

Output: the measure of the leverage of the ith vector of the values of predictor variables on the fitted regression model.

rank — integer, intent(out), optional

Output: the rank of the predictor variables.

Note: if the design matrix X is of full rank, a QR decomposition method is used; otherwise, a singular value decomposition method is used instead.

tol — real(kind=*wp*), intent(in), optional

Input: the value of `tol` is used to decide if the independent variables are of full rank and if not what is the rank of the independent variables. The smaller the value of `tol` the stricter the criterion for selecting the singular value decomposition.

Default: `tol` $= 10^{-6}$.

Constraints: `tol` > 0.0.

error — type(nag_error), intent(inout), optional

The NAG *fl*90 error-handling argument. See the Essential Introduction, or the module document `nag_error_handling` (1.2). You are recommended to omit this argument if you are unsure how to use it. If this argument is supplied, it *must* be initialized by a call to `nag_set_error` before this procedure is called.

4 Error Codes

Fatal errors (error%level = 3):

error%code	Description
301	An input argument has an invalid value.
302	An array argument has an invalid shape.
303	Array arguments have inconsistent shapes.
304	Invalid presence of an optional argument.
320	The procedure was unable to allocate enough memory.

Failures (error%level = 2):

error%code	Description
201	Convergence failure. The singular value decomposition used for the calculation of parameter estimates has failed to converge. However, this is unlikely to occur in most cases.

Warnings (error%level = 1):

error%code	Description
101	The degrees of freedom for the residuals is zero. This is because the number of parameters in the model is equal to the number of effective observations on the response variable.

5 Examples of Usage

A complete example of the use of this procedure appears in Example 2 of this module document.

6 Further Comments

6.1 Mathematical Background

The general linear regression model is defined by

$$y = X\beta + \varepsilon$$

where

y is a response (or dependent) variable with n observations;

X is a matrix of m predictor variables measured on n individuals, also called the $n \times m$ design matrix, of rank p;

β is a vector of unknown regression parameters of length m; and

ε is a vector of unknown random errors of length n where $\mathrm{var}(\varepsilon) = V\sigma^2$ and V is a known diagonal matrix.

6.2 Algorithmic Detail

If $V = I$ (identity matrix), a least-squares estimation method is used. If $V \neq I$, for any given weighted matrix $W \propto V^{-1}$, a weighted least-squares estimation method is used.

The least-squares estimates $\hat{\beta}$ of the regression parameters β minimize $(y - X\hat{\beta})^T(y - X\hat{\beta})$ and the weighted least-squares estimates minimize $(y - X\hat{\beta})^T W(y - X\hat{\beta})$.

The QR decomposition of the design matrix X (or $W^{1/2}X$ in the weighted case) is as follows:

$$X = QR^* \quad (\text{or} \quad W^{1/2}X = QR^*),$$

where $R^* = \begin{pmatrix} R \\ 0 \end{pmatrix}$ and R is a $m \times m$ upper triangular matrix and Q is an $n \times n$ orthogonal matrix. If R is of full rank, then $\hat{\beta}$ is the solution to

$$R\hat{\beta} = c_1,$$

where $c = Q^T y$ (or $Q^T W^{1/2} y$) and c_1 is the first m elements of c. If R is not of full rank a solution is obtained by means of a singular value decomposition (SVD) of R,

$$R = Q_* \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} P^T,$$

where D is a $p \times p$ diagonal matrix with non-zero diagonal elements, p being the rank of R and Q_* and P are $m \times m$ orthogonal matrices. This gives the solution

$$\hat{\beta} = P_1 D^{-1} Q_{*_1}^T c_1,$$

P_1 being the first p columns of P, i.e., $P = (P_1 P_0)$ and Q_{*_1} being the first p columns of Q_*.

The residuals from the regression model are $r_i = y_i - \hat{y}_i$, where $\hat{y}_i = X\hat{\beta}$ are the fitted values which can also be written as Hy. H is symmetric and is called the *hat* matrix of dimension $n \times n$. The ith diagonal element of H, h_i $(0 \leq h_i \leq 1)$, is a measure of the leverage of the ith values of the predictor variables (i.e., $x_{i1}, \ldots, x_{im}$) on the fitted regression model; when h_i is near unity, $\hat{y}_i \approx y_i$.

