Journal of Geophysical Research: Earth Surface

Supporting Information for

Salt marsh erosion rates and boundary features in Barnegat Bay, New Jersey, USA

Nicoletta Leonardi[1,3], Zafer Defne[2], Neil K. Ganju[2], Sergio Fagherazzi[3]

[1]Department of Geography and Planning, University of Liverpool, UK

[2]United States Geological Survey, Woods Hole MA, USA

[3]Department of Earth and Environment, Boston University, Boston , MA, USA

Contents of this file

Methods: Wave thrust subroutine

Results: (Figure S1) Hurricanes tracks interesting New Jersey from 2007 to 2013; Images of before and after Hurricane Sandy.

Introduction

We provide the code necessary to run the offline wave thrust calculations presented in the paper. The same code has been implemented into the COWAST modeling suite (http://woodshole.er.usgs.gov/operations/modeling/COAWST/).

We further provide information relevant to the occurrence of Hurricane in the area for the period in between 2007, and 2013 (Figure S1).

```
function [EFstar,WFstar,EF_TOT,WF_TOT,bou_wd_nan,bou_wd_msl_nan,
bou_wd_msl_nan2,lat_rho,lon_rho,thrust_w_noangle,...
thrust_w,thrust_w_msl,thrust_w_wet,thrust_w_noangle_msl,thrust_w_noangle_wet,mask_wd_tonelli,thrust_w_t
onelli,Pwave2,Hwave2] =...
risk (flag_mask_msl,start,nt,T_hours,grid_type,taucr,Pcr,percent,name_history_file,name_swan_file);
% LAST UPDATE = Fabruary 2_April2015
% EVALUATE MARSH THRUST, WAVE FACTOR AND EROSION FACTOR AT MARSH BOUNDARIES
% [For Erosion and Wave Factors see:
% Fagherazzi and Wiberg, 2009; Mariotti and Fagherazzi,
% 2010 "Influence of storm surges and sea level on shallow tidal basin
% erosive processes" (euqation 15 and 17);
% For wave Thrust see: Tonelli et al;, 2010
% "Modeling wave impact on salt marsh boundaries"].
% For wave thrust see also: "water waves mechanics for engineers and
% scientists" by Dean and Darlymple].
% Marsh thrust values are calculated with and without the correction from
% wave angle.
% For each cell if one side of the cell is sheletred from other
% cells, that side is not exposed to waves.
% Each cell has four normals directed towards the center of the cell.
% the angle of the normals is respect to the north and clockwise.
% You can apply the Tonelli mask to reduce the value of the wave thrust
% when marsh is submerged
%
% --------------------------------INPUT-----------------------------------
% 1) name_hystory_file = is the name of the ROMS history file. Insert it
% using ''. e.g [...]=risk ('ocean_his.nc','swan_ocean.nc',....
% 2) name_swan_file = is the name of the SWAN output file or roms input
% file if SWAN was runned in another simulation like in shoreface test case
%
% 3) flag_mask_msl==1 it means you want contour at mean sea level
% flag_mask_msl==0 it means you don't want contour at mean sea level
%
% 4)start= initial time step to do the anlysis
% 5)nt= number of time steps for the analysis
% 6) T_hours time during which you want to average the Erosion factor and
% the wave factor
% 7) grid_type; if equal to 1 it means that have latitude and longitude
% If equal to 0 it means I only have x and y
% 8) taucr, the critical shear stress for the marsh boundary
% 9) Pcr, the critical wave power marsh boundary
% 10) percent is the percentage of time to be inundated to be considered
% tidal flat. For example, In shoreface the wet dry mask is at the
% longitudinal grid point 45 for 101 time steps (for 120 total time step)
% While it is at the longitudianl grid point 44 for 20 time step. If
% percent=0. Points at longitudinal coordiante 44 are tidal flat. If
% percent=21 points at longitudinal coordiante 44 are NOT tidal flat.
% --------------------------------OUTPUT----------------------------------
% Asterisks mean that they are the main outputs.
%
% --EFstar is the erosion factor for each point of the boundary for each time
% step (EFstar=Area_cell.*(bstr-bucr))
%
% --WFstar is the wave factor for each point of the boundary for each time
% step (WFstar=Pwave.*length exposed site)
%
% **--EF_TOT is the erosion factor of the entire boundary averaged over a
% period of T_hours (T_hours defined in the input)
%
% **--WF_TOT is the wave factor of the entire boundary averaged over a
% period of T_hours (T_hours defined in the input)
%
% --bou_wd_nan is the wet contour for each time step. points equal to one
% are the last wet points
%
% --bou_wd_msl_nan is the wet contour at mean sea level. Is the contour for
% which the marsh thrust is evaluated with the tonelli
%
% --bou_wd_msl_nan2 is the wet contour and it is thick three cell instead
% fo one cell.
%
% --thrust_w is the map of the marsh thrust for the entire domain. you need
% to multiply this for the wet contour or for the contour at mean sea level,
% or for the contour which is three cell thick to have the thrust on the
% marsh. The results will be thrust_w_msl or thrust_w_wet.
%
```

```
% **--thrust_w_msl thrust along the contour at mean sea level for each time step.
% This imply that marshes elevation are at mean sea level. You can apply
% the tonelli mask to obtain the thrust reduction. This take into accout
% wave direction.
%
% **--thrust_w_wet thrust along the wet contour for each time step. The wet
% contour change with the time step and depends on the water level.
% You CAN NOT apply the Tonelli mask to this because Tonelli mask make
