Chapter 15: Business cycle analysis

Grocer contains several tools used for the business cycle analysis: usual filters (part 1); Bry –Boschan and Harding-Pagan turning point determination and the Banerji test of the lead of one series with respect to another one (part 2); spectral analysis (part 3)[1].

1. The HP, BK and CF filters.

Since Hodrick and Prescott have proposed the so called HP filter[2], there has been much research on the subject and 2 other filters have emerged: the Baxter-King filter[3] and the Christiano-Fitzgerald filter[4]. These 3 filters are available in Grocer, under the respective names: hpfiler, bkfilter and cffilter.

- The syntax of the hpfilter is as follows :

ynew=hpfilter (yold, lambda)

yold can be a vector or a timeseries: ynew will have the same type and, if yold is a ts, then ynew will have the same time-span as yold. Lambda is the smoothing parameter, which determines the relative variance of the filtered series, ynew and the "cycle" yold-ynew.

- The syntax of the bkfilter is as follows:

[ynew,AA]=bkfilter(yold,pl,pu,nfix,typ)

with:

- yold can be a vector or a timeseries: ynew will have the same type and, if yold is a ts, then ynew will have the same time-span as yold;
- typ is a string representing the type of the used filter. If typ is set to 'fixed', then the filter is the original Baxter-King that uses a filter of fixed length. If typ is set to 'vari', then the length of the filter is adapted at each point, in order to take into account the maximum number of observations while being symmetric around the considered observation: for instance, the tenth filtered value is based upon the ten first observations and the nine following ones. This calculation allows one to filter the series from the second until the last but one observations, while the classical filter looses, at the start of the period, a number of observations equal to the –fixed– length of the filter. typ is on optional argument: if omitted, it is set to 'fixed';
- pl and pu are the lower and upper periods that define the cycle (6 and 32 for business cycle frequencies on quarterly data for instance);
- nfix is the length of the filter, when typ is set to 'fixed'; the number of discarded observations (at least one) if the filter is of variable length;
- ynew is the filtered series;
- AA is the matrix applied to yold to obtain ynew.

The syntax of the Christiano- Fitzgerald filter is as follows:

[ynew,AA]=cffilter(yold,pl,pu,[options])

- yold can be a vector or a timeseries: ynew will have the same type and, if yold is a ts, then ynew will have the same time-span as yold;

[1] Except for the filters, all programs presented in this chapter have been written by Emmanuel Michaux.

[2] See Hodrick, R.J. and E.C. Prescott (1997): "Post-war US Business Cycles: an Empirical Investigation", Journal of Money, Credit and Banking, 29, 1-16.

[3] See Baxter, M. and R.G. King (1999) : "Measuring Business Cycles: approximate bandpass filters for Economic Time Series", Review of Economics and Statistics, 81, 575-93.

[4] See Christiano L. and Terry J. Fitzgerald (2003) : "The band pass filter", International Economic Review", Volume 44, Issue 2, May.

- pl and pu are the lower and upper periods that define the cycle (6 and 32 for business cycle periods on quarterly data for instance);
- options can be:
  - 'root=0' or 'root=1' whether you want to assume 0 or 1 unit root in time series (default: root = 1);
  - 'drift=0' or 'drift=1' whether you want to assume no drift or a time trend in time series (default: drift = 1);
  - filt= 'asym' or 'sym' or 'flsym' whether you want to use the asymmetric Filter, the symmetric Filter or the Fixed Length Symmetric Filter (default='asym')
  - 'nfix=xx' where xx is the fixed length (useful only for the symmetric filter, that is filt=flsym; default: nfix=-1).
  - thet = a (nx1) vector corresponding to the MA representation of x (if root=0) or delta(x) (if root =1; default: theta=1)
- ynew is the filtered series;
- AA is the matrix applied to yold to obtain ynew.

You can find an example in the file filters_d.sci in the 'macros/grocer/filters' sub-directory of your Scilab directory. This example uses the quarterly data for the german GDP from the second quarter of 1960 to the fourth one of 2002.

```
-->load('SCI/scied/grocer/v0.98/filters/pibrfa.dat')
```

// load the data

```
-->deu_gdp_hp=hpfilter(deu_gdp,1600);
```

// use the hp filter with the classical value of 1600

```
-->deu_gdp_bk12=bkfilter(deu_gdp,6,32,12,'fixed');
```

// use the fixed length Baxter-King filter for business cycle frequencies (6 to 32 quarters) and with length=12

```
-->deu_gdp_bkasym=bkfilter(deu_gdp,6,32,1,'vari');
```

// use the variable length Baxter-King filter for business cycle frequencies (6 to 32 quarters) and discard
// only the first and last observations

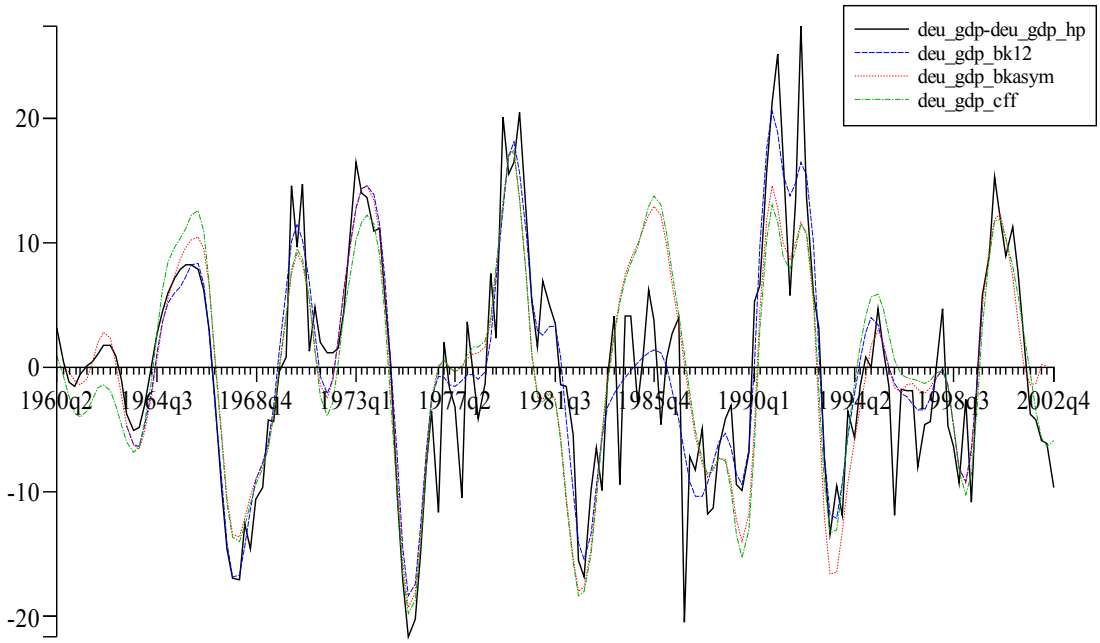
```
-->deu_gdp_cff=cffilter(deu_gdp,6,32);
```

// use the Christiano-Fitzgerald filter for business cycle frequencies (6 to 32 quarters)

The corresponding results can be graphed as follows (see graph on the next page):

```
-->pltseries('deu_gdp-deu_gdp_hp','deu_gdp_bk12','deu_gdp_bkasym','deu_gdp_cff',
'bounds=[''1960q2'';''2002q4'']','yaxex=0','title=filters on DEU GDP')
```

filters on DEU GDP



2. Turning point determination. (*)

The datation of peaks and troughs of the business cycles is still a hand-made procedure where human judgment remains prominent (see for instance how the NBER datation committee sets the dates of the American recessions). There are however a few computerized approaches that performs a good job at providing systematized datations. Two of these approaches have been implemented in grocer (by Emmanuel Michaux) in a function called brybos: this function provides the computerized approach proposed by Bry and Boschan[5] to mimic the NBER procedure on monthly time series and the adaptation of this method to quarterly time series that has been proposed by Harding and Pagan[6].

