Machine Learning: Programming Exercise 5

Regularized Linear Regression and Bias vs. Variance

In this exercise, you will implement regularized linear regression and use it to study models with different bias-variance properties.

Files needed for this exercise

- `ex5.mlx` - MATLAB Live Script that steps you through the exercise
- `ex5data1.mat` - Dataset
- `submit.m` - Submission script that sends your solutions to our servers
- `featureNormalize.m` - Feature normalization function
- `fmincg.m` - Function minimization routine (similar to `fminunc`)
- `plotFit.m` - Plot a polynomial fit
- `trainLinearReg.m` - Trains linear regression using your cost function
- *`linearRegCostFunction.m` - Regularized linear regression cost function
- `*learningCurve.m` - Generates a learning curve
- *`polyFeatures.m` - Maps data into polynomial feature space
- *`validationCurve.m` - Generates a cross validation curve

Table of Contents

1. Regularized Linear Regression

In the first half of the exercise, you will implement regularized linear regression to predict the amount of water flowing out of a dam using the change of water level in a reservoir. In the next half, you will go through some diagnostics of debugging learning algorithms and examine the effects of bias vs. variance.

1.1 Visualizing the dataset

We will begin by visualizing the dataset containing historical records on the change in the water level, x, and the amount of water flowing out of the dam, y. This dataset is divided into three parts:

- A **training** set that your model will learn on: `X, y`

- A **cross validation** set for determining the regularization parameter: `Xval, yval`
- A **test** set for evaluating performance. These are 'unseen' examples which your model did not see during training: `Xtest, ytest`

The code below will plot the training data (Figure 1). In the following parts, you will implement linear regression and use that to fit a straight line to the data and plot learning curves. Following that, you will implement polynomial regression to find a better fit to the data.

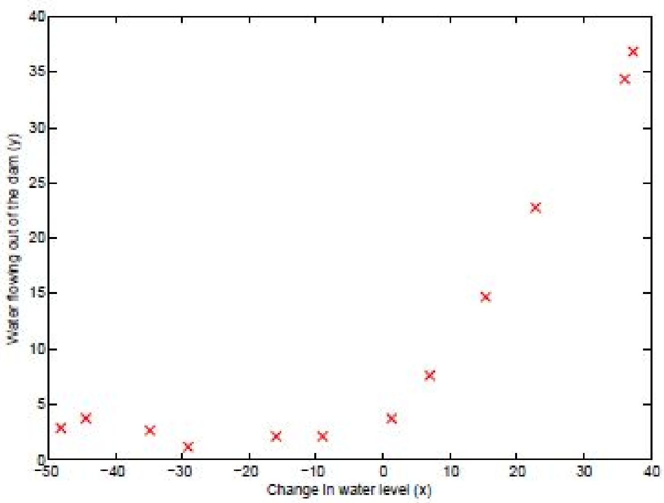
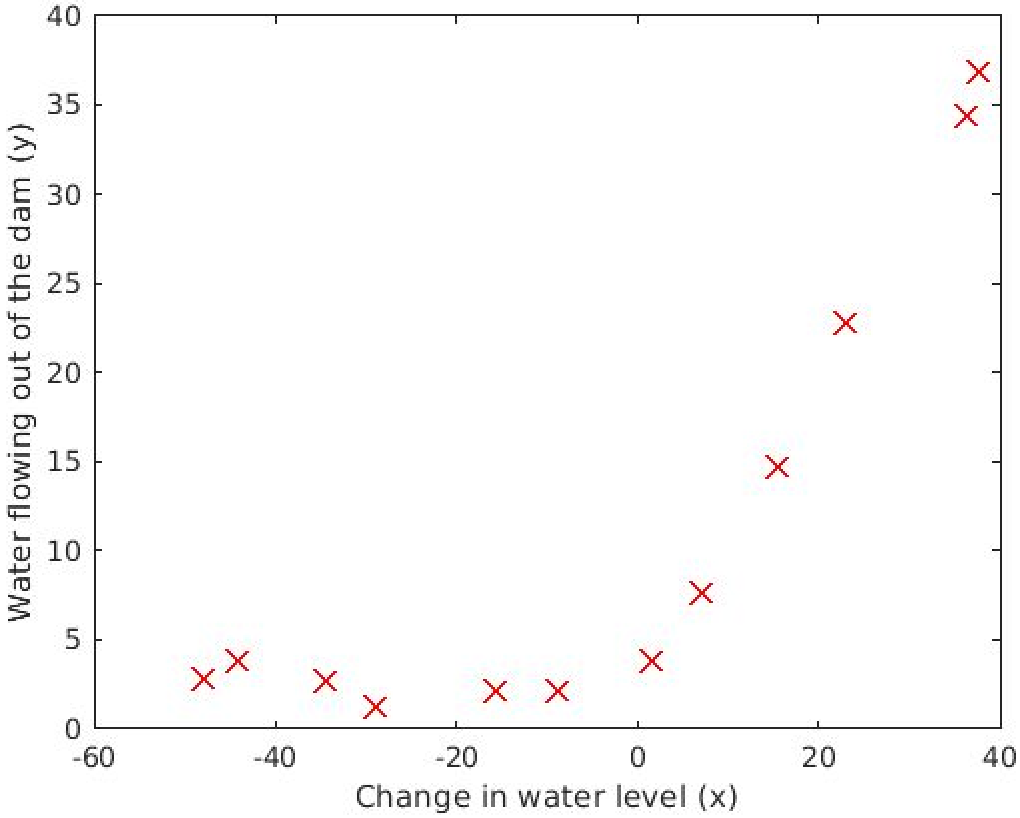


Figure 1: Data

```
% Load from ex5data1:
% You will have X, y, Xval, yval, Xtest, ytest in your environment
load ('ex5data1.mat');
% m = Number of examples
m = size(X, 1);
```

```
% Plot training data
figure;
plot(X, y, 'rx', 'MarkerSize', 10, 'LineWidth', 1.5);
xlabel('Change in water level (x)');
ylabel('Water flowing out of the dam (y)');
```



1.2 Regularized linear regression cost function

Recall that regularized linear regression has the following cost function:

$$J(\theta) = \frac{1}{2m}\left(\sum\nolimits_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2\right) + \frac{\lambda}{2m}\left(\sum\nolimits_{j=1}^{n} \theta_j^2\right)$$

where λ is a regularization parameter which controls the degree of regularization (thus, helps preventing overtting). The regularization term puts a penalty on the overall cost J. As the magnitudes of the model parameters θ_j increase, the penalty increases as well. Note that you should not regularize the θ_0 term. (In MATLAB, the θ_0 term is represented as `theta(1)` since indexing in MATLAB starts from 1).

You should now complete the code in the file `linearRegCostFunction.m`. Your task is to write a function to calculate the regularized linear regression cost function. If possible, try to vectorize your code and avoid writing loops. When you are finished, the code below will run your cost function using `theta` initialized at `[1; 1]`. You should expect to see an output of 303.993.

```
theta = [1 ; 1];
```

```
J = linearRegCostFunction([ones(m, 1) X], y, theta, 1);
fprintf('Cost at theta = [1 ; 1]: %f', J);
```

```
Cost at theta = [1 ; 1]: 303.993192
```

You should now submit your solutions. Enter **`submit`** *at the command prompt, then enter or confirm your login and token when prompted.*

1.3 Regularized linear regression gradient

Correspondingly, the partial derivative of regularized linear regression's cost for θ_j is defined as

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)} \quad \text{for} \quad j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \quad \text{for} \quad j = 0$$

In `linearRegCostFunction.m`, add code to calculate the gradient, returning it in the variable `grad`. When you are finished, the code below will run your gradient function using theta initialized at [1; 1]. You should expect to see a gradient of [-15.30; 598.250].

```
[J, grad] = linearRegCostFunction([ones(m, 1) X], y, theta, 1);
fprintf('Gradient at theta = [1 ; 1]:  [%f; %f] \n',grad(1), grad(2));
```

```
Gradient at theta = [1 ; 1]:  [-15.303016; 598.250744]
```

You should now submit your solutions. Enter **`submit`** *at the command prompt, then enter or confirm your login and token when prompted.*

1.4 Fitting linear regression

Once your cost function and gradient are working correctly, the code in this section will run the code in `trainLinearReg.m` to compute the optimal values of θ. This training function uses `fmincg` to optimize the cost function. In this part, we set regularization parameter λ to zero. Because our current implementation of linear regression is trying to fit a 2-dimensional θ, regularization will not be incredibly helpful for a θ of such low dimension. In the later parts of the exercise, you will be using polynomial regression with regularization.

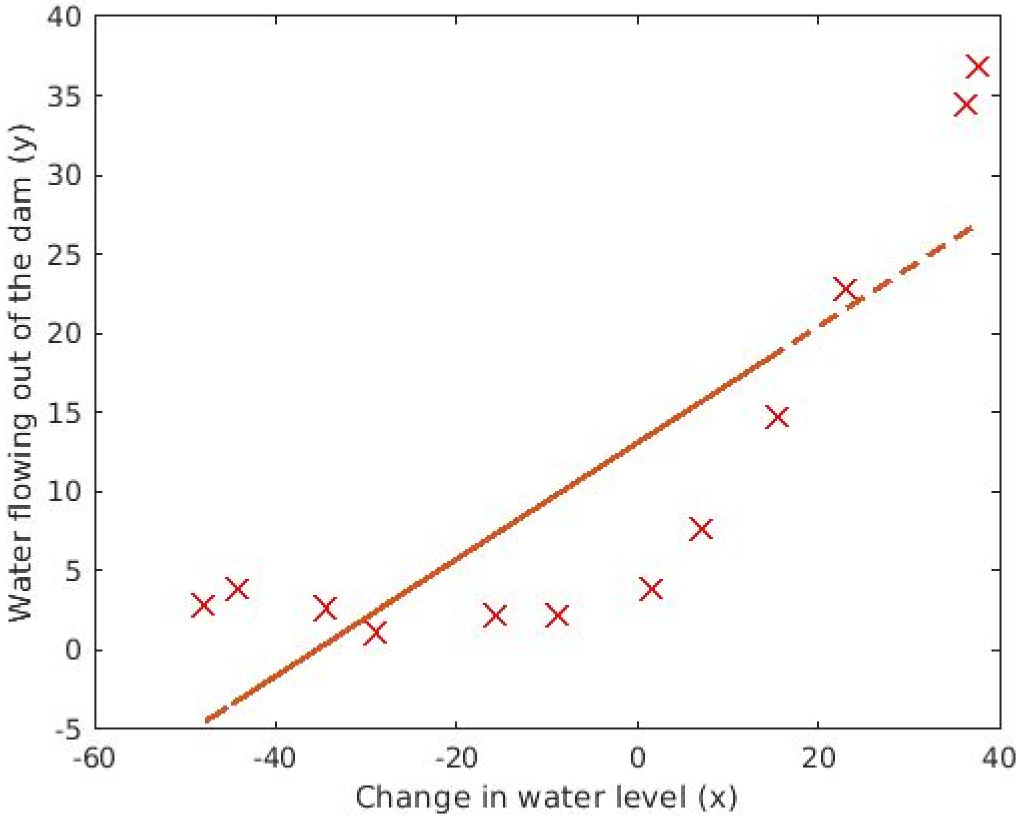
```
%  Train linear regression with lambda = 0
```

```
lambda = 0;
[theta] = trainLinearReg([ones(m, 1) X], y, lambda);
```

```
Iteration     1 | Cost: 1.052435e+02
Iteration     2 | Cost: 2.237391e+01
Iteration     3 | Cost: 2.237391e+01
Iteration     4 | Cost: 2.237391e+01
Iteration     5 | Cost: 2.237391e+01
Iteration     6 | Cost: 2.237391e+01
```

Finally, the code below should also plot the best fit line, resulting in an image similar to Figure 2. The best fit line tells us that the model is not a good fit to the data because the data has a nonlinear pattern.

```
%  Plot fit over the data
figure;
plot(X, y, 'rx', 'MarkerSize', 10, 'LineWidth', 1.5);
xlabel('Change in water level (x)');
ylabel('Water flowing out of the dam (y)');
hold on;
plot(X, [ones(m, 1) X]*theta, '--', 'LineWidth', 2)
hold off;
```



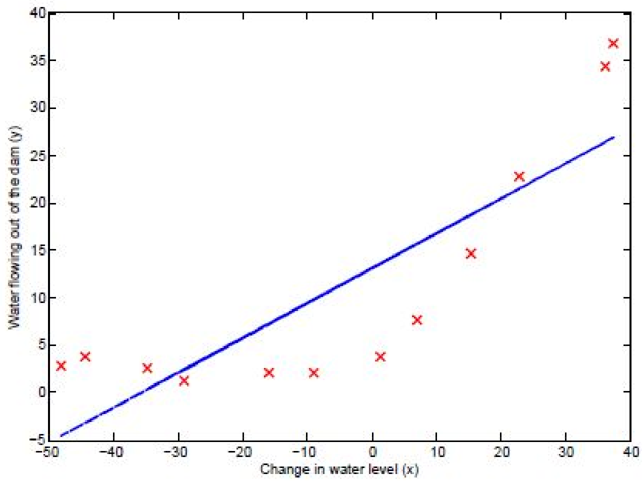


Figure 2: Linear Fit

While visualizing the best fit as shown is one possible way to debug your learning algorithm, it is not always easy to visualize the data and model. In the next section, you will implement a function to generate learning curves that can help you debug your learning algorithm even if it is not easy to visualize the data.

2. Bias-variance

An important concept in machine learning is the bias-variance tradeoff. Models with high bias are not complex enough for the data and tend to underfit, while models with high variance overfit the training data. In this part of the exercise, you will plot training and test errors on a learning curve to diagnose bias-variance problems.

2.1 Learning curves

You will now implement code to generate the learning curves that will be useful in debugging learning algorithms. Recall that a learning curve plots training and cross validation error as a function of training set size. Your job is to fill in `learningCurve.m` so that it returns a vector of errors for the training set and cross validation set.

To plot the learning curve, we need a training and cross validation set error for different training set sizes. To obtain different training set sizes, you should use different subsets of the original training set `X`. Specically, for a training set size of `i`, you should use the first `i` examples (i.e., `X(1:i,:)` and `y(1:i))`. You can use the `trainLinearReg` function to find the θ parameters. Note that `lambda` is passed as a parameter to the `learningCurve` function. After learning the θ parameters, you should compute the error on the training and cross validation sets. Recall that the training error for a dataset is defined as

$$J_{\text{train}}(\theta) = \frac{1}{2m}\left[\sum\nolimits_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2\right]$$

In particular, note that the training error does not include the regularization term. One way to compute the training error is to use your existing cost function and set λ to 0 only when using it to compute the training error and cross validation error. When you are computing the training set error, make sure you compute it on the training subset (i.e., `X(1:n,:)` and `y(1:n)`, instead of the entire training set). However, for the cross validation error, you should compute it over the entire cross validation set. You should store the computed errors in the vectors `error_train` and `error_val`.

In Figure 3, you can observe that both the train error and cross validation error are high when the number of training examples is increased. This reflects a high bias problem in the model - the linear regression model is too simple and is unable to fit our dataset well. In the next section, you will implement polynomial regression to fit a better model for this dataset.

