NAG Library Function Document

nag_tsa_spectrum_bivar_cov (g13ccc)

1 Purpose

nag_tsa_spectrum_bivar_cov (g13ccc) calculates the smoothed sample cross spectrum of a bivariate time series using one of four lag windows: rectangular, Bartlett, Tukey or Parzen.

2 Specification

```
#include <nag.h>
#include <nagg13.h>
void nag_tsa_spectrum_bivar_cov (Integer nxy,
     NagMeanOrTrend mtxy_correction, double pxy, Integer iw, Integer mw,
     Integer ish, Integer ic, Integer nc, double cxy[], double cyx[],
     Integer kc, Integer l, double xg[], double yg[], Complex g[],
     Integer *ng, NagError *fail)
```

3 Description

The smoothed sample cross spectrum is a complex valued function of frequency ω, $f_{xy}(\omega) = cf(\omega) + iqf(\omega)$, defined by its real part or co-spectrum

$$cf(\omega) = \frac{1}{2\pi} \sum_{k=-M+1}^{M-1} w_k C_{xy}(k+S)\cos(\omega k)$$

and imaginary part or quadrature spectrum

$$qf(\omega) = \frac{1}{2\pi} \sum_{k=-M+1}^{M-1} w_k C_{xy}(k+S)\sin(\omega k)$$

where $w_k = w_{-k}$, for $k = 0, 1, \ldots, M-1$, is the smoothing lag window as defined in the description of nag_tsa_spectrum_univar_cov (g13cac). The alignment shift S is recommended to be chosen as the lag k at which the cross-covariances $c_{xy}(k)$ peak, so as to minimize bias.

The results are calculated for frequency values

$$\omega_j = \frac{2\pi j}{L}, \quad j = 0, 1, \ldots, [L/2],$$

where [] denotes the integer part.

The cross-covariances $c_{xy}(k)$ may be supplied by you, or constructed from supplied series $x_1, x_2, \ldots, x_n$; $y_1, y_2, \ldots, y_n$ as

$$c_{xy}(k) = \frac{\sum_{t=1}^{n-k} x_t y_{t+k}}{n}, \quad k \geq 0$$

$$c_{xy}(k) = \frac{\sum_{t=1-k}^{n} x_t y_{t+k}}{n} = c_{yx}(-k), \quad k < 0$$

this convolution being carried out using the finite Fourier transform.

The supplied series may be mean and trend corrected and tapered before calculation of the cross-covariances, in exactly the manner described in nag_tsa_spectrum_univar_cov (g13cac) for univariate spectrum estimation. The results are corrected for any bias due to tapering.

The bandwidth associated with the estimates is not returned. It will normally already have been calculated in previous calls of nag_tsa_spectrum_univar_cov (g13cac) for estimating the univariate spectra of y_t and x_t.

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: **nxy** – Integer *Input*

On entry: n, the length of the time series x and y.

Constraint: **nxy** ≥ 1.

2: **mtxy_correction** – NagMeanOrTrend *Input*

On entry: if cross-covariances are to be calculated by the function (**ic** $= 0$), **mtxy_correction** must specify whether the data is to be initially mean or trend corrected.

mtxy_correction = Nag_NoCorrection
For no correction.

mtxy_correction = Nag_Mean
For mean correction.

mtxy_correction = Nag_Trend
For trend correction.

If cross-covariances are supplied (**ic** $\neq 0$), **mtxy_correction** should be set to **mtxy_correction** = Nag_NoCorrection

Constraint: if **ic** $= 0$, **mtxy_correction** = Nag_NoCorrection, Nag_Mean or Nag_Trend.

3: **pxy** – double *Input*

On entry: if cross-covariances are to be calculated by the function (**ic** $= 0$), **pxy** must specify the proportion of the data (totalled over both ends) to be initially tapered by the split cosine bell taper. A value of 0.0 implies no tapering.

If cross-covariances are supplied (**ic** $\neq 0$), **pxy** is not used.

Constraint: if **ic** $= 0$, $0.0 \leq$ **pxy** ≤ 1.0.

4: **iw** – Integer *Input*

On entry: the choice of lag window.

iw $= 1$
Rectangular.

iw $= 2$
Bartlett.

iw $= 3$
Tukey.

iw $= 4$
Parzen.

