Contents

- BE700 HW1 Prob 1
- Part 1
- Part 2
- Part 3
- Part 4
- Echoing final values

BE700 HW1 Prob 1

```
clear all
close all
warning('off', 'all') %warnings got annoying
tic
```

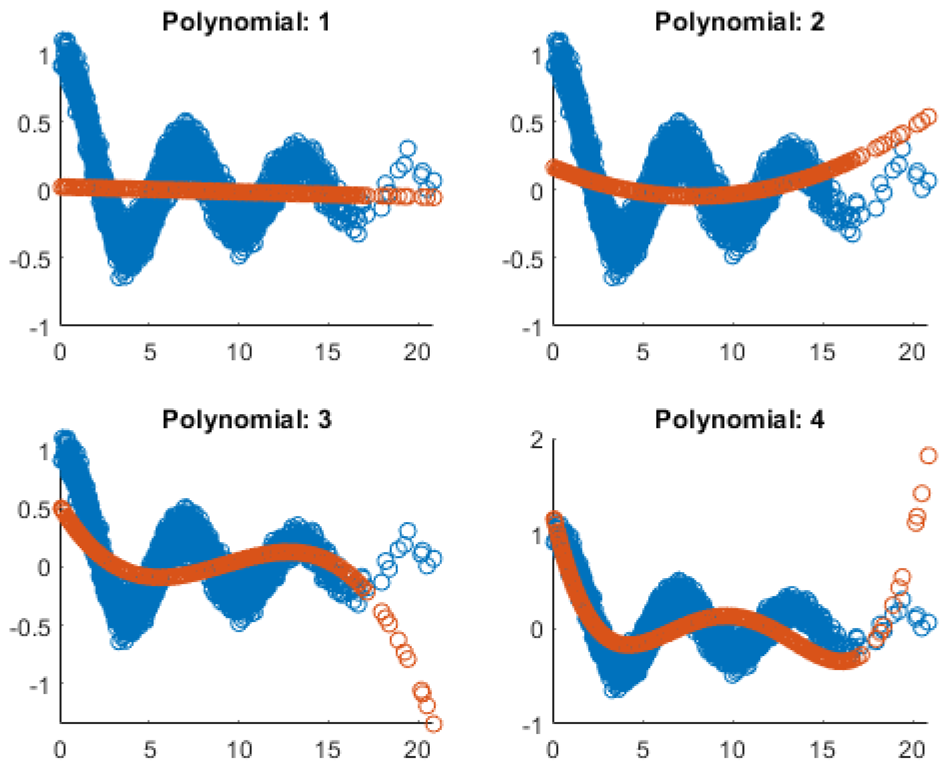
Part 1

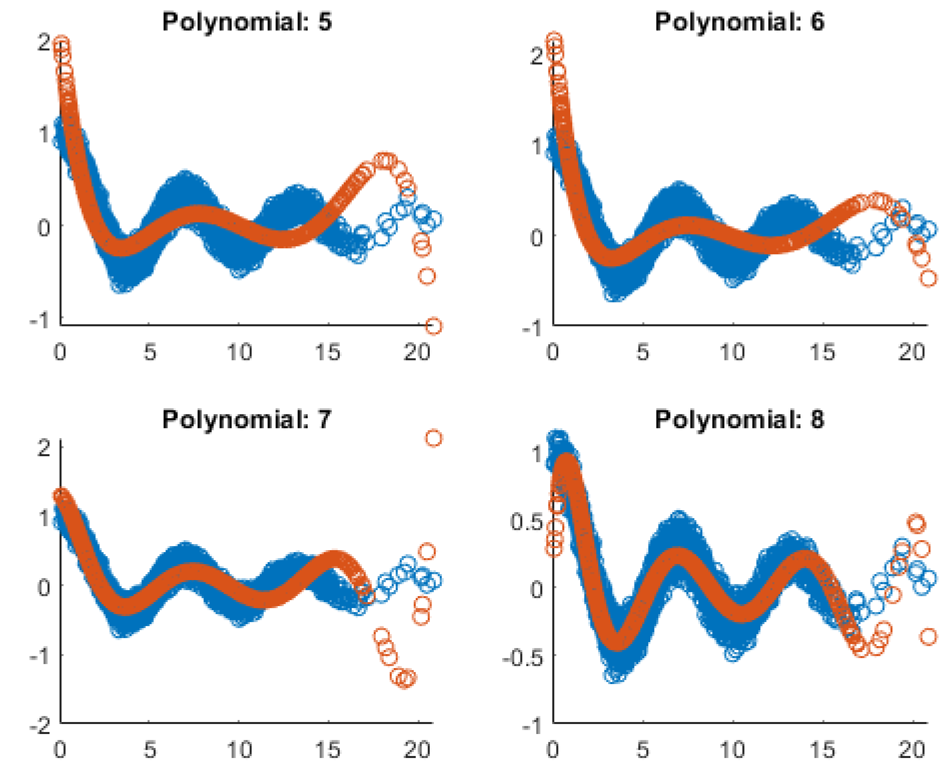
```
[x1,x2,y] = textread('Problem1_BesselData.txt', '%f%f%f', 'headerlines',1);

R = (x1.^2 + x2.^2).^0.5;

% scatter(R,y)

for P=1:14
    A= ones(length(R),P+1);
    for n1 = 1:P
        A(:,n1+1) = R.^n1;
    end
    B_hat.temp = (A' * A) \ (A' * y);
    B_hat.i{P} = fliplr(B_hat.temp');
end


yfit = ones(1,5000);
R2 = R';

for P=1:14
    yfit(P,:) = polyval(cell2mat(B_hat.i(P)), R2);
end

%resids

for P=1:14
    r(:,P) = y - yfit(P,:)';
    r2_sum(:,P) = sum(r(:,P).^2);
end

tfields = {'Polynomial order (p)'
                   '||r||^2 for a regular least-squares fit'
                   };

po = 1:1:14;
r2sumt = r2_sum';
```

