

## NAG Library Function Document

### nag\_tsa\_spectrum\_univar\_cov (g13cac)

#### 1 Purpose

nag\_tsa\_spectrum\_univar\_cov (g13cac) calculates the smoothed sample spectrum of a univariate time series using one of four lag windows – rectangular, Bartlett, Tukey or Parzen window.

#### 2 Specification

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

void nag_tsa_spectrum_univar_cov (Integer nx, Integer mtx, double px,
    Integer iw, Integer mw, Integer ic, Integer nc, double c[], Integer kc,
    Integer l, Nag_LoggedSpectra lg_spect, Integer nxg, double xg[],
    Integer *ng, double stats[], NagError *fail)
```

#### 3 Description

The smoothed sample spectrum is defined as

$$\hat{f}(\omega) = \frac{1}{2\pi} \left( C_0 + 2 \sum_{k=1}^{M-1} w_k C_k \cos(\omega k) \right),$$

where  $M$  is the window width, and is calculated for frequency values

$$\omega_i = \frac{2\pi i}{L}, \quad i = 0, 1, \dots, [L/2],$$

where  $[]$  denotes the integer part.

The autocovariances  $C_k$  may be supplied by you, or constructed from a time series  $x_1, x_2, \dots, x_n$ , as

$$C_k = \frac{1}{n} \sum_{t=1}^{n-k} x_t x_{t+k},$$

the fast Fourier transform (FFT) being used to carry out the convolution in this formula.

The time series may be mean or trend corrected (by classical least squares), and tapered before calculation of the covariances, the tapering factors being those of the split cosine bell:

$$\begin{aligned} & \frac{1}{2} \left( 1 - \cos\left(\pi\left(t - \frac{1}{2}\right)/T\right) \right), & 1 \leq t \leq T \\ & \frac{1}{2} \left( 1 - \cos\left(\pi\left(n - t + \frac{1}{2}\right)/T\right) \right), & n + 1 - T \leq t \leq n \\ & 1, & \text{otherwise,} \end{aligned}$$

where  $T = \left\lceil \frac{np}{2} \right\rceil$  and  $p$  is the tapering proportion.

The smoothing window is defined by

$$w_k = W\left(\frac{k}{M}\right), \quad k \leq M - 1,$$

which for the various windows is defined over  $0 \leq \alpha < 1$  by rectangular:

$$W(\alpha) = 1$$

Bartlett:

$$W(\alpha) = 1 - \alpha$$

Tukey:

$$W(\alpha) = \frac{1}{2}(1 + \cos(\pi\alpha))$$

Parzen:

$$W(\alpha) = 1 - 6\alpha^2 + 6\alpha^3, \quad 0 \leq \alpha \leq \frac{1}{2}$$

$$W(\alpha) = 2(1 - \alpha)^3, \quad \frac{1}{2} < \alpha < 1.$$

The sampling distribution of  $\hat{f}(\omega)$  is approximately that of a scaled  $\chi_d^2$  variate, whose degrees of freedom  $d$  is provided by the function, together with multiplying limits  $mu$ ,  $ml$  from which approximate 95% confidence intervals for the true spectrum  $f(\omega)$  may be constructed as  $[ml \times \hat{f}(\omega), mu \times \hat{f}(\omega)]$ . Alternatively,  $\log \hat{f}(\omega)$  may be returned, with additive limits.

The bandwidth  $b$  of the corresponding smoothing window in the frequency domain is also provided. Spectrum estimates separated by (angular) frequencies much greater than  $b$  may be assumed to be independent.

## 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: **nx** – Integer *Input*  
*On entry:*  $n$ , the length of the time series.  
*Constraint:*  $nx \geq 1$ .
- 2: **mtx** – Integer *Input*  
*On entry:* if covariances are to be calculated by the function (**ic** = 0), **mtx** must specify whether the data are to be initially mean or trend corrected.  
**mtx** = 0  
 For no correction.  
**mtx** = 1  
 For mean correction.  
**mtx** = 2  
 For trend correction.  
*Constraint:* if **ic** = 0,  $0 \leq \mathbf{mtx} \leq 2$   
 If covariances are supplied (**ic**  $\neq$  0), **mtx** is not used.
- 3: **px** – double *Input*  
*On entry:* if covariances are to be calculated by the function (**ic** = 0), **px** must specify the proportion of the data (totalled over both ends) to be initially tapered by the split cosine bell taper.  