Constraint: $1 \leq$ **iw** ≤ 4.

5: **mw** – Integer *Input*

On entry: M, the 'cut-off' point of the lag window, relative to any alignment shift that has been applied. Windowed cross-covariances at lags $(-$**mw** $+$ **ish**$)$ or less, and at lags $($**mw** $+$ **ish**$)$ or greater are zero.

Constraints:

mw ≥ 1;
mw $+ |$**ish**$| \leq$ **nxy**.

6: **ish** – Integer *Input*

On entry: S, the alignment shift between the x and y series. If x leads y, the shift is positive.

Constraint: $-$**mw** $<$ **ish** $<$ **mw**.

7: **ic** – Integer *Input*

On entry: indicates whether cross-covariances are to be calculated in the function or supplied in the call to the function.

ic $= 0$
Cross-covariances are to be calculated.

ic $\neq 0$
Cross-covariances are to be supplied.

8: **nc** – Integer *Input*

On entry: the number of cross-covariances to be calculated in the function or supplied in the call to the function.

Constraint: **mw** $+ |$**ish**$| \leq$ **nc** $\leq$ **nxy**.

9: **cxy**[**nc**] – double *Input/Output*

On entry: if **ic** $\neq 0$, **cxy** must contain the **nc** cross-covariances between values in the y series and earlier values in time in the x series, for lags from 0 to $($**nc** $- 1)$.

If **ic** $= 0$, **cxy** need not be set.

On exit: if **ic** $= 0$, **cxy** will contain the **nc** calculated cross-covariances.

If **ic** $\neq 0$, the contents of **cxy** will be unchanged.

10: **cyx**[**nc**] – double *Input/Output*

On entry: if **ic** $\neq 0$, **cyx** must contain the **nc** cross-covariances between values in the y series and later values in time in the x series, for lags from 0 to $($**nc** $- 1)$.

If **ic** $= 0$, **cyx** need not be set.

On exit: if **ic** $= 0$, **cyx** will contain the **nc** calculated cross-covariances.

If **ic** $\neq 0$, the contents of **cyx** will be unchanged.

11: **kc** – Integer *Input*

On entry: if **ic** $= 0$, **kc** must specify the order of the fast Fourier transform (FFT) used to calculate the cross-covariances. **kc** should be a product of small primes such as 2^m where m is the smallest integer such that $2^m \geq n +$ **nc**.

If $\mathbf{ic} \neq 0$, that is if covariances are supplied, **kc** is not used.

Constraint: $\mathbf{kc} \geq \mathbf{nxy} + \mathbf{nc}$. The largest prime factor of **kc** must not exceed 19, and the total number of prime factors of **kc**, counting repetitions, must not exceed 20. These two restrictions are imposed by the internal FFT algorithm used.

12: **l** – Integer *Input*

On entry: L, the frequency division of the spectral estimates as $\frac{2\pi}{L}$. Therefore it is also the order of the FFT used to construct the sample spectrum from the cross-covariances. **l** should be a product of small primes such as 2^m where m is the smallest integer such that $2^m \geq 2M - 1$.

Constraint: $\mathbf{l} \geq 2 \times \mathbf{mw} - 1$. 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.

13: **xg**[*dim*] – double *Input/Output*

Note: the dimension, *dim*, of the array **xg** must be at least

$\max(\mathbf{kc}, \mathbf{l})$, when $\mathbf{ic} = 0$;

.

On entry: if the cross-covariances are to be calculated ($\mathbf{ic} = 0$) **xg** must contain the **nxy** data points of the x series. If covariances are supplied ($\mathbf{ic} \neq 0$) **xg** may contain any values.

On exit: contains the real parts of the **ng** complex spectral estimates in elements **xg**[0] to **xg**[**ng** $- 1$], and **xg**[**ng**] to **xg**[$dim - 1$] contain 0.0. The y series leads the x series.

14: **yg**[*dim*] – double *Input/Output*

Note: the dimension, *dim*, of the array **yg** must be at least

$\max(\mathbf{kc}, \mathbf{l})$, when $\mathbf{ic} = 0$;

.

On entry: if the cross-covariances are to be calculated ($\mathbf{ic} = 0$) **yg** must contain the **nxy** data points of the y series. If covariances are supplied ($\mathbf{ic} \neq 0$) **yg** may contain any values.

