

CIEG 675

Homework #5 Due **Tuesday October 9, 2007**

In a m-file do the following and verify it works by copy and pasting into the command window or running your m-file.

Load the data file from my website called frfsep2006\_H\_T.txt. It is a file from Duck, North Carolina of peak wave height (column 1) and period (column 2) in 8 m water depth. Each data point is obtained over 3 hours.

- 1) First remove all NaN's (these are location in the huge original data matrix where wave heights and periods were not recorded). To do this, use your experience with the `find` command and a new command called **isnan**. The output from `isnan` is either 1 where the data is a nan or 0 where the data is not a nan. Data can be removed from a vector or matrix by setting it equal to empty defined in matlab as `[]`.

```
A= load('frfsep2006.txt');
```

```
%% prob 1 %%
```

```
l=find(isnan(A(:,1))==1); % find all the rows in column 1 that are nans  
% expected to be same for all the rows in col2  
% as well.
```

```
A(l,:)=[]; % set those rows to empty means remove them from the variable
```

- 2) make a time vector starting at 0 with 3 hour spacing and plot the data as 2 subplots.

```
t=0:3:2000;
```

```
t=t(1:length(A)); % gets it to be the right length
```

```
subplot(211)
```

```
plot(t,A(:,1))
```

```
set(gca,'XTickLabel',[])
```

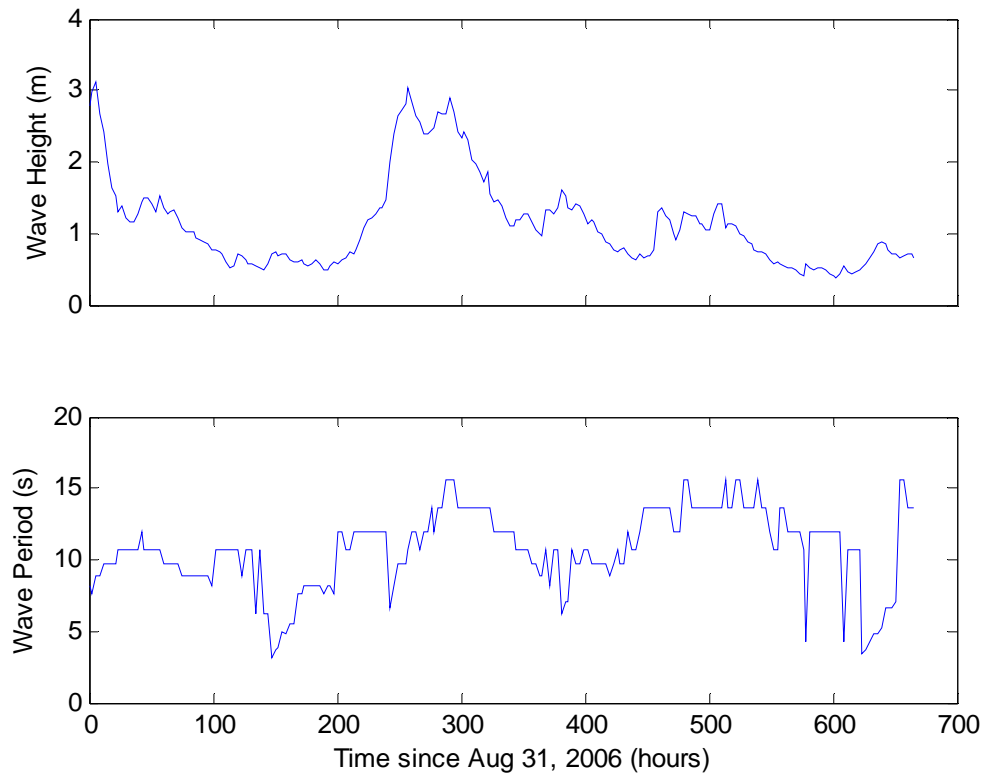
```
ylabel('Wave Height (m)');
```

```
subplot(212)
```

```
plot(t,A(:,2))
```

```
ylabel('Wave Period (s)');
```

```
xlabel('Time since Aug 31, 2006 (hours)');
```



- 3) Calculate the mean, median, min and max significant wave heights and periods for this month.

```
ME=mean(A,2);
```

```
MED=median(A,2); % staitical measures of the data.
```

```
    % Using the 2 means to operate down the columns of A
```

```
    % permits it to be done in one step.
```

```
MI=min(A,2);
```

```
MA=max(A,2);
```