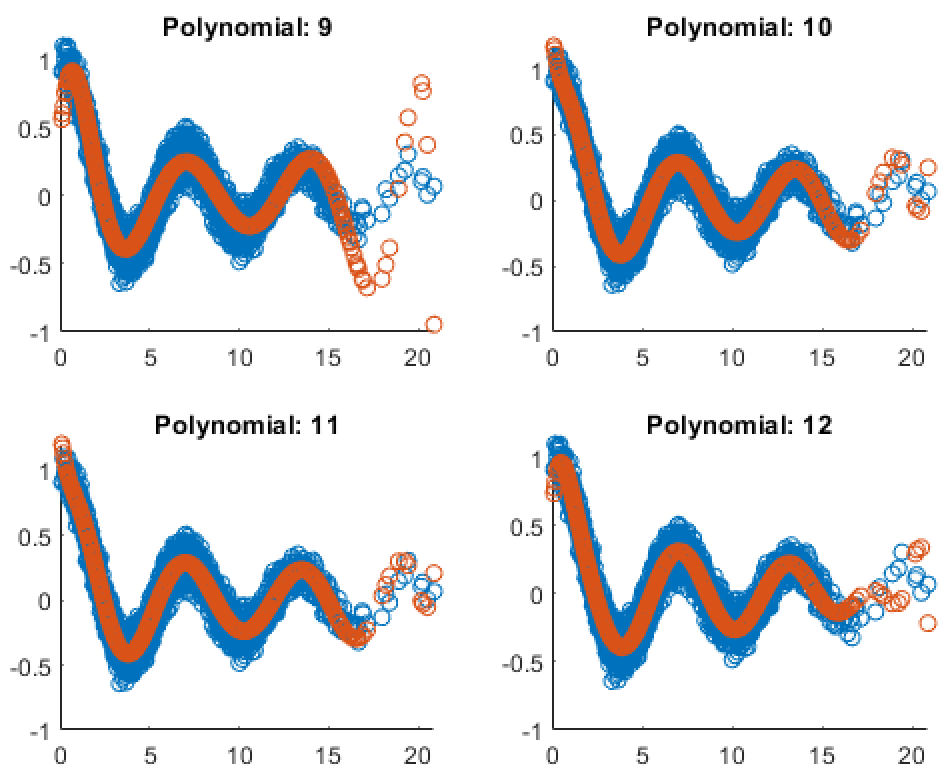
```
po_table = table(po',r2sumt, 'VariableNames', tfields)

%plots

figure
%3 2x2, 1x1
z = 1;
for pID = 1:4

    subplot(2,2,pID);
    scatter(R,y)
    hold on
    scatter(R',yfit(z,:))
    title(['Polynomial: ', num2str(z)])
    z = z+1;
end

figure
for pID = 1:4

    subplot(2,2,pID);
    scatter(R,y)
    hold on
    scatter(R',yfit(z,:))
    title(['Polynomial: ', num2str(z)])
    z = z+1;
end

figure
for pID = 1:4

    subplot(2,2,pID);
    scatter(R,y)
    hold on
    scatter(R',yfit(z,:))
    title(['Polynomial: ', num2str(z)])
    z = z+1;
end

figure
for pID = 1:2

    subplot(2,1,pID);
    scatter(R,y)
    hold on
    scatter(R',yfit(z,:))
    title(['Polynomial: ', num2str(z)])
    z = z+1;
end
```

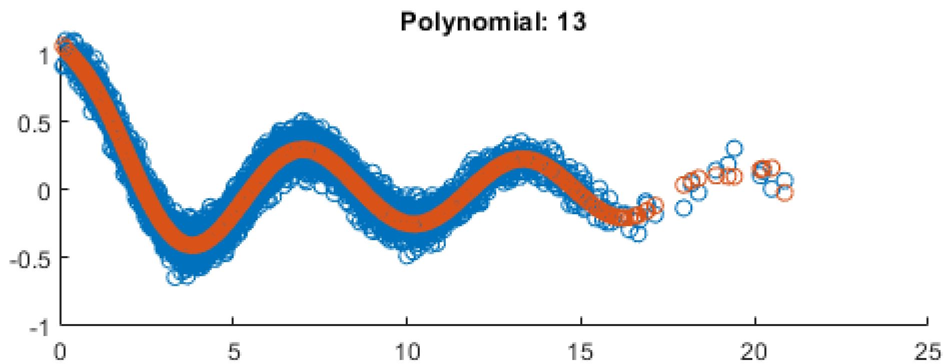
```
po_table =

  14×2 table

    Polynomial order (p)    ||r||^2 for a regular least-squares fit
    ____________________    _______________________________________

             1                              425.17
             2                              412.42
```

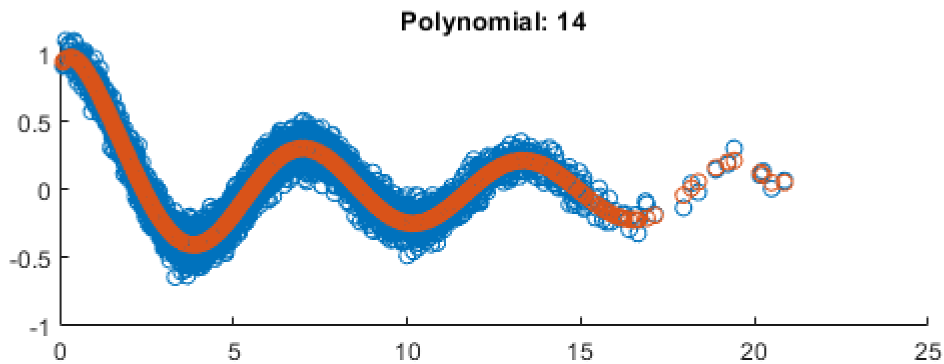
3	369.4
4	268.61
5	168.66
6	164.73
7	105.34
8	47.384
9	43.922
10	28.828
11	28.81
12	29.38
13	27.887
14	27.722