The regression model may include a constant usually known as intercept or mean term. When this constant term is specified, the number of parameters associated with the model is increased by one, so that in addition to estimating the regression coefficients, the constant term is also estimated. This is taken care of in this procedure. Also, a subset of the predictor variables may be of interest to be included in the model, a facility to do this is provided. Various parameter estimates which include regression coefficients (and constant term, if specified), their standard errors, variance-covariance matrix and several other estimates may be returned.

Example 1: Setting up an ANOVA table for a simple linear regression analysis

This example program shows how `nag_mult_lin_reg` returns various estimates for an ANOVA table for a simple linear regression analysis of data with missing values. It also shows other parameter estimates associated with the analysis.

1 Program Text

Note. The listing of the example program presented below is double precision. Single precision users are referred to Section 5.2 of the Essential Introduction for further information.

```
PROGRAM nag_lin_reg_ex01

  ! Example Program Text for nag_lin_reg
  ! NAG fl90, Release 3. NAG Copyright 1997.

  ! .. Use Statements ..
  USE nag_examples_io, ONLY : nag_std_out, nag_std_in
  USE nag_lin_reg, ONLY : nag_simple_lin_reg
  ! .. Implicit None Statement ..
  IMPLICIT NONE
  ! .. Intrinsic Functions ..
  INTRINSIC KIND
  ! .. Parameters ..
  INTEGER, PARAMETER :: wp = KIND(1.0D0)
  REAL (wp), PARAMETER :: x_miss = 0.0_wp
  REAL (wp), PARAMETER :: y_miss = 99.0_wp
  ! .. Local Scalars ..
  INTEGER :: i, n
  ! .. Local Arrays ..
  REAL (wp) :: result(21)
  REAL (wp), ALLOCATABLE :: x(:), y(:)
  ! .. Executable Statements ..

  WRITE (nag_std_out,*) 'Example Program Results for nag_lin_reg_ex01'

  READ (nag_std_in,*)          ! Skip heading in data file
  READ (nag_std_in,*) n

  ALLOCATE (x(n),y(n))         ! Allocate storage

  DO i = 1, n
    READ (nag_std_in,*) x(i), y(i)
  END DO

  CALL nag_simple_lin_reg(x,y,result,x_miss=x_miss,y_miss=y_miss)

  WRITE (nag_std_out,*)
  WRITE (nag_std_out,*) &
   '             ***** Descriptive Statistics *****           '
  WRITE (nag_std_out,*)
  WRITE (nag_std_out,*) 'Variable             Mean        Std-deviation'
  WRITE (nag_std_out,*)
  WRITE (nag_std_out,'(1x,a,5x,f8.4,7x,f10.4)') 'response (Y) ', &
   result(2), result(4), 'predictor (X)', result(1), result(3)
  WRITE (nag_std_out,*)
  WRITE (nag_std_out,*)
  WRITE (nag_std_out,*) &
   '            ***** Analysis of variance (ANOVA) table *****       '
  WRITE (nag_std_out,*)
  WRITE (nag_std_out,*) 'Source of variation      D.F      &
```

```
       &Sum of squares      Mean square     F-value'
      WRITE (nag_std_out,*)
      WRITE (nag_std_out,'(1x,a,15x,f3.0,6x,f14.4,6x,f11.4,4x,f8.4)') &
       'Regression', result(13), result(12), result(14:15), 'Error     ', &
       result(17), result(16), result(18)
      WRITE (nag_std_out,*)
      WRITE (nag_std_out,'(1x,a,15x,f3.0,6x,f14.4)') 'Total     ', result(20), &
       result(19)
      WRITE (nag_std_out,*)
      WRITE (nag_std_out,'(a,f7.4)') ' Coeff. of determination (R^2)= ', &
       result(5)**2
      WRITE (nag_std_out,*)
      WRITE (nag_std_out,*)
      WRITE (nag_std_out,*) &
       '            ***** Estimates of regression parameters *****      '
      WRITE (nag_std_out,*)
      WRITE (nag_std_out,*) &
       '              Estimate          std-error         t-value  '
      WRITE (nag_std_out,*)
      WRITE (nag_std_out,'(1x,a,5x,f8.4,9x,f9.4,9x,f7.4)') 'intercept', &
       result(7), result(9), result(11), 'slope    ', result(6), result(8), &
       result(10)

      DEALLOCATE (x,y)             ! Deallocate storage

    END PROGRAM nag_lin_reg_ex01
```