4) Make a histogram of each data set using appropriate bin centers.

```
[NH,XH]=hist(A(:,1),[0:0.25:3.5]); % get hist data or wave height  
% if you ask for output, you have to use bar  
% command to plot it up.
```

```
[NT,XT]=hist(A(:,2),[3:0.5:16]); % hist data of wave period
```

```
clf
```

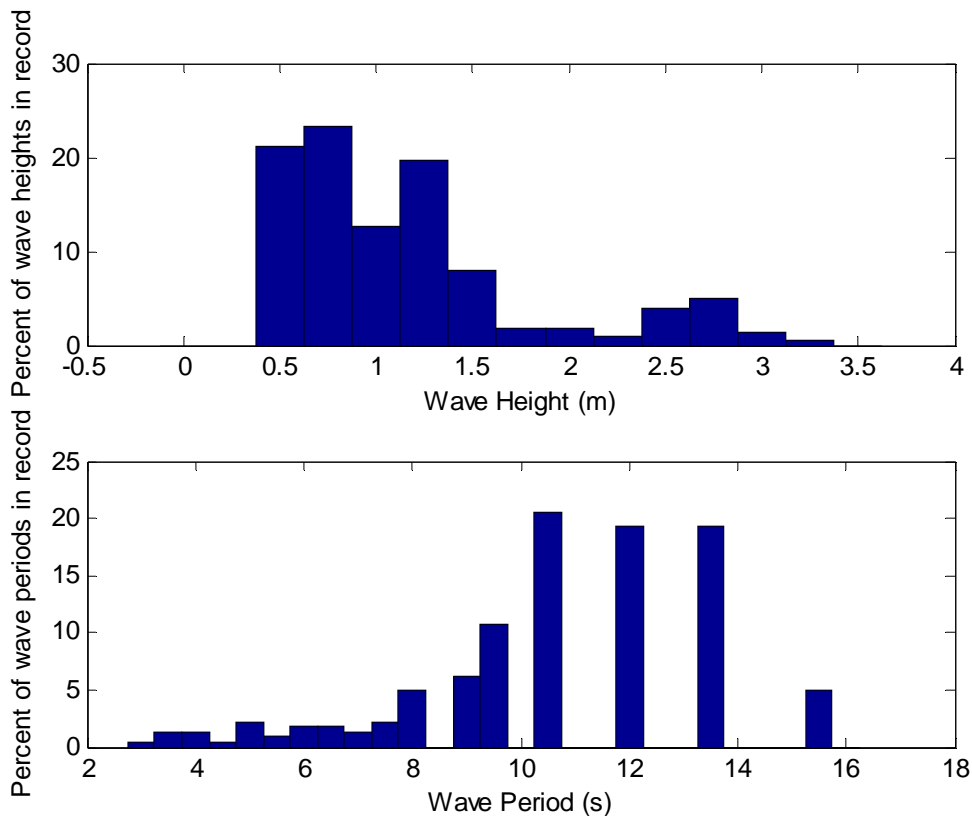
```
subplot(211)
```

```
bar(XH,100*NH/sum(NH),1); % the 1 says make the bars the full width between centers  
% dividing by the sum of Nh, make it a percent contained in  
% bin, versus number of hits
```

```
xlabel('Wave Height (m)');  
ylabel('Percent of wave heights in record');
```

```
subplot(212)
```

```
bar(XT,100*NT/sum(NT),1);  
xlabel('Wave Period (s)');  
ylabel('Percent of wave periods in record');
```

