



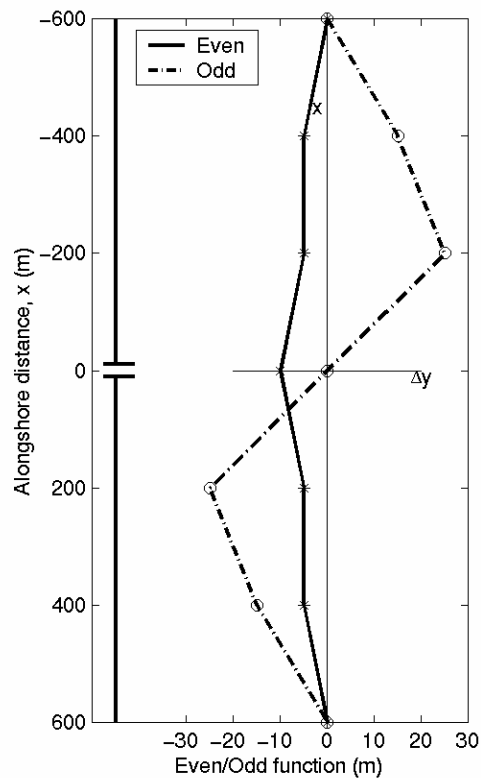
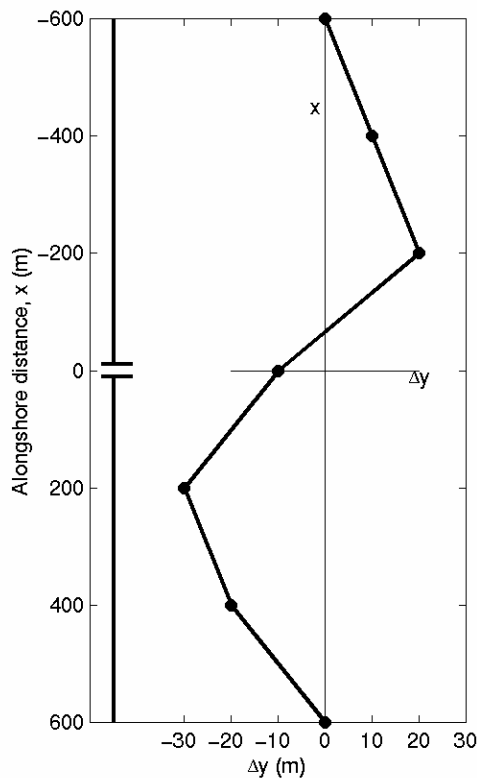
1. Write a Matlab code that will take as input a vector of alongshore positions, corresponding beach width or volume (it should not care which) and a central location to perform even and odd analysis of shoreline change. SEE CODE AT END

2. DD 6.2 (turn in your code with your results)

Using Eqs (6.20 and 6.21) we arrive at the following values for the even and odd functions:

<b>Location, x</b>	<b><math>\Delta y</math></b>	<b><math>\Delta y_{\text{even}}</math> (m)</b>	<b><math>\Delta y_{\text{odd}}</math> (m)</b>
<b>-600</b>	0	0	0
<b>-400</b>	10	-5	15
<b>-200</b>	20	-5	25
<b>0</b>	-10	-10	0
<b>200</b>	-30	-5	-25
<b>400</b>	-20	-5	-15
<b>600</b>	0	0	0

Or in plotted form:



**(b1) From the standpoint of the entire odd function, is any volume loss associated with the odd function.**

Since the odd function is anti-symmetric about the structure, the sum of the odd components must be zero. This can be verified by summing the odd components in the table above.

**(b2) From the standpoint of the entire even function, is any volume loss associated with the even function.**

Since the even function is symmetric, there is no restriction that the components must sum to zero. Indeed, in this case we find that there is a net volume loss of sand associated with the even function as denoted by the negative values in the table and figure above.