2 Program Data

```
Example Program Data for nag_lin_reg_ex01
  8                  : number of data points
  1.0       20.0     : X, Y
  0.0       15.5
  4.0       28.3
  7.5       45.0
  2.5       24.5
  0.0       10.0
 10.0       99.0
  5.0       31.2
```

3 Program Results

```
 Example Program Results for nag_lin_reg_ex01

              ***** Descriptive Statistics *****

 Variable             Mean        Std-deviation

 response (Y)       29.8000           9.4787
 predictor (X)       4.0000           2.4749


             ***** Analysis of variance (ANOVA) table *****

 Source of variation      D.F      Sum of squares      Mean square     F-value

 Regression                1.            345.0940         345.0940     72.4682
 Error                     3.             14.2860           4.7620

 Total                     4.            359.3800
 Coeff. of determination (R^2)=  0.9602


             ***** Estimates of regression parameters *****

               Estimate          std-error         t-value

 intercept      14.7878            2.0155          7.3370
 slope           3.7531            0.4409          8.5128
```

Example 2: Multiple linear regression analysis

This example program shows how `nag_mult_lin_reg` returns various estimates associated with a multiple linear regression analysis. It illustrates three different cases:

- all the variables are included in the model but no constant term is included;
- as for the first case except that the constant term is included in the model;
- as for the second case except that variable 2 is excluded from the model.