By definition, these procedures apply to time series: the first argument of brybos is therefore a time series; if the data you are working with is a vector, then transform it into a time series by the function reshape. (see chapter 3). The simplest call to brybos is therefore the following:

```
-->rbb1 = brybos('series')
```

[5] G. Bry & C. Boschan (1971), "Cyclical Analysis of Time Series: Selected Procedures and Computer Programs", NBER Technical Paper n°20.

[6] D. Harding & A. Pagan (2002), "Dissecting the Cycle: a Methodological Investigation", Journal of Monetary Economics, n°49, pp. 365-381

But the user can enter, if she wants, any of the following options:

- The choice of the procedure between Bry-Boschan and Harding-Pagan is given by the option 'proc ="bb'" and 'proc ="hp'" respectively. Default is Harding-Pagan.
- The minimal duration between two consecutive peaks can be given by the option 'M=n'; for instance imposing a minimal duration of 20 time units between the peaks and between troughs involves the option 'M=20'. Default value is 15 for monthly data and 5 for quarterly data.
- The minimal number of periods separating a turn form borders can be given by the option 'e=n'. For instance imposing that a peak or a trough must not begin before time unit n° 10 involves the option 'e=10'. Default value is 5 for monthly data and 2 for quarterly data.
- The minimal phase between a peak and a trough and conversely can be given by the option 'm=n'; for instance imposing a minimal duration of 10 time units between a peak and a trough involves the option 'M=10'. Default value is 6 for monthly data and 2 for quarterly data.
- The number n, determining a two-sided interval over which peaks and troughs are defined as local extrema, can be given by the option 'k=n'; for instance imposing that a peak and a trough must be a local maximum or minimum with respect to the 10 preceding and following months is given by the option 'k=10'. Default value is 5 for monthly data and 2 for quarterly data.
- The function allows to complement the original Harding-Pagan method by a prefiltering of the data with a spencer curve or a centered moving average of order k. This choice can be imposed by entering the option 'spenc=1' or 'ma=k'.
- The number of time units of cyclical dominance in the Bry-Boschan procedure can be chosen by the option 'imcd = n'. Default is automatically determined.

As usual, the results are stored in a results tlist (say rbb) which contains the dates of peaks (rbb('P')), troughs (rbb('PT')), average duration from peak to peak (rbb('DPP')), average duration from trough to trough (rbb('DTT')), average duration from peak to trough (rbb('DPT')), average duration from trough to peak rbb('DTP'), average amplitude from peak to trough (rbb('APT')), average amplitude from trough to peak rbb('ATP').

Examples can be found in function brybos_d.

```
-->load('SCI/macros/grocer/db/pigiron.dat');
```
// load Pig-Iron data used by Mark. W. Watson (1993), "Business Cycle Durations and Postwar Stabilization
// of the U.S. Economy", American Economic Review, Vol. 84, n°1, pp. 24-46

```
-->bounds()
```
// set bounds to the maximum

```
-->rbb1 = brybos('log(pigiron)','proc=''bb''')
```
// perform the Bry-Boschan procedure

```
Warning - outlier found
Dates    Observed value Replacement value
1894m6   2.6561679      2.361797
1903m12  3.550279       3.3072529
1908m1   3.7105619      3.5180911
1919m10  4.3494024      4.0963426
1922m8   4.2826571      4.070564
1933m7   3.8405122      4.0569888
1934m6   3.9342181      4.1635596
1939m5   4.2120357      4.0145796
```
// outliers are found and removed from the original time series

```
Turning points dating results of log(pigiron)
Method: Bry-Boschan complete procedure
Estimation period: 1877m1-1941m12

Peaks   Troughs
1878m3  1878m9
1883m2  1885m1
1890m5  1891m4
1891m11 1893m10
1895m11 1896m10
1900m1  1900m10
1903m6  1903m12
1907m6  1908m1
1910m2  1911m1
1913m2  1914m12
1917m4  1918m1
1918m9  1919m10
1920m10 1921m7
1923m5  1924m7
1926m4  1927m12
1929m5  1932m8
1937m8  1938m6
```

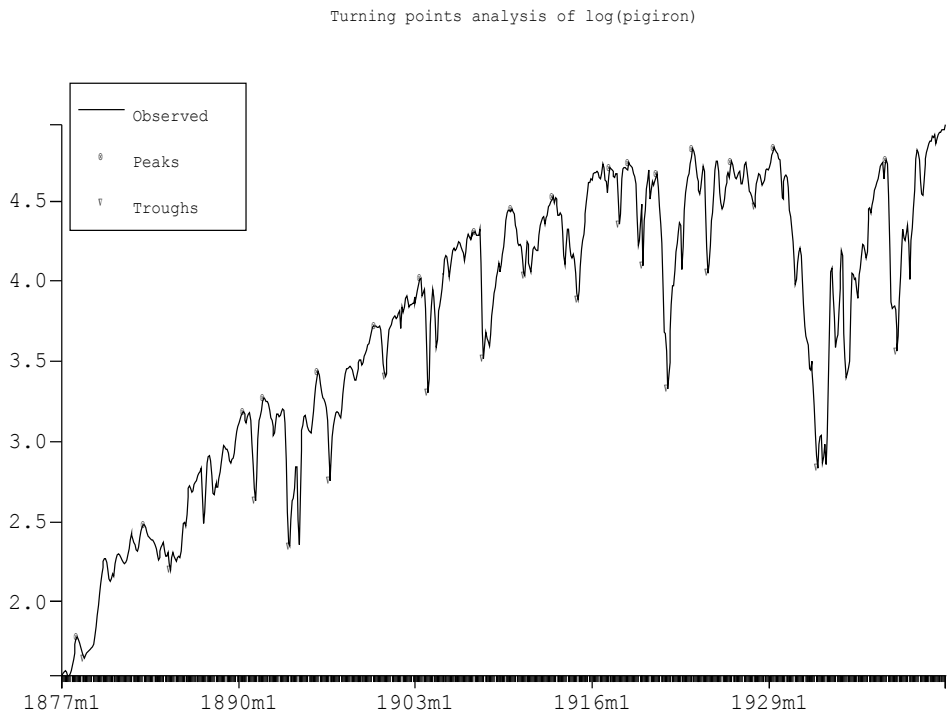
// peaks and troughs are given

```
Cycle caracteristics:
Average duration from peak to peak: 44.5625
Average duration from trough to trough: 44.8125
Average duration from peak to trough: 14.294118
Average duration from trough to peak: 30
Average amplitude from peak to trough: -0.7066871
Average amplitude from trough to peak: 0.8619982
```

// number of characteristics of the business cycle datation are provided

```
                    *
                *     *
```

And the following graph is provided, where peaks are materialized by circles and troughs by triangles:



If you have 2 series for which you have determined their turning points, then you can test if one series leads the other by the mean of Banerji's test[7]. This test is programmed in function banerji. The simplest call to this function is the following:

rba = banerji(Pr,Tr,Pt,Tt)

where:

- Pr is a vector of Grocer dates, the dates of the peaks of the reference series;
- Tr is a vector of Grocer dates, the dates of the troughs of the reference series;
- Pt is a vector of Grocer dates, the dates of the peaks of the tested series;
- Tr is a vector of Grocer dates, the dates of the troughs of the tested series;

The function then tests if the first turning point of each series are of the same nature. If it is not the case, then the program withdraws the first of these turning points and signals it to the user. The program does the same for the last turning points. It also tests if the reference series has more turning points than the tested one: then the supplementary cycles are removed (this is also signalled to the user).