**(c) Given  $B = 2$  m and  $h^* = 6$  m, what is the annual loss of sediment to the system.**

Since the odd function integrates to zero as mentioned in part b1, we need only consider the even function here. From the equation for volume generated on an equilibrium beach profile (Section 7.5.1.1) we see that

$$\Delta V = \Delta y(h_* + B) \text{ at any location on the profile.}$$

To obtain the annual sediment loss in the system, we must integrate the above equation across our measurements as

$$Volume\ lost = \int_{-600}^{600} \Delta y(h_* + B) dx$$

This is done in a discretized fashion assuming the average incremental volume loss between two successive measurements. For example, the *even* shoreline recessions associated with locations  $x = 400\text{ m}$  and  $x = 600\text{ m}$  are  $-5\text{ m}$  and  $0\text{ m}$  respectively. The associated average volume loss based on an average shoreline recession of  $\Delta y = \frac{-5 + 0}{2} = -2.5\text{ m}$  at a central location of

$$x = \frac{400 + 600}{2} = 500\text{ m} \text{ is}$$

$\Delta V = -2.5\text{ m} * (6\text{ m} + 2\text{ m}) * 200\text{ m} = -4000\text{ m}^3$ . Continuing for the remaining sections of coastline and summing yields

Average Location of calculated change	$\Delta V$ ( $\text{m}^3/\text{year}$ )
-500	-4,000
-300	-8,000
-100	-12,000
100	-12,000
300	-8,000
500	-4,000
<b>TOTAL VOLUME LOST</b>	<b>-48,000</b>

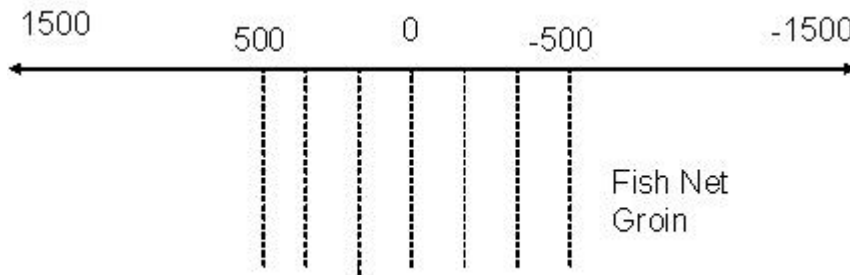
**(d) What is the direction of the longshore sediment transport? Discuss your response.**

Since the odd function shows accretion on the negative side of the jetty and erosion on the positive side of the jetty, it is likely that the alongshore current and sediment transport are from negative to positive in this coordinate system. The reasoning is simply that currents drive sediment from down coast. The sediment reach a barrier and are deposited. On the downdrift side, the currents are still capable of transporting sediment and do so, but because there is no longer a supply from the updrift side, erosion occurs.

**(e) Assume the updrift jetty is so short that sand is transported into the inlet. Interpret the cause of net loss of sediment to the shoreline system.**

If sediment is lost from the shoreline system, they are permanently deposited in regions where they are no longer accessible to the shoreline. The possibilities are offshore or into the bay behind the inlet. The sediment that is carried into the inlet is transported into the bay by tidal currents and forms a flood tidal shoal where sediment is trapped. In addition, an ebb tidal shoal may form outside the inlet trapping sediment there as well.