Polynomial: 5
Polynomial: 6
Polynomial: 7
Polynomial: 8
0 5 10 15 20


Polynomial: 9
Polynomial: 10
Polynomial: 11
Polynomial: 12
0 5 10 15 20





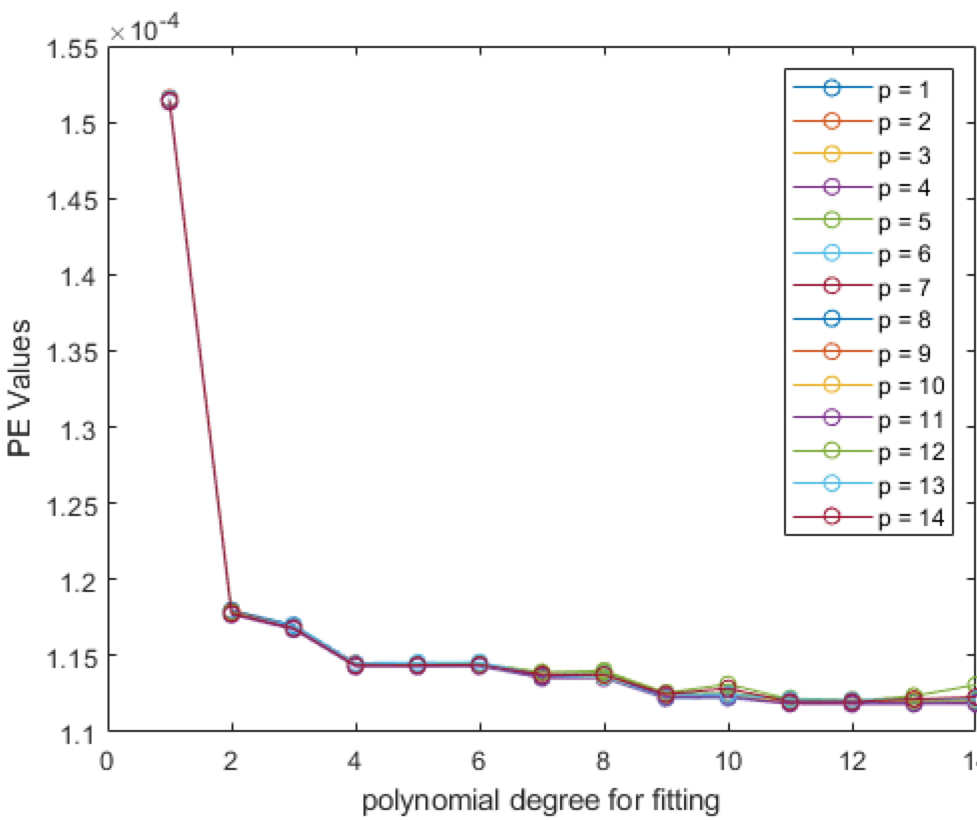
Part 2

```
for numPEcurves = 1:1:20
    bins = randsample(5000,5000);
    tot = reshape(bins,1000,5);

    for p = 1:1:14

        for mse_val = 1:1:5
          tot2 = tot;
          tot2(:,mse_val) = [];
          rn.train{mse_val} = r(tot2);
          rn.test{mse_val} = r(tot(:,mse_val));
          yn.train{mse_val} = y(tot2);
          yn.test{mse_val} = y(tot(:,mse_val));

          x = rn.train{mse_val}(:);
          ynew = yn.train{mse_val}(:);

          %now do LSQ
          AA = ones(length(x),p+1);
          for n2 = 1:p
              AA(:,n2+1) = x.^n2;

          end
          B_hat2.t = (AA' * AA) \ (AA' * ynew);
          B_hat2.i{numPEcurves}{p} = fliplr(B_hat2.t');

          %now that we have coeffs
          %use polyval to fit on top of test data
          %obtain mse, average,save final
          %ultimately 20x14 mat
```

```
            yfit2.t = polyval(B_hat2.i{numPEcurves}{p}, rn.test{mse_val}');
            yfit2.i{numPEcurves}{p} = yfit2.t;

            diff12.t{mse_val} = (yfit2.t - yn.test{mse_val}');
            mse(mse_val) = 1/1000 * sum(diff12.t{mse_val}.^2);
        end
        MSE.t = 1/5 * sum(mse);
        mse_tot(numPEcurves,p) = MSE.t;
%           pause(10)
    end
end

tfields = {'p = 1' 'p = 2' 'p = 3' 'p = 4' 'p = 5' 'p = 6' 'p = 7' 'p = 8' 'p = 9' 'p = 10' 'p = 11'
'p = 12' 'p = 13' 'p = 14'};


pe_table = table(mse_tot(:,1), mse_tot(:,2), mse_tot(:,3) ,mse_tot(:,4), mse_tot(:,5), ...
 mse_tot(:,6), mse_tot(:,7), mse_tot(:,8), mse_tot(:,9), mse_tot(:,10), mse_tot(:,11),
mse_tot(:,12), ...
 mse_tot(:,13), mse_tot(:,14),'VariableNames', tfields)


figure
for pp = 1:1:14
    plot(1:1:14, mse_tot(pp,:), '-o')
    hold on
end
legend(tfields)
ylabel('PE Values')
xlabel('polynomial degree for fitting')
```

```
pe_table =

  20×14 table

      p = 1         p = 2         p = 3         p = 4         p = 5         p = 6         p = 7
p = 8         p = 9         p = 10        p = 11        p = 12        p = 13        p = 14
    __________    __________    __________    __________    __________    __________    __________
__________    __________    __________    __________    __________    __________    __________

    0.00015143    0.00011772    0.00011676    0.00011433    0.00011433    0.00011434    0.00011357
0.00011359    0.00011225    0.00011234    0.00011194    0.00011195    0.00011199      0.000112
    0.00015139    0.00011775    0.00011677    0.00011433    0.00011434    0.00011437    0.00011353
0.00011354    0.00011222    0.00011224    0.00011187    0.00011184    0.00011185    0.00011186
    0.00015155    0.00011776    0.00011687    0.00011442    0.00011444    0.00011446    0.00011371
0.00011372    0.00011244    0.00011241    0.00011202    0.00011199    0.00011208    0.00011209
    0.00015136     0.0001177    0.00011673    0.00011429     0.0001143    0.00011431    0.00011354
0.00011352     0.0001122    0.00011223    0.00011185    0.00011182    0.00011185    0.00011185
    0.00015144     0.0001178    0.00011687    0.00011446    0.00011449    0.00011453    0.00011375
0.00011387    0.00011249    0.00011253     0.0001122    0.00011212    0.00011217    0.00011218
    0.00015143    0.00011772    0.00011677    0.00011434    0.00011435    0.00011439    0.00011361
0.0001136    0.00011225    0.00011226    0.00011191    0.00011189      0.0001119    0.00011192
    0.00015153    0.00011787    0.00011693    0.00011451    0.00011453    0.00011455    0.00011377
0.00011376    0.00011245    0.00011247     0.0001121    0.00011207    0.00011209    0.00011211
    0.00015161    0.00011798    0.00011702    0.00011453    0.00011455    0.00011457    0.00011376
0.00011377    0.00011245    0.00011247    0.00011208    0.00011206    0.00011208    0.00011211
     0.0001516    0.00011789    0.00011693    0.00011451    0.00011451    0.00011452    0.00011373
0.00011375    0.00011242    0.00011248    0.00011209    0.00011209     0.0001121    0.00011212
    0.00015142    0.00011784    0.00011685    0.00011446     0.0001145    0.00011454    0.00011373
```

```
0.0001137    0.00011239    0.00011241    0.00011204    0.00011205    0.00011208    0.00011211
    0.00015133    0.00011771    0.00011673    0.00011431    0.00011434    0.00011443    0.00011363
0.00011378    0.00011231    0.00011233    0.00011187    0.00011187    0.00011189     0.0001119
    0.00015142    0.00011784    0.00011685    0.00011448    0.00011449    0.00011452    0.00011393
0.00011402    0.00011257    0.00011313    0.00011215    0.00011204    0.00011236    0.00011309
    0.00015154    0.00011777    0.00011696    0.00011448    0.00011454    0.00011458    0.00011376
0.00011378    0.00011247    0.00011246    0.00011212    0.00011207    0.00011216    0.00011218
    0.00015143    0.00011778    0.00011682    0.00011438    0.00011439    0.00011441    0.00011377
0.00011375    0.00011248    0.00011286    0.00011198    0.00011198    0.00011215    0.00011229
    0.00015136    0.00011778    0.00011682    0.00011435    0.00011436    0.00011437     0.0001136
0.00011358     0.0001123    0.00011236    0.00011196    0.00011196    0.00011197    0.00011198
    0.00015155    0.00011772    0.00011683    0.00011435    0.00011436    0.00011438    0.00011358
0.00011361    0.00011225    0.00011231    0.00011192    0.00011193    0.00011194    0.00011197
    0.00015148    0.00011768    0.00011673    0.00011429     0.0001143    0.00011432    0.00011352
0.00011352    0.00011224    0.00011224    0.00011191    0.00011189    0.00011192    0.00011193
    0.00015156    0.00011777    0.00011685    0.00011439     0.0001144    0.00011444    0.00011367
0.00011366    0.00011237    0.00011239    0.00011205    0.00011205    0.00011206    0.00011207
    0.00015145    0.00011776    0.00011679    0.00011436    0.00011436    0.00011438    0.00011361
0.00011359    0.00011229    0.00011231    0.00011197    0.00011197     0.000112    0.00011203
    0.00015155    0.00011786     0.0001169    0.00011447    0.00011451    0.00011452    0.00011372
0.0001137    0.00011239    0.00011241    0.00011205    0.00011202    0.00011203    0.00011204
```



