SIOC 221A Homework 6 **Julia Dohner**
Prof. Sarah Gille Due Date: November 9, 2017

1. Evaluate whether using a 50% overlap modifies the degrees of freedom.

The ratio of the high to low empirical error bars from the Monte Carlo simulation is 1.93 with windowing, and 2.19 without windowing. I calculated the error bars using several different values for my effective degrees of freedom. First, I tried the scenario in which I declared that by splitting the data into segments with 50% overlap, I doubled the effective degrees of freedom (so nu = 2*39, for 39 segments). This yielded a ratio of high to low error bars of 1.88. Next, I followed the first option in the table from Lecture 9 that stated that when applying a Hanning window, the effective degrees of freedom are 8/3 * # of segments. This treatment yielded an error ratio of 2.15. Finally, I tried the second option from the Lecture 9 table in which the degrees of freedom are 0.75 * # of segments, which yielded a ratio of 2.84. From these trials, I determined that treating each segment as an independent segment of data came closest to replicating the error from the Monte Carlo trials. This is clearly problematic because we know that when creating segments with 50% overlap, the segments certainly are not independent. It also goes to show that the estimates for the factor by which the degrees of freedom increase in the Lecture 9 table is problematic, since neither estimate yielded a better result than simply treating each segment as an independent segment of data.

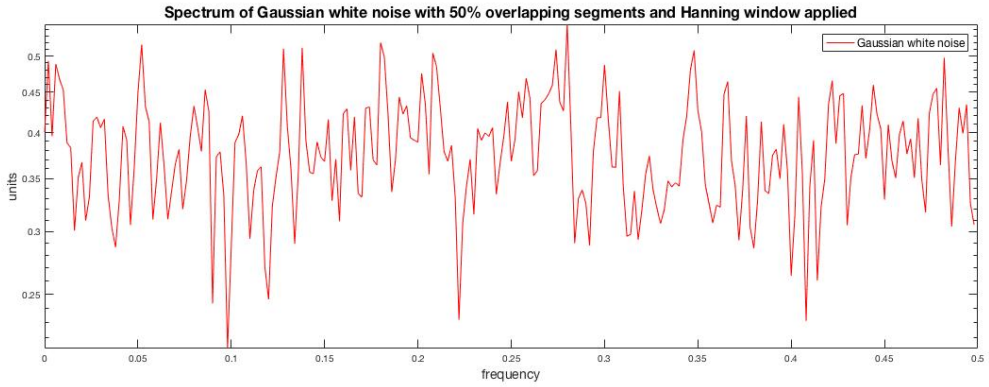
Running the Monte Carlo simulation without windowing yields a high to low error ratio of 2.19, which more closely matches the result when multiplying the degrees of freedom by the factor of (8/3) in the Hanning window case. Sadly this is likely a happy coincidence, since there's no explanation for why the Hanning window factor would better match the error estimate for when a Hanning window is NOT applied. This again goes to show the faultiness of the factors listed in the table in the Lecture 9 notes.

The use of overlapping segments will not ever decrease the number of degrees of freedom. At a minimum, I will still have the same number of degrees of freedom I would've had had I not segmented the data. Using overlapping segments and applying a window should increase the number of effective degrees of freedom, since there are more separate segments of data with different parts of the data emphasized between segments.

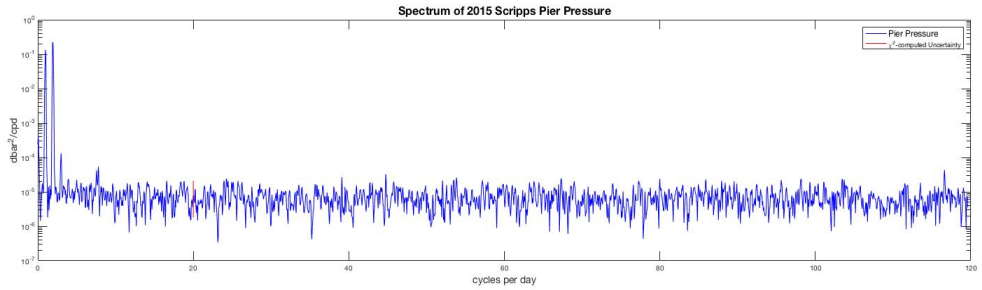
If I don't window, the overlapping segments do not reduce my effective degrees of freedom; theoretically, they should not change the degrees of freedom, since I am not adding any new information by creating overlapping segments without windowing.

