High resolution spectral analysis software

Tryphon Georgiou

The following is a brief report on the uses of a framework and certain matlab-based routines for high resolution spectral analysis of time series. First we address scalar time-series and how to use the software to resolve sinusoids in particular. We then provide examples for multivariable analysis and system identification.

1 Scalar time-series analysis

We begin we time-domain data and some prior information as to the frequency range of interest. A filter-bank (one input many outputs) is then design with suitable properties, namely, a bandpass characteristic over the frequency range of interest. It consists of a dynamical system

$$x_{k+1} = Ax_k + Bu_k$$

with A, B matrices of size $n \times n$, and $n \times m$ respectively. Of course, when the time-series u_k is scalar, $m = 1$. The time-series is considered to have zero mean and is adjusted accordingly. An observation record

$$\{u_1, u_2, \ldots, u_N\}$$

is typically available, and on the basis of that an estimate of the state-covariance

$$P = E\{x_k x_k^*\}$$

is obtained using routine `dlsim_complex.m`. Relevant theory and the corresponding routines

Name	usage
`sm.m`	`[fr_lines,ampl_lines]=sm(P,A,B,k)`
`me.m`	`me_spect=me(P,A,B,omega)`
`envlp.m`	`env=envlp(P,A,B,omega,noiselevel)`

are then used for spectral analysis. They determine (a) spectral lines consistent with the data, (b) a candidate spectrum for u_k which is consistent with the data P and is of maximal entropy, and (c) an envelop for the amplitude of all consistent with the data spectral lines.

The resolution of the above routines strongly depends on the choice of A, B and on the variance of the estimator for the state covariance P. Tradeoffs

between robustness and resolution using such methods is the subject of a Ph.D. thesis by A.Nasiri-Amini (that will be available in August 2005). It provides guidelines for optimal design of *input-to-state* filters and theoretical bounds for the expected gains in resolution.

In practice, as a rule of thumb, there are two parameters that dictate the performance of the relevant spectral estimators: the time-constant of the *input-to-state* filter

$$G(z) = (zI - A)^{-1}B$$

and its bandpass character. Routines `cjordan2.m` and `mjordan.m` can be used for designing suitable (A, B) pairs for scalar and vectorial time-series, respectively. Typically one needs only specify a (complex) eigenvalue(s) for A and the size of the corresponding Jordan block(s). The modulus of the eigenvalues dictates the time-constant of $G(z)$ and the phase specifies the band pass character. Finally, the pair is then normalized to satisfy

$$AA^* + BB^* = I$$

where I is the identity, for numerical reasons. Typically, A can be chosen to have one Jordan block (when u_k is scalar) or as a Kronecker product of such a matrix with the identity (as in `mjordan.m`).

Routine `demo1.m` exemplifies the performance of the above for an academic example of separating two sinusoids in background noise. Because of the bandpass character of $G(s)$ and the fact that the framework relies on the state-covariance of $G(s)$, the performance of all the above is impervious to color noise (as long as it is relatively white over the passband of $G(s)$). Yet, in the example we use white noise for simplicity. A typical observation record is given in Figure 1 below.

Then, Figure 2 displays a typical output. The "true" spectrum of the time-series u_k is represented by the red "noise level" and two red arrows for the sinusoids at frequencies 1.3 and 1.35 [rad/sec]. The subplots on the right represent zoom-in of those on the left, focusing on a frequency range of interest $[1.2, 1.5]$. Fft-based reconstruction is shown in green, and our high resolutions methods in blue. These, again, give spectral lines (shown in the first line as blue arrows), envelop of such (shown on the thirds line), and maximum entropy spectra (shown last).

Content of file: `demo1.m`

The following commands were used to generate Figures 1 and 2.

```
% This test file compares fft-based estimation
% with spectral envelopes, subspace methods, and maximum entropy.
figure(1), clf

% Setting up the signal
N=100; mag0=1.8;mag1=1.5; o1=1.3; mag2=2; o2=1.35;
```