Part 3

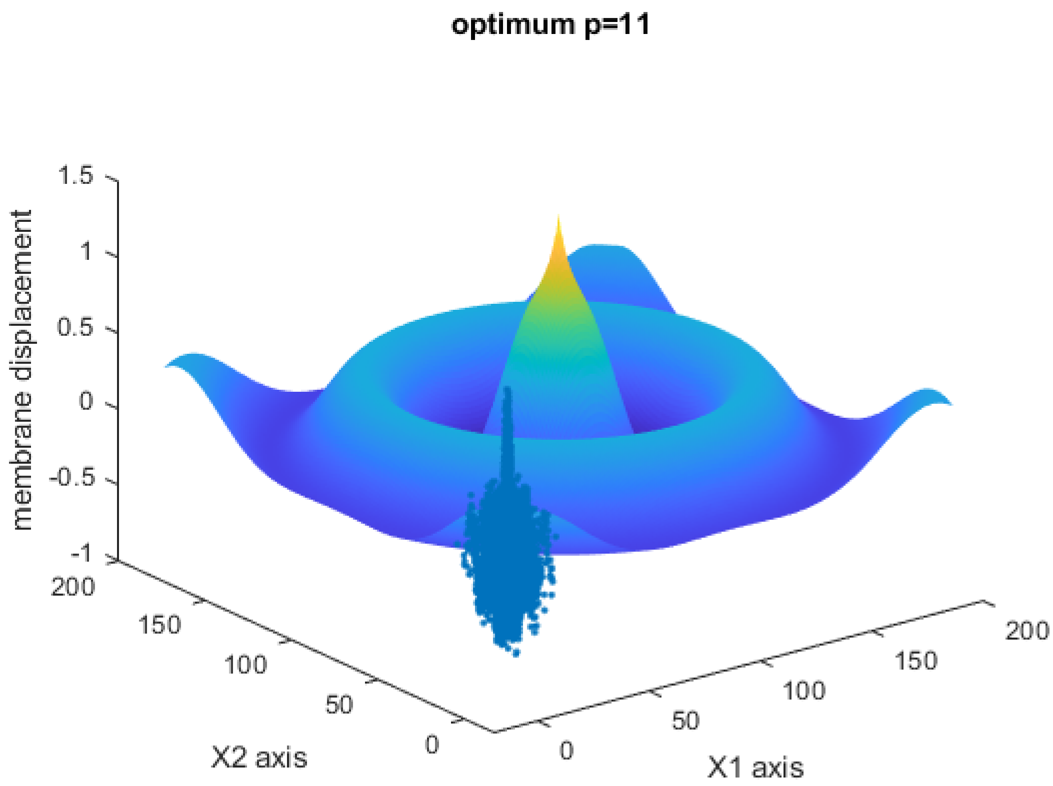
```
disp(' Polynomial model p=11 gives the optimum fit, visually it is the second model to describe the
data, (second lowest ||r||^2) and has agreement on the PE plot.')
disp(' Initially visually I assumed 10 would be best since it was the first, however looking at the
PE plot, there seems to be some disagreement, thus I chose p = 11')
```

Polynomial model p=11 gives the optimum fit, visually it is the second model to describe the data, (second lowest ||r||^2) and has agreement on the PE plot.
Initially visually I assumed 10 would be best since it was the first, however looking at the PE plot, there seems to be some disagreement, thus I chose p = 11

Part 4

```
%p = 11


figure
plot3(x1, x2, y, '.')
hold on
x1_mesh = -10: 0.1 : 10;
x2_mesh = -10: 0.1 : 10;
[X1, X2] = meshgrid(x1_mesh, x2_mesh);
Rmesh = sqrt(X1.^2   +  X2.^2);

beta = B_hat.i{1,11};
Z = polyval( B_hat.i{1,11}, Rmesh);
a = surf(Z);
a.EdgeColor = 'none';

title('optimum p=11')
xlabel('X1 axis')
ylabel('X2 axis')
zlabel('membrane displacement')
disp('I am confused as to why the surf plot is centered around 100 rather than 0, the overall shape matches the ripples in the original data.')
```

```
I am confused as to why the surf plot is centered around 100 rather than 0, the overall shape matches the ripples in the original data.
```



Echoing final values

```
diary vjprob1.txt
echo on
disp('Part 1')
po_table
disp('Part 2')
pe_table

disp('Part 3')
disp(' Polynomial model p=11 gives the optimum fit, visually it is the second model to describe the
data, (second lowest ||r||^2) and has agreement on the PE plot.')
disp(' Initially visually I assumed 10 would be best since it was the first, however looking at the
PE plot, there seems to be some disagreement, thus I chose p = 11')

echo off
```

```
disp('Part 1')
Part 1
po_table

po_table =

  14×2 table

    Polynomial order (p)    ||r||^2 for a regular least-squares fit
    ____________________    _______________________________________

             1                               425.17
             2                               412.42
             3                                369.4
```

```
             4                              268.61
             5                              168.66
             6                              164.73
             7                              105.34
             8                              47.384
             9                              43.922
            10                              28.828
            11                               28.81
            12                               29.38
            13                              27.887
            14                              27.722

disp('Part 2')
Part 2
pe_table

pe_table =

  20×14 table

      p = 1          p = 2          p = 3          p = 4          p = 5          p = 6          p = 7
p = 8          p = 9          p = 10         p = 11         p = 12         p = 13         p = 14
    __________    __________    __________    __________    __________    __________    __________
__________    __________    __________    __________    __________    __________    __________

    0.00015143    0.00011772    0.00011676    0.00011433    0.00011433    0.00011434    0.00011357
0.00011359    0.00011225    0.00011234    0.00011194    0.00011195    0.00011199      0.000112
    0.00015139    0.00011775    0.00011677    0.00011433    0.00011434    0.00011437    0.00011353
0.00011354    0.00011222    0.00011224    0.00011187    0.00011184    0.00011185    0.00011186
    0.00015155    0.00011776    0.00011687    0.00011442    0.00011444    0.00011446    0.00011371
0.00011372    0.00011244    0.00011241    0.00011202    0.00011199    0.00011208    0.00011209
    0.00015136     0.0001177    0.00011673    0.00011429     0.0001143    0.00011431    0.00011354
0.00011352     0.0001122    0.00011223    0.00011185    0.00011182    0.00011185    0.00011185
    0.00015144     0.0001178    0.00011687    0.00011446    0.00011449    0.00011453    0.00011375
0.00011387    0.00011249    0.00011253     0.0001122    0.00011212    0.00011217    0.00011218
    0.00015143    0.00011772    0.00011677    0.00011434    0.00011435    0.00011439    0.00011361
0.0001136    0.00011225    0.00011226    0.00011191    0.00011189     0.0001119    0.00011192
    0.00015153    0.00011787    0.00011693    0.00011451    0.00011453    0.00011455    0.00011377
0.00011376    0.00011245    0.00011247     0.0001121    0.00011207    0.00011209    0.00011211
    0.00015161    0.00011798    0.00011702    0.00011453    0.00011455    0.00011457    0.00011376
0.00011377    0.00011245    0.00011247    0.00011208    0.00011206    0.00011208    0.00011211
     0.0001516    0.00011789    0.00011693    0.00011451    0.00011451    0.00011452    0.00011373
0.00011375    0.00011242    0.00011248    0.00011209    0.00011209     0.0001121    0.00011212
    0.00015142    0.00011784    0.00011685    0.00011446     0.0001145    0.00011454    0.00011373
0.0001137    0.00011239    0.00011241    0.00011204    0.00011205    0.00011208    0.00011211
    0.00015133    0.00011771    0.00011673    0.00011431    0.00011434    0.00011443    0.00011363
0.00011378    0.00011231    0.00011233    0.00011187    0.00011187    0.00011189     0.0001119
    0.00015142    0.00011784    0.00011685    0.00011448    0.00011449    0.00011452    0.00011393
0.00011402    0.00011257    0.00011313    0.00011215    0.00011204    0.00011236    0.00011309
    0.00015154    0.00011777    0.00011696    0.00011448    0.00011454    0.00011458    0.00011376
0.00011378    0.00011247    0.00011246    0.00011212    0.00011207    0.00011216    0.00011218
    0.00015143    0.00011778    0.00011682    0.00011438    0.00011439    0.00011441    0.00011377
0.00011375    0.00011248    0.00011286    0.00011198    0.00011198    0.00011215    0.00011229
    0.00015136    0.00011778    0.00011682    0.00011435    0.00011436    0.00011437     0.0001136
0.00011358     0.0001123    0.00011236    0.00011196    0.00011196    0.00011197    0.00011198
    0.00015155    0.00011772    0.00011683    0.00011435    0.00011436    0.00011438    0.00011358
0.00011361    0.00011225    0.00011231    0.00011192    0.00011193    0.00011194    0.00011197
    0.00015148    0.00011768    0.00011673    0.00011429     0.0001143    0.00011432    0.00011352
0.00011352    0.00011224    0.00011224    0.00011191    0.00011189    0.00011192    0.00011193
    0.00015156    0.00011777    0.00011685    0.00011439     0.0001144    0.00011444    0.00011367
0.00011366    0.00011237    0.00011239    0.00011205    0.00011205    0.00011206    0.00011207
```

```
    0.00015145    0.00011776    0.00011679    0.00011436    0.00011436    0.00011438    0.00011361
0.00011359    0.00011229    0.00011231    0.00011197    0.00011197      0.000112    0.00011203
    0.00015155    0.00011786     0.0001169    0.00011447    0.00011451    0.00011452    0.00011372
0.0001137    0.00011239    0.00011241    0.00011205    0.00011202    0.00011203    0.00011204


disp('Part 3')
Part 3
disp(' Polynomial model p=11 gives the optimum fit, visually it is the second model to describe the
data, (second lowest ||r||^2) and has agreement on the PE plot.')
 Polynomial model p=11 gives the optimum fit, visually it is the second model to describe the data,
(second lowest ||r||^2) and has agreement on the PE plot.
disp(' Initially visually I assumed 10 would be best since it was the first, however looking at the
PE plot, there seems to be some disagreement, thus I chose p = 11')
 Initially visually I assumed 10 would be best since it was the first, however looking at the PE
plot, there seems to be some disagreement, thus I chose p = 11

echo off
```