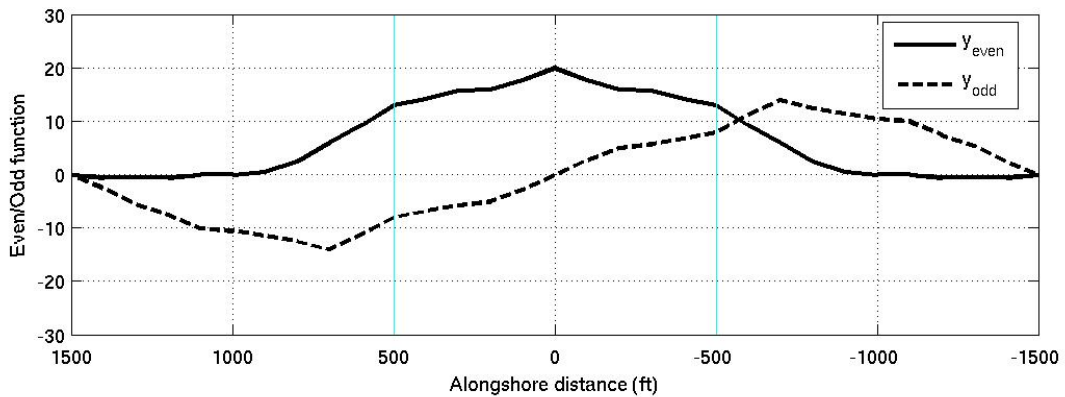
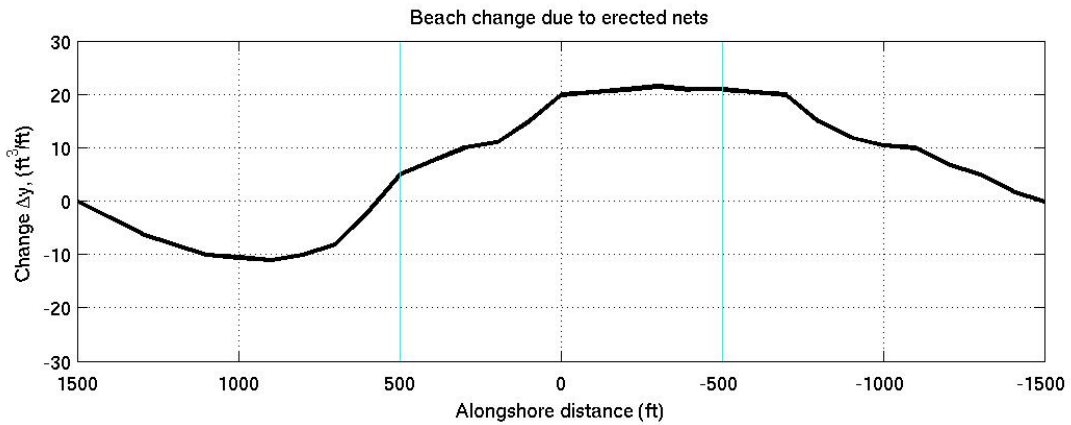
3. An innovative method of erosion control is being tested in Florida. The approach uses a series of cross-shore-directed fish nets that are supposed to cause deposition within. The claim is that the nets will trap sediment from offshore and will not affect the adjacent beaches. Based on the data given, what is your analysis? Did the survey extend to the undisturbed shoreline? What does the even function pattern suggest? What does the odd function pattern suggest? Do you think the nets only trap sediment moving in the cross-shore direction? *Go to my webpage to download a matlab-readable text file so you do not have to re-type these numbers.*



The measured longshore distribution of volumetric change is given as

Alongshore Location	Volume Change $\text{ft}^3/\text{ft}$
1500	0
1400	-3
1300	-6
1200	-8
1100	-10
1000	-10.5
900	-11
800	-10
700	-8
600	-2
500	5
400	7.5
300	10
200	11
100	15
0	20
-100	20.5
-200	21
-300	21.5
-400	21
-500	21
-600	20.5

-700	20
-800	15
-900	12
-1000	10.5
-1100	10
-1200	7
-1300	5
-1400	2
-1500	0



### Analysis:

The survey looks as though it did extend to the undisturbed shoreline since the volumetric changes was zero (barely) at the ends in the alongshore direction.

Based on the odd function, I expect an alongshore current moving from the negative top positive direction (right to left as I have plotted). If the alongshore drift was negligible, the odd function should have been nearly zero at all locations. Thus, it appears that the nets do indeed trap sediment from the alongshore drift.

The even function suggests that the nets do trap sediment moving in the cross-shore direction as the amount of sediment accumulation is greatest at net centers and tapers off from there. However, because of the shape of the odd function, we cannot say that the nets trap sediment only moving in the cross-shore direction.

