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

```c
#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 $\omega$ as

$$
A(\omega) = \frac{|f_{xy}(\omega)|}{\sqrt{f_{xx}(\omega)f_{yy}(\omega)}}
$$

$$
W(\omega) = \frac{f_{xy}(\omega)^2}{f_{xx}(\omega)f_{yy}(\omega)}
$$

where:

- $f_{xy}(\omega)$ and $f_{yx}(\omega)$ 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). The value of $A(\omega)$ is returned by the function.

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 $\omega$, 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 $\text{arctanh} \left( \sqrt{W(i)} \right)$ is Normal with variance $1/d$.

4 References

5 Arguments
1: $\mathbf{xg}$ - const double
   \textit{Input}
   \textit{On entry:} the $\mathbf{ng}$ univariate spectral estimates, $f_{xx}(\omega)$, for the $x$ series.

2: $\mathbf{yg}$ - const double
   \textit{Input}
   \textit{On entry:} the $\mathbf{ng}$ univariate spectral estimates, $f_{yy}(\omega)$, for the $y$ series.

3: $\mathbf{xyg}$ - const Complex
   \textit{Input}
   \textit{On entry:} $f_{xy}(\omega)$, the $\mathbf{ng}$ bivariate spectral estimates for the $x$ and $y$ series. The $x$ series leads the $y$ series.
   \textit{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: $\mathbf{ng}$ - Integer
   \textit{Input}
   \textit{On entry:} the number of spectral estimates in each of the arrays $\mathbf{xg}$, $\mathbf{yg}$ and $\mathbf{xyg}$. It is also the number of cross amplitude spectral and squared coherency estimates.
   \textit{Constraint:} $\mathbf{ng} \geq 1$.

5: $\mathbf{stats}[4]$ - const double
   \textit{Input}
   \textit{On entry:} the 4 associated statistics for the univariate spectral estimates for the $x$ and $y$ series. $\mathbf{stats}[0]$ contains the degrees of freedom, $\mathbf{stats}[1]$ and $\mathbf{stats}[2]$ contain the lower and upper bound multiplying factors respectively and $\mathbf{stats}[3]$ contains the bandwidth.
   \textit{Constraints:}
   \begin{align*}
   \mathbf{stats}[0] &\geq 3.0; \\
   0.0 &< \mathbf{stats}[1] \leq 1.0; \\
   \mathbf{stats}[2] &\geq 1.0.
   \end{align*}

6: $\mathbf{ca}$ - double
   \textit{Output}
   \textit{On exit:} the $\mathbf{ng}$ cross amplitude spectral estimates $\hat{A}(\omega)$ at each frequency of $\omega$.

7: $\mathbf{calw}$ - double
   \textit{Output}
   \textit{On exit:} the $\mathbf{ng}$ lower bounds for the $\mathbf{ng}$ cross amplitude spectral estimates.

8: $\mathbf{caup}$ - double
   \textit{Output}
   \textit{On exit:} the $\mathbf{ng}$ upper bounds for the $\mathbf{ng}$ cross amplitude spectral estimates.

9: $\mathbf{t}$ - double *
   \textit{Output}
   \textit{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 \( \omega \).

11: **scw**[ng] – double

*Output*

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

12: **scu**[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** = \langle value\rangle.

Constraint: **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: **stats**[1] = \langle value\rangle.

**NE_REAL_ARG_LE**

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

**NE_REAL_ARG_LT**

On entry, **stats**[0] must not be less than 3.0: **stats**[0] = \langle value\rangle.

**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.
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 $n_g$.

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 $\frac{20}{25}$ 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, *yg = 0;
    INIT_FAIL(fail);

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

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

    return 0;
}
```
#include <stdio.h>
#include <stdlib.h>

void main() {
    int x, y, nxy;
    double x[KC], y[KC], calw[NGMAX], caup[NGMAX], sc[NGMAX], sclw[NGMAX], scup[NGMAX];
    double pxy, pw, mw, is;

    // Allocate memory for arrays
    if (!(x = NAG_ALLOC(KC, double)) ||
        !(y = NAG_ALLOC(KC, 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;
    }

    // Read data from stdin
    for (i = 1; i <= nxy; ++i) {
        scanf("%lf ", &x[i - 1]);
        scanf("%lf ", &y[i - 1]);
    }

    // Set parameters for call to nag_tsa_spectrum_univar (g13cbc) and g13cdc
    pxy = 0.1;
    pw = 0.5;
    mw = 16;
    is = 3;

    // Obtain univariate spectrum for the x and the y series
    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;
    }

    // Call 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");
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", t);
printf(" Squared coherency\n\n");
printf(" Lower Upper\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);
NAG_FREE(exit_status);
10.2 Program Data

```
-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
0.987 1.263 1.775 1.976 1.934 1.866 1.832 1.767
1.608 1.265 0.790 0.115 0.331 0.645 0.960 1.409
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 -1.520 -1.302 -1.123
```

