MAYANK GARG
Pattern Recognition
Project#1

Source files –

1. CurveFit.m – All parts are there which uses the 10 data points. Proper comment and section are made in the file.
2. Ciplot.m – one online file was used to plot the confidence interval. This file is to plot is a nice manner and has nothing to do with the code and algorithm.
3. generateDiffData.m – used to generate the different data sets -10,20..60 samples

Data file-

1. data10.mat – have 10 points
2. data20.mat – have 20 points
3. data30.mat – have 30 points
4. data40.mat – have 40 points
5. data50.mat – have 50 points
6. data60.mat – have 60 points

All extra credit parts are completed.

PART 1- error minimization

*Used the equation given in the notes – w= $(X^TX)^{-1}X^Tt$ to calculate the w vector.

In the file CurveFit.m cells

```
PART 1 curve fit for M-5
PART 1 curve fit for different Ms
PART 1 curve fit for different data set
```

Output figure are -

Curve fit for M=5;

One sample solution for M=5. Further the figure for different M and data points are given

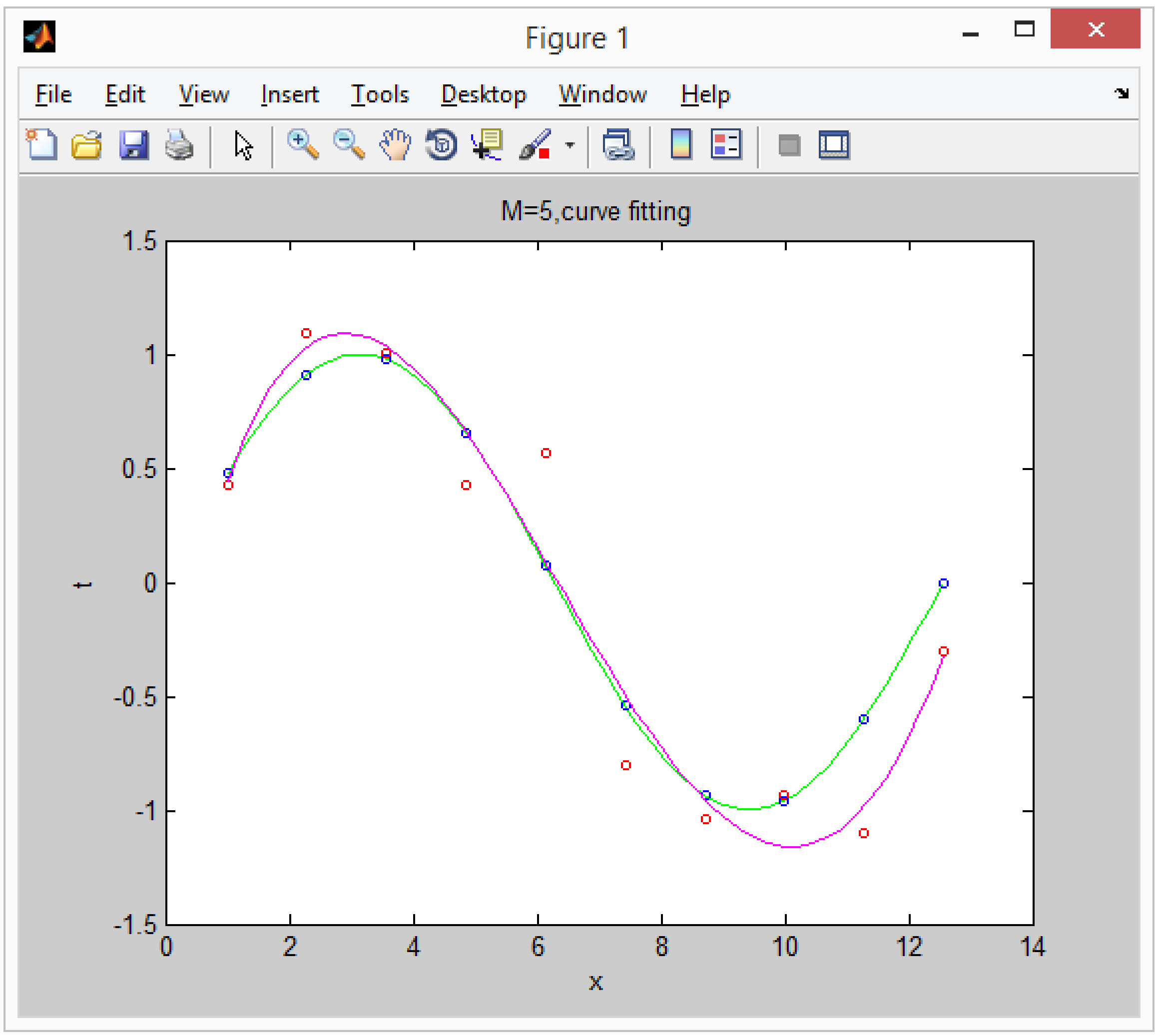
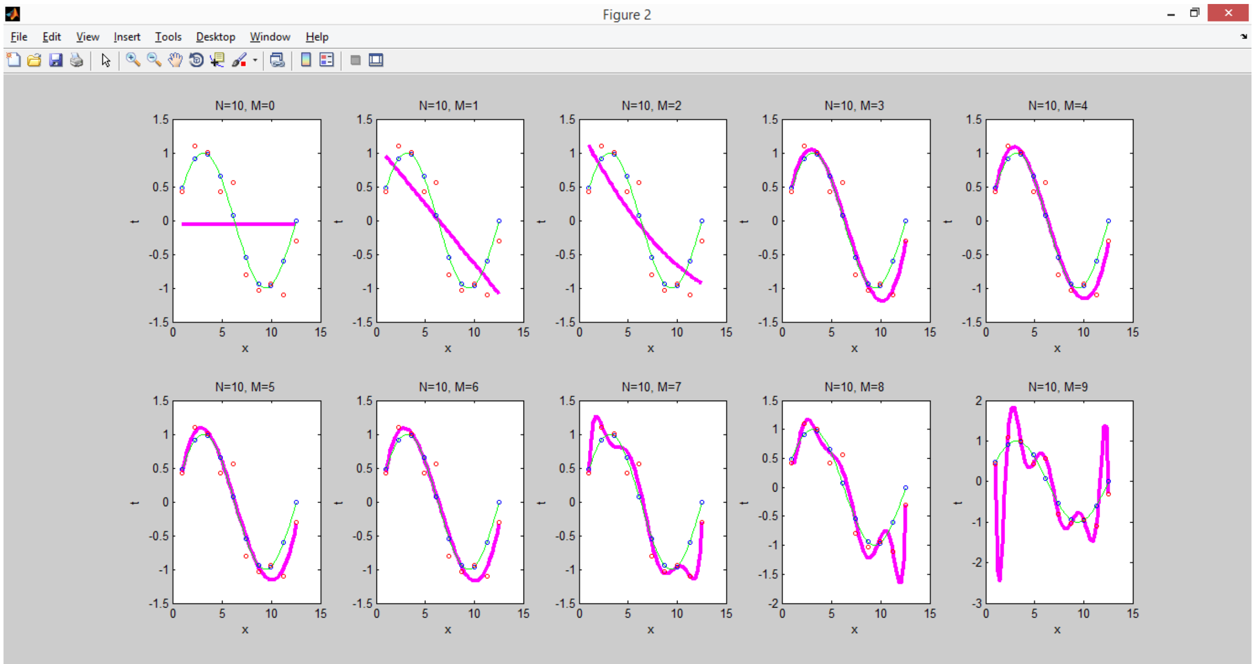