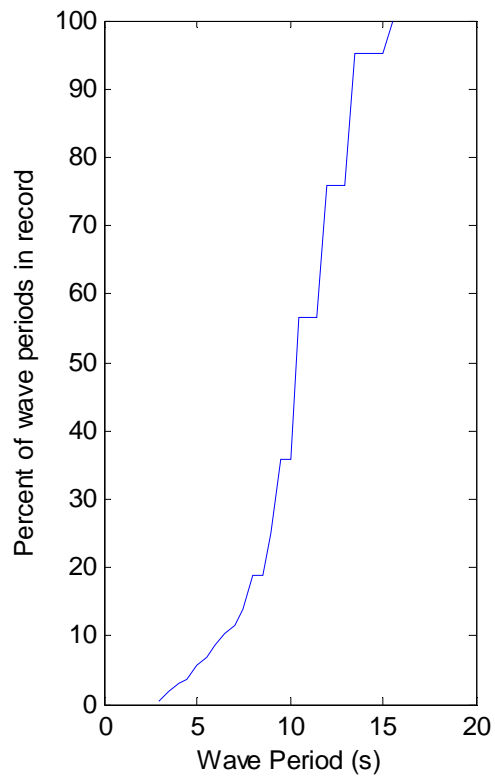
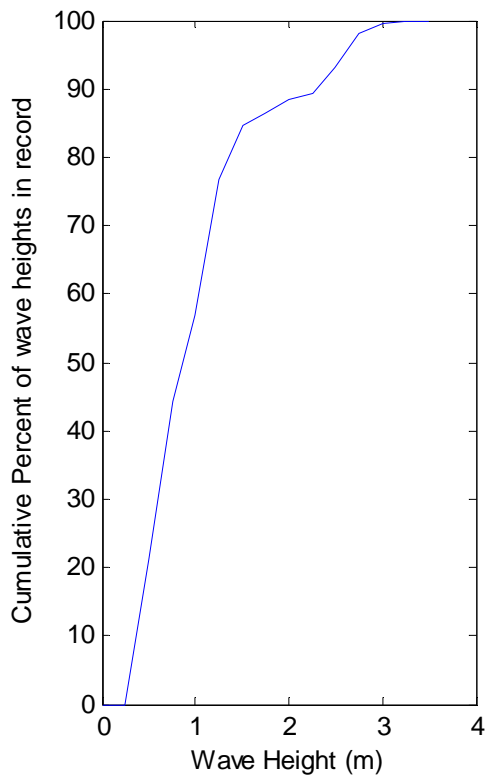


5) Plot the cumulative probability of wave height and period from these data sets.

```
% since we have the percents already from above, we just use cumsum and it  
% better get to 100 %
```

```
clf  
subplot(121)  
plot(XH,cumsum(100*NH/sum(NH))) % uses the cumsum function  
xlabel('Wave Height (m)');  
ylabel('Cumulative Percent of wave heights in record');
```

```
subplot(122)  
plot(XT,cumsum(100*NT/sum(NT)))  
xlabel('Wave Period (s)');  
ylabel('Percent of wave periods in record');  
axis([0 20 0 100])
```



- 6) Develop a data set that is 10000 points long composed of random numbers from a normal distribution. Make a histogram of this data set and then overlay the Gaussian curve on top. You will have to determine how to normalize your plot or histogram to get the y-axis to scale appropriately.

```
data = randn(10000,1); % 10000 random points from gaussian distribution
```

```
clf
```

```
[N,X]=hist(data,[-5:0.25:5]); % get histogram data using specified centers
```

```
bar(X,N/sum(N),1); % use bar to make the histogram (bar) plot
```

```
hold on % hold plot for overlay
```

```
%% now add the gaussian function. To do so, we need to know the mean and
```

```
%% standard deviation
```

```
ME=mean(data);
```

```
ST=std(data); % since from a normal distribution,
```

