

## NAG Library Function Document

### nag\_tsa\_multi\_part\_lag\_corr (g13dnc)

## 1 Purpose

nag\_tsa\_multi\_part\_lag\_corr (g13dnc) calculates the sample partial lag correlation matrices of a multivariate time series. A set of  $\chi^2$ -statistics and their significance levels are also returned. A call to nag\_tsa\_multi\_cross\_corr (g13dmc) is usually made prior to calling this function in order to calculate the sample cross-correlation matrices.

## 2 Specification

```
#include <nag.h>
#include <nagg13.h>

void nag_tsa_multi_part_lag_corr (Integer k, Integer n, Integer m,
    const double r0[], const double r[], Integer *maxlag, double parlag[],
    double x[], double pvalue[], NagError *fail)
```

## 3 Description

Let  $W_t = (w_{1t}, w_{2t}, \dots, w_{kt})^T$ , for  $t = 1, 2, \dots, n$ , denote  $n$  observations of a vector of  $k$  time series. The partial lag correlation matrix at lag  $l$ ,  $P(l)$ , is defined to be the correlation matrix between  $W_t$  and  $W_{t+l}$ , after removing the linear dependence on each of the intervening vectors  $W_{t+1}, W_{t+2}, \dots, W_{t+l-1}$ . It is the correlation matrix between the residual vectors resulting from the regression of  $W_{t+l}$  on the carriers  $W_{t+l-1}, \dots, W_{t+1}$  and the regression of  $W_t$  on the same set of carriers; see Heyse and Wei (1985).

$P(l)$  has the following properties.

- (i) If  $W_t$  follows a vector autoregressive model of order  $p$ , then  $P(l) = 0$  for  $l > p$ ;
- (ii) When  $k = 1$ ,  $P(l)$  reduces to the univariate partial autocorrelation at lag  $l$ ;
- (iii) Each element of  $P(l)$  is a properly normalized correlation coefficient;
- (iv) When  $l = 1$ ,  $P(l)$  is equal to the cross-correlation matrix at lag 1 (a natural property which also holds for the univariate partial autocorrelation function).

Sample estimates of the partial lag correlation matrices may be obtained using the recursive algorithm described in Wei (1990). They are calculated up to lag  $m$ , which is usually taken to be at most  $n/4$ . Only the sample cross-correlation matrices ( $\hat{R}(l)$ , for  $l = 0, 1, \dots, m$ ) and the standard deviations of the series are required as input to nag\_tsa\_multi\_part\_lag\_corr (g13dnc). These may be computed by nag\_tsa\_multi\_cross\_corr (g13dmc). Under the hypothesis that  $W_t$  follows an autoregressive model of order  $s - 1$ , the elements of the sample partial lag matrix  $\hat{P}(s)$ , denoted by  $\hat{P}_{ij}(s)$ , are asymptotically Normally distributed with mean zero and variance  $1/n$ . In addition the statistic

$$X(s) = n \sum_{i=1}^k \sum_{j=1}^k \hat{P}_{ij}(s)^2$$

has an asymptotic  $\chi^2$ -distribution with  $k^2$  degrees of freedom. These quantities,  $X(l)$ , are useful as a diagnostic aid for determining whether the series follows an autoregressive model and, if so, of what order.

## 4 References

Heyse J F and Wei W W S (1985) The partial lag autocorrelation function *Technical Report No. 32* Department of Statistics, Temple University, Philadelphia

Wei W W S (1990) *Time Series Analysis: Univariate and Multivariate Methods* Addison–Wesley

## 5 Arguments

- 1:    **k** – Integer *Input*  
*On entry:*  $k$ , the dimension of the multivariate time series.  
*Constraint:*  $k \geq 1$ .
  
- 2:    **n** – Integer *Input*  
*On entry:*  $n$ , the number of observations in each series.  
*Constraint:*  $n \geq 2$ .
  
- 3:    **m** – Integer *Input*  
*On entry:*  $m$ , the number of partial lag correlation matrices to be computed. Note this also specifies the number of sample cross-correlation matrices that must be contained in the array **r**.  
*Constraint:*  $1 \leq m < n$ .
  