Published with MATLAB® R2020b

Contents

- BE700 HW1 Prob 2
- loading and prep
- Part 2
- Part 3
- part 4
- Echoing Results

BE700 HW1 Prob 2

```
clear all
close all
warning('off', 'all') %warnings got annoying
```

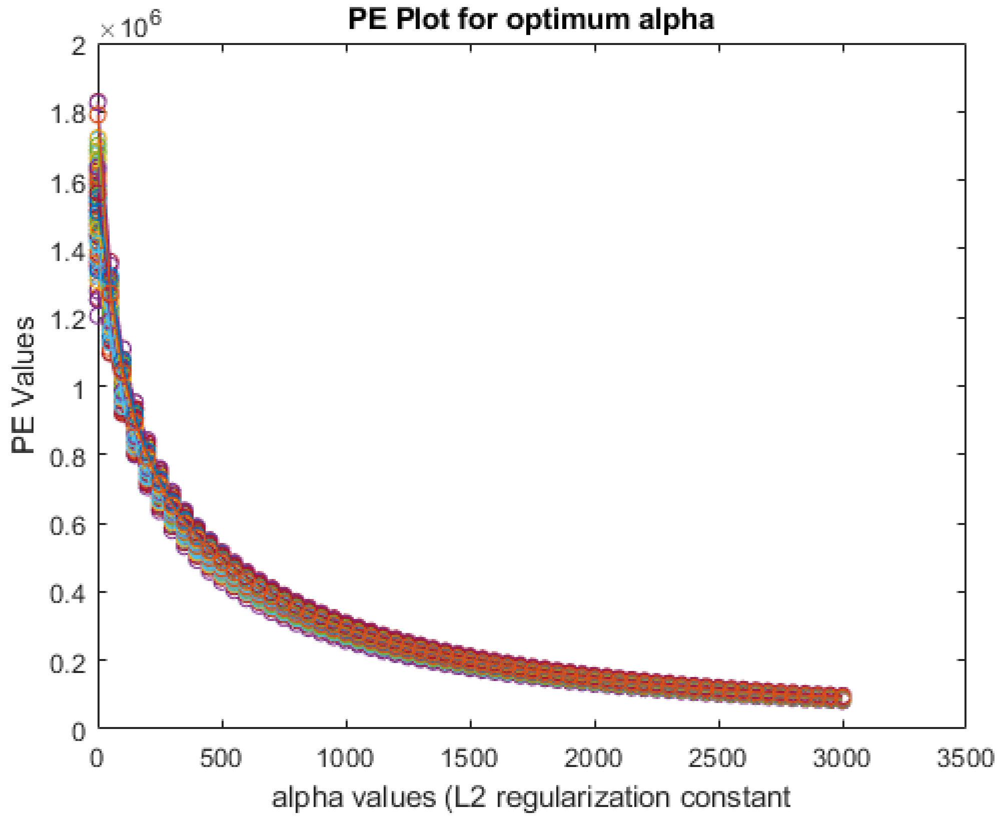
loading and prep

```
temp = readtable('winequality_red.csv', 'HeaderLines',1);
DataTot = temp{:,:};

%only want factor 3 and 1

t = DataTot(:,3);
y = DataTot(:,1);

%Instead of making randsample bin uneven, I am going to delete last 5
%values

t = t(1:1595,:);
y = y(1:1595,:);

%standardize

u = (t - mean(t))./std(t);
```

Part 2

```
i = 1;
for numPEcurves = 1:1:100
    bins = randsample(1595,1595);
    tot = reshape(bins,319,5);

    for p = 1:50:3000+1

        for mse_val = 1:1:5
          tot2 = tot;
          tot2(:,mse_val) = [];
          rn.train{mse_val} = u(tot2);
          rn.test{mse_val} = u(tot(:,mse_val));
          yn.train{mse_val} = y(tot2);
          yn.test{mse_val} = y(tot(:,mse_val));

          x = rn.train{mse_val}(:);
          ynew = yn.train{mse_val}(:);
```

```
            %use ridge to get coeffs
            for n2 = 1:9
                X(:,n2) = x.^n2;
            end

            B_hat2.t = ridge(ynew, X, p);
            B_hat2.i{numPEcurves}{p} = B_hat2.t;
            yfit2.t = polyval(B_hat2.i{numPEcurves}{p}', rn.test{mse_val}');
            yfit2.i{numPEcurves}{p} = yfit2.t;

            diff12.t{mse_val} = (yfit2.t - yn.test{mse_val}');
            mse(mse_val) = 1/319 * sum(diff12.t{mse_val}.^2);
        end

        MSE.t = 1/5 * sum(mse);
        mse_tot(numPEcurves,p) = MSE.t;


%           mse_tot(mse_tot==0) = [];
%           pause(10)
    end
end
mse_tot_fx = mse_tot(:,1:50:3001);
figure
for pp = 1:1:100
    plot(1:50:3001, mse_tot_fx(pp,:), '-o')
    hold on
end
% legend(tfields)
ylabel('PE Values')
xlabel('alpha values (L2 regularization constant')
title('PE Plot for optimum alpha')

for n2 = 1:9
    X_full_standardized(:,n2) = u.^n2;
end

alpha = 1251;
w_L2 = fliplr(ridge(y, X_full_standardized, alpha, 0));
```