10.3 Program Results

```
Cross amplitude spectrum

<table>
<thead>
<tr>
<th>Value</th>
<th>Lower bound</th>
<th>Upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>53.8</td>
<td>53.0</td>
<td>54.0</td>
</tr>
<tr>
<td>56.8</td>
<td>56.0</td>
<td>57.0</td>
</tr>
<tr>
<td>45.6</td>
<td>45.0</td>
<td>46.0</td>
</tr>
<tr>
<td>50.1</td>
<td>49.3</td>
<td>50.3</td>
</tr>
<tr>
<td>56.6</td>
<td>55.8</td>
<td>57.0</td>
</tr>
<tr>
<td>51.6</td>
<td>50.8</td>
<td>52.0</td>
</tr>
<tr>
<td>55.0</td>
<td>54.2</td>
<td>56.0</td>
</tr>
<tr>
<td>50.5</td>
<td>49.7</td>
<td>51.0</td>
</tr>
<tr>
<td>51.6</td>
<td>50.8</td>
<td>52.0</td>
</tr>
<tr>
<td>54.6</td>
<td>53.8</td>
<td>55.0</td>
</tr>
<tr>
<td>53.2</td>
<td>52.4</td>
<td>54.0</td>
</tr>
<tr>
<td>55.5</td>
<td>54.7</td>
<td>56.0</td>
</tr>
<tr>
<td>57.8</td>
<td>57.0</td>
<td>58.0</td>
</tr>
<tr>
<td>57.0</td>
<td>56.2</td>
<td>58.0</td>
</tr>
<tr>
<td>55.5</td>
<td>54.7</td>
<td>56.0</td>
</tr>
</tbody>
</table>

Mark 25
```
Squared coherency test statistic = 0.1926

<table>
<thead>
<tr>
<th>Value</th>
<th>Lower</th>
<th>Upper</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.9562</td>
<td>0.9124</td>
</tr>
<tr>
<td>1</td>
<td>0.9539</td>
<td>0.9079</td>
</tr>
<tr>
<td>2</td>
<td>0.9567</td>
<td>0.9134</td>
</tr>
<tr>
<td>3</td>
<td>0.9591</td>
<td>0.9181</td>
</tr>
<tr>
<td>4</td>
<td>0.9428</td>
<td>0.8864</td>
</tr>
<tr>
<td>5</td>
<td>0.9048</td>
<td>0.8148</td>
</tr>
<tr>
<td>6</td>
<td>0.8738</td>
<td>0.7586</td>
</tr>
<tr>
<td>7</td>
<td>0.8715</td>
<td>0.7545</td>
</tr>
<tr>
<td>8</td>
<td>0.8453</td>
<td>0.7087</td>
</tr>
<tr>
<td>9</td>
<td>0.8198</td>
<td>0.6654</td>
</tr>
<tr>
<td>10</td>
<td>0.7770</td>
<td>0.5956</td>
</tr>
<tr>
<td>11</td>
<td>0.7033</td>
<td>0.4838</td>
</tr>
<tr>
<td>12</td>
<td>0.6541</td>
<td>0.4152</td>
</tr>
<tr>
<td>13</td>
<td>0.5702</td>
<td>0.3090</td>
</tr>
<tr>
<td>14</td>
<td>0.4474</td>
<td>0.1786</td>
</tr>
<tr>
<td>15</td>
<td>0.3945</td>
<td>0.1318</td>
</tr>
<tr>
<td>16</td>
<td>0.3260</td>
<td>0.0802</td>
</tr>
<tr>
<td>17</td>
<td>0.1887</td>
<td>0.0115</td>
</tr>
<tr>
<td>18</td>
<td>0.0390</td>
<td>0.0000</td>
</tr>
<tr>
<td>19</td>
<td>0.0618</td>
<td>0.0000</td>
</tr>
<tr>
<td>20</td>
<td>0.0524</td>
<td>0.0000</td>
</tr>
<tr>
<td>21</td>
<td>0.0428</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>22</td>
<td>0.0578</td>
<td>0.0000</td>
</tr>
<tr>
<td>23</td>
<td>0.0282</td>
<td>0.0000</td>
</tr>
<tr>
<td>24</td>
<td>0.0220</td>
<td>0.0000</td>
</tr>
<tr>
<td>25</td>
<td>0.0807</td>
<td>0.0000</td>
</tr>
<tr>
<td>26</td>
<td>0.0754</td>
<td>0.0000</td>
</tr>
<tr>
<td>27</td>
<td>0.0167</td>
<td>0.0000</td>
</tr>
<tr>
<td>28</td>
<td>0.0331</td>
<td>0.0000</td>
</tr>
<tr>
<td>29</td>
<td>0.0777</td>
<td>0.0000</td>
</tr>
<tr>
<td>30</td>
<td>0.1414</td>
<td>0.0014</td>
</tr>
<tr>
<td>31</td>
<td>0.0739</td>
<td>0.0000</td>
</tr>
<tr>
<td>32</td>
<td>0.0107</td>
<td>0.0000</td>
</tr>
<tr>
<td>33</td>
<td>0.0008</td>
<td>0.0000</td>
</tr>
<tr>
<td>34</td>
<td>0.0488</td>
<td>0.0000</td>
</tr>
<tr>
<td>35</td>
<td>0.0760</td>
<td>0.0000</td>
</tr>
<tr>
<td>36</td>
<td>0.0738</td>
<td>0.0000</td>
</tr>
<tr>
<td>37</td>
<td>0.0719</td>
<td>0.0000</td>
</tr>
<tr>
<td>38</td>
<td>0.0438</td>
<td>0.0000</td>
</tr>
<tr>
<td>39</td>
<td>0.0168</td>
<td>0.0000</td>
</tr>
<tr>
<td>40</td>
<td>0.0183</td>
<td>0.0000</td>
</tr>
</tbody>
</table>