1 Program Text

Note. The listing of the example program presented below is double precision. Single precision users are referred to Section 5.2 of the Essential Introduction for further information.

```
PROGRAM nag_lin_reg_ex02

  ! Example Program Text for nag_lin_reg
  ! NAG fl90, Release 3. NAG Copyright 1997.

  ! .. Use Statements ..
  USE nag_examples_io, ONLY : nag_std_out, nag_std_in
  USE nag_lin_reg, ONLY : nag_mult_lin_reg
  USE nag_write_mat, ONLY : nag_write_tri_mat
  ! .. Implicit None Statement ..
  IMPLICIT NONE
  ! .. Intrinsic Functions ..
  INTRINSIC KIND
  ! .. Parameters ..
  INTEGER, PARAMETER :: wp = KIND(1.0D0)
  ! .. Local Scalars ..
  INTEGER :: i, j, m, n, p, rank
  REAL (wp) :: resid_sum_sq
  ! .. Local Arrays ..
  REAL (wp), ALLOCATABLE :: beta(:), cov(:), lev(:), resid(:), std_err(:), &
   x(:,:), y(:)
  LOGICAL, ALLOCATABLE :: var_in_model(:)
  ! .. Executable Statements ..

  WRITE (nag_std_out,*) 'Example Program Results for nag_lin_reg_ex02'

  READ (nag_std_in,*)          ! Skip heading in data file
  READ (nag_std_in,*) n, m

  ALLOCATE (x(n,m),y(n),beta(m+1),std_err(m+1),resid(n),lev(n), &
   var_in_model(m),cov(((m+1)*(m+2))/2)) ! Allocate storage

  DO i = 1, n
    READ (nag_std_in,*) x(i,:), y(i)
  END DO

  DO j = 1, 3
    SELECT CASE (j)
    CASE (1)
      WRITE (nag_std_out,'(/1x,a)') 'CASE 1: No constant term is included'

      CALL nag_mult_lin_reg(x,y,beta,add_alpha=.FALSE.,std_err=std_err, &
       rank=rank,cov=cov,resid=resid,resid_sum_sq=resid_sum_sq,lev=lev)
      ! number of the independent variables in the model p
      ! all x's are included and no constant term
      p = m
    CASE (2)
```

```
          WRITE (nag_std_out,'(/1x,a)') 'CASE 2: Including a constant term'

          CALL nag_mult_lin_reg(x,y,beta,std_err=std_err,rank=rank,cov=cov, &
           resid=resid,resid_sum_sq=resid_sum_sq,lev=lev)
          ! number of the independent variables in the model p
          ! all x's are included and a constant term
          p = m + 1
        CASE (3)
          var_in_model = .TRUE.
          var_in_model(2) = .FALSE.
          WRITE (nag_std_out,'(/1x,a)') &
           'CASE 3: Including a constant term and excluding variable no 2'

          CALL nag_mult_lin_reg(x,y,beta,std_err=std_err,rank=rank, &
           var_in_model=var_in_model,cov=cov,resid=resid, &
           resid_sum_sq=resid_sum_sq,lev=lev)
          ! number of the independent variables in the model p
          ! one x is excluded and a constant term
          p = m
        END SELECT
        IF (rank==p) THEN
          WRITE (nag_std_out,'(1x,a,i4)') 'The model is of full rank, rank=', &
           rank
        ELSE
          WRITE (nag_std_out,'(1x,a,i4)') &
           'The model is not of full rank, rank=', rank
        END IF
        WRITE (nag_std_out,'(1x,a,6i11)') 'x variables       ', (i,i=1,m+1)
        WRITE (nag_std_out,'(1x,a,6f11.6)') 'Estimates of beta ', beta
        WRITE (nag_std_out,'(1x,a,6f11.6)') 'Std_error for beta', std_err

        CALL nag_write_tri_mat('u',cov,format='f11.6',int_col_labels=.TRUE., &
         title='Estimate of covariance matrix',int_row_labels=.TRUE., &
         indent=16)

        WRITE (nag_std_out,*)
        WRITE (nag_std_out,*) ' Residuals       Leverages'
        DO i = 1, n
          WRITE (nag_std_out,'(1x,f10.4,6x,f10.4)') resid(i), lev(i)
        END DO
        WRITE (nag_std_out,'(1x,a,f8.4)') 'Residual sum of squares = ', &
         resid_sum_sq
      END DO

      DEALLOCATE (x,y,beta,std_err,resid,lev,var_in_model, &
       cov)                         ! Deallocate storage

    END PROGRAM nag_lin_reg_ex02
```

2 Program Data

```
Example Program Data for nag_lin_reg_ex02
  18       4                               : n, m (size of matrix X)
  20.0     50.0  75.0   15.0   27.0        : x1, x2, x3, x4, y
  27.0     55.0  60.0   20.0   23.0
  22.0     62.0  68.0   16.0   18.0
  27.0     55.0  60.0   20.0   26.0
  24.0     75.0  72.0    8.0   23.0
  30.0     62.0  73.0   18.0   27.0
  32.0     79.0  71.0   11.0   30.0
  24.0     75.0  72.0    8.0   23.0
  22.0     62.0  68.0   16.0   22.0
```

```
 27.0     55.0   60.0    20.0    24.0
 40.0     90.0   78.0    32.0    16.0
 32.0     79.0   71.0    11.0    28.0
 50.0     84.0   72.0    12.0    31.0
 40.0     90.0   78.0    32.0    22.0
 20.0     50.0   75.0    15.0    24.0
 50.0     84.0   72.0    12.0    31.0
 30.0     62.0   73.0    18.0    29.0
 27.0     55.0   60.0    20.0    22.0
```

