NAG Library Function Document

nag_regsn_quant_linear_iid (g02qfc)

1 Purpose

nag_regsn_quant_linear_iid (g02qfc) performs a multiple linear quantile regression, returning the parameter estimates and associated confidence limits based on an assumption of Normal, independent, identically distributed errors. nag_regsn_quant_linear_iid (g02qfc) is a simplified version of nag_regsn_quant_linear (g02qgc).

2 Specification

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

void nag_regsn_quant_linear_iid (Integer n, Integer m, const double x[],
     const double y[], Integer ntau, const double tau[], double *df,
     double b[], double bl[], double bu[], Integer info[], NagError *fail)
```

3 Description

Given a vector of n observed values, $y = \{y_i : i = 1, 2, \ldots, n\}$, an $n \times p$ design matrix X, a column vector, x, of length p holding the ith row of X and a quantile $\tau \in (0, 1)$, nag_regsn_quant_linear_iid (g02qfc) estimates the p-element vector β as the solution to

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}} \sum_{i=1}^{n} \rho_\tau \left(y_i - x_i^{\mathrm{T}} \beta\right) \tag{1}$$

where ρ_τ is the piecewise linear loss function $\rho_\tau(z) = z(\tau - I(z < 0))$, and $I(z < 0)$ is an indicator function taking the value 1 if $z < 0$ and 0 otherwise.

nag_regsn_quant_linear_iid (g02qfc) assumes Normal, independent, identically distributed (IID) errors and calculates the asymptotic covariance matrix from

$$\Sigma = \frac{\tau(1-\tau)}{n} (s(\tau))^2 \left(X^{\mathrm{T}} X\right)^{-1}$$

where s is the sparsity function, which is estimated from the residuals, $r_i = y_i - x_i^{\mathrm{T}} \hat{\beta}$ (see Koenker (2005)).

Given an estimate of the covariance matrix, $\hat{\Sigma}$, lower, $\hat{\beta}_L$, and upper, $\hat{\beta}_U$, limits for a 95% confidence interval are calculated for each of the p parameters, via

$$\hat{\beta}_{Li} = \hat{\beta}_i - t_{n-p,0.975} \sqrt{\hat{\Sigma}_{ii}}, \hat{\beta}_{Ui} = \hat{\beta}_i + t_{n-p,0.975} \sqrt{\hat{\Sigma}_{ii}}$$

where $t_{n-p,0.975}$ is the 97.5 percentile of the Student's t distribution with $n - k$ degrees of freedom, where k is the rank of the cross-product matrix $X^{\mathrm{T}} X$.

Further details of the algorithms used by nag_regsn_quant_linear_iid (g02qfc) can be found in the documentation for nag_regsn_quant_linear (g02qgc).

4 References

Koenker R (2005) *Quantile Regression* Econometric Society Monographs, Cambridge University Press, New York

5 Arguments

1: **n** – Integer *Input*

On entry: n, the number of observations in the dataset.

Constraint: $\mathbf{n} \geq 2$.

2: **m** – Integer *Input*

On entry: p, the number of variates in the model.

Constraint: $1 \leq \mathbf{m} < \mathbf{n}$.

3: $\mathbf{x}[\mathbf{n} \times \mathbf{m}]$ – const double *Input*

Note: where $\mathbf{X}(i,j)$ appears in this document, it refers to the array element $\mathbf{x}[(i-1) \times \mathbf{m} + j - 1]$.

On entry: X, the design matrix, with the ith value for the jth variate supplied in $\mathbf{X}(i,j)$, for $i = 1, 2, \ldots, \mathbf{n}$ and $j = 1, 2, \ldots, \mathbf{m}$.

4: $\mathbf{y}[\mathbf{n}]$ – const double *Input*

On entry: y, observations on the dependent variable.

5: **ntau** – Integer *Input*

On entry: the number of quantiles of interest.

Constraint: $\mathbf{ntau} \geq 1$.

6: $\mathbf{tau}[\mathbf{ntau}]$ – const double *Input*

On entry: the vector of quantiles of interest. A separate model is fitted to each quantile.

Constraint: $\sqrt{\epsilon} < \mathbf{tau}[l-1] < 1 - \sqrt{\epsilon}$ where ϵ is the ***machine precision*** returned by nag_machine_precision (X02AJC), for $l = 1, 2, \ldots, \mathbf{ntau}$.

7: **df** – double * *Output*

On exit: the degrees of freedom given by $n - k$, where n is the number of observations and k is the rank of the cross-product matrix $X^{\mathrm{T}}X$.

