NAG C Library Function Document

nag_tsa_multi_cross_corr (g13dmc)

1 Purpose

nag_tsa_multi_cross_corr (g13dmc) calculates the sample cross-correlation (or cross-covariance) matrices of a multivariate time series.

2 Specification

```
void nag_tsa_multi_cross_corr (Nag_CovOrCorr matrix, Integer k, Integer n,
     Integer m, const double w[], double wmean[], double r0[], double r[],
     NagError *fail)
```

3 Description

Let $W_t = (w_{1t}, w_{2t}, \ldots, w_{kt})^T$, for $t = 1, 2, \ldots, n$, denote n observations of a vector of k time series. The sample cross-covariance matrix at lag l is defined to be the k by k matrix $\hat{C}(l)$, whose (i, j)th element is given by

$$\hat{C}_{ij}(l) = \frac{1}{n}\sum_{t=l+1}^{n}(w_{i(t-l)} - \bar{w}_i)(w_{jt} - \bar{w}_j), \quad l = 0, 1, 2, \ldots, m;\ i = 1, 2, \ldots, k;\ j = 1, 2, \ldots, k,$$

where $\bar{w}_i$ and $\bar{w}_j$ denote the sample means for the ith and jth series respectively. The sample cross-correlation matrix at lag l is defined to be the k by k matrix $\hat{R}(l)$, whose (i, j)th element is given by

$$\hat{R}_{ij}(l) = \frac{\hat{C}_{ij}(l)}{\sqrt{\hat{C}_{ii}(0)\hat{C}_{jj}(0)}}, \quad l = 0, 1, 2, \ldots, m;\ i = 1, 2, \ldots, k;\ j = 1, 2, \ldots, k.$$

The number of lags, m, is usually taken to be at most $n/4$.

If W_t follows a vector moving average model of order q, then it can be shown that the theoretical cross-correlation matrices $(R(l))$ are zero beyond lag q. In order to help spot a possible cut-off point, the elements of $\hat{R}(l)$ are usually compared to their approximate standard error of $1/\sqrt{n}$. For further details see, for example, Wei (1990).

The routine uses a single pass through the data to compute the means and the cross-covariance matrix at lag zero. The cross-covariance matrices at further lags are then computed on a second pass through the data.

4 References

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

West D H D (1979) Updating mean and variance estimates: An improved method *Comm. ACM* **22** 532–555

5 Parameters

1: **matrix** – Nag_CovOrCorr *Input*

On entry: indicates whether the cross-covariance or cross-correlation matrices are to be computed.

If **matrix** = **Nag_AutoCov**, then the cross-covariance matrices are computed.

If **matrix** = **Nag_AutoCorr**, then the cross-correlation matrices are computed.

Constraint: **matrix** = **Nag_AutoCov** or **Nag_AutoCorr**.

2: **k** – Integer *Input*

On entry: the dimension, k, of the multivariate time series.

Constraint: **k** ≥ 1.

3: **n** – Integer *Input*

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

Constraint: **n** ≥ 2.

4: **m** – Integer *Input*

On entry: the number, m, of cross-correlation (or cross-covariance) matrices to be computed. If in doubt set **m** $= 10$. However it should be noted that **m** is usually taken to be at most **n**/4.

Constraint: $1 \leq$ **m** $<$ **n**.

5: **w**[dim] – const double *Input*

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

On entry: **w**$[(t-1)k+i-1]$ must contain the value for series i at time t, for $i = 1, 2, \ldots, k$; $t = 1, 2, \ldots, n$.

6: **wmean**[**k**] – double *Output*

On exit: the means, $\bar{w}_i$, for $i = 1, 2, \ldots, k$.

7: **r0**[dim] – double *Output*

Note: the dimension, dim, of the array **r0** must be at least **k** $\times$ **k**.

On exit: if **matrix** $=$ **Nag_AutoCov**, **r0**$[(j-1)k+i-1]$ contains the (i,j)th element of the sample cross-covariance matrix. If **matrix** $=$ **Nag_AutoCorr**, **r0**$[(j-1)k+i-1]$, $i \neq j$ contains the (i,j)th element of the sample cross-correlation matrix and **r0**$[(i-1)k+i-1]$ contains the standard deviation of the ith series.

8: **r**[dim] – double *Output*

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

On exit: if **matrix** $=$ **Nag_AutoCov**, **r**$[(l-1)k^2+(j-1)k+i-1]$ contains the (i,j)th element of the sample cross-covariance matrix at lag l. If **matrix** $=$ **Nag_AutoCorr**, then it contains the (i,j)th element of the sample cross-correlation matrix lag l, for $l = 1, 2, \ldots, m$; $i = 1, 2, \ldots, k$; $j = 1, 2, \ldots, k$.

9: **fail** – NagError * *Input/Output*

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

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

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

NE_INT_2

On entry, **m** < 1 or **m** $\geq$ **n**: **m** = ⟨*value*⟩, **n** = ⟨*value*⟩.

NE_ZERO_VARIANCE

On entry, at least one of the series is such that all its elements are practically identical giving zero (or near zero) variance.

NE_ALLOC_FAIL

Memory allocation failed.

NE_BAD_PARAM

On entry, parameter ⟨*value*⟩ 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

For a discussion of the accuracy of the one-pass algorithm used to compute the sample cross-covariances at lag zero see West (1979). For the other lags a two-pass algorithm is used to compute the cross-covariances; the accuracy of this algorithm is also discussed in West (1979). The accuracy of the cross-correlations will depend on the accuracy of the computed cross-covariances.