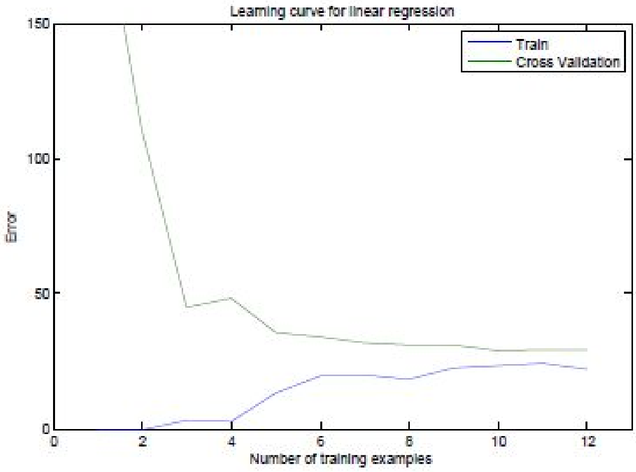


Figure 3: Linear regression learning curve

When you are finished, run the code below to compute the learning curves and produce a plot similar to Figure 3.

```
lambda = 0;

[error_train, error_val] = learningCurve([ones(m, 1) X], y,
[ones(size(Xval, 1), 1) Xval], yval, lambda);
```

```
Iteration     1 | Cost: 2.663868e-01

Iteration     2 | Cost: 3.944305e-31

Iteration     3 | Cost: 0.000000e+00

Iteration     1 | Cost: 4.282328e-01

Iteration     2 | Cost: 8.295365e-30

Iteration     3 | Cost: 4.930381e-32

Iteration     1 | Cost: 1.021540e+02

Iteration     2 | Cost: 3.286595e+00

Iteration     1 | Cost: 1.438726e+02

Iteration     2 | Cost: 1.035224e+02

Iteration     3 | Cost: 7.536716e+01

Iteration     4 | Cost: 1.615422e+01

Iteration     5 | Cost: 3.619255e+00

Iteration     6 | Cost: 2.842916e+00

Iteration     7 | Cost: 2.842916e+00

Iteration     8 | Cost: 2.842770e+00

Iteration     9 | Cost: 2.842731e+00
```

```
Iteration    10 | Cost: 2.842729e+00
Iteration    11 | Cost: 2.842678e+00
Iteration     1 | Cost: 1.592641e+02
Iteration     2 | Cost: 2.404966e+01
Iteration     3 | Cost: 2.354137e+01
Iteration     4 | Cost: 2.281160e+01
Iteration     5 | Cost: 2.276969e+01
Iteration     6 | Cost: 2.224060e+01
Iteration     7 | Cost: 1.920606e+01
Iteration     8 | Cost: 1.475292e+01
Iteration     9 | Cost: 1.430565e+01
Iteration    10 | Cost: 1.388881e+01
Iteration    11 | Cost: 1.327330e+01
Iteration    12 | Cost: 1.323519e+01
Iteration    13 | Cost: 1.319411e+01
Iteration    14 | Cost: 1.317355e+01
Iteration    15 | Cost: 1.315411e+01
Iteration    16 | Cost: 1.315405e+01
Iteration    17 | Cost: 1.315405e+01
Iteration    18 | Cost: 1.315405e+01
Iteration    19 | Cost: 1.315405e+01
Iteration    20 | Cost: 1.315405e+01
Iteration    21 | Cost: 1.315405e+01
Iteration    22 | Cost: 1.315405e+01
Iteration     1 | Cost: 1.531141e+02
Iteration     2 | Cost: 1.350947e+02
Iteration     3 | Cost: 1.137334e+02
Iteration     4 | Cost: 4.404199e+01
Iteration     5 | Cost: 2.957435e+01
Iteration     6 | Cost: 2.719561e+01
Iteration     7 | Cost: 1.949283e+01
Iteration     8 | Cost: 1.949128e+01
Iteration     9 | Cost: 1.945708e+01
Iteration    10 | Cost: 1.944472e+01
```

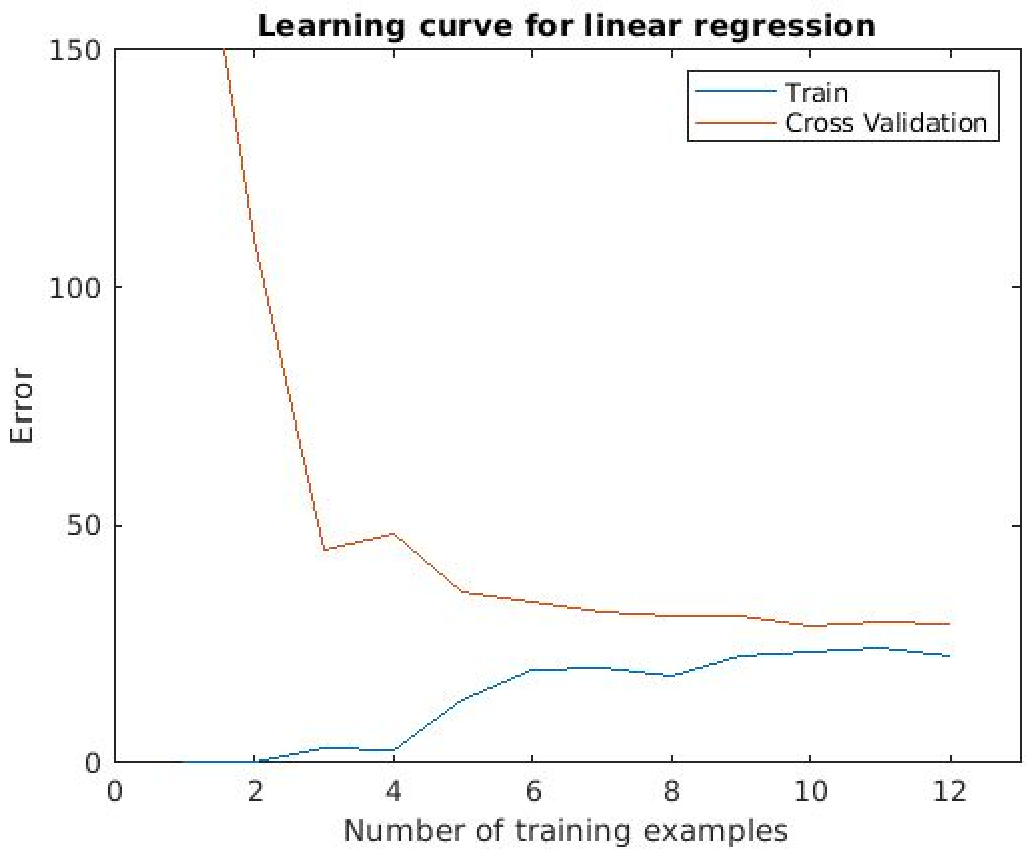
```
Iteration    11 | Cost: 1.944472e+01
Iteration    12 | Cost: 1.944471e+01
Iteration    13 | Cost: 1.944453e+01
Iteration    14 | Cost: 1.944425e+01
Iteration    15 | Cost: 1.944411e+01
Iteration    16 | Cost: 1.944396e+01
Iteration    17 | Cost: 1.944396e+01
Iteration    18 | Cost: 1.944396e+01
Iteration    19 | Cost: 1.944396e+01
Iteration    20 | Cost: 1.944396e+01
Iteration    21 | Cost: 1.944396e+01
Iteration    22 | Cost: 1.944396e+01
Iteration    23 | Cost: 1.944396e+01
Iteration    24 | Cost: 1.944396e+01
Iteration    25 | Cost: 1.944396e+01
Iteration    26 | Cost: 1.944396e+01
Iteration    27 | Cost: 1.944396e+01
Iteration    28 | Cost: 1.944396e+01
Iteration    29 | Cost: 1.944396e+01
Iteration    30 | Cost: 1.944396e+01
Iteration    31 | Cost: 1.944396e+01
Iteration    32 | Cost: 1.944396e+01
Iteration    33 | Cost: 1.944396e+01
Iteration    34 | Cost: 1.944396e+01
Iteration    35 | Cost: 1.944396e+01
Iteration     1 | Cost: 1.383936e+02
Iteration     2 | Cost: 1.210275e+02
Iteration     3 | Cost: 1.013004e+02
Iteration     4 | Cost: 3.457729e+01
Iteration     5 | Cost: 2.808710e+01
Iteration     6 | Cost: 2.732288e+01
Iteration     7 | Cost: 2.011513e+01
Iteration     8 | Cost: 2.011508e+01
Iteration     9 | Cost: 2.010693e+01
```

```
Iteration    10 | Cost: 2.010640e+01
Iteration    11 | Cost: 2.010629e+01
Iteration    12 | Cost: 2.010382e+01
Iteration    13 | Cost: 2.009852e+01
Iteration    15 | Cost: 2.009852e+01
Iteration     1 | Cost: 1.237772e+02
Iteration     2 | Cost: 1.202532e+02
Iteration     3 | Cost: 1.195134e+02
Iteration     4 | Cost: 9.334231e+01
Iteration     5 | Cost: 4.813526e+01
Iteration     6 | Cost: 2.677826e+01
Iteration     7 | Cost: 1.953374e+01
Iteration     8 | Cost: 1.831672e+01
Iteration     9 | Cost: 1.817286e+01
Iteration    11 | Cost: 1.817286e+01
Iteration     1 | Cost: 1.089984e+02
Iteration     2 | Cost: 1.064701e+02
Iteration     3 | Cost: 1.054742e+02
Iteration     4 | Cost: 2.266786e+01
Iteration     5 | Cost: 2.266786e+01
Iteration     6 | Cost: 2.266758e+01
Iteration     7 | Cost: 2.260941e+01
Iteration     8 | Cost: 2.260941e+01
Iteration     9 | Cost: 2.260941e+01
Iteration    10 | Cost: 2.260941e+01
Iteration    11 | Cost: 2.260941e+01
Iteration     1 | Cost: 1.108611e+02
Iteration     2 | Cost: 2.497543e+01
Iteration     3 | Cost: 2.496421e+01
Iteration     4 | Cost: 2.494838e+01
Iteration     5 | Cost: 2.493176e+01
Iteration     6 | Cost: 2.490653e+01
Iteration     7 | Cost: 2.474414e+01
Iteration     8 | Cost: 2.326176e+01
```

```
Iteration     9 | Cost: 2.326176e+01
Iteration    10 | Cost: 2.326173e+01
Iteration    11 | Cost: 2.326172e+01
Iteration    12 | Cost: 2.326162e+01
Iteration    13 | Cost: 2.326150e+01
Iteration    14 | Cost: 2.326150e+01
Iteration    15 | Cost: 2.326146e+01
Iteration    16 | Cost: 2.326146e+01
Iteration    17 | Cost: 2.326146e+01
Iteration    18 | Cost: 2.326146e+01
Iteration    19 | Cost: 2.326146e+01
Iteration    20 | Cost: 2.326146e+01
Iteration    21 | Cost: 2.326146e+01
Iteration    22 | Cost: 2.326146e+01
Iteration     1 | Cost: 1.023394e+02
Iteration     2 | Cost: 2.443039e+01
Iteration     3 | Cost: 2.443033e+01
Iteration     4 | Cost: 2.442972e+01
Iteration     5 | Cost: 2.441042e+01
Iteration     6 | Cost: 2.435852e+01
Iteration     7 | Cost: 2.431735e+01
Iteration     8 | Cost: 2.431733e+01
Iteration     9 | Cost: 2.431727e+01
Iteration    10 | Cost: 2.431725e+01
Iteration    12 | Cost: 2.431725e+01
Iteration    13 | Cost: 2.431725e+01
Iteration     1 | Cost: 1.052435e+02
Iteration     2 | Cost: 2.237391e+01
Iteration     3 | Cost: 2.237391e+01
Iteration     4 | Cost: 2.237391e+01
Iteration     5 | Cost: 2.237391e+01
Iteration     6 | Cost: 2.237391e+01
```

```
plot(1:m, error_train, 1:m, error_val);
```

```
title('Learning curve for linear regression')
legend('Train', 'Cross Validation')
xlabel('Number of training examples')
ylabel('Error')
axis([0 13 0 150])
```




```
fprintf('# Training Examples\tTrain Error\tCross Validation Error\n');
```

```
# Training Examples Train Error   Cross Validation Error
```

```
for i = 1:m
   fprintf('  \t%d\t\t%f\t%f\n', i, error_train(i), error_val(i));
end
```

```
    1           0.000000     205.121096
    2           0.000000     110.300366
    3           3.286595     45.010231
    4           2.842678     48.368911
```

5	13.154049	35.865165
6	19.443963	33.829962
7	20.098522	31.970986
8	18.172859	30.862446
9	22.609405	31.135998
10	23.261462	28.936207
11	24.317250	29.551432
12	22.373906	29.433818

3. Polynomial regression

The problem with our linear model was that it was too simple for the data and resulted in underfitting (high bias). In this part of the exercise, you will address this problem by adding more features. For use polynomial regression, our hypothesis has the form:

$$\begin{aligned} h_\theta(x) &= \theta_0 + \theta_1 * (waterLevel) + \theta_2 * (waterLevel)^2 + \cdots + \theta_p * (waterLevel)^p \\ &= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_p x_p \end{aligned}$$

Notice that by defining $x_1 = (\text{waterLevel}), x_2 = (\text{waterLevel})^2, \ldots, x_p = (\text{waterLevel})^p$, we obtain a linear regression model where the features are the various powers of the original value (*waterLevel*).

Now, you will add more features using the higher powers of the existing feature x in the dataset. Your task in this part is to complete the code in `polyFeatures.m` so that the function maps the original training set `X` of size $m \times 1$ into its higher powers. Specifically, when a training set X of size $m \times 1$ is passed into the function, the function should return a $m \times p$ matrix `X_poly`, where column 1 holds the original values of `X`, column 2 holds the values of `X.^2`, column 3 holds the values of `X.^3`, and so on. Note that you don't have to account for the zero-th power in this function. Now that you have a function that will map features to a higher dimension, the code in the next section will apply it to the training set, the test set, and the cross validation set (which you haven't used yet).

You should now submit your solutions. Enter **`submit`** *at the command prompt, then enter or confirm your login and token when prompted.*

3.1 Learning Polynomial Regression

After you have completed `polyFeatures.m`, run the code below to train polynomial regression using your linear regression cost function. Keep in mind that even though we have polynomial terms in our feature vector, we are still solving a linear regression optimization problem. The polynomial terms have simply turned into features that we can use for linear regression. We are using the same cost function and gradient that you wrote for the earlier part of this exercise.