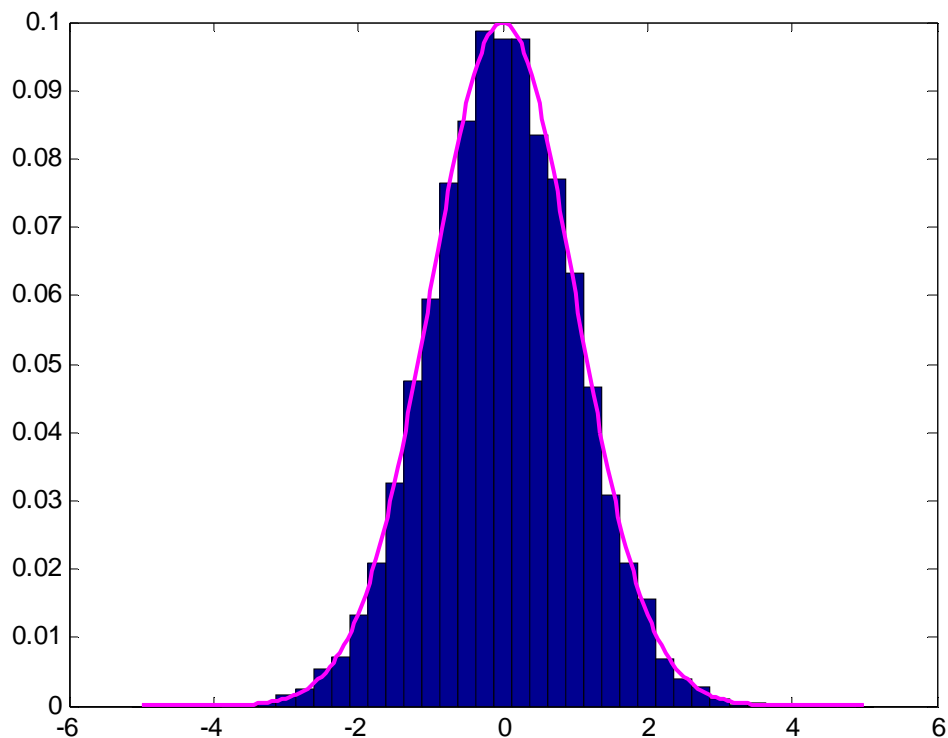
```
    % these better be pretty close to 0 and 1 respectively
```

```
%% now make the gaussian function.
```

```
xg = -5:0.05:5; % some x vector
```

```
yg = 1/(sqrt(2*pi)*ST) * exp(-(xg-ME).^2/(2*ST^2)); % the normal distribution
```

```
plot(xg,0.25*yg,'m','linewidth',2); % plot it and scale by 0.25 to get the peaks to about  
line up
```



7) From the following data you will generate, perform a power spectral density calculation using pwelch

```
[Pxx,F] =PWELCH(X,WINDOW,NOVERLAP,NFFT,Fs).
```

Make the time vector, t, long say out to 1000 with spacing of 0.01. Mess with the different parameters in pwelch to see what they do to change the result. We talked briefly about them in class.

```
y = 4*cos(2*pi*t) + sin(2*pi*t/0.2) + 0.01*randn(1,length(t))
```