On exit: contains the imaginary parts of the **ng** complex spectral estimates in elements **yg**[0] to **yg**[**ng** $- 1$], and **yg**[**ng**] to **yg**[$dim - 1$] contain 0.0. The y series leads the x series.

15: **g**[$\mathbf{l}/\mathbf{2} + \mathbf{1}$] – Complex *Output*

On exit: the complex vector that contains the **ng** cross spectral estimates in elements **g**[0] to **g**[**ng** $- 1$]. The y series leads the x series.

16: **ng** – Integer * *Output*

On exit: the number, $[\mathbf{l}/2] + 1$, of complex spectral estimates.

17: **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.
See Section 3.2.1.2 in the Essential Introduction for further information.

NE_BAD_PARAM

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

NE_INT

On entry, **ic** $= 0$ and **mtxy_correction** $\neq$ Nag_NoCorrection, Nag_Mean or Nag_Trend: **mtxy_correction** $= \langle value \rangle$.

On entry, **iw** $= \langle value \rangle$.
Constraint: **iw** $= 1$, 2, 3 or 4.

On entry, **mw** $= \langle value \rangle$.
Constraint: **mw** ≥ 1.

On entry, **nxy** $= \langle value \rangle$.
Constraint: **nxy** ≥ 1.

NE_INT_2

On entry, **ish** $= \langle value \rangle$ and **mw** $= \langle value \rangle$.
Constraint: $|$**ish**$| \leq$ **mw**.

On entry, **l** $= \langle value \rangle$ and **mw** $= \langle value \rangle$.
Constraint: **l** $\geq 2 \times$ **mw** $- 1$.

On entry, **nc** $= \langle value \rangle$ and **nxy** $= \langle value \rangle$.
Constraint: **nc** $\leq$ **nxy**.

NE_INT_3

On entry, **kc** $= \langle value \rangle$, **nxy** $= \langle value \rangle$ and **nc** $= \langle value \rangle$.
Constraint: if **ic** $= 0$, **kc** $\geq$ **nxy** $+$ **nc**.

On entry, **mw** $= \langle value \rangle$, **ish** $= \langle value \rangle$ and **nxy** $= \langle value \rangle$.
Constraint: **mw** $+ |$**ish**$| \leq$ **nxy**.

On entry, **nc** $= \langle value \rangle$, **mw** $= \langle value \rangle$ and **ish** $= \langle value \rangle$.
Constraint: **nc** $\geq$ **mw** $+ |$**ish**$|$.

NE_INT_REAL

On entry, **pxy** $= \langle value \rangle$.
Constraint: if **ic** $= 0$, **pxy** ≤ 1.0.

On entry, **pxy** $= \langle value \rangle$.
Constraint: if **ic** $= 0$, **pxy** ≥ 0.0.

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 the Essential Introduction 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 the Essential Introduction for further information.

NE_PRIME_FACTOR

kc has a prime factor exceeding 19, or more than 20 prime factors (counting repetitions): **kc** $= \langle value \rangle$.

l has a prime factor exceeding 19, or more than 20 prime factors (counting repetitions): **l** $= \langle value \rangle$.

7 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

nag_tsa_spectrum_bivar_cov (g13ccc) carries out two FFTs of length **kc** to calculate the sample cross-covariances and one FFT of length L to calculate the sample spectrum. The timing of nag_tsa_spectrum_bivar_cov (g13ccc) is therefore dependent on the choice of these values. The time taken for an FFT of length n is approximately proportional to $n\log(n)$ (but see Section 9 in nag_sum_fft_realherm_1d (c06pac) for further details).

10 Example

This example reads two time series of length 296. It then selects mean correction, a 10% tapering proportion, the Parzen smoothing window and a cut-off point of 35 for the lag window. The alignment shift is set to 3 and 50 cross-covariances are chosen to be calculated. The program then calls nag_tsa_spectrum_bivar_cov (g13ccc) to calculate the cross spectrum and then prints the cross-covariances and cross spectrum.