% sense if the contour is fixed in time and interested by different water
% level depths. This take into accoutn wave direction
%
% --thrust_w_noangle_msl thrust at msl without the angle correction
%
% --thrust_w_noangle_wet thrust wet contour without angle correction
%
% **--thrust_w_tonelli wave thrust along mean sea level with marsh thrust
% reduction in case of salt marsh submergence
%
% **--mask_wd_tonelli mask to apply to thrust_w_msl to obtain thrust_w_tonelli
%
% start=100;
% nt=10;
% T_hours=3;
% grid_type=1;
% taucr=0;
% Pcr=0;
% percent=0;
% flag_mask_msl=1;
%
name_history_file='http://geoport.whoi.edu/thredds/dodsC/clay/usgs/users/abeudin/chinco/tidewindwave/hi
s_v3.nc';
% name_history_file='last.nc';
%
% [EFstar,WFstar,EF_TOT,WF_TOT,bou_wd_nan,bou_wd_msl_nan,
bou_wd_msl_nan2,lat_rho,lon_rho,thrust_w_noangle,...
%
thrust_w,thrust_w_msl,thrust_w_wet,thrust_w_noangle_msl,thrust_w_noangle_wet,mask_wd_tonelli,thrust_w_t
onelli] =...
% risk (flag_mask_msl,start,nt,T_hours,grid_type,taucr,Pcr,percent,name_history_file);
%
% %-------------------------------------------------------------------------
% %-------------------------------------------------------------------------
% %-------------------------------------------------------------------------
% % EXPORT DATA FROM HISTORY FILE
% %-------------------------------------------------------------------------
warning off;
g=9.81;
T=T_hours.*60.*60;
nc=ncgeodataset(name_history_file);
hc=(nc{'hc'}(:));
t=(nc{'ocean_time'}(:)); % Time vector
dt =(t(2)-t(1)); % dt_hour = time inetrval between following hystory record
dt_hour = dt./60./60;
nn=round((T./dt)-1); % nn=number of interval to average
n_T=t(1:nn:end)'; % t evaluated at the boundary between subsequent time intervals
ntav=length(n_T); % number of intervals
if grid_type==0;
x_rho=squeeze(nc{'x_rho'});
number_cell=size(x_rho);
nx=number_cell(1); % nx = number of x_rho points
ny=number_cell(2); % ny = number of y_rho points
else
lon_rho=squeeze(nc{'lon_rho'}(:));
number_cell=size(lon_rho);
nx=number_cell(1); % nx = number of x_rho points
ny=number_cell(2); % ny = number of y_rho points
end
% %-------------------------------------------------------------------------
% % FIND THE WET-DRY BOUNDARY AT MEAN SEA LEVEL
% % (assume that marshes elevation is at mean sea level)
% %-------------------------------------------------------------------------
if flag_mask_msl==1
dt24=floor(12.*60.*60./dt); % number of time step to reach the first 24 hours
if nt>dt24; % If you are analyzing more than 24 hours use this to find the boundary
at msl
[bou_wd_msl,bou_wd_msl_nan,bou_wd_msl_nan2]=contour_msl(nc,dt24,nx,ny);
```

```
else % if you have less than 24 hours you cannot evaluate the boundary at
mean sea level
bou_wd_msl=nan; % give nan value because you cannot calculate them becuase you don't
have entire tidal cycle.
bou_wd_msl_nan=nan; % give nan value because you cannot calculate them
bou_wd_msl_nan2=nan; % give nan value because you cannot calculate them
end
else
bou_wd_msl=nan; % give nan value because the user don't want it
bou_wd_msl_nan=nan; % give nan value because the user don't want it
bou_wd_msl_nan2=nan; % give nan value because the user don't want it
end
% %--------------------------------------------------------------------------
% % INITIALIZE VARIABLES
% %--------------------------------------------------------------------------
thrust_w_msl=nan(nt,nx,ny); % initialize Marsh Thrust for the boundary at mean sea level
thrust_w_msl_tonelli=nan(nt,nx,ny); % initialize Marsh Thrust for the boundary at mean sea level
TONELLI(=REDUCTION WHEN CREST HIGHER THAN MARSH)
thrust_w=nan(nt,nx,ny); % initialize Marsh Thrust
bou_wd=zeros(nt,nx,ny); % Initialize the wet-dry boundary
mask_wd_tonelli=zeros(nt,nx,ny); % Mask Tonelli (to apply if I want to reduce the thrust when th marsh
is
submerged)
EFstar=zeros(nt,nx,ny); % Initialize shear stress factor every time step
WFstar=zeros(nt,nx,ny); % Initialize wave factor every time step
EF=zeros(ntav,nx,ny); % Initialize shear stress factor averaged every nn time steps
EF_TOT=zeros(ntav,1);
WF=zeros(ntav,nx,ny); % Initialize wave factor averaged every nn time steps
Atf=zeros(ntav,1);
bucr=taucr.*ones(nx,ny); % Initialize critical value of shear stress in the domain
dir_r=zeros(nx,ny); % Initialize direction of the right side of the cells
dir_l=zeros(nx,ny); % Initialize direction of the left side of the cells
dir_b=zeros(nx,ny); % Initialize direction of the bottom side of the cells
dir_u=zeros(nx,ny); % Initialize direction of the upper side of the cells
bou_wd_r=zeros(nx,ny); % Initialize areas of the wet-dry boundary with left neighbours (these
are used in the "find neighbours section)
bou_wd_l=zeros(nx,ny); % Initialize areas of the wet-dry boundary with right neighbours (these
are used in the "find neighbours section)
bou_wd_b=zeros(nx,ny); % Initialize areas of the wet-dry boundary with upper neighbours (these
are used in the "find neighbours section)
bou_wd_up=zeros(nx,ny); % Initialize areas of the wet-dry boundary with bottom neighbours (these
are used in the "find neighbours section)
gg=0; % Index for erosion and wave factor
% %--------------------------------------------------------------------------
% % DOWNLOAD THE DATA FROM NETCDF FILES