8: $\mathbf{b}[\mathbf{m} \times \mathbf{ntau}]$ – double *Output*

Note: where $\mathbf{B}(j,l)$ appears in this document, it refers to the array element $\mathbf{b}[(l-1) \times \mathbf{m} + j - 1]$.

On exit: $\hat{\beta}$, the estimates of the parameters of the regression model, with $\mathbf{B}(j,l)$ containing the coefficient for the variable in column j of $\mathbf{X}$, estimated for $\tau = \mathbf{tau}[l-1]$.

9: $\mathbf{bl}[\mathbf{m} \times \mathbf{ntau}]$ – double *Output*

Note: where $\mathbf{BL}(j,l)$ appears in this document, it refers to the array element $\mathbf{bl}[(l-1) \times \mathbf{m} + j - 1]$.

On exit: $\hat{\beta}_L$, the lower limit of a 95% confidence interval for $\hat{\beta}$, with $\mathbf{BL}(j,l)$ holding the lower limit associated with $\mathbf{B}(j,l)$.

10: $\mathbf{bu}[\mathbf{m} \times \mathbf{ntau}]$ – double *Output*

Note: where $\mathbf{BU}(j,l)$ appears in this document, it refers to the array element $\mathbf{bu}[(l-1) \times \mathbf{m} + j - 1]$.

On exit: $\hat{\beta}_U$, the upper limit of a 95% confidence interval for $\hat{\beta}$, with $\mathbf{BU}(j,l)$ holding the upper limit associated with $\mathbf{B}(j,l)$.

11: **info**[**ntau**] – Integer *Output*

On exit: **info**[l] holds additional information concerning the model fitting and confidence limit calculations when $\tau = \textbf{tau}[l]$.

Code	Warning
0	Model fitted and confidence limits calculated successfully.
1	The function did not converge whilst calculating the parameter estimates. The returned values are based on the estimate at the last iteration.
2	A singular matrix was encountered during the optimization. The model was not fitted for this value of τ.
8	The function did not converge whilst calculating the confidence limits. The returned limits are based on the estimate at the last iteration.
16	Confidence limits for this value of τ could not be calculated. The returned upper and lower limits are set to a large positive and large negative value respectively.

It is possible for multiple warnings to be applicable to a single model. In these cases the value returned in **info** is the sum of the corresponding individual nonzero warning codes.

12: **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.

NE_BAD_PARAM

On entry, argument ⟨*value*⟩ had an illegal value.

NE_INT

On entry, **n** = ⟨*value*⟩.
Constraint: **n** ≥ 2.

On entry, **ntau** = ⟨*value*⟩.
Constraint: **ntau** ≥ 1.

NE_INT_2

On entry, **m** = ⟨*value*⟩ and **n** = ⟨*value*⟩.
Constraint: $1 \leq \textbf{m} < \textbf{n}$.

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_REAL_ARRAY

On entry, **tau**[⟨*value*⟩] = ⟨*value*⟩ is invalid.

NW_POTENTIAL_PROBLEM

A potential problem occurred whilst fitting the model(s).
Additional information has been returned in **info**.

7 Accuracy

Not applicable.

8 Parallelism and Performance

nag_regsn_quant_linear_iid (g02qfc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_regsn_quant_linear_iid (g02qfc) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

Calling nag_regsn_quant_linear_iid (g02qfc) is equivalent to calling nag_regsn_quant_linear (g02qgc) with

order = Nag_RowMajor, **intcpt** = Nag_NoIntercept,

no weights supplied, i.e., **wt** set to **NULL**,

pddat = **m**,

setting each element of **isx** to 1,

ip = **m**,

Interval Method = 'IID', and

Significance Level = 0.95.

10 Example

A quantile regression model is fitted to Engels 1857 study of household expenditure on food. The model regresses the dependent variable, household food expenditure, against household income. An intercept is included in the model by augmenting the dataset with a column of ones.

10.1 Program Text

```
/* nag_regsn_quant_linear_iid (g02qfc) Example Program.
 *
 * Copyright 2011, Numerical Algorithms Group.
 *
 * Mark 23, 2011.
 */
/* Pre-processor includes */
#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg02.h>