10.1 Program Text

```
/* nag_tsa_spectrum_bivar_cov (g13ccc) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 7, 2002.
 */

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

int main(void)
{
  /* Scalars */
  double         pxy;
  Integer        exit_status, i, ic, ii, ish, iw, kc, lf,
                 mw, nc, ng, nxy, nxyg;

  /* Arrays */
  double         *cxy = 0, *cyx = 0, *xg = 0, *yg = 0;
  Complex        *g = 0;

  NagMeanOrTrend mtxy;

  NagError       fail;


  INIT_FAIL(fail);

  exit_status = 0;

  printf(
          "nag_tsa_spectrum_bivar_cov (g13ccc) 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] ", &nxy, &nc, &ic);
#else
  scanf("%"NAG_IFMT"%"NAG_IFMT"%"NAG_IFMT"%*[^\n] ", &nxy, &nc, &ic);
#endif

  if (nxy > 0 && nc > 0)
    {
      /* Set parameters for call to nag_tsa_spectrum_bivar_cov (g13ccc) */
      /* Mean correction and 10 percent taper */
      mtxy = Nag_Mean;
      pxy = 0.1;

      /* Parzen window and zero covariance at lag 35 */
      iw = 4;
      mw = 35;

      /* Alignment shift of 3, 50 covariances to be calculated */
      ish = 3;
      kc = 350;
      lf = 80;

      if (ic == 0)
        nxyg = MAX(kc, lf);
      else
        nxyg = lf;

      /* Allocate arrays xg, yg, cxy and cyx */
      if (!(xg = NAG_ALLOC(nxyg, double)) ||
          !(yg = NAG_ALLOC(nxyg, double)) ||
          !(cxy = NAG_ALLOC(nc, double)) ||
          !(cyx = NAG_ALLOC(nc, double)) ||
          !(g = NAG_ALLOC((lf/2)+1, Complex)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }

      if (ic == 0)
        {
          for (i = 1; i <= nxy; ++i)
#ifdef _WIN32
            scanf_s("%lf", &xg[i-1]);
#else
            scanf("%lf", &xg[i-1]);
#endif
#ifdef _WIN32
          scanf_s("%*[^\n] ");
#else
          scanf("%*[^\n] ");
#endif
          for (i = 1; i <= nxy; ++i)
#ifdef _WIN32
            scanf_s("%lf", &yg[i-1]);
#else
            scanf("%lf", &yg[i-1]);
#endif
#ifdef _WIN32
          scanf_s("%*[^\n] ");
#else
          scanf("%*[^\n] ");
#endif
        }
      else
        {
          for (i = 1; i <= nc; ++i)
```

```
#ifdef _WIN32
            scanf_s("%lf", &cxy[i-1]);
#else
            scanf("%lf", &cxy[i-1]);
#endif
#ifdef _WIN32
          scanf_s("%*[^\n] ");
#else
          scanf("%*[^\n] ");
#endif
          for (i = 1; i <= nc; ++i)
#ifdef _WIN32
            scanf_s("%lf", &cyx[i-1]);
#else
            scanf("%lf", &cyx[i-1]);
#endif
#ifdef _WIN32
          scanf_s("%*[^\n] ");
#else
          scanf("%*[^\n] ");
#endif
        }

      /* nag_tsa_spectrum_bivar_cov (g13ccc).
       * Multivariate time series, smoothed sample cross spectrum
       * using rectangular, Bartlett, Tukey or Parzen lag window
       */
      nag_tsa_spectrum_bivar_cov(nxy, mtxy, pxy, iw, mw, ish, ic, nc, cxy, cyx,
                                 kc, lf, xg, yg, g, &ng, &fail);
      if (fail.code != NE_NOERROR)
        {
          printf(
                  "Error from nag_tsa_spectrum_bivar_cov (g13ccc).\n%s\n",
                  fail.message);
          exit_status = 1;
          goto END;
        }

      printf("\n");
      printf("                 Returned cross covariances\n");
      printf("\n");
      printf(" Lag     XY       YX    Lag     XY       YX    Lag"
              "     XY       YX\n");
      for (i = 1; i <= nc; i += 3)
        {
          for (ii = i; ii <= MIN(i+2, nc); ++ii)
            printf("%4"NAG_IFMT"%9.4f%9.4f ", ii-1, cxy[ii-1],
                    cyx[ii-1]);
          printf("\n");
        }

      printf("\n");
      printf("                     Returned sample spectrum\n");
      printf("\n");
      printf("        Real  Imaginary        Real  Imaginary  "
              "      Real  Imaginary\n");
      printf(" Lag    part     part   Lag    part     part   Lag"
              "    part     part\n");
      for (i = 1; i <= ng; i += 3)
        {
          for (ii = i; ii <= MIN(i+2, ng); ++ii)
            printf("%4"NAG_IFMT"%9.4f%9.4f ", ii-1, g[ii-1].re,
                    g[ii-1].im);
          printf("\n");
        }
    }

 END:
  NAG_FREE(cxy);
  NAG_FREE(cyx);
  NAG_FREE(xg);
```

```
  NAG_FREE(yg);
  NAG_FREE(g);

  return exit_status;
}
```