% %--------------------------------------------------------------------------
for n_t=1:nt;
ff=start+n_t;
if nargin>9 % The user input both a ocean histroy file and a swan file
[lat_rho,lat_u,lat_v,lon_rho,lon_u,lon_v,x_rho,y_rho,...
x_u,y_u,x_v,y_v,z,mask_wd,mask_wd_u,mask_wd_v,bustr_rho,bvstr_rho,Hwave,Pwave_top,Pwave_bot,Lwave,Dwave
,ndims_h,h,ro0]...
= download(grid_type,ff,name_history_file,name_swan_file);
else % The user only input a ocean histroy file and a swan file
[lat_rho,lat_u,lat_v,lon_rho,lon_u,lon_v,x_rho,y_rho,...
x_u,y_u,x_v,y_v,z,mask_wd,mask_wd_u,mask_wd_v,bustr_rho,bvstr_rho,Hwave,Pwave_top,Pwave_bot,Lwave,Dwave
,ndims_h,h,ro0]...
= download(grid_type,ff,name_history_file);
end
ff=n_t;
% %--------------------------------------------------------------------------
% % ALL DATA USED BELOW NEED TO BE DOWNLOADED BEFORE WITH THE THER SCRIPT
% %--------------------------------------------------------------------------
% % PRELIMINARY OPERATIONS WITH DATA
% %--------------------------------------------------------------------------
size_rho=size(x_rho); % Number of rho points
size_u=size(x_u); % Number of u points (note x_u has same dimension than y_u)
size_v=size(x_v); % Number of v points (note x_v has same diemanion than y_v)
x_rho_t(:,:)=x_rho; % rho points
y_rho_t(:,:)=y_rho;
lon_rho_t(:,:)=lon_rho;
lat_rho_t(:,:)=lat_rho;
lon_u_t(:,:)=lon_u;
lat_u_t(:,:)=lat_u;
lon_v_t(:,:)=lon_v;
lat_v_t(:,:)=lat_v;
```

```
depth_all=z+h; % Depth for each ocean time, for every rho point
depth=depth_all.*mask_wd; % Depth for each ocean time, only wet cells
l_x=abs([2*(x_u(:,1)-x_rho(:,1)),...
x_u(:,2:end)-x_u(:,1:end-1),... % Cells width along x axis. Centers are the rho points
2*(x_rho(:,end)-x_u(:,end))]); % For the boundary points the width is found considerin the half cell
going from rho to u.
l_y=abs([2*(y_v(1,:)-y_rho(1,:));... % Cells width along y axis. Centers are the rho points
y_v(2:end,:)-y_v(1:end-1,:);... % For the boundary points the width is found considerin the half cell
going from rho to u.
2*(y_rho(end,:)-y_v(end,:))]);
% %-------------------------------------------------------------------------
% % FIND THE WET POINTS OF BOUNDARIES AND TIDAL FLAT
% %-------------------------------------------------------------------------
% % method:
% 1 1 1 0 0 0 -
% 1 1 1 0 0 0
% _____________________________
% 0 0 1 0
%
% % THIS BOUNDARY IS DEFINED AT THE RHO POINTS:For this use the u and v
% % coordinate to find it.
% % The position of 1 is the last dry point (try plot mask_wd_u and
% % bou_wd_u1 for exmaple. I do this operation back and forth (using
% % bou_wd_u1/ bou_wd_u2 and summing them together.
% % I do the same for
% % bou_wd_v1/bou_wd_v2 using the mask_wd_v . I do it with the mask_wd_u and
% % mask_wd_v To get all the boundary points that I would not get because
% % mask_wd_u and mask_wd_v have differetn size than mask_wd_rho.
%%
% %-------------------------------------------------------------------------
% CALCULATE BOUNDARY (APRIL 8 2015)
%-------------------------------------------------------------------------
% This is for the vertical direction
%-------------------------------------------------------------------------
bou_wd1=zeros(nt,nx,ny);
bou_wd1(ff,1:nx-1,:)=(mask_wd(1:end-1,:)... % boundary wet and dry. Points equl to 1 are last wet
point; points
equalt to -1 are last dry point
-mask_wd(2:end,:));
bou_wd1((bou_wd1==-1))=0; % I exclude from the result the dry points because I only want the wet
ones
% bou_wd1((bou_wd1~=0))=1;% should not be necessary but retry
% Now I do the other direction because I don't know if it is 000111 or
% 111000
bou_wd2=zeros(nt,nx,ny);
bou_wd2(ff,2:nx,:)=(mask_wd(2:end,:)... % boundary wet and dry. Points equl to 1 are last wet point;
points
equalt to -1 are last dry point
-mask_wd(1:end-1,:));
bou_wd2((bou_wd2==-1))=0;
bou_wd2((bou_wd2~=0))=1;
bou_wdv=bou_wd1+bou_wd2;
bou_wdv(bou_wdv==-1)=0;
% bou_wdv(bou_wd~=0)=1;%should not be necessary but retry
% close all % do plot to test whta it did
% pcolor(squeeze(bou_wdv(ff,1:10,1:10)));
% figure
% pcolor(squeeze(mask_wd(1:10,1:10)));
%-------------------------------------------------------------------------
% This is for the horizontal direction
%-------------------------------------------------------------------------
bou_wd1=zeros(nt,nx,ny);
bou_wd1(ff,:,1:ny-1)=(mask_wd(:,1:end-1)... % boundary wet and dry. Points equl to 1 are last wet
point; points
equalt to -1 are last dry point
-mask_wd(:,2:end));
bou_wd1((bou_wd1==-1))=0; % I exclude from the result the dry points because I only want the wet
ones
% bou_wd1((bou_wd1~=0))=1; %should not be necessary but retry
% Now I do the other direction because I don't know if it is 000111 or
% 111000
bou_wd2=zeros(nt,nx,ny);
bou_wd2(ff,:,2:ny)=(mask_wd(:,2:end)... % boundary wet and dry. Points equl to 1 are last wet point;
points
equalt to -1 are last dry point
-mask_wd(:,1:end-1));
```

```
bou_wd2((bou_wd2==-1))=0;
bou_wd2((bou_wd2~=0))=1;
bou_wdh=bou_wd1+bou_wd2;
bou_wdh(bou_wdh==-1)=0;
% bou_wdh(bou_wd~=0)=1;%should not be necessary but retry
% close all % plot to see waht it did
% pcolor(squeeze(bou_wdh(ff,1:10,1:10)));
% figure
% pcolor(squeeze(mask_wd(1:10,1:10)));
%--------------------------------------------------------------------------
% calculate bou_wd_nan