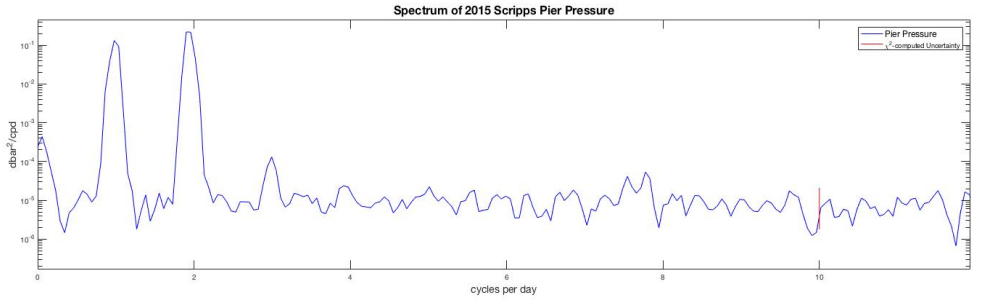
I think my results would have changed had I considered red noise because segmenting and windowing data with actual structure (in contrast to white noise) will perturb the results in a more significant way, whereas white noise should theoretically be unperturbed by these manipulations.

In the end, I'd say that using the 50% overlap does modify my degrees of freedom, but only in the case that I apply a Hanning window as well. This conclusion is more supported by our class lectures and thinking about what is happening when we window data, and less so by the results of my homework experimentation, unfortunately.



