NAG Library Function Document
nag_full_step_regsn (g02efc)

1 Purpose

nag_full_step_regsn (g02efc) calculates a full stepwise selection from \( p \) variables by using Clarke’s sweep algorithm on the correlation matrix of a design and data matrix, \( Z \). The (weighted) variance-covariance, (weighted) means and sum of weights of \( Z \) must be supplied.

2 Specification

```c
#include <nag.h>
#include <nagg02.h>

void nag_full_step_regsn (Integer m, Integer n, const double wmean[],
const double c[], double sw, Integer isx[], double fin, double fout,
double tau, double b[], double se[], double *rsq, double *rms,
Integer *df,
void (*monfun)(Nag_FullStepwise flag, Integer var, double val,
Nag_Comm *comm),
Nag_Comm *comm, NagError *fail)
```

3 Description

The general multiple linear regression model is defined by

\[
y = \beta_0 + X\beta + \epsilon,
\]

where

- \( y \) is a vector of \( n \) observations on the dependent variable,
- \( \beta_0 \) is an intercept coefficient,
- \( X \) is an \( n \) by \( p \) matrix of \( p \) explanatory variables,
- \( \beta \) is a vector of \( p \) unknown coefficients, and
- \( \epsilon \) is a vector of length \( n \) of unknown, Normally distributed, random errors.

nag_full_step_regsn (g02efc) employs a full stepwise regression to select a subset of explanatory variables from the \( p \) available variables (the intercept is included in the model) and computes regression coefficients and their standard errors, and various other statistical quantities, by minimizing the sum of squares of residuals. The method applies repeatedly a forward selection step followed by a backward elimination step and halts when neither step updates the current model.

The criterion used to update a current model is the variance ratio of residual sum of squares. Let \( s_1 \) and \( s_2 \) be the residual sum of squares of the current model and this model after undergoing a single update, with degrees of freedom \( q_1 \) and \( q_2 \), respectively. Then the condition:

\[
\frac{(s_2 - s_1) / (q_2 - q_1)}{s_1 / q_1} > f_1,
\]

must be satisfied if a variable \( k \) will be considered for entry to the current model, and the condition:

\[
\frac{(s_1 - s_2) / (q_1 - q_2)}{s_1 / q_1} < f_2,
\]

must be satisfied if a variable \( k \) will be considered for removal from the current model, where \( f_1 \) and \( f_2 \) are user-supplied values and \( f_2 \leq f_1 \).
In the entry step the entry statistic is computed for each variable not in the current model. If no variable is associated with a test value that exceeds $f_1$ then this step is terminated; otherwise the variable associated with the largest value for the entry statistic is entered into the model.

In the removal step the removal statistic is computed for each variable in the current model. If no variable is associated with a test value less than $f_2$ then this step is terminated; otherwise the variable associated with the smallest value for the removal statistic is removed from the model.

The data values $X$ and $y$ are not provided as input to the function. Instead, summary statistics of the design and data matrix $Z = (X \mid y)$ are required.

Explanatory variables are entered into and removed from the current model by using sweep operations on the correlation matrix $R$ of $Z$, given by:

$$
R = 
\begin{pmatrix}
1 & \ldots & r_{1p} & r_{1y} \\
\vdots & \ddots & \vdots & \vdots \\
r_{p1} & \ldots & 1 & r_{py} \\
r_{y1} & \ldots & r_{yp} & 1
\end{pmatrix},
$$

where $r_{ij}$ is the correlation between the explanatory variables $i$ and $j$, for $i = 1, 2, \ldots, p$ and $j = 1, 2, \ldots, p$, and $r_{yi}$ (and $r_{iy}$) is the correlation between the response variable $y$ and the $i$th explanatory variable, for $i = 1, 2, \ldots, p$.

A sweep operation on the $k$th row and column ($k \leq p$) of $R$ replaces:

- $r_{kk}$ by $-1/r_{kk}$;
- $r_{ik}$ by $r_{ik}/r_{kk}$, $i = 1, 2, \ldots, p + 1$ ($i \neq k$);
- $r_{kj}$ by $r_{kj}/r_{kk}$, $j = 1, 2, \ldots, p + 1$ ($j \neq k$);
- $r_{ij}$ by $r_{ij} - r_{ik}r_{kj}/r_{kk}$, $i = 1, 2, \ldots, p + 1$ ($i \neq k$); $j = 1, 2, \ldots, p + 1$ ($j \neq k$).