%--------------------------------------------------------------------------
bou_wd=bou_wdv+bou_wdh;
bou_wd(bou_wd~=0)=1;
bou_wd_nan=bou_wd;
bou_wd_nan(bou_wd==0)=nan;
% % THIS IS THE TEST. TO work it should be all2
% test2=squeeze(bou_wd_nan(ff,:,:))+mask_wd;
% pcolor((test2));shading flat
% %--------------------------------------------------------------------------
% %--------------------------------------------------------------------------
% % FIND ANGLE
% %--------------------------------------------------------------------------
% % Find the angle respect to cartesian cordiante system.
% % Angles are measured clockwise from the north. Zero means North. 270
% % means west.
% % Find orthogonal direction to the 4 sides of each rho_point (i.e the
% % orthogonals are lines connecting different rho points)
% % |
% % x___v____x
% % -->| rho- |<-- rho
% % |x_______x|
% % ^
% % |
% % The angles of the orthogonal refer TO THE NORTH CLOCKWISE DIRECTION
% % the normal as the head toward the surface like in the sketch.
% % The angle is the angle orthogonal to the four sides. It differs from
% % the variable angle which is an average at the rho points.
% % I associate to each rho point 4 directions. For the boundary rho points
% % one of the directions is missed (because the grid ends and there are
% % not other rho points on one of the sides. So I associated to it a nan
% %--------------------------------------------------------------------------
% %--------------------------------------------------------------------------
if grid_type==1
dir_b(2:end-1,:)=dir_cell(lat_rho_t(2:end-1,:),lat_rho_t(1:end-2,:),...
lon_rho_t(2:end-1,:),lon_rho_t(1:end-2,:)); % Direction orthogonal bottom side rho point, using
Adjacent rho points
dir_u(2:end-1,:)=dir_cell(lat_rho_t(2:end-1,:),lat_rho_t(3:end,:),...
lon_rho_t(2:end-1,:),lon_rho_t(3:end,:)); % Direction orthogonal upper side rho point, using Adjacent
rho points
dir_l(:,2:end-1)=dir_cell(lat_rho_t(:,2:end-1),lat_rho_t(:,1:end-2),...
lon_rho_t(:,2:end-1),lon_rho_t(:,1:end-2)); % Direction orthogonal left side rho point, using Adjacent
rho points
dir_r(:,2:end-1)=dir_cell(lat_rho_t(:,2:end-1),lat_rho_t(:,3:end),...
lon_rho_t(:,2:end-1),lon_rho_t(:,3:end)); % Direction orthogonal right side rho point, using Adjacent
rho points
dir_r(:,1)=-dir_l(:,2); % It is equal to the opposite direction of the subsequent point. But the
normal point opposite so I adds ign minus
dir_r(:,end)=nan; % It does not exist because there are not rho points after it
dir_l(:,1)=nan; % It does not exist because therea are not rho points before it
dir_l(:,end)=-dir_r(:,end-1);
dir_u(1,:)=-dir_b(2,:);
dir_u(end,:)=nan;
dir_b(1,:)=nan; % It does not exist becasue there is not another rho points on the left
side
dir_b(end,:)=-dir_u(end-1,:);
else %lat is y and lon is x
dir_b(2:end-1,:)=dir_cell(y_rho_t(2:end-1,:),y_rho_t(1:end-2,:),...
x_rho_t(2:end-1,:),x_rho_t(1:end-2,:)); % Direction orthogonal bottom side rho point, using Adjacent
rho points
dir_u(2:end-1,:)=dir_cell(y_rho_t(2:end-1,:),y_rho_t(3:end,:),...
x_rho_t(2:end-1,:),x_rho_t(3:end,:)); % Direction orthogonal upper side rho point, using Adjacent rho
points
dir_l(:,2:end-1)=dir_cell(y_rho_t(:,2:end-1),y_rho_t(:,1:end-2),...
x_rho_t(:,2:end-1),x_rho_t(:,1:end-2)); % Direction orthogonal left side rho point, using Adjacent rho
points
```

```
dir_r(:,2:end-1)=dir_cell(y_rho_t(:,2:end-1),y_rho_t(:,3:end),...
x_rho_t(:,2:end-1),x_rho_t(:,3:end)); % Direction orthogonal right side rho point, using Adjacent rho
points
dir_r(:,1)=-dir_l(:,2); % It is equal to the opposite direction of the subsequent point. But the
normal point opposite so I adds ign minus
dir_r(:,end)=nan; % It does not exist because there are not rho points after it
dir_l(:,1)=nan; % It does not exist because therea are not rho points before it
dir_l(:,end)=-dir_r(:,end-1);
dir_u(1,:)=-dir_b(2,:);
dir_u(end,:)=nan;
dir_b(1,:)=nan; % It does not exist becasue there is not another rho points on the left
side
dir_b(end,:)=-dir_u(end-1,:);
end
% %--------------------------------------------------------------------------
% % FIND NEIGHBOURS
% %--------------------------------------------------------------------------
% % To find the neighbours:
% % Method: I have 4 possible directions from one boundary cell to the
% % others. For the boundary points equal to 1 are wet;
% % While points equal to 0 are dry.
% % I slide the wet dry mask on the right and I sum with the original mask.
% % I can the following results for each cell:
% % 2=the cell is wet and I have a wet neighbour on the left;
% % 1= the cell is dry and I have a wet neighbour on the left;
% % 1=The cell is wet and I have a dry neighbour on the left
% % For all the cases where I have a result equal to 1 I cannot
% % calculate the trhust at that point for a given direction either because
% % the point is dry (and It would be deleted in any case with the boundary
% % contour) or because the point is sheltered from that given direction.
% % Example:
% % Original matrix
% % ___|____|____|____|_____
% % ___|__1_|____|____|_____
% % ___|__1_|__1_|____|_____
% % ___|____|____|__1_|___1_
% % | | | |
% % I add to the original the following matrix (slided to the rigth)
% % ___|____|____|____|____|
% % ___|____|__1_|____|____|
% % ___|____|__1_|__1_|____|
% % ___|____|____|____|__1_|