#define B(i,j) b[(j) * m + i]
#define BU(i,j) bu[(j) * m + i]
#define BL(i,j) bl[(j) * m + i]
#define X(i,j) x[(i) * m + j]

int main(void)
{
  /* Integer scalar and array declarations */
  Integer i, j, l, m, n, ntau;
  Integer *info = 0;
  Integer exit_status = 0;

  /* NAG structures */
  NagError fail;

  /* Double scalar and array declarations */
  double df;
```

```
double *b = 0, *bl = 0, *bu = 0, *tau = 0, *x = 0, *y = 0;

/* Initialise the error structure */
INIT_FAIL(fail);

printf("nag_regsn_quant_linear_iid (g02qfc) Example Program Results\n\n");

/* Skip heading in data file */
scanf("%*[^\n] ");

/* Read in the problem size */
scanf("%ld%ld%ld%*[^\n] ",&n,&m,&ntau);

/* Allocate memory for input arrays */
if (!(y = NAG_ALLOC(n, double)) ||
    !(tau = NAG_ALLOC(ntau, double)) ||
    !(x = NAG_ALLOC(n*m, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

/* Read in the data */
for (i = 0; i < n; i++) {
  for (j = 0; j < m; j++)
    scanf("%lf", &X(i,j));
  scanf("%lf",&y[i]);
}
scanf("%*[^\n] ");

/* Read in the quantiles required */
for (l = 0; l < ntau; l++) {
  scanf("%lf",&tau[l]);
}
scanf("%*[^\n] ");

/* Allocate memory for output arrays */
if (!(b = NAG_ALLOC(m*ntau, double)) ||
    !(info = NAG_ALLOC(ntau, Integer)) ||
    !(bl = NAG_ALLOC(m*ntau, double)) ||
    !(bu = NAG_ALLOC(m*ntau, double)))
  {
    printf("Allocation failure\n");
    exit_status = -1;
    goto END;
  }

/* nag_regsn_quant_linear_iid (g02qfc).Quantile linear regression, simple
   interface, independent, identically distributed (IID) errors */
nag_regsn_quant_linear_iid(n,m,x,y,ntau,tau,&df,b,bl,bu,info,&fail);
if (fail.code != NE_NOERROR) {
  printf("Error from nag_regsn_quant_linear_iid (g02qfc).\n%s\n",
          fail.message);
  if (fail.code == NW_POTENTIAL_PROBLEM) {
    printf("Additional error information: ");
    for (i = 0; i < ntau; i++)
      printf("%ld ",info[i]);
    printf("\n");
  } else {
    exit_status = 1;
    goto END;
  }
}

/* Display the parameter estimates */
for (l = 0; l < ntau; l++) {
  printf(" Quantile: %6.3f\n\n", tau[l]);
  printf("         Lower   Parameter   Upper\n");
  printf("         Limit   Estimate    Limit\n");
  for (j = 0; j < m; j++) {
```

```
      printf(" %3ld   %7.3f   %7.3f   %7.3f\n", j + 1, BL(j,l), B(j,l),
              BU(j,l));
    }
    printf("\n\n");
  }

 END:

  NAG_FREE(info);
  NAG_FREE(b);
  NAG_FREE(bl);
  NAG_FREE(bu);
  NAG_FREE(tau);
  NAG_FREE(x);
  NAG_FREE(y);