Then the value of the test for all leads from 1 to 4. If the user wants to change this number of leads, then she has to enter the option 'lead=x' where x is the chosen number. If she does not want to print the results, then the usual option 'noprint' is also available.

The program displays the value of the test, together with the rejection p-value: if it is greater than 0.95, then the null hypothesis of no-lead at the 5% level is rejected. The results of the test as well as the sum of the

[7]A. Banerji (1999),"The lead profile and other non-parametric tools to evaluate survey series as leading indicators" , 24e CIRET Conference.

differences between the turning points of the 2 timeseries, the characteristics of the supplementary cycles are stored in the output results tlist.

As an example take the Belgian and French cycles, as measured by the business surveys: the corresponding data are in the database BusinessClimate.dat.

```
-->load('SCI/macros/grocer/db/BusinessClimate.dat');
```

Calculate their turning points by the Bry-Boschan procedure:

```
-->bounds('1990m1','2005m5'); rbe = brybos('clim_be','proc=''bb''');
-->rfr =brybos('clim_fr','proc=''bb'''); bounds()
```

And test the lead of the Belgian business cycle over the French one:

```
--> rba =banerji(rfr('P'),rfr('T'),rbe('P'), rbe('T'),'lead=4');
```

Which results in:

```
Banerji test of leading profile
H0 : no k-periods leading of competing series

# of extra-cycle in the reference series detected: 1
Dates are:
  P       T
  1991m2 1992m2

H0  sum rejection prob.
k<1 17  98.4375
k<2 7   80.859375
k<3 -3  26.367188
k<4 -13 2.734375

                         *
                      *     *
```

These results that, a 5% level, only the hypothesis of a 1 month lead is not rejected.

(°)3. Spectral analysis.

There are two ways in Grocer to estimate the spectrum, co-spectrum, dynamic correlation and cohesion of time series: a parametric and a non-parametric one.

3.1 Non-parametric spectral analysis(°°)

Non-parametric spectral analysis is performed in grocer by function spectral. This function takes time series as input, as well as various options. The simplest call to spectral is therefore:

```
--> rspec = spectral('series1','series2');
```

The user can enter, if she wants, any of the following options:

- The lag window size for the calculation of the autocorrelation function can be given by the option 'trunc=n'. The default value is round(sqrt(T))
- A vector of weights applied to each series in the calculation of the cohesion of the cycle between the entered time series can be entered trough the option 'weight=v' where v is a (N x 1) vector. For instance 'weight=[0.25;0.5;0.25]'. By default all series have the same weight.
- If the user wants the program to perform block-bootstrap estimation of confidence band, then she must enter the option 'boot =1'. In that case, she can choose the number of draws with the option 'B=n' (default: 200) and the size of the blocks with the option 'sb=xx' (default: 8).
- Various printing options can be entered: 'spec=1' to plot the spectrum of the series, 'cospe=1', to plot their cospectrum, 'dcorr=1' to plot their dynamic correlation, 'phase=1' to plot the phase spectrum, 'coher=1' to plot their coherency, 'cohes=1' to plot their cohesion.

The results are saved in a results tlist (say rspec), which contains :

- the matrix of the cospectrum (rspec('cospe')):

if there are more than one ts:

- the matrix of cohesion (rspec('cohes'));
- the matrix of coherence (rspec('coher'));
- the matrix of dynamic correlations (rspec('dcorr')):
- the matrix of standardized phase spectrum (ts rspec('phase'))

if confidence bands have been calculated:

- vector of upper and lower bound for cohesion (rspec('ucohes') and rspec('lcohes'))
- vector of upper and lower bound for coherency (rspec('ucoher') and rspec('lcoher'))
- rspec('ucospe') matrix of upper bound for cospectrum
- rspec('lcospe') matrix of upper bound for cospectrum
- rspec('udcorr') matrix of upper bound for dynamic correlations
- rspec('ldcorr') matrix of lower bound for dynamic correlations
- rspec('uphase') matrix of upper bound for phase spectrum
- rspec('lphase') matrix of lower bound for phase spectrum

moreover, the results tlist contains:

- a boolean indicating the presence of time series (rspec('prests'))
- the name of the series (rspec('namex'))
- the matrix of the values of the series (rspec('x'))
- the bounds of the calculations if any (rspec('bounds'))

As an example, load the series of the French, German GDP

```
-->load('SCI\macros\grocer\db\specgdp.dat');
```

Then get the names of the variables available in the database

```
-->lvar =dblist('SCI\macros\grocer\db\specgdp.dat');
```

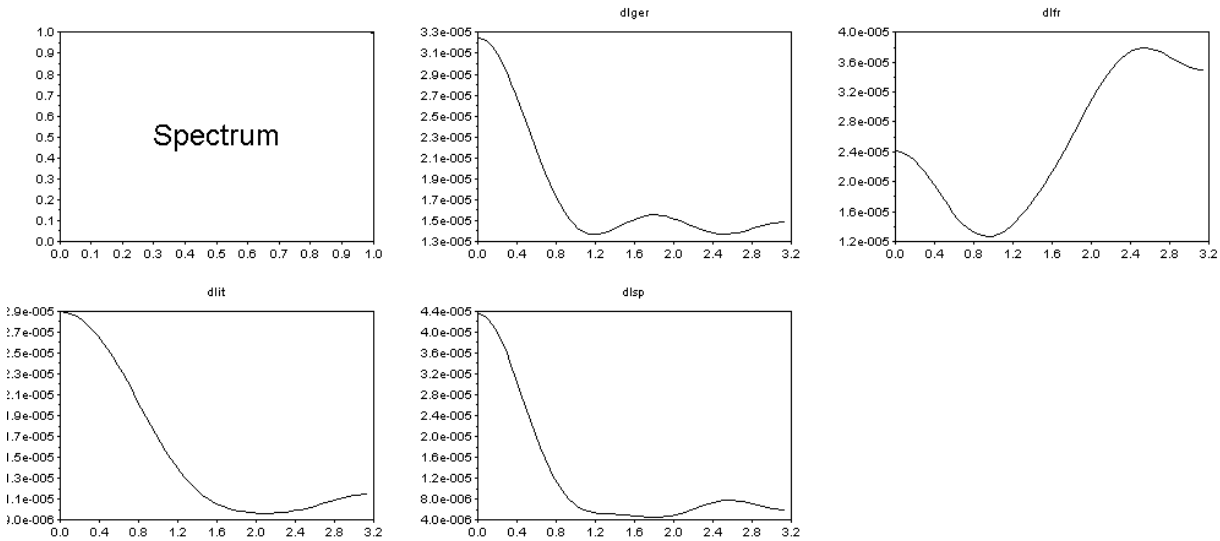
Transform them by log and first difference:

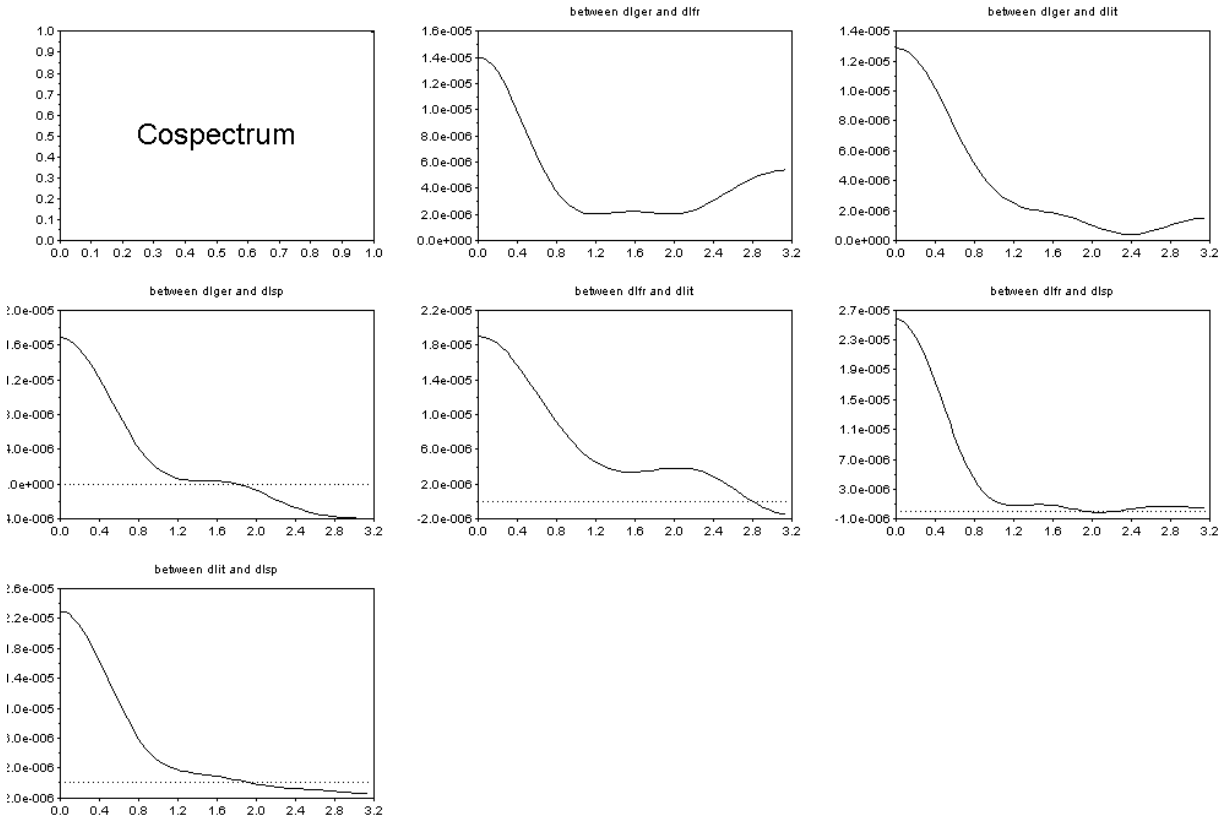
```
-->for i = 1:size(lvar,1)
-->execstr('dl'+lvar(i)+' = delts(log('+lvar(i)+'))')
-->end
```

And calculates their spectrum and cospectrum with a truncation window of size 6:

```
-->rspec = spectral('dlger','dlfr','dlit','dlsp','trunc=6','spec=1','cospe=1');
```

Two series of graphs are produced:

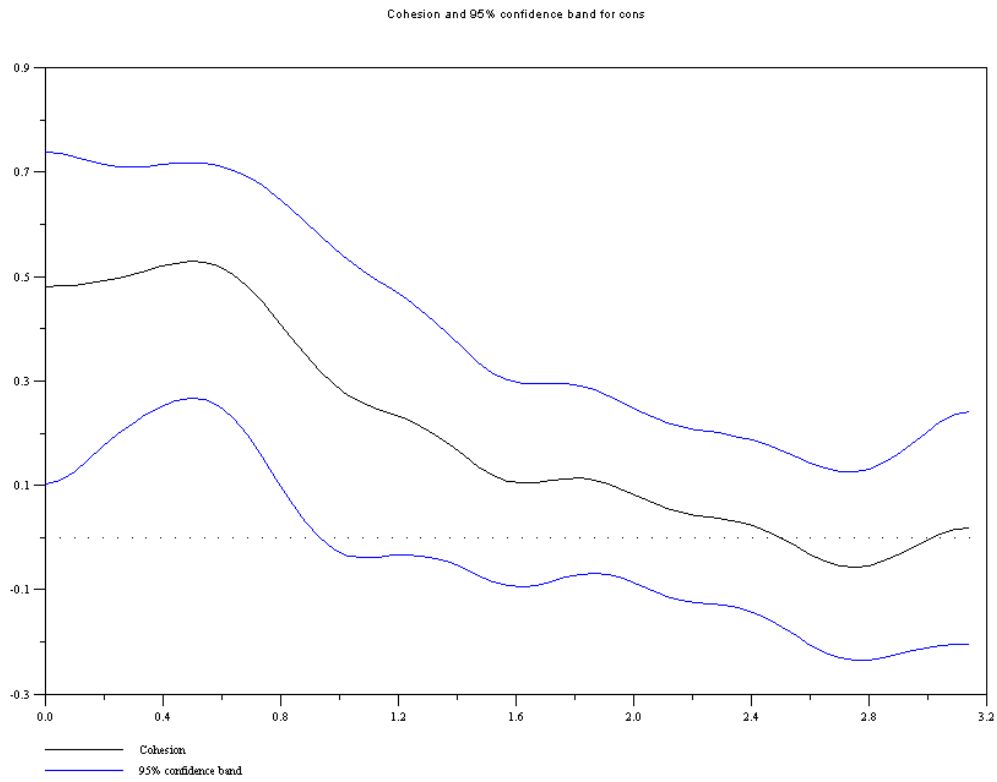

Spectrum
dlger
dlfr
dlit
dlsp

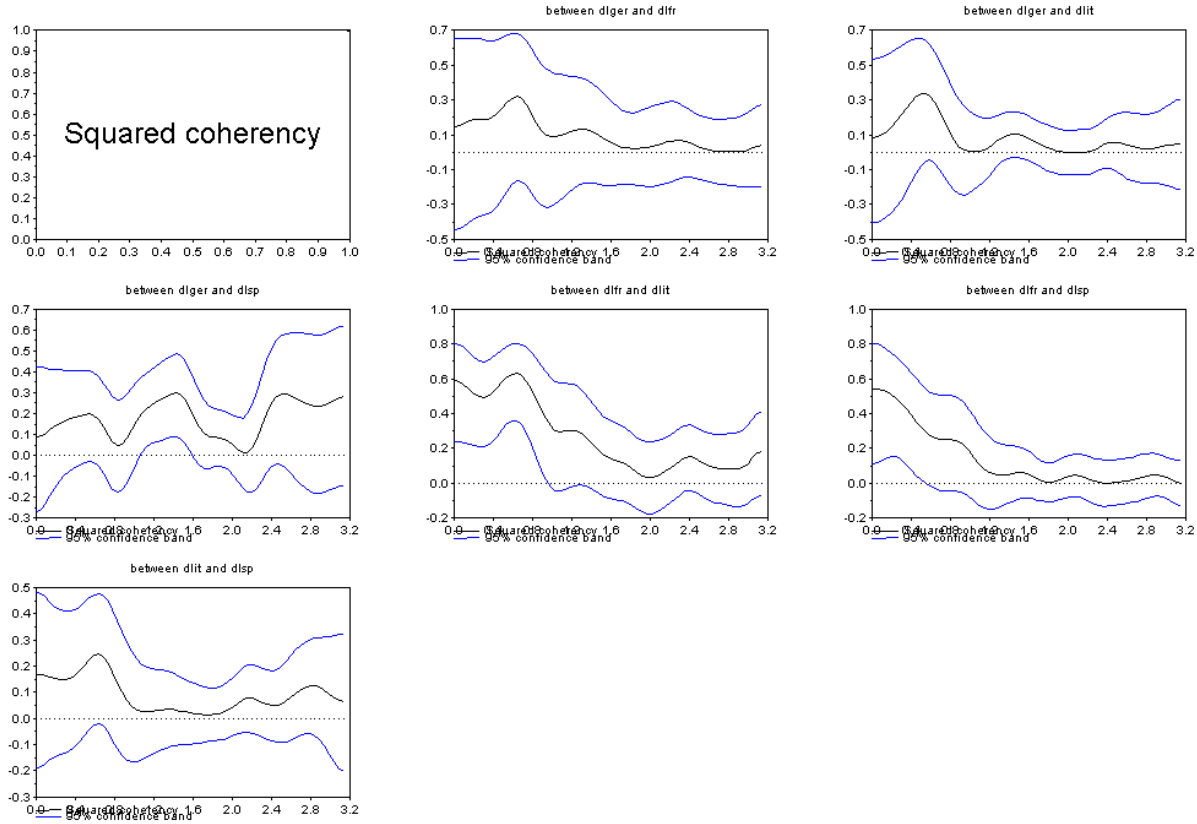

Cospectrum
between dlger and dlfr
between dlger and dlit
between dlger and dlsp
between dlfr and dlit
between dlfr and dlsp
between dlit and dlsp

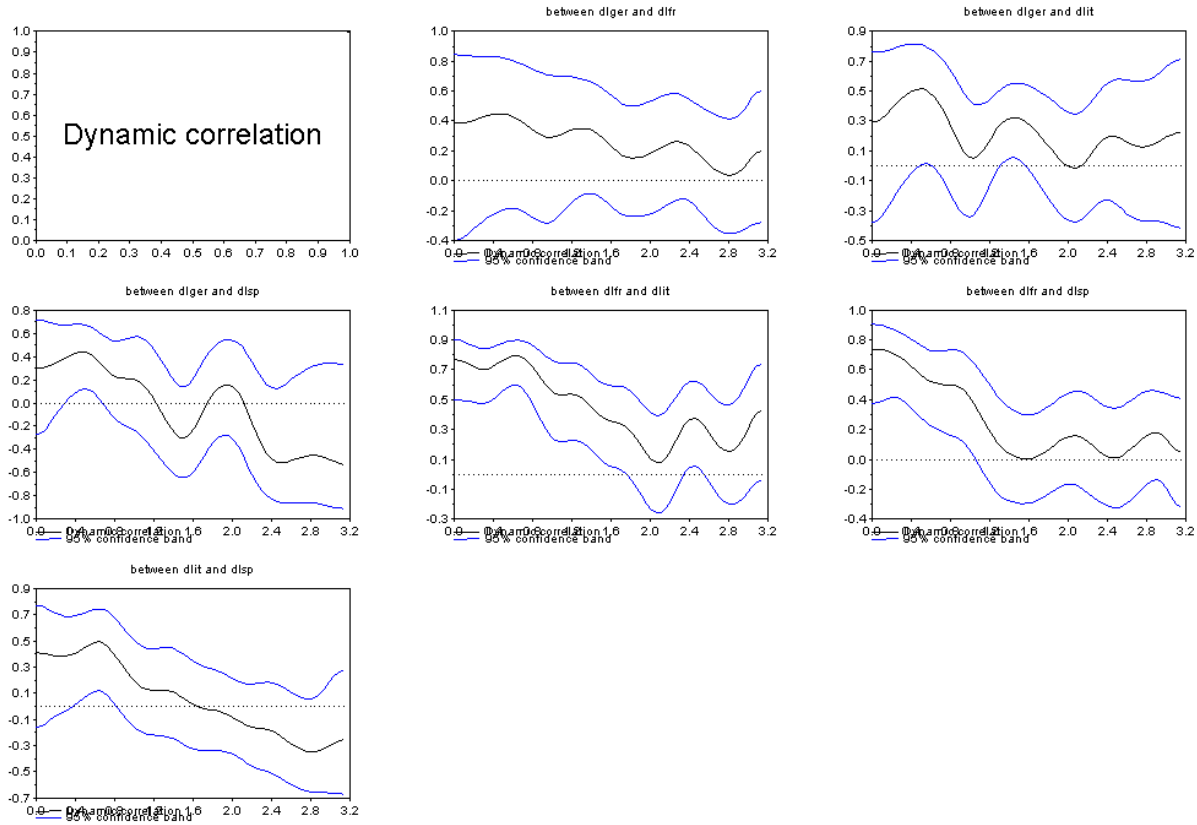
You can obtain the dynamic correlations, coherency and cohesion, with confidence bands estimated from 200 draws and a truncation window size of 12, by the following instruction:

```
-->rspec = spectral('dlger','dlfr','dlit','dlsp','trunc=12',...
-->'dcorr=1','coher=1','cohes=1','boot=1','B=200','sb=8');
```

Three series of graphs are then produced:




Squared coherency
between dlger and dlfr
between dlger and dlit
between dlger and dlsp
between dlfr and dlit
between dlfr and dlsp
between dlit and dlsp
95% confidence band


Dynamic correlation
between dlger and dlfr
between dlger and dlit
between dlger and dlsp
between dlfr and dlit
between dlfr and dlsp
between dlit and dlsp
95% confidence band

(°)3.2 Parametric spectral analysis

Parametric spectral analysis is performed by function specvarma. The analysis is based on a parametric model that must be first estimated by the user, by an ols estimation, a var estimation or a varma one. Therefore specvarma function has as its first argument a tlist result, provided by function ols, var or varma. The next arguments are almost the same as in spectral, except those relating to the bootstrap which are here unnecessary.

Therefore the user can enter:

- The lag window size for the calculation of the autocorrelation function can be given by the option 'trunc=n'. The default value is round(sqrt(T))
- A vector of weights applied to each series in the calculation of the cohesion of the cycle between the entered time series can be entered trough the option 'weight=v' where v is a (N x 1) vector. For instance 'weight=[0.25;0.5;0.25]'. By default all series have the same weight.
- Various printing options can be entered: 'spec=1' to plot the spectrum of the series, 'cospe=1', to plot their cospectrum, 'dcorr=1' to plot their dynamic correlation, 'phase=1 'to plot the phase spectrum, 'coher=1' to plot their coherency, 'cohes=1' to plot their cohesion.

There is a last option: the user can compute the confidence interval of the estimated spectrum, cospectrum, dynamic correlation, phase spectrum, coherency and cohesion, with the delta method (option 'ic=1').

To have a flavour of the delta method, consider a VAR(k)[8] estimated on y_t a vector of n variables. Let $\hat{\pi}_T = vec(\Pi_T)$ be the $(nk \times 1)$ vector of coefficients resulting from OLS estimation and Σ_T the variance-covariance matrix of the residual. If the VAR process is stationary and residuals are *iid* then

$$\begin{bmatrix} \sqrt{T}(\hat{\pi}_T - \pi) \\ \sqrt{T}\left(vech(\hat{\Sigma}_T) - vech(\Sigma)\right) \end{bmatrix} \to N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \Sigma \otimes Q & 0 \\ 0 & \Lambda_{22} \end{pmatrix}\right)$$

where $Q = E(x_t x_t')$ with $x_t = [y_{t-1}' \, y_{t-2}' \cdots y_{t-k}']$, and the elements of Λ_{22} corresponding to the covariance between estimated variances. To understand how this matrix is obtained, consider the unique matrix D_n such that $D_n vech(\Sigma) = vec(\Sigma)$. Further define $D_n^{+\cdot} = (D_n' D_n)^{-1} D_n'$ [9]such that $D_n^{+\cdot} D_n = I$. It turns that the matrix Λ_{22} can be written as $\Lambda_{22} = 2D_n^{+\cdot}(\Sigma \otimes \Sigma) D_n^{+\cdot\,\prime}$ [10].