3 Program Results

```
Example Program Results for nag_lin_reg_ex02

CASE 1: No constant term is included
The model is of full rank, rank=   4
x variables                 1          2          3          4          5
Estimates of beta    0.514121  -0.288284   0.499899  -0.369595   0.000000
Std_error for beta   0.139212   0.103325   0.075231   0.120192   0.000000
                Estimate of covariance matrix
                            1          2          3          4          5
                1    0.019380  -0.010741   0.002944  -0.003539   0.000000
                2               0.010676  -0.006134   0.001536   0.000000
                3                          0.005660  -0.003440   0.000000
                4                                     0.014446   0.000000
                5                                                0.000000

 Residuals       Leverages
   -0.8167          0.3014
    2.3723          0.0934
   -3.5167          0.1036
    5.3723          0.0934
   -0.7536          0.2965
   -0.3900          0.1179
    4.8953          0.1649
   -0.7536          0.2965
    0.4833          0.1036
    3.3723          0.0934
   -5.7844          0.3851
    2.8953          0.1649
   -2.0478          0.4437
    0.2156          0.3851
   -3.8167          0.3014
   -2.0478          0.4437
    1.6100          0.1179
    1.3723          0.0934
Residual sum of squares = 153.6990

CASE 2: Including a constant term
The model is of full rank, rank=   5
x variables                 1          2          3          4          5
Estimates of beta    0.467935  -0.235195   0.155734  -0.407349  22.624577
Std_error for beta   0.113795   0.085614   0.133618   0.098147   7.818752
                Estimate of covariance matrix
                            1          2          3          4          5
                1    0.012949  -0.007329   0.003827  -0.002110  -0.124799
                2               0.007330  -0.006200   0.000767   0.143450
                3                          0.017854  -0.000702  -0.929953
                4                                     0.009633  -0.102013
                5                                               61.132881

 Residuals       Leverages
```

```
    1.2066          0.3694
   -0.5202          0.2324
   -4.4094          0.1169
    2.4798          0.2324
   -1.1695          0.2994
    0.8831          0.1448
    4.4056          0.1689
   -1.1695          0.2994
   -0.4094          0.1169
    0.4798          0.2324
   -3.2865          0.4888
    2.4056          0.1689
   -1.5896          0.4472
    2.7135          0.4888
   -1.7934          0.3694
   -1.5896          0.4472
    2.8831          0.1448
   -1.5202          0.2324
Residual sum of squares =  93.4861

CASE 3: Including a constant term and excluding variable no 2
The model is of full rank, rank=   4
x variables                 1          2          3          4          5
Estimates of beta    0.232771   0.000000  -0.043217  -0.382753  27.227544
Std_error for beta   0.090830   0.000000   0.136034   0.118405   9.252035
                Estimate of covariance matrix
                            1          2          3          4          5
                1    0.008250   0.000000  -0.003483  -0.001971   0.027346
                2               0.000000   0.000000   0.000000   0.000000
                3                          0.018505  -0.000078  -1.186739
                4                                     0.014020  -0.171734
                5                                               85.600145

 Residuals       Leverages
    4.0996          0.2152
   -0.2643          0.2312
   -5.2857          0.1027
    2.7357          0.2312
   -3.6404          0.1869
    2.8337          0.0747
    2.6024          0.1090
   -3.6404          0.1869
   -1.2857          0.1027
    0.7357          0.2312
   -4.9194          0.4396
    0.6024          0.1090
   -0.1615          0.4096
    1.0806          0.4396
    1.0996          0.2152
   -0.1615          0.4096
    4.8337          0.0747
   -1.2643          0.2312
Residual sum of squares = 147.7570
```

References

[1] Cook R D and Weisberg S (1986) *Residuals and Influence in Regression* Chapman and Hall

[2] Draper N R and Smith H (1985) *Applied Regression Analysis* Wiley (2nd Edition)

[3] Golub G H and Van Loan C F (1983) *Matrix Computations.* John Hopkins University Press (1st Edition).

[4] Hammarling S (1985) The singular value decomposition in multivariate statistics *ACM Signum Newsletter* **20(3)** 2–25

[5] McCullagh P and Nelder J A (1983) *Generalized Linear Models* Chapman and Hall

[6] Searle S R (1971) *Linear Models* Wiley