% % | | | |
% % And I obtain:
% % ___|____|____|____|_____
% % ___|__1_|__1_|____|_____
% % ___|__1_|__2_|____|_____
% % ___|____|____|__1_|___2_
% % | | | |
% % I repeat the same operation for the four directions
% %--------------------------------------------------------------------------
% % An isolated cell can be: 1)a water cell with four dry neighbours or 2)
% % dry cell with 4 wet neighbours. in the second cell it is going to be
% % set to zero when multipliying fot eh wet contour.
% %--------------------------------------------------------------------------
% %--------------------------------------------------------------------------
Isolated_cell_l=zeros(nx,ny); % Initialize possible isolated water cell found sliding to the right
Isolated_cell_r=zeros(nx,ny); % Initialize possible isolated water cell found sliding to the left
Isolated_cell_up=zeros(nx,ny); % Initialize possible isolated water cell found sliding to the bottom
Isolated_cell_b=zeros(nx,ny); % Initialize possible isolated water cell found sliding up
bou_wd_l(:,2:end)=mask_wd(:,1:end-1); % Find neighbours on th right
sum_l=((mask_wd(:,:))+bou_wd_l); % Sum of original matrix and matrix slided to the right. The result
can
only be 1 or 2;
l_star=find(sum_l~=2); % if the sum is two it means it is a wet cell with a wet neighbour which
can be attacked by waves
Isolated_cell_l(l_star)=1; % Set to 1 the cells which could be Isolated
sum_l(l_star)=nan; % Set to nan values which CANNOT be attacked either becuase tehy are dry
cell or because they are sheltered
sum_l(sum_l==2)=1; % Set to 1 values which can be attacked
% Find neighbours on the left
bou_wd_r(:,1:end-1)=mask_wd(:,2:end);
sum_r=(mask_wd(:,:)+bou_wd_r);
r_star=find(sum_r~=2);
Isolated_cell_r(r_star)=1;
sum_r(r_star)=nan;
```

```
sum_r(sum_r==2)=1;
% Find neighbours at the bottom
bou_wd_up(2:end,:)=mask_wd(1:end-1,:);
sum_u=(mask_wd(:,:)+bou_wd_up);
u_star=find(sum_u~=2);
Isolated_cell_up(u_star)=1;
sum_u(u_star)=nan;
sum_u(sum_u==2)=1;
% Find neighburs at the top
bou_wd_b(1:end-1,:)=mask_wd(2:end,:);
sum_b=(mask_wd(:,:)+bou_wd_b);
b_star=find(sum_b~=2);
Isolated_cell_b(b_star)=1;
sum_b(b_star)=nan;
sum_b(sum_b==2)=1;
% Find the Isolated cell
Isolated_cell=Isolated_cell_r+Isolated_cell_l+...
Isolated_cell_b+Isolated_cell_up;
Isolated_cell(Isolated_cell~=4)=nan; % This is a necessary (not sufficient0 condition to be an isolated
cell
Isolated_cell(Isolated_cell==4)=1; % I set possible isolated cell equl to 1
Isolated_cell=Isolated_cell+mask_wd;
Isolated_cell(Isolated_cell~=2)=0; % Where this sum is zero it means that the cell is both Isolated and
wet
Isolated_cell(Isolated_cell==2)=1;
toadd_isolated=find(Isolated_cell(:,:)==1); % Points where thrust need to be evaluated because they are
isolated
cells
bou_wd(ff,toadd_isolated)=1; % Add isolated points to the boundary points
sum_r(toadd_isolated)=1; % Add isolated points to the wet right neighbours
sum_l(toadd_isolated)=1; % Add isolated points to the wet left neighbours
sum_u(toadd_isolated)=1; % Add isolated points to the wet up neighbours
sum_b(toadd_isolated)=1; % Add isolated points tothe wet bottom neighbours
% Note: sum_r,_l,_u,_b have values 1 or 0; dir_r,...is the angle in deg
exp_r=dir_r.*sum_r.*squeeze(bou_wd(ff,:,:)); % Only maintain the direction of the RIGHT sides of the
rho points where
there ae NOT neighbours
exp_l=dir_l.*sum_l.*squeeze(bou_wd(ff,:,:)); % Only maintain the direction of the LEFT sides of the rho
points where
there are NOT neighbours
exp_u=dir_u.*sum_u.*squeeze(bou_wd(ff,:,:)); % Only maintain the direction of the UPPER sides of the
rho points where
there are NOT neighbours
exp_b=dir_b.*sum_b.*squeeze(bou_wd(ff,:,:)); % Only maintain the direction of the BOTTOM sides of the
rho points
where there are NOT neighbours
% %-------------------------------------------------------------------------
% % CALCULATE THE MARSH THRUST
% %-------------------------------------------------------------------------
% % also see water waves mechanics for engineering and scientists
% % Dean Darlymple, pg 84 for formula. pg 77 for sketch of the variables.
% % Thrust=inetgral in z of the dynamic pressure;
% % Dynamic pressure = ro.*g.*max(z_thrust).*Kp(z_thrust);
% %
% % Kp=pressure response factor:
% % Kp(z_thrust)=(cosh(kw.*(h+z_thrust))./(cosh(kw.*h))
% %
% % The integral of the pressure response factor in z_thrust:
% % Integral_Kp=sinh(kw*(h + z_thrust))/(kw*cosh(h*kw))
% %
% % z_thrust is the vertical coordinate going from -h to + z
% %(roms variable). z_thrust is thus 0 at msl, positive above msl and
% % negative below msl. below I directly use the expression of the integral
% %-------------------------------------------------------------------------
% %-------------------------------------------------------------------------
% % THRUST WITH ANGLE CORRECTION
% % (270 deg)--> ------- <--(90 deg)
% % ^
% % |
% % (0 deg)
% % If the absolute value of the difference between wave direction and
% % normal directed inside the cell is between 90 and 270 deg it means that
% % the flow is blowing opposite direction
% %-------------------------------------------------------------------------
kw=2*pi./Lwave; % wave number
```

```
Integral_Kp=sinh(kw.*(h+0))./(kw.*cosh(h.*kw)); % Integral of Pressure response factor at msl; above
msl I use