Remind that the spectrum of any VAR process can be express as $S_y(\omega) = \Psi(e^{i\omega}) \Sigma \Psi(e^{i\omega})'$ where $\Psi(L) = I - \Phi(L)$ is the vector of lagged polynomials. So the the spectrum is a non-linear transformation of the parameters of the VAR process. One can exploit this result to compute the asymptotic distribution of the spectrum by means of the *delta method* which uses first order Taylor approximation to recover the asymptotic distribution of the spectrum.

Let F be such that $S_y(\omega) = F(\Psi, \Sigma) = F(\Phi)$ and denote by Ξ the variance-covariance matrix of the stacked parameters of the VAR process, the delta method tells us that $\sqrt{T}(\hat{S}_y(\omega) - S_y(\omega)) \to N(0, V_S)$ where $V_S = F_\Phi' \Xi F_\Phi$ with F_Φ the vector of first derivative (in GROCER the derivatives are numerically computed) of the spectrum according to the parameters.

As an example, load the database specgdp.dat in your library macros\grocer\db\ in Scilab library:

[8]The extension VARMA process is straightfoward.

[9] D_n and $D_n^{+\cdot}$ are compute in grocer , respectively in *duplication* and *dnplus* function.

[10]See Hamilton, James D. (1994), *Time Series Analysis*, ed. Princeton, p. 298-301.

```
-->load('SCI\macros\grocer\db\specgdp.dat');
```

Then get the names of the variables available in the database

```
-->lvar =dblist('SCI\macros\grocer\db\specgdp.dat');
```

Transform the data by log and first difference and demean data

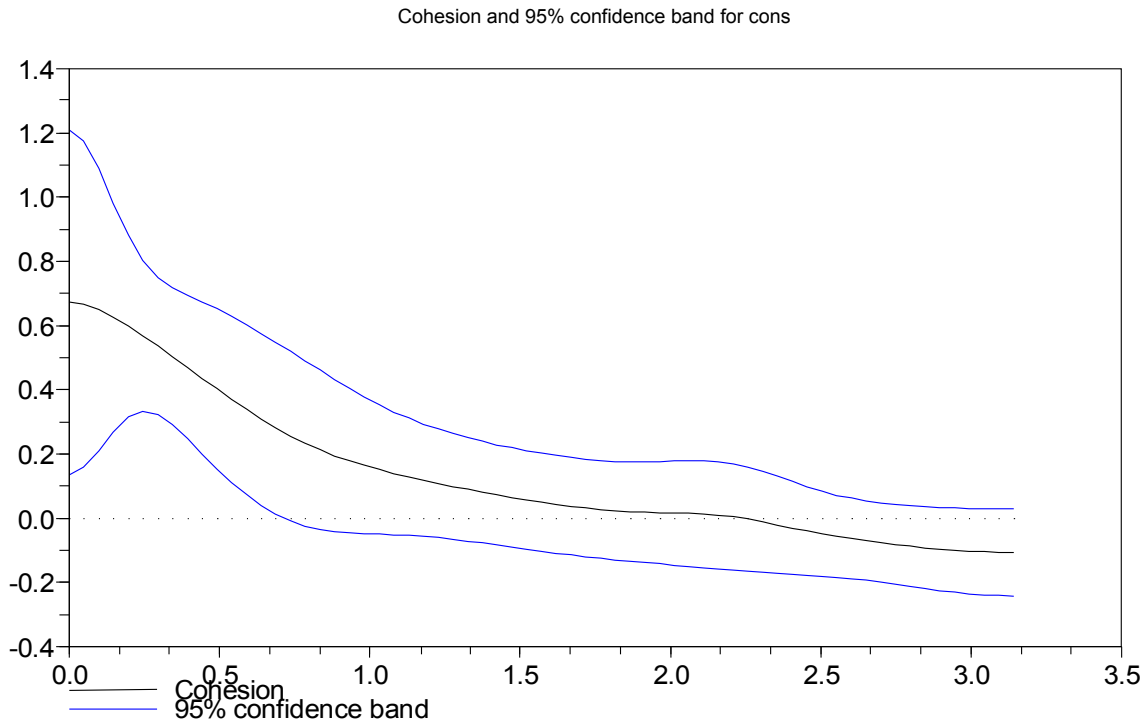
```
-->for i = 1:size(lvar,1)
-->   execstr('dl'+lvar(i)+' = delts(log('+lvar(i)+'))')
-->   execstr('dl'+lvar(i)+'_m = dl'+lvar(i)+' - mean(dl'+lvar(i)+')')
-->end
```

Estimate a var(4) model without constant and without printing the results:

```
-->rspec = specvarma(rvar,'cohes=1','ic=1');
-->rvar = var(4,'endo=dlfr_m;dlger_m;dlit_m;dlsp_m','nocte','noprint')
```

Estimate and print the cohesion of var model and compute 95% interval confidence band, with the delta method and see the result on the graph window (reported below):

```
-->rspec = specvarma(rvar,'cohes=1','ic=1');
```



4. The business cycle functions and their specifications.

Abuild_____calculation for the CF filter

CALLING SEQUENCE

A=Abuild(nn,nq,g,root)

PARAMETERS

INPUT:
* nn = # of observations
* nq = truncation lag
* g = a (nx1) vector (equal to the convolution of the ma part of the process with its reverse
* root = the # of unit roots in the process (0 or 1)

OUPTUT:
* A = a (nnxnn) matrix

DESCRIPTION
Builds the nn x nn A matrix in A.12 in Christiano-Fitzgrald (1999) paper
if root == 1 (unit root), Abig is used to construct all but the last 2 rows of the A matrix
if root == 0 (no unit root), Abig is used to construct the entire A matrix

Example:
Used in Grocer function cffilter:
A = Abuild(T,nq,g,root)

banerji_____tests the lead of a series against another one

CALLING SEQUENCE

rba = banerji(Pr,Tr,Pt,Tt,varargin)

PARAMETERS

INPUT:
* Pr = a vector of through dates for the reference series
* Tr = a vector of through dates for the reference series
* Pt = a vector of through dates for the competing series
* Tt = a vector of through dates for the competing series
* arg1, arg2: optional arguments:
  - lead=xx' = number of leads to test (default is 4)
  - 'noprint' = if the user doesn't want to print the results

---

OUPTUT:
rba a results tlist with :
- rba('signi') = confidence for rejection of the null hypothesis of lead not significant
- rba('sum_ini') = initial sum of time difference

- rba('lead') = number of tested leads (default is 4)
- rba('dates_extra')= dates of extra-cycle in the reference serie
- rba('PT_extra') = type of each dates (P or T) in the extra-cycle

DESCRIPTION
Randomization test to evaluate the leading profile of a series against another one. Test the null H0: leads are not significant. A high level function that works with peaks and troughs and with options

REFERENCE
A. Banerji (1999),"The lead profile and other non-parametric tools to evaluate survey series as leading indicators" , 24e CIRET Conference.

Example:
bounds('1990m1','2005m5'); rbe = brybos('clim_be','proc="bb"'); rfr =brybos('clim_fr','proc="bb"'); bounds();
Pr = rfr('P') ; Tr = rfr('T') ; Pt = rbe('P') ; Tt = rbe('T') ; rba =banerji(Pr,Tr,Pt,Tt,'lead=4');

This example is taken from the function banerji_d. It tests if the Belgian cycle leads the French one, as they are measured by business surveys.

banerji0_____tests the lead of a series against another one

CALLING SEQUENCE

rba = banerji0(Pr,Tr,Pt,Tt,varargin)

PARAMETERS

INPUT:
* y = a vector of difference in timing turns
* lead = number of leads to test (default is 4)

OUPTUT:
rba a results tlist with :
- rbar('signi') = confidence for rejection of the null
- hypothesis of lead not significant
- rbar('sum_ini') = initial sum of time difference
- rbar('lead') = number of tested leads (default is 4))

DESCRIPTION
Randomization test to evaluate the leading profile of a series against another one. Test the null H0: leads are not significant. Low level function that works directly on the difference between 2 cycles.

Example:
rba = banerji0(grocer_y,grocer_lead)

This example is taken from the function banerji.

brybos_____Bry-Boschan turning points datation

CALLING SEQUENCE

rbb = brybos(namey0,varargin)

PARAMETERS

INPUT:
* namey0 = a ts
* varargin = options:
  - 'M=xx' = minimal duration Peak-to-Peak or trough-to-trough
  - 'e=xx' = min # of periods separating a turn form borders
  - 'm=xx' = minimal phase
  - 'k=xx' = # to determine the local min or max
  - 'ma=xx' = use of filtered data by a moving xx-centered moving average
  - 'spenc=1' = filters data by a Spencer curve
  - 'proc ="bb"' = performs bry-boschan specific procedure
    else performs Harding-Pagan dating rules ('proc ="hp"')
  - 'mcd = xx' = user defined month of cyclical dominance for bry-boschan procedure
  - 'noprint' if the user does not want to print the results

OUTPUT:
a results tlist with:
- rbb('meth') = method used ('bb' or 'hp')
- rbb('P') = dates of peaks
- rbb('T') = dates of troughs
- rbb('DPP') = average duration from peak to peak
- rbb('DTT') = average duration from trough to trough
- rbb('DPT') = average duration from peak to trough
- rbb('DTP') = average duration from trough to peak
- rbb('APT') = average amplitude from peak to trough
- rbb('ATP') = average amplitude from trough to peak
- rbb('filter') = if the data are filtered before analysis
- rbb('prests') = boolean indicating the presence or absence of a time series in the regression
- rbb('namey') = name of the y variable
- rbb('bounds') = the bounds of the data

DESCRIPTION
Computes Bry-Boschan or Harding and Pagan turning points dating rules.

REFERENCES:
* D. Harding & A. Pagan (2002), "Dissecting the Cycle: a Methodological Investigation", Journal of Monetary Economics, n°49, pp. 365-381
* G. Bry & C. Boschan (1971), "Cyclical Analysis of Time Series: Selected Procedures and Computer Programs", NBER Technical Paper n°20

Examples:
1) load('SCI/macros/grocer/db/pigiron.dat'); rbb1 = brybos('lpigiron','proc="bb"');

2) load('SCI/macros/grocer/db/pigiron.dat'); rbb2 = brybos('lpigiron','proc="hp"','spenc=1')

Examples taken from function brybos_d. The Pig-Iron data used by M.Watson (1993), ("Business Cycle Durations and Postwar Stabilization of the U.S. Economy", American Economic Review, Vol. 84, n°1, pp. 24-46) are used.

In example 1, the datation is provided by the Bry-Boschan method ('proc="bb"') and with defaults values for all parameters.

In example 2, the datation is provided by the Harding_Pagan method ('proc="hp"') and the smoothing pass is made with a Spencer curve ('spenc=1').

bge_____solution for integral of B(z)g(z)(1/z)^j

CALLING SEQUENCE

y=bge(jj,nq,B,cc)

PARAMETERS

INPUT:
* jj = exponent of 1/z
* nq = truncation lag
* B = a (nx1) vector
* cc = (nx1) vector (discretisation of function g)

OUTPUT:
y = the result of the integration

DESCRIPTION
Closed form solution for integral of B(z)g(z)(1/z)^j (eqn 16) in Christiano-Fitzgerald (1999) paper. If nq > 0, jj >= 0; if nq = 0, y = 2*pi*B(absj+1)*cc(1);

Example:
Used in Grocer function cffilter:
dj = bge(i-2,nq,B,cc)

bkfilter_____Baxter-King filter

CALLING SEQUENCE

[fX,AA]=bkfilter(namey,pl,pu,nfix,typ)

PARAMETERS

INPUT:
* namey = a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real

vector between quotes
* pl = minimum period of oscillation of desired component
* pu = maximum period of oscillation of desired component
(2<=pl<pu<infinity).
* nfix = length of the fixed filter (if typ = 'sym')
or: dropped observations at the beginning and end of resulting variable
* typ = 'fixed' for a fixed length filter (default)
or: 'vari' for a variable length filter

OUPTUT:
* fX = the ts or(nX1) vector containing the filtered data
* AA = filtering matrix

DESCRIPTION
Computes Baxter-King filter.

Examples:
1) deu_gdp_bk12=bkfilter(deu_gdp,6,32,12,'fixed')
2) deu_gdp_bkasym=bkfilter(deu_gdp,6,32,1,'vari')

Examples taken from function filters_d. Both examples applies to the quarterly German gdp and extracts business cycle frequencies (6 to 32 quarters). In the first case, the fixed length filter is used, with length 12. In the second case, the variable length filter is used, and the first and last observations are discarded.

cffilter_____Christiano-Fitzgerald filter

CALLING SEQUENCE

[fX,AA]=cffilter(namey,pl,pu,arg1,…,argn)

PARAMETERS

* namey = a time series, a real (nx1) vector or a string equal to the name of a time series or a (nx1) real vector between quotes
* pl = minimum period of oscillation of desired component
* pu = maximum period of oscillation of desired component
(2<=pl<pu<infinity).
* argi = an optional argument that can be
- 'root=0' or 'root=1' whether you want to assume 0 or 1 unit root in time series (default: root = 1);
- 'drift=0' or 'drift=1' whether you want to assume no drift or a time trend in time series (default: drift = 1);
- 'filt=asym' or 'filt=sym' or 'filt=flsym' whether you want to use the asymmetric Filter, the symmetric Filter or the Fixed Length Symmetric Filter (default='asym')
- 'nfix=xx' where xx is the fixed length (useful only for the symmetric filter, that is filt=flsym) (default: nfix=-1).
- thet = a (nx1) vector corresponding to the MA representation of x (if root=0) or delta(x) (if root =1) (default: theta=1)

OUPTUT:
* fX = vector (nX1) containing filtered data
* AA = filtering matrix

NOTE:
The filtered data (fX) associated with some filters include zeros at the beginning and end of the data set. The filter is not available for these points. You should truncate these points from your analysis. Examples include fixed length filters and filters for user supplied time series with length(thet)>1. The default filter returns all nonzero data.

DESCRIPTION
Computes Christiano-Fitzgerald filter.