8 Further Comments

The time taken is roughly proportional to mnk^2.

9 Example

This program computes the sample cross-correlation matrices of two time series of length 48, up to lag 10. It also prints the cross-correlation matrices together with plots of symbols indicating which elements of the correlation matrices are significant. 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 line depending on whether the elements are significant in the positive or negative direction.

9.1 Program Text

```
/* nag_tsa_multi_cross_corr (g13dmc) Example Program.
 *
 * Copyright 2002 Numerical Algorithms Group.
 *
 * Mark 7, 2002.
 */

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

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

int main(void)
{
  /* Scalars */
  Integer exit_status, i, j, k, m, n, pdw;
  NagError fail;
  Nag_CovOrCorr matrix;

  /* Arrays */
```

```
  double *r0 = 0, *r = 0, *w = 0, *wmean = 0;

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

  INIT_FAIL(fail);
  exit_status = 0;

  Vprintf("g13dmc Example Program Results\n");

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

  Vscanf("%ld%ld%ld%*[^\n] ", &k, &n, &m);

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

      pdw = k;

      for (i = 1; i <= k; ++i)
        {
          for (j = 1; j <= n; ++j)
            Vscanf("%lf", &W(i,j));
          Vscanf("%*[^\n] ");
        }

      matrix  = Nag_AutoCorr;

      g13dmc(matrix, k, n, m, w, wmean, r0, r, &fail);
      if (fail.code != NE_NOERROR)
        {
          Vprintf("Error from g13dmc.\n%s\n",   fail.message);
          exit_status = 1;
          goto END;
        }
      cprint(k, n, k, m, wmean, r);
    }

 END:
  if (r0) NAG_FREE(r0);
  if (r) NAG_FREE(r);
  if (w) NAG_FREE(w);
  if (wmean) NAG_FREE(wmean);

  return exit_status;
}

/* Print the correlation matrices and indicator symbols. */
static void cprint(Integer k, Integer n, Integer ik, Integer m,
                   double *wmean, double *r)
{
  /* Scalars */
  double c1, c2, c3, c5, c6, c7, cnst, sum;
  Integer i2, i, j, lf, llf, ii;

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

#define R(I,J,K) r[((K-1)*ik + (J-1))*ik + I - 1]

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

```
  Vprintf("\n");
  Vprintf(" THE MEANS\n");
  Vprintf(" ---------\n");
  Vprintf("  ");
  for (i = 1; i <= k; ++i)
    {
      Vprintf("%10.3f", wmean[i-1]);
      if (i % 2 == 0 || i == k)
        Vprintf("\n");
    }

  Vprintf("\n");
  Vprintf(" CROSS-CORRELATION MATRICES\n");
  Vprintf(" --------------------------\n");
  for (lf = 1; lf <= m; ++lf)
    {
      Vprintf("\n");
      Vprintf(" Lag = %2ld\n", lf);
      for (i = 1; i <= k; i++)
        {
          for (j = 1; j <= k; j++)
            Vprintf("%9.3f", R(i,j,lf));
          Vprintf("\n");
        }
    }

  /* Print indicator symbols to indicate significant elements. */
  Vprintf("\n");
  Vprintf(" Standard error = 1 / SQRT(N) = %5.3f\n", cnst);
  Vprintf("\n");
  Vprintf(" TABLES OF INDICATOR SYMBOLS\n");
  Vprintf(" ---------------------------\n");
  Vprintf("\n");
  Vprintf(" For Lags 1 to %2ld\n", m);
  Vprintf("\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)
        {
          Vprintf("\n");
          Vprintf("\n");
          if (i == j)
            Vprintf("Auto-correlation function for series %2ld\n", i);
          else
            Vprintf("Cross-correlation function for series %2ld"
                    " and series%2ld\n", i, j);
          Vprintf("\n");

          /* Clear the last plot with blanks */
          sprintf(&rec[0][0], "              0.005  :");
          sprintf(&rec[1][0], "        +     0.01   :");
          sprintf(&rec[2][0], "              0.05   :");
            sprintf(&rec[3][0], "    Sig. Level         : - - - - - - - - - -
Lags");
          sprintf(&rec[4][0], "              0.05   :");
          sprintf(&rec[5][0], "        -     0.01   :");
          sprintf(&rec[6][0], "              0.005  :");
          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 = R(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][llf] = '*';
            }

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

  return;
}
```

9.2 Program Data

```
g13dmc 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 12.140 : End of time series
```

9.3 Program Results

```
g13dmc Example Program Results

 THE MEANS
 ---------
       4.370     7.868

 CROSS-CORRELATION MATRICES
 --------------------------

 Lag =  1
    0.736    0.174
    0.211    0.555

 Lag =  2
```

```
    0.456    0.076
    0.069    0.260

 Lag =  3
    0.379    0.014
    0.026   -0.038

 Lag =  4
    0.322    0.110
    0.093   -0.236

 Lag =  5
    0.341    0.269
    0.087   -0.250

 Lag =  6
    0.363    0.344
    0.132   -0.227

 Lag =  7
    0.280    0.425
    0.207   -0.128

 Lag =  8
    0.248    0.522
    0.197   -0.085

 Lag =  9
    0.240    0.266
    0.254    0.075

 Lag = 10
    0.162   -0.020
    0.267    0.005

 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  :
```