 If covariances are supplied (**ic**  $\neq$  0), **px** must specify the proportion of data tapered before the supplied covariances were calculated and after any mean or trend correction. **px** is required for the calculation of output statistics. A value of 0.0 implies no tapering.  
*Constraint:*  $0.0 \leq \mathbf{px} \leq 1.0$ .

- 4: **iw** – Integer *Input*  
*On entry:* the choice of lag window.  
**iw** = 1  
 Rectangular.  
**iw** = 2  
 Bartlett.  
**iw** = 3  
 Tukey.  
**iw** = 4  
 Parzen.  
*Constraint:*  $1 \leq \mathbf{iw} \leq 4$ .
- 5: **mw** – Integer *Input*  
*On entry:*  $M$ , the ‘cut-off’ point of the lag window. Windowed covariances at lag  $M$  or greater are zero.  
*Constraint:*  $1 \leq \mathbf{mw} \leq \mathbf{nx}$ .
- 6: **ic** – Integer *Input*  
*On entry:* indicates whether covariances are to be calculated in the function or supplied in the call to the function.  
**ic** = 0  
 Covariances are to be calculated.  
**ic**  $\neq$  0  
 Covariances are to be supplied.
- 7: **nc** – Integer *Input*  
*On entry:* the number of covariances to be calculated in the function or supplied in the call to the function.  
*Constraint:*  $\mathbf{mw} \leq \mathbf{nc} \leq \mathbf{nx}$ .
- 8: **c[nc]** – double *Input/Output*  
*On entry:* if **ic**  $\neq$  0, **c** must contain the **nc** covariances for lags from 0 to (**nc** – 1), otherwise **c** need not be set.  
*On exit:* if **ic** = 0, **c** will contain the **nc** calculated covariances.  
 If **ic**  $\neq$  0, the contents of **c** will be unchanged.
- 9: **kc** – Integer *Input*  
*On entry:* if **ic** = 0, **kc** must specify the order of the fast Fourier transform (FFT) used to calculate the covariances.  
 If **ic**  $\neq$  0, that is covariances are supplied, **kc** is not used.  
*Constraint:*  $\mathbf{kc} \geq \mathbf{nx} + \mathbf{nc}$ .
- 10: **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 covariances.  
*Constraint:*  $\mathbf{l} \geq 2 \times \mathbf{mw} - 1$ .

- 11: **lg\_spect** – Nag\_LoggedSpectra *Input*  
*On entry:* indicates whether unlogged or logged spectral estimates and confidence limits are required.  
**lg\_spect** = Nag\_Unlogged  
 Unlogged.  
**lg\_spect** = Nag\_Logged  
 Logged.  
*Constraint:* **lg\_spect** = Nag\_Unlogged or Nag\_Logged.
- 12: **nxg** – Integer *Input*  
*On entry:* the dimension of the array **xg**.  
*Constraints:*  
 if **ic** = 0, **nxg** ≥ max(**kc**, **l**);  
 if **ic** ≠ 0, **nxg** ≥ **l**.
- 13: **xg[nxg]** – double *Input/Output*  
*On entry:* if the covariances are to be calculated, then **xg** must contain the **nx** data points. If covariances are supplied, **xg** may contain any values.  
*On exit:* contains the **ng** spectral estimates,  $\hat{f}(\omega_i)$ , for  $i = 0, 1, \dots, [L/2]$  in **xg**[0] to **xg**[**ng** – 1] respectively (logged if **lg\_spect** = Nag\_Logged). The elements **xg**[ $i - 1$ ], for  $i = \mathbf{ng} + 1, \dots, \mathbf{nxg}$  contain 0.0.
- 14: **ng** – Integer \* *Output*  
*On exit:* the number of spectral estimates,  $[L/2] + 1$ , in **xg**.
- 15: **stats**[4] – double *Output*  
*On exit:* four associated statistics. These are the degrees of freedom in **stats**[0], the lower and upper 95% confidence limit factors in **stats**[1] and **stats**[2] respectively (logged if **lg\_spect** = Nag\_Logged), and the bandwidth in **stats**[3].
- 16: **fail** – NagError \* *Input/Output*  
 The NAG error argument (see Section 2.7 in How to Use the NAG Library and its Documentation).

## 6 Error Indicators and Warnings

### NE\_ALLOC\_FAIL

Dynamic memory allocation failed.

