* **Generate an AR model by solving normal equations**
* Estimate:
  + **Direct estimate method**
  + use **xcorr( )**
* AR normal equations
  + Linear equation with Toeplitz Hermitian form
  + **Use Rxx = toeplitz( );**

% compute AR model using autocorrelation (solve for a1)

autocorrx = xcorr(hrs);

[m n] = size(autocorrx);

r0 = (m+1)/2;

rx = autocorrx(r0+1:r0+P);

Rxx = toeplitz(autocorrx(r0:r0+P-1));

a1 = Rxx\(-1\*rx);

* **Plot an estimated power spectrum with the AR model spectrum**
* **Estimate Power spectrum** 
  + Oscillations of 𝜎 best viewed in **frequency** domain
  + Estimate

% compute signal power spectrum

Px = (1/N) \* (abs(fft(hrs,N))).^2;

% compute frequency response

pSpecL = (N+1)/2;

[h,w] = freqz(1,[1 a1'],pSpecL);

Ph = abs(h).^2;

% plot signal power spectrum and AR model power spectrum

figure(),

plot(w,10\*log10(Px(1:pSpecL))); hold on;

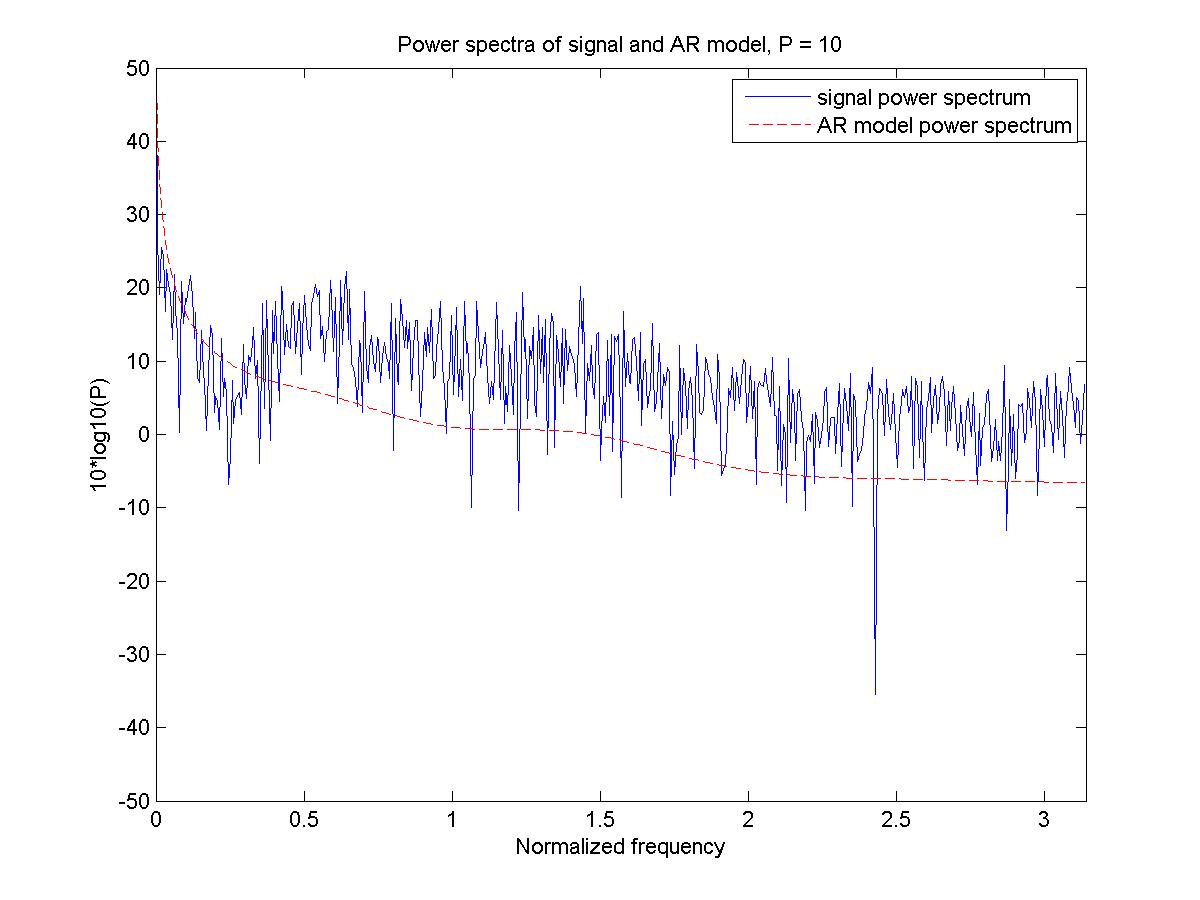
plot(abs(w),10\*log10(Ph),'r--');

legend('signal power spectrum', 'AR model power spectrum');

title(['Power spectra of signal and AR model, P = ' num2str(P)]);

axis([0 pi -50 50]); xlabel('Normalized frequency'); ylabel('10\*log10(P)');

saveas(gcf,'autocorr\_Ph.jpg');



**Figure 1** Power Spectra using autocorr() function

* **Show the innovation process and its power spectrum**

% compute innovation process with inverse filter (solve for Pv)

v = filter(-1\*[1 a1'],1,hrs);

Pv = (1/N) \* (abs(fft(v,N))).^2;

% plot innovation process and its power spectrum

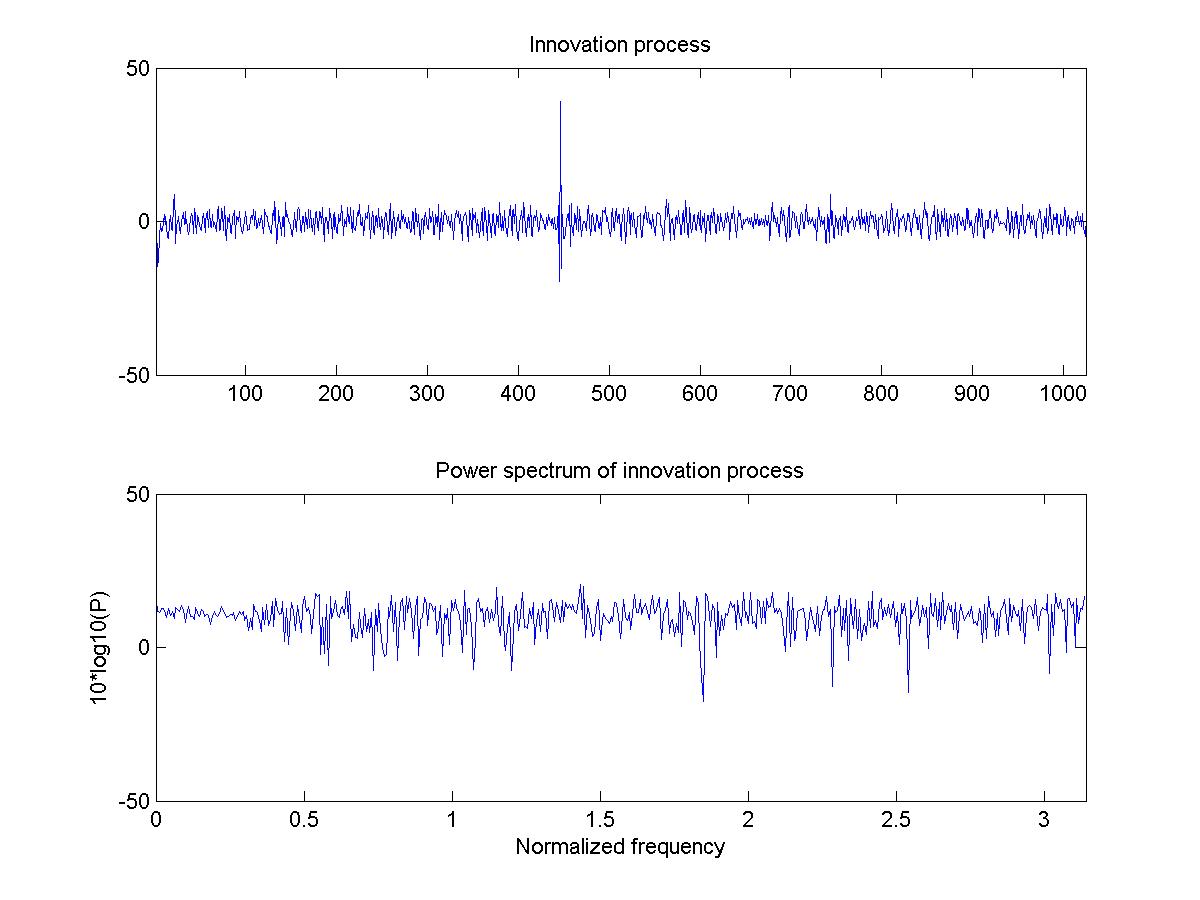
figure;

subplot(2,1,1);plot(v);title('Innovation process');axis([1 N -50 50]);

subplot(2,1,2);plot(w,10\*log10(Pv(1:pSpecL)));title('Power spectrum of innovation process');

axis([0 pi -50 50]); ylabel('10\*log10(P)'); xlabel('Normalized frequency');

saveas(gcf,'autocorr\_Pv.jpg');



**Figure 2** Modeling v[n] using autocorr() function

* **Compute the LPC coefficient using lpc(x,P)**

%% Linear Prediction

% compute AR model with lpc

P = 10;

a = lpc(hrs,P);

% compute linear prediction with AR model

est\_x = filter(-1\*[0 a(2:end)],1,hrs);

* **Compare your AR results with LPC**

% compute frequency response

pSpecL = (N+1)/2;

[h,w] = freqz(1,[a'],pSpecL);

Ph = abs(h).^2;

% plot signal power spectrum and AR model power spectrum

figure(),

plot(w,10\*log10(Px(1:pSpecL))); hold on;

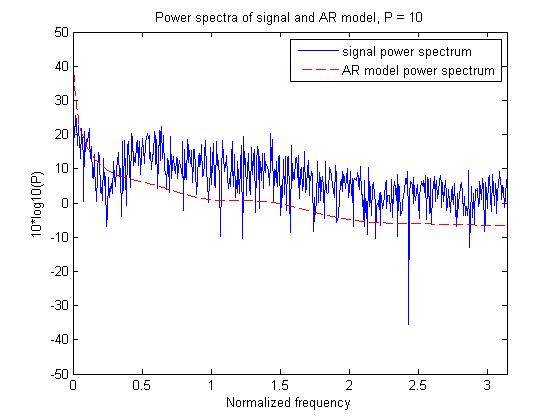
plot(abs(w),10\*log10(Ph),'r--');

legend('signal power spectrum', 'AR model power spectrum');

title(['Power spectra of signal and AR model, P = ' num2str(P)]);

axis([0 pi -50 50]); xlabel('Normalized frequency'); ylabel('10\*log10(P)');

saveas(gcf,'autocorr\_Ph.jpg');

****

**Figure 3** Power Spectra using lpc( ) function

* **Show the linear prediction estimation error**

%% LPC Estimation Error

% show linear prediction estimation error

err = est\_x - hrs;

figure;

subplot(2,1,1);plot(err);

title('Linear prediction estimation error');

axis([1 N -50 50]);

saveas(gcf,'lpc\_err.jpg');

% estimated error power spectrum

Perror = (1/N) \* (abs(fft(err,N))).^2;

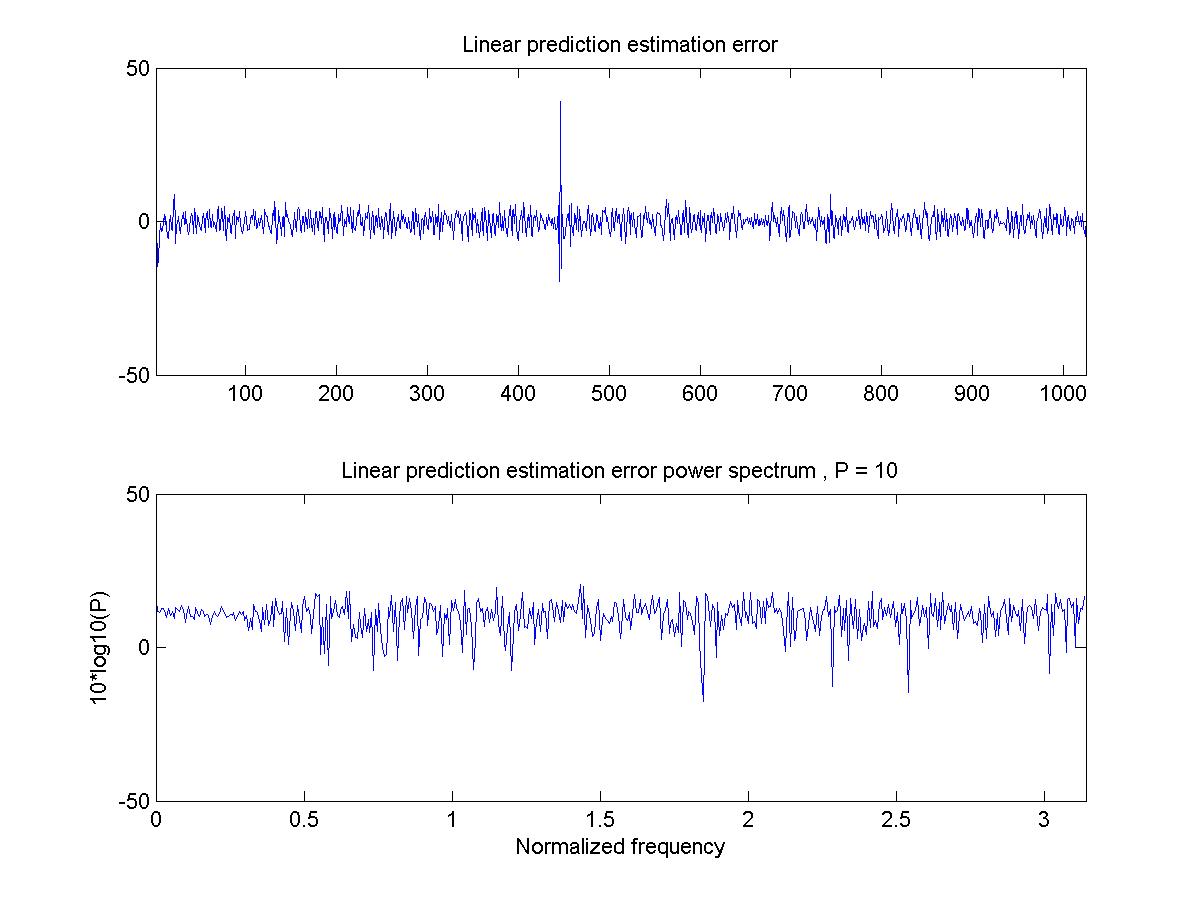
subplot(2,1,2);plot(w,10\*log10(Perror(1:pSpecL)));

