NAG Library Function Document

nag_tsa_noise_spectrum_bivar (g13cgc)

1 Purpose

For a bivariate time series, nag_tsa_noise_spectrum_bivar (g13cgc) calculates the noise spectrum together with multiplying factors for the bounds and the impulse response function and its standard error, from the univariate and bivariate spectra.

2 Specification

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

```
void nag_tsa_noise_spectrum_bivar (const double xg[], const double yg[],
     const Complex xyg[], Integer ng, const double stats[], Integer l,
     Integer n, double er[], double *erlw, double *erup, double rf[],
     double *rfse, NagError *fail)
```

3 Description

An estimate of the noise spectrum in the dependence of series y on series x at frequency ω is given by

$$f_{y|x}(\omega) = f_{yy}(\omega)(1 - W(\omega))$$

where $W(\omega)$ is the squared coherency described in G13GEF and $f_{yy}(\omega)$ is the univariate spectrum estimate for series y. Confidence limits on the true spectrum are obtained using multipliers as described for G13CAF, but based on $(d-2)$ degrees of freedom.

If the dependence of y_t on x_t can be assumed to be represented in the time domain by the one sided relationship

$$y_t = v_0 x_t + v_1 x_{t-1} + \cdots + n_t$$

where the noise n_t is independent of x_t, then it is the spectrum of this noise which is estimated by $f_{y|x}(\omega)$.

Estimates of the impulse response function $v_0, v_1, v_2, \ldots$ may also be obtained as

$$v_k = \frac{1}{\pi}\int_0^\pi \mathrm{Re}\left(\frac{\exp(ik\omega)f_{xy}(\omega)}{f_{xx}(\omega)}\right)$$

where Re indicates the real part of the expression. For this purpose it is essential that the univariate spectrum for x, $f_{xx}(\omega)$,and the cross spectrum, $f_{xy}(\omega)$ be supplied to this function for a frequency range

$$\omega_l = \left[\frac{2\pi l}{L}\right], 0 \leq l \leq [L/2],$$

where [] denotes the integer part, the integral being approximated by a finite Fourier transform.

An approximate standard error is calculated for the estimates v_k. Significant values of v_k in the locations described as anticipatory responses in the argument array **rf**, indicate that feedback exists from y_t to x_t. This will bias the estimates of v_k in any causal dependence of y_t on $x_t, x_{t-1}, \ldots$.

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

Note: the two univariate and 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 noise spectral 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 degree 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: **l** – Integer *Input*

On entry: the frequency division, L, of the spectral estimates as $2\pi/L$, as input to nag_tsa_spectrum_univar (g13cbc) and nag_tsa_spectrum_bivar (g13cdc).

Constraints:

ng $= [$**l**$/2] + 1$;
The largest prime factor of **l** must not exceed 19, and the total number of prime factors of **l**, counting repetitions, must not exceed 20. These two restrictions are imposed by the internal FFT algorithm used.

7: **n** – Integer *Input*

On entry: the number of points in each of the time series x and y. **n** should have the same value as **nxy** in the call of nag_tsa_spectrum_bivar_cov (g13ccc) or nag_tsa_spectrum_bivar (g13cdc) which calculated the smoothed sample cross spectrum. **n** is used in calculating the impulse response function standard error (**rfse**).

Constraint: **n** ≥ 1.

8: **er**[**ng**] – double *Output*

On exit: the **ng** estimates of the noise spectrum, $\hat{f}_{y|x}(\omega)$ at each frequency.

9: **erlw** – double * *Output*

On exit: the noise spectrum lower limit multiplying factor.

10: **erup** – double * *Output*

On exit: the noise spectrum upper limit multiplying factor.

11: **rf**[**l**] – double *Output*

On exit: the impulse response function. Causal responses are stored in ascending frequency in **rf**[0] to **rf**[**ng** – 1] and anticipatory responses are stored in descending frequency in **rf**[**ng**] to **rf**[**l**].

12: **rfse** – double * *Output*

On exit: the impulse response function standard error.

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

The NAG error argument (see Section 3.6 in the Essential Introduction).

6 Error Indicators and Warnings

NE_2_INT_ARG_CONS

On entry, $\mathbf{l} = \langle value \rangle$ while $\mathbf{ng} = \langle value \rangle$. These arguments must satisfy $\mathbf{ng} = [\mathbf{l}/2] + 1$ when $\mathbf{ng} > 0$.

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_BIVAR_SPECTRAL_ESTIM_ZERO

A bivariate spectral estimate is zero.

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

NE_FACTOR_GT

At least one of the prime factors of **l** is greater than 19.

NE_INT_ARG_LT

On entry, $\mathbf{n} = \langle value \rangle$.
Constraint: $\mathbf{n} \geq 1$.

On entry, $\mathbf{ng} = \langle value \rangle$.
Constraint: $\mathbf{ng} \geq 1$.

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: $\mathbf{stats}[1] = \langle value \rangle$.

NE_REAL_ARG_LE

On entry, **stats**[1] must not be less than or equal to 0.0: $\mathbf{stats}[1] = \langle value \rangle$.

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 noise spectrum is zero and the contribution to the impulse response function at this frequency is zero.

NE_TOO_MANY_FACTORS

l has more than 20 prime factors.

NE_UNIVAR_SPECTRAL_ESTIM_NEG

A bivariate spectral estimate is negative.

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

7 Accuracy

The computation of the noise is stable and yields good accuracy. The FFT is a numerically stable process, and any errors introduced during the computation will normally be insignificant compared with uncertainty in the data.

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by nag_tsa_noise_spectrum_bivar (g13cgc) 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_noise_spectrum_bivar (g13cgc) to calculate the noise spectrum and its confidence limits multiplying factors, the impulse response function and its standard error. It then prints the results.

10.1 Program Text

```
/* nag_tsa_noise_spectrum_bivar (g13cgc) 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 LMAX   80
#define KC     8*L
#define NGMAX  KC
#define L      LMAX
#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   erlw, erup, pw, pxy, rfse;
  double   *er = 0, *rf = 0, *stats = 0, *x = 0, *xg = 0, *y = 0, *yg = 0;

  INIT_FAIL(fail);

  printf(
         "nag_tsa_noise_spectrum_bivar (g13cgc) 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 (!(stats = NAG_ALLOC(4, double)) ||
          !(x = NAG_ALLOC(KC, double)) ||
          !(y = NAG_ALLOC(KC, double)) ||
          !(er = NAG_ALLOC(NGMAX, double)) ||
          !(rf = NAG_ALLOC(LMAX, 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_noise_spectrum_bivar (g13cgc).
 * Multivariate time series, noise spectrum, bounds, impulse
 * response function and its standard error
 */
nag_tsa_noise_spectrum_bivar(xg, yg, xyg, ng, stats, l, nxy, er, &erlw,
                             &erup, rf, &rfse, &fail);
if (fail.code != NE_NOERROR)
  {
    printf(
            "Error from nag_tsa_noise_spectrum_bivar (g13cgc).\n%s\n",
            fail.message);
    exit_status = 1;
    goto END;
  }

printf("\n");
printf("           Noise spectrum\n\n");
for (j = 1; j <= ng; ++j)
  printf("%6"NAG_IFMT"%16.4f\n", j - 1, er[j - 1]);

printf("\nNoise spectrum bounds multiplying factors\n\n");
printf("Lower =%10.4f", erlw);
printf("     Upper =%10.4f\n\n", erup);
printf("Impulse response function\n\n");
for (j = 1; j <= l; ++j)
  printf("%6"NAG_IFMT"%16.4f\n", j - 1, rf[j - 1]);
printf("\nImpulse response function standard error =%10.4f\n",
        rfse);
    }
  NAG_FREE(xg);
  NAG_FREE(yg);
  NAG_FREE(xyg);
 END:
  NAG_FREE(stats);
  NAG_FREE(x);
  NAG_FREE(y);
  NAG_FREE(er);
  NAG_FREE(rf);
  return exit_status;
}
```

10.2 Program Data

```
nag_tsa_noise_spectrum_bivar (g13cgc) 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_noise_spectrum_bivar (g13cgc) Example Program Results

          Noise spectrum

     0          0.9012
     1          0.9074
     2          0.5925
     3          0.3101
     4          0.2915
     5          0.3019
     6          0.3041
     7          0.2183
     8          0.1700
     9          0.0952
    10          0.0597
    11          0.0493
    12          0.0356
    13          0.0299
    14          0.0200
    15          0.0128
    16          0.0084
    17          0.0042
    18          0.0025
    19          0.0022
    20          0.0022
    21          0.0022
    22          0.0023
    23          0.0024
    24          0.0021
    25          0.0017
    26          0.0014
    27          0.0014
    28          0.0014
    29          0.0010
    30          0.0011
    31          0.0010
    32          0.0009
    33          0.0011
    34          0.0009
    35          0.0009
    36          0.0008
    37          0.0007
    38          0.0011
    39          0.0016
    40          0.0017

Noise spectrum bounds multiplying factors

Lower =    0.6298     Upper =    1.8291

Impulse response function

     0         -0.0478
     1          0.0564
     2         -0.0530
     3         -0.5944
     4         -0.7189
     5         -0.8270
     6         -0.4772
     7         -0.2705
     8         -0.0322
     9          0.0165
    10         -0.0810
    11         -0.0289
    12         -0.0107
    13         -0.0395
    14          0.0475
    15          0.0136
    16          0.0153
    17         -0.0050
    18         -0.0033
    19         -0.0403
    20          0.0260
    21         -0.0023
    22          0.0066
    23          0.0094
    24         -0.0010
    25         -0.0130
    26          0.0014
    27          0.0107
    28         -0.0023
    29         -0.0076
    30          0.0313
    31         -0.0074
    32          0.0028
    33          0.0012
    34          0.0012
    35          0.0081
    36         -0.0071
    37         -0.0104
    38         -0.0093
    39          0.0083
    40          0.0028
    41         -0.0096
    42         -0.0001
    43         -0.0024
    44          0.0028
    45         -0.0103
    46          0.0089
    47         -0.0031
    48         -0.0002
    49          0.0068
    50         -0.0025
    51         -0.0063
    52          0.0124
    53         -0.0092
    54         -0.0026
    55         -0.0011
    56          0.0128
    57          0.0136
    58          0.0132
    59         -0.0144
    60         -0.0181
    61          0.0048
    62          0.0114
    63          0.0084
    64         -0.0205
    65         -0.0150
    66          0.0126
    67          0.0057
    68         -0.0284
    69          0.0020
    70          0.0296
    71          0.0287
    72         -0.0396
    73          0.0476
    74         -0.0992
    75         -0.0004
    76         -0.0164
    77        -0.0273
    78         0.0298
    79        -0.0858

Impulse response function standard error =    0.0809
```