hydostatic approximation and I don't need Kp
% F_pd_h=sinh(kw.*(h-h))./(kw.*cosh(h.*kw)); % This is always zero
Fw1=ro0.*9.81.*Hwave.*Integral_Kp./1000; % Dynamic pressure below m.s.l (divide 1000 to have KN); roms
variable z is max(z_thrust)
Fw2=(ro0*9.81.*Hwave).*Hwave./2./1000; % Idrostatic approximation above m.s.l (divide 1000 to have KN);
Fw=Fw1+Fw2; % Total Wave thrust above mean sea lveel
thrust_r=abs(Fw.*cosd(exp_r-Dwave));
thrust_r(abs(exp_r-Dwave)>=90 & abs(exp_r-Dwave)<=270)=0; % If the angle is higher than 90 and smaller
than 270 it means that that
waves are
thrust_r(isnan(thrust_r))=0; % arriving opposite direction respect to the cell side(impact other
side, e.g impact right
% and not the left side of the cell.
thrust_l=abs(Fw.*cosd(exp_l-Dwave));
thrust_l(abs(exp_l-Dwave)>=90 & abs(exp_l-Dwave)<=270)=0;
thrust_l(isnan(thrust_l))=0;
thrust_b=abs(Fw.*cosd(exp_b-Dwave));
thrust_b(abs(exp_b-Dwave)>=90 & abs(exp_b-Dwave)<=270)=0;
thrust_b(isnan(thrust_b))=0;
thrust_u=abs(Fw.*cosd(exp_u-Dwave));
thrust_u(abs(exp_u-Dwave)>=90 & abs(exp_u-Dwave)<=270)=0;
thrust_u(isnan(thrust_u))=0;
% %-------------------------------------------------------------------------
% % THRUST WITH NO ANGLE CORRECTION
% %-------------------------------------------------------------------------
thrust_r_noangle=Fw.*sum_r;
thrust_r_noangle(isnan(thrust_r_noangle))=0;
thrust_l_noangle=Fw.*sum_l;
thrust_l_noangle(isnan(thrust_l_noangle))=0;
thrust_b_noangle=Fw.*sum_b;
thrust_b_noangle(isnan(thrust_b_noangle))=0;
thrust_u_noangle=Fw.*sum_u;
thrust_u_noangle(isnan(thrust_u_noangle))=0;
% %-------------------------------------------------------------------------
% thrust_w(ff,:,:)=(sqrt(thrust_r.^2+thrust_l.^2+thrust_u.^2+thrust_b.^2)); % Thrust everywhere
thrust_w(ff,:,:)=((thrust_r+thrust_l+thrust_u+thrust_b)); % Thrust everywhere
% thrust_w_noangle(ff,:,:)=(sqrt(thrust_r_noangle.^2+thrust_l_noangle.^2+... % Thrust everywhere
without angle correction
% thrust_u_noangle.^2+thrust_b_noangle.^2));
thrust_w_noangle(ff,:,:)=((thrust_r_noangle+thrust_l_noangle+... % Thrust everywhere without angle
correction
thrust_u_noangle+thrust_b_noangle));
thrust_w_msl(ff,:,:)=bou_wd_msl_nan.*...
squeeze(thrust_w(ff,:,:)); % Thrust MEAN SEA LEVEL
thrust_w_wet(ff,:,:)=squeeze(bou_wd_nan(ff,:,:)).*... % Thrust along the WET CONTOUR
squeeze(thrust_w(ff,:,:));
thrust_w_noangle_msl(ff,:,:)=bou_wd_msl_nan.*... % Thrust MEAN SEA LEVEL without angle correction
squeeze(thrust_w_noangle(ff,:,:));
thrust_w_noangle_wet(ff,:,:)=squeeze(bou_wd_nan(ff,:,:)).*... % Thrust along the WET CONTOUR without
angle correction
squeeze(thrust_w_noangle(ff,:,:));
mask_wd_tonelli1=mask_wd;
%note:I INITIally used hc but then tought that it is better to use 0 in any case so you don't
%care about hc value used by user
% mask_wd_tonelli1(depth_all>hc & depth_all<0.2)=...
% %%%%---INITIAL WAVE THRUST REDUCTION %%%%
% mask_wd_tonelli1(depth_all>0 & depth_all<0.2)=...
% 1-0.45.*(depth_all(depth_all>0 & depth_all<0.2)./0.2); % Reduce up to 45% the thrust when 0.1< depth
<0.2
% mask_wd_tonelli1(depth_all>0.2 & depth_all<0.4)=...
% 0.55-0.55.*2.5.*(depth_all(depth_all>0.2 & depth_all<0.4)-0.2); % Reduce up to the 0.275% the thrust
when 0.2<depth<0.4
% mask_wd_tonelli1(depth_all>0.4)=0.275; % Constant reduction whent he depth is higher than 0.4
% mask_wd_tonelli(ff,:,:)=mask_wd_tonelli1; % Tonelli mask to use to reduce thrust when submerged
%
% thrust_w_tonelli(ff,:,:)=bou_wd_msl_nan.*... % Thrust along the WET CONTOUR without angle correction
% squeeze(thrust_w(ff,:,:)).*...
% mask_wd_tonelli1;
%
% thrust_w_tonelli(ff,:,:)=squeeze(bou_wd_nan(ff,:,:)).*... % Thrust along the WET CONTOUR without
angle correction
% squeeze(thrust_w(ff,:,:)).*...
% mask_wd_tonelli1;
%%%%%%---NEW REDUCTION %%%%
```

```
mask_wd_tonelli1(depth_all>0 & depth_all<0.4)=...
1-0.6.*(depth_all(depth_all>0 & depth_all<0.4)./0.4);
thrust_w_tonelli(ff,:,:)=bou_wd_msl_nan.*... % Thrust along the WET CONTOUR without angle correction
squeeze(thrust_w(ff,:,:)).*...
mask_wd_tonelli1;
thrust_w_tonelli(ff,:,:)=squeeze(bou_wd_nan(ff,:,:)).*... % Thrust along the WET CONTOUR without angle
correction
squeeze(thrust_w(ff,:,:)).*...
mask_wd_tonelli1;
% %-----------------------------------------------------------------------
% % CALCULATE THE RISK PARAMETER
% %-----------------------------------------------------------------------
% %-----------------------------------------------------------------------
% % EROSION FACTOR AND WAVE FACTOR
% %-----------------------------------------------------------------------
bstr=sqrt(bustr_rho.^2+bvstr_rho.^2).*cosd(exp_r-Dwave); % Shear stress asscoiated to each point
Area_cell=sqrt(l_x.*l_y); % Area associated to each rho point
% %-------------------------------------------------------------------------
% %CALCULATE THE WAVE POWER
% %-------------------------------------------------------------------------
wavenumber_k=2.*pi./(Lwave);
angular_frequency_sigma=sqrt(g.*wavenumber_k.*tanh(wavenumber_k.*depth));
group_celerity=0.5*(angular_frequency_sigma./wavenumber_k).*(1+(2.*wavenumber_k.*depth)./(sinh(2.*waven
umber_k.*depth)));
Pwave=group_celerity.*ro0.*9.81.*Hwave.^2./8;
Pwave2(ff,:,:)=Pwave;
Hwave2(ff,:,:)=Hwave;
l_r=(squeeze(bou_wd(ff,:,:)).*sum_r.*(l_y)); l_r(isnan(l_r)==1)=0;
