NAG Library Function Document

nag_tsa_cross_spectrum_bivar (g13cec)

1 Purpose

For a bivariate time series, nag_tsa_cross_spectrum_bivar (g13cec) calculates the cross amplitude spectrum and squared coherency, together with lower and upper bounds from the univariate and bivariate (cross) spectra.

2 Specification

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

```
void nag_tsa_cross_spectrum_bivar (const double xg[], const double yg[],
     const Complex xyg[], Integer ng, const double stats[], double ca[],
     double calw[], double caup[], double *t, double sc[], double sclw[],
     double scup[], NagError *fail)
```

3 Description

Estimates of the cross amplitude spectrum $A(\omega)$ and squared coherency $W(\omega)$ are calculated for each frequency ω as

$$\begin{aligned} A(\omega) &= \left|f_{xy}(\omega)\right| = \sqrt{cf(\omega)^2 + qf(\omega)^2} \text{ and} \\ W(\omega) &= \frac{\left|f_{xy}(\omega)\right|^2}{f_{xx}(\omega)f_{yy}(\omega)} \end{aligned}$$

where:

$cf(\omega)$ and $qf(\omega)$ are the co-spectrum and quadrature spectrum estimates between the series, i.e., the real and imaginary parts of the cross spectrum $f_{xy}(\omega)$ as obtained using nag_tsa_spectrum_bivar_cov (g13ccc) or nag_tsa_spectrum_bivar (g13cdc). $f_{xx}(\omega)$ and $f_{yy}(\omega)$ are the univariate spectrum estimates for the two series as obtained using nag_tsa_spectrum_univar_cov (g13cac) or nag_tsa_spectrum_univar (g13cbc). The same type and amount of smoothing should be used for these estimates, and this is specified by the degrees of freedom and bandwidth values which are passed from the calls of nag_tsa_spectrum_univar_cov (g13cac) or nag_tsa_spectrum_univar (g13cbc).

Upper and lower 95% confidence limits for the cross amplitude are given approximately by

$$A(\omega)\left[1 \pm \left(1.96/\sqrt{d}\right)\sqrt{W(\omega)^{-1} + 1}\right],$$

except that a negative lower limit is reset to 0.0, in which case the approximation is rather poor. You are therefore particularly recommended to compare the coherency estimate $W(\omega)$ with the critical value T derived from the upper 5% point of the F-distribution on $(2, d-2)$ degrees of freedom:

$$T = \frac{2F}{d - 2 + 2F}$$

where d is the degrees of freedom associated with the univariate spectrum estimates. The value of T is returned by the function.

The hypothesis that the series are unrelated at frequency ω, i.e., that both the true cross amplitude and coherency are zero, may be rejected at the 5% level if $W(\omega) > T$. Tests at two frequencies separated by more than the bandwidth may be taken to be independent.

The confidence limits on $A(\omega)$ are strictly appropriate only at frequencies for which the coherency is significant. The same applies to the confidence limits on $W(\omega)$ which are however calculated at all frequencies using the approximation that $\operatorname{arctanh}\left(\sqrt{W(l)}\right)$ is Normal with variance $1/d$.

4 References

Bloomfield P (1976) *Fourier Analysis of Time Series: An Introduction* Wiley

Jenkins G M and Watts D G (1968) *Spectral Analysis and its Applications* Holden–Day

5 Arguments

1: **xg**[**ng**] – const double *Input*

On entry: the **ng** univariate spectral estimates, $f_{xx}(\omega)$, for the x series.

2: **yg**[**ng**] – const double *Input*

On entry: the **ng** univariate spectral estimates, $f_{yy}(\omega)$, for the y series.

3: **xyg**[**ng**] – const Complex *Input*

On entry: $f_{xy}(\omega)$, the **ng** bivariate spectral estimates for the x and y series. The x series leads the y series.

Note: the two univariate and the bivariate spectra must each have been calculated using the same amount of smoothing. The frequency width and the shape of the window and the frequency division of the spectral estimates must be the same. The spectral estimates and statistics must also be unlogged.

