The **ghosts** package

Allin Cottrell and Jack Lucchetti

version 1.0

**Abstract**

The name of this package is an acronym, that stands for *Gretl/Hansl Offering for Spectra of Time Series* and gives the user the most common tools for spectral analysis of time series.

**Contents**

1 Spectral density estimation 2
   1.1 Detrending .................................. 2
   1.2 Tapering .................................. 2
   1.3 Smoothing .................................. 2
   1.4 So, in practice............................ 4
   1.5 An example ................................. 4
   1.6 The GUI interface ......................... 6

2 Alphabetical list of functions 7

3 Changelog 8
1 Spectral density estimation

In short: you create a list holding the series you want the multivariate spectrum for. Then you have to make a decision on (a) detrending (b) tapering and (c) smoothing. Apart from detrending and tapering, smoothing is especially important because normally you don’t want the raw periodogram (which is, as well known, inconsistent). More on this later.

When you’ve made your decision, you can compute the spectrum via something like

$$\text{bundle } \text{Spec} = \text{mvspec}(X)$$

Optional arguments are used for fancier choices (see below).

Internally, we use the Fast Fourier Transform as implemented in Frigo and Johnson (2005), which yields excellent numerical performance. After completion, you’ll be able to generate all the usual pretty plots by using the generated bundle as the main ingredient.

1.1 Detrending

Ideally, your data should have zero mean and contain no trends. You have the choice of the package doing some of the work for you. The fifth, optional, argument to the \texttt{mvspec} function can be 0, 1 or 2; these correspond, respectively, to “do nothing”, “remove the mean”, “remove a linear trend”. The default is 2.

1.2 Tapering

Tapering amounts to “fading in” the first observations and “fading out” the last ones. A nice explanation of why this is a good idea is in Bloomfield (2000), sec. 6.2.

The data are tapered as in

$$\tilde{x}_t = w_t x_t$$

where

$$w_t = \begin{cases} 
1 - \cos\left(\frac{\pi (t+0.5)}{m}\right) & t < m \\
1 - \cos\left(\frac{2 \pi (T-t+0.5)}{2}ight) & t > T - m \\
1 & \text{otherwise}
\end{cases}$$

An example of the tapering filter weights is shown in Figure 1.

To toggle the value of \(m\), the argument you pass to the \texttt{mvspec} function is a scalar \(\alpha\), which is interpreted as the ratio \(m/T\). If what you want, for whatever reason, is the raw periodogram, be sure turn tapering off by setting \(\alpha\) (and hence, \(m\)) to 0.

1.3 Smoothing

Unless you want the raw periodogram, a modified Daniell kernel is used to smooth the spectrum. This is computed via a vector of \textit{spans}, which are odd integers. Each element of this vector indicates the order of a moving average operator. The first element is applied to the raw data, the second (if present)
Tapering with $T = 200$ and $m = 20$

is applied to the result of the first one, and so on. In short, the result of the operation is a convolution of moving averages.

For example, suppose you have that the vector `spans` is $(5, 7, 3)$. First you take a 5-terms centred moving average of the data, then you compute a 7-terms centred moving average of the result and finally you compute a 3-terms centred moving average of that.\(^1\) An example is shown in Figure 2.

The `spans` argument must be specified as the second one, right after the list containing the series. If omitted, an automatic choice will be used, defined as a vector of $n$ elements, all equal to $2n + 1$. The value $n$ is computed as

$$n = \left\lceil \frac{\log_2 T}{3} \right\rceil,$$

where $T$ is the sample size. So for example if you have 100 observations, the `spans` vector will be $(7, 7, 7)$, but if you have 1000 `spans` will equal $(9, 9, 9, 9)$. You turn smoothing off by setting `spans` to 0.

\(^1\)This would be using a plain Daniell kernel; what we use is in fact the modified Daniell kernel gives half weights to the endpoints: for example, a 5-terms moving average would have coefficients $(1/8, 1/4, 1/4, 1/4, 1/8)$, as in $\bar{x}_t = \frac{x_t - 2x_{t+2} + x_{t+4}}{8} + \frac{x_{t-1} + x_{t+1} + x_{t+2}}{4}$. 

3
1.4 So, in practice...

So, in practice, when you issue the command `bundle S = mvspec(X,spans)`, for each series in the list `X`:

1. the series is pre-processed (mean removal, or detrending);
2. the result gets tapered; the default value for $\alpha$ is 0.1.
3. the FFT is computed and smoothed by using the `spans` argument;
4. the spectral density estimate at density $\lambda$ is the outer product of the corresponding element of the smoothed FFT; this is returned as the bundle element `spectrum`.

Each row of `spectrum` contains the estimated spectrum at frequency $\omega$, in “vech” form. Nota bene: the multivariate spectrum is a sequence of complex matrices, one per frequency. The new version of `ghosts` uses the new facility, introduced in `gretl` 2019d whereby complex matrices are natively supported.

For displaying the customary plots, you can use the three plotting functions `spplot`, `coplot` and `phplot` for plotting a spectrum, a (squared) coherence, or a phase plot, respectively. See the example below.