  return(exit_status);
}
```

10.2 Program Data

```
nag_regsn_quant_linear_iid (g02qfc) Example Program Data
      235       2          5                                 :: n, m, ntau
1.0   420.1577  255.8394   1.0   800.7990  572.0807   1.0   643.3571  459.8177
1.0   541.4117  310.9587   1.0  1245.6964  907.3969   1.0  2551.6615  863.9199
1.0   901.1575  485.6800   1.0  1201.0002  811.5776   1.0  1795.3226  831.4407
1.0   639.0802  402.9974   1.0   634.4002  427.7975   1.0  1165.7734  534.7610
1.0   750.8756  495.5608   1.0   956.2315  649.9985   1.0   815.6212  392.0502
1.0   945.7989  633.7978   1.0  1148.6010  860.6002   1.0  1264.2066  934.9752
1.0   829.3979  630.7566   1.0  1768.8236 1143.4211   1.0  1095.4056  813.3081
1.0   979.1648  700.4409   1.0  2822.5330 2032.6792   1.0   447.4479  263.7100
1.0  1309.8789  830.9586   1.0   922.3548  590.6183   1.0  1178.9742  769.0838
1.0  1492.3987  815.3602   1.0  2293.1920 1570.3911   1.0   975.8023  630.5863
1.0   502.8390  338.0014   1.0   627.4726  483.4800   1.0  1017.8522  645.9874
1.0   616.7168  412.3613   1.0   889.9809  600.4804   1.0   423.8798  319.5584
1.0   790.9225  520.0006   1.0  1162.2000  696.2021   1.0   558.7767  348.4518
1.0   555.8786  452.4015   1.0  1197.0794  774.7962   1.0   943.2487  614.5068
1.0   713.4412  512.7201   1.0   530.7972  390.5984   1.0  1348.3002  662.0096
1.0   838.7561  658.8395   1.0  1142.1526  612.5619   1.0  2340.6174 1504.3708
1.0   535.0766  392.5995   1.0  1088.0039  708.7622   1.0   587.1792  406.2180
1.0   596.4408  443.5586   1.0   484.6612  296.9192   1.0  1540.9741  692.1689
1.0   924.5619  640.1164   1.0  1536.0201 1071.4627   1.0  1115.8481  588.1371
1.0   487.7583  333.8394   1.0   678.8974  496.5976   1.0  1044.6843  511.2609
1.0   692.6397  466.9583   1.0   671.8802  503.3974   1.0  1389.7929  700.5600
1.0   997.8770  543.3969   1.0   690.4683  357.6411   1.0  2497.7860 1301.1451
1.0   506.9995  317.7198   1.0   860.6948  430.3376   1.0  1585.3809  879.0660
1.0   654.1587  424.3209   1.0   873.3095  624.6990   1.0  1862.0438  912.8851
1.0   933.9193  518.9617   1.0   894.4598  582.5413   1.0  2008.8546 1509.7812
1.0   433.6813  338.0014   1.0  1148.6470  580.2215   1.0   697.3099  484.0605
1.0   587.5962  419.6412   1.0   926.8762  543.8807   1.0   571.2517  399.6703
1.0   896.4746  476.3200   1.0   839.0414  588.6372   1.0   598.3465  444.1001
1.0   454.4782  386.3602   1.0   829.4974  627.9999   1.0   461.0977  248.8101
1.0   584.9989  423.2783   1.0  1264.0043  712.1012   1.0   977.1107  527.8014
1.0   800.7990  503.3572   1.0  1937.9771  968.3949   1.0   883.9849  500.6313
1.0   502.4369  354.6389   1.0   698.8317  482.5816   1.0   718.3594  436.8107
1.0   713.5197  497.3182   1.0   920.4199  593.1694   1.0   543.8971  374.7990
1.0   906.0006  588.5195   1.0  1897.5711 1033.5658   1.0  1587.3480  726.3921
1.0   880.5969  654.5971   1.0   891.6824  693.6795   1.0  4957.8130 1827.2000
1.0   796.8289  550.7274   1.0   889.6784  693.6795   1.0   969.6838  523.4911
1.0   854.8791  528.3770   1.0  1221.4818  761.2791   1.0   419.9980  334.9998
1.0  1167.3716  640.4813   1.0   544.5991  361.3981   1.0   561.9990  473.2009
1.0   523.8000  401.3204   1.0  1031.4491  628.4522   1.0   689.5988  581.2029
1.0   670.7792  435.9990   1.0  1462.9497  771.4486   1.0  1398.5203  929.7540
1.0   377.0584  276.5606   1.0   830.4353  757.1187   1.0   820.8168  591.1974
1.0   851.5430  588.3488   1.0   975.0415  821.5970   1.0   875.1716  637.5483
1.0  1121.0937  664.1978   1.0  1337.9983 1022.3202   1.0  1392.4499  674.9509
1.0   625.5179  444.8602   1.0   867.6427  679.4407   1.0  1256.3174  776.7589
1.0   805.5377  462.8995   1.0   725.7459  538.7491   1.0  1362.8590  959.5170
1.0   558.5812  377.7792   1.0   989.0056  679.9981   1.0  1999.2552 1250.9643
1.0   884.4005  553.1504   1.0  1525.0005  977.0033   1.0  1209.4730  737.8201
```