3. **NOT ASSIGNED!** Write a Matlab code that performs an EOF analysis on a matrix of data. Assume the data you will eventually use will be time separated beach profiles. The output should be the data variance, the ordered eigenvalues, the eigenfunctions and the variation of the eigenfunctions. Go to my webpage and download the matlab file containing several beach profiles from Duck, NC in 2005. The first column is the cross-shore coordinate and the rest of the columns are elevations at different times. They are spaced between 1 and 2 months. What does the EOF analysis tell you? What does the dominant mode represent? If you were to go out to Duck and had to guess what the profile was like, what would you choose? How wrong would you be? What do you think the second mode tells you? How much variance does the third mode describe? Would you likely need to consider it in any further analyses you might perform? SEE CODE AT END

HINTS (i.e. recipe): 1) interp all profiles to gridded x locations (otherwise cannot use this technique this way)

2) put all profiles into a matrix, D, where each row is a profile

**%% STEPS 1 and 2 WERE DONE FOR YOU WHEN I MADE THE FILE**

3) create the normalized data-data matrix  $D^T D / (IK)$ , where I is the number of locations and K the number of surveys

4) find eigenvectors and eigenvalues using matlabs eig function

5) Sort the eigenvalues in descending order

6) Sort the eigenvectors according to the sorted eigenvalues (i.e. they must correspond)

7) You can determine the temporal variation in the eigenvectors by multiplying D times the eigenvectors

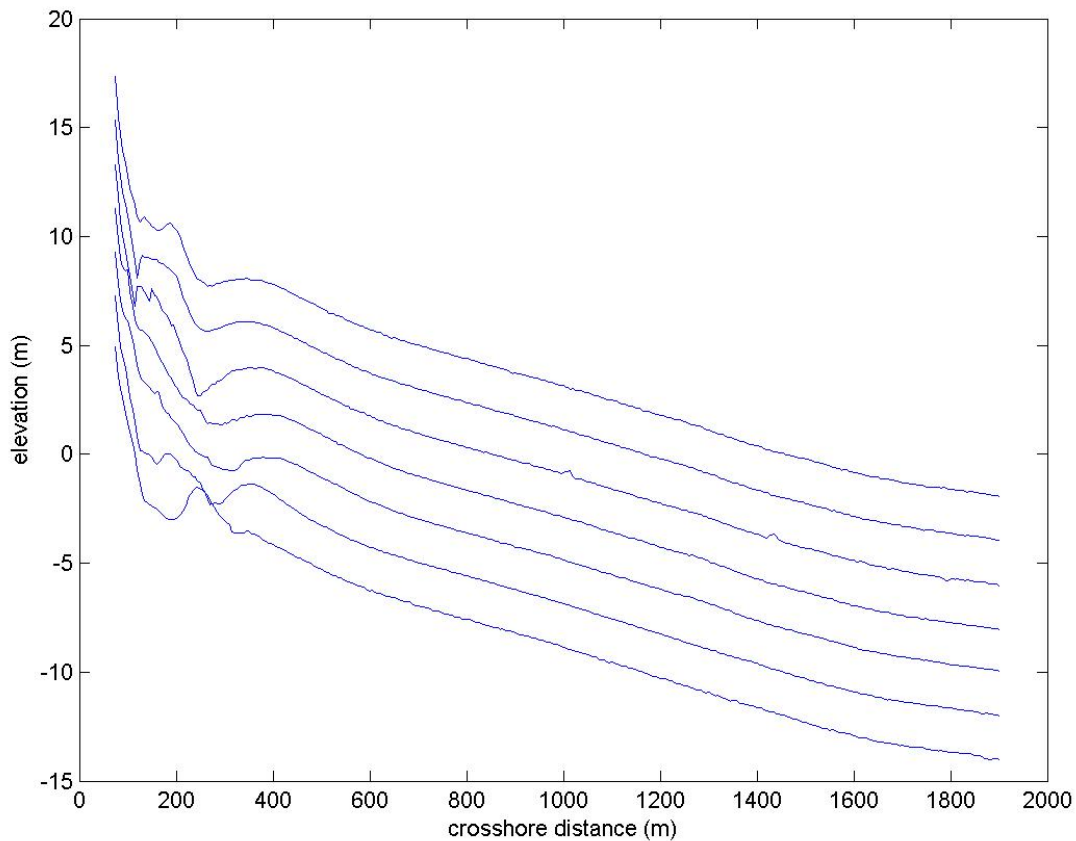
8) You can reconstruct the “mean” profile using just the first eigenvector after it is properly scaled. That means you must multiply it by the square root of  $(I * \text{eigenvalue}(1))$

9) the other eigenvectors tell you the bar/berm (2) and terrace (3) functions.

