```
data = importdata('channel_data.txt');

% Conversion constants:

miles2km = 1.609344;

% Ref. for long2miles and lat2miles:
% https://www.usgs.gov/faqs/how-much-distance-does-degree-minute-and-second-cover-↙
your-maps

long2miles = 54.6;
lat2miles = 69;

% Extracting components of data.

xdata = data(:,1);
ydata = data(:,2);
height = data(:,3);

%%

% Computing differences in x- and y-direction in km.

xdif = (xdata(1:end-1)-xdata(2:end)).*lat2miles*miles2km;
ydif = (ydata(1:end-1)-ydata(2:end)).*long2miles*miles2km;

% Computing distance from mediterrenean sea.

dist = zeros(1,length(xdata))';

for i = 2:length(xdata)
    dist(i) = dist(i-1)+sqrt(xdif(i-1).^2+ydif(i-1).^2);
end

%%

% Linear spacing from 0 to end of river with spacing of 250 meters.

numPoints = floor(dist(end)/0.25)+2;

% Mutl. by 10 to round to nearest 100th meter.

d = linspace(0,floor(dist(end)*10),numPoints-1)'./10;

%%

% Doing linear interpolation of height.

interHeight = interp1(dist,height,d);

%%

figure(1)
plot(d,interHeight);

%%

% Exporting for Julia.

csvwrite('interHeight.csv',interHeight);

%%
```

```
% Importing from Julia.

xValues = table2array(readtable("xValues.csv"));
dirtRemoved = table2array(readtable("RValues.csv"));

%%

%Takes the hight from the start and removes the dirt

newInterHeight = interHeight-dirtRemoved;

%%

figure(2)
title('Minimizing number of bombs');
plot(d,interHeight,'b');
hold on
plot(d,newInterHeight,'r');
hold all
for i = 1:length(xValues)
    if xValues(i) == 1
        xline((i-1)*0.25);
    end
end
legend('Height before bombing','Height after bombing','Placement of bombs');
xlabel('Distance from sea (km)');
ylabel('Height from sea level (m)');

%%

% Plotting the new height map with bombs.

% figure(3)
% plot(d(1:end),newInterHeight);
% ylim([-300 0]);

% hold on
% for i = 1:length(objectiveFunction)
%     if objectiveFunction(i) == 1
%         xline(i*0.25);
%     end
% end

%%

% Importing results from non-linear objective function.

xValuesNonLinear = table2array(readtable("xValuesNonLinear.csv"));
dirtRemovedNonLinear = table2array(readtable("RValuesNonLinear.csv"));

newInterHeightNonLinear = interHeight-dirtRemovedNonLinear;

%%

% Plotting the new height map with bombs.

figure(4)
title('Maximizing smoothness');
plot(d,interHeight,'b');
hold on
plot(d,newInterHeightNonLinear,'r');
```

```
hold all
for i = 1:length(xValuesNonLinear)
    if xValuesNonLinear(i) == 1
        xline((i-1)*0.25);
    end
end
legend('Height before bombing','Height after bombing','Placement of bombs');
xlabel('Distance from sea (km)');
ylabel('Height from sea level (m)');

%% 5

% Implementing a constraint in Julia

xValuesNonLinearNoNeighbors = table2array(readtable("xValuesNonLinearNoNeighbors.↙
csv"));
dirtRemovedNonLinearNoNeighbors = table2array(readtable("RValuesNonLinearNoNeighbors.↙
csv"));

newInterHeightNonLinearNoNeighbors = interHeight-dirtRemovedNonLinearNoNeighbors;

%%

figure(5)
plot(d,interHeight,'b');
title('Maximizing smoothness without neighbouring bombs');
hold on
plot(d,newInterHeightNonLinearNoNeighbors,'r');
hold all
for i = 1:length(xValuesNonLinearNoNeighbors)
    if xValuesNonLinearNoNeighbors(i) == 1
        xline((i-1)*0.25);
    end
end
legend('Height before bombing','Height after bombing','Placement of bombs');
xlabel('Distance from sea (km)');
ylabel('Height from sea level (m)');

%%

xBombs = table2array(readtable("xValuesNonLinearExtended.csv"));
yBombs = table2array(readtable("yValuesNonLinearExtended.csv"));
% zBombs = table2array(readtable("zValuesNonLinearExtended.csv"));

dirtRemovedNonLinearExtended = table2array(readtable("RValuesNonLinearExtended.csv"));

newInterHeightNonLinearExtended = interHeight-dirtRemovedNonLinearExtended;

%%

figure(6)
plot(d,interHeight,'b');
hold on
plot(d,newInterHeightNonLinearExtended,'r');
for i = 1:length(newInterHeightNonLinearExtended)
    if xBombs(i) == 1
        hold on
        xline((i-1)*0.25,'g');
    end
    if yBombs(i) == 1
        hold on
        xline((i-1)*0.25,'m');
```

```
    end
end

legend('Height before bombing','Height after bombing','Placement of bombs');
legend('Height before bombing','Height after bombing','Setting 1 bomb','Setting 2↙
bomb');

xlabel('Distance from sea (km)');
ylabel('Height from sea level (m)');

%% Plots together

figure(7)
p1 = plot(d,newInterHeight,'g','LineWidth',4);
hold on
p2 = plot(d,newInterHeightNonLinear,'c','LineWidth',3);
hold on
p3 = plot(d,newInterHeightNonLinearNoNeighbors,'r','LineWidth',2);
hold on
p4 = plot(d,newInterHeightNonLinearExtended,'b','LineWidth',1);
legend("Minimizing bombs","Maximizing smoothness","Maximizing smoothness without↙
neighbors","Maximizing smoothness with yield dilation")
xlabel('Distance from sea (km)');
ylabel('Height from sea level (m)');
% p1.Color(4) = 0.6;
% p2.Color(4) = 0.6;
% p3.Color(4) = 0.6;
% p4.Color(4) = 0.6;

%% Number of bombs

Minbombs=sum(xValues);
SmoothingNumberbombs=sum(xValuesNonLinear);
NoneigboorNumberbombs=sum(xValuesNonLinearNoNeighbors);
ExtendedNumberBombs=nnz(xBombs)+nnz(yBombs);

table(Minbombs,SmoothingNumberbombs,NoneigboorNumberbombs,ExtendedNumberBombs)

distancebetweenbombs=xValuesNonLinearNoNeighbors-xValuesNonLinear;

%%

table(std(newInterHeight),std(newInterHeightNonLinear),std↙
(newInterHeightNonLinearNoNeighbors),std(newInterHeightNonLinearExtended))

%%

objFuncMod1 = dirtRemoved - interHeight - 10;
objFuncMod2 = dirtRemovedNonLinear - interHeight - 10;
objFuncMod3 = dirtRemovedNonLinearNoNeighbors - interHeight - 10;
objFuncMod4 = dirtRemovedNonLinearExtended - interHeight - 10;

%%

table(sum(objFuncMod1),sum(objFuncMod2),sum(objFuncMod3),sum(objFuncMod4))

%%

figure(8)
plot(d,dirtRemovedNonLinear,d,dirtRemovedNonLinearNoNeighbors);
```