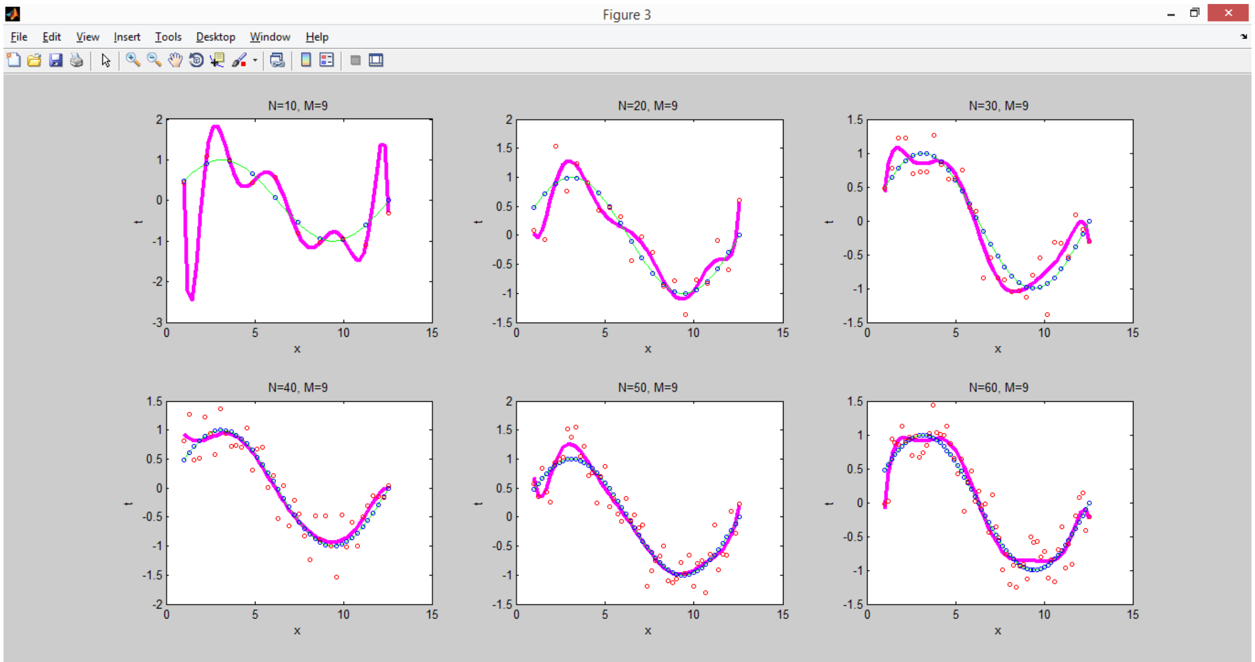
Figure for different M values, it can be seen clearly that higher M will fit the data in a better way. But a very high M as M=9 can over fit the data.



And corresponding W values -

M ->	0	1	2	3	4	5	6	7	8	9
w0	-0.06598	1.131038	1.385969	-0.37104	-0.54724	-0.56815	-0.73823	-7.361	6.986696	106.0109
w1		-0.17647	-0.28326	1.068401	1.269649	1.300813	1.605276	15.12802	-17.2297	-258.881
w2			0.007872	-0.23375	-0.29552	-0.30945	-0.48811	-10.1941	16.85011	244.1954
w3				0.011873	0.018793	0.021374	0.068612	3.411126	-7.98056	-119.981
w4					-0.00026	-0.00046	-0.00667	-0.6234	2.091019	34.61853
w5						6.18E-06	0.000401	0.062797	-0.31875	-6.17963
w6							-9.70E-06	-0.00327	0.028045	0.690362
w7								6.86E-05	-0.00132	-0.04696
w8									2.55E-05	0.001778
w9										-2.87E-05

This problem of overfitting can be solved if more data points are available as can be seen here for M=9 and different N=10,20..60.



