NAG Library Function Document

nag_tsa_multi_auto_corr_part (g13dbc)

1 Purpose

nag_tsa_multi_auto_corr_part (g13dbc) calculates the multivariate partial autocorrelation function of a multivariate time series.

2 Specification

```
#include <nag.h>
#include <nagg13.h>
void nag_tsa_multi_auto_corr_part (const double c0[], const double c[],
     Integer ns, Integer nl, Integer nk, double p[], double *v0, double v[],
     double d[], double db[], double w[], double wb[], Integer *nvp,
     NagError *fail)
```

3 Description

The input is a set of lagged autocovariance matrices $C_0, C_1, C_2, \ldots, C_m$. These will generally be sample values such as are obtained from a multivariate time series using nag_tsa_multi_cross_corr (g13dmc).

The main calculation is the recursive determination of the coefficients in the finite lag (forward) prediction equation

$$x_t = \Phi_{l,1}x_{t-1} + \cdots + \Phi_{l,l}x_{t-l} + e_{l,t}$$

and the associated backward prediction equation

$$x_{t-l-1} = \Psi_{l,1}x_{t-l} + \cdots + \Psi_{l,l}x_{t-1} + f_{l,t}$$

together with the covariance matrices D_l of $e_{l,t}$ and G_l of $f_{l,t}$.

The recursive cycle, by which the order of the prediction equation is extended from l to $l+1$, is to calculate

$$M_{l+1} = C_{l+1}^{\mathrm{T}} - \Phi_{l,1}C_l^{\mathrm{T}} - \cdots - \Phi_{l,l}C_1^{\mathrm{T}} \tag{1}$$

then $\Phi_{l+1,l+1} = M_{l+1}D_l^{-1}, \quad \Psi_{l+1,l+1} = M^{\mathrm{T}}{}_{l+1}G_l^{-1}$

from which

$$\Phi_{l+1,j} = \Phi_{l,j} - \Phi_{l+1,l+1}\Psi_{l,l+1-j}, \quad j = 1, 2, \ldots, l \tag{2}$$

and

$$\Psi_{l+1,j} = \Psi_{l,j} - \Psi_{l+1,l+1}\Phi_{l,l+1-j}, \quad j = 1, 2, \ldots, l. \tag{3}$$

Finally, $D_{l+1} = D_l - M_{l+1}\Phi^{\mathrm{T}}{}_{l+1,l+1}$ and $G_{l+1} = G_l - M^{\mathrm{T}}{}_{l+1}\Psi^{\mathrm{T}}{}_{l+1,l+1}$.

(Here T denotes the transpose of a matrix.)

The cycle is initialized by taking (for $l = 0$)

$$D_0 = G_0 = C_0.$$

In the step from $l = 0$ to 1, the above equations contain redundant terms and simplify. Thus (1) becomes $M_1 = C^{\mathrm{T}}{}_1$ and neither (2) or (3) are needed.

Quantities useful in assessing the effectiveness of the prediction equation are generalized variance ratios

$$v_l = \det D_l / \det C_0, \quad l = 1, 2, \ldots$$

and multiple squared partial autocorrelations

$$p_l^2 = 1 - v_l/v_{l-1}.$$

4 References

Akaike H (1971) Autoregressive model fitting for control *Ann. Inst. Statist. Math.* **23** 163–180

Whittle P (1963) On the fitting of multivariate autoregressions and the approximate canonical factorization of a spectral density matrix *Biometrika* **50** 129–134

5 Arguments

1: **c0**[**ns** × **ns**] – const double *Input*

On entry: contains the zero lag cross-covariances between the **ns** series as returned by nag_tsa_multi_cross_corr (g13dmc). (**c0** is assumed to be symmetric, upper triangle only is used.)

2: **c**[**ns** × **ns** × **nl**] – const double *Input*

On entry: the k cross-covariances as returned by nag_tsa_multi_cross_corr (g13dmc).