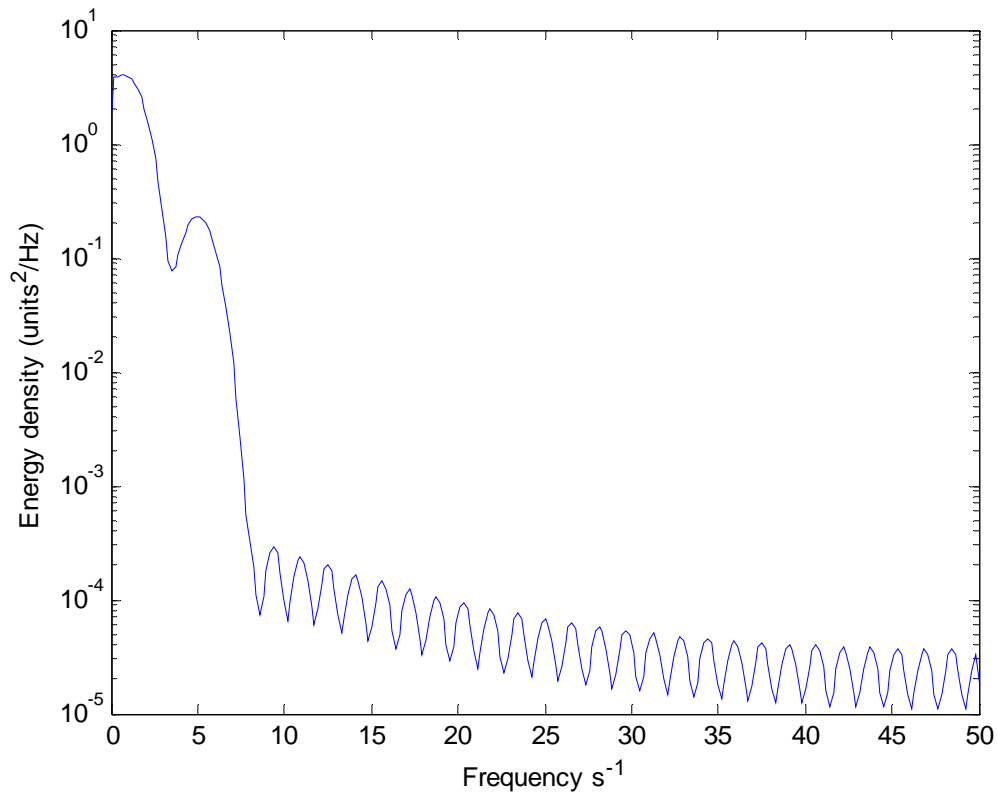
To see data from pwelch, you normally plot on a semilogy plot with Pxx as a function of F (frequency). Tell me what the plot tells you with respect to the given function y.

```
t=0:0.01:1000; % time vector
y=4*cos(2*pi*t) + sin(2*pi*t/0.2)+0.01*randn(1,length(t)); % data generated

%% now look at PSD calculation
[Pxx,F] =PWELCH(X,WINDOW,NOVERLAP,NFFT,Fs).
% Pxx is the output from the calculation,
% F are the frequencies where data obtained.
% X is data in
% window is the window length (how many times to chop up data) smaller
%     number is more smooth data, want bigger number so number of
%     frequency bins averaged over is smaller.
% noverlap how many samples should overlap from section to section (50% is
common and default)
% NFFT is how many frequency bins give an answer for.
%     smaller numbers smooths results more, really get NFFT/2
% Fs is the frequency of data coming in. For us it is 1/0.01 = 100

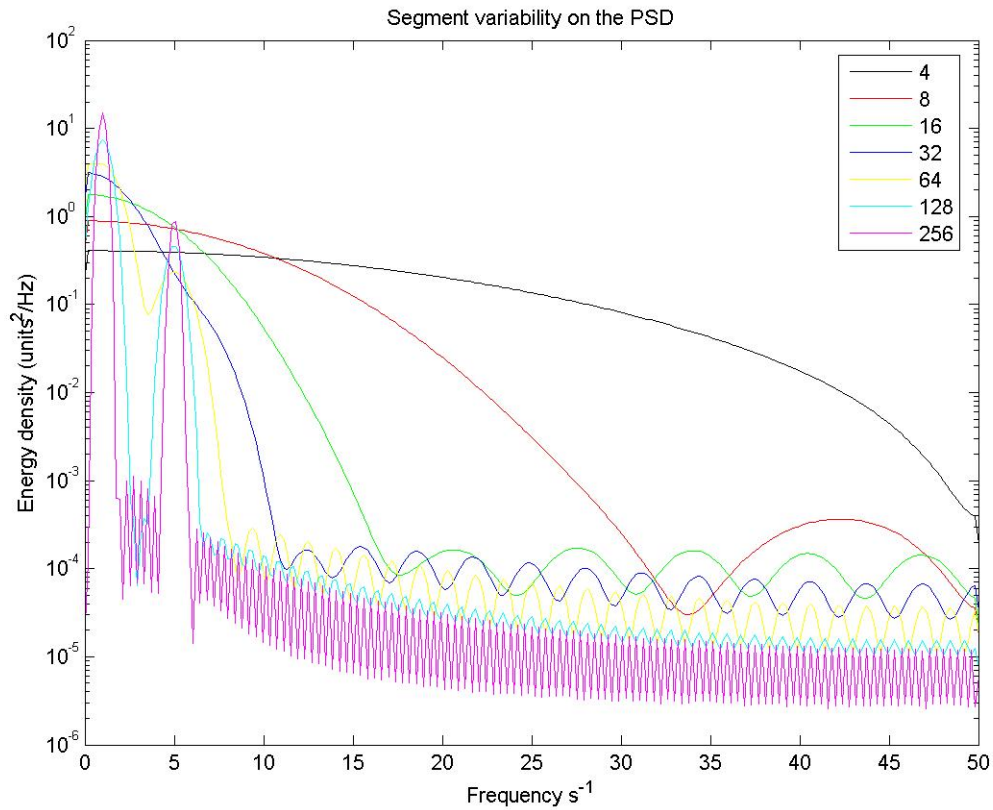
[Pxx,F] =pwelch(y,64,[],512,100); % 64 data segments, 50 % overlap ([] here means
use default)
                                % length(y)/512 adjacent frequency bins averaged, 100
                                % Hz data

clf
semilogy(F,Pxx);
xlabel('Frequency s^{-1}');
ylabel('Energy density (units^2/Hz)');
```