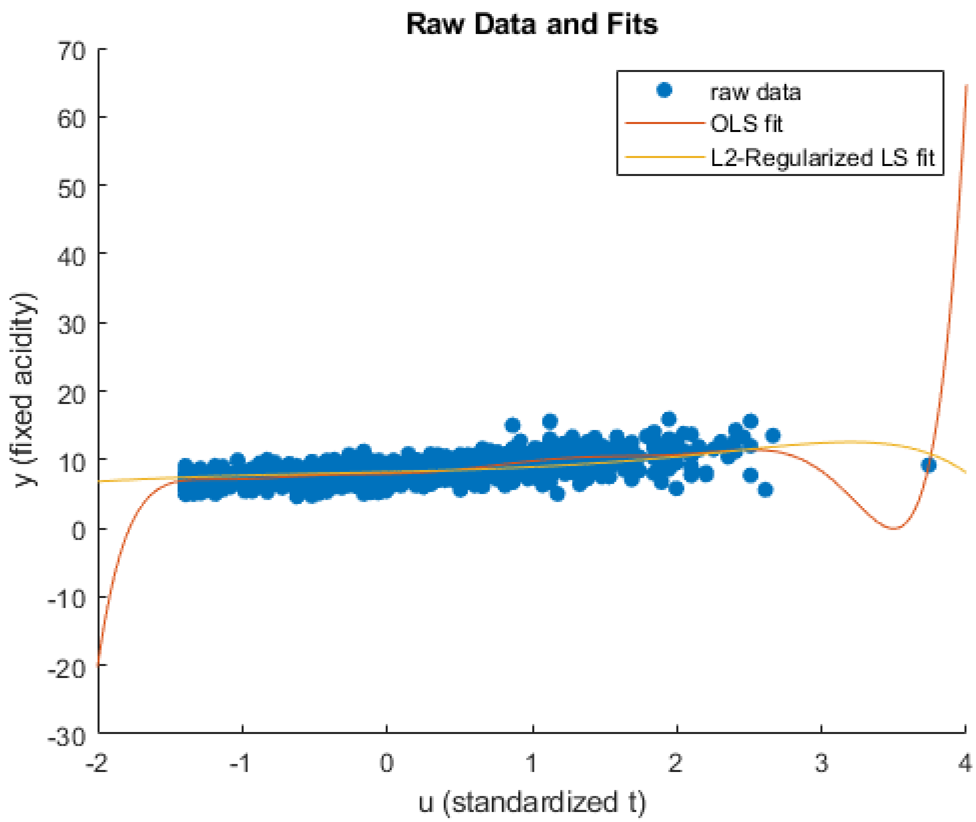
Part 3

```
for P=9
    A= ones(length(u),P+1);
    for n1 = 1:P
        A(:,n1+1) = u.^n1;
    end
    B_hat.temp = (A' * A) \ (A' * y);
    B_hat.i{P} = fliplr(B_hat.temp');
end

w_ordinary = fliplr(B_hat.temp);
```

part 4

```
figure
scatter(u,y, 'o','filled')
hold on
%
% hold on
r_dense = -2:0.003:4;
OLSfit = polyval(flipud(w_ordinary), r_dense);
% OLSfit = OLSfit./100
plot( r_dense, OLSfit);
hold on
L2fit = polyval(flipud(w_L2), r_dense);
plot(r_dense,L2fit)


legend('raw data', 'OLS fit', 'L2-Regularized LS fit')
ylabel('y (fixed acidity)')
```

```
xlabel('u (standardized t)')
title('Raw Data and Fits')
```



Echoing Results

```
diary vjprob2.txt
echo on
disp('Part 2')

fprintf('Alpha optimum is %d\n', alpha)

w_L2

disp('Part 3')
w_ordinary

echo off
```

```
disp('Part 2')
Part 2

fprintf('Alpha optimum is %d\n', alpha)
Alpha optimum is 1251

w_L2

w_L2 =

    8.1915
    0.5291
```

```
    0.1076
    0.0933
    0.0024
    0.0012
   -0.0005
   -0.0002
   -0.0001
   -0.0000


disp('Part 3')
Part 3
w_ordinary

w_ordinary =

    7.9364
    0.4596
    0.9538
    1.9247
   -0.8556
   -1.3396
    0.6032
    0.2417
   -0.1618
    0.0217


echo off
```

HW1_Jha_Vibhav

March 5, 2021

0.1 HW1 Problem 3

0.2 Name: Vibhav Jha

0.2.1 Imports

[1]:
```
import nbconvert
import pandas as pd
import numpy as np
import numpy.linalg as lin
import matplotlib.pyplot as plt
```

0.2.2 1. Loading the data set

a.

[2]:
```
df = pd.read_csv ('data.csv')
```

b.

[3]:
```
df.shape
```

[3]:
```
(569, 33)
```

The dataframe lists itself as having 33 entries, id to Unnamed:32. The description on Kaggle lists it as having 32 columns, as it does not count Unnamed 32, which is an empty column.

c.

[4]:
```
cols = df.columns
print(cols)
```

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst', 'Unnamed: 32'],
      dtype='object')
```

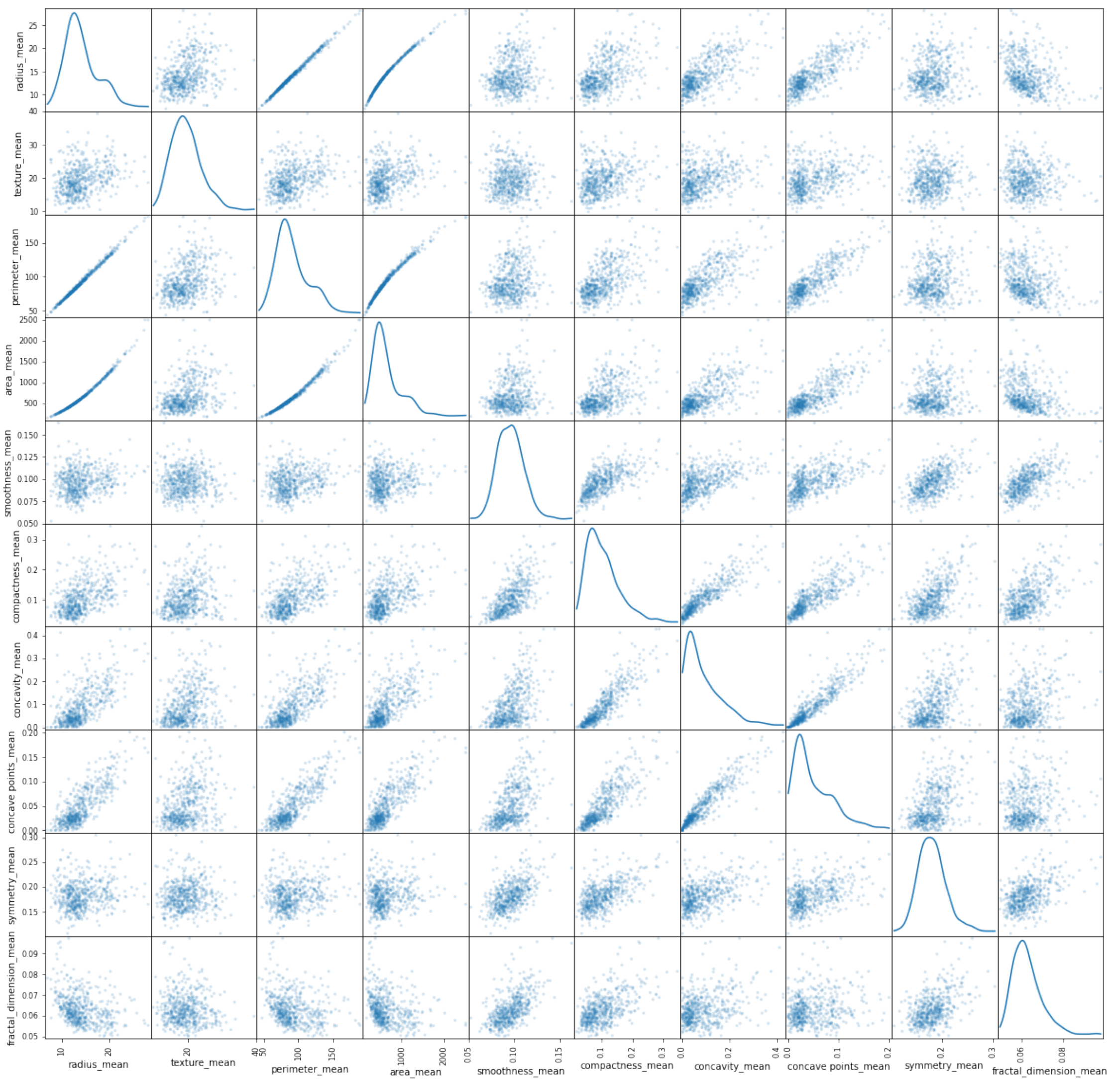
0.2.3 2. Matrix scatter plot

[5]:
```
meanonlydf = df.filter(like = '_mean')
pd.plotting.scatter_matrix(meanonlydf, alpha = 0.2, figsize=(20, 20),␣
 ↪diagonal='kde')
```

[5]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D4A3A490>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5119970>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D514ADC0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5182280>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D51AF6D0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D51DDA60>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D51DDB50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5213040>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D526C850>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5299CA0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D52D3160>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D52FF5B0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5328A00>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5357E50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D538F2E0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D53BA760>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D53E6BE0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D54120D0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D544C4C0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5127790>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5255E50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5398970>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D54FC520>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D55279D0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5554E20>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D558D2B0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D55B7700>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D55E5B50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5612FA0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5648490>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D56768E0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D56A4D30>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D56DC1F0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5709640>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5732A90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D575FEE0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D57973D0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D57C4820>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D57F0C70>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D581C160>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5855550>,
```

```
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D587F9A0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D58AEDF0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D58E6280>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D59136D0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D593FB20>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D596CF70>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D59A5400>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D69A0850>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D69D0820>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D69F9FA0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6A2E760>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6A59EE0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6A8B6A0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6AB6E20>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6AEA5E0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6B13D60>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6B4A580>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6B73D00>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6BA74C0>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6BD1C40>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6C08400>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6C30B80>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6C65340>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6C8CBB0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6CC4370>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6CECAF0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6D212B0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6D4AA30>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6D81220>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6DA99A0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6DDF160>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6E068E0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6E30100>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6E668B0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6E8F0D0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D53F5A90>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D561F5E0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5742400>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D588F940>],
[<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D5981490>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6F9E460>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6FC98B0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D6FF8D00>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D7030190>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D705C5E0>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D7086A60>,
 <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D70B4EB0>,
```

```
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D70EC340>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D7119790>],
 [<matplotlib.axes._subplots.AxesSubplot object at 0x00000290D7146BE0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D71710D0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D71AA4C0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D71D5910>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D7202D60>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D723C1F0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D7268640>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D7293A90>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D72C0EE0>,
  <matplotlib.axes._subplots.AxesSubplot object at 0x00000290D72F8370>]],
dtype=object)
```