See Section 2.3.1.2 in How to Use the NAG Library and its Documentation for further information.

### NE\_BAD\_PARAM

On entry, argument  $\langle value \rangle$  had an illegal value.

### NE\_CONFID\_LIMITS

The calculation of confidence limit factors has failed.

Spectral estimates (logged if requested) are returned in **xg**, and degrees of freedom and bandwidth in **stats**.

**NE\_INT**

On entry, **ic** = 0 and **mtx** < 0: **mtx** =  $\langle value \rangle$ .

On entry, **ic** = 0 and **mtx** > 2: **mtx** =  $\langle value \rangle$ .

On entry, **iw** =  $\langle value \rangle$ .

Constraint: **iw** = 1, 2, 3 or 4.

On entry, **mw** =  $\langle value \rangle$ .

Constraint: **mw**  $\geq$  1.

On entry, **nx** =  $\langle value \rangle$ .

Constraint: **nx**  $\geq$  1.

**NE\_INT\_2**

On entry, **l** =  $\langle value \rangle$  and **mw** =  $\langle value \rangle$ .

Constraint: **l**  $\geq$  2  $\times$  **mw** - 1.

On entry, **mw** =  $\langle value \rangle$  and **nx** =  $\langle value \rangle$ .

Constraint: **mw**  $\leq$  **nx**.

On entry, **nc** =  $\langle value \rangle$  and **mw** =  $\langle value \rangle$ .

Constraint: **nc**  $\geq$  **mw**.

On entry, **nc** =  $\langle value \rangle$  and **nx** =  $\langle value \rangle$ .

Constraint: **nc**  $\leq$  **nx**.

On entry, **nxg** =  $\langle value \rangle$  and **l** =  $\langle value \rangle$ .

Constraint: if **ic**  $\neq$  0, **nxg**  $\geq$  **l**.

**NE\_INT\_3**

On entry, **kc** =  $\langle value \rangle$ , **nx** =  $\langle value \rangle$  and **nc** =  $\langle value \rangle$ .

Constraint: if **ic** = 0, **kc**  $\geq$  (**nx** + **nc**).

On entry, **nxg** =  $\langle value \rangle$ , **kc** =  $\langle value \rangle$  and **l** =  $\langle value \rangle$ .

Constraint: if **ic** = 0, **nxg**  $\geq$  max(**kc**, **l**).

**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 2.7.6 in How to Use the NAG Library and its Documentation for further information.

**NE\_NO\_LICENCE**

Your licence key may have expired or may not have been installed correctly.

See Section 2.7.5 in How to Use the NAG Library and its Documentation for further information.

**NE\_REAL**

On entry, **px** =  $\langle value \rangle$ .

Constraint: **px**  $\leq$  1.0.

On entry, **px** =  $\langle value \rangle$ .

Constraint: **px**  $\geq$  0.0.

**NE\_SPECTRAL\_ESTIMATES**

One or more spectral estimates are negative.

Unlogged spectral estimates are returned in **xg**, and the degrees of freedom, unlogged confidence limit factors and bandwidth in **stats**.

## 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

`nag_tsa_spectrum_univar_cov` (g13cac) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

`nag_tsa_spectrum_univar_cov` (g13cac) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the x06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this function. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

## 9 Further Comments

`nag_tsa_spectrum_univar_cov` (g13cac) carries out two FFTs of length `kc` to calculate the covariances and one FFT of length `l` to calculate the sample spectrum. The time taken by the function 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 a time series of length 256. It selects the mean correction option, a tapering proportion of 0.1, the Parzen smoothing window and a cut-off point for the window at lag 100. It chooses to have 100 auto-covariances calculated and unlogged spectral estimates at a frequency division of  $2\pi/200$ . It then calls `nag_tsa_spectrum_univar_cov` (g13cac) to calculate the univariate spectrum and statistics and prints the autocovariances and the spectrum together with its 95% confidence multiplying limits.

### 10.1 Program Text