Note that the function told us we had two sinusoids with frequency of 1 and frequency of  $1/0.2 = 5$ . Thus we expect to see peaks in the energy at these frequencies and we do. If values for `noverlap` and `nfft` are chosen incorrectly then these peaks may get smoothed over. All the extra small peaks and ringing comes from the random noise we added by the `randn` term that has a lot of different frequencies, but all are high frequency noise.

```
%% see what happens as we change the # of data segments
%% holding other values the same
clf
segments=[4 8 16 32 64 128 256];
colors='krgbycm';
for k=1:length(segments);
    [Pxx,F]=pwelch(y,segments(k),[],512,100);
    txt=sprintf('semilogy(F,Pxx,"%s")',colors(k))
    eval(txt)
    hold on
end
legend('4','8','16','32','64','128','256')
title('Segment variability on the PSD');
xlabel('Frequency s^{-1}');
ylabel('Energy density (units^2/Hz)');
```

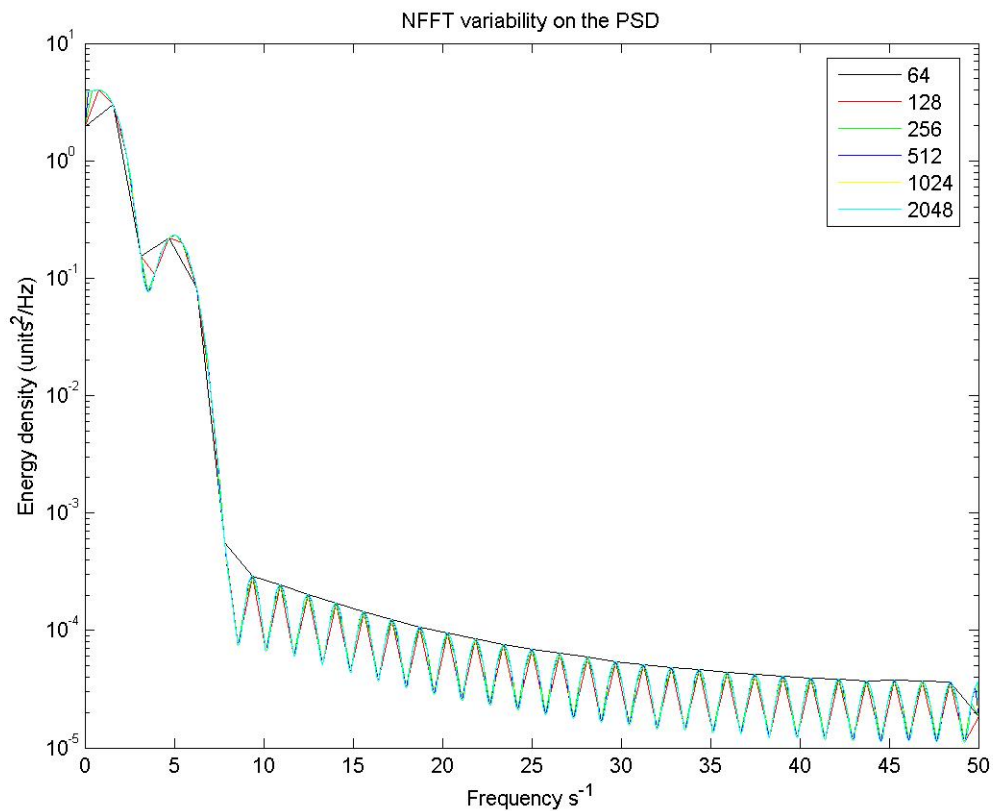


%% if only a small number of segments are selected there is way too much averaging and the spectral peaks are not seen where they should be. Once we get up to 64 and above, the spectral peaks appear where they should but there is still some broadening of the peak. Some of this is known as spectral leakage and is due to a finite record length.

```

%% NOW see what happens as we change the NFFT #
%% holding other values the same
clf
NFFT=[64 128 256 512 1024 2048];
colors='krgbycm';
for k=1:length(NFFT);
    [Pxx,F] =pwelch(y,64,[],NFFT(k),100);
    txt=sprintf('semilogy(F,Pxx,"%s")',colors(k))
    eval(txt)
    hold on
end
legend('64','128','256','512','1024','2048')
title('NFFT variability on the PSD');
xlabel('Frequency s^{-1}');
ylabel('Energy density (units^2/Hz)');

```



As we alter the number of points contained in each FFT estimate, the resulting plot changes less. Factors of 2 work best and for the small values used here, the peaks are still in the right spots but not resolved very well. Note that once we get to about 256 points in each FFT, the peaks where we expect them are better resolved but still show

some spectral spread. There is much noise in the higher frequencies some of it attributed to the noise that was added to the signal through the rand function.