0.2.4 3. Calculations

a.

[6]:
```
summary = df['diagnosis'].value_counts()
print(summary)
```

```
B    357
M    212
Name: diagnosis, dtype: int64
```

b.

[7]:
```
data1 = df.to_numpy()
data2 = meanonlydf.to_numpy()
a = []
an = []
an3 = []
for i in range(10):
    a.append(np.mean(data2[:,i]))
    an.append(df.columns[i+2])
    an3.append(np.std(data2[:,i]))

data = {'ColumnNames': an, 'Mean': a, 'Standard Deviation': an3}

df2 = pd.DataFrame(data)

print(df2)
```

```
              ColumnNames        Mean  Standard Deviation
0             radius_mean   14.127292            3.520951
1            texture_mean   19.289649            4.297255
2          perimeter_mean   91.969033           24.277619
3               area_mean  654.889104          351.604754
4         smoothness_mean    0.096360            0.014052
5        compactness_mean    0.104341            0.052766
6          concavity_mean    0.088799            0.079650
7     concave points_mean    0.048919            0.038769
8           symmetry_mean    0.181162            0.027390
9  fractal_dimension_mean    0.062798            0.007054
```

c.

[8]:
```
#malignant
monly =  df[(df['diagnosis'] == 'M')]
monlymeans = monly.filter(like = '_mean')
data3 = monlymeans.to_numpy()

cd = []
cd3 = []
for i in range(10):
    cd.append(np.mean(data3[:,i]))
```

```python
    cd3.append(np.std(data3[:,i]))

data3 = {'Malignant ColumnNames': an, 'Mean': cd, 'Standard Deviation': cd3}
df3 = pd.DataFrame(data3)
print(df3)
print()

#Benign
bonly =  df[(df['diagnosis'] == 'B')]
bonlymeans = bonly.filter(like = '_mean')
data4 = bonlymeans.to_numpy()

cd = []
cd3 = []
for i in range(10):
    cd.append(np.mean(data4[:,i]))
    cd3.append(np.std(data4[:,i]))

data4_x = {'Benign ColumnNames': an, 'Mean': cd, 'Standard Deviation': cd3}
df4 = pd.DataFrame(data4_x)
print(df4)
```

```
   Malignant ColumnNames        Mean  Standard Deviation
0            radius_mean   17.462830            3.196406
1           texture_mean   21.604906            3.770546
2         perimeter_mean  115.365377           21.803048
3              area_mean  978.376415          367.069174
4        smoothness_mean    0.102898            0.012578
5       compactness_mean    0.145188            0.053860
6         concavity_mean    0.160775            0.074842
7    concave points_mean    0.087990            0.034293
8          symmetry_mean    0.192909            0.027573
9 fractal_dimension_mean    0.062680            0.007555

      Benign ColumnNames        Mean  Standard Deviation
0            radius_mean   12.146524            1.778016
1           texture_mean   17.914762            3.989525
2         perimeter_mean   78.075406           11.790889
3              area_mean  462.790196          134.098909
4        smoothness_mean    0.092478            0.013427
5       compactness_mean    0.080085            0.033703
6         concavity_mean    0.046058            0.043381
7    concave points_mean    0.025717            0.015886
8          symmetry_mean    0.174186            0.024772
9 fractal_dimension_mean    0.062867            0.006738
```

d.

[9]:
```
poff = data4[:,1][data4[:,1] > 15]

Bgreater15 = 100*(len(poff)/len(bonly))

print('Percentage of Benign Tumors with Radius at least 15: ', Bgreater15)
```

```
Percentage of Benign Tumors with Radius at least 15:  75.63025210084034
```

0.2.5 4. OLS

a.

[11]:
```
y = df.area_mean.to_numpy()
x = df.radius_mean.to_numpy()

#linear
Aone = np.ones(len(x))
Atwo = x
Atot = np.vstack((Aone,Atwo)).T
#inter = Atot.T.dot(Atot)
reshapey = np.reshape(y, (569,1))
coeff = np.dot(lin.inv(np.dot(Atot.T, Atot)), np.dot(Atot.T, reshapey))

print('Linear OLS Coeff(constant, 1): ', coeff)

yfitlin = np.polyval([coeff[1], coeff[0]], x)

y_diff = y - yfitlin

r2 = np.square(y_diff)

r2sumlin = np.sum(r2)

print('Sum of Residuals Squared: ',r2sumlin)
#abc = np.polyfit(x,y,1,full=True)
#print(abc) #sanity check
```

```
Linear OLS Coeff(constant, 1):  [[-738.0367042 ]
 [  98.59821922]]
Sum of Residuals Squared:  1767428.9562542238
```

b.