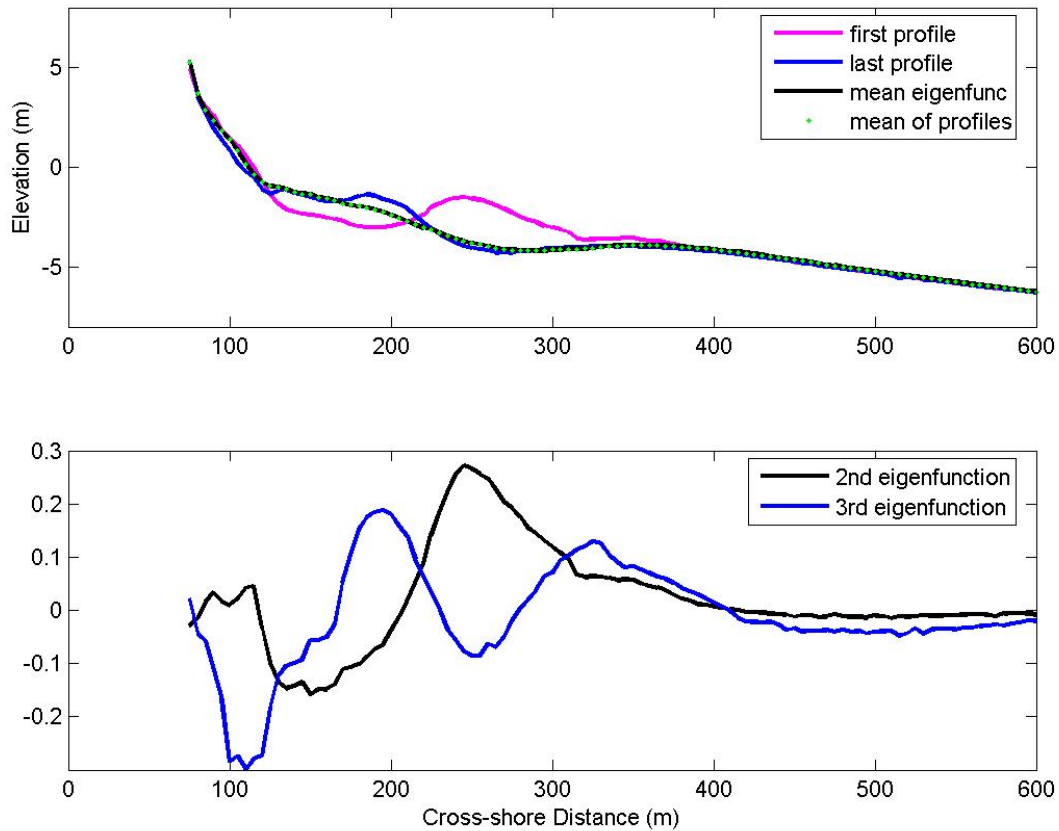
10) the amount of variance explained by each is just each eigenvalue over the sum of the eigenvalues (since the sum of the eigenvalues is the variance).

11) To reconstruct each individual profile (which is a waste of time since you started with them), Just multiply  $D \cdot \text{eigenvector} \cdot \text{eigenvector}^T$ . This will let you make sure you did things correctly, though. Note you won't get it exactly, but the profiles differences should be in the  $10^{-8}$  or lower range. Scaling not needed here since it essentially comes from the magnitude of the data matrix.