For this part of the exercise, you will be using a polynomial of degree 8. It turns out that if we run the training directly on the projected data, it will not work well as the features would be badly scaled (e.g., an example with $x = 40$ will now have a feature $x_8 = 40^8 = 6:5 \times 10^{12}$) Therefore, you will need to use feature normalization. Before learning the parameters θ for the polynomial regression, code in below will first call `featureNormalize` to normalize the features of the training set, storing the `mu`, `sigma` parameters separately. We have already implemented this function for you and it is the same function from the first exercise.

```
p = 8;
```

```
% Map X onto Polynomial Features and Normalize
X_poly = polyFeatures(X, p);
[X_poly, mu, sigma] = featureNormalize(X_poly);  % Normalize
X_poly = [ones(m, 1), X_poly];                   % Add Ones
```

```
% % Map X_poly_test and normalize (using mu and sigma)
X_poly_test = polyFeatures(Xtest, p);
X_poly_test = X_poly_test-mu; % uses implicit expansion instead of bsxfun
X_poly_test = X_poly_test./sigma; % uses implicit expansion instead of bsxfun
X_poly_test = [ones(size(X_poly_test, 1), 1), X_poly_test];         % Add Ones
```

```
% Map X_poly_val and normalize (using mu and sigma)
X_poly_val = polyFeatures(Xval, p);
X_poly_val = X_poly_val-mu; % uses implicit expansion instead of bsxfun
```

```
X_poly_val = X_poly_val./sigma; % uses implicit expansion instead of bsxfun
X_poly_val = [ones(size(X_poly_val, 1), 1), X_poly_val];          % Add Ones
```

```
fprintf('Normalized Training Example 1:\n');
```

```
Normalized Training Example 1:
```

```
fprintf('  %f  \n', X_poly(1, :));
```

```
 1.000000
 -0.362141
 -0.755087
 0.182226
 -0.706190
 0.306618
 -0.590878
 0.344516
 -0.508481
```

```
lambda = 0;
[theta] = trainLinearReg(X_poly, y, lambda);
```

```
Iteration     1 | Cost: 8.273077e+01
Iteration     2 | Cost: 2.687496e+01
Iteration     3 | Cost: 1.327780e+01
Iteration     4 | Cost: 3.455324e+00
Iteration     5 | Cost: 2.870493e+00
Iteration     6 | Cost: 2.404364e+00
Iteration     7 | Cost: 2.372779e+00
Iteration     8 | Cost: 1.771555e+00
Iteration     9 | Cost: 1.210317e+00
Iteration    10 | Cost: 9.412009e-01
Iteration    11 | Cost: 7.612337e-01
Iteration    12 | Cost: 6.958010e-01
Iteration    13 | Cost: 6.271154e-01
Iteration    14 | Cost: 4.960190e-01
```

```
Iteration    15 | Cost: 4.835655e-01
Iteration    16 | Cost: 4.697379e-01
Iteration    17 | Cost: 4.651876e-01
Iteration    18 | Cost: 4.585744e-01
Iteration    19 | Cost: 4.574363e-01
Iteration    20 | Cost: 4.529489e-01
Iteration    21 | Cost: 4.480480e-01
Iteration    22 | Cost: 4.187935e-01
Iteration    23 | Cost: 3.953974e-01
Iteration    24 | Cost: 3.813301e-01
Iteration    25 | Cost: 3.712891e-01
Iteration    26 | Cost: 3.642143e-01
Iteration    27 | Cost: 3.611337e-01
Iteration    28 | Cost: 3.579340e-01
Iteration    29 | Cost: 3.465612e-01
Iteration    30 | Cost: 3.455646e-01
Iteration    31 | Cost: 3.373482e-01
Iteration    32 | Cost: 3.183435e-01
Iteration    33 | Cost: 2.918079e-01
Iteration    34 | Cost: 2.907547e-01
Iteration    35 | Cost: 2.869493e-01
Iteration    36 | Cost: 2.855904e-01
Iteration    37 | Cost: 2.836223e-01
Iteration    38 | Cost: 2.831013e-01
Iteration    39 | Cost: 2.819369e-01
Iteration    40 | Cost: 2.804084e-01
Iteration    41 | Cost: 2.730627e-01
Iteration    42 | Cost: 2.647005e-01
Iteration    43 | Cost: 2.624268e-01
Iteration    44 | Cost: 2.616021e-01
Iteration    45 | Cost: 2.581657e-01
Iteration    46 | Cost: 2.564444e-01
Iteration    47 | Cost: 2.563460e-01
Iteration    48 | Cost: 2.550587e-01
```

```
Iteration    49 | Cost: 2.548502e-01
Iteration    50 | Cost: 2.539370e-01
Iteration    51 | Cost: 2.537053e-01
Iteration    52 | Cost: 2.533873e-01
Iteration    53 | Cost: 2.530916e-01
Iteration    54 | Cost: 2.511142e-01
Iteration    55 | Cost: 2.506830e-01
Iteration    56 | Cost: 2.358841e-01
Iteration    57 | Cost: 2.307865e-01
Iteration    58 | Cost: 2.301286e-01
Iteration    59 | Cost: 2.291855e-01
Iteration    60 | Cost: 2.291023e-01
Iteration    61 | Cost: 2.289326e-01
Iteration    62 | Cost: 2.288875e-01
Iteration    63 | Cost: 2.285192e-01
Iteration    64 | Cost: 2.284545e-01
Iteration    65 | Cost: 2.280451e-01
Iteration    66 | Cost: 2.239808e-01
Iteration    67 | Cost: 2.189897e-01
Iteration    68 | Cost: 2.189532e-01
Iteration    69 | Cost: 2.184894e-01
Iteration    70 | Cost: 2.183188e-01
Iteration    71 | Cost: 2.180967e-01
Iteration    72 | Cost: 2.179654e-01
Iteration    73 | Cost: 2.177326e-01
Iteration    74 | Cost: 2.173176e-01
Iteration    75 | Cost: 2.167163e-01
Iteration    76 | Cost: 2.164755e-01
Iteration    77 | Cost: 2.158295e-01
Iteration    78 | Cost: 2.155046e-01
Iteration    79 | Cost: 2.148354e-01
Iteration    80 | Cost: 2.147422e-01
Iteration    81 | Cost: 2.144075e-01
Iteration    82 | Cost: 2.143693e-01
```

```
Iteration    83 | Cost: 2.141972e-01
Iteration    84 | Cost: 2.141192e-01
Iteration    85 | Cost: 2.141103e-01
Iteration    86 | Cost: 2.138550e-01
Iteration    87 | Cost: 2.119028e-01
Iteration    88 | Cost: 2.046340e-01
Iteration    89 | Cost: 2.042429e-01
Iteration    90 | Cost: 2.041980e-01
Iteration    91 | Cost: 2.036113e-01
Iteration    92 | Cost: 2.017186e-01
Iteration    93 | Cost: 2.013463e-01
Iteration    94 | Cost: 2.004686e-01
Iteration    95 | Cost: 1.999644e-01
Iteration    96 | Cost: 1.963093e-01
Iteration    97 | Cost: 1.958168e-01
Iteration    98 | Cost: 1.957121e-01
Iteration    99 | Cost: 1.944667e-01
Iteration   100 | Cost: 1.939541e-01
Iteration   101 | Cost: 1.925358e-01
Iteration   102 | Cost: 1.917685e-01
Iteration   103 | Cost: 1.911452e-01
Iteration   104 | Cost: 1.885483e-01
Iteration   105 | Cost: 1.885395e-01
Iteration   106 | Cost: 1.883740e-01
Iteration   107 | Cost: 1.882346e-01
Iteration   108 | Cost: 1.878807e-01
Iteration   109 | Cost: 1.854029e-01
Iteration   110 | Cost: 1.824164e-01
Iteration   111 | Cost: 1.815231e-01
Iteration   112 | Cost: 1.813991e-01
Iteration   113 | Cost: 1.809461e-01
Iteration   114 | Cost: 1.809177e-01
Iteration   115 | Cost: 1.807522e-01
Iteration   116 | Cost: 1.806954e-01
```

```
Iteration   117 | Cost: 1.806214e-01
Iteration   118 | Cost: 1.806134e-01
Iteration   119 | Cost: 1.804196e-01
Iteration   120 | Cost: 1.803485e-01
Iteration   121 | Cost: 1.801379e-01
Iteration   122 | Cost: 1.799709e-01
Iteration   123 | Cost: 1.796702e-01
Iteration   124 | Cost: 1.795360e-01
Iteration   125 | Cost: 1.794017e-01
Iteration   126 | Cost: 1.793716e-01
Iteration   127 | Cost: 1.792366e-01
Iteration   128 | Cost: 1.792247e-01
Iteration   129 | Cost: 1.791994e-01
Iteration   130 | Cost: 1.791799e-01
Iteration   131 | Cost: 1.791255e-01
Iteration   132 | Cost: 1.789013e-01
Iteration   133 | Cost: 1.787711e-01
Iteration   134 | Cost: 1.787515e-01
Iteration   135 | Cost: 1.786736e-01
Iteration   136 | Cost: 1.786366e-01
Iteration   137 | Cost: 1.786283e-01
Iteration   138 | Cost: 1.785914e-01
Iteration   139 | Cost: 1.785566e-01
Iteration   140 | Cost: 1.785410e-01
Iteration   141 | Cost: 1.785000e-01
Iteration   142 | Cost: 1.783813e-01
Iteration   143 | Cost: 1.774892e-01
Iteration   144 | Cost: 1.736926e-01
Iteration   145 | Cost: 1.720728e-01
Iteration   146 | Cost: 1.720706e-01
Iteration   147 | Cost: 1.711048e-01
Iteration   148 | Cost: 1.708287e-01
Iteration   149 | Cost: 1.700181e-01
Iteration   150 | Cost: 1.691457e-01
```

```
Iteration   151 | Cost: 1.686950e-01
Iteration   152 | Cost: 1.686500e-01
Iteration   153 | Cost: 1.682155e-01
Iteration   154 | Cost: 1.681349e-01
Iteration   155 | Cost: 1.679686e-01
Iteration   156 | Cost: 1.678474e-01
Iteration   157 | Cost: 1.678009e-01
Iteration   158 | Cost: 1.677610e-01
Iteration   159 | Cost: 1.677220e-01
Iteration   160 | Cost: 1.676748e-01
Iteration   161 | Cost: 1.676457e-01
Iteration   162 | Cost: 1.674182e-01
Iteration   163 | Cost: 1.672913e-01
Iteration   164 | Cost: 1.667528e-01
Iteration   165 | Cost: 1.667463e-01
Iteration   166 | Cost: 1.667039e-01
Iteration   167 | Cost: 1.665557e-01
Iteration   168 | Cost: 1.664827e-01
Iteration   169 | Cost: 1.664219e-01
Iteration   170 | Cost: 1.654633e-01
Iteration   171 | Cost: 1.646207e-01
Iteration   172 | Cost: 1.622449e-01
Iteration   173 | Cost: 1.569990e-01
Iteration   174 | Cost: 1.559918e-01
Iteration   175 | Cost: 1.546440e-01
Iteration   176 | Cost: 1.540979e-01
Iteration   177 | Cost: 1.521063e-01
Iteration   178 | Cost: 1.515208e-01
Iteration   179 | Cost: 1.509163e-01
Iteration   180 | Cost: 1.500590e-01
Iteration   181 | Cost: 1.494642e-01
Iteration   182 | Cost: 1.487400e-01
Iteration   183 | Cost: 1.478801e-01
Iteration   184 | Cost: 1.476145e-01
```

```
Iteration   185 | Cost: 1.475262e-01
Iteration   186 | Cost: 1.474076e-01
Iteration   187 | Cost: 1.471583e-01
Iteration   188 | Cost: 1.465851e-01
Iteration   189 | Cost: 1.451885e-01
Iteration   190 | Cost: 1.451664e-01
Iteration   191 | Cost: 1.449961e-01
Iteration   192 | Cost: 1.449027e-01
Iteration   193 | Cost: 1.448725e-01
Iteration   194 | Cost: 1.418962e-01
Iteration   195 | Cost: 1.414781e-01
Iteration   196 | Cost: 1.408975e-01
Iteration   197 | Cost: 1.407741e-01
Iteration   198 | Cost: 1.407328e-01
Iteration   199 | Cost: 1.401835e-01
Iteration   200 | Cost: 1.401403e-01
```

After learning the parameters θ, the code below will generate two plots (Figures 4,5) for polynomial regression with $\lambda = 0$.

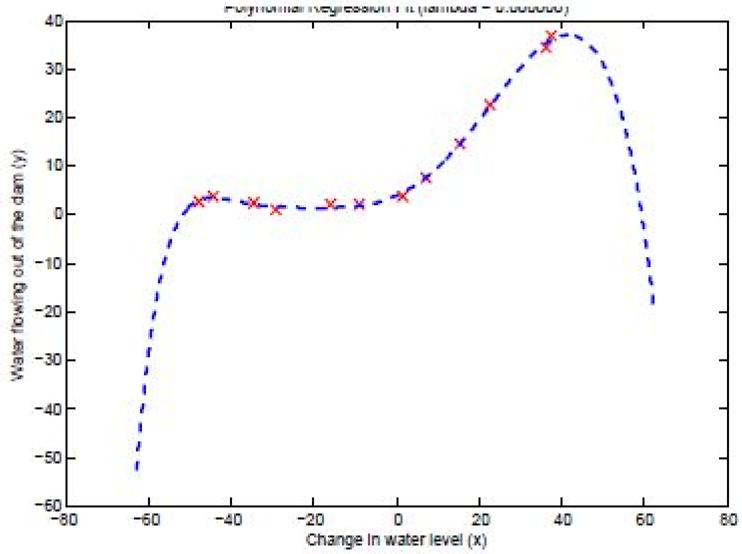


Figure 4: Polynomial fit, $\lambda = 0$

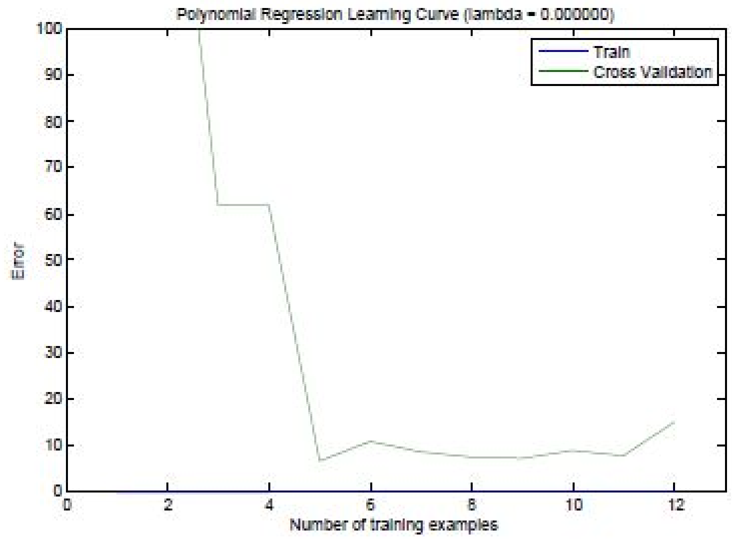


Figure 5: Polynomial learning curve, $\lambda = 0$

From Figure 4, you should see that the polynomial fit is able to follow the datapoints very well - thus, obtaining a low training error. However, the polynomial fit is very complex and even drops off at the extremes. This is an indicator that the polynomial regression model is overfitting the training data and will not generalize well.

```
% Plot training data and fit
plot(X, y, 'rx', 'MarkerSize', 10, 'LineWidth', 1.5);
plotFit(min(X), max(X), mu, sigma, theta, p);
xlabel('Change in water level (x)');
```

```
ylabel('Water flowing out of the dam (y)');

title (sprintf('Polynomial Regression Fit (lambda = %f)', lambda));

[error_train, error_val] = learningCurve(X_poly, y, X_poly_val, yval, lambda);

plot(1:m, error_train, 1:m, error_val);

title(sprintf('Polynomial Regression Learning Curve (lambda = %f)', lambda));

xlabel('Number of training examples')

ylabel('Error')

axis([0 13 0 100])

legend('Train', 'Cross Validation')
```

To better understand the problems with the unregularized $\lambda = 0$ model, you can see that the learning curve (Figure 5) shows the same effect where the low training error is low, but the cross validation error is high. There is a gap between the training and cross validation errors, indicating a high variance problem. One way to combat the overfitting (high-variance) problem is to add regularization to the model. In the next section, you will get to try different λ parameters to see how regularization can lead to a better model.