3: **ns** – Integer *Input*

On entry: k, the number of time series whose cross-covariances are supplied in **c** and **c0**.

Constraint: **ns** ≥ 1.

4: **nl** – Integer *Input*

On entry: m, the maximum lag for which cross-covariances are supplied in **c**.

Constraint: **nl** ≥ 1.

5: **nk** – Integer *Input*

On entry: the number of lags to which partial auto-correlations are to be calculated.

Constraint: $1 \leq$ **nk** $\leq$ **nl**.

6: **p**[**nk**] – double *Output*

On exit: the multiple squared partial autocorrelations from lags 1 to **nvp**; that is, **p**$[l-1]$ contains p_l^2, for $l = 1, 2, \ldots,$ **nvp**. For lags **nvp** $+ 1$ to **nk** the elements of **p** are set to zero.

7: **v0** – double * *Output*

On exit: the lag zero prediction error variance (equal to the determinant of **c0**).

8: **v**[**nk**] – double *Output*

On exit: the prediction error variance ratios from lags 1 to **nvp**; that is, **v**$[l-1]$ contains v_l, for $l = 1, 2, \ldots,$ **nvp**. For lags **nvp** $+ 1$ to **nk** the elements of **v** are set to zero.

9: **d**[**ns** × **ns** × **nk**] – double *Output*

On exit: the prediction error variance matrices at lags 1 to **nvp**, **d**$[(l-1)k^2 + (j-1)k + i - 1]$ contains the (i, j)th prediction error covariance of series i and series j at lag l. Series j leads series i.

10: **db**[**ns** × **ns**] – double *Output*

On exit: the backward prediction error variance matrix at lag **nvp**, **db**$[(j-1)k+i-1]$ contains the backward prediction error covariance of series i and series j.

11: **w**[**ns** × **ns** × **nk**] – double *Output*

On exit: the prediction coefficient matrices at lags 1 to **nvp**, **w**$[(l-1)k^2+(j-1)k+i-1]$ contains the jth prediction coefficient of series i at lag l (i.e., the (i,j)th element of $\Phi_{L,l}$).

12: **wb**[**ns** × **ns** × **nk**] – double *Output*

On exit: the backward prediction coefficient matrices at lags 1 to **nvp**, **wb**$[(l-1)k^2+(j-1)+i-1]$ contains the jth backward prediction coefficient of series i at lag l (i.e., the (i,j)th element of $\Psi_{L,l}$).

13: **nvp** – Integer * *Output*

On exit: the maximum lag, L, for which calculation of **p**, **v**, **d**, **db**, **w** and **wb** was successful. If the function completes successfully **nvp** will equal **nk**.

14: **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

NE_ALLOC_FAIL

Dynamic memory allocation failed.
See Section 3.2.1.2 in How to Use the NAG Library and its Documentation for further information.

NE_BAD_PARAM

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

NE_INT

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

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

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

NE_INT_2

On entry, **nk** = ⟨*value*⟩ and **nl** = ⟨*value*⟩.
Constraint: **nk** ≤ **nl**.

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 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 3.6.5 in How to Use the NAG Library and its Documentation for further information.

NE_POS_DEF

At lag $\mathbf{nvp}+1 \leq \mathbf{nk}$, **d** is not positive definite, $\mathbf{nvp} = \langle value \rangle$ and $\mathbf{nk} = \langle value \rangle$.

c0 is not positive definite.

7 Accuracy

The conditioning of the problem depends on the prediction error variance ratios. Very small values of these may indicate loss of accuracy in the computations.

8 Parallelism and Performance

nag_tsa_multi_auto_corr_part (g13dbc) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

nag_tsa_multi_auto_corr_part (g13dbc) 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' Notefor your implementation for any additional implementation-specific information.

9 Further Comments

The time taken by nag_tsa_multi_auto_corr_part (g13dbc) is roughly proportional to $\mathbf{nk}^2 \times \mathbf{ns}^3$.