l_l=squeeze(bou_wd(ff,:,:)).*sum_l.*(l_y); l_l(isnan(l_l)==1)=0;
l_u=squeeze(bou_wd(ff,:,:)).*sum_b.*(l_x); l_u(isnan(l_u)==1)=0;
l_b=squeeze(bou_wd(ff,:,:)).*sum_u.*(l_x); l_b(isnan(l_b)==1)=0;
ltot=l_r+l_l+l_u+l_b; % length of the areas which are NOT sheltered by other cells
EFstar(ff,:,:)=(Area_cell.*(abs(bstr)-bucr)); %******REMEMBER, YOU HAVE DONE THIS WITH TOTAL SHEAR
STRESS YOU CAN
SEPARATE CURRENT AND WAVE SHEAR STRESS( SEE OUTPUTS SWAN BUSTR_W BUSTR_C)
WFstar(ff,:,:)=((Pwave-Pcr).*ltot);
% %%%%%%%%%%%%%%%%%%%%%%%%
% if (n_T(gg+1)-(dt+dt.*ff))==0;
% % % If the time step is multiple of the time you want to average the Erosion/Wave factor
%
% % rem(dt_hour.*ff,T_hours)<=dt_hour;
%
% gg=gg+1;
% tf=squeeze(sum(bou_wd(ff-nn+1:ff,:,:),1)); % Sum the wet points at the boundary between different
times
% % Points different from zero are the wet boundary points for at leats
one time step
% tf(tf > percent)=1; % The defaul value fro percent is zero. meaning thet if one point is the
wet boundary for % at least one simulation time step It is part of the tidal flat
% tf(tf < percent)=0; % Don't consider tidal flat points below the percentage
% Atf(gg)=sum(sum(sum((l_x.*l_y).*tf))); % Tidal flat area is the sum of each cell area creating the
tidal flat.
%
% % EF(gg,:,:)=((squeeze(sum(EFstar(ff-nn:ff,:,:),1))./(T)));
% % WF(gg,:,:)=squeeze(sum(WFstar(ff-nn:ff,:,:),1))./(T);
%
% %i think I dont need to divide by T but by the number of layers that i
% %sum
% EF(gg,:,:)=((squeeze(sum(EFstar(ff-nn+1:ff,:,:),1))./(nn)));
% WF(gg,:,:)=squeeze(sum(WFstar(ff-nn+1:ff,:,:),1))./(nn);
% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%
% EF_tosum=EF(gg,:,:); % Create new variable for the erosion factor to delete NaN and set
them equal to zero
% EF_tosum(isnan(EF_tosum)==1)=0;
% WF_tosum=WF(gg,:,:);
% WF_tosum(isnan(WF_tosum)==1)=0; % Create new variable for the wave factor to delete NaN and set them
equal to zero
%
% EF_TOT(gg)=sum(EF_tosum(:))./Atf(gg);
% WF_TOT(gg)=sum(WF_tosum(:))./(sum(ltot(:)));
%
% end
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end
hh=0;
```

```
for d=nn:nn:nt
hh=hh+1;
tf=squeeze(sum(bou_wd(d-nn+1:d,:,:),1)); % Sum the wet points at the boundary between different times
% Points different from zero are the wet boundary points for at leats
one time step
tf(tf > percent)=1; % The defaul value fro percent is zero. meaning thet if one point is the
wet boundary for % at least one simulation time step It is part of the tidal flat
tf(tf < percent)=0; % Don't consider tidal flat points below the percentage
Atf(hh)=sum(sum(sum((l_x.*l_y).*tf))); % Tidal flat area is the sum of each cell area creating the
tidal flat.
EF(hh,:,:)=((squeeze(sum(EFstar(d-nn+1:d,:,:),1))./(nn)));
WF(hh,:,:)=squeeze(sum(WFstar(d-nn+1:d,:,:),1))./(nn);
EF_tosum=EF(hh,:,:); % Create new variable for the erosion factor to delete NaN and set them
equal to zero
EF_tosum(isnan(EF_tosum)==1)=0;
WF_tosum=WF(hh,:,:);
WF_tosum(isnan(WF_tosum)==1)=0; % Create new variable for the wave factor to delete NaN and set them
equal to zero
EF_TOT(hh)=sum(EF_tosum(:))./Atf(hh);
WF_TOT(hh)=sum(WF_tosum(:))./(sum(ltot(:)));
end
```

```
------bin data-----
all1=(er);
all2=(thrust_mean_tonelli_all_new_isolated(dd30)');
latitude1=(coordinate(d_isolated(dd30),1));
longitude1=(coordinate(d_isolated(dd30),2));
all_12=[all1;all2;latitude1';longitude1']';
ap=find(latitude1<=39.54);
bp=find(longitude1<=-74.38);
as=find(latitude1>=39.7);
bs=find(longitude1>=-74.15);
cp=intersect(ap,bp);
cs=intersect(as,bs);
c=[1:length(all1)]';
for i=1:length(cs)
    ss(i)=find(c==cs(i));
end
c(ss)=[];
for i=1:length(cp)
    pp(i)=find(c==cp(i));
end
c(pp)=[];
all_12=[all1(c);all2(c);latitude1(c)';longitude1(c)']';
all12=sortrows(all_12,1);
latitude=all12(:,3);longitude=all12(:,4);
all12=all12(:,1:2);
% a=find(isnan(all12)==1)
% all12(a,:)=[];
maxi=nanmax(all12(:,1));
mini=nanmin(all12(:,1));
bins =linspace(mini, maxi, numbins);
% bins1=exp(linspace(log(0.00001),log(treshold),n));
% bins2 =linspace(treshold, maxi,numbins);
% bins=[bins1 bins2];
nn=length(all1)./(length(bins));
marker_index=[];
clear mu stdi mux stdi2 stdi1 marker_index
marker_index=[];
for k = 2:numbins-1
  ind = find(all12(:,1)<=bins(k) & all12(:,1)>bins(k-1));
  if (~isempty(ind))==1
    mu(k) = nanmean(all12(ind,1));
    stdi(k)=nanstd(all12(ind,1));
    mux(k)=nanmean(all12(ind,2));
    stdi2(k)=nanstd(all12(ind,2));
    lat(k)=nanmean(latitude(ind));
    latstd(k)=std(latitude(ind));
    long(k)=nanmean(longitude(ind));
    longstd(k)=std(longitude(ind));
  else
    mu(k) = nan;
    stdi(k)=nan;
    mux(k)=nan;
    stdi2(k)=nanstd(all12(ind,2));
  end
end
for k = 2:numbins-1
    if mux(k)>mux(k-1)
  ind = find(all12(:,2)<=mux(k) & all12(:,2)>=mux(k-1));
    elseif mux(k)<mux(k-1)
       ind = find(all12(:,2)<=mux(k-1) & all12(:,2)>=mux(k));
    end
  if (~isempty(ind));
    stdi1(k)=nanstd(all12(ind,1));
  else
     stdi1(k)=nan;
  end
end
mm=[mux;mu;stdi1;stdi2]';
mm(mm(:,1)==0,:)=[];
mm(mm(:,2)==0,:)=[];
```

```
mm(mm(:,2)<0.08)=nan;
figure
hold on
stem(mm(:,1),mm(:,2)+mm(:,3),'c','MarkerSize',2);
stem(mm(:,1),mm(:,2)-mm(:,3),'w','MarkerSize',1);
plot(mm(:,1),mm(:,2)-mm(:,3),'co','MarkerSize',2);
plot(mm(:,1),mm(:,2),'ro','MarkerFaceColor','r');
excel=[mm(:,1),mm(:,2)]
excel(isnan(mm(:,1))==1 | isnan(mm(:,2))==1,:)=[];
```

