1 Purpose

nag_tsa_multi_part_regsn (g13dpc) calculates the sample partial autoregression matrices of a multivariate time series. A set of likelihood ratio statistics and their significance levels are also returned. These quantities are useful for determining whether the series follows an autoregressive model and, if so, of what order.

2 Specification

```c
void nag_tsa_multi_part_regsn (Integer k, Integer n, const double z[], Integer m, Integer *maxlag, double parlag[], double se[], double qq[], double x[], double pvalue[], double loglhd[], NagError *fail)
```

3 Description

Let \( W_t = (w_{1t}, w_{2t}, \ldots, w_{kt})^T \), for \( t = 1, 2, \ldots, n \), denote a vector of \( k \) time series. The partial autoregression matrix at lag \( l \), \( P_l \), is defined to be the last matrix coefficient when a vector autoregressive model of order \( l \) is fitted to the series. \( P_l \) has the property that if \( W_t \) follows a vector autoregressive model of order \( p \) then \( P_l = 0 \) for \( l > p \).

Sample estimates of the partial autoregression matrices may be obtained by fitting autoregressive models of successively higher orders by multivariate least squares; see Tiao and Box (1981) and Wei (1990). These models are fitted using a QR algorithm based on the routines nag_regsn_mult_linear_addrem_obs (g02dcc) and nag_regsn_mult_linear_delete_var (g02dfc). They are calculated up to lag \( m \), which is usually taken to be at most \( n/4 \).

The routine also returns the asymptotic standard errors of the elements of \( \hat{P}_l \) and an estimate of the residual variance-covariance matrix \( \hat{\Sigma}_l \), for \( l = 1, 2, \ldots, m \). If \( S_l \) denotes the residual sum of squares and cross products matrix after fitting an \( AR(l) \) model to the series then under the null hypothesis \( H_0 : P_l = 0 \) the test statistic

\[
X_l = -((n - m - 1) - \frac{1}{2} - lk) \log \left( \frac{|S_l|}{|S_{l-1}|} \right)
\]

is asymptotically distributed as \( \chi^2 \) with \( k^2 \) degrees of freedom. \( X_l \) provides a useful diagnostic aid in determining the order of an autoregressive model. (Note that \( \hat{\Sigma}_l = S_l/(n - l) \).) The routine also returns an estimate of the maximum of the log-likelihood function for each AR model that has been fitted.

4 References


5 Parameters

1: \( k \) – Integer  
   \( \text{Input} \)

   \( \text{On entry:} \) the number of time series, \( k \).

   \( \text{Constraint:} \; k \geq 1. \)
g13dpc

NAG C Library Manual

2: n – Integer

On entry: the number of observations in the time series, n.

Constraint: n \geq 4.

3: z[dim] – const double

Note: the dimension, dim, of the array z must be at least k \times n.

On entry: z(t(i - 1)k + i - 1) must contain the value for the ith series at time t, for i = 1, 2, \ldots, k;
\quad t = 1, 2, \ldots, n.

4: m – Integer

On entry: the number, m, of partial autoregression matrices to be computed. If in doubt set m = 10.

Constraint: m \geq 1 and n - m - (k \times m + 1) \geq k.

5: maxlag – Integer *

On exit: the maximum lag up to which partial autoregression matrices (along with their likelihood ratio statistics and their significance levels) have been successfully computed. On a successful exit maxlag will equal m. If fail.code = MATRIX_ILL_CONDITIONED on exit then maxlag will be less than m.

6: parlag[dim] – double

Note: the dimension, dim, of the array parlag must be at least k \times k \times m.

On exit: parlag(l - 1)k^2 + (j - 1)k + i - 1 contains an estimate of the (i, j)th element of the partial autoregression matrix at lag l, for l = 1, 2, \ldots, maxlag; i = 1, 2, \ldots, k; j = 1, 2, \ldots, k.

7: se[dim] – double

Note: the dimension, dim, of the array se must be at least k \times k \times m.

On exit: se(l - 1)k^2 + (j - 1)k + i - 1 contains an estimate of the standard error of the corresponding element in parlag.

8: qq[dim] – double

Note: the dimension, dim, of the array qq must be at least k \times k \times m.

On exit: qq(l - 1)k^2 + (j - 1)k + i - 1 contains an estimate of the (i, j)th element of the residual variance-covariance matrix, \hat{\Sigma}_l, for l = 1, 2, \ldots, maxlag; i = 1, 2, \ldots, k; j = 1, 2, \ldots, k.