If sample autocorrelation matrices are used as input, then the output will be relevant to the original series scaled by their standard deviations. If these autocorrelation matrices are produced by nag_tsa_multi_cross_corr (g13dmc), you must replace the diagonal elements of C_0 (otherwise used to hold the series variances) by 1.

10 Example

This example reads the autocovariance matrices for four series from lag 0 to 5. It calls nag_tsa_multi_auto_corr_part (g13dbc) to calculate the multivariate partial autocorrelation function and other related matrices of statistics up to lag 3. It prints the results.

10.1 Program Text

```
/* nag_tsa_multi_auto_corr_part (g13dbc) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

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

int main(void)
{
  /* Scalars */
  double v0;
```

```
  Integer exit_status, i1, i, j, j1, k, nk, nl, ns, nvp, pdc0, pddb;
  NagError fail;

  /* Arrays */
  double *c0 = 0, *c = 0, *d = 0, *db = 0, *p = 0, *v = 0, *w = 0, *wb = 0;

#define C(I, J, K)  c[((K-1)*ns + (J-1))*ns + I - 1]
#define D(I, J, K)  d[((K-1)*ns + (J-1))*ns + I - 1]
#define W(I, J, K)  w[((K-1)*ns + (J-1))*ns + I - 1]
#define WB(I, J, K) wb[((K-1)*ns + (J-1))*ns + I - 1]

#ifdef NAG_COLUMN_MAJOR
#define C0(I, J)    c0[(J-1)*pdc0 + I - 1]
#define DB(I, J)    db[(J-1)*pddb + I - 1]
#else
#define C0(I, J)    c0[(I-1)*pdc0 + J - 1]
#define DB(I, J)    db[(I-1)*pddb + J - 1]
#endif

  INIT_FAIL(fail);

  exit_status = 0;

  printf("nag_tsa_multi_auto_corr_part (g13dbc) Example Program Results\n");

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

  /* Read series length, and numbers of lags */
#ifdef _WIN32
  scanf_s("%" NAG_IFMT "%" NAG_IFMT "%" NAG_IFMT "%*[^\n] ", &ns, &nl, &nk);
#else
  scanf("%" NAG_IFMT "%" NAG_IFMT "%" NAG_IFMT "%*[^\n] ", &ns, &nl, &nk);
#endif

  if (ns > 0 && nl > 0 && nk > 0) {
    /* Allocate arrays */
    if (!(c0 = NAG_ALLOC(ns * ns, double)) ||
        !(c = NAG_ALLOC(ns * ns * nl, double)) ||
        !(d = NAG_ALLOC(ns * ns * nk, double)) ||
        !(db = NAG_ALLOC(ns * ns, double)) ||
        !(p = NAG_ALLOC(nk, double)) ||
        !(v = NAG_ALLOC(nk, double)) ||
        !(w = NAG_ALLOC(ns * ns * nk, double)) ||
        !(wb = NAG_ALLOC(ns * ns * nk, double)))
    {
      printf("Allocation failure\n");
      exit_status = -1;
      goto END;
    }

    pdc0 = ns;
    pddb = ns;

    /* Read autocovariances */
    for (i = 1; i <= ns; ++i) {
      for (j = 1; j <= ns; ++j)
#ifdef _WIN32
        scanf_s("%lf", &C0(i, j));
#else
        scanf("%lf", &C0(i, j));
#endif
    }
#ifdef _WIN32
    scanf_s("%*[^\n] ");
#else
    scanf("%*[^\n] ");
```

```
#endif

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

    /* Call routine to calculate multivariate partial
       autocorrelation function */

    /* nag_tsa_multi_auto_corr_part (g13dbc).
     * Multivariate time series, multiple squared partial
     * autocorrelations
     */
    nag_tsa_multi_auto_corr_part(c0, c, ns, nl, nk, p, &v0, v, d, db, w, wb,
                                 &nvp, &fail);
    if (fail.code != NE_NOERROR) {
      printf("Error from nag_tsa_multi_auto_corr_part (g13dbc).\n%s\n",
              fail.message);
      exit_status = 1;
      goto END;
    }

    if (fail.code == NE_NOERROR || fail.code == NE_POS_DEF) {
      printf("\n");
      printf("Number of valid parameters =%10" NAG_IFMT "\n", nvp);

      printf("\n");
      printf("Multivariate partial autocorrelations\n");

      for (i1 = 1; i1 <= nk; ++i1) {
        printf("%13.5f", p[i1 - 1]);
        if (i1 % 5 == 0 || i1 == nk)
          printf("\n");
      }

      printf("\n");
      printf("Zero lag predictor error variance determinant\n");
      printf("followed by error variance ratios\n");
      printf("%12.5f", v0);

      for (i1 = 1; i1 <= nk; ++i1) {
        printf("%13.5f", v[i1 - 1]);
        if (i1 % 5 == 0 || i1 == nk)
          printf("\n");
      }

      printf("\n");
      printf("Prediction error variances\n");
      printf("\n");

      for (k = 1; k <= nk; ++k) {
        printf("Lag =%5" NAG_IFMT "\n", k);
        for (i = 1; i <= ns; ++i) {
          for (j1 = 1; j1 <= ns; ++j1) {
            printf("%13.5f", D(i, j1, k));
            if (j1 % 5 == 0 || j1 == ns)
              printf("\n");
          }
        }
```

```
        printf("\n");
      }

      printf("Last backward prediction error variances\n");
      printf("\n");
      printf("Lag =%5" NAG_IFMT "\n", nvp);

      for (i = 1; i <= ns; ++i) {
        for (j1 = 1; j1 <= ns; ++j1) {
          printf("%13.5f", DB(i, j1));
          if (j1 % 5 == 0 || j1 == ns)
            printf("\n");
        }
      }

      printf("\n");
      printf("Prediction coefficients\n");
      printf("\n");

      for (k = 1; k <= nk; ++k) {
        printf("Lag =%5" NAG_IFMT "\n", k);
        for (i = 1; i <= ns; ++i) {
          for (j1 = 1; j1 <= ns; ++j1) {
            printf("%13.5f", W(i, j1, k));
            if (j1 % 5 == 0 || j1 == ns)
              printf("\n");
          }
        }
        printf("\n");
      }
      printf("Backward prediction coefficients\n");
      printf("\n");

      for (k = 1; k <= nk; ++k) {
        printf("Lag =%5" NAG_IFMT "\n", k);
        for (i = 1; i <= ns; ++i) {
          for (j1 = 1; j1 <= ns; ++j1) {
            printf("%13.5f", WB(i, j1, k));
            if (j1 % 5 == 0 || j1 == ns)
              printf("\n");
          }
        }
        printf("\n");
      }
    }
  }

END:
  NAG_FREE(c0);
  NAG_FREE(c);
  NAG_FREE(d);
  NAG_FREE(db);
  NAG_FREE(p);
  NAG_FREE(v);
  NAG_FREE(w);
  NAG_FREE(wb);

  return exit_status;
}
```