The $k$th explanatory variable is eligible for entry into the current model if it satisfies the collinearity tests: $r_{kk} > \tau$ and

$$
\left(\frac{r_{ii} - r_{ik}r_{ki}}{r_{kk}}\right) \tau \leq 1,
$$

for a user-supplied value (> 0) of $\tau$ and where the index $i$ runs over explanatory variables in the current model. The sweep operation is its own inverse, therefore pivoting on an explanatory variable $k$ in the current model has the effect of removing it from the model.

Once the stepwise model selection procedure is finished, the function calculates:

(a) the least squares estimate for the $i$th explanatory variable included in the fitted model;
(b) standard error estimates for each coefficient in the final model;
(c) the square root of the mean square of residuals and its degrees of freedom;
(d) the multiple correlation coefficient.

The function makes use of the symmetry of the sweep operations and correlation matrix which reduces by almost one half the storage and computation required by the sweep algorithm, see Clarke (1981) for details.

4 References


*Appl. Statist.* 31 166–169


5 Arguments

1: \( m \) – Integer

*Input*

*On entry:* the number of explanatory variables available in the design matrix, \( Z \).

*Constraint:* \( m > 1 \).

2: \( n \) – Integer

*Input*

*On entry:* the number of observations used in the calculations.

*Constraint:* \( n > 1 \).

3: \( \text{wmean}[m + 1] \) – const double

*Input*

*On entry:* the mean of the design matrix, \( Z \).

4: \( c[dim] \) – const double

*Input*

*Note:* the dimension, \( dim \), of the array \( c \) must be at least \((m + 1) \times (m + 2)/2\).

*On entry:* the upper-triangular variance-covariance matrix packed by column for the design matrix, \( Z \). Because the function computes the correlation matrix \( R \) from \( c \), the variance-covariance matrix need only be supplied up to a scaling factor.

5: \( sw \) – double

*Input*

*On entry:* if weights were used to calculate \( c \) then \( sw \) is the sum of positive weight values; otherwise \( sw \) is the number of observations used to calculate \( c \).

*Constraint:* \( sw > 1.0 \).

6: \( \text{isx}[m] \) – Integer

*Input/Output*

*On entry:* the value of \( \text{isx}[j - 1] \) determines the set of variables used to perform full stepwise model selection, for \( j = 1, 2, \ldots, m \).

- \( \text{isx}[j - 1] = -1 \)
  - To exclude the variable corresponding to the \( j \)th column of \( X \) from the final model.

- \( \text{isx}[j - 1] = 1 \)
  - To consider the variable corresponding to the \( j \)th column of \( X \) for selection in the final model.

- \( \text{isx}[j - 1] = 2 \)
  - To force the inclusion of the variable corresponding to the \( j \)th column of \( X \) in the final model.

*Constraint:* \( \text{isx}[j - 1] = -1, 1 \) or 2, for \( j = 1, 2, \ldots, m \).

*On exit:* the value of \( \text{isx}[j - 1] \) indicates the status of the \( j \)th explanatory variable in the model.

- \( \text{isx}[j - 1] = -1 \)
  - Forced exclusion.

- \( \text{isx}[j - 1] = 0 \)
  - Excluded.

- \( \text{isx}[j - 1] = 1 \)
  - Selected.

- \( \text{isx}[j - 1] = 2 \)
  - Forced selection.
7: \textbf{fin} – double \hspace{1cm} \textit{Input}

\textit{On entry:} the value of the variance ratio which an explanatory variable must exceed to be included in a model.

\textit{Suggested value:} \textbf{fin} = 4.0

\textit{Constraint:} \textbf{fin} > 0.0.

8: \textbf{fout} – double \hspace{1cm} \textit{Input}

\textit{On entry:} the explanatory variable in a model with the lowest variance ratio value is removed from the model if its value is less than \textbf{fout}. \textbf{fout} is usually set equal to the value of \textbf{fin}; a value less than \textbf{fin} is occasionally preferred.

\textit{Suggested value:} \textbf{fout} = \textbf{fin}

\textit{Constraint:} 0.0 \leq \textbf{fout} \leq \textbf{fin}.

9: \textbf{tau} – double \hspace{1cm} \textit{Input}

\textit{On entry:} the tolerance, $\tau$, for detecting collinearities between variables when adding or removing an explanatory variable from a model. Explanatory variables deemed to be collinear are excluded from the final model.

\textit{Suggested value:} \textbf{tau} = 1.0 \times 10^{-6}

\textit{Constraint:} \textbf{tau} > 0.0.