3.2 Optional (ungraded) exercise: Adjusting the regularization parameter

In this section, you will get to observe how the regularization parameter affects the bias-variance of regularized polynomial regression. You should now modify the the `lambda` parameter in the code below and try $\lambda = 1, 100$.

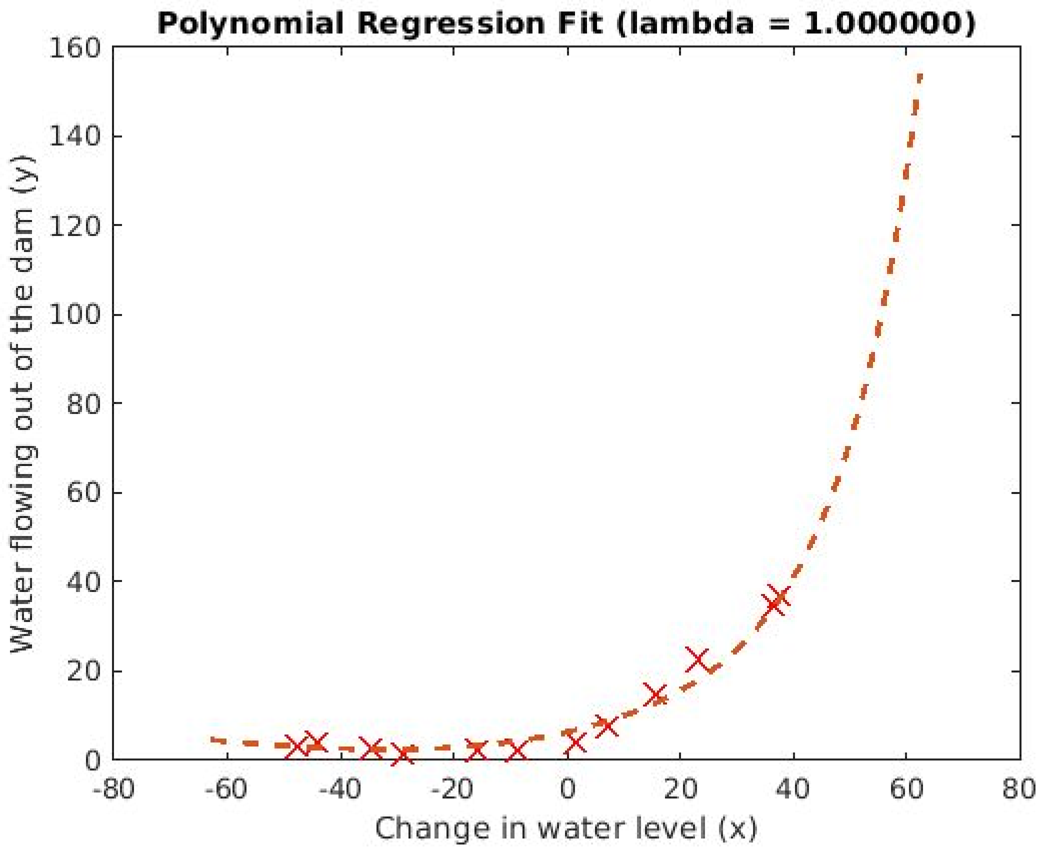
```
% Choose the value of lambda

lambda = 1;

[theta] = trainLinearReg(X_poly, y, lambda);
```

```
Iteration     1 | Cost: 8.320954e+01

Iteration     2 | Cost: 2.907694e+01

Iteration     3 | Cost: 1.613078e+01

Iteration     4 | Cost: 9.152504e+00

Iteration     5 | Cost: 8.191432e+00

Iteration     6 | Cost: 7.658009e+00

Iteration     7 | Cost: 7.558221e+00
```

```
Iteration     8 | Cost: 7.492678e+00
Iteration     9 | Cost: 7.446070e+00
Iteration    10 | Cost: 7.369658e+00
Iteration    11 | Cost: 7.324704e+00
Iteration    12 | Cost: 7.303917e+00
Iteration    13 | Cost: 7.279810e+00
Iteration    14 | Cost: 7.274783e+00
Iteration    15 | Cost: 7.272165e+00
Iteration    16 | Cost: 7.269806e+00
Iteration    17 | Cost: 7.268765e+00
Iteration    18 | Cost: 7.268261e+00
Iteration    19 | Cost: 7.268193e+00
Iteration    20 | Cost: 7.268160e+00
Iteration    21 | Cost: 7.268151e+00
Iteration    22 | Cost: 7.268150e+00
Iteration    23 | Cost: 7.268149e+00
Iteration    24 | Cost: 7.268148e+00
Iteration    25 | Cost: 7.268148e+00
Iteration    26 | Cost: 7.268148e+00
Iteration    27 | Cost: 7.268148e+00
Iteration    28 | Cost: 7.268148e+00
Iteration    29 | Cost: 7.268148e+00
Iteration    30 | Cost: 7.268148e+00
Iteration    31 | Cost: 7.268148e+00
Iteration    32 | Cost: 7.268148e+00
Iteration    33 | Cost: 7.268148e+00
Iteration    34 | Cost: 7.268148e+00
Iteration    35 | Cost: 7.268148e+00
Iteration    36 | Cost: 7.268148e+00
Iteration    37 | Cost: 7.268148e+00
Iteration    38 | Cost: 7.268148e+00
Iteration    39 | Cost: 7.268148e+00
Iteration    40 | Cost: 7.268148e+00
Iteration    41 | Cost: 7.268148e+00
```

```
Iteration    42 | Cost: 7.268148e+00

Iteration    43 | Cost: 7.268148e+00

Iteration    44 | Cost: 7.268148e+00

Iteration    45 | Cost: 7.268148e+00

Iteration    46 | Cost: 7.268148e+00

Iteration    47 | Cost: 7.268148e+00

Iteration    48 | Cost: 7.268148e+00

Iteration    49 | Cost: 7.268148e+00

Iteration    50 | Cost: 7.268148e+00

Iteration    51 | Cost: 7.268148e+00

Iteration    52 | Cost: 7.268148e+00

Iteration    53 | Cost: 7.268148e+00

Iteration    54 | Cost: 7.268148e+00

Iteration    55 | Cost: 7.268148e+00

Iteration    56 | Cost: 7.268148e+00

Iteration    57 | Cost: 7.268148e+00

Iteration    58 | Cost: 7.268148e+00
```

```
% Plot training data and fit

plot(X, y, 'rx', 'MarkerSize', 10, 'LineWidth', 1.5);

plotFit(min(X), max(X), mu, sigma, theta, p);

xlabel('Change in water level (x)');

ylabel('Water flowing out of the dam (y)');

title (sprintf('Polynomial Regression Fit (lambda = %f)', lambda));
```




```
[error_train, error_val] = learningCurve(X_poly, y, X_poly_val, yval, lambda);
```

```
Iteration     1 | Cost: 5.280051e-01
Iteration     2 | Cost: 2.083531e-01
Iteration     3 | Cost: 9.810873e-02
Iteration     4 | Cost: 1.501116e-03
Iteration     5 | Cost: 1.089480e-03
Iteration     6 | Cost: 4.451497e-31
Iteration     7 | Cost: 3.829364e-31
Iteration     8 | Cost: 1.258474e-31
Iteration     9 | Cost: 1.098718e-31
Iteration     1 | Cost: 2.032751e-01
Iteration     2 | Cost: 1.041953e-01
Iteration     3 | Cost: 7.367762e-02
Iteration     4 | Cost: 7.279642e-02
Iteration     5 | Cost: 7.269797e-02
Iteration     6 | Cost: 7.269533e-02
```

```
Iteration     7 | Cost: 7.269502e-02
Iteration     8 | Cost: 7.269437e-02
Iteration     9 | Cost: 7.269436e-02
Iteration    10 | Cost: 7.269436e-02
Iteration    11 | Cost: 7.269436e-02
Iteration    12 | Cost: 7.269436e-02
Iteration    13 | Cost: 7.269436e-02
Iteration    14 | Cost: 7.269436e-02
Iteration    15 | Cost: 7.269436e-02
Iteration    16 | Cost: 7.269436e-02
Iteration    17 | Cost: 7.269436e-02
Iteration    18 | Cost: 7.269436e-02
Iteration    19 | Cost: 7.269436e-02
Iteration    20 | Cost: 7.269436e-02
Iteration     1 | Cost: 3.958682e+01
Iteration     2 | Cost: 2.172738e+01
Iteration     3 | Cost: 2.007128e+01
Iteration     4 | Cost: 1.908475e+01
Iteration     5 | Cost: 1.903782e+01
Iteration     6 | Cost: 1.872107e+01
Iteration     7 | Cost: 1.855211e+01
Iteration     8 | Cost: 1.852112e+01
Iteration     9 | Cost: 1.850029e+01
Iteration    10 | Cost: 1.849891e+01
Iteration    11 | Cost: 1.849887e+01
Iteration    12 | Cost: 1.849879e+01
Iteration    13 | Cost: 1.849879e+01
Iteration    14 | Cost: 1.849879e+01
Iteration    15 | Cost: 1.849879e+01
Iteration    16 | Cost: 1.849879e+01
Iteration    17 | Cost: 1.849879e+01
Iteration    18 | Cost: 1.849879e+01
Iteration    19 | Cost: 1.849879e+01
Iteration    20 | Cost: 1.849879e+01
```

```
Iteration    21 | Cost: 1.849879e+01
Iteration    22 | Cost: 1.849879e+01
Iteration     1 | Cost: 1.883439e+01
Iteration     2 | Cost: 1.695119e+01
Iteration     3 | Cost: 1.546680e+01
Iteration     4 | Cost: 1.486258e+01
Iteration     5 | Cost: 1.462699e+01
Iteration     6 | Cost: 1.460173e+01
Iteration     7 | Cost: 1.458789e+01
Iteration     8 | Cost: 1.458160e+01
Iteration     9 | Cost: 1.457865e+01
Iteration    10 | Cost: 1.457604e+01
Iteration    11 | Cost: 1.457595e+01
Iteration    12 | Cost: 1.457591e+01
Iteration    13 | Cost: 1.457590e+01
Iteration    14 | Cost: 1.457588e+01
Iteration    15 | Cost: 1.457587e+01
Iteration    16 | Cost: 1.457586e+01
Iteration    17 | Cost: 1.457586e+01
Iteration    18 | Cost: 1.457586e+01
Iteration    19 | Cost: 1.457586e+01
Iteration    20 | Cost: 1.457586e+01
Iteration    21 | Cost: 1.457586e+01
Iteration    22 | Cost: 1.457586e+01
Iteration    23 | Cost: 1.457586e+01
Iteration    24 | Cost: 1.457586e+01
Iteration    25 | Cost: 1.457586e+01
Iteration    26 | Cost: 1.457586e+01
Iteration    27 | Cost: 1.457586e+01
Iteration    28 | Cost: 1.457586e+01
Iteration    29 | Cost: 1.457586e+01
Iteration    30 | Cost: 1.457586e+01
Iteration    31 | Cost: 1.457586e+01
Iteration    32 | Cost: 1.457586e+01
```

```
Iteration    33 | Cost: 1.457586e+01
Iteration    34 | Cost: 1.457586e+01
Iteration    35 | Cost: 1.457586e+01
Iteration    36 | Cost: 1.457586e+01
Iteration    37 | Cost: 1.457586e+01
Iteration    38 | Cost: 1.457586e+01
Iteration    39 | Cost: 1.457586e+01
Iteration    40 | Cost: 1.457586e+01
Iteration    41 | Cost: 1.457586e+01
Iteration    42 | Cost: 1.457586e+01
Iteration    43 | Cost: 1.457586e+01
Iteration    44 | Cost: 1.457586e+01
Iteration    45 | Cost: 1.457586e+01
Iteration    46 | Cost: 1.457586e+01
Iteration    47 | Cost: 1.457586e+01
Iteration    48 | Cost: 1.457586e+01
Iteration    49 | Cost: 1.457586e+01
Iteration    50 | Cost: 1.457586e+01
Iteration     1 | Cost: 9.507932e+01
Iteration     2 | Cost: 3.745281e+01
Iteration     3 | Cost: 1.603853e+01
Iteration     4 | Cost: 1.259101e+01
Iteration     5 | Cost: 1.224802e+01
Iteration     6 | Cost: 1.203684e+01
Iteration     7 | Cost: 1.177021e+01
Iteration     8 | Cost: 1.170389e+01
Iteration     9 | Cost: 1.167454e+01
Iteration    10 | Cost: 1.166570e+01
Iteration    11 | Cost: 1.166402e+01
Iteration    12 | Cost: 1.166161e+01
Iteration    13 | Cost: 1.166152e+01
Iteration    14 | Cost: 1.166127e+01
Iteration    15 | Cost: 1.166103e+01
Iteration    16 | Cost: 1.166090e+01
```

```
Iteration    17 | Cost: 1.166090e+01
Iteration    18 | Cost: 1.166084e+01
Iteration    19 | Cost: 1.166072e+01
Iteration    20 | Cost: 1.166072e+01
Iteration    21 | Cost: 1.166072e+01
Iteration    22 | Cost: 1.166072e+01
Iteration    23 | Cost: 1.166072e+01
Iteration    24 | Cost: 1.166072e+01
Iteration    25 | Cost: 1.166072e+01
Iteration    26 | Cost: 1.166072e+01
Iteration    27 | Cost: 1.166072e+01
Iteration    28 | Cost: 1.166072e+01
Iteration    29 | Cost: 1.166072e+01
Iteration    30 | Cost: 1.166072e+01
Iteration    31 | Cost: 1.166072e+01
Iteration    32 | Cost: 1.166072e+01
Iteration    33 | Cost: 1.166072e+01
Iteration    34 | Cost: 1.166072e+01
Iteration    35 | Cost: 1.166072e+01
Iteration    36 | Cost: 1.166072e+01
Iteration    37 | Cost: 1.166072e+01
Iteration    38 | Cost: 1.166072e+01
Iteration    39 | Cost: 1.166072e+01
Iteration    40 | Cost: 1.166072e+01
Iteration    41 | Cost: 1.166072e+01
Iteration    43 | Cost: 1.166072e+01
Iteration    44 | Cost: 1.166072e+01
Iteration    45 | Cost: 1.166072e+01
Iteration    46 | Cost: 1.166072e+01
Iteration    47 | Cost: 1.166072e+01
Iteration     1 | Cost: 7.115385e+01
Iteration     2 | Cost: 2.111452e+01
Iteration     3 | Cost: 1.314175e+01
Iteration     4 | Cost: 1.054644e+01
```

```
Iteration     5 | Cost: 1.036773e+01
Iteration     6 | Cost: 1.034120e+01
Iteration     7 | Cost: 1.026977e+01
Iteration     8 | Cost: 1.022714e+01
Iteration     9 | Cost: 1.019759e+01
Iteration    10 | Cost: 1.015385e+01
Iteration    11 | Cost: 1.014049e+01
Iteration    12 | Cost: 1.013163e+01
Iteration    13 | Cost: 1.013124e+01
Iteration    14 | Cost: 1.012855e+01
Iteration    15 | Cost: 1.012593e+01
Iteration    16 | Cost: 1.011958e+01
Iteration    17 | Cost: 1.011929e+01
Iteration    18 | Cost: 1.011892e+01
Iteration    19 | Cost: 1.011888e+01
Iteration    20 | Cost: 1.011885e+01
Iteration    21 | Cost: 1.011883e+01
Iteration    22 | Cost: 1.011882e+01
Iteration    23 | Cost: 1.011882e+01
Iteration    24 | Cost: 1.011882e+01
Iteration    25 | Cost: 1.011882e+01
Iteration    26 | Cost: 1.011882e+01
Iteration    27 | Cost: 1.011882e+01
Iteration    28 | Cost: 1.011882e+01
Iteration    29 | Cost: 1.011882e+01
Iteration    30 | Cost: 1.011882e+01
Iteration    31 | Cost: 1.011882e+01
Iteration    32 | Cost: 1.011882e+01
Iteration    33 | Cost: 1.011882e+01
Iteration    34 | Cost: 1.011882e+01
Iteration    35 | Cost: 1.011882e+01
Iteration    36 | Cost: 1.011882e+01
Iteration    37 | Cost: 1.011882e+01
Iteration    38 | Cost: 1.011882e+01
```

```
Iteration    39 | Cost: 1.011882e+01
Iteration    40 | Cost: 1.011882e+01
Iteration    41 | Cost: 1.011882e+01
Iteration    42 | Cost: 1.011882e+01
Iteration    43 | Cost: 1.011882e+01
Iteration    44 | Cost: 1.011882e+01
Iteration    45 | Cost: 1.011882e+01
Iteration    46 | Cost: 1.011882e+01
Iteration    47 | Cost: 1.011882e+01
Iteration    48 | Cost: 1.011882e+01
Iteration    49 | Cost: 1.011882e+01
Iteration    50 | Cost: 1.011882e+01
Iteration    51 | Cost: 1.011882e+01
Iteration    52 | Cost: 1.011882e+01
Iteration    53 | Cost: 1.011882e+01
Iteration    54 | Cost: 1.011882e+01
Iteration     1 | Cost: 7.518060e+01
Iteration     2 | Cost: 2.616238e+01
Iteration     3 | Cost: 1.560487e+01
Iteration     4 | Cost: 1.001814e+01
Iteration     5 | Cost: 9.677558e+00
Iteration     6 | Cost: 9.664421e+00
Iteration     7 | Cost: 9.518755e+00
Iteration     8 | Cost: 9.489240e+00
Iteration     9 | Cost: 9.459083e+00
Iteration    10 | Cost: 9.453023e+00
Iteration    11 | Cost: 9.432509e+00
Iteration    12 | Cost: 9.427300e+00
Iteration    13 | Cost: 9.420825e+00
Iteration    14 | Cost: 9.420341e+00
Iteration    15 | Cost: 9.419615e+00
Iteration    16 | Cost: 9.419036e+00
Iteration    17 | Cost: 9.417454e+00
Iteration    18 | Cost: 9.416487e+00
```

```
Iteration    19 | Cost: 9.416371e+00
Iteration    20 | Cost: 9.416339e+00
Iteration    21 | Cost: 9.416337e+00
Iteration    22 | Cost: 9.416333e+00
Iteration    23 | Cost: 9.416331e+00
Iteration    24 | Cost: 9.416323e+00
Iteration    25 | Cost: 9.416319e+00
Iteration    26 | Cost: 9.416318e+00
Iteration    27 | Cost: 9.416318e+00
Iteration    28 | Cost: 9.416317e+00
Iteration    29 | Cost: 9.416317e+00
Iteration    30 | Cost: 9.416317e+00
Iteration    31 | Cost: 9.416317e+00
Iteration    32 | Cost: 9.416317e+00
Iteration    33 | Cost: 9.416317e+00
Iteration    34 | Cost: 9.416317e+00
Iteration    35 | Cost: 9.416317e+00
Iteration    36 | Cost: 9.416317e+00
Iteration    37 | Cost: 9.416317e+00
Iteration    38 | Cost: 9.416317e+00
Iteration    39 | Cost: 9.416317e+00
Iteration    40 | Cost: 9.416317e+00
Iteration    41 | Cost: 9.416317e+00
Iteration    42 | Cost: 9.416317e+00
Iteration    43 | Cost: 9.416317e+00
Iteration    44 | Cost: 9.416317e+00
Iteration    45 | Cost: 9.416317e+00
Iteration    46 | Cost: 9.416317e+00
Iteration    47 | Cost: 9.416317e+00
Iteration    48 | Cost: 9.416317e+00
Iteration    49 | Cost: 9.416317e+00
Iteration    50 | Cost: 9.416317e+00
Iteration    51 | Cost: 9.416317e+00
Iteration    52 | Cost: 9.416317e+00
```

```
Iteration    53 | Cost: 9.416317e+00
Iteration    54 | Cost: 9.416317e+00
Iteration    55 | Cost: 9.416317e+00
Iteration    57 | Cost: 9.416317e+00
Iteration    59 | Cost: 9.416317e+00
Iteration     1 | Cost: 6.611004e+01
Iteration     2 | Cost: 1.227749e+01
Iteration     3 | Cost: 1.079462e+01
Iteration     4 | Cost: 8.838726e+00
Iteration     5 | Cost: 8.699382e+00
Iteration     6 | Cost: 8.507287e+00
Iteration     7 | Cost: 8.364305e+00
Iteration     8 | Cost: 8.333738e+00
Iteration     9 | Cost: 8.291157e+00
Iteration    10 | Cost: 8.288967e+00
Iteration    11 | Cost: 8.279162e+00
Iteration    12 | Cost: 8.274785e+00
Iteration    13 | Cost: 8.265375e+00
Iteration    14 | Cost: 8.258230e+00
Iteration    15 | Cost: 8.257980e+00
Iteration    16 | Cost: 8.257039e+00
Iteration    17 | Cost: 8.256708e+00
Iteration    18 | Cost: 8.256325e+00
Iteration    19 | Cost: 8.256301e+00
Iteration    20 | Cost: 8.256290e+00
Iteration    21 | Cost: 8.256253e+00
Iteration    22 | Cost: 8.256243e+00
Iteration    23 | Cost: 8.256231e+00
Iteration    24 | Cost: 8.256230e+00
Iteration    25 | Cost: 8.256227e+00
Iteration    26 | Cost: 8.256225e+00
Iteration    27 | Cost: 8.256225e+00
Iteration    28 | Cost: 8.256224e+00
Iteration    29 | Cost: 8.256224e+00
```

```
Iteration    30 | Cost: 8.256224e+00
Iteration    31 | Cost: 8.256224e+00
Iteration    32 | Cost: 8.256224e+00
Iteration    33 | Cost: 8.256224e+00
Iteration    34 | Cost: 8.256224e+00
Iteration    35 | Cost: 8.256224e+00
Iteration    36 | Cost: 8.256224e+00
Iteration    37 | Cost: 8.256224e+00
Iteration    38 | Cost: 8.256224e+00
Iteration    39 | Cost: 8.256224e+00
Iteration    40 | Cost: 8.256224e+00
Iteration    41 | Cost: 8.256224e+00
Iteration    42 | Cost: 8.256224e+00
Iteration    43 | Cost: 8.256224e+00
Iteration    44 | Cost: 8.256224e+00
Iteration    45 | Cost: 8.256224e+00
Iteration    46 | Cost: 8.256224e+00
Iteration    47 | Cost: 8.256224e+00
Iteration    48 | Cost: 8.256224e+00
Iteration    49 | Cost: 8.256224e+00
Iteration    50 | Cost: 8.256224e+00
Iteration    51 | Cost: 8.256224e+00
Iteration    52 | Cost: 8.256224e+00
Iteration    53 | Cost: 8.256224e+00
Iteration    54 | Cost: 8.256224e+00
Iteration    55 | Cost: 8.256224e+00
Iteration     1 | Cost: 6.224886e+01
Iteration     2 | Cost: 1.460426e+01
Iteration     3 | Cost: 1.051408e+01
Iteration     4 | Cost: 8.070570e+00
Iteration     5 | Cost: 8.021920e+00
Iteration     6 | Cost: 7.958519e+00
Iteration     7 | Cost: 7.928378e+00
Iteration     8 | Cost: 7.906501e+00
```

```
Iteration     9 | Cost: 7.876402e+00
Iteration    10 | Cost: 7.847865e+00
Iteration    11 | Cost: 7.843141e+00
Iteration    12 | Cost: 7.812629e+00
Iteration    13 | Cost: 7.806123e+00
Iteration    14 | Cost: 7.802387e+00
Iteration    15 | Cost: 7.802258e+00
Iteration    16 | Cost: 7.802197e+00
Iteration    17 | Cost: 7.802175e+00
Iteration    18 | Cost: 7.802145e+00
Iteration    19 | Cost: 7.802124e+00
Iteration    20 | Cost: 7.802116e+00
Iteration    21 | Cost: 7.802087e+00
Iteration    22 | Cost: 7.802079e+00
Iteration    23 | Cost: 7.802078e+00
Iteration    24 | Cost: 7.802077e+00
Iteration    25 | Cost: 7.802077e+00
Iteration    26 | Cost: 7.802077e+00
Iteration    27 | Cost: 7.802077e+00
Iteration    28 | Cost: 7.802077e+00
Iteration    29 | Cost: 7.802076e+00
Iteration    30 | Cost: 7.802076e+00
Iteration    31 | Cost: 7.802076e+00
Iteration    32 | Cost: 7.802076e+00
Iteration    33 | Cost: 7.802076e+00
Iteration    34 | Cost: 7.802076e+00
Iteration    35 | Cost: 7.802076e+00
Iteration    36 | Cost: 7.802076e+00
Iteration    37 | Cost: 7.802076e+00
Iteration    38 | Cost: 7.802076e+00
Iteration    39 | Cost: 7.802076e+00
Iteration    40 | Cost: 7.802076e+00
Iteration    41 | Cost: 7.802076e+00
Iteration    42 | Cost: 7.802076e+00
```

```
Iteration    43 | Cost: 7.802076e+00
Iteration    44 | Cost: 7.802076e+00
Iteration    45 | Cost: 7.802076e+00
Iteration    46 | Cost: 7.802076e+00
Iteration    47 | Cost: 7.802076e+00
Iteration    48 | Cost: 7.802076e+00
Iteration    49 | Cost: 7.802076e+00
Iteration    50 | Cost: 7.802076e+00
Iteration    51 | Cost: 7.802076e+00
Iteration    52 | Cost: 7.802076e+00
Iteration    53 | Cost: 7.802076e+00
Iteration    54 | Cost: 7.802076e+00
Iteration    55 | Cost: 7.802076e+00
Iteration     1 | Cost: 5.861091e+01
Iteration     2 | Cost: 8.689923e+00
Iteration     3 | Cost: 7.624823e+00
Iteration     4 | Cost: 7.423501e+00
Iteration     5 | Cost: 7.267058e+00
Iteration     6 | Cost: 7.245380e+00
Iteration     7 | Cost: 7.156642e+00
Iteration     8 | Cost: 7.148104e+00
Iteration     9 | Cost: 7.117055e+00
Iteration    10 | Cost: 7.081459e+00
Iteration    11 | Cost: 7.077965e+00
Iteration    12 | Cost: 7.068155e+00
Iteration    13 | Cost: 7.064677e+00
Iteration    14 | Cost: 7.064563e+00
Iteration    15 | Cost: 7.064456e+00
Iteration    16 | Cost: 7.064432e+00
Iteration    17 | Cost: 7.064409e+00
Iteration    18 | Cost: 7.064407e+00
Iteration    19 | Cost: 7.064405e+00
Iteration    20 | Cost: 7.064401e+00
Iteration    21 | Cost: 7.064400e+00
```

```
Iteration    22 | Cost: 7.064399e+00
Iteration    23 | Cost: 7.064399e+00
Iteration    24 | Cost: 7.064399e+00
Iteration    25 | Cost: 7.064399e+00
Iteration    26 | Cost: 7.064398e+00
Iteration    27 | Cost: 7.064398e+00
Iteration    28 | Cost: 7.064398e+00
Iteration    29 | Cost: 7.064398e+00
Iteration    30 | Cost: 7.064398e+00
Iteration    31 | Cost: 7.064398e+00
Iteration    32 | Cost: 7.064398e+00
Iteration    33 | Cost: 7.064398e+00
Iteration    34 | Cost: 7.064398e+00
Iteration    35 | Cost: 7.064398e+00
Iteration    36 | Cost: 7.064398e+00
Iteration    37 | Cost: 7.064398e+00
Iteration    38 | Cost: 7.064398e+00
Iteration    39 | Cost: 7.064398e+00
Iteration    40 | Cost: 7.064398e+00
Iteration    41 | Cost: 7.064398e+00
Iteration    42 | Cost: 7.064398e+00
Iteration    43 | Cost: 7.064398e+00
Iteration    44 | Cost: 7.064398e+00
Iteration    45 | Cost: 7.064398e+00
Iteration    46 | Cost: 7.064398e+00
Iteration    47 | Cost: 7.064398e+00
Iteration    48 | Cost: 7.064398e+00
Iteration    49 | Cost: 7.064398e+00
Iteration    50 | Cost: 7.064398e+00
Iteration    51 | Cost: 7.064398e+00
Iteration    52 | Cost: 7.064398e+00
Iteration    53 | Cost: 7.064398e+00
Iteration    55 | Cost: 7.064398e+00
Iteration    56 | Cost: 7.064398e+00
```

