**TEST 3 REGULARIZED LINEAR REGRESSION**

Patel, Femina ID: N03094115

**Answer 1:**

function v = LinRegRegularized(xData,yData,x\_vals,n,lambda)

m = length(xData);

m2 = length(x\_vals);

for i=1:m

for j=0:n

X(i,j+1) = xData(i)^j;

end

end

dg=eye(n+1);

dg(1,1)=0;

theta= pinv((X' \* X) + (lambda .\* dg)) \* (X' \* yData);for i=1:m2

for j=0:n

XINPUT(i,j+1) = x\_vals(i)^j;

end

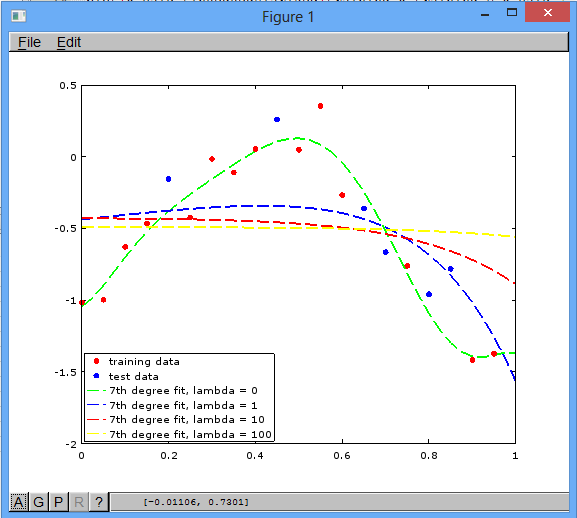
end

v = theta' \* XINPUT';

end

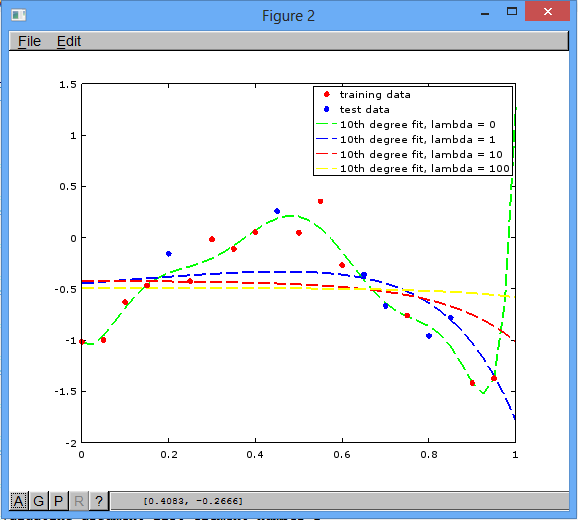
**Answer 2(a):**

**Regression Curve for degree 7 polynomial for lambda = 0, 1, 10, 100**



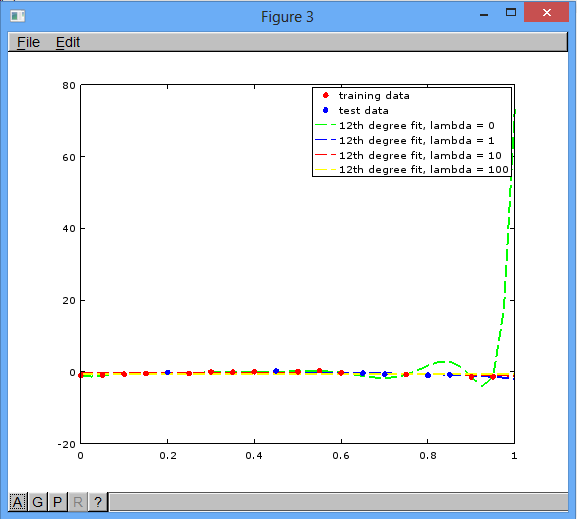
**Answer 2(b):**

**Regression Curve for degree 10 polynomial for lambda = 0, 1, 10, 100**



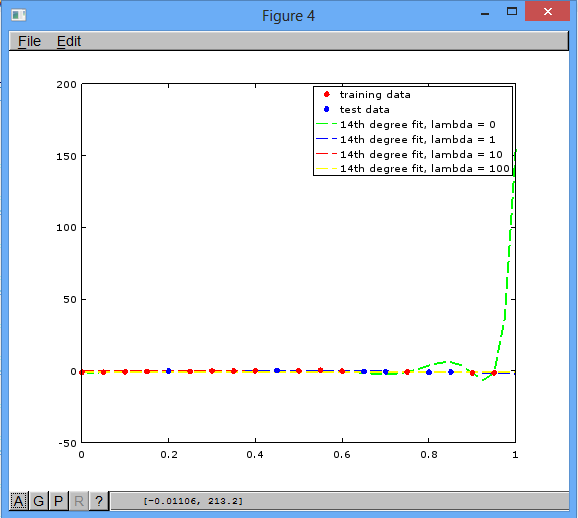
**Answer 2(c):**

**Regression Curve for degree 12 polynomial for lambda = 0, 1, 10, 100**



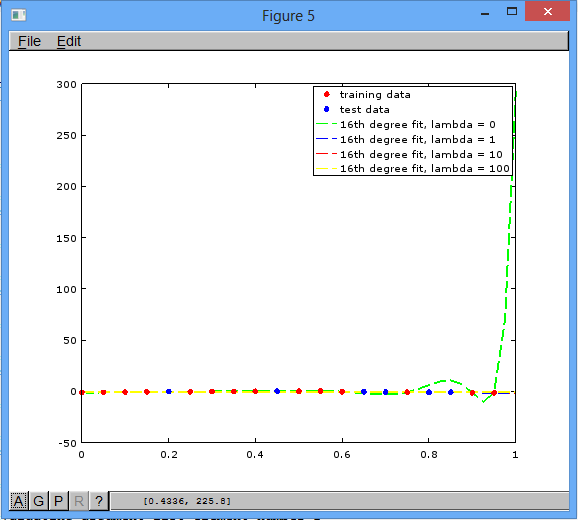
**Answer 2(d):**

**Regression Curve for degree 14 polynomial for lambda = 0, 1, 10, 100**

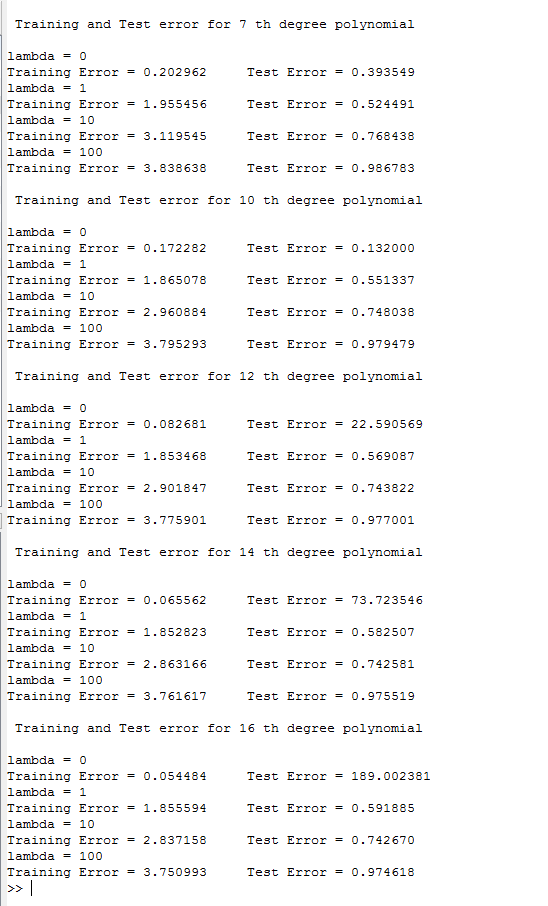


**Answer 2(e):**

**Regression Curve for degree 16 polynomial for lambda = 0, 1, 10, 100**



**Answer 3:**



**Answer 4:**

Overfitting occurs when the model or the algorithm fits the data too well. Higher the degree of polynomial, more is the overfitting because it can cover most of the training data points in the model.

While applying regularization, as we increase the value of lambda, there will be an increase in the cost function of the training data. For lower degree of polynomial, cost function for test data will also increase with increase in lambda values. For higher degree of polynomial, with increase in lambda values, there will be decrease in the cost function of the test data to a great extent and after that with increase in lambda again the value of cost function for test data will increase gradually.

**Patel\_Exercises.m**

training\_data=load('trainData.txt');

test\_data=load('testData.txt');

training\_x=training\_data(:,1);

training\_y=training\_data(:,2);

test\_x=test\_data(:,1);

test\_y=test\_data(:,2);

polynomial =[7; 10; 12; 14; 16];

lambda = [0; 1; 10; 100];

x\_vals\_train = training\_x;

x\_vals\_test = test\_x;

figure;

plot(training\_x,training\_y,'ro','markersize',5,'markerfacecolor','red');

hold on;

plot(test\_x,test\_y,'bo','markersize',5,'markerfacecolor','blue');

x\_vals=(0:0.025:1)';

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,7,0),'g','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,7,1),'b','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,7,10),'r','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,7,100),'y','linestyle','--','linewidth',2);

legend('training data','test data','7th degree fit, lambda = 0','7th degree fit, lambda = 1','7th degree fit, lambda = 10','7th degree fit, lambda = 100','location','southwest');

figure;

plot(training\_x,training\_y,'ro','markersize',5,'markerfacecolor','red');

hold on;

plot(test\_x,test\_y,'bo','markersize',5,'markerfacecolor','blue');

x\_vals=(0:0.025:1)';

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,10,0),'g','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,10,1),'b','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,10,10),'r','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,10,100),'y','linestyle','--','linewidth',2);

legend('training data','test data','10th degree fit, lambda = 0','10th degree fit, lambda = 1','10th degree fit, lambda = 10','10th degree fit, lambda = 100');

figure;

plot(training\_x,training\_y,'ro','markersize',5,'markerfacecolor','red');

hold on;

plot(test\_x,test\_y,'bo','markersize',5,'markerfacecolor','blue');

x\_vals=(0:0.025:1)';

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,12,0),'g','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,12,1),'b','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,12,10),'r','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,12,100),'y','linestyle','--','linewidth',2);

legend('training data','test data','12th degree fit, lambda = 0','12th degree fit, lambda = 1','12th degree fit, lambda = 10','12th degree fit, lambda = 100');

figure;

plot(training\_x,training\_y,'ro','markersize',5,'markerfacecolor','red');

hold on;

plot(test\_x,test\_y,'bo','markersize',5,'markerfacecolor','blue');

x\_vals=(0:0.025:1)';

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,14,0),'g','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,14,1),'b','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,14,10),'r','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,14,100),'y','linestyle','--','linewidth',2);

legend('training data','test data','14th degree fit, lambda = 0','14th degree fit, lambda = 1','14th degree fit, lambda = 10','14th degree fit, lambda = 100');

figure;

plot(training\_x,training\_y,'ro','markersize',5,'markerfacecolor','red');

hold on;

plot(test\_x,test\_y,'bo','markersize',5,'markerfacecolor','blue');

x\_vals=(0:0.025:1)';

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,16,0),'g','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,16,1),'b','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,16,10),'r','linestyle','--','linewidth',2);

plot(x\_vals,Patel\_LinRegRegularized(training\_x,training\_y,x\_vals,16,100),'y','linestyle','--','linewidth',2);

legend('training data','test data','16th degree fit, lambda = 0','16th degree fit, lambda = 1','16th degree fit, lambda = 10','16th degree fit, lambda = 100');%}

for n = 1:length(polynomial)

printf("\n Training and Test error for %d th degree polynomial \n\n",polynomial(n));

for l = 1:length(lambda)

printf("lambda = %d \n",lambda(l));

[predict\_training] = Patel\_LinRegRegularized(training\_x, training\_y, x\_vals\_train,polynomial(n),lambda(l));

train\_error = 0;

for i = 1:length(x\_vals\_train)

train\_error = train\_error + (training\_y(i) - predict\_training(i))^2;

end

[predict\_test] = Patel\_LinRegRegularized(training\_x, training\_y, x\_vals\_test,polynomial(n),lambda(l));

test\_error = 0;

for j = 1:length(x\_vals\_test)

test\_error = test\_error + (test\_y(j) - predict\_test(j))^2;

end

printf("Training Error = %f",train\_error);

printf(" Test Error = %f ",test\_error);

disp("");

end

end