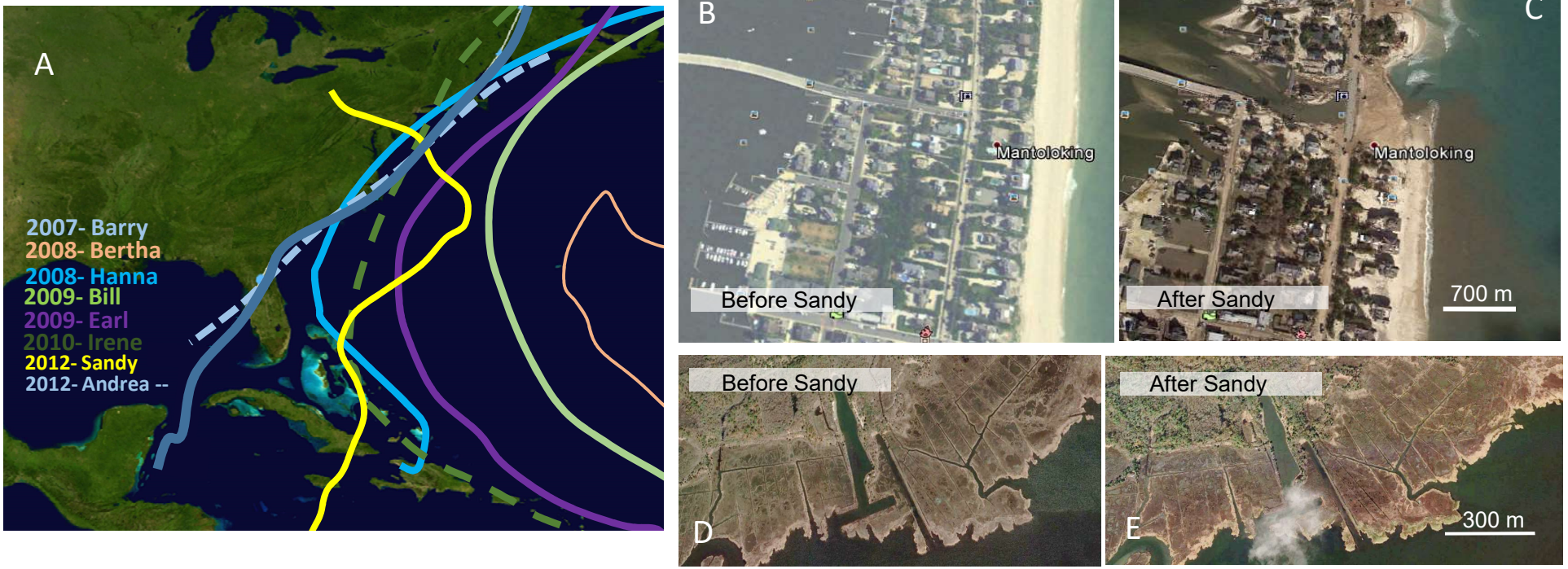
A
2007- Barry
2008- Bertha
2008- Hanna
2009- Bill
2009- Earl
2010- Irene
2012- Sandy
2012- Andrea --
B
Mantoloking
Before Sandy
C
Mantoloking
After Sandy
700 m
D
Before Sandy
E
After Sandy
300 m

Figure S1 A) Path of the major Hurricanes interesting New Jersey for the period from 2007 to 2013 (Data from National Hurricane Centre, http://www.nhc.noaa.gov/).

Of these, only Irene (2010), and Sandy (2012) made landfall in New Jersey ad caused Hurricane force winds in the area.

Sandy has been classified as the most destructive in the history of New Jersey [Blake et al., 2012; National Hurricane Center; Brandon et al., 2014].

B) Before and after Hurricane Sandy images for a sandy beach in Barnegat Bay (40° 2'25.99"N, 74° 2'55.38"W). Significant erosion is noticeable for this beach. Courtesy of Google Earth.

C) Before and after Hurricane Sandy images (40° 1'52.90"N, 74° 4'13.17"W) around 2 km distance from previous point. No significant erosion is noticeable for this salt marsh. Courtesy of Google Earth.