Corresponding W values –

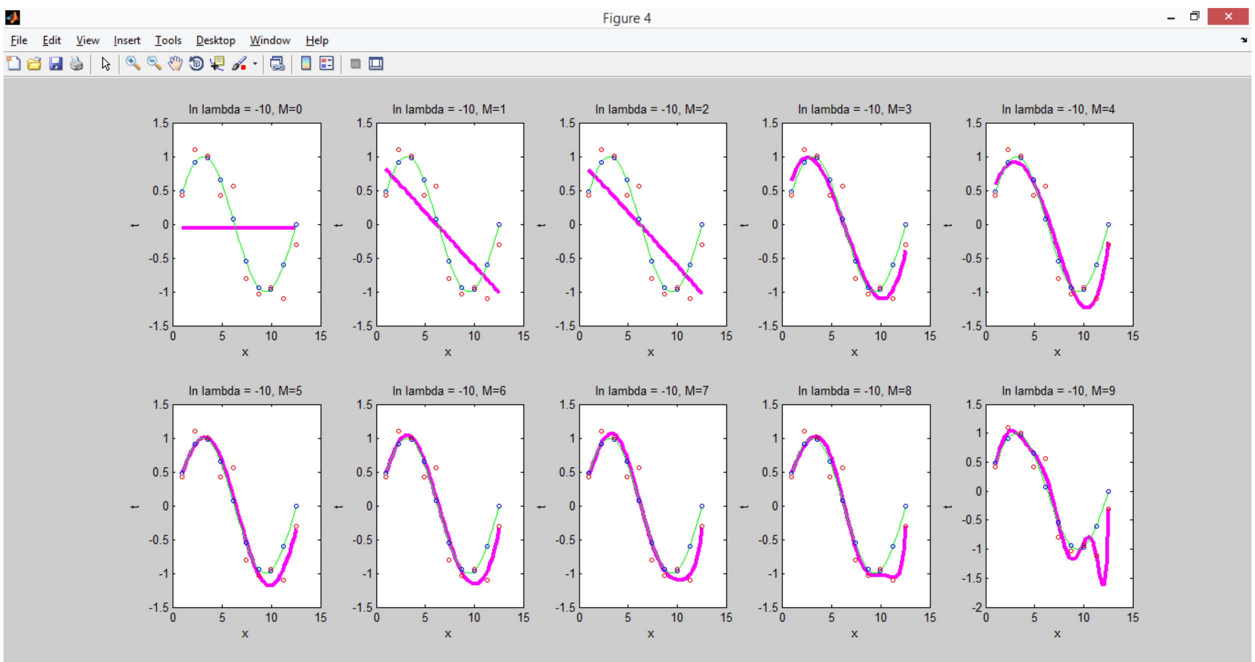
N ->	10	20	30	40	50	60
w0	106.0109	7.14127	-7.34548	1.084917	12.11708	-7.68201
w1	-258.881	-16.6665	15.85007	0.403677	-26.7195	14.31337
w2	244.1954	13.82069	-11.4527	-1.19207	23.34799	-9.1342
w3	-119.981	-5.09179	4.042501	0.885479	-10.3404	2.750211
w4	34.61853	0.890905	-0.72843	-0.30167	2.658099	-0.34334
w5	-6.17963	-0.05307	0.055836	0.055331	-0.42151	-0.01294
w6	0.690362	-0.00556	0.001346	-0.00585	0.041785	0.008991
w7	-0.04696	0.001116	-0.00055	0.000355	-0.00252	-0.00107
w8	0.001778	-6.81E-05	3.60E-05	-1.14E-05	8.42E-05	5.59E-05
w9	-2.87E-05	1.48E-06	-7.95E-07	1.47E-07	-1.20E-06	-1.12E-06

PART 2 –

Corresponding cells in the CurveFit.m matlab file-

```
%% PART 2 add regulariization
%% PART 2 same M, same data, Different lambda
%% PART 2, M =9 lambda=exp(-10), Different Data Points
```

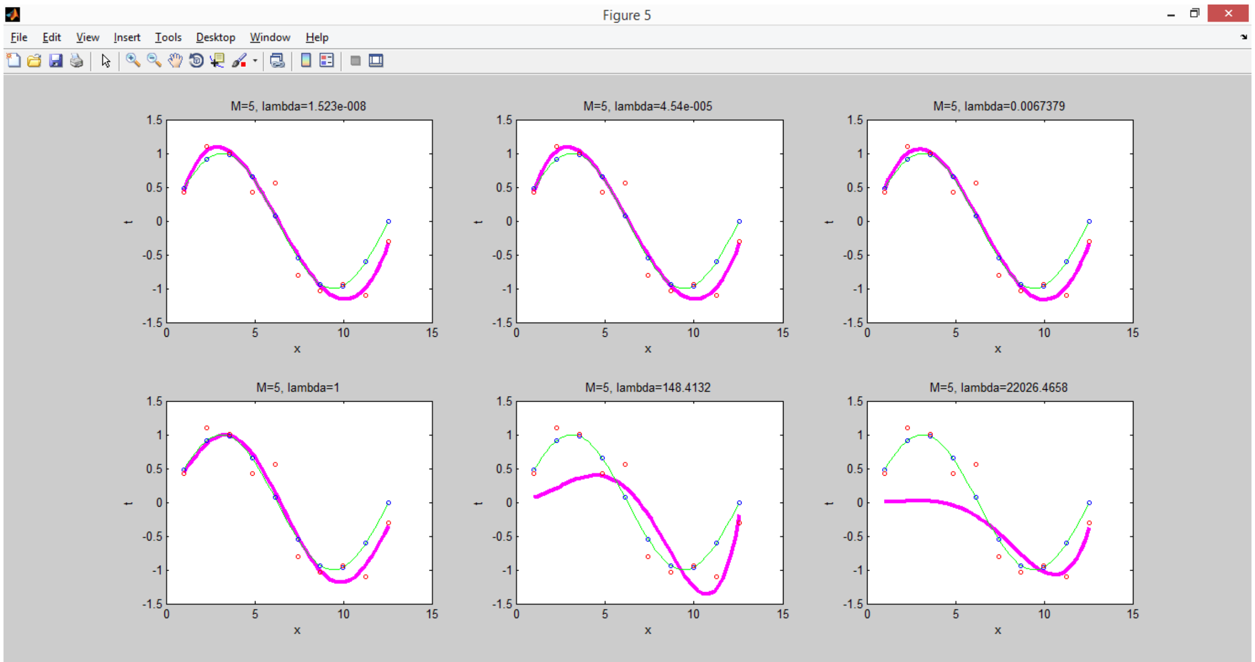
Overfitting problem can also be addressed using the regularization as shown here when N=10, and ln lambda=-1, and different M. Even for M=9 it is less overfit compare to without lambda use.