1.5 An example

In this example, we load three time series from the St. Louis FED database and compute the multivariate spectrum. Then, we produce a few plots.

```gretl
include ghosts.gfn

### load the data
clear
open fedstl.bin
data houstnsa
data paynsa
data unratensa

### perform a few basic transformations
series l_house_start = log(houstnsa)
series g_payroll = 100 * ldiff(paynsa)
series d_unemp = diff(unratensa)

list L = l_house_start g_payroll d_unemp
spans = {11,11} # set to 0 to disable smoothing
taper = 0.1    # percentage of tapered obs
smpl +1 ;

### compute the spectrum
zoo = mvspec(L, spans, taper)
```

The code above will fill the bundle `zoo` with all the relevant quantities. If you want to have a look at the estimated spectrum, you will need to inspect the `spectrum` element. For example, the code
spec = zoo.spectrum
omega = zoo.freq
loop i = 1 .. 5 --quiet
  S = unvech(spec[i,])'
  printf "spectrum at %g (pi * %g)\n", omega[i], omega[i]/$pi
  print S
endloop

will produce the following output:

spectrum at 0 (pi * 0)
S (3 x 3)
0.0387 + 0.0000i  0.0021 + 0.0000i  -0.0004 - 0.0000i
0.0021 - 0.0000i  0.0036 + 0.0000i  -0.0017 - 0.0000i
-0.0004 + 0.0000i  -0.0017 + 0.0000i  0.0011 + 0.0000i

spectrum at 0.00867843 (pi * 0.00276243)
S (3 x 3)
0.0386 + 0.0000i  0.0021 + 0.0010i  -0.0005 - 0.0007i
0.0021 - 0.0010i  0.0036 + 0.0000i  -0.0017 - 0.0002i
-0.0005 + 0.0007i  -0.0017 + 0.0002i  0.0011 + 0.0000i

spectrum at 0.0173569 (pi * 0.00552486)
S (3 x 3)
0.0380 + 0.0000i  0.0024 + 0.0019i  -0.0007 - 0.0013i
0.0024 - 0.0019i  0.0039 + 0.0000i  -0.0019 - 0.0003i
-0.0007 + 0.0013i  -0.0019 + 0.0003i  0.0012 + 0.0000i

spectrum at 0.0260353 (pi * 0.00828729)
S (3 x 3)
0.0369 + 0.0000i  0.0028 + 0.0025i  -0.0009 - 0.0019i
0.0028 - 0.0025i  0.0041 - 0.0000i  -0.0020 - 0.0004i
-0.0009 + 0.0019i  -0.0020 + 0.0004i  0.0013 + 0.0000i

spectrum at 0.0347137 (pi * 0.0110497)
S (3 x 3)
0.0350 - 0.0000i  0.0034 + 0.0030i  -0.0013 - 0.0023i
0.0034 - 0.0030i  0.0045 - 0.0000i  -0.0022 - 0.0006i
-0.0013 + 0.0023i  -0.0022 + 0.0006i  0.0014 + 0.0000i

Note that at frequency 0 the spectrum is a real matrix, but for all other frequencies you get a Hermitian complex matrix. In principle, you can produce the customary spectrum, coherence and phase plots by extracting the relevant columns of the spectrum, but the ghosts package provides three convenience functions to make your life easier.

The code below produces the three graphs shown in figure 3.

logscale = 1
spplot(zoo, 1, logscale)
1.6 The GUI interface

As a convenience to the user, a GUI interface is provided to access some of the functionality provided by the *ghosts* package. The GUI hook will be found under the *Model > Time Series > Multivariate spectra* menu entry.

The GUI interface is meant to produce one graph at a time, and therefore the user has the choice between a spectrum plot, a coherence plot and a phase plot (note that for a spectrum plot, the second series argument has no effect). Figure 4 shows the dialog.

```r
coplot(zoo, 1, 2)
phplot(zoo, 3, 2)
```

for a full description of the three graphical functions, go to Section 2.

More examples can be found in the *examples* subdirectory to the *ghosts* package.
2 Alphabetical list of functions

function void coplot(bundle S, scalar i, scalar j)

Plots the squared coherence between two variables.

1. S: bundle created by mvspec
2. i,j: progressive number of series for which the plot is wanted

function bundle mvspec(list X, matrix spans,
scalar taper, int prefilt)

1. X: input list
2. spans: vector of odd integers, used for periodogram smoothing; set to 0
   for no smoothing. Optional: if omitted, the default is to use a vector of \(n\)
elements, each equal to \(2n + 1\), where \(n\) is computed as per equation (1).
3. taper: amount of tapering; optional, default 10%.
4. prefilt: integer. 0 for no prefiltering, 1 for mean removal, 2 for trend
   removal. Default=2.

function void phplot(bundle S, scalar i, scalar j, bool cont)
Plots the phase plot between two variables.

1. S: bundle created by \texttt{mvspec}
2. i,j: pregressive number of series for which the plot is wanted
3. cont: Boolean, force continuity of the phase plot (default=yes)

\begin{verbatim}
function void spplot(bundle S, scalar i, int logscale)
\end{verbatim}

Plots the spectrum for one variable.

1. S: bundle created by \texttt{mvspec}
2. i: pregressive number of series for which the spectrum is wanted
3. logscale: an integer: 0 = no logs, 1 = log scale on the Y axis, 2 = log scale on the X axis, 3 = both (default = 1)

3 Changelog

- 0.3 → 0.4: remove the dependency on the \texttt{cmatrix} package in view of gretl’s new built-in complex matrix support.

- 0.2 → 0.3: require dependency on the \texttt{cmatrix} package; consequently, remove from the package all complex-matrix functions.

- 0.12 → 0.2: add option for log xscale in spectrum plot; GUI hook; \texttt{ceigg} and \texttt{clog} functions.

- 0.11 → 0.12: make the \texttt{spans} argument to \texttt{mvspec} optional and hush a warning.

References