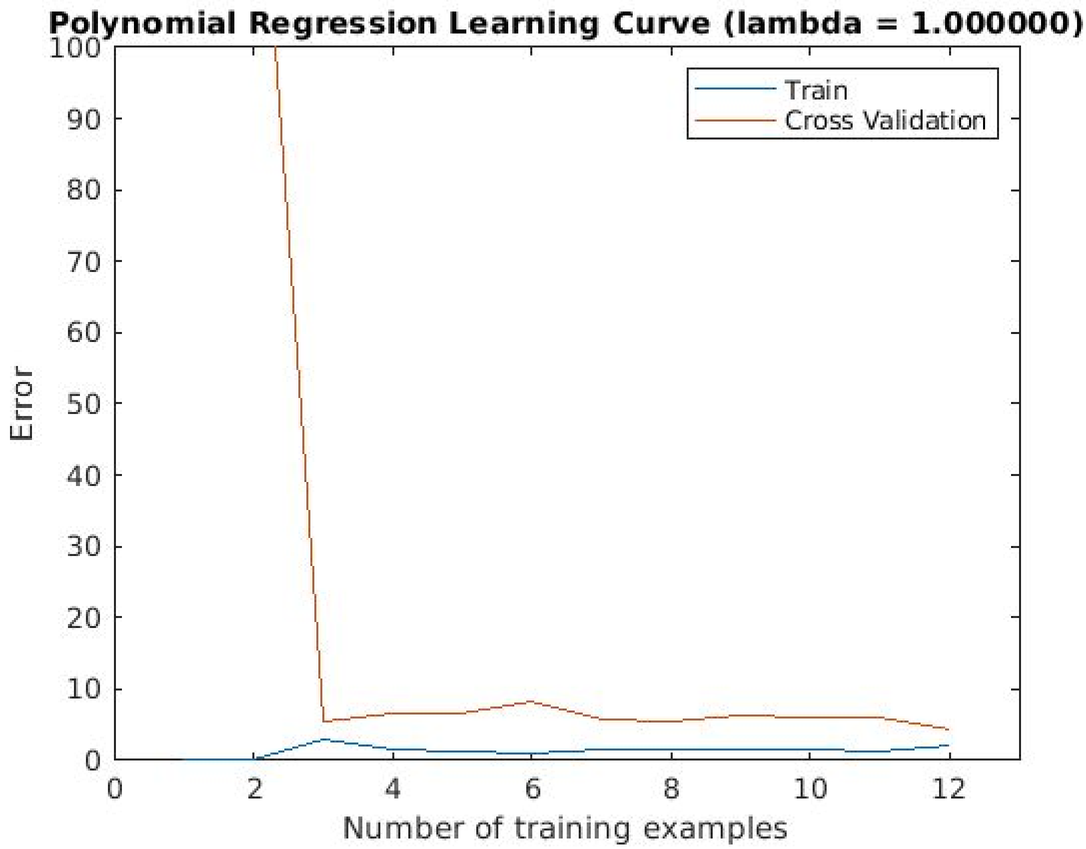
```
Iteration    57 | Cost: 7.064398e+00
Iteration     1 | Cost: 7.248552e+01
Iteration     2 | Cost: 2.583449e+01
Iteration     3 | Cost: 8.711534e+00
Iteration     4 | Cost: 6.790164e+00
Iteration     5 | Cost: 6.687293e+00
Iteration     6 | Cost: 6.663962e+00
Iteration     7 | Cost: 6.595243e+00
Iteration     8 | Cost: 6.538932e+00
Iteration     9 | Cost: 6.513460e+00
Iteration    10 | Cost: 6.477429e+00
Iteration    11 | Cost: 6.451620e+00
Iteration    12 | Cost: 6.449203e+00
Iteration    13 | Cost: 6.428503e+00
Iteration    14 | Cost: 6.426517e+00
Iteration    15 | Cost: 6.424335e+00
Iteration    16 | Cost: 6.424266e+00
Iteration    17 | Cost: 6.424039e+00
Iteration    18 | Cost: 6.423949e+00
Iteration    19 | Cost: 6.423880e+00
Iteration    20 | Cost: 6.423857e+00
Iteration    21 | Cost: 6.423835e+00
Iteration    22 | Cost: 6.423806e+00
Iteration    23 | Cost: 6.423802e+00
Iteration    24 | Cost: 6.423797e+00
Iteration    25 | Cost: 6.423792e+00
Iteration    26 | Cost: 6.423788e+00
Iteration    27 | Cost: 6.423785e+00
Iteration    28 | Cost: 6.423785e+00
Iteration    29 | Cost: 6.423785e+00
Iteration    30 | Cost: 6.423785e+00
Iteration    31 | Cost: 6.423785e+00
Iteration    32 | Cost: 6.423785e+00
Iteration    33 | Cost: 6.423784e+00
```

```
Iteration    34 | Cost: 6.423784e+00
Iteration    35 | Cost: 6.423784e+00
Iteration    36 | Cost: 6.423784e+00
Iteration    37 | Cost: 6.423784e+00
Iteration    38 | Cost: 6.423784e+00
Iteration    39 | Cost: 6.423784e+00
Iteration    40 | Cost: 6.423784e+00
Iteration    41 | Cost: 6.423784e+00
Iteration    42 | Cost: 6.423784e+00
Iteration    43 | Cost: 6.423784e+00
Iteration    44 | Cost: 6.423784e+00
Iteration    45 | Cost: 6.423784e+00
Iteration    46 | Cost: 6.423784e+00
Iteration    47 | Cost: 6.423784e+00
Iteration    48 | Cost: 6.423784e+00
Iteration    49 | Cost: 6.423784e+00
Iteration    50 | Cost: 6.423784e+00
Iteration    51 | Cost: 6.423784e+00
Iteration    52 | Cost: 6.423784e+00
Iteration    53 | Cost: 6.423784e+00
Iteration    54 | Cost: 6.423784e+00
Iteration    55 | Cost: 6.423784e+00
Iteration    56 | Cost: 6.423784e+00
Iteration    57 | Cost: 6.423784e+00
Iteration    58 | Cost: 6.423784e+00
Iteration    59 | Cost: 6.423784e+00
Iteration    60 | Cost: 6.423784e+00
Iteration    61 | Cost: 6.423784e+00
Iteration    62 | Cost: 6.423784e+00
Iteration    63 | Cost: 6.423784e+00
Iteration    64 | Cost: 6.423784e+00
Iteration    65 | Cost: 6.423784e+00
Iteration    66 | Cost: 6.423784e+00
Iteration     1 | Cost: 8.320954e+01
```

```
Iteration     2 | Cost: 2.907694e+01
Iteration     3 | Cost: 1.613078e+01
Iteration     4 | Cost: 9.152504e+00
Iteration     5 | Cost: 8.191432e+00
Iteration     6 | Cost: 7.658009e+00
Iteration     7 | Cost: 7.558221e+00
Iteration     8 | Cost: 7.492678e+00
Iteration     9 | Cost: 7.446070e+00
Iteration    10 | Cost: 7.369658e+00
Iteration    11 | Cost: 7.324704e+00
Iteration    12 | Cost: 7.303917e+00
Iteration    13 | Cost: 7.279810e+00
Iteration    14 | Cost: 7.274783e+00
Iteration    15 | Cost: 7.272165e+00
Iteration    16 | Cost: 7.269806e+00
Iteration    17 | Cost: 7.268765e+00
Iteration    18 | Cost: 7.268261e+00
Iteration    19 | Cost: 7.268193e+00
Iteration    20 | Cost: 7.268160e+00
Iteration    21 | Cost: 7.268151e+00
Iteration    22 | Cost: 7.268150e+00
Iteration    23 | Cost: 7.268149e+00
Iteration    24 | Cost: 7.268148e+00
Iteration    25 | Cost: 7.268148e+00
Iteration    26 | Cost: 7.268148e+00
Iteration    27 | Cost: 7.268148e+00
Iteration    28 | Cost: 7.268148e+00
Iteration    29 | Cost: 7.268148e+00
Iteration    30 | Cost: 7.268148e+00
Iteration    31 | Cost: 7.268148e+00
Iteration    32 | Cost: 7.268148e+00
Iteration    33 | Cost: 7.268148e+00
Iteration    34 | Cost: 7.268148e+00
Iteration    35 | Cost: 7.268148e+00
```

```
Iteration    36 | Cost: 7.268148e+00
Iteration    37 | Cost: 7.268148e+00
Iteration    38 | Cost: 7.268148e+00
Iteration    39 | Cost: 7.268148e+00
Iteration    40 | Cost: 7.268148e+00
Iteration    41 | Cost: 7.268148e+00
Iteration    42 | Cost: 7.268148e+00
Iteration    43 | Cost: 7.268148e+00
Iteration    44 | Cost: 7.268148e+00
Iteration    45 | Cost: 7.268148e+00
Iteration    46 | Cost: 7.268148e+00
Iteration    47 | Cost: 7.268148e+00
Iteration    48 | Cost: 7.268148e+00
Iteration    49 | Cost: 7.268148e+00
Iteration    50 | Cost: 7.268148e+00
Iteration    51 | Cost: 7.268148e+00
Iteration    52 | Cost: 7.268148e+00
Iteration    53 | Cost: 7.268148e+00
Iteration    54 | Cost: 7.268148e+00
Iteration    55 | Cost: 7.268148e+00
Iteration    56 | Cost: 7.268148e+00
Iteration    57 | Cost: 7.268148e+00
Iteration    58 | Cost: 7.268148e+00
```

```
plot(1:m, error_train, 1:m, error_val);

title(sprintf('Polynomial Regression Learning Curve (lambda = %f)',
lambda));

xlabel('Number of training examples')

ylabel('Error')

axis([0 13 0 100])

legend('Train', 'Cross Validation')
```