10: \textbf{b}_{m+1} – double \hspace{1cm} \textit{Output}

\textit{On exit:} \textbf{b}[0] contains the estimate for the intercept term in the fitted model. If \textbf{isx}[j - 1] \neq 0 then \textbf{b}[j] contains the estimate for the $j$th explanatory variable in the fitted model; otherwise \textbf{b}[j] = 0.

11: \textbf{se}_{m+1} – double \hspace{1cm} \textit{Output}

\textit{On exit:} \textbf{se}[j - 1] contains the standard error for the estimate of \textbf{b}[j - 1], for $j = 1, 2, \ldots, m + 1$.

12: \textbf{rsq} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} the $R^2$-statistic for the fitted regression model.

13: \textbf{rms} – double * \hspace{1cm} \textit{Output}

\textit{On exit:} the mean square of residuals for the fitted regression model.

14: \textbf{df} – Integer * \hspace{1cm} \textit{Output}

\textit{On exit:} the number of degrees of freedom for the sum of squares of residuals.

15: \textbf{monfun} – function, supplied by the user \hspace{1cm} \textit{External Function}

You may define your own function or specify the NAG defined default function nag_full_step_regsn_monfun (g02efh). If this facility is not required then the NAG defined null function macro NULLFN can be substituted.

The specification of \textbf{monfun} is:

\begin{verbatim}
void monfun (Nag_FullStepwise flag, Integer var, double val,
            Nag_Comm *comm)
\end{verbatim}
1:  **flag** – Nag_FullStepwise

   *Input*

   *On entry:* the value of **flag** indicates the stage of the stepwise selection of explanatory variables.

   **flag** = Nag_AddVar 
   Variable var was added to the current model.

   **flag** = Nag_BeginBackward 
   Beginning the backward elimination step.

   **flag** = Nag_ColinearVar 
   Variable var failed the collinearity test and is excluded from the model.

   **flag** = Nag_DropVar 
   Variable var was dropped from the current model.

   **flag** = Nag_BeginForward 
   Beginning the forward selection step.

   **flag** = Nag_NoRemoveVar 
   Backward elimination did not remove any variables from the current model.

   **flag** = Nag_BeginStepwise 
   Starting stepwise selection procedure.

   **flag** = Nag_VarianceRatio 
   The variance ratio for variable var takes the value val.

   **flag** = Nag_FinishStepwise 
   Finished stepwise selection procedure.

2:  **var** – Integer

   *Input*

   *On entry:* the index of the explanatory variable in the design matrix Z to which **flag** pertains.

3:  **val** – double

   *Input*

   *On entry:* if **flag** = Nag_VarianceRatio, **val** is the variance ratio value for the coefficient associated with explanatory variable index var.

4:  **comm** – Nag_Comm *

   Pointer to structure of type Nag_Comm; the following members are relevant to monfun.

   user – double *

   iuser – Integer *

   p – Pointer

   The type Pointer will be void *. Before calling nag_full_step_regsn (g02efc) you may allocate memory and initialize these pointers with various quantities for use by monfun when called from nag_full_step_regsn (g02efc) (see Section 3.2.1.1 in the Essential Introduction).

16:  **comm** – Nag_Comm *

   The NAG communication argument (see Section 3.2.1.1 in the Essential Introduction).

17:  **fail** – NagError *

   *Input/Output*

   The NAG error argument (see Section 3.6 in the Essential Introduction).
6  Error Indicators and Warnings

NE_ALLOC_FAIL
  Dynamic memory allocation failed.
  See Section 3.2.1.2 in the Essential Introduction for further information.

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

NE_FREE_VARS
  No free variables from which to select.
  At least one element of isx should be set to 1.

NE_INT
  On entry, \( m = \langle value \rangle \).
  Constraint: \( m > 1 \).
  On entry, \( n = \langle value \rangle \).
  Constraint: \( n > 1 \).

NE_INT_ARRAY_ELEM_CONS
  On entry, invalid value for isx\[\langle value \rangle\] = \langle value \rangle.

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.
  An unexpected error has been triggered by this function. Please contact NAG.
  See Section 3.6.6 in the Essential Introduction for further information.

NE_MODEL_INFEASIBLE
  All variables are collinear, no model to select.

NE_NO_LICENCE
  Your licence key may have expired or may not have been installed correctly.
  See Section 3.6.5 in the Essential Introduction for further information.

NE_NOT_POS_DEF
  The design and data matrix \( Z \) is not positive definite, results may be inaccurate. All output is
  returned as documented.