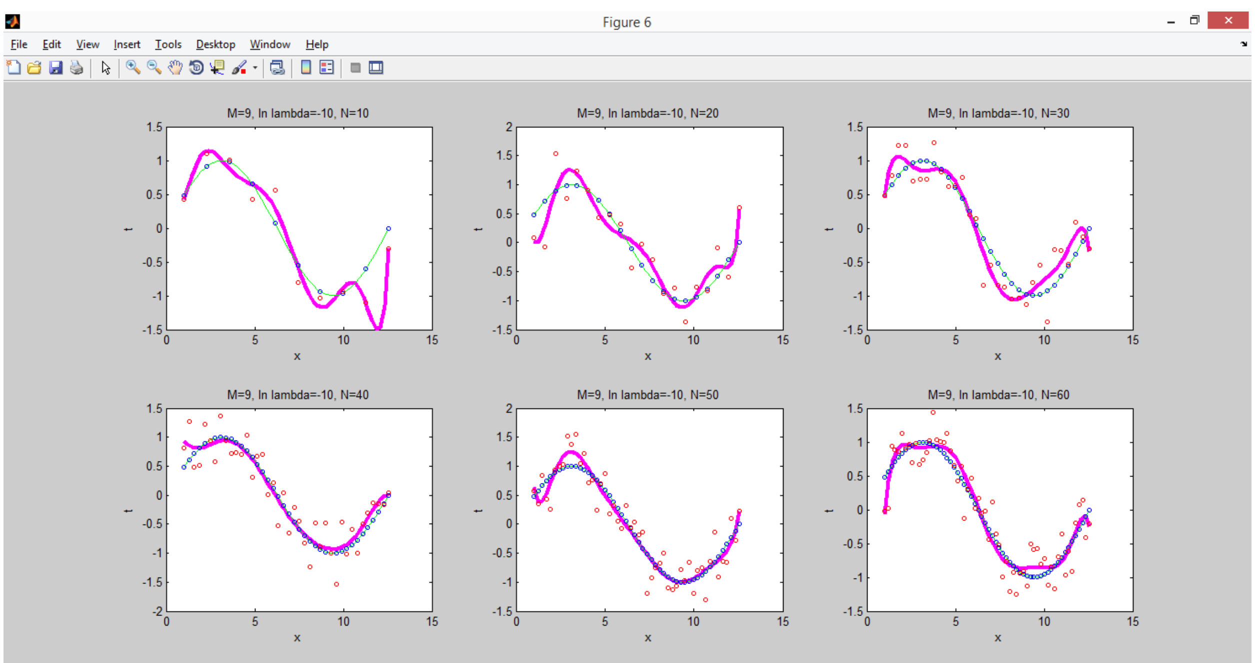
Corresponding W values-

M ->	0	1	2	3	4	5	6	7	8	9
w0	-0.06364	0.971686	0.956165	0.046455	0.167398	0.147263	0.107578	0.147833	0.146822	0.054254
w1		-0.15825	-0.15095	0.777945	0.489493	0.255008	0.225585	0.197778	0.251176	0.175887
w2			-0.00056	-0.18541	-0.06904	0.135573	0.229465	0.129681	0.174465	0.255576
w3				0.009614	-0.00545	-0.05737	-0.10095	-0.01447	-0.12223	0.121133
w4					0.000607	0.005688	0.013516	-0.01232	0.042228	-0.19569
w5						-1.69E-04	-0.00078	0.00278	-0.00947	0.075732
w6							1.73E-05	-0.00021	0.001179	-0.01417
w7								5.71E-06	-7.27E-05	0.001417
w8									1.74E-06	-7.26E-05
w9										1.50E-06

Next figure is for different values of the lambda and same value of M=5, N=10. Here a very interesting observation is that higher value of regularization lambda tends to flatten the prediction. Lambda is addition in cost function and when the cost function is minimized it reduces the value of all coefficients and thus flatten the prediction.



Same lambda, same M value but the data points are different. As expected more data points tends to give better estimation and better curve fitting (not the overfitting) can be seen for higher N in next figure.