4: **ng** – Integer *Input*

On entry: the number of spectral estimates in each of the arrays **xg**, **yg** and **xyg**. It is also the number of cross amplitude spectral and squared coherency estimates.

Constraint: **ng** ≥ 1.

5: **stats**[**4**] – const double *Input*

On entry: the 4 associated statistics for the univariate spectral estimates for the x and y series. **stats**[0] contains the degrees of freedom, **stats**[1] and **stats**[2] contain the lower and upper bound multiplying factors respectively and **stats**[3] contains the bandwidth.

Constraints:

stats[0] ≥ 3.0;
$0.0 <$ **stats**[1] ≤ 1.0;
stats[2] ≥ 1.0.

6: **ca**[**ng**] – double *Output*

On exit: the **ng** cross amplitude spectral estimates $\hat{A}(\omega)$ at each frequency of ω.

7: **calw**[**ng**] – double *Output*

On exit: the **ng** lower bounds for the **ng** cross amplitude spectral estimates.

8: **caup**[**ng**] – double *Output*

On exit: the **ng** upper bounds for the **ng** cross amplitude spectral estimates.

9: **t** – double * *Output*

On exit: the critical value for the significance of the squared coherency, T.

10: **sc**[**ng**] – double *Output*

On exit: the **ng** squared coherency estimates, $\hat{W}(\omega)$ at each frequency ω.

11: **sclw**[**ng**] – double *Output*

On exit: the **ng** lower bounds for the **ng** squared coherency estimates.

12: **scup**[**ng**] – double *Output*

On exit: the **ng** upper bounds for the **ng** squared coherency estimates.

13: **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_BIVAR_SPECTRAL_ESTIM_ZERO

A bivariate spectral estimate is zero.

For this frequency the cross amplitude spectrum is set to zero, and the contributions to the impulse response function and its standard error are set to zero.

NE_INT_ARG_LT

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

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_ARG_GT

On entry, **stats**[1] must not be greater than 1.0: **stats**[1] = ⟨*value*⟩.

NE_REAL_ARG_LE

On entry, **stats**[1] must not be less than or equal to 0.0: **stats**[1] = ⟨*value*⟩.

NE_REAL_ARG_LT

On entry, **stats**[0] must not be less than 3.0: **stats**[0] = ⟨*value*⟩.

On entry, **stats**[2] must not be less than 1.0: **stats**[2] = ⟨*value*⟩.

NE_SQUARED_FREQ_GT_ONE

A calculated value of the squared coherency exceeds one.

For this frequency the squared coherency is reset to one with the result that the cross amplitude spectrum is zero and the contribution to the impulse response function at this frequency is zero.

NE_UNIVAR_SPECTRAL_ESTIM_NEG

A bivariate spectral estimate is negative.

For this frequency the cross amplitude spectrum is set to zero, and the contributions to the impulse response function and its standard error are set to zero.

NE_UNIVAR_SPECTRAL_ESTIM_ZERO

A bivariate spectral estimate is zero.

For this frequency the cross amplitude spectrum is set to zero, and the contributions to the impulse response function and its standard error are set to zero.

7 Accuracy

All computations are very stable and yield good accuracy.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by nag_tsa_cross_spectrum_bivar (g13cec) is approximately proportional to **ng**.

10 Example

The example program reads the set of univariate spectrum statistics, the 2 univariate spectra and the cross spectrum at a frequency division of $\frac{2\pi}{20}$ for a pair of time series. It calls nag_tsa_cross_spectrum_bivar (g13cec) to calculate the cross amplitude spectrum and squared coherency and their bounds and prints the results.

10.1 Program Text

```
/* nag_tsa_cross_spectrum_bivar (g13cec) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 4, 1996.
 * Mark 8 revised, 2004.
 *
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <naga02.h>
#include <nagg13.h>

#define L      80
#define KC     8*L
#define NGMAX  KC
#define NXYMAX 300

int main(void)
{

  Complex  *xyg = 0;
  Integer  exit_status = 0, i, is, j, kc = KC, l = L, mw, ng, nxy;
  NagError fail;
  double   *ca = 0, *calw = 0, *caup = 0, pw, pxy, *sc = 0, *sclw = 0;
  double   *scup = 0, stats[4], t, *x = 0, *xg = 0, *y = 0, *yg = 0;
  INIT_FAIL(fail);

  printf(
          "nag_tsa_cross_spectrum_bivar (g13cec) Example Program Results\n");

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

```
  scanf("%*[^\n] ");
#endif

#ifdef _WIN32
  scanf_s("%"NAG_IFMT" ", &nxy);
#else
  scanf("%"NAG_IFMT" ", &nxy);
#endif
  if (nxy > 0 && nxy <= NXYMAX)
    {
      if (!(x = NAG_ALLOC(KC, double)) ||
          !(y = NAG_ALLOC(KC, double)) ||
          !(ca = NAG_ALLOC(NGMAX, double)) ||
          !(calw = NAG_ALLOC(NGMAX, double)) ||
          !(caup = NAG_ALLOC(NGMAX, double)) ||
          !(sc = NAG_ALLOC(NGMAX, double)) ||
          !(sclw = NAG_ALLOC(NGMAX, double)) ||
          !(scup = NAG_ALLOC(NGMAX, double)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }
      for (i = 1; i <= nxy; ++i)
#ifdef _WIN32
        scanf_s("%lf ", &x[i - 1]);
#else
        scanf("%lf ", &x[i - 1]);
#endif
      for (i = 1; i <= nxy; ++i)
#ifdef _WIN32
        scanf_s("%lf ", &y[i - 1]);
#else
        scanf("%lf ", &y[i - 1]);
#endif

      /* Set parameters for call to nag_tsa_spectrum_univar (g13cbc) and g13cdc
       *  with mean correction and 10 percent taper
       */
      pxy = 0.1;
      /* Window shape parameter and zero covariance at lag 16 */
      pw = 0.5;
      mw = 16;
      /* Alignment shift of 3 */
      is = 3;

      /* Obtain univariate spectrum for the x and the y series */
      /* nag_tsa_spectrum_univar (g13cbc).
       * Univariate time series, smoothed sample spectrum using
       * spectral smoothing by the trapezium frequency (Daniell)
       * window
       */
      nag_tsa_spectrum_univar(nxy, Nag_Mean, pxy, mw, pw, l, kc, Nag_Unlogged,
                              x, &xg, &ng, stats, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_tsa_spectrum_univar (g13cbc).\n%s\n",
                 fail.message);
          exit_status = 1;
          goto END;
        }
      /* nag_tsa_spectrum_univar (g13cbc), see above. */
      nag_tsa_spectrum_univar(nxy, Nag_Mean, pxy, mw, pw, l, kc, Nag_Unlogged,
                              y, &yg, &ng, stats, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_tsa_spectrum_univar (g13cbc).\n%s\n",
                 fail.message);
          exit_status = 1;
          goto END;
        }
```

```
      /* Obtain cross spectrum of the bivariate series */
      /* nag_tsa_spectrum_bivar (g13cdc).
       * Multivariate time series, smoothed sample cross spectrum
       * using spectral smoothing by the trapezium frequency
       * (Daniell) window
       */
      nag_tsa_spectrum_bivar(nxy, Nag_Mean, pxy, mw, is, pw, l, kc, x, y, &xyg,
                             &ng, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf("Error from nag_tsa_spectrum_bivar (g13cdc).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }

      /* nag_tsa_cross_spectrum_bivar (g13cec).
       * Multivariate time series, cross amplitude spectrum,
       * squared coherency, bounds, univariate and bivariate
       * (cross) spectra
       */
      nag_tsa_cross_spectrum_bivar(xg, yg, xyg, ng, stats, ca, calw, caup, &t,
                                   sc, sclw, scup, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf(
                  "Error from nag_tsa_cross_spectrum_bivar (g13cec).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }

      printf("\n");
      printf("       Cross amplitude spectrum\n\n");
      printf("                     Lower     Upper\n");
      printf("           Value     bound     bound\n\n");
      for (j = 1; j <= ng; ++j)
        printf(" %5"NAG_IFMT"%10.4f%10.4f%10.4f\n",
                j - 1, ca[j - 1], calw[j - 1], caup[j - 1]);
      printf("\n");
      printf(" Squared coherency test statistic =%12.4f\n\n", t);
      printf("         Squared coherency\n\n");
      printf("                     Lower     Upper\n");
      printf("           Value     bound     bound\n\n");
      for (j = 1; j <= ng; ++j)
        printf(" %5"NAG_IFMT"%10.4f%10.4f%10.4f\n",
                j - 1, sc[j - 1], sclw[j - 1], scup[j - 1]);
    }
  NAG_FREE(xg);
  NAG_FREE(yg);
  NAG_FREE(xyg);
 END:
  NAG_FREE(x);
  NAG_FREE(y);
  NAG_FREE(ca);
  NAG_FREE(calw);
  NAG_FREE(caup);
  NAG_FREE(sc);
  NAG_FREE(sclw);
  NAG_FREE(scup);
  return exit_status;
}
```

10.2 Program Data

```
nag_tsa_cross_spectrum_bivar (g13cec) Example Program Data
296
-0.109  0.000  0.178  0.339  0.373  0.441  0.461  0.348
 0.127 -0.180 -0.588 -1.055 -1.421 -1.520 -1.302 -0.814
-0.475 -0.193  0.088  0.435  0.771  0.866  0.875  0.891
 0.987  1.263  1.775  1.976  1.934  1.866  1.832  1.767
 1.608  1.265  0.790  0.360  0.115  0.088  0.331  0.645
 0.960  1.409  2.670  2.834  2.812  2.483  1.929  1.485
 1.214  1.239  1.608  1.905  2.023  1.815  0.535  0.122
 0.009  0.164  0.671  1.019  1.146  1.155  1.112  1.121
 1.223  1.257  1.157  0.913  0.620  0.255 -0.280 -1.080
-1.551 -1.799 -1.825 -1.456 -0.944 -0.570 -0.431 -0.577
-0.960 -1.616 -1.875 -1.891 -1.746 -1.474 -1.201 -0.927
-0.524  0.040  0.788  0.943  0.930  1.006  1.137  1.198
 1.054  0.595 -0.080 -0.314 -0.288 -0.153 -0.109 -0.187
-0.255 -0.299 -0.007  0.254  0.330  0.102 -0.423 -1.139
-2.275 -2.594 -2.716 -2.510 -1.790 -1.346 -1.081 -0.910
-0.876 -0.885 -0.800 -0.544 -0.416 -0.271  0.000  0.403
 0.841  1.285  1.607  1.746  1.683  1.485  0.993  0.648
 0.577  0.577  0.632  0.747  0.999  0.993  0.968  0.790
 0.399 -0.161 -0.553 -0.603 -0.424 -0.194 -0.049  0.060
 0.161  0.301  0.517  0.566  0.560  0.573  0.592  0.671
 0.933  1.337  1.460  1.353  0.772  0.218 -0.237 -0.714
-1.099 -1.269 -1.175 -0.676  0.033  0.556  0.643  0.484
 0.109 -0.310 -0.697 -1.047 -1.218 -1.183 -0.873 -0.336
 0.063  0.084  0.000  0.001  0.209  0.556  0.782  0.858
 0.918  0.862  0.416 -0.336 -0.959 -1.813 -2.378 -2.499
-2.473 -2.330 -2.053 -1.739 -1.261 -0.569 -0.137 -0.024
-0.050 -0.135 -0.276 -0.534 -0.871 -1.243 -1.439 -1.422
-1.175 -0.813 -0.634 -0.582 -0.625 -0.713 -0.848 -1.039
-1.346 -1.628 -1.619 -1.149 -0.488 -0.160 -0.007 -0.092
-0.620 -1.086 -1.525 -1.858 -2.029 -2.024 -1.961 -1.952
-1.794 -1.302 -1.030 -0.918 -0.798 -0.867 -1.047 -1.123
-0.876 -0.395  0.185  0.662  0.709  0.605  0.501  0.603
 0.943  1.223  1.249  0.824  0.102  0.025  0.382  0.922
 1.032  0.866  0.527  0.093 -0.458 -0.748 -0.947 -1.029
-0.928 -0.645 -0.424 -0.276 -0.158 -0.033  0.102  0.251
 0.280  0.000 -0.493 -0.759 -0.824 -0.740 -0.528 -0.204
 0.034  0.204  0.253  0.195  0.131  0.017 -0.182 -0.262
53.8 53.6 53.5 53.5 53.4 53.1 52.7 52.4 52.2 52.0 52.0 52.4 53.0 54.0 54.9 56.0
56.8 56.8 56.4 55.7 55.0 54.3 53.2 52.3 51.6 51.2 50.8 50.5 50.0 49.2 48.4 47.9
47.6 47.5 47.5 47.6 48.1 49.0 50.0 51.1 51.8 51.9 51.7 51.2 50.0 48.3 47.0 45.8
45.6 46.0 46.9 47.8 48.2 48.3 47.9 47.2 47.2 48.1 49.4 50.6 51.5 51.6 51.2 50.5
50.1 49.8 49.6 49.4 49.3 49.2 49.3 49.7 50.3 51.3 52.8 54.4 56.0 56.9 57.5 57.3
56.6 56.0 55.4 55.4 56.4 57.2 58.0 58.4 58.4 58.1 57.7 57.0 56.0 54.7 53.2 52.1
51.6 51.0 50.5 50.4 51.0 51.8 52.4 53.0 53.4 53.6 53.7 53.8 53.8 53.8 53.3 53.0
52.9 53.4 54.6 56.4 58.0 59.4 60.2 60.0 59.4 58.4 57.6 56.9 56.4 56.0 55.7 55.3
55.0 54.4 53.7 52.8 51.6 50.6 49.4 48.8 48.5 48.7 49.2 49.8 50.4 50.7 50.9 50.7
50.5 50.4 50.2 50.4 51.2 52.3 53.2 53.9 54.1 54.0 53.6 53.2 53.0 52.8 52.3 51.9
51.6 51.6 51.4 51.2 50.7 50.0 49.4 49.3 49.7 50.6 51.8 53.0 54.0 55.3 55.9 55.9
54.6 53.5 52.4 52.1 52.3 53.0 53.8 54.6 55.4 55.9 55.9 55.2 54.4 53.7 53.6 53.6
53.2 52.5 52.0 51.4 51.0 50.9 52.4 53.5 55.6 58.0 59.5 60.0 60.4 60.5 60.2 59.7
59.0 57.6 56.4 55.2 54.5 54.1 54.1 54.4 55.5 56.2 57.0 57.3 57.4 57.0 56.4 55.9
55.5 55.3 55.2 55.4 56.0 56.5 57.1 57.3 56.8 55.6 55.0 54.1 54.3 55.3 56.4 57.2
57.8 58.3 58.6 58.8 58.8 58.6 58.0 57.4 57.0 56.4 56.3 56.4 56.4 56.0 55.2 54.0
53.0 52.0 51.6 51.6 51.1 50.4 50.0 50.0 52.0 54.0 55.1 54.5 52.8 51.4 50.8 51.2
52.0 52.8 53.8 54.5 54.9 54.9 54.8 54.4 53.7 53.3 52.8 52.6 52.6 53.0 54.3 56.0
57.0 58.0 58.6 58.5 58.3 57.8 57.3 57.0
```

10.3 Program Results

```
nag_tsa_cross_spectrum_bivar (g13cec) Example Program Results

       Cross amplitude spectrum

                    Lower     Upper
           Value    bound     bound
```

```
     0    6.1563    3.6901   10.2708
     1    5.9415    3.5602    9.9155
     2    4.3034    2.5797    7.1791
     3    2.6086    1.5642    4.3504
     4    1.8249    1.0918    3.0502
     5    1.1865    0.7060    1.9941
     6    0.9616    0.5694    1.6239
     7    0.6893    0.4080    1.1644
     8    0.4629    0.2728    0.7855
     9    0.2398    0.1407    0.4087
    10    0.1199    0.0698    0.2060
    11    0.0769    0.0441    0.1342
    12    0.0463    0.0262    0.0819
    13    0.0301    0.0166    0.0545
    14    0.0158    0.0083    0.0301
    15    0.0089    0.0046    0.0175
    16    0.0050    0.0024    0.0103
    17    0.0016    0.0007    0.0040
    18    0.0004    0.0001    0.0024
    19    0.0004    0.0001    0.0018
    20    0.0003    0.0001    0.0016
    21    0.0003    0.0000    0.0015
    22    0.0004    0.0001    0.0018
    23    0.0003    0.0000    0.0026
    24    0.0002    0.0000    0.0028
    25    0.0004    0.0001    0.0015
    26    0.0003    0.0001    0.0010
    27    0.0001    0.0000    0.0018
    28    0.0002    0.0000    0.0013
    29    0.0003    0.0001    0.0010
    30    0.0004    0.0001    0.0010
    31    0.0002    0.0001    0.0009
    32    0.0001    0.0000    0.0022
    33    0.0000    0.0000    4.4716
    34    0.0001    0.0000    0.0007
    35    0.0002    0.0000    0.0006
    36    0.0002    0.0000    0.0006
    37    0.0001    0.0000    0.0006
    38    0.0001    0.0000    0.0007
    39    0.0001    0.0000    0.0012
    40    0.0001    0.0000    0.0012

 Squared coherency test statistic =      0.1926

        Squared coherency

                    Lower     Upper
          Value     bound     bound

     0    0.9562    0.9124    0.9783
     1    0.9539    0.9079    0.9772
     2    0.9567    0.9134    0.9786
     3    0.9591    0.9181    0.9798
     4    0.9428    0.8864    0.9716
     5    0.9048    0.8148    0.9523
     6    0.8738    0.7586    0.9362
     7    0.8715    0.7545    0.9350
     8    0.8453    0.7087    0.9212
     9    0.8198    0.6654    0.9075
    10    0.7770    0.5956    0.8841
    11    0.7033    0.4838    0.8422
    12    0.6541    0.4152    0.8131
    13    0.5702    0.3090    0.7609
    14    0.4474    0.1786    0.6776
    15    0.3945    0.1318    0.6385
    16    0.3260    0.0802    0.5844
    17    0.1887    0.0115    0.4580
    18    0.0390    0.0000    0.2566
    19    0.0618    0.0000    0.2975
    20    0.0524    0.0000    0.2815
    21    0.0428    0.0000    0.2639
```

```
    22     0.0578     0.0000     0.2907
    23     0.0282     0.0000     0.2337
    24     0.0220     0.0000     0.2188
    25     0.0807     0.0000     0.3268
    26     0.0754     0.0000     0.3188
    27     0.0167     0.0000     0.2047
    28     0.0331     0.0000     0.2445
    29     0.0777     0.0000     0.3223
    30     0.1414     0.0014     0.4059
    31     0.0739     0.0000     0.3166
    32     0.0107     0.0000     0.1861
    33     0.0008     0.0000     0.1358
    34     0.0488     0.0000     0.2750
    35     0.0760     0.0000     0.3198
    36     0.0738     0.0000     0.3165
    37     0.0719     0.0000     0.3135
    38     0.0438     0.0000     0.2657
    39     0.0168     0.0000     0.2049
    40     0.0183     0.0000     0.2091
```