For each of these values, the code should generate a polynomial fit to the data and also a learning curve.

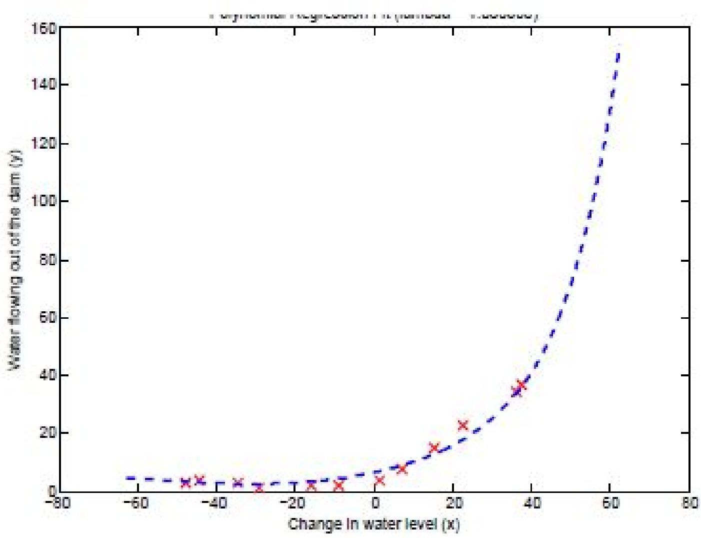


Figure 6: Polynomial fit, $\lambda = 1$

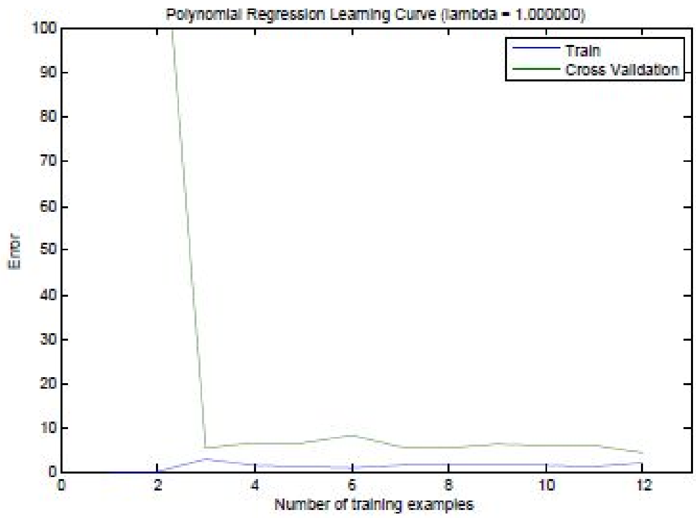


Figure 7: Polynomial learning curve, $\lambda = 1$

For $\lambda = 1$, you should see a polynomial fit that follows the data trend well (Figure 6) and a learning curve (Figure 7) showing that both the cross validation and training error converge to a relatively low value. This shows the $\lambda = 1$ regularized polynomial regression model does not have the high bias or high-variance problems. In effect, it achieves a good trade-off between bias and variance.

For $\lambda = 100$, you should see a polynomial fit (Figure 8) that does not follow the data well. In this case, there is too much regularization and the model is unable to fit the training data.

Figure 7: Polynomial learning curve, $\lambda = 1$

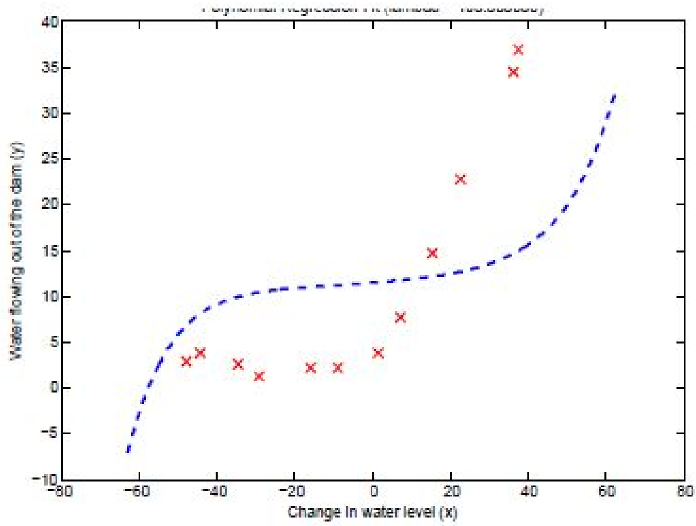


Figure 8: Polynomial fit, $\lambda = 100$

3.3 Selecting lambda using a cross validation set

From the previous parts of the exercise, you observed that the value of λ can significantly affect the results of regularized polynomial regression on the training and cross validation set. In particular, a model without regularization ($\lambda = 0$) fits the training set well, but does not generalize. Conversely, a model with too much regularization ($\lambda = 100$) does not fit the training set and testing set well. A good choice of λ (e.g. $\lambda = 1$) can provide a good fit to the data.

In this section, you will implement an automated method to select the parameter. Concretely, you will use a cross validation set to evaluate how good each λ value is. After selecting the best λ value using the cross validation set, we can then evaluate the model on the test set to estimate how well the model will perform on actual unseen data. Your task is to complete the code in `validationCurve.m`. Specifically, you should should use the `trainLinearReg` function to train the model using different values of λ and compute the training error and cross validation error. The function will try λ in the following range: {0, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10}.