Corresponding W values –

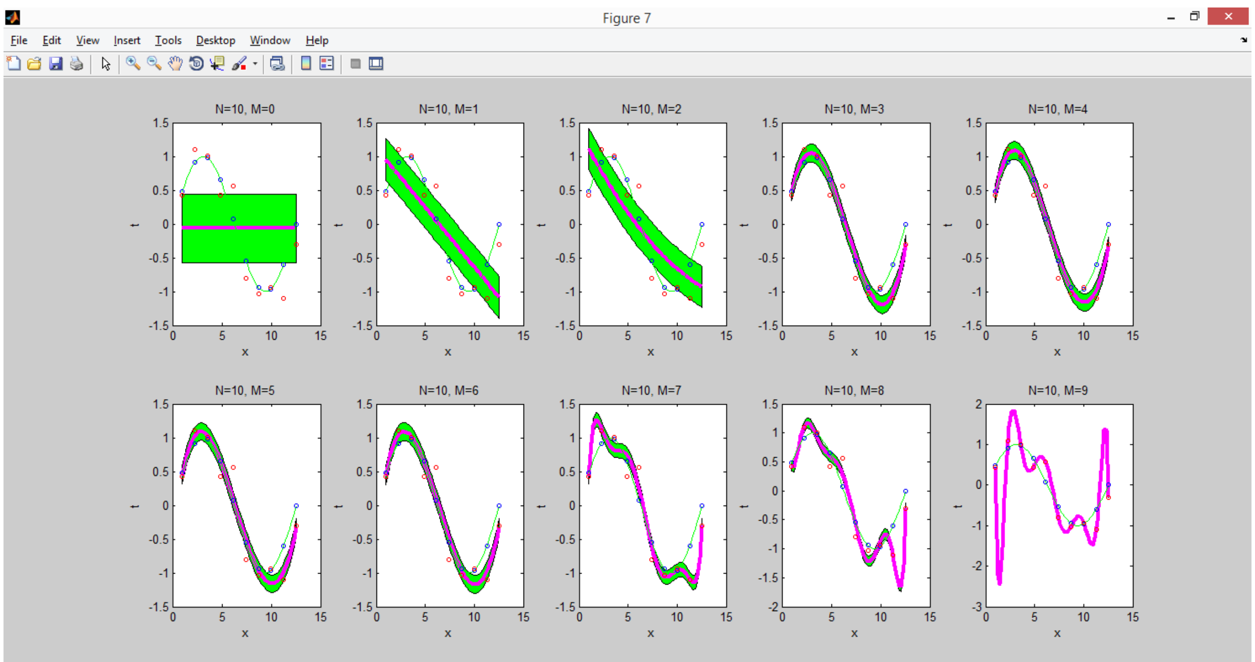
N ->	10	20	30	40	50	60
w0	2.442273	4.255234	-5.36731	1.164929	9.301184	-6.48641
w1	-7.17105	-9.54346	11.04647	0.207899	-20.0007	11.48285
w2	8.664431	6.991436	-6.89414	-1.00447	17.0439	-6.49404
w3	-4.64734	-1.6814	1.782131	0.791545	-7.23755	1.456511
w4	1.330516	-0.10499	-0.07185	-0.27414	1.76122	0.029348
w5	-0.21811	0.125985	-0.06171	0.050363	-0.26146	-0.07928
w6	0.020555	-0.02563	0.014471	-0.00529	0.023952	0.016368
w7	-0.00105	0.00248	-0.00144	0.000317	-0.00131	-0.00157
w8	2.39E-05	-1.20E-04	6.95E-05	-9.97E-06	3.88E-05	7.46E-05
w9	-1.03E-07	2.31E-06	-1.33E-06	1.24E-07	-4.68E-07	-1.42E-06

PART 3

Cells for the code in matlab file CurveFit.m

```
%% PART 3 different M
%% PART 3 Different data points
```

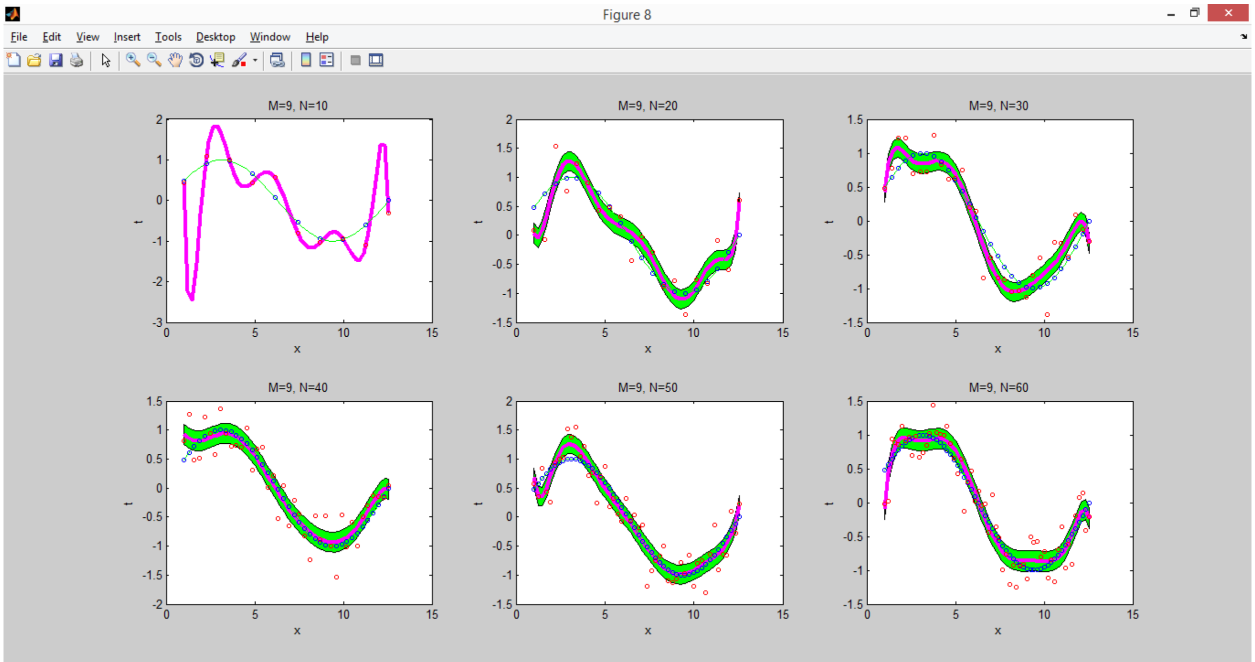
w and beta are calculated as given in notes. After that the 95 percent confidence interval is calculated. For that first beta is used to calculate the variance and then the variance and mean prediction value y_predict is used to calculate the 95 percentage confidence interval. The green area in the figure represents that 95 percent confidence interval. An m file ciplot.m is used this curve. This function file was downloaded online and only used to plot the green filled plot in nice way.



And corresponding Ws-

M ->	0	1	2	3	4	5	6	7	8	9
w0	-0.06598	1.131038	1.385969	-0.37104	-0.54724	-0.56815	-0.73823	-7.361	6.986696	106.0109
w1		-0.17647	-0.28326	1.068401	1.269649	1.300813	1.605276	15.12802	-17.2297	-258.881
w2			0.007872	-0.23375	-0.29552	-0.30945	-0.48811	-10.1941	16.85011	244.1954
w3				0.011873	0.018793	0.021374	0.068612	3.411126	-7.98056	-119.981
w4					-0.00026	-0.00046	-0.00667	-0.6234	2.091019	34.61853
w5						6.18E-06	0.000401	0.062797	-0.31875	-6.17963
w6							-9.70E-06	-0.00327	0.028045	0.690362
w7								6.86E-05	-1.32E-03	-0.04696
w8									2.55E-05	1.78E-03
w9										-2.87E-05

For more number of points the prediction is better and overfitting can be avoided as shown in the next figure. For M=9, there is overfitting for 10 data points but no overfitting for 60 data points.