```

/* nag_tsa_spectrum_univar_cov (g13cac) Example Program.
 *
 * NAGPRODCODE Version.
 *
 * Copyright 2016 Numerical Algorithms Group.
 *
 * Mark 26, 2016.
 */

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

int main(void)
{
    /* Scalars */
    double px;
    Integer exit_status, i, ic, iw, kc, lf, mtx, mw, nc, ng, nx, nxg;
    NagError fail;

    /* Arrays */
    double *c = 0, *xg = 0;
    double stats[4];

    INIT_FAIL(fail);

    exit_status = 0;

```

```

printf("nag_tsa_spectrum_univar_cov (g13cac) 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 "%*[\n] ", &nx, &nc);
#else
scanf("%" NAG_IFMT "%" NAG_IFMT "%*[\n] ", &nx, &nc);
#endif

if (nx > 0 && nc > 0) {
    mtx = 1;
    px = 0.1;
    iw = 4;
    mw = 100;
    ic = 0;
    kc = 360;
    lf = 200;

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

    /* Allocate memory */
    if (!(c = NAG_ALLOC(nc, double)) || !(xg = NAG_ALLOC(nxg, double)))
    {
        printf("Allocation failure\n");
        exit_status = -1;
        goto END;
    }

    for (i = 1; i <= nx; ++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

    /* nag_tsa_spectrum_univar_cov (g13cac).
     * Univariate time series, smoothed sample spectrum using
     * rectangular, Bartlett, Tukey or Parzen lag window
     */
    nag_tsa_spectrum_univar_cov(nx, mtx, px, iw, mw, ic, nc, c, kc, lf,
                               Nag_Unlogged, nxg, xg, &ng, stats, &fail);
    if (fail.code != NE_NOERROR) {
        printf("Error from nag_tsa_spectrum_univar_cov (g13cac).\n%s\n",
              fail.message);
        exit_status = 1;
        goto END;
    }

    printf("\n");

    printf("Covariances\n");
    for (i = 1; i <= nc; ++i) {
        printf("%11.4f", c[i - 1]);
        if (i % 6 == 0 || i == nc)
            printf("\n");
    }
}

```

```

printf("\n");
printf("Degrees of freedom =%4.1f      Bandwidth =%7.4f\n",
      stats[0], stats[3]);
printf("\n");
printf("95 percent confidence limits -      Lower =%7.4f  "
      "Upper =%7.4f\n", stats[1], stats[2]);
printf("\n");
printf("      Spectrum      Spectrum      Spectrum" "      Spectrum\n");
printf("      estimate      estimate      estimate" "      estimate\n");
for (i = 1; i <= ng; ++i) {
    printf("%4" NAG_IFMT "%10.4f", i, xg[i - 1]);
    if (i % 4 == 0 || i == ng)
        printf("\n");
}
}
END:
NAG_FREE(c);
NAG_FREE(xg);

return exit_status;
}

```

## 10.2 Program Data

nag\_tsa\_spectrum\_univar\_cov (g13cac) Example Program Data

```

256 100
 5.0  11.0  16.0  23.0  36.0  58.0  29.0  20.0  10.0  8.0  3.0  0.0
 0.0  2.0  11.0  27.0  47.0  63.0  60.0  39.0  28.0  26.0  22.0  11.0
21.0  40.0  78.0 122.0 103.0  73.0  47.0  35.0  11.0  5.0  16.0  34.0
70.0  81.0 111.0 101.0  73.0  40.0  20.0  16.0  5.0  11.0  22.0  40.0
60.0  80.9  83.4  47.7  47.8  30.7  12.2  9.6  10.2  32.4  47.6  54.0
62.9  85.9  61.2  45.1  36.4  20.9  11.4  37.8  69.8 106.1 100.8  81.6
66.5  34.8  30.6  7.0  19.8  92.5 154.4 125.9  84.8  68.1  38.5  22.8
10.2  24.1  82.9 132.0 130.9 118.1  89.9  66.6  60.0  46.9  41.0  21.3
16.0  6.4  4.1  6.8  14.5  34.0  45.0  43.1  47.5  42.2  28.1  10.1
 8.1  2.5  0.0  1.4  5.0  12.2  13.9  35.4  45.8  41.1  30.1  23.9
15.6  6.6  4.0  1.8  8.5  16.6  36.3  49.6  64.2  67.0  70.9  47.8
27.5  8.5  13.2  56.9 121.5 138.3 103.2  85.7  64.6  36.7  24.2  10.7
15.0  40.1  61.5  98.5 124.7  96.3  66.6  64.5  54.1  39.0  20.6  6.7
 4.3  22.7  54.8  93.8  95.8  77.2  59.1  44.0  47.0  30.5  16.3  7.3
37.6  74.0 139.0 111.2 101.6  66.2  44.7  17.0  11.3  12.4  3.4  6.0
32.3  54.3  59.7  63.7  63.5  52.2  25.4  13.1  6.8  6.3  7.1  35.6
73.0  85.1  78.0  64.0  41.8  26.2  26.7  12.1  9.5  2.7  5.0  24.4
42.0  63.5  53.8  62.0  48.5  43.9  18.6  5.7  3.6  1.4  9.6  47.4
57.1 103.9  80.6  63.6  37.6  26.1  14.2  5.8  16.7  44.3  63.9  69.0
77.8  64.9  35.7  21.2  11.1  5.7  8.7  36.1  79.7 114.4 109.6  88.8
67.8  47.5  30.6  16.3  9.6  33.2  92.6 151.6 136.3 134.7  83.9  69.4
31.5  13.9  4.4  38.0

```

## 10.3 Program Results

nag\_tsa\_spectrum\_univar\_cov (g13cac) Example Program Results

Covariances