12) If you only want to reconstruct the data with the first  $m$  dominant modes, then Just multiply  $D \cdot \text{eigenvector}(:, 1:m) \cdot \text{eigenvector}(:, 1:m)^T$ . Scaling not needed here since it essentially comes from the magnitude of the data matrix.



Profiles separated vertically by 2 m.



The EOF analysis tells us that the dominant mode (black in upper panel) is essentially the mean profile (green dots). Thus for this time period of we went out and guessed the profile, the mean probably makes the most sense. We would be wrong in an rms sense between 0.12 and 0.44 m.

The mean eigenfunction describes 99.93 percent of the variance while the second and third eigenfunctions describe 0.05 and 0.01 % respectively. Note that we only have about 9 months of data and thus the analysis works much better when we have covered many more months to include repeated variations between storm and swell profiles. Since the 2<sup>nd</sup> and 3<sup>rd</sup> eigenfunctions explain so little of the variance we could probably ignore them in future analysis of this short data set unless we were specifically interested in the small amount of variability in regard to them.

```

function [yloc,yeven,yodd,deltas]=even_odd2(loc,dy, center,testflag);
% function [yeven,yodd,deltas]=even_odd2(loc,dy, center,[testflag]);
% loc is vector of locations
% dy is vector of shoreline change (or volume change)
% assumes that center defines the central alongshore location of
% measurements (e.g. in case there are more locs updrift than downdrift of
% shoreline.
% testflag is for testing homework problems.
% testflag = 2 implies problem 2
% testflag = 3 implies problem 3

%outs
% yloc are the y locations corresponding to the even/odd functions
% about the central location
% yeven is the even function
% yodd is the odd function
% deltas are the dy from cent to make sure you did everything correct

%% find where the central location equals the location in the location
%% vector

if nargin == 4
    if testflag==2
        loc=[-600:200:600];
        dy=[0 10 20 -10 -30 -20 0];
        center=0;

    elseif testflag ==3
        loc=[1500:-100:-1600];
        dy=[0 -3 -6 -8 -10 -10.5 -11 -10 -8 -2 5 7.5 10 11 15 20 20.5 21 21.5 ...
            21 21 20.5 20 15 12 10.5 10 7 5 2 0 0];
        center =0;
        % note I added extra location to make sure code worked in finding
        % center
    else
        'wrong option for test flag'
        return
    end
end

p=find(loc==center);

% if no p found
if isempty(p)==1

```

```

'the central location does not match any location given'
return

else % make new vectors for loc and dy that are located about center

m=length(loc); % how many entries
lengthbefore=p-1; % how many points left of center
lengthafter=m-p; % how many points right of center.

halflength=min(lengthbefore,lengthafter); % so we know how many points either side to use

%redefine the locs and dy corresponding to central location
yloc=loc(p-halflength:p+halflength);
dy=dy(p-halflength:p+halflength);
end

% calculate even and odd functions
for k=1:length(yloc);
yeven(k)=(dy(k)+dy(end-k+1))/2;
yodd(k)=(dy(k)-dy(end-k+1))/2;
deltas(k)=yloc(k)-yloc(end-k+1);
end

% do some plotting (may want to do a set(gca,'xdir','reverse')
% depending on how the alongshore coord is defined
subplot(211)
plot(yloc,dy,'k','LineWidth',2),
xlabel('Alongshore distance')
ylabel('Volume Change, \Delta V (m^3/m)');
grid on

subplot(212)
plot(yloc,yeven,'k','LineWidth',2);
hold on
plot(yloc,yodd,'k--','LineWidth',2);
legend('\Delta V_{even}','\Delta V_{odd}')
xlabel('Alongshore distance')
ylabel('Even/Odd function, \Delta V (m^3/m)');
grid on