Corresponding W values-

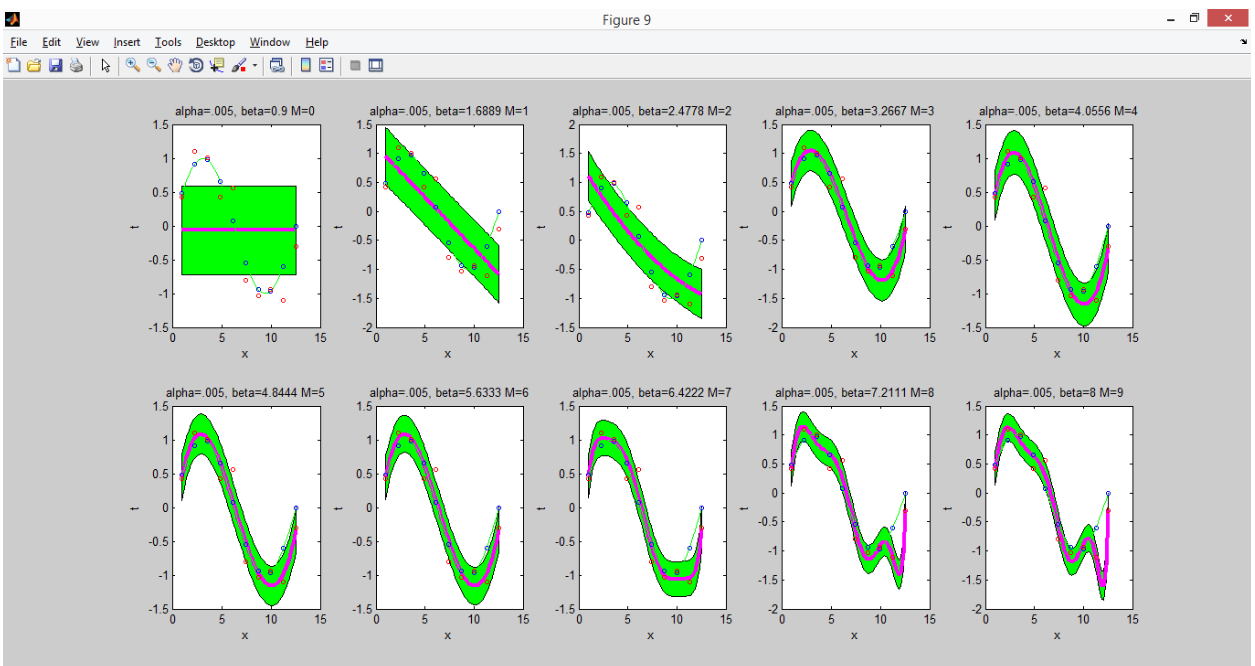
N ->	10	20	30	40	50	60
w0	106.0109	7.14127	-7.34548	1.084917	12.11708	-7.68201
w1	-258.881	-16.6665	15.85007	0.403677	-26.7195	14.31337
w2	244.1954	13.82069	-11.4527	-1.19207	23.34799	-9.1342
w3	-119.981	-5.09179	4.042501	0.885479	-10.3404	2.750211
w4	34.61853	0.890905	-0.72843	-0.30167	2.658099	-0.34334
w5	-6.17963	-0.05307	0.055836	0.055331	-0.42151	-0.01294
w6	0.690362	-0.00556	0.001346	-0.00585	0.041785	0.008991
w7	-0.04696	0.001116	-0.00055	0.000355	-0.00252	-0.00107
w8	1.78E-03	-6.81E-05	3.60E-05	-1.14E-05	8.42E-05	5.59E-05
w9	-2.87E-05	1.48E-06	-7.95E-07	1.47E-07	-1.20E-06	-1.12E-06

PART 4-

Corresponding cells in the CurveFit.m file.

```
%% PART 4 different M
%% PART 4 Different data
```

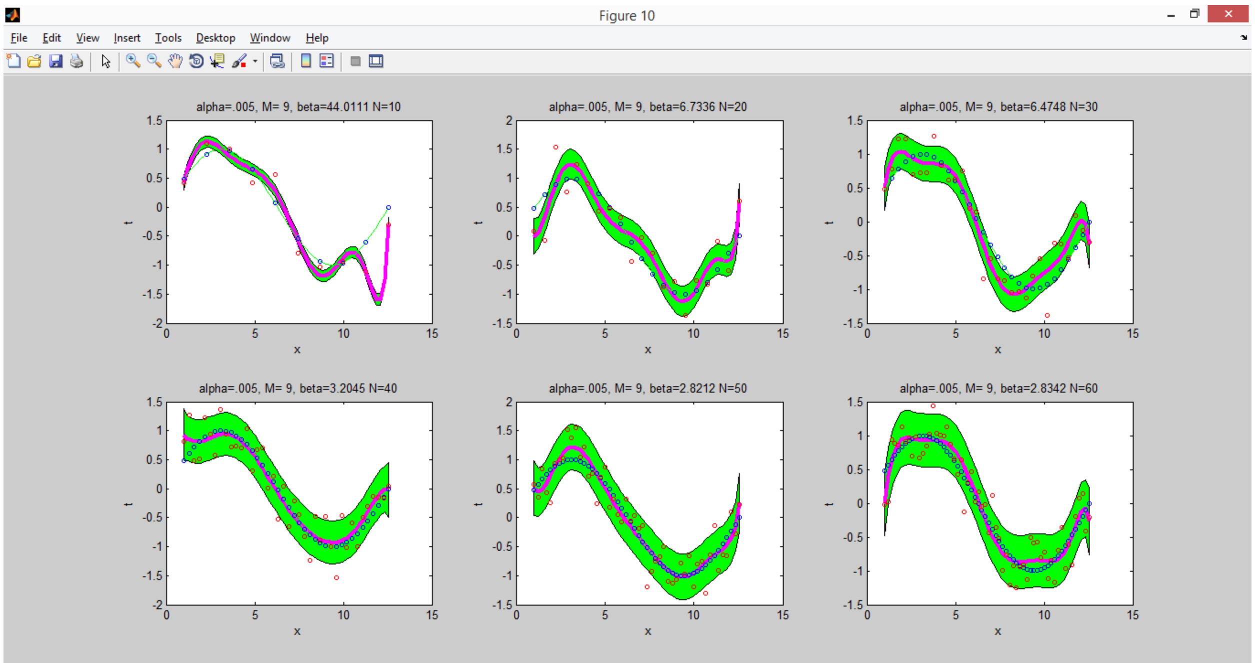
In this part the variance is calculated for each estimated point. So the estimation is much better compare to part 3 and also it less likely to be overfit as can be seen for M=9 case.