2. Compute a spectrum of the pressure data.





```
% file SIOC 221A HW 6
%
% author Julia Dohner, with help from Annie Adelson
%
% due date November 9, 2017

clear all; close all;


%% Evaluate whether using a 50% overlap modifies the degrees of freedom

% 10,000 is the number of datapoints
N = 500; % length of each chunk of data (aka segment length)
M = 20; % number of segments splitting data into

% non-Monte Carlo case, using overlapping segments and Hanning window
longVector = randn(N*M,1);
longVector_1 = reshape(longVector,N,M);
longVector_2 = longVector(N/2+1:length(longVector)-N/2,:);
longVector_2 = reshape(longVector_2,N,M-1);
longVector_3 = [longVector_1 longVector_2];

longVector_3 = detrend(longVector_3).*(hann(500)*ones(1,39));
longVector_3 = fft(longVector_3);
longVector_3amp = (abs(longVector_3(1:N/2+1,:)).^2)/N;
longVector_3amp = longVector_3amp(2:251,:);
longVector_3mean = mean(longVector_3amp, 2);
longVector_3mean = longVector_3mean';
frequency=(0:N/2-1)/N;

figure
semilogy(frequency,longVector_3mean, '-r')
xlabel('\fontsize{14}frequency')
ylabel('\fontsize{14}units')
title('\fontsize{16}Spectrum of Gaussian white noise with 50% overlapping segments and Hanning↙
window applied')
legend('\fontsize{12}Gaussian white noise');


%% Monte Carlo without windowing
for i=1:200
    longVector_MC(:,:,i) = randn(N*M,1); % 200 times one long vector
    longVector_MC1(:,:,i) = reshape(longVector_MC(:,:,i),N,M);
    longVector_MC2(:,:,i) = longVector_MC(N/2+1:(N*M)-N/2,:,i); % 200 times one long vector↙
with 50% offset
    longVector_MC3(:,:,i) = reshape(longVector_MC2(:,:,i),[N,M-1]); % 200 times reshaped vector↙
with 50% offset
    longVector_MC4(:,:,i) = [longVector_MC1(:,:,i) longVector_MC3(:,:,i)];
    % detrend, multiply by hanning window
    %longVector_MC4(:,:,i) = detrend(longVector_MC4(:,:,i)).*(hann(500)*ones(1,39));
    longVector_MC4(:,:,i) = fft(longVector_MC4(:,:,i)); % take fft
    longVector_MC4amp(:,:,i) = ((8/3)^(1/2)).*(abs(longVector_MC4(1:N/2+1,:,i)).^2)/N; %↙
scaling by root(8/3) to account for E attenuation
    longVector_MC4amp(2:N/2,:,i) = 2*longVector_MC4amp(2:N/2,:,i);
    longVector_MC5amp(:,:,i) = longVector_MC4amp(2:251,:,i); % dumping the mean

end

% average over 20 realizations within each 500x20 matrix (so averaging
```

```
% across the columns)
longVector_MCmean = mean(longVector_MC5amp,2);
% turn mean amplitude result into a column vector
longVector_MCpdf = longVector_MCmean(:);
err_low_MC = prctile(longVector_MCpdf,2.5);
err_high_MC = prctile(longVector_MCpdf,97.5);
ratio_monteCarlo = err_high_MC/err_low_MC;

% % expectation of error bars based on number of degrees of freedom available
% % with use of overlapping segments:
% %nu = (8/3)*M; % where M = number of segments
% nu = (8/3)*20;
% %nu = 2*39; % if 39 independent segments, 2 DOF for each segment
% %nu = 0.75*39;
% err_high = nu/chi2inv(0.05/2,nu);
% err_low = nu/chi2inv(1-0.05/2,nu);
% ratio_chi2 = err_high/err_low;
%
% % comparing to Monte Carlo value of 1.93...
% % 2 * number of segments: ratio = 1.88 <- this is the closest
% % 8/3 * number of segments: ratio = 2.15
% % 0.75 * number of segments: ratio = 2.84



%% do this 200 times for Monte Carlo with windowing

for i=1:200
    longVector_MCw(:,:,i) = randn(N*M,1); % 200 times one long vector
    longVector_MCw1(:,:,i) = reshape(longVector_MCw(:,:,i),N,M);
    longVector_MCw2(:,:,i) = longVector_MCw(N/2+1:(N*M)-N/2,:,i); % 200 times one long vector↙
with 50% offset
    longVector_MCw3(:,:,i) = reshape(longVector_MCw2(:,:,i),[N,M-1]); % 200 times reshaped↙
vector with 50% offset
    longVector_MCw4(:,:,i) = [longVector_MCw1(:,:,i) longVector_MCw3(:,:,i)];
    % detrend, multiply by hanning window
    longVector_MCw4(:,:,i) = detrend(longVector_MCw4(:,:,i)).*(hann(500)*ones(1,39));
    longVector_MCw4(:,:,i) = fft(longVector_MCw4(:,:,i)); % take fft
    longVector_MCw4amp(:,:,i) = ((8/3)^(1/2)).*(abs(longVector_MCw4(1:N/2+1,:,i)).^2)/N; %↙
scaling by root(8/3) to account for E attenuation