10.2 Program Data

```
nag_tsa_multi_auto_corr_part (g13dbc) Example Program Data
     4     5     3
  .10900E-01 -.77917E-02  .13004E-02  .12654E-02
 -.77917E-02  .57040E-01  .24180E-02  .14409E-01
  .13004E-02  .24180E-02  .43960E-01 -.21421E-01
  .12654E-02  .14409E-01 -.21421E-01  .72289E-01
  .45889E-02  .46510E-03 -.13275E-03  .77531E-02
 -.24419E-02 -.11667E-01 -.21956E-01 -.45803E-02
```

```
  .11080E-02 -.80479E-02  .13621E-01 -.85868E-02
 -.50614E-03  .14045E-01 -.10087E-02  .12269E-01
  .18652E-02 -.64389E-02  .88307E-02 -.24808E-02
 -.11865E-01  .72367E-02 -.19802E-01  .59069E-02
 -.80307E-02  .14306E-01  .14546E-01  .13510E-01
 -.21791E-02 -.29528E-01 -.15887E-01  .88308E-03
 -.80550E-04 -.37759E-02  .75463E-02 -.42276E-02
  .41447E-02 -.37987E-02  .19332E-02 -.17564E-01
 -.10582E-01  .67733E-02  .69832E-02  .61747E-02
  .41352E-02 -.16013E-01  .17043E-01 -.13412E-01
  .76079E-03 -.10134E-02  .11870E-01 -.41651E-02
  .36014E-02 -.36375E-02 -.25571E-01  .50218E-02
 -.13924E-01  .11718E-01 -.59088E-02  .59297E-02
  .10739E-01 -.14571E-01  .13816E-01 -.12588E-01
 -.64365E-03 -.44556E-02  .51334E-02  .71587E-03
  .63617E-02  .15217E-03  .27270E-02 -.22261E-02
 -.85855E-02  .14468E-02 -.28698E-02  .44384E-02
  .68339E-02 -.21790E-02  .13759E-01  .28217E-03
```