Corresponding W values-

M ->	0	1	2	3	4	5	6	7	8	9
w0	-0.06594	1.129548	1.382514	-0.36687	-0.53299	-0.52475	-0.56005	-2.47569	-0.87915	-0.52839
w1		-0.1763	-0.28219	1.06572	1.255557	1.242029	1.303535	5.416392	0.379735	0.257303
w2			0.007804	-0.23332	-0.2916	-0.28514	-0.32032	-3.43259	2.278188	1.47048
w3				0.011854	0.018383	0.017131	0.026237	1.149061	-1.90878	-0.9986
w4					-0.00024	-0.00014	-0.00131	-0.21669	0.65972	0.241673
w5						-3.17E-06	7.06E-05	0.022569	-0.11959	-0.01707
w6							-1.79E-06	-0.00121	0.011852	-0.00266
w7								2.60E-05	-6.07E-04	0.000581
w8									1.25E-05	-3.97E-05
w9										9.54E-07

In the next figure, different N values are used. And as a general observation the higher number of data points tends to reduce the chances of the overfit. Here the prediction in case of N=60 is much more generalized than the case of N=10(not overfit even for M=9).



Corresponding w values –

N ->	10	20	30	40	50	60
w0	-0.52839	1.392984	-2.29113	1.29476	3.144406	-3.52085
w1	0.257303	-2.59319	3.691887	-0.14451	-5.35867	4.544831
w2	1.47048	0.437374	-0.01983	-0.63709	3.348044	-0.09541
w3	-0.9986	1.541562	-1.5801	0.59506	-0.515	-1.64726
w4	0.241673	-1.03328	0.893093	-0.2135	-0.17733	0.915607
w5	-0.01707	0.290876	-0.23267	0.038963	0.083774	-0.23583
w6	-0.00266	-0.04391	0.033387	-0.00397	-0.01445	0.033662
w7	0.000581	0.003712	-0.00271	0.000224	0.001286	-0.00273
w8	-3.97E-05	-1.66E-04	1.17E-04	-6.38E-06	-5.87E-05	1.18E-04
w9	9.54E-07	3.05E-06	-2.09E-06	6.48E-08	1.09E-06	-2.11E-06