After you have completed the code, the code below will run your function and plot a cross validation curve of error v.s λ that allows you select which λ parameter to use. You should see a plot similar to Figure 9.

```
[lambda_vec, error_train, error_val] = validationCurve(X_poly, y,
X_poly_val, yval);
```

```
Iteration     1 | Cost: 3.386454e-01
Iteration     2 | Cost: 9.860761e-32
Iteration     4 | Cost: 0.000000e+00
Iteration     1 | Cost: 5.388366e-02
Iteration     2 | Cost: 5.797847e-04
Iteration     3 | Cost: 3.213941e-04
Iteration     4 | Cost: 3.060895e-04
Iteration     5 | Cost: 3.033762e-04
Iteration     6 | Cost: 2.472631e-04
Iteration     7 | Cost: 2.223894e-04
Iteration     8 | Cost: 1.984294e-04
Iteration     9 | Cost: 1.978392e-04
Iteration    10 | Cost: 1.975906e-04
Iteration    11 | Cost: 1.975157e-04
Iteration    12 | Cost: 1.975029e-04
Iteration    13 | Cost: 1.974982e-04
Iteration    14 | Cost: 1.974969e-04
Iteration    15 | Cost: 1.974958e-04
Iteration    16 | Cost: 1.974942e-04
Iteration    17 | Cost: 1.974806e-04
Iteration    18 | Cost: 1.970315e-04
Iteration    19 | Cost: 1.967079e-04
Iteration    20 | Cost: 1.965110e-04
Iteration    21 | Cost: 1.962168e-04
Iteration    22 | Cost: 1.961492e-04
Iteration    23 | Cost: 1.960509e-04
Iteration    24 | Cost: 1.960472e-04
Iteration    25 | Cost: 1.960456e-04
Iteration    26 | Cost: 1.960433e-04
Iteration    27 | Cost: 1.960432e-04
Iteration    28 | Cost: 1.960430e-04
Iteration    29 | Cost: 1.960428e-04
```

```
Iteration    30 | Cost: 1.960427e-04
Iteration    31 | Cost: 1.960414e-04
Iteration    32 | Cost: 1.960359e-04
Iteration    33 | Cost: 1.960187e-04
Iteration    34 | Cost: 1.960011e-04
Iteration    35 | Cost: 1.959654e-04
Iteration    36 | Cost: 1.959627e-04
Iteration    37 | Cost: 1.959501e-04
Iteration    38 | Cost: 1.959494e-04
Iteration    39 | Cost: 1.959494e-04
Iteration    40 | Cost: 1.959494e-04
Iteration    41 | Cost: 1.959494e-04
Iteration    42 | Cost: 1.959494e-04
Iteration    43 | Cost: 1.959494e-04
Iteration    44 | Cost: 1.959494e-04
Iteration    45 | Cost: 1.959494e-04
Iteration    46 | Cost: 1.959494e-04
Iteration    47 | Cost: 1.959494e-04
Iteration    48 | Cost: 1.959494e-04
Iteration    49 | Cost: 1.959494e-04
Iteration    50 | Cost: 1.959494e-04
Iteration    51 | Cost: 1.959494e-04
Iteration    52 | Cost: 1.959494e-04
Iteration    53 | Cost: 1.959494e-04
Iteration    54 | Cost: 1.959494e-04
Iteration    55 | Cost: 1.959494e-04
Iteration    56 | Cost: 1.959494e-04
Iteration    57 | Cost: 1.959494e-04
Iteration    58 | Cost: 1.959494e-04
Iteration    59 | Cost: 1.959494e-04
Iteration    60 | Cost: 1.959494e-04
Iteration    61 | Cost: 1.959494e-04
Iteration    62 | Cost: 1.959494e-04
Iteration    63 | Cost: 1.959494e-04
```

```
Iteration    64 | Cost: 1.959494e-04
Iteration    65 | Cost: 1.959494e-04
Iteration    66 | Cost: 1.959494e-04
Iteration    67 | Cost: 1.959494e-04
Iteration    68 | Cost: 1.959494e-04
Iteration    69 | Cost: 1.959494e-04
Iteration    70 | Cost: 1.959494e-04
Iteration    71 | Cost: 1.959494e-04
Iteration    72 | Cost: 1.959494e-04
Iteration    73 | Cost: 1.959494e-04
Iteration    74 | Cost: 1.959494e-04
Iteration    75 | Cost: 1.959494e-04
Iteration    76 | Cost: 1.959494e-04
Iteration    77 | Cost: 1.959494e-04
Iteration    78 | Cost: 1.959494e-04
Iteration    79 | Cost: 1.959494e-04
Iteration    80 | Cost: 1.959494e-04
Iteration    81 | Cost: 1.959494e-04
Iteration    82 | Cost: 1.959494e-04
Iteration    83 | Cost: 1.959494e-04
Iteration    84 | Cost: 1.959494e-04
Iteration    85 | Cost: 1.959494e-04
Iteration    86 | Cost: 1.959494e-04
Iteration    87 | Cost: 1.959494e-04
Iteration    89 | Cost: 1.959494e-04
Iteration    90 | Cost: 1.959494e-04
Iteration    91 | Cost: 1.959494e-04
Iteration     1 | Cost: 3.026187e+01
Iteration     2 | Cost: 1.050362e+00
Iteration     3 | Cost: 5.596101e-01
Iteration     4 | Cost: 9.037458e-02
Iteration     5 | Cost: 7.972936e-02
Iteration     6 | Cost: 7.923857e-02
Iteration     7 | Cost: 7.881213e-02
```

```
Iteration     8 | Cost: 7.862362e-02
Iteration     9 | Cost: 7.858012e-02
Iteration    10 | Cost: 7.851506e-02
Iteration    11 | Cost: 7.837818e-02
Iteration    12 | Cost: 7.808366e-02
Iteration    13 | Cost: 7.748911e-02
Iteration    14 | Cost: 7.705873e-02
Iteration    15 | Cost: 7.472727e-02
Iteration    16 | Cost: 7.000875e-02
Iteration    17 | Cost: 6.844997e-02
Iteration    18 | Cost: 6.675654e-02
Iteration    19 | Cost: 6.654234e-02
Iteration    20 | Cost: 6.650770e-02
Iteration    21 | Cost: 6.647131e-02
Iteration    22 | Cost: 6.643264e-02
Iteration    23 | Cost: 6.642765e-02
Iteration    24 | Cost: 6.642594e-02
Iteration    25 | Cost: 6.642439e-02
Iteration    26 | Cost: 6.642057e-02
Iteration    27 | Cost: 6.641177e-02
Iteration    28 | Cost: 6.639360e-02
Iteration    29 | Cost: 6.636870e-02
Iteration    30 | Cost: 6.634901e-02
Iteration    31 | Cost: 6.633095e-02
Iteration    32 | Cost: 6.626401e-02
Iteration    33 | Cost: 6.618462e-02
Iteration    34 | Cost: 6.617562e-02
Iteration    35 | Cost: 6.617390e-02
Iteration    36 | Cost: 6.617273e-02
Iteration    37 | Cost: 6.617170e-02
Iteration    38 | Cost: 6.616450e-02
Iteration    39 | Cost: 6.616355e-02
Iteration    40 | Cost: 6.616287e-02
Iteration    41 | Cost: 6.616230e-02
```

```
Iteration    42 | Cost: 6.616198e-02
Iteration    43 | Cost: 6.616187e-02
Iteration    44 | Cost: 6.616147e-02
Iteration    45 | Cost: 6.616108e-02
Iteration    46 | Cost: 6.615995e-02
Iteration    47 | Cost: 6.615251e-02
Iteration    48 | Cost: 6.612901e-02
Iteration    49 | Cost: 6.610427e-02
Iteration    50 | Cost: 6.609196e-02
Iteration    51 | Cost: 6.608307e-02
Iteration    52 | Cost: 6.608287e-02
Iteration    53 | Cost: 6.608215e-02
Iteration    54 | Cost: 6.608144e-02
Iteration    55 | Cost: 6.608109e-02
Iteration    56 | Cost: 6.608104e-02
Iteration    57 | Cost: 6.608102e-02
Iteration    58 | Cost: 6.608095e-02
Iteration    59 | Cost: 6.608055e-02
Iteration    60 | Cost: 6.608018e-02
Iteration    61 | Cost: 6.607912e-02
Iteration    62 | Cost: 6.607547e-02
Iteration    63 | Cost: 6.607350e-02
Iteration    64 | Cost: 6.607172e-02
Iteration    65 | Cost: 6.607161e-02
Iteration    66 | Cost: 6.607159e-02
Iteration    67 | Cost: 6.607156e-02
Iteration    68 | Cost: 6.607156e-02
Iteration    69 | Cost: 6.607156e-02
Iteration    70 | Cost: 6.607156e-02
Iteration    71 | Cost: 6.607156e-02
Iteration    72 | Cost: 6.607156e-02
Iteration    73 | Cost: 6.607156e-02
Iteration    74 | Cost: 6.607156e-02
Iteration    75 | Cost: 6.607155e-02
```

```
Iteration    76 | Cost: 6.607154e-02
Iteration    77 | Cost: 6.607154e-02
Iteration    78 | Cost: 6.607150e-02
Iteration    79 | Cost: 6.607149e-02
Iteration    80 | Cost: 6.607148e-02
Iteration    81 | Cost: 6.607147e-02
Iteration    82 | Cost: 6.607147e-02
Iteration    83 | Cost: 6.607147e-02
Iteration    84 | Cost: 6.607147e-02
Iteration    85 | Cost: 6.607147e-02
Iteration    86 | Cost: 6.607146e-02
Iteration    87 | Cost: 6.607145e-02
Iteration    88 | Cost: 6.607145e-02
Iteration    89 | Cost: 6.607144e-02
Iteration    90 | Cost: 6.607143e-02
Iteration    91 | Cost: 6.607143e-02
Iteration    92 | Cost: 6.607143e-02
Iteration    93 | Cost: 6.607143e-02
Iteration    94 | Cost: 6.607143e-02
Iteration    95 | Cost: 6.607143e-02
Iteration    96 | Cost: 6.607143e-02
Iteration    97 | Cost: 6.607143e-02
Iteration    98 | Cost: 6.607143e-02
Iteration    99 | Cost: 6.607143e-02
Iteration   100 | Cost: 6.607142e-02
Iteration   101 | Cost: 6.607141e-02
Iteration   102 | Cost: 6.607139e-02
Iteration   103 | Cost: 6.607138e-02
Iteration   104 | Cost: 6.607138e-02
Iteration   105 | Cost: 6.607138e-02
Iteration   106 | Cost: 6.607138e-02
Iteration   107 | Cost: 6.607138e-02
Iteration   108 | Cost: 6.607138e-02
Iteration   109 | Cost: 6.607138e-02
```

```
Iteration   110 | Cost: 6.607138e-02
Iteration   111 | Cost: 6.607138e-02
Iteration   112 | Cost: 6.607138e-02
Iteration   113 | Cost: 6.607138e-02
Iteration   114 | Cost: 6.607138e-02
Iteration   115 | Cost: 6.607138e-02
Iteration   116 | Cost: 6.607138e-02
Iteration     1 | Cost: 9.871955e-01
Iteration     2 | Cost: 3.772825e-01
Iteration     3 | Cost: 2.193567e-01
Iteration     4 | Cost: 1.925157e-01
Iteration     5 | Cost: 1.805322e-01
Iteration     6 | Cost: 1.737281e-01
Iteration     7 | Cost: 1.698168e-01
Iteration     8 | Cost: 1.698001e-01
Iteration     9 | Cost: 1.697665e-01
Iteration    10 | Cost: 1.696779e-01
Iteration    11 | Cost: 1.696638e-01
Iteration    12 | Cost: 1.691695e-01
Iteration    13 | Cost: 1.691657e-01
Iteration    14 | Cost: 1.691603e-01
Iteration    15 | Cost: 1.691490e-01
Iteration    16 | Cost: 1.691438e-01
Iteration    17 | Cost: 1.691410e-01
Iteration    18 | Cost: 1.691401e-01
Iteration    19 | Cost: 1.691388e-01
Iteration    20 | Cost: 1.691370e-01
Iteration    21 | Cost: 1.691346e-01
Iteration    22 | Cost: 1.691328e-01
Iteration    23 | Cost: 1.691319e-01
Iteration    24 | Cost: 1.691266e-01
Iteration    25 | Cost: 1.691190e-01
Iteration    26 | Cost: 1.691153e-01
Iteration    27 | Cost: 1.691149e-01
```

```
Iteration    28 | Cost: 1.691141e-01
Iteration    29 | Cost: 1.691135e-01
Iteration    30 | Cost: 1.691133e-01
Iteration    31 | Cost: 1.691080e-01
Iteration    32 | Cost: 1.691029e-01
Iteration    33 | Cost: 1.690958e-01
Iteration    34 | Cost: 1.690955e-01
Iteration    35 | Cost: 1.690952e-01
Iteration    36 | Cost: 1.690952e-01
Iteration    37 | Cost: 1.690946e-01
Iteration    38 | Cost: 1.690945e-01
Iteration    39 | Cost: 1.690944e-01
Iteration    40 | Cost: 1.690944e-01
Iteration    41 | Cost: 1.690944e-01
Iteration    42 | Cost: 1.690944e-01
Iteration    43 | Cost: 1.690944e-01
Iteration    44 | Cost: 1.690944e-01
Iteration    45 | Cost: 1.690944e-01
Iteration    46 | Cost: 1.690944e-01
Iteration    47 | Cost: 1.690944e-01
Iteration    48 | Cost: 1.690944e-01
Iteration    49 | Cost: 1.690944e-01
Iteration    50 | Cost: 1.690944e-01
Iteration    51 | Cost: 1.690944e-01
Iteration    52 | Cost: 1.690944e-01
Iteration    53 | Cost: 1.690944e-01
Iteration    54 | Cost: 1.690944e-01
Iteration    55 | Cost: 1.690944e-01
Iteration    56 | Cost: 1.690944e-01
Iteration    57 | Cost: 1.690944e-01
Iteration    58 | Cost: 1.690944e-01
Iteration    59 | Cost: 1.690944e-01
Iteration    60 | Cost: 1.690944e-01
Iteration    61 | Cost: 1.690944e-01
```

```
Iteration    62 | Cost: 1.690944e-01
Iteration    63 | Cost: 1.690944e-01
Iteration    64 | Cost: 1.690944e-01
Iteration    65 | Cost: 1.690944e-01
Iteration    66 | Cost: 1.690944e-01
Iteration    67 | Cost: 1.690944e-01
Iteration    68 | Cost: 1.690944e-01
Iteration    69 | Cost: 1.690944e-01
Iteration    70 | Cost: 1.690944e-01
Iteration    71 | Cost: 1.690944e-01
Iteration    72 | Cost: 1.690944e-01
Iteration    73 | Cost: 1.690944e-01
Iteration    74 | Cost: 1.690944e-01
Iteration    75 | Cost: 1.690944e-01
Iteration    76 | Cost: 1.690944e-01
Iteration    77 | Cost: 1.690944e-01
Iteration    78 | Cost: 1.690944e-01
Iteration    79 | Cost: 1.690944e-01
Iteration    80 | Cost: 1.690944e-01
Iteration    81 | Cost: 1.690944e-01
Iteration    82 | Cost: 1.690944e-01
Iteration    83 | Cost: 1.690944e-01
Iteration    84 | Cost: 1.690944e-01
Iteration    85 | Cost: 1.690944e-01
Iteration    86 | Cost: 1.690944e-01
Iteration    87 | Cost: 1.690944e-01
Iteration    88 | Cost: 1.690944e-01
Iteration    89 | Cost: 1.690944e-01
Iteration    90 | Cost: 1.690944e-01
Iteration    91 | Cost: 1.690944e-01
Iteration    92 | Cost: 1.690944e-01
Iteration    93 | Cost: 1.690944e-01
Iteration    94 | Cost: 1.690944e-01
Iteration    95 | Cost: 1.690944e-01
```

```
Iteration    96 | Cost: 1.690944e-01
Iteration    97 | Cost: 1.690944e-01
Iteration    98 | Cost: 1.690944e-01
Iteration    99 | Cost: 1.690944e-01
Iteration   100 | Cost: 1.690944e-01
Iteration   101 | Cost: 1.690944e-01
Iteration   102 | Cost: 1.690944e-01
Iteration   103 | Cost: 1.690944e-01