```

1152.9733  937.3289  494.9243  14.8648  -342.8548  -514.6479
-469.2733  -236.6896  109.0608  441.3498  637.4571  641.9954
 454.0505  154.5960  -136.8016  -343.3911  -421.8441  -374.4095
-241.1943  -55.6140  129.4067  267.4248  311.8293  230.2807
 56.4402  -146.4689  -320.9948  -406.4077  -375.6384  -273.5936
-132.6214  11.0791  126.4843  171.3391  122.6284  -11.5482
-169.2623  -285.2358  -331.4567  -302.2945  -215.4832  -107.8732
 -3.4126  73.2521  98.0831  71.8949  17.0985  -27.5632
 -76.7900  -110.5354  -126.1383  -121.1043  -103.9362  -67.4619
 -10.8678  58.5009  116.4587  140.0961  129.5928  66.3211
 -35.5487  -135.3894  -203.7149  -216.2161  -152.7723  -30.4361
 99.3397  188.9594  204.9047  148.4056  34.4975  -103.7840
-208.5982  -252.4128  -223.7600  -120.8640  23.3565  156.0956
 227.7642  228.5123  172.3820  87.4911  -21.2170  -117.5282
-176.3634  -165.1218  -75.1308  67.1634  195.7290  279.3039

```



290.8258 225.3811 104.0784 -44.4731 -162.7355 -207.7480  
 -165.2444 -48.5473 118.8872 265.0045

Degrees of freedom = 9.0 Bandwidth = 0.1165

95 percent confidence limits - Lower = 0.4731 Upper = 3.3329

	Spectrum estimate		Spectrum estimate		Spectrum estimate		Spectrum estimate
1	210.4696	2	428.2020	3	810.1419	4	922.5900
5	706.1605	6	393.4052	7	207.6481	8	179.0657
9	170.1320	10	133.0442	11	103.6752	12	103.0644
13	141.5173	14	194.3041	15	266.5730	16	437.0181
17	985.3130	18	2023.1574	19	2681.8980	20	2363.7439
21	1669.9001	22	1012.1320	23	561.4822	24	467.2741
25	441.9977	26	300.1985	27	172.0184	28	114.7823
29	79.1533	30	49.4882	31	27.0902	32	16.8081
33	27.5111	34	59.4429	35	97.0145	36	119.3664
37	116.6737	38	87.3142	39	54.9570	40	42.9781
41	46.6097	42	53.6206	43	50.6050	44	36.7780
45	25.6285	46	24.8555	47	30.2626	48	31.5642
49	27.3351	50	22.4443	51	18.5418	52	15.2425
53	12.0207	54	12.6846	55	18.3975	56	19.3058
57	12.6103	58	7.9511	59	7.1333	60	5.4996
61	3.4182	62	3.2359	63	5.3836	64	8.5225
65	10.0610	66	7.9483	67	4.2261	68	3.2631
69	5.5751	70	7.8491	71	9.3694	72	11.0791
73	10.1386	74	6.3158	75	3.6375	76	2.6561
77	1.8026	78	1.0103	79	1.0693	80	2.3950
81	4.0822	82	4.6221	83	4.0672	84	3.8460
85	4.8489	86	6.3964	87	6.4762	88	4.9457
89	4.4444	90	5.2131	91	5.0389	92	4.6141
93	5.8722	94	7.9268	95	7.9486	96	5.7854
97	4.5495	98	5.2696	99	6.3893	100	6.5216
101	6.2129						

---