9: x[m] – double

On exit: x[l - 1] contains X_l, the likelihood ratio statistic at lag l, for l = 1, 2, \ldots, maxlag.

10: pvalue[m] – double

On exit: pvalue[l - 1] contains the significance level of the statistic in the corresponding element of x.

11: loglhd[m] – double

On exit: loglhd[l - 1] contains an estimate of the maximum of the log-likelihood function when an
\quad AR(l - 1) model has been fitted to the series for l = 1, 2, \ldots, maxlag.

12: fail – NagError *

The NAG error parameter (see the Essential Introduction).
6 Error Indicators and Warnings

**NE_INT**

On entry, \( k = \langle \text{value} \rangle \).
Constraint: \( k \geq 1 \).

On entry, \( m = \langle \text{value} \rangle \).
Constraint: \( m \geq 1 \).

On entry, \( n = \langle \text{value} \rangle \).
Constraint: \( n \geq 4 \).

**NE_INT_3**

On entry, \( n - m - (k \times m + 1) < k: k = \langle \text{value} \rangle, m = \langle \text{value} \rangle \) and \( n = \langle \text{value} \rangle \).

**MATRIX_ILL_CONDITIONED**

The recursive equations used to compute the partial autoregression matrices are ill-conditioned.
They have been computed up to lag \( \langle \text{value} \rangle \).

**NE_ALLOC_FAIL**

Memory allocation failed.

**NE_BAD_PARAM**

On entry, parameter \( \langle \text{value} \rangle \) had an illegal value.

**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 consult NAG for assistance.

7 Accuracy

The computations are believed to be stable.

8 Further Comments

The time taken is roughly proportional to \( nmk \).

For each order of autoregressive model that has been estimated, nag_tsa_multi_part_regsn (g13dpc) returns the maximum of the log-likelihood function. An alternative means of choosing the order of a vector AR process is to choose the order for which Akaike's information criterion is smallest. That is, choose the value of \( l \) for which \(-2 \times \logh_{ld[l]} + 2lk^2\) is smallest. The user should be warned that this does not always lead to the same choice of \( l \) as indicated by the sample partial autoregression matrices and the likelihood ratio statistics.

9 Example

A program to compute the sample partial autoregression matrices of two time series of length 48 up to lag 10.
9.1 Program Text

/* nag_tsa_multi_part_regsn (g13dpc) Example Program. 
* Copyright 2002 Numerical Algorithms Group.
* Mark 7, 2002.
*/

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagg13.h>

static void zprint(Integer, Integer, double *, double *, double *, double *, double *);

int main(void)
{
    /* Scalars */
    Integer exit_status, i, j, k, m, maxlag, n, tl, pdz;
    NagError fail;

    /* Arrays */
    double *loglhd = 0, *parlag = 0, *pvalue = 0, *qq = 0, *se = 0, *z = 0, *x = 0;
    #define Z(I,J) z[(J-1)*pdz+I-1]

    INIT_FAIL(fail);
    exit_status = 0;

    Vprintf("g13dpc Example Program Results\n");
    exit_status = 0;

    Vscanf("%ld%ld%ld%*[\n] ", &k, &n, &m);
    if (k > 0 && n >= 1 && m >= 1)
    {
        /* Allocate arrays */
        tl = m*k+1;
        if (!loglhd || !parlag || !pvalue || !qq || !se || !z || !x)
        {
            Vprintf("Allocation failure\n");
            exit_status = -1;
            goto END;
        }
        pdz = k;

        for (i = 1; i <= k; ++i)
        {
            for (j = 1; j <= n; ++j)
            {
                Vscanf("%lf", &Z(i,j));
                Vscanf("%*[\n] ");
            }
            gl3dpc(k, n, z, m, &maxlag, parlag, se, qq, x, pvalue, loglhd, &fail);
            if (fail.code != NE_NOERROR)
            {
                Vprintf("Error from gl3dpc.\n\n", fail.message);
                exit_status = 1;
            }
        }
    }

END:

}
goto END;
}
zprint(k, maxlag, parlag, se, qq, x, pvalue);
}

END:
if (loglhd) NAG_FREE(loglhd);
if (parlag) NAG_FREE(parlag);
if (pvalue) NAG_FREE(pvalue);
if (qq) NAG_FREE(qq);
if (se) NAG_FREE(se);
if (z) NAG_FREE(z);
if (x) NAG_FREE(x);
return exit_status;
}

static void zprint(Integer k, Integer maxlag, double *parlag,
                    double *se, double *qq, double *x, double *pvalue)
{
    /* Scalars */
    double sum;
    Integer i2, i, j, lf;
    /* Arrays */
    char st[7];
    #define SE(I,J,K) se[((K-1)*k + (J-1))*k+I-1]
    #define QQ(I,J,K) qq[(((K-1)*k + (J-1))*k+I-1]
    #define PARLAG(I,J,K) parlag[(((K-1)*k + (J-1))*k+I-1]
    if (k > 1)
    {
        Vprintf("\n");
        Vprintf("Partial Autoregression Matrices Indicator"
                " Residual Chi-Square Pvalue\n");
        Vprintf(" Symbols"
                " Variances Statistic\n");
        Vprintf(" ------------------------------- ---------
                ------- -------- ------");
    }
    if (k == 1)
    {
        Vprintf("\n");
        Vprintf("Partial Autoregression Function Indicator"
                " Residual Chi-Square Pvalue\n");
        Vprintf(" Symbols"
                " Variances\n");
        Vprintf(" " " "
                " --------- -------- ------");
    }
    for (lf = 1; lf <= maxlag; ++lf)
    {
        for (j = 1; j <= k; ++j)
        {
            sum = PARLAG(1,j,lf);
            st[j] = '.';
            if (sum > SE(1,j,lf)*1.96)
                st[j] = '+';
            if (sum < SE(1,j,lf)*-1.96)
                st[j] = '-';
        }
        if (k == 1)
        {
            Vprintf("\n");
            Vprintf(" Lag %2ld :", lf);
            for (j = 1; j <= k; ++j)
for (i2 = 1; i2 <= k; ++i2)
  Vprintf("%c", st[i2]);
Vprintf("%14.3f%13.3f%9.3f\n", QQ(1,1,lf), x[lf-1], pvalue[lf-1]);
Vprintf("\n");
  for (j = 1; j <= k; ++j)
    Vprintf("(%6.3f ) ", SE(1,j,lf));
Vprintf("\n");
else if (k == 2)
{
  Vprintf("\n");
  Vprintf(" Lag %2ld :", lf);
  for (j = 1; j <= k; ++j)
    Vprintf("%8.3f", PARLAG(1,j,lf));
Vprintf("\n");
  for (i2 = 1; i2 <= k; ++i2)
    Vprintf("%c", st[i2]);
Vprintf("%13.3f %12.3f%8.3f\n", QQ(1,1,lf), x[lf-1], pvalue[lf-1]);
Vprintf("\n");
  for (j = 1; j <= k; ++j)
    Vprintf("(%5.3f) ", SE(1,j,lf));
Vprintf("\n");
}
else if (k == 3)
{
  Vprintf("\n");
  Vprintf(" Lag %2ld ", 1f);
  for (j = 1; j <= k; ++j)
    Vprintf("%8.3f", PARLAG(1,j,lf));
Vprintf("\n");
  for (i2 = 1; i2 <= k; ++i2)
    Vprintf("%c", st[i2]);
Vprintf("%12.3f%13.3f%9.3f\n", QQ(1,1,lf), x[lf-1], pvalue[lf-1]);
Vprintf("\n");
  for (j = 1; j <= k; ++j)
    Vprintf("(%5.3f) ", SE(1,j,lf));
Vprintf("\n");
}
else if (k == 4)
{
  Vprintf("\n");
  Vprintf(" Lag %2ld\n", 1f);
  for (j = 1; j <= k; ++j)
    Vprintf("%8.3f ", PARLAG(1,j,lf));
Vprintf("\n");
  for (i2 = 1; i2 <= k; ++i2)
    Vprintf("%c", st[i2]);
Vprintf("%12.3f%13.3f%9.3f\n", QQ(1,1,lf), x[lf-1], pvalue[lf-1]);
Vprintf("\n");
  for (j = 1; j <= k; ++j)
    Vprintf("(%5.3f) ", SE(1,j,lf));
Vprintf("\n");
}
for (i = 2; i <= k; ++i)
{
  for (j = 1; j <= k; ++j)
  {
    sum = PARLAG(i,j,lf);
    st[j] = '.';
    if (sum > SE(i,j,lf) * 1.96)
      st[j] = '+';
    if (sum < SE(i,j,lf) * -1.96)
      st[j] = '-';
  }
if (k == 2)
{
  Vprintf(" ");
  for (j = 1; j <= k; ++j)
Vprintf("%8.3f", PARLAG(i,j,lf));
Vprintf(" ");
for (i2 = 1; i2 <= k; ++i2)
    Vprintf("%c", st[i2]);
Vprintf("%13.3f\n", QQ(i,i,lf));
Vprintf(" ");
for (j = 1; j <= k; ++j)
    Vprintf("(%5.3f) ", SE(i,j,lf));
Vprintf("\n");
}
else if (k == 3)
{
    Vprintf(" ");
    for (j = 1; j <= k; ++j)
        Vprintf("%8.3f ", PARLAG(i,j,lf));
    for (i2 = 1; i2 <= k; ++i2)
        Vprintf("%c", st[i2]);
    Vprintf("%12.3f\n", QQ(i,i,lf));
    Vprintf(" ");
    for (j = 1; j <= k; ++j)
        Vprintf("(%5.3f) ", SE(i,j,lf));
    Vprintf("\n");
}
else if (k == 4)
{
    for (j = 1; j <= k; ++j)
        Vprintf("%8.3f ", PARLAG(i,j,lf));
    for (i2 = 1; i2 <= k; ++i2)
        Vprintf("%c", st[i2]);
    Vprintf("%12.3f\n", QQ(i,i,lf));
    Vprintf(" ");
    for (j = 1; j <= k; ++j)
        Vprintf("(%5.3f) ", SE(i,j,lf));
    Vprintf("\n");
}
}
return;

9.2 Program Data
g13dpc Example Program Data
2 48 10 : k, no. of series, n, no. of obs in each series, m, no. of lags
2.620 1.490 1.170 0.850 -0.350 0.240 2.440 2.580
3.450 1.650 1.290 4.090 6.320 5.000 4.780 4.110
7.290 7.840 7.550 7.320 7.970 7.760 7.000 8.350
4.080 5.060 4.940 6.650 7.940 10.760 11.890 5.850

9.3 Program Results
g13dpc Example Program Results

<table>
<thead>
<tr>
<th>Partial Autoregression Matrices</th>
<th>Indicator Symbols</th>
<th>Residual Variances</th>
<th>Chi-Square Statistic</th>
<th>Pvalue</th>
</tr>
</thead>
<tbody>
<tr>
<td>--------------------------------</td>
<td>------------------</td>
<td>--------------------</td>
<td>----------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Lag 1 : 0.757 0.062</td>
<td>+, 2.731 49.884 0.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(0.092) (0.092)</td>
<td>(0.061) 0.570 + 5.440</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(0.129) (0.130)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lag</td>
<td>Value 1</td>
<td>Value 2</td>
<td>Value 3</td>
<td>Value 4</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>2</td>
<td>-0.161</td>
<td>-0.135</td>
<td>2.530</td>
<td>3.347</td>
</tr>
<tr>
<td></td>
<td>(0.145)</td>
<td>(0.109)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.237</td>
<td>0.044</td>
<td>1.755</td>
<td>13.962</td>
</tr>
<tr>
<td></td>
<td>(0.128)</td>
<td>(0.095)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-0.098</td>
<td>0.152</td>
<td>0.047</td>
<td>5.486</td>
</tr>
<tr>
<td></td>
<td>(0.134)</td>
<td>(0.099)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.257</td>
<td>-0.026</td>
<td>0.237</td>
<td>5.184</td>
</tr>
<tr>
<td></td>
<td>(0.141)</td>
<td>(0.106)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-0.075</td>
<td>0.112</td>
<td>0.402</td>
<td>4.786</td>
</tr>
<tr>
<td></td>
<td>(0.156)</td>
<td>(0.111)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>-0.054</td>
<td>0.097</td>
<td>0.574</td>
<td>3.838</td>
</tr>
<tr>
<td></td>
<td>(0.166)</td>
<td>(0.121)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.147</td>
<td>0.041</td>
<td>0.147</td>
<td>10.991</td>
</tr>
<tr>
<td></td>
<td>(0.188)</td>
<td>(0.128)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>-0.039</td>
<td>0.099</td>
<td>1.322</td>
<td>3.936</td>
</tr>
<tr>
<td></td>
<td>(0.251)</td>
<td>(0.140)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.189</td>
<td>0.131</td>
<td>1.206</td>
<td>3.175</td>
</tr>
<tr>
<td></td>
<td>(0.275)</td>
<td>(0.157)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-0.183</td>
<td>-0.040</td>
<td>2.201</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.371)</td>
<td>(0.212)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>