NE_REAL
  On entry, \( \text{fin} = \langle value \rangle \).
  Constraint: \( \text{fin} > 0.0 \).
  On entry, \( \text{sw} = \langle value \rangle \).
  Constraint: \( \text{sw} > 1.0 \).
  On entry, \( \text{tau} = \langle value \rangle \).
  Constraint: \( \text{tau} > 0.0 \).

NE_REAL_2
  On entry, \( \text{fout} = \langle value \rangle; \text{fin} = \langle value \rangle \).
  Constraint: \( 0.0 \leq \text{fout} \leq \text{fin} \).
7 Accuracy

Nag_full_step_regsn (g02efc) returns a warning if the design and data matrix is not positive definite.

8 Parallelism and Performance

Not applicable.

9 Further Comments

Although the condition for removing or adding a variable to the current model is based on a ratio of variances, these values should not be interpreted as $F$-statistics with the usual interpretation of significance unless the probability levels are adjusted to account for correlations between variables under consideration and the number of possible updates (see, e.g., Draper and Smith (1985)).

Nag_full_step_regsn (g02efc) allocates internally $\mathcal{O}(4 \times m + (m + 1) \times (m + 2)/2 + 2)$ of double storage.

10 Example

This example calculates a full stepwise model selection for the Hald data described in Dempster (1969). Means, the upper-triangular variance-covariance matrix and the sum of weights are calculated by nag_sum_sqs (g02buc). The NAG defined default monitor function nag_full_step_regsn_monfun (g02efh) is used to print information at each step of the model selection process.

10.1 Program Text

/* nag_full_step_regsn (g02efc) Example Program. */
/* Copyright 2014 Numerical Algorithms Group. */
/* Mark 8, 2004. */
/
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg02.h>

int main(void)
{
    /* Scalars */
    double fin, fout, rms, rsq, sw, tau;
    Integer df, exit_status, i, j, m, n, pdx;

    /* Arrays */
    double *b = 0, *c = 0, *se = 0, *wmean = 0, *x = 0;
    Integer *isx = 0;

    /* Nag types */
    Nag_OrderType order;
    Nag_SumSquare mean;
    Nag_Comm comm;
    NagError fail;

    #ifdef NAG_COLUMN_ORDER
    #define X(I, J) x[(J-1)*pdx +I-1]
    order = Nag_ColMajor;
    #else
    #define X(I, J) x[(I-1)*pdx +J-1]
    #endif
    #endif

    /* Nag statements */
    Nag_Initialize(nag);

    /* Main program */
    /* ... */

    /* Nag performance statistics */
    Nag_Print(nag);

    return 0;
}
order = Nag_RowMajor;
#endif

INIT_FAIL(fail);
exit_status = 0;

printf("nag_full_step_regsn (g02efc) Example Program Results\n\n");