[12]:
```
#quadratic
Athree = np.square(x)
Atot2 = np.vstack((Aone,Atwo,Athree)).T
coeff_2 = np.dot(lin.inv(np.dot(Atot2.T, Atot2)), np.dot(Atot2.T, reshapey))

print('Quadratic OLS Coeff(constant, 1, 2): ', coeff_2)
```

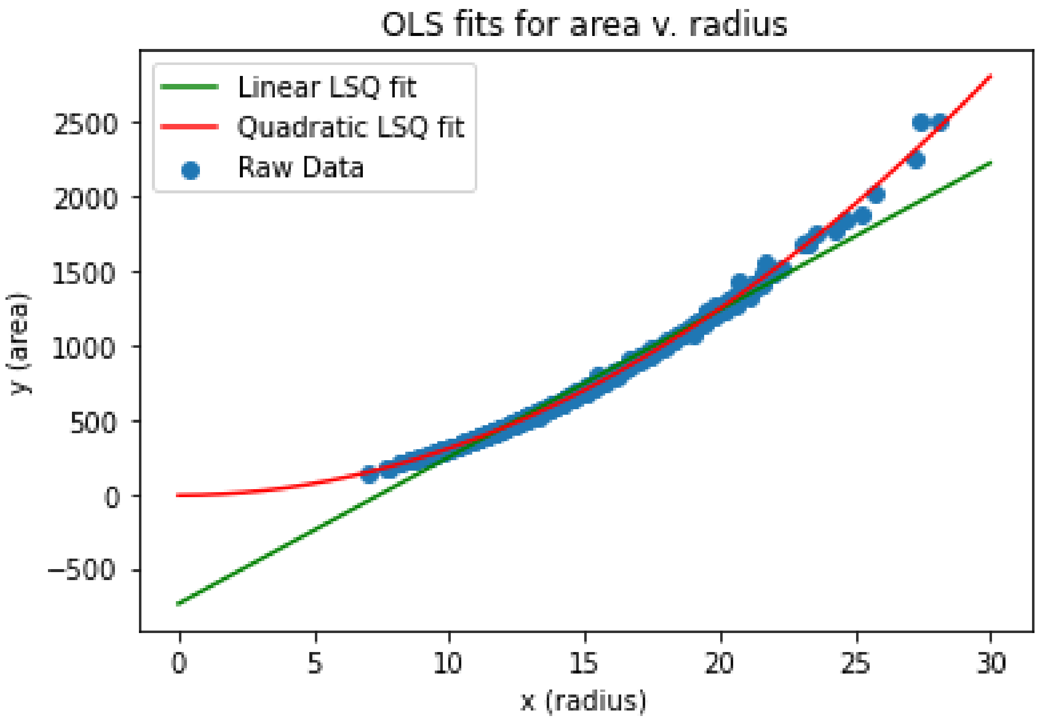
```
yfitqd = np.polyval([coeff_2[2], coeff_2[1], coeff_2[0]], x)

y_diffqd = y - yfitqd

r2qd = np.square(y_diffqd)

r2sumqd = np.sum(r2qd)

print('Sum of Residuals Squared: ',r2sumqd)
#abc = np.polyfit(x,y,2,full=True)
#print(abc) #sanity check
```

```
Quadratic OLS Coeff(constant, 1, 2):  [[-10.5164038 ]
 [  0.43684601]
 [  3.10992516]]
Sum of Residuals Squared:  123097.70230710594
```

0.2.6 5. Plots

a.

[13]:
```
fig, ax = plt.subplots()

ax.scatter(x,y, label='Raw Data')
yfitlin = np.polyval([coeff[1], coeff[0]], np.arange(0,30,0.0528))
plt.plot(np.arange(0,30,0.0528), yfitlin, c= "green", label='Linear LSQ fit')
yfitqd = np.polyval([coeff_2[2], coeff_2[1], coeff_2[0]], np.arange(0,30,0.0528))
plt.plot(np.arange(0,30,0.0528), yfitqd, c= "red", label='Quadratic LSQ fit')

plt.xlabel('x (radius)')
plt.ylabel('y (area)')
plt.title("OLS fits for area v. radius")

plt.legend()
plt.show()
```

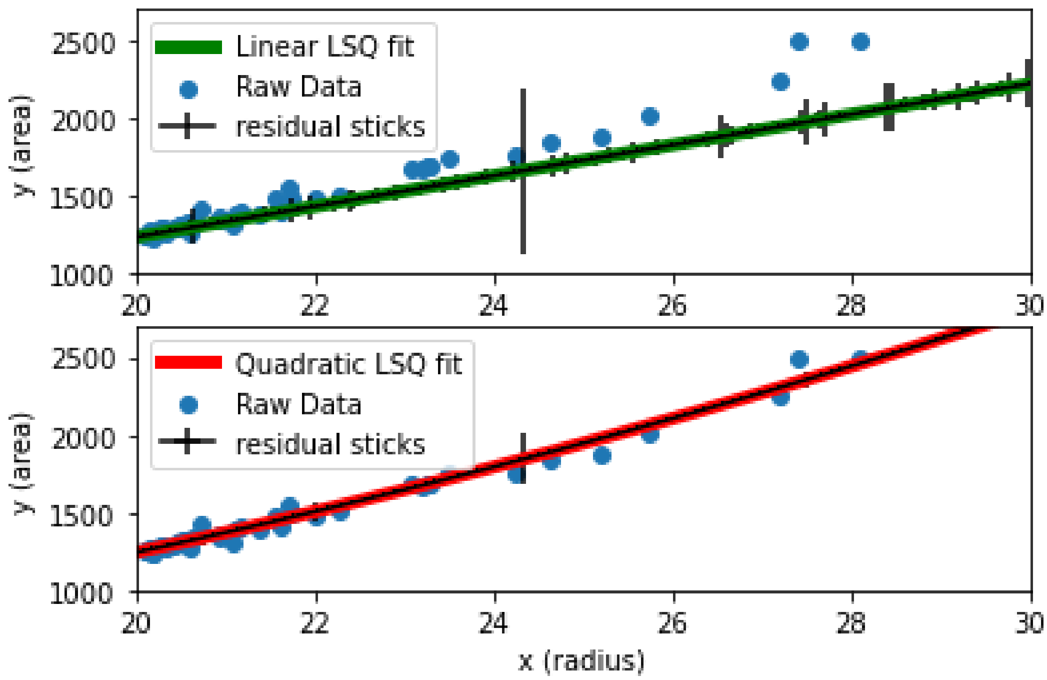


b.

[15]:
```
fig2, (ax1, ax2) = plt.subplots(2, 1)

ax1.scatter(x,y, label='Raw Data')
ax1.plot(np.arange(0,30,0.0528), yfitlin, c= "green", label='Linear LSQ fit', ␣
↪linewidth=5)
xx = np.arange(0,30,0.0528)
ax1.errorbar(xx, yfitlin, yerr = y_diff, marker=',', c='black', label='residual␣
↪sticks')
ax1.set_xlim([20, 30])
ax1.set_ylim([1000, 2700])

ax1.set_ylabel('y (area)')
ax1.legend()

ax2.scatter(x,y, label='Raw Data')
ax2.plot(np.arange(0,30,0.0528), yfitqd, c= "red", label='Quadratic LSQ fit',␣
↪linewidth=5)
y2 = y-yfitqd
ax2.errorbar(xx, yfitqd, xerr = 0, yerr = y_diffqd, marker=',', c='black',␣
↪label='residual sticks')
plt.xlim([20, 30])
ax2.set_ylim([1000, 2700])
```

```
plt.xlabel('x (radius)')
plt.ylabel('y (area)')
plt.legend()
```

[15]: <matplotlib.legend.Legend at 0x290daae27c0>



[ ]: