```
%EE513 HW#6
%Charles Vanderpool
% This script Performs an LPC Analyisis on an audia signal and
Estimates
% the Length of the vocal tract. It also plays the LPC Filtered
signals as
% well as the Adjusted LPC filtered signals.
%
%
%
%% initialize
%[sig, fs] = audioread('aaa3.wav');
[sig, fs] = audioread('uhh2.wav');% import wav file to analyze
c = 345; %speed of sound
lpcorder = 10; %LPC order
border = 5; %butterworth filter order
winlen = 1024; %window length in samples
wind = hamming(winlen);%generate window
SNR = -10; %snr in db
cutfreq = 80;%butterworth cutoff frequency
tax = (0:length(sig)-1)/fs;%generate time axis

%std = 10^(noiselev/20);
%awgn = randn(1, length(sig)) .*std .*sqrt(2);
%% High pass filter
[b, a] = butter(border, 2*cutfreq/fs, 'high'); %filter LF recording
noise
filtsig = filtfilt(b, a, sig);

%% LPC analysis
[acs, err] = lpc(filtsig, lpcorder);

%% Power spectral density
[psdccseq, psdax] = pwelch(filtsig, wind, winlen-256, winlen*2, fs);

dbpsd = 10*log10(psdccseq); %use log scale
maxp = max(dbpsd); %find min and max for formant lines
minp = min(dbpsd);

%% plot PSD
figure(1);
plot(psdax, dbpsd); title('PSD of Voice Signal')
xlabel('Frequency'); ylabel('Amplitude (dB)');
pause;

%% find roots of lpc filter
poles = (roots(acs));
freqroots = (fs/2).*angle(poles)./pi; %get frequency locations of
poles
nfreqroots = find(freqroots > 0 & freqroots < fs/2); %cut conjugate
poles
```

```
%% indicate formant locations
hold;
for k = 1:length(nfreqroots)
    plot([freqroots(nfreqroots(k)), freqroots(nfreqroots(k))], [minp,
maxp], 'k--');
end
hold off;

%% determine formant frequencies and vocal tract length
ffreqs = sort(freqroots(nfreqroots));
for k = 1:length(nfreqroots)
    lengths(k) = (2*k-1)*c/(4*ffreqs(k));
end
traclen = mean(lengths);
sprintf('Estimated Vocal Tract Length: %d', traclen); %this estimation
seems to be inaccurate when not using aaa3.wav
pause;
%% PART B

sigpow = sum(abs(sig.^2))/length(sig); %find power in signal
noiselev = sigpow*10^(-.5); %determine noise amplitude
awgn = randn(length(sig), 1) .* noiselev; %generate noise
nsig = sig + awgn; %add noise

predictseq = filter(1, acs, nsig); %apply lpc filters
errseq     = filter(acs, 1, nsig);

[psdpre, psdax] = pwelch(predictseq, wind, winlen-256, winlen*2, fs);
figure(5555);
plot(psdax, psdpre); title('PSD of Predicted sequence');
xlabel('Frequency'); ylabel('Power');

disp('Playing Predicted Sequence');
figure(45);
plot(tax, predictseq); title('Predicted Sequence');
xlabel('Time (S)'); ylabel('Amplitude');
soundsc(predictseq, fs);
pause

figure(67);
plot(tax, errseq);title('Predicted Sequence');
xlabel('Time (S)'); ylabel('Amplitude');
disp('Playing error sequence');
soundsc(errseq, fs);
pause;

%% PART C
w = [0:.001:2*pi]; %omega axis
figure(70); %plot poles
plot(real(poles), imag(poles), 'xr', real(exp(j*w)), imag(exp(j*w)),
'b');
title('Pole-Zero plot Of LPC Filter');
```

```
xlabel('Real'); ylabel('Imag');
abspol = abs(poles); %find pole magnitudes

normpol = poles ./ abspol; %normalize poles
magshift = .2; %set shift factor
newpols = normpol * magshift; %shift poles

[zb, za] = zp2tf([], newpols, 1); %create new filter
shiftsig= filter(zb, za, nsig); %use new filter

zr = roots(za); %find poles

soundsc(shiftsig, fs);
disp('Playing Pole-Shifted Signal')

figure(500); %plot shifted poles
plot(real(zr), imag(zr), 'xr', real(exp(j*w)), imag(exp(j*w)), 'b');
title('Pole-Zero plot with shifted poles');
xlabel('Real'); ylabel('Imag');




% figure(1);
% plot(tax, sig); title('Signal Waveform');
% xlabel('Time (s)'); ylabel('Amplitude');
```