10.3 Program Results

```
nag_tsa_multi_auto_corr_part (g13dbc) Example Program Results

Number of valid parameters =         3

Multivariate partial autocorrelations
      0.64498      0.92669      0.84300

Zero lag predictor error variance determinant
followed by error variance ratios
     0.00000      0.35502      0.02603      0.00409

Prediction error variances

Lag =    1
      0.00811     -0.00511      0.00159     -0.00029
     -0.00511      0.04089      0.00757      0.01843
      0.00159      0.00757      0.03834     -0.01894
     -0.00029      0.01843     -0.01894      0.06760

Lag =    2
      0.00354     -0.00087     -0.00075     -0.00105
     -0.00087      0.01946      0.00535      0.00566
     -0.00075      0.00535      0.01900     -0.01071
     -0.00105      0.00566     -0.01071      0.04058

Lag =    3
      0.00301     -0.00087     -0.00054      0.00065
     -0.00087      0.01824      0.00872      0.00247
     -0.00054      0.00872      0.00935     -0.00216
      0.00065      0.00247     -0.00216      0.02254

Last backward prediction error variances

Lag =    3
      0.00331     -0.00392     -0.00106      0.00592
     -0.00392      0.01890      0.00348     -0.00330
     -0.00106      0.00348      0.01003     -0.01054
      0.00592     -0.00330     -0.01054      0.03336

Prediction coefficients

Lag =    1
      0.81861      0.23399     -0.17097      0.09256
      0.06738     -0.48720     -0.14064      0.04295
      0.15036      0.11924     -0.36725     -0.42092
     -0.70971      0.02998      0.59779      0.34610

Lag =    2
     -0.34049     -0.13370      0.40610     -0.02183
```

```
     -1.27574     -0.13591     -0.65779     -0.11267
     -0.45439      0.19379      0.63420      0.33920
     -0.43237     -0.54848     -0.62897      0.16670

Lag =    3
      0.16437      0.13858      0.01290      0.03463
      0.39291      0.07407     -0.08802     -0.15361
     -1.29240     -0.24489      0.30235      0.39442
      0.89768     -0.39040      0.25151     -0.28304

Backward prediction coefficients

Lag =    1
      0.41541      0.06149      0.15319      0.05079
      0.12370     -0.26471     -0.22721      0.48503
     -0.86933     -0.47373      0.37924      0.13814
      1.30779     -0.09178     -1.45398     -0.21967

Lag =    2
     -0.06740     -0.12255     -0.13673     -0.09730
     -1.24801      0.03090      0.51706     -0.28925
      0.98045     -0.20194      0.16307     -0.10869
     -1.68389     -0.74589      0.52900      0.41580

Lag =    3
      0.03794      0.10491     -0.21635      0.08015
      0.75392      0.22603     -0.25661     -0.47450
     -0.00338      0.05636     -0.08818      0.12723
      0.55022     -0.41232      0.71649     -0.14565
```