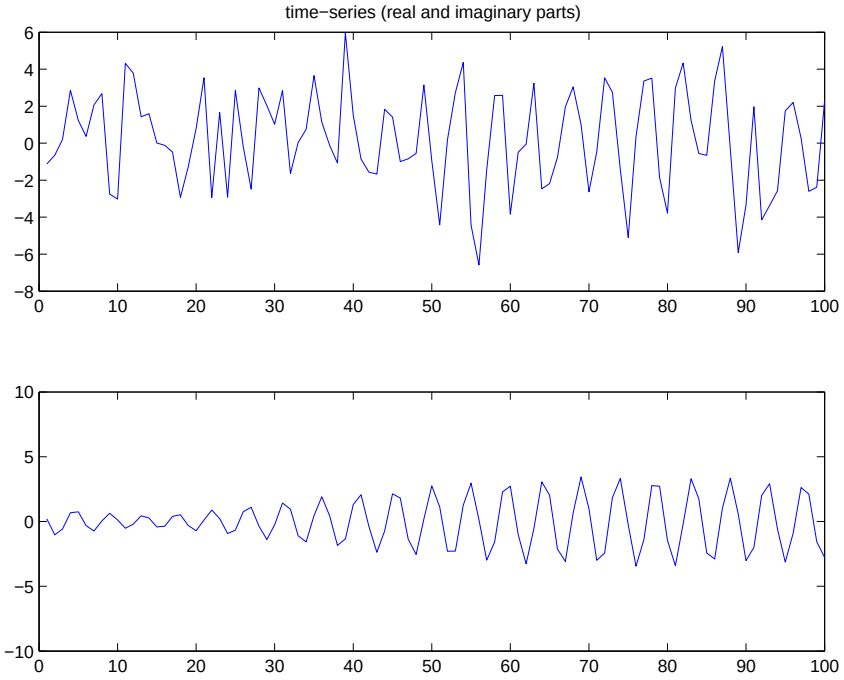

Figure 1: Time-series

```
  t=0:N-1; t=t(:);
  y=mag0*randn(N,1)+mag1*exp(j*(o1*t+2*pi*rand))+mag2*exp(j*(o2*t+2*pi*rand));

% plotting true spectrum
subplot(4,2,1), arrow([o1 o2],[sqrt(mag1) sqrt(mag2)]), hold on
                plot( [0 2*pi],[mag0 mag0],'r')
subplot(4,2,2), arrow([o1 o2],[sqrt(mag1) sqrt(mag2)]), hold on
                plot( [0 2*pi],[mag0 mag0],'r')
                axis([1.2 1.5 0 2])
title('true spectrum')

% plotting the fft spectra
               NN=2048; th=linspace(0,2*pi,NN);

               subplot(4,2,3),
               plot(th,abs(fft(y.*hamming(N),NN))/sqrt(N),'g'), hold on
               plot(th,abs(fft(y,NN))/sqrt(N),'g-.')

               subplot(4,2,4), hold off
               plot(th,abs(fft(y.*hamming(N),NN))/sqrt(N),'g'), hold on
```

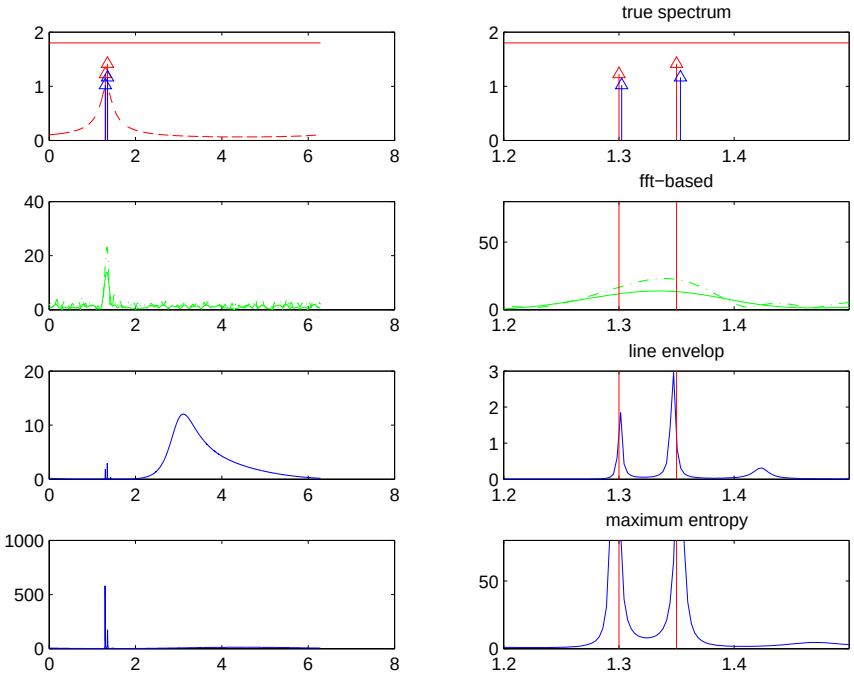

Figure 2: Original and reconstructed power spectra

```
                plot(th,abs(fft(y,NN))/sqrt(N),'g-.')
                vaxis=axis; axis([1.2 1.5 vaxis(3) 2*vaxis(4)]);
                plot([o1 o1],[vaxis(3) 2*vaxis(4)],'r');
                plot([o2 o2],[vaxis(3) 2*vaxis(4)],'r');
title('fft-based')

% setting up filter parameters and plotting response
thetamid=1.325; [Ah,bh]=cjordan2([5],[0.88*exp(thetamid*j)]);
              sv=Rsigma(Ah,bh,th);
              subplot(4,2,1), plot(th(:),sv(:),'r--')

% obtaining state statistics
P=dlsim_complex(Ah,bh,y');

% obtaining and plotting subspace analysis results
[omega_ss,residues_ss]=sm(P,Ah,bh,2);
               subplot(4,2,1), hold on, arrowb(omega_ss,residues_ss)
              subplot(4,2,2), hold on, arrowb(omega_ss,residues_ss)

% obtaining and plotting spectral envelope and me-spectrum
```

```
sqrtrho=envlp(P,Ah,bh,th,.2);
spectrum=me(P,Ah,bh,th);
              subplot(4,2,5), plot(th,sqrtrho.^2),
              subplot(4,2,6), plot(th,sqrtrho.^2),
              hold on, axis([1.2 1.5 0 3]),
              plot([o1 o1],[0 3],'r'),
              plot([o2 o2],[0 3],'r'),
title('line envelop')

              subplot(4,2,7), plot(th,spectrum),
              subplot(4,2,8), plot(th,spectrum),
              hold on, axis([1.2 1.5 vaxis(3) 2*vaxis(4)]),
              plot([o1 o1],[vaxis(3) 2*vaxis(4)],'r'),
              plot([o2 o2],[vaxis(3) 2*vaxis(4)],'r'),
title('maximum entropy')

%print -depsc demo1.eps
%subplot(2,1,1), plot(real(y))
%title('time-series (real and imaginary parts)')
%subplot(2,1,2), plot(imag(y))
%axis([0 100 -10 10])
%print -depsc demo1_timesignal.eps
```