Example:
deu_gdp_cff=cffilter(deu_gdp,6,32)

Example taken from function filters_d. It applies to the quarterly German gdp and extracts business cycle frequencies (6 to 32 quarters).

hpfilter_____Hodrick Prescott filter

CALLING SEQUENCE

[hpy]=hpfilter(y,lambda)

PARAMETERS

INPUT:
* y = either
  . a time series, or
  . a real (nx1) vector, or
  . a string equal to the name of a time series or a (nx1) real vector between quotes
* lambda = the smoothing parameter

OUTPUT:
hpy= the smoothed filtered series of the same type than y (if y is not a string) or evstr(y) (if y is a string)

DESCRIPTION
Hodrick Prescott filter.

Example:
Xf=hpfilter(x,1600)

spectral_____non parametric spectral analysis of time series

CALLING SEQUENCE

[rspec]=spectral(arg1,...,argn)

PARAMETERS

INPUT:
* arg1,..,argn = arguments which can be:
  - a string equal to the name of a time series or a (nxk) real vector or matrix between quotes
  - a real (nx1) vector
  - a real (nxk) matrix
  - a list of such elements
  - the string 'noprint' if the user doesn't want to display the results of the regression
  - optional arguments that can be the following:
    o 'trunc=xx' : lag window size (the default value is k=round(sqrt(T)))
    o 'weight=xx' : (N x 1) vector of weights (equal weights by default) for cohesion
    o 'spec=1' : plot spectrum
    o .'dcorr=1' : plot dynamic correlation
    o 'phase=1' : plot phase spectrum
    o 'coher=1' : plot coherency
    o 'cohes=1' : plot cohesion
    o 'boot =1' : performs block-bootstrap estimation of confidence band
    o 'B=xx' : number of draws for bootstrap replication (default is 200)
    o 'sb=xx' : size of blocks for block-bootstrap (default is 8)

OUTPUT:
rspec a tlist result:
. rspec('cospe') = matrix of cospectra
. rspec('cohes') = matrix of cohesion (if more than one ts)
. rspec('coher') = matrix of coherence (if more than one ts)
. rspec('dcorr') = matrix of dynamic correlations (if more than one ts)
. rspec('phase') = matrix of standardized phase spectrum (if more than one ts)
. rspec('order') = order of arrival of variable in cross-products
. rspec('ucohes') = matrix of upper bound for cohesion
. rspec('lcohes') = matrix of lower bound for cohesion
. rspec('ucoher') = matrix of upper bound for coherency
. rspec('lcoher') = matrix of lower bound for coherency
. rspec('ucospe') = matrix of upper bound for cospectra
. rspec('lcospe') = matrix of upper bound for cospectra
. rspec('udcorr') = matrix of upper bound for dynamic correlations
. rspec('ldcorr') = matrix of lower bound for dynamic correlations
. rspec('uphase') = matrix of upper bound for phase spectrum
. rspec('lphase') = matrix of lower bound for phase spectrum
. rspec('prests') = a boolean indicating the presence of time series
. rspec('namex') = name of the series
. rspec('x') = matrix of the values of the series
. rspec('bounds') = bounds of the calculations if any

DESCRIPTION
Estimates parametric spectral density of VARMA process.

REFERENCE:
C. Croux, M. Forni and L. Reichlin (2001), "A Measure of Comovements for Economic Indicators: Theory and Empirics", The Review of Economics and Statistics, Vol. 83(2), p. 232-241§.

Example:
1) rspec = spectral('dlger','dlfr','dlit','dlsp','trunc=6','spec=1','cospe=1');
2) rspec = spectral('dlger','dlfr','dlit','trunc=12','dcorr=1','coher=1','cohes=1','boot=1','B=200','sb=8');

Examples taken from function spectral_d.

In example 1, variations of the logarithm of the German, French, Italian and Spanish GDP are studied. Their co-spectrum is calculated with a truncation lag equal to 6('trunc=6'); their spectrum ('spec=1') and co-spectrum are plotted ('cospe=1').

In example 2, variations of the logarithm of the German, French and Italian GDP are studied. Their co-spectrum is calculated with a truncation lag equal to 12 ('trunc=12'); their dynamic correlation ('dcorr=1'), coherency ('coher=1') and cohesion are plotted ('cohes=1'). Block-bootstrap estimation of confidence band is provided ('boot=1') with 200 draws ('B=200') and a block size of 8 ('sb=8').

specvarma_____parametric spectral analysis of time series

CALLING SEQUENCE

rspec = specvarma(result,arg1,...,argn)

PARAMETERS

INPUT:
* result = a tlist of results of estimation for VAR, VARMA, OLS models
* arg1,..,argn = arguments which can be:
  - a string equal to the name of a time series or a (nxk) real vector or matrix between quotes
  - a real (nx1) vector
  - a real (nxk) matrix
  - a list of such elements
  - the string 'noprint' if the user doesn't want to display the results of the regression
  - optional arguments that can be the following:
    o 'trunc=xx' : lag window size (the default value is k=round(sqrt(T)))
    o 'weight=xx' : (N x 1) vector of weights (equal weights by default) for cohesion
    o 'spec=1' : plot spectrum
    o .'dcorr=1' : plot dynamic correlation
    o 'phase=1' : plot phase spectrum
    o 'coher=1' : plot coherency
    o 'cohes=1' : plot cohesion
    o 'boot =1' : performs block-bootstrap estimation of confidence band
    o 'B=xx' : number of draws for bootstrap replication (default is 200)
    o 'sb=xx' : size of blocks for block-bootstrap (default is 8)

OUTPUT:
rspec a tlist result:
. rspec('cospe') = matrix of cospectra
. rspec('cohes') = matrix of cohesion (if more than one TS)
. rspec('coher') = matrix of coherence (if more than one TS)
. rspec('dcorr') = matrix of dynamic correlations (if more than one TS)
. rspec('phase') = matrix of standardized phase spectrum (if more than one TS)
. rspec('order') = order of arrival of variable in cross-products
. rspec('ucohes') = matrix of upper bound for cohesion
. rspec('lcohes') = matrix of lower bound for cohesion
. rspec('ucoher') = matrix of upper bound for coherency
. rspec('lcoher') = matrix of lower bound for coherency
. rspec('ucospe') = matrix of upper bound for cospectra
. rspec('lcospe') = matrix of upper bound for cospectra
. rspec('udcorr') = matrix of upper bound for dynamic correlations
. rspec('ldcorr') = matrix of lower bound for dynamic correlations
. rspec('uphase') = matrix of upper bound for the phase spectrum
. rspec('lphase') = matrix of lower bound for the phase spectrum
. rspec('omega') = vector of frequencies
. rspec('order') = order of arrival of variable in cross-products
. rspec('AR') = matrix of AR coefficients AR=[A1 .. Ap]
. rspec('MA') = matrix of MA coefficients MA=[B1 .. Bq]

DESCRIPTION
This function provides standard spectral related tools to analyse stationary time series. It performs: spectrum, cospectrum, phase spectrum, coherence, dynamic correlation and cohesion. The method is parametric: the function is based upon the a priori estimation of a var or varma model, whose results are taken as an entry of function specvarma.

REFERENCE:
C. Croux, M. Forni and L. Reichlin (2001), "A Measure of Comovements for Economic Indicators: Theory and Empirics", The Review of Economics and Statistics, Vol. 83(2), p. 232-241§.

Example:
rvar = var(4,'endo=dlfr_m;dlger_m;dlit_m;dlsp_m','nocte'); rspec = specvarma(rvar,'cohes=1','ic=1');

Example taken from function specvarma_d. A var model is estimated on the variations of logarithm of the French, German, Italian and Spanish gdp. The cohesion of these 4 series is plotted ('cohes=1') with the confidence band calculated by the mean of the delta method.