/* Skip heading in data file */
#endif_WIN32
  scanf_s("%*[\n"]);
#else
  scanf("%*[\n"]);
#endif
#endif_WIN32
  scanf_s("%"NAG_IFMT" %"NAG_IFMT" %lf %lf %lf", &n, &m, &fin, 
          &fout, &tau);
#else
  scanf("%"NAG_IFMT" %"NAG_IFMT" %lf %lf %lf", &n, &m, &fin, 
          &fout, &tau);
#endif
#endif_WIN32
  scanf("%*[\n"]);
#else
  scanf("%*[\n"]);
#endif
if (n > 1 && m > 1) {
  /* Allocate memory */
  if (!(b = NAG_ALLOC(m+1, double)) || 
      !(c = NAG_ALLOC((m+1)*(m+2)/2, double)) || 
      !(se = NAG_ALLOC(m+1, double)) || 
      !(wmean = NAG_ALLOC(m+1, double)) || 
      !(x = NAG_ALLOC(n * (m+1), double)) || 
      !(isx = NAG_ALLOC(m, Integer)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }
  else {
    printf("Invalid n or m.\n");
    exit_status = 1;
    return exit_status;
  }
#endif_NAG_COLUMN_ORDER
  pdx = n;
#else
  pdx = m+1;
#endif
  for (i = 1; i <= n; ++i) {
    for (j = 1; j <= m+1; ++j) {
      #ifndef_WIN32
        scanf_s("%lf", &X(i, j));
      #else
        scanf("%lf", &X(i, j));
      #endif
    }
  }
  for (j = 1; j <= m; ++j) {
    #ifndef_WIN32
      scanf("%*[\n"]);
    #else
      scanf("%*[\n"]);
    #endif
  }
}
```c
#include "nag.c"
#include "nag_i.c"

char *NAGPRINTF(const char *fmt, ...) {
    va_list args;
    va_start(args, fmt);
    return nag_vprintf(fmt, args);
}
#endif

#define _WIN32

nag_sum_sqs(order, mean, n, m+1, x, pdx, 0, &sw, wmean, c, &fail);  
if (fail.code != NE_NOERROR) {
    printf("Error from nag_sum_sqs (g02buc).
%s
.", fail.message);
    exit_status = 1;
    goto END;
}

fflush(stdout);

nag_full_step_regsn(m, n, wmean, c, sw, isx, fin, fout, tau, b, se, &rsq,  
&rms, &df, nag_full_step_regsn_monfun, &comm, &fail);
if (fail.code != NE_NOERROR) {
    printf("Error from nag_full_step_regsn (g02efc).
%s
.", fail.message);
    exit_status = 1;
    goto END;
}

/* Display summary information for fitted model */
printf("\n");
printf("Fitted Model Summary\n");
printf("%10s %-10s%19s
", "Term", " Estimate", "Standard Error");
printf("%10s %11.3e%17.3e\n", "Intercept:", b[0], se[0]);
for (i = 1; i <= m; ++i) {
    j = isx[i-1];
    if (j == 1 || j == 2) {
        printf("%-10s%3\n%11.3e%17.3e\n", "Variable:", i, b[i], se[i]);
    }
}
printf("\n");
printf("RMS: %-12.3e\n
", rms);

END:
NAG_FREE(b);
NAG_FREE(c);
NAG_FREE(se);
NAG_FREE(wmean);
NAG_FREE(x);
NAG_FREE(isx);
return exit_status;
}
```
10.2 Program Data

nag_full_step_regsn (g02efc) Example Program Data
13 4 4.0 2.0 1.0e-6 1 : N,M,FIN,FOUT,TAU,MONLEV
7.0 26.0 6.0 60.0 78.5
1.0 29.0 15.0 52.0 74.3
11.0 56.0 8.0 20.0 104.3
11.0 31.0 8.0 47.0 87.6
7.0 52.0 6.0 33.0 95.9
11.0 55.0 9.0 22.0 109.2
3.0 71.0 17.0 6.0 102.7
1.0 31.0 22.0 44.0 72.5
2.0 54.0 18.0 22.0 93.1
21.0 47.0 4.0 26.0 115.9
1.0 40.0 23.0 34.0 83.8
11.0 66.0 9.0 12.0 113.3
10.0 68.0 8.0 12.0 109.4 : End of X array of size N by M+1
1 1 1 1 : Array ISX

10.3 Program Results

nag_full_step_regsn (g02efc) Example Program Results

Starting Stepwise Selection

Forward Selection
Variable 1 Variance ratio = 1.260E+01
Variable 2 Variance ratio = 2.196E+01
Variable 3 Variance ratio = 4.403E+00
Variable 4 Variance ratio = 2.280E+01

Adding variable 4 to model

Backward Selection
Variable 4 Variance ratio = 2.280E+01
Keeping all current variables

Forward Selection
Variable 1 Variance ratio = 1.082E+02
Variable 2 Variance ratio = 1.725E-01
Variable 3 Variance ratio = 4.029E+01

Adding variable 1 to model

Backward Selection
Variable 1 Variance ratio = 1.082E+02
Variable 4 Variance ratio = 1.593E+02
Keeping all current variables

Forward Selection
Variable 2 Variance ratio = 5.026E+00
Variable 3 Variance ratio = 4.236E+00

Adding variable 2 to model

Backward Selection
Variable 1 Variance ratio = 1.540E+02
Variable 2 Variance ratio = 5.026E+00
Variable 4 Variance ratio = 1.863E+00

Dropping variable 4 from model

Forward Selection
Variable 3 Variance ratio = 1.832E+00
Variable 4 Variance ratio = 1.863E+00

Finished Stepwise Selection
Fitted Model Summary

<table>
<thead>
<tr>
<th>Term</th>
<th>Estimate</th>
<th>Standard Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>5.258e+01</td>
<td>2.294e+00</td>
</tr>
<tr>
<td>Variable: 1</td>
<td>1.468e+00</td>
<td>1.213e-01</td>
</tr>
<tr>
<td>Variable: 2</td>
<td>6.623e-01</td>
<td>4.585e-02</td>
</tr>
</tbody>
</table>

RMS: 5.790e+00