```

```

%% hw4_3 (NOT ASSIGNED)

if 0==1
% get the data first
x=75:5:1900;

l=dir('FRFBATHY\*.txt');
clear D

D(:,1)=x';
for k=1:length(l);
    fname=['FRFBATHY\' l(k).name'];
    A=load(fname);
    D(:,k+1)=interp1(A(:,1),A(:,3),x);
end

offset=-2;
for k=1:length(l);
    offset=offset+2;
    plot(x,D(:,k+1)+offset)
    hold on
end
xlabel('crossshore distance (m)');
ylabel('elevation (m)');

D=D(:,2:end);
D=D';
end % ignore since data given in the file

load hw4_duck2005bathy.mat % load the data
[m,n]=size(D);

[eigvals, eigvect, eigvecvar, variance]=EOF_datacorrmethod(D,0); %call the EOF function
given below

%% percent explained
pr=eigvals(1:5)/variance

%meanprof=D*eigvect(:,1)*eigvect(:,1)'; % the first one is the reconstruction using just one
eigenvector
meanprof=-eigvect(:,1)*sqrt(n*eigvals(1));
meanmean=mean(D);

clf
subplot(211)

```

```
plot(x,D(1,:),'m','linewidth',2);
hold on
plot(x,D(end,:),'b','linewidth',2);
plot(x,meanprof,'k','linewidth',2);
plot(x,meanmean,'g.','markersize',5);
axis([0 600 -8 8])
legend('first profile','last profile','mean eigenfunc', 'mean of profiles');
ylabel('Elevation (m)')
```

```
subplot(212)
plot(x,eigvect(:,2),'k','linewidth',2)
hold on
plot(x,eigvect(:,3),'b','linewidth',2)
axis([0 600 -.3 0.3])
legend('2nd eigenfunction','3rd eigenfunction')
xlabel('Cross-shore Distance (m)')
```

```
for k=1:7;
rmserr(k)=sqrt(mean ( (D(k,:)-meanprof).^2)); % rms error between EOF mean and each profile
end
```

```

function [eigvals, eigvect, eigvecvar, variance]=EOF_datacorrmethod(D,normflag);
%
%function [eigvals, eigvect, eigvecvar, variance]=EOF_datacorrmethod(D,[normflag]);
%input:
% D is the data matrix with time as rows and locations as columns
%   for beach profiles, each row would be a profile.
% normflag = 1 means remove mean and normalize by std
%   = 0 or left unassigned means don't do
%
%
%output:
%eigvals are the eigenvalues
%eigvec are the eigenvectors
%eigvecvar are the variation of the eigenvectors over time
%variance is the variance of the data (the sum of the eigenvalues)

nargin

if nargin==1
    normflag=0;
end

[m,n]=size(D); % size of matrix
normval=m*n; % normalization value, the IK

% first remove mean
if normflag==1
for k=1:m; % the number of data sets
    D(k,:)=(D(k,:)-mean(D(k,:)));
    D(k,:)=D(k,:)/std(D(k,:));
    %D(k,:)=detrend(D(k,:));
    'hi'
end
end % normflag loop

%% make data data correlation matrix
DD=(D'*D)/normval;

% find eigenvalues and eigenvectors
[eigvect,eigval]=eig(DD); % evecst are columns

[lambda,k]=sort(diag(-1*eigval)); % sort in descending order
lambda=-lambda; % get proper sign back on lambda after sorting
eigvals=lambda; % return the vector not matrix
variance = sum(lambda); % find variance
eigvect=eigvect(:,k); % sort eigvectors to correspond to eigenvalues

```

```
eigvecvar=D*eigvect; % these are the coefficients, and their temporal variation
```

```
%% make some plots
```

```
subplot(3,1,1)
col='kbrmgyc';
for k=1:m
    h1=plot(D(k,:));
    set(h1,'color',col(mod(k,7)+1))
    hold on
end
title('Profiles')
xlabel('measurment number')
```

```
subplot(3,1,2)
for k=1:min(m,6)
h1=plot(eigvect(:,k));
    set(h1,'color',col(mod(k,7)+1))
hold on
end
xlabel('measurment number')
title('1st 6 Eigenvectors')
```

```
subplot(3,1,3)
for k=1:min(m,6)
h1=plot(eigvecvar(:,k));
    set(h1,'color',col(mod(k,7)+1))
hold on
end
```

```
xlabel('time')
ylabel('variation');
title('Eigenvector variation');
```