    longVector_MCw4amp(2:N/2,:,i) = 2*longVector_MCw4amp(2:N/2,:,i);
    longVector_MCw5amp(:,:,i) = longVector_MCw4amp(2:251,:,i); % dumping the mean

end

% average over 20 realizations within each 500x20 matrix (so averaging
% across the columns)
longVector_MCwmean = mean(longVector_MCw5amp,2);
% turn mean amplitude result into a column vector
longVector_MCwpdf = longVector_MCwmean(:);
err_low_MCw = prctile(longVector_MCwpdf,2.5);
err_high_MCw = prctile(longVector_MCwpdf,97.5);
ratio_monteCarloWindow = err_high_MCw/err_low_MCw;

% expectation of error bars based on number of degrees of freedom available
% with use of overlapping segments:
%nu = (8/3)*M; % where M = number of segments
nu = (8/3)*20;
%nu = 2*39; % if 39 independent segments, 2 DOF for each segment
```

```
%nu = 0.75*39;
err_high = nu/chi2inv(0.05/2,nu);
err_low = nu/chi2inv(1-0.05/2,nu);
ratio_chi2 = err_high/err_low;

% comparing to Monte Carlo value of 1.93 (with window) and 2.19 (with
% window):
% 2 * number of segments: ratio = 1.88 <- this is the closest
% 8/3 * number of segments: ratio = 2.15
% 0.75 * number of segments: ratio = 2.84




%% compute a spectrum of the pressure data you used in HW3, 4

% create empty arrays to hold time and temp data
time = [];
pressure = [];

time = [time; ncread(strcat('http://sccoos.org/thredds/dodsC/autoss/scripps_pier-2015.↙
nc'),'time')];
pressure = [pressure; ncread(strcat('http://sccoos.org/thredds/dodsC/autoss/scripps_pier-2015.↙
nc'),'pressure')];

% remove bad data using the flagged data from .nc file
pressure_flagPrimary = [];
pressure_flagPrimary = [pressure_flagPrimary; ncread(strcat('http://sccoos.↙
org/thredds/dodsC/autoss/scripps_pier-2015.nc'),'pressure_flagPrimary')];

% looping through to remove bad data from pressure record
for i = 1:length(pressure)
    if pressure_flagPrimary(i) ~= 1
        pressure(i) = nan;
    end
end

% consider a period with equal increments

% this for the first 34 days of the 2015 record
% divide into 2 segments because need at least 14 days to resolve 12-hr tidal
% timescale

% time differences
time = double(time);
t_diff = diff(time);

% finding segment of data with even spacing
cutoff = find(t_diff(1:82236)>t_diff(1),1);
pressure_sub = pressure(1:cutoff-1); % subsampled pressure
pressure_sub = pressure_sub(1:length(pressure_sub)-3);
time_sub = time(1:cutoff-1); % subsampled times
date0=datenum(1970,1,1); % give reference date (first date)
time_sub = double(time_sub)/86400+date0; % in units of days (conversion: seconds/day)

M_pressure = 2; % number of segments
N_pressure = length(pressure_sub)/M_pressure; % datapoints per segment

% split into two segments
```

```
pressure_sub1 = reshape(pressure_sub,N_pressure,M_pressure);
pressure_sub2 = pressure_sub(N_pressure/2+1:length(pressure_sub)-N_pressure/2);
pressure_sub2 = reshape(pressure_sub2,N_pressure,M_pressure-1);
pressure_sub3 = [pressure_sub1 pressure_sub2]; % 3 segments, overlapping

pressure_sub3 = detrend(pressure_sub3).*(hann(4154)*ones(1,3));
pressure_sub3 = fft(pressure_sub3);
T = (4154*361)/(2*24*3600);% total time in days, so datapoints*time interval
normalizationFactor = T/(4154^2); % 4154 = number of data points
pressure_sub3amp = 2.*(abs(pressure_sub3(1:N_pressure/2+1,:)).^2).*normalizationFactor; %↙
amplitude of first half
pressure_sub3mean = mean(pressure_sub3amp,2);
pressure_sub3mean = pressure_sub3mean'; % turn into row vector

frequency = (0:2078-1)/(2*2078*361)*(24*3600);

nu = 2*3; % DOF = 2*number of segments
err_high_pressure = nu/chi2inv(0.05/2,nu);
err_low_pressure = nu/chi2inv(1-0.05/2,nu);
ratio_chi2_pressure = err_high_pressure/err_low_pressure;


figure
semilogy(frequency,pressure_sub3mean, '-b', [10 10],[err_low_pressure err_high_pressure]↙
*pressure_sub3mean(1000), '-r');
xlabel('\fontsize{14}cycles per day')
ylabel('\fontsize{14}dbar^{2}/cpd')
title('\fontsize{16}Spectrum of 2015 Scripps Pier Pressure');
legend('\fontsize{12}Pier Pressure','\chi^{2}-computed Uncertainty');
```