MATLAB CODE –

```
%% start code for project #1: linear regression
%pattern recognition, CSE583/EE552
%Weina Ge, Aug 2008

% imp note - only y_predit is extra plotted, rest of the plots have
% the same meaning as given, only the blue plot was tuned in green
% for better viewing
clear
close all

%load the data points
load data10.mat

%plot the groud truth curve
figure(1)
xx = linspace(1,4*pi,50);
yy = sin(.5*xx);
plot(xx,yy,'g-');
hold on
plot(x,y,'o','MarkerSize',3);
%plot the noisy observations
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')

%% PART 1 curve fit for M-5

M=5;
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
w=inv(X'*X)*X'*t';
y_predict = X_plot*w ;
plot(xx,y_predict,'m-');
title('M=5,curve fitting')



%% PART 1 curve fit for different Ms
figure(2)
W=[];
for M=0:9
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% calculation of the w vector as given in notes
w=inv(X'*X)*X'*t';
```

```
% storing w for the report
W=[W,[w;NaN(9-M,1)]];
y_predict = X_plot*w ;
%plots
subplot(2,5,M+1)
plot(xx,y_predict,'m-','LineWidth',3);
hold on
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['N=10, M=',num2str(M)])
end
W2=W;

%% PART 1 curve fit for different data set
figure(3)
M=9;
W=[];
for k =1:6
npts = 10*k;
name=['data',num2str(npts),'.mat'];% loading diffenernt data file
load(name)
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% calculation of w as given in notes
w=inv(X'*X)*X'*t';
W=[W,[w;NaN(9-M,1)]];  % storing w
y_predict = X_plot*w ; % rediction using w
%plots
subplot(2,3,k)
plot(xx,y_predict,'m-','LineWidth',3);
hold on
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['N=',num2str(npts),', M=9'])


end
W3=W;

%% PART 2 add regularization

lambda=exp(-1);
load data10.mat
figure(4)
W=[];
for M=0:9
```

```
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% calculation of w as given in the notes
w=inv(X'*X+lambda*eye(M+1))*X'*t';
W=[W,[w;NaN(9-M,1)]];% storing w
y_predict = X_plot*w ; % y prediction
%plots
subplot(2,5,M+1)
plot(xx,y_predict,'m-','LineWidth',3);
hold on
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['ln lambda = -1, M=',num2str(M)])
end
W4=W;



%% PART 2 same M, same data, Different lambda

M=5;
load data10.mat
figure(5)
W=[];
L=[exp(-18),exp(-10),exp(-5),exp(0),exp(5),exp(10)]; % differnt lambda values
for k=1:6
lambda=L(k);
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% calculation of w as given in notes
w=inv(X'*X+lambda*eye(M+1))*X'*t';
W=[W,[w;NaN(9-M,1)]]; % storing w
y_predict = X_plot*w ;  % predicting y
%plots
subplot(2,3,k)
plot(xx,y_predict,'m-','LineWidth',3);
hold on
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
%plot the noisy observations
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['M=5, lambda=',num2str(L(k))])
end
W5=W;
```

```matlab
%% PART 2, M =9 lambda=exp(-10), Different Data Points

M=9;
W=[];
figure(6)
lambda=exp(-10);
for k =1:6
npts = 10*k;
name=['data',num2str(npts),'.mat'];
load(name)
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% calculation of w as given in notes
w=inv(X'*X+lambda*eye(M+1))*X'*t';
W=[W,[w;NaN(9-M,1)]]; % string w
y_predict = X_plot*w ;  % y prediction
%plots
subplot(2,3,k)
plot(xx,y_predict,'m-','LineWidth',3);
hold on
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['M=9, ln lambda=-10, N=',num2str(npts)])
end
W6=W;

%% PART 3 different M

load data10.mat
figure(7)
W=[];
for M=0:9
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% clculation of we as per notes
w=inv(X'*X)*X'*t';
W=[W,[w;NaN(9-M,1)]]; % storing w
y_predict = X_plot*w; % prediction of y
beta=10/((t'-X*w)'*(t'-X*w));  % calculation of beta as given
% using the beta, variance is calcuted and 95 percent
% confidence interval is plot
% higher and lower limit of y_prict is calcluted using the
% 95 percent confdence interval method
y_predict_u= X_plot*w+1.96*sqrt(1/(beta*10)); % upper bound of 95 CI
```

```
y_predict_l= X_plot*w-1.96*sqrt(1/(beta*10)); % lower bound of 95 CI
%plots
subplot(2,5,M+1)
ciplot(y_predict_l,y_predict_u,xx,[0 1 0])
hold on
plot(xx,y_predict,'m-','LineWidth',3);
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['N=10, M=',num2str(M)])
end
W7=W;

%% PART 3 Different data points

figure(8)
W=[];
M=9;
for k =1:6
npts = 10*k;
name=['data',num2str(npts),'.mat']; % loaading different data files
load(name)
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% calculation of w as given in the notes
w=inv(X'*X)*X'*t';
W=[W,[w;NaN(9-M,1)]]; % storing w
y_predict = X_plot*w ;  % y prediction
beta=10/((t'-X*w)'*(t'-X*w)); % beta calculation as given
y_predict_u= X_plot*w+1.96*sqrt(1/(beta*npts)); % upper bound of 95 CI
y_predict_l= X_plot*w-1.96*sqrt(1/(beta*npts)); % lower bound of 95 CI
%plots
subplot(2,3,k)
ciplot(y_predict_l,y_predict_u,xx,[0 1 0])
hold on
plot(xx,y_predict,'m-','LineWidth',3);
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['M=9, N=',num2str(npts)])
end
W8=W;

%% PART 4 different M
load data10.mat
figure(9)
W=[];
beta_v=linspace(.9,8,10); % assumed beta values
```

```
alpha=.005;
for M=0:9
X=[];
X_plot=[];
beta=beta_v(M+1);
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
% w calcuation according the method given in notes
w=beta*inv(beta*X'*X+alpha*eye(M+1))*X'*t';
W=[W,[w;NaN(9-M,1)]]; % storing the w values
y_predict = X_plot*w; % y prediction
% calculation of S inverse
S_inv=alpha*eye(M+1)+beta*X_plot'*X_plot;
S=inv(S_inv); % calculation S
% calculation of sigma square
sigma_sq=(1/beta)+diag(X_plot*S*X_plot');
% using the sigma square 95 percent confidence interval is calculated
y_predict_u= X_plot*w+1.96*sqrt(sigma_sq/10); % Upper limit of 95 CI
y_predict_l= X_plot*w-1.96*sqrt(sigma_sq/10); % Lower limit of 95 CI
% plots
subplot(2,5,M+1)
ciplot(y_predict_l,y_predict_u,xx,[0 1 0])
hold on
plot(xx,y_predict,'m-','LineWidth',3);
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['alpha=.005, beta=',num2str(beta),' M=',num2str(M)])
end
W9=W;

%% PART 4 Different data
figure(10)
M=9;
W=[];
beta_v=linspace(.9,8,10); % assumed beta values
alpha=.005;
for k =1:6
npts = 10*k;
name=['data',num2str(npts),'.mat'];
load(name)
X=[];
X_plot=[];
for i=0:M
    X=[X x'.^i];
    X_plot=[X_plot xx'.^i];
end
beta=beta_v(M+1);
% w calcuation according the method given in notes
w=beta*inv(beta*X'*X+alpha*eye(M+1))*X'*t';
W=[W,[w;NaN(9-M,1)]]; % storing the w values
y_predict = X_plot*w ; % y prediction
beta=10/((t'-X*w)'*(t'-X*w));
```

```
% calculation of S inverse
S_inv=alpha*eye(M+1)+beta*X_plot'*X_plot;
S=inv(S_inv); % calculation S
% calculation of sigma square
sigma_sq=(1/beta)+diag(X_plot*S*X_plot');
% using the sigma square 95 percent confidence interval is calculated
y_predict_u= X_plot*w+1.96*sqrt(sigma_sq/10); % Upper limit of 95 CI
y_predict_l= X_plot*w-1.96*sqrt(sigma_sq/10); % Lower limit of 95 CI
% plots
subplot(2,3,k)
ciplot(y_predict_l,y_predict_u,xx,[0 1 0])
hold on
plot(xx,y_predict,'m-','LineWidth',3);
plot(xx,yy,'g-');
plot(x,y,'o','MarkerSize',3);
plot(x,t,'ro','MarkerSize',3);
xlabel('x')
ylabel('t')
title(['alpha=.005, M= 9, beta=',num2str(beta),' N=',num2str(npts)])
end
W10=W;

%%
```