10.2 Program Data

```
nag_tsa_spectrum_bivar_cov (g13ccc) Example Program Data
   296 50 0
 -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_spectrum_bivar_cov (g13ccc) Example Program Results

                    Returned cross covariances

 Lag      XY        YX     Lag      XY        YX     Lag      XY        YX
   0  -1.6700  -1.6700      1  -2.0581  -1.3606      2  -2.4859  -1.1383
   3  -2.8793  -0.9926      4  -3.1473  -0.9009      5  -3.2239  -0.8382
   6  -3.0929  -0.7804      7  -2.7974  -0.7074      8  -2.4145  -0.6147
   9  -2.0237  -0.5080     10  -1.6802  -0.4032     11  -1.4065  -0.3159
  12  -1.2049  -0.2554     13  -1.0655  -0.2250     14  -0.9726  -0.2238
  15  -0.9117  -0.2454     16  -0.8658  -0.2784     17  -0.8180  -0.3081
  18  -0.7563  -0.3257     19  -0.6750  -0.3315     20  -0.5754  -0.3321
  21  -0.4701  -0.3308     22  -0.3738  -0.3312     23  -0.3023  -0.3332
  24  -0.2665  -0.3384     25  -0.2645  -0.3506     26  -0.2847  -0.3727
  27  -0.3103  -0.3992     28  -0.3263  -0.4152     29  -0.3271  -0.4044
  30  -0.3119  -0.3621     31  -0.2837  -0.2919     32  -0.2568  -0.2054
  33  -0.2427  -0.1185     34  -0.2490  -0.0414     35  -0.2774   0.0227
  36  -0.3218   0.0697     37  -0.3705   0.1039     38  -0.4083   0.1356
  39  -0.4197   0.1805     40  -0.3920   0.2460     41  -0.3241   0.3319
  42  -0.2273   0.4325     43  -0.1216   0.5331     44  -0.0245   0.6199
  45   0.0528   0.6875     46   0.1074   0.7329     47   0.1448   0.7550
  48   0.1713   0.7544     49   0.1943   0.7349

                        Returned sample spectrum

        Real  Imaginary          Real  Imaginary          Real  Imaginary
 Lag    part       part   Lag    part       part   Lag    part       part
   0  -6.5500   0.0000      1  -5.4267  -1.9842      2  -3.1323  -2.7307
   3  -1.2649  -2.3998      4  -0.2102  -1.7520      5   0.3411  -1.1903
   6   0.6063  -0.7420      7   0.6178  -0.3586      8   0.4391  -0.1008
   9   0.2422   0.0061     10   0.1233   0.0409     11   0.0574   0.0529
  12   0.0174   0.0452     13  -0.0008   0.0289     14  -0.0058   0.0161
  15  -0.0051   0.0084     16  -0.0027   0.0040     17  -0.0010   0.0015
  18  -0.0006   0.0006     19  -0.0005   0.0003     20  -0.0003   0.0003
  21  -0.0003   0.0004     22  -0.0003   0.0003     23  -0.0003   0.0002
  24  -0.0004   0.0001     25  -0.0004  -0.0000     26  -0.0003  -0.0001
  27  -0.0002  -0.0001     28  -0.0001   0.0001     29  -0.0002   0.0003
  30  -0.0003   0.0002     31  -0.0002   0.0001     32  -0.0001   0.0000
  33  -0.0000  -0.0000     34   0.0001  -0.0001     35   0.0001  -0.0002
  36   0.0001  -0.0001     37   0.0001  -0.0001     38   0.0001  -0.0001
  39   0.0001  -0.0001     40   0.0001   0.0000
```