- 4:    **r0**[ $k \times k$ ] – const double *Input*  
*On entry:* the sample cross-correlations at lag zero/standard deviations as provided by nag\_tsa\_multi\_cross\_corr (g13dmc), that is, **r0**[( $j-1$ ) $k+i-1$ ] must contain the ( $i,j$ )th element of the sample cross-correlation matrix at lag zero if  $i \neq j$  and the standard deviation of  $i = j$ , for  $i = 1, 2, \dots, k$  and  $j = 1, 2, \dots, k$ .
  
- 5:    **r**[ $k \times k \times m$ ] – const double *Input*  
*On entry:* the sample cross-correlations as provided by nag\_tsa\_multi\_cross\_corr (g13dmc), that is, **r**[( $l-1$ ) $k^2 + (j-1)k + i - 1$ ] must contain the ( $i,j$ )th element of the sample cross-correlation at lag  $l$ , for  $l = 1, 2, \dots, m$ ,  $i = 1, 2, \dots, k$  and  $j = 1, 2, \dots, k$ , where series  $j$  leads series  $i$ .
  
- 6:    **maxlag** – Integer \* *Output*  
*On exit:* the maximum lag up to which partial lag correlation matrices (along with  $\chi^2$ -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**.
  
- 7:    **parlag**[ $k \times k \times m$ ] – double *Input/Output*  
*On exit:* **parlag**[( $l-1$ ) $k^2 + (j-1)k + i - 1$ ] contains the ( $i,j$ )th element of the sample partial lag correlation matrix at lag  $l$ , for  $l = 1, 2, \dots, \mathbf{maxlag}$ ,  $i = 1, 2, \dots, k$  and  $j = 1, 2, \dots, k$ .
  
- 8:    **x**[**m**] – double *Output*  
*On exit:* **x**[ $l-1$ ] contains the  $\chi^2$ -statistic at lag  $l$ , for  $l = 1, 2, \dots, \mathbf{maxlag}$ .
  
- 9:    **pvalue**[**m**] – double *Output*  
*On exit:* **pvalue**[ $l-1$ ] contains the significance level of the corresponding  $\chi^2$ -statistic in **x**, for  $l = 1, 2, \dots, \mathbf{maxlag}$ .

10: **fail** – NagError \*

*Input/Output*

The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

### MATRIX\_ILL\_CONDITIONED

The recursive equations used to compute the partial lag correlation matrices are ill-conditioned (they have been computed up to lag  $\langle value \rangle$ ).

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.

See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

### NE\_BAD\_PARAM

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

### NE\_INT

On entry,  $\mathbf{k} = \langle value \rangle$ .

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

On entry,  $\mathbf{m} = \langle value \rangle$ .

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

On entry,  $\mathbf{n} = \langle value \rangle$ .

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

### NE\_INT\_2

On entry,  $\mathbf{m} = \langle value \rangle$  and  $\mathbf{n} = \langle value \rangle$ .

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

### 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.

An unexpected error has been triggered by this function. Please contact NAG.

See Section 2.7.6 in How to Use the NAG Library and its Documentation for further information.

### NE\_NO\_LICENCE

Your licence key may have expired or may not have been installed correctly.

See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

## 7 Accuracy

The accuracy will depend upon the accuracy of the sample cross-correlations.

## 8 Parallelism and Performance

nag\_tsa\_multi\_part\_lag\_corr (g13dnc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag\_tsa\_multi\_part\_lag\_corr (g13dnc) 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 x06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

## 9 Further Comments

The time taken is roughly proportional to  $m^2 k^3$ .

If you have calculated the sample cross-correlation matrices in the arrays **r0** and **r**, without calling `nag_tsa_multi_cross_corr` (g13dmc), then care must be taken to ensure they are supplied as described in Section 5. In particular, for  $l \geq 1$ ,  $\hat{R}_{ij}(l)$  must contain the sample cross-correlation coefficient between  $w_{i(t-l)}$  and  $w_{jt}$ .

The function `nag_tsa_multi_auto_corr_part` (g13dbc) computes squared partial autocorrelations for a specified number of lags. It may also be used to estimate a sequence of partial autoregression matrices at lags  $1, 2, \dots$  by making repeated calls to the function with the argument **nk** set to  $1, 2, \dots$ . The  $(i, j)$ th element of the sample partial autoregression matrix at lag  $l$  is given by  $W(i, j, l)$  when **nk** is set equal to  $l$  on entry to `nag_tsa_multi_auto_corr_part` (g13dbc). Note that this is the ‘Yule–Walker’ estimate. Unlike the partial lag correlation matrices computed by `nag_tsa_multi_part_lag_corr` (g13dnc), when  $W_t$  follows an autoregressive model of order  $s - 1$ , the elements of the sample partial autoregressive matrix at lag  $s$  do not have variance  $1/n$ , making it very difficult to spot a possible cut-off point. The differences between these matrices are discussed further by Wei (1990).

Note that `nag_tsa_multi_auto_corr_part` (g13dbc) takes the sample cross-covariance matrices as input whereas this function requires the sample cross-correlation matrices to be input.

## 10 Example

This example computes the sample partial lag correlation matrices of two time series of length 48, up to lag 10. The matrices, their  $\chi^2$ -statistics and significance levels and a plot of symbols indicating which elements of the sample partial lag correlation matrices are significant are printed. Three \* represent significance at the 0.5% level, two \* represent significance at the 1% level and a single \* represents significance at the 5% level. The \* are plotted above or below the central line depending on whether the elements are significant in a positive or negative direction.

### 10.1 Program Text

```
/* nag_tsa_multi_part_lag_corr (g13dnc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

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

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

int main(void)
{
    /* Scalars */
    Integer exit_status, i, j, k, m, maxlag, n, kmax;
    /* Arrays */
    double *parlag = 0, *pvalue = 0, *r0 = 0, *r = 0, *w = 0, *wmean = 0;
    double *x = 0;
    Nag_CovOrCorr matrix;
```

```

NagError fail;

#define W(I, J) w[(J-1)*kmax + I - 1]

INIT_FAIL(fail);

exit_status = 0;

printf("nag_tsa_multi_part_lag_corr (g13dnc) Example Program Results\n");

/* Skip heading in data file */
#ifdef _WIN32
    scanf_s("%*[\n] ");
#else
    scanf("%*[\n] ");
#endif

#ifdef _WIN32
    scanf_s("%" NAG_IFMT "%" NAG_IFMT "%" NAG_IFMT "%*[\n] ", &k, &n, &m);
#else
    scanf("%" NAG_IFMT "%" NAG_IFMT "%" NAG_IFMT "%*[\n] ", &k, &n, &m);
#endif

if (k > 0 && n >= 1 && m >= 1) {
    /* Allocate arrays */
    if (!(parlag = NAG_ALLOC(k * k * m, double)) ||
        !(pvalue = NAG_ALLOC(m, double)) ||
        !(r0 = NAG_ALLOC(k * k, double)) ||
        !(r = NAG_ALLOC(k * k * m, double)) ||
        !(w = NAG_ALLOC(k * n, double)) ||
        !(wmean = NAG_ALLOC(k, double)) || !(x = NAG_ALLOC(m, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

    kmax = k;

    for (i = 1; i <= k; ++i) {
        for (j = 1; j <= n; ++j)
#ifdef _WIN32
            scanf_s("%lf", &W(i, j));
#else
            scanf("%lf", &W(i, j));
#endif
    }
#ifdef _WIN32
    scanf_s("%*[\n] ");
#else
    scanf("%*[\n] ");
#endif
}

matrix = Nag_AutoCorr;

/* nag_tsa_multi_cross_corr (g13dmc).
 * Multivariate time series, sample cross-correlation or
 * cross-covariance matrices
 */
nag_tsa_multi_cross_corr(matrix, k, n, m, w, wmean, r0, r, &fail);
if (fail.code != NE_NOERROR) {
    printf("Error from nag_tsa_multi_cross_corr (g13dmc).\n%s\n",
        fail.message);
    exit_status = 1;
    goto END;
}

/* nag_tsa_multi_part_lag_corr (g13dnc).
 * Multivariate time series, sample partial lag correlation
 * matrices, chi^2 statistics and significance levels
 */

```

```

    nag_tsa_multi_part_lag_corr(k, n, m, r0, r, &maxlag, parlag, x, pvalue,
                                &fail);
    if (fail.code != NE_NOERROR) {
        printf("Error from nag_tsa_multi_part_lag_corr (g13dnc).\n%s\n",
               fail.message);
        exit_status = 1;
        goto END;
    }
    zprint(k, n, m, k, parlag, x, pvalue);
}

END:
    NAG_FREE(parlag);
    NAG_FREE(pvalue);
    NAG_FREE(r0);
    NAG_FREE(r);
    NAG_FREE(w);
    NAG_FREE(wmean);
    NAG_FREE(x);

    return exit_status;
}

/* Print the partial lag correlation matrices. */
static void zprint(Integer k, Integer n, Integer m, Integer ik,
                   double *parlag, double *x, double *pvalue)
{
    /* Scalars */
    double c1, c2, c3, c5, c6, c7, cnst, sum;
    Integer i2, i, j, lf, llf, ii, jj;

    /* Arrays */
    char rec[7][80];

#define PARLAG(I, J, K) parlag[((K-1)*ik + (J-1))*ik + I-1]

    cnst = 1.0 / sqrt((double) n);

    printf("\n");
    printf(" PARTIAL LAG CORRELATION MATRICES\n");
    printf(" ----- \n");

    for (lf = 1; lf <= m; ++lf) {
        printf("\n");
        printf(" Lag = %2" NAG_IFMT "\n", lf);
        for (ii = 1; ii <= k; ii++) {
            for (jj = 1; jj <= k; jj++)
                printf("%9.3f", PARLAG(ii, jj, lf));
            printf("\n");
        }
    }

    printf("\n");
    printf(" Standard error = 1 / SQRT(N) = %5.3f\n", cnst);

    /* Print indicator symbols to indicate significant elements. */
    printf("\n");
    printf(" TABLES OF INDICATOR SYMBOLS\n");
    printf(" ----- \n");
    printf("\n");
    printf(" For Lags 1 to %2" NAG_IFMT "\n", m);
    printf("\n");

    /* Set up the critical values */
    c1 = cnst * 3.29;
    c2 = cnst * 2.58;
    c3 = cnst * 1.96;
    c5 = -c3;
    c6 = -c2;
    c7 = -c1;

```

```

for (i = 1; i <= k; ++i) {
    for (j = 1; j <= k; ++j) {
        printf("\n");
        printf("\n");
        if (i == j) {
            printf("Auto-correlation function for series %2" NAG_IFMT "\n", i);
            printf("\n");
        }
        else {
            printf("Cross-correlation function for series %2" NAG_IFMT " "
                " and series%2" NAG_IFMT "\n", i, j);
            printf("\n");
        }
    }

    /* Clear the last plot with blanks */
#ifdef _WIN32
    sprintf_s(&rec[0][0], 80, "          0.005  :");
#else
    sprintf(&rec[0][0], "          0.005  :");
#endif
#ifdef _WIN32
    sprintf_s(&rec[1][0], 80, "          +    0.01  :");
#else
    sprintf(&rec[1][0], "          +    0.01  :");
#endif
#ifdef _WIN32
    sprintf_s(&rec[2][0], 80, "          0.05  :");
#else
    sprintf(&rec[2][0], "          0.05  :");
#endif
#ifdef _WIN32
    sprintf_s(&rec[3][0], 80,
        "      Sig. Level      : - - - - - Lags");
#else
    sprintf(&rec[3][0], "      Sig. Level      : - - - - - Lags");
#endif
#ifdef _WIN32
    sprintf_s(&rec[4][0], 80, "          0.05  :");
#else
    sprintf(&rec[4][0], "          0.05  :");
#endif
#ifdef _WIN32
    sprintf_s(&rec[5][0], 80, "          -    0.01  :");
#else
    sprintf(&rec[5][0], "          -    0.01  :");
#endif
#ifdef _WIN32
    sprintf_s(&rec[6][0], 80, "          0.005  :");
#else
    sprintf(&rec[6][0], "          0.005  :");
#endif
    for (i2 = 0; i2 < 7; ++i2) {
        for (ii = strlen(&rec[i2][0]); ii < 80; ii++)
            rec[i2][ii] = ' ';
    }

    for (lf = 1; lf <= m; ++lf) {
        llf = lf * 2 + 21;
        sum = PARLAG(i, j, lf);
        /* Check for significance */
        if (sum > c1)
            rec[0][llf] = '*';
        if (sum > c2)
            rec[1][llf] = '*';
        if (sum > c3)
            rec[2][llf] = '*';
        if (sum < c5)
            rec[4][llf] = '*';
        if (sum < c6)
            rec[5][llf] = '*';
        if (sum < c7)

```

```

        rec[6][11f] = '*';
    }

    /* Print */
    for (i2 = 0; i2 < 7; ++i2) {
        /* Terminate the string */
        for (ii = 80; ii > 1 && rec[i2][ii - 1] == ' '; ii--);
        rec[i2][ii] = '\\setminus 0';
        /* Print the string */
        printf("%s\\n", &rec[i2][0]);
    }
}

/* Print the chi-square statistics and p-values. */
printf("\\n");
printf(" Lag      Chi-square statistic      P-value\\n");
printf(" ---      -----      -----\\n");
printf("\\n");

for (lf = 1; lf <= m; ++lf)
    printf("%4" NAG_IFMT " %17.3f %18.4f\\n", lf, x[lf - 1], pvalue[lf - 1]);

return;
}

```

## 10.2 Program Data

nag\_tsa\_multi\_part\_lag\_corr (g13dnc) Example Program Data

2 48 10 : k, no. of series, n, no. of obs in each series, m, no. of lags

-1.490	-1.620	5.200	6.230	6.210	5.860	4.090	3.180
2.620	1.490	1.170	0.850	-0.350	0.240	2.440	2.580
2.040	0.400	2.260	3.340	5.090	5.000	4.780	4.110
3.450	1.650	1.290	4.090	6.320	7.500	3.890	1.580
5.210	5.250	4.930	7.380	5.870	5.810	9.680	9.070
7.290	7.840	7.550	7.320	7.970	7.760	7.000	8.350
7.340	6.350	6.960	8.540	6.620	4.970	4.550	4.810
4.750	4.760	10.880	10.010	11.620	10.360	6.400	6.240
7.930	4.040	3.730	5.600	5.350	6.810	8.270	7.680
6.650	6.080	10.250	9.140	17.750	13.300	9.630	6.800
4.080	5.060	4.940	6.650	7.940	10.760	11.890	5.850
9.010	7.500	10.020	10.380	8.150	8.370	10.730	12.145

: End of time series

## 10.3 Program Results

nag\_tsa\_multi\_part\_lag\_corr (g13dnc) Example Program Results

PARTIAL LAG CORRELATION MATRICES  
-----

Lag = 1  
0.736      0.174  
0.211      0.555

Lag = 2  
-0.187    -0.083  
-0.180    -0.072

Lag = 3  
0.278    -0.007  
0.084    -0.213

Lag = 4  
-0.084    0.227  
0.128    -0.176

Lag = 5  
0.236    0.238  
-0.047   -0.046



```
Lag = 6
-0.016    0.087
 0.100   -0.081
```

```
Lag = 7
-0.036    0.261
 0.126    0.012
```

```
Lag = 8
 0.077    0.381
 0.027   -0.149
```

```
Lag = 9
-0.065   -0.387
 0.189    0.057
```

```
Lag = 10
-0.026   -0.286
 0.028   -0.173
```

Standard error = 1 / SQRT(N) = 0.144

#### TABLES OF INDICATOR SYMBOLS

-----

For Lags 1 to 10

#### Auto-correlation function for series 1

```

          0.005 : *
+         0.01 : *
          0.05 : *
Sig. Level : - - - - - Lags
          0.05 :
-         0.01 :
          0.005 :
```

#### Cross-correlation function for series 1 and series 2

```

          0.005 :
+         0.01 :      *
          0.05 :      *
Sig. Level : - - - - - Lags
          0.05 :      * *
-         0.01 :      *
          0.005 :
```

#### Cross-correlation function for series 2 and series 1

```

          0.005 :
+         0.01 :
          0.05 :
Sig. Level : - - - - - Lags
          0.05 :
-         0.01 :
          0.005 :
```

#### Auto-correlation function for series 2

```

          0.005 : *
+         0.01 : *
          0.05 : *
Sig. Level : - - - - - Lags
          0.05 :
-         0.01 :
          0.005 :
```

Lag	Chi-square statistic	P-value
---	-----	-----
1	44.362	0.0000
2	3.824	0.4304
3	6.219	0.1834
4	5.094	0.2778
5	5.609	0.2303
6	1.170	0.8830
7	4.098	0.3929
8	8.371	0.0789
9	9.244	0.0553
10	5.435	0.2455

---