axis([0 pi -50 50]);

title(['Linear prediction estimation error power spectrum , P = ' num2str(P)]);

ylabel('10\*log10(P)');xlabel('Normalized frequency');

saveas(gcf,'lpc\_Perr.jpg');

****

**Figure 4** Error of LPC AR model

* **Computer the signal to noise ratio (SNR) of LPC model**

% Compute Signal to Noise Ratio of LPC model

SNR=sum(est\_x.^2)/sum(err.^2);

% SNR for P=10 is 348.69

* **Show SNR in dB as a function of model order P**

% SNR in dB as function of model order P

P=100;

SNR=[];

dBSNR=[];

for i=1:P

% compute AR

a=lpc(hrs,i);

% compute linear prediction with AR model

est\_x = filter(-1\*[0 a(2:end)],1,hrs);

% solve for e

err=hrs-est\_x;

%solve for SNR in dB

SNR(i)=sum(est\_x.^2)/sum(err.^2);

dBSNR(i)=db(SNR(i));

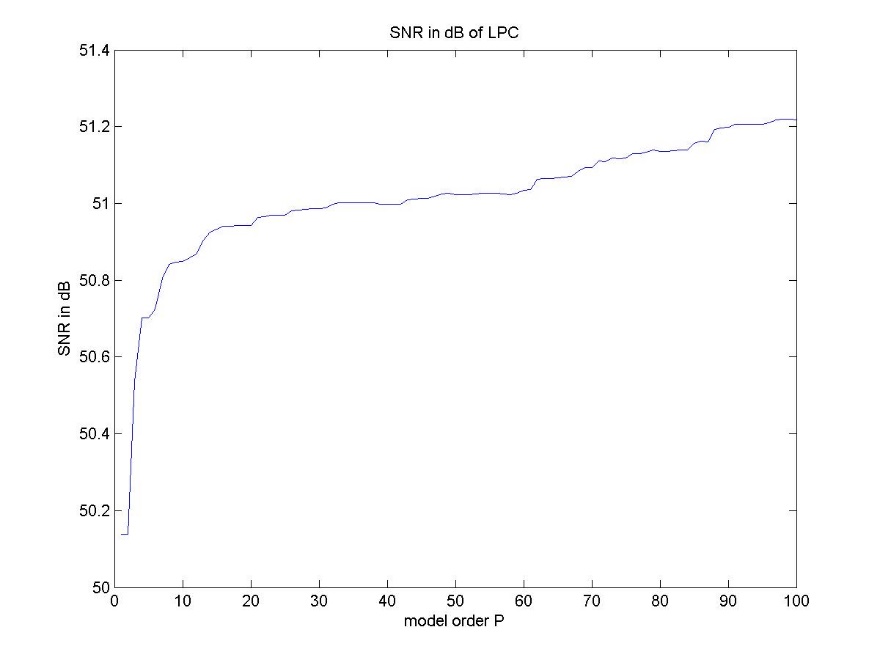
end

figure(),plot(1:100,dBSNR)

title('SNR in dB of LPC');

ylabel('SNR in dB'); xlabel('model order P');

saveas(gcf,'lpc\_dbSNR.jpg');



**Figure 5** dBSNR vs. model order P