```
1.0  1257.4989  810.8962    1.0   672.1960  561.2015   1.0  1125.0356  810.6772
1.0  2051.1789 1067.9541    1.0   923.3977  728.3997   1.0  1827.4010  983.0009
1.0  1466.3330 1049.8788    1.0   472.3215  372.3186   1.0  1014.1540  708.8968
1.0   730.0989  522.7012    1.0   590.7601  361.5210   1.0   880.3944  633.1200
1.0  2432.3910 1424.8047    1.0   831.7983  620.8006   1.0   873.7375  631.7982
1.0   940.9218  517.9196    1.0  1139.4945  819.9964   1.0   951.4432  608.6419
1.0  1177.8547  830.9586    1.0   507.5169  360.8780   1.0   473.0022  300.9999
1.0  1222.5939  925.5795    1.0   576.1972  395.7608   1.0   601.0030  377.9984
1.0  1519.5811 1162.0024    1.0   696.5991  442.0001   1.0   713.9979  397.0015
1.0   687.6638  383.4580    1.0   650.8180  404.0384   1.0   829.2984  588.5195
1.0   953.1192  621.1173    1.0   949.5802  670.7993   1.0   959.7953  681.7616
1.0   953.1192  621.1173    1.0   497.1193  297.5702   1.0  1212.9613  807.3603
1.0   953.1192  621.1173    1.0   570.1674  353.4882   1.0   958.8743  696.8011
1.0   939.0418  548.6002    1.0   724.7306  383.9376   1.0  1129.4431  811.1962
1.0  1283.4025  745.2353    1.0   408.3399  284.8008   1.0  1943.0419 1305.7201
1.0  1511.5789  837.8005    1.0   638.6713  431.1000   1.0   539.6388  442.0001
1.0  1342.5821  795.3402    1.0  1225.7890  801.3518   1.0   463.5990  353.6013
1.0   511.7980  418.5976    1.0   715.3701  448.4513   1.0   562.6400  468.0008
1.0   689.7988  508.7974    1.0   800.4708  577.9111   1.0   736.7584  526.7573
1.0  1532.3074  883.2780    1.0   975.5974  570.5210   1.0  1415.4461  890.2390
1.0  1056.0808  742.5276    1.0  1613.7565  865.3205   1.0  2208.7897 1318.8033
1.0   387.3195  242.3202    1.0   608.5019  444.5578   1.0   636.0009  331.0005
1.0   387.3195  242.3202    1.0   958.6634  680.4198   1.0   759.4010  416.4015
1.0   410.9987  266.0010    1.0   835.9426  576.2779   1.0  1078.8382  596.8406
1.0   499.7510  408.4992    1.0  1024.8177  708.4787   1.0   748.6413  429.0399
1.0   832.7554  614.7588    1.0  1006.4353  734.2356   1.0   987.6417  619.6408
1.0   614.9986  385.3184    1.0   726.0000  433.0010   1.0   788.0961  400.7990
1.0   887.4658  515.6200    1.0   494.4174  327.4188   1.0  1020.0225  775.0209
1.0  1595.1611 1138.1620    1.0   776.5958  485.5198   1.0  1230.9235  772.7611
1.0  1807.9520  993.9630    1.0   415.4407  305.4390   1.0   440.5174  306.5191
1.0   541.2006  299.1993    1.0   581.3599  468.0008   1.0   743.0772  522.6019
1.0  1057.6767  750.3202                                   :: (x[1..m],y)[1..n]
0.10    0.25      0.50      0.75    0.90                   :: tau[1..ntau]
```

10.3 Program Results

```
nag_regsn_quant_linear_iid (g02qfc) Example Program Results

 Quantile:  0.100

        Lower   Parameter   Upper
        Limit   Estimate    Limit
   1    74.946   110.142   145.337
   2     0.370     0.402     0.433


 Quantile:  0.250

        Lower   Parameter   Upper
        Limit   Estimate    Limit
   1    64.232    95.483   126.735
   2     0.446     0.474     0.502


 Quantile:  0.500

        Lower   Parameter   Upper
        Limit   Estimate    Limit
   1    55.399    81.482   107.566
   2     0.537     0.560     0.584


 Quantile:  0.750

        Lower   Parameter   Upper
        Limit   Estimate    Limit
   1    41.372    62.396    83.421
   2     0.625     0.644     0.663
```

```
 Quantile:  0.900

         Lower   Parameter   Upper
         Limit   Estimate    Limit
   1    26.829    67.351   107.873
   2     0.650     0.686     0.723
```