Iteration   104 | Cost: 1.690944e-01
Iteration   105 | Cost: 1.690944e-01
Iteration   106 | Cost: 1.690944e-01
Iteration   107 | Cost: 1.690944e-01
Iteration   108 | Cost: 1.690944e-01
Iteration   109 | Cost: 1.690944e-01
Iteration   110 | Cost: 1.690944e-01
Iteration   111 | Cost: 1.690944e-01
Iteration   112 | Cost: 1.690944e-01
Iteration   113 | Cost: 1.690944e-01
Iteration   114 | Cost: 1.690944e-01
Iteration   115 | Cost: 1.690944e-01
Iteration   116 | Cost: 1.690944e-01
Iteration   117 | Cost: 1.690944e-01
Iteration   118 | Cost: 1.690944e-01
Iteration   119 | Cost: 1.690944e-01
Iteration   120 | Cost: 1.690944e-01
Iteration   121 | Cost: 1.690944e-01
Iteration   122 | Cost: 1.690944e-01
Iteration   123 | Cost: 1.690944e-01
Iteration   124 | Cost: 1.690944e-01
Iteration   125 | Cost: 1.690944e-01
Iteration   126 | Cost: 1.690944e-01
Iteration   127 | Cost: 1.690944e-01
Iteration   128 | Cost: 1.690944e-01
Iteration   129 | Cost: 1.690944e-01
```

```
Iteration   130 | Cost: 1.690944e-01
Iteration   131 | Cost: 1.690944e-01
Iteration   132 | Cost: 1.690944e-01
Iteration   133 | Cost: 1.690944e-01
Iteration   134 | Cost: 1.690944e-01
Iteration   135 | Cost: 1.690944e-01
Iteration   136 | Cost: 1.690944e-01
Iteration   137 | Cost: 1.690944e-01
Iteration   138 | Cost: 1.690944e-01
Iteration   139 | Cost: 1.690944e-01
Iteration   140 | Cost: 1.690944e-01
Iteration   141 | Cost: 1.690944e-01
Iteration   142 | Cost: 1.690944e-01
Iteration   143 | Cost: 1.690944e-01
Iteration   144 | Cost: 1.690944e-01
Iteration   145 | Cost: 1.690944e-01
Iteration   146 | Cost: 1.690944e-01
Iteration   147 | Cost: 1.690944e-01
Iteration   148 | Cost: 1.690944e-01
Iteration   149 | Cost: 1.690944e-01
Iteration   150 | Cost: 1.690944e-01
Iteration   151 | Cost: 1.690944e-01
Iteration   152 | Cost: 1.690944e-01
Iteration   153 | Cost: 1.690944e-01
Iteration   155 | Cost: 1.690944e-01
Iteration   156 | Cost: 1.690944e-01
Iteration   157 | Cost: 1.690944e-01
Iteration   158 | Cost: 1.690944e-01
Iteration   159 | Cost: 1.690944e-01
Iteration   160 | Cost: 1.690944e-01
Iteration   161 | Cost: 1.690944e-01
Iteration   162 | Cost: 1.690944e-01
Iteration   163 | Cost: 1.690944e-01
Iteration   165 | Cost: 1.690944e-01
```

```
Iteration     1 | Cost: 9.154177e+01
Iteration     2 | Cost: 2.731395e+01
Iteration     3 | Cost: 2.459108e+00
Iteration     4 | Cost: 1.223656e+00
Iteration     5 | Cost: 7.220077e-01
Iteration     6 | Cost: 4.937379e-01
Iteration     7 | Cost: 4.847767e-01
Iteration     8 | Cost: 4.769500e-01
Iteration     9 | Cost: 4.710355e-01
Iteration    10 | Cost: 4.658126e-01
Iteration    11 | Cost: 4.567342e-01
Iteration    12 | Cost: 4.483295e-01
Iteration    13 | Cost: 4.352336e-01
Iteration    14 | Cost: 4.337936e-01
Iteration    15 | Cost: 4.320122e-01
Iteration    16 | Cost: 4.315520e-01
Iteration    17 | Cost: 4.257567e-01
Iteration    18 | Cost: 4.246872e-01
Iteration    19 | Cost: 4.218415e-01
Iteration    20 | Cost: 4.206777e-01
Iteration    21 | Cost: 4.190893e-01
Iteration    22 | Cost: 4.189428e-01
Iteration    23 | Cost: 4.183320e-01
Iteration    24 | Cost: 4.180058e-01
Iteration    25 | Cost: 4.173922e-01
Iteration    26 | Cost: 4.156960e-01
Iteration    27 | Cost: 4.152327e-01
Iteration    28 | Cost: 4.151070e-01
Iteration    29 | Cost: 4.147728e-01
Iteration    30 | Cost: 4.116337e-01
Iteration    31 | Cost: 4.100717e-01
Iteration    32 | Cost: 4.035485e-01
Iteration    33 | Cost: 4.029565e-01
Iteration    34 | Cost: 4.020798e-01
```

```
Iteration    35 | Cost: 4.016623e-01
Iteration    36 | Cost: 4.014125e-01
Iteration    37 | Cost: 4.011556e-01
Iteration    38 | Cost: 4.011193e-01
Iteration    39 | Cost: 4.006432e-01
Iteration    40 | Cost: 4.006045e-01
Iteration    41 | Cost: 4.004508e-01
Iteration    42 | Cost: 4.004403e-01
Iteration    43 | Cost: 4.001150e-01
Iteration    44 | Cost: 4.000050e-01
Iteration    45 | Cost: 3.995455e-01
Iteration    46 | Cost: 3.994510e-01
Iteration    47 | Cost: 3.993362e-01
Iteration    48 | Cost: 3.993177e-01
Iteration    49 | Cost: 3.992968e-01
Iteration    50 | Cost: 3.992361e-01
Iteration    51 | Cost: 3.990850e-01
Iteration    52 | Cost: 3.987626e-01
Iteration    53 | Cost: 3.986615e-01
Iteration    54 | Cost: 3.985502e-01
Iteration    55 | Cost: 3.985311e-01
Iteration    56 | Cost: 3.984787e-01
Iteration    57 | Cost: 3.984075e-01
Iteration    58 | Cost: 3.983329e-01
Iteration    59 | Cost: 3.981891e-01
Iteration    60 | Cost: 3.981562e-01
Iteration    61 | Cost: 3.980533e-01
Iteration    62 | Cost: 3.979801e-01
Iteration    63 | Cost: 3.979633e-01
Iteration    64 | Cost: 3.979532e-01
Iteration    65 | Cost: 3.979508e-01
Iteration    66 | Cost: 3.979502e-01
Iteration    67 | Cost: 3.979496e-01
Iteration    68 | Cost: 3.979473e-01
```

```
Iteration    69 | Cost: 3.979438e-01
Iteration    70 | Cost: 3.979387e-01
Iteration    71 | Cost: 3.979336e-01
Iteration    72 | Cost: 3.979307e-01
Iteration    73 | Cost: 3.979300e-01
Iteration    74 | Cost: 3.979291e-01
Iteration    75 | Cost: 3.979285e-01
Iteration    76 | Cost: 3.979233e-01
Iteration    77 | Cost: 3.979209e-01
Iteration    78 | Cost: 3.979118e-01
Iteration    79 | Cost: 3.979090e-01
Iteration    80 | Cost: 3.979060e-01
Iteration    81 | Cost: 3.979054e-01
Iteration    82 | Cost: 3.979048e-01
Iteration    83 | Cost: 3.978988e-01
Iteration    84 | Cost: 3.978967e-01
Iteration    85 | Cost: 3.978791e-01
Iteration    86 | Cost: 3.978741e-01
Iteration    87 | Cost: 3.978695e-01
Iteration    88 | Cost: 3.978688e-01
Iteration    89 | Cost: 3.978654e-01
Iteration    90 | Cost: 3.978648e-01
Iteration    91 | Cost: 3.978643e-01
Iteration    92 | Cost: 3.978617e-01
Iteration    93 | Cost: 3.978611e-01
Iteration    94 | Cost: 3.978604e-01
Iteration    95 | Cost: 3.978603e-01
Iteration    96 | Cost: 3.978600e-01
Iteration    97 | Cost: 3.978600e-01
Iteration    98 | Cost: 3.978597e-01
Iteration    99 | Cost: 3.978591e-01
Iteration   100 | Cost: 3.978589e-01
Iteration   101 | Cost: 3.978581e-01
Iteration   102 | Cost: 3.978579e-01
```

```
Iteration   103 | Cost: 3.978576e-01
Iteration   104 | Cost: 3.978576e-01
Iteration   105 | Cost: 3.978575e-01
Iteration   106 | Cost: 3.978573e-01
Iteration   107 | Cost: 3.978562e-01
Iteration   108 | Cost: 3.978562e-01
Iteration   109 | Cost: 3.978562e-01
Iteration   110 | Cost: 3.978562e-01
Iteration   111 | Cost: 3.978560e-01
Iteration   112 | Cost: 3.978560e-01
Iteration   113 | Cost: 3.978558e-01
Iteration   114 | Cost: 3.978556e-01
Iteration   115 | Cost: 3.978555e-01
Iteration   116 | Cost: 3.978554e-01
Iteration   117 | Cost: 3.978554e-01
Iteration   118 | Cost: 3.978554e-01
Iteration   119 | Cost: 3.978554e-01
Iteration   120 | Cost: 3.978554e-01
Iteration   121 | Cost: 3.978553e-01
Iteration   122 | Cost: 3.978552e-01
Iteration   123 | Cost: 3.978552e-01
Iteration   124 | Cost: 3.978549e-01
Iteration   125 | Cost: 3.978549e-01
Iteration   126 | Cost: 3.978548e-01
Iteration   127 | Cost: 3.978548e-01
Iteration   128 | Cost: 3.978545e-01
Iteration   129 | Cost: 3.978544e-01
Iteration   130 | Cost: 3.978543e-01
Iteration   131 | Cost: 3.978543e-01
Iteration   132 | Cost: 3.978543e-01
Iteration   133 | Cost: 3.978543e-01
Iteration   134 | Cost: 3.978542e-01
Iteration   135 | Cost: 3.978542e-01
Iteration   136 | Cost: 3.978541e-01
```

```
Iteration   137 | Cost: 3.978541e-01
Iteration   138 | Cost: 3.978541e-01
Iteration   139 | Cost: 3.978541e-01
Iteration   140 | Cost: 3.978541e-01
Iteration   141 | Cost: 3.978541e-01
Iteration   142 | Cost: 3.978541e-01
Iteration   143 | Cost: 3.978541e-01
Iteration   144 | Cost: 3.978541e-01
Iteration   145 | Cost: 3.978541e-01
Iteration   146 | Cost: 3.978541e-01
Iteration   147 | Cost: 3.978541e-01
Iteration   148 | Cost: 3.978541e-01
Iteration   149 | Cost: 3.978541e-01
Iteration   150 | Cost: 3.978541e-01
Iteration   151 | Cost: 3.978541e-01
Iteration   152 | Cost: 3.978540e-01
Iteration   153 | Cost: 3.978540e-01
Iteration   154 | Cost: 3.978540e-01
Iteration   155 | Cost: 3.978540e-01
Iteration   156 | Cost: 3.978540e-01
Iteration   157 | Cost: 3.978540e-01
Iteration   158 | Cost: 3.978540e-01
Iteration   159 | Cost: 3.978540e-01
Iteration   160 | Cost: 3.978540e-01
Iteration   161 | Cost: 3.978540e-01
Iteration   162 | Cost: 3.978540e-01
Iteration   163 | Cost: 3.978540e-01
Iteration   164 | Cost: 3.978540e-01
Iteration   165 | Cost: 3.978540e-01
Iteration   166 | Cost: 3.978540e-01
Iteration   167 | Cost: 3.978540e-01
Iteration   168 | Cost: 3.978540e-01
Iteration   169 | Cost: 3.978540e-01
Iteration   170 | Cost: 3.978540e-01
```

```
Iteration   171 | Cost: 3.978540e-01
Iteration   172 | Cost: 3.978539e-01
Iteration   173 | Cost: 3.978539e-01
Iteration   174 | Cost: 3.978539e-01
Iteration   175 | Cost: 3.978539e-01
Iteration   176 | Cost: 3.978539e-01
Iteration   177 | Cost: 3.978539e-01
Iteration   178 | Cost: 3.978539e-01
Iteration   179 | Cost: 3.978539e-01
Iteration   180 | Cost: 3.978539e-01
Iteration   181 | Cost: 3.978539e-01
Iteration   182 | Cost: 3.978539e-01
Iteration   183 | Cost: 3.978539e-01
Iteration   184 | Cost: 3.978539e-01
Iteration   185 | Cost: 3.978539e-01
Iteration   186 | Cost: 3.978539e-01
Iteration   187 | Cost: 3.978539e-01
Iteration   188 | Cost: 3.978539e-01
Iteration   189 | Cost: 3.978539e-01
Iteration   190 | Cost: 3.978539e-01
Iteration   191 | Cost: 3.978539e-01
Iteration   192 | Cost: 3.978539e-01
Iteration   193 | Cost: 3.978539e-01
Iteration   194 | Cost: 3.978539e-01
Iteration   195 | Cost: 3.978539e-01
Iteration   196 | Cost: 3.978539e-01
Iteration   197 | Cost: 3.978539e-01
Iteration   198 | Cost: 3.978539e-01
Iteration   199 | Cost: 3.978539e-01
Iteration   200 | Cost: 3.978539e-01
Iteration     1 | Cost: 6.728733e+01
Iteration     2 | Cost: 1.310440e+01
Iteration     3 | Cost: 3.015205e+00
Iteration     4 | Cost: 1.732456e+00
```

```
Iteration     5 | Cost: 1.329985e+00
Iteration     6 | Cost: 1.165647e+00
Iteration     7 | Cost: 1.164835e+00
Iteration     8 | Cost: 1.162783e+00
Iteration     9 | Cost: 1.156619e+00
Iteration    10 | Cost: 1.154833e+00
Iteration    11 | Cost: 1.147953e+00
Iteration    12 | Cost: 1.147653e+00
Iteration    13 | Cost: 1.147348e+00
Iteration    14 | Cost: 1.146999e+00
Iteration    15 | Cost: 1.146595e+00
Iteration    16 | Cost: 1.145325e+00
Iteration    17 | Cost: 1.142868e+00
Iteration    18 | Cost: 1.141438e+00
Iteration    19 | Cost: 1.140517e+00
Iteration    20 | Cost: 1.139175e+00
Iteration    21 | Cost: 1.139113e+00
Iteration    22 | Cost: 1.138331e+00
Iteration    23 | Cost: 1.138067e+00
Iteration    24 | Cost: 1.137172e+00
Iteration    25 | Cost: 1.137007e+00
Iteration    26 | Cost: 1.136795e+00
Iteration    27 | Cost: 1.136759e+00
Iteration    28 | Cost: 1.136605e+00
Iteration    29 | Cost: 1.136378e+00
Iteration    30 | Cost: 1.136074e+00
Iteration    31 | Cost: 1.135940e+00
Iteration    32 | Cost: 1.135862e+00
Iteration    33 | Cost: 1.135073e+00
Iteration    34 | Cost: 1.134353e+00
Iteration    35 | Cost: 1.134230e+00
Iteration    36 | Cost: 1.134212e+00
Iteration    37 | Cost: 1.134203e+00
Iteration    38 | Cost: 1.134202e+00
```

```
Iteration    39 | Cost: 1.134197e+00
Iteration    40 | Cost: 1.134187e+00
Iteration    41 | Cost: 1.134184e+00
Iteration    42 | Cost: 1.134170e+00
Iteration    43 | Cost: 1.134167e+00
Iteration    44 | Cost: 1.134164e+00
Iteration    45 | Cost: 1.134163e+00
Iteration    46 | Cost: 1.134163e+00
Iteration    47 | Cost: 1.134161e+00
Iteration    48 | Cost: 1.134157e+00
Iteration    49 | Cost: 1.134156e+00
Iteration    50 | Cost: 1.134156e+00
Iteration    51 | Cost: 1.134156e+00
Iteration    52 | Cost: 1.134156e+00
Iteration    53 | Cost: 1.134155e+00
Iteration    54 | Cost: 1.134155e+00
Iteration    55 | Cost: 1.134155e+00
Iteration    56 | Cost: 1.134155e+00
Iteration    57 | Cost: 1.134155e+00
Iteration    58 | Cost: 1.134155e+00
Iteration    59 | Cost: 1.134154e+00
Iteration    60 | Cost: 1.134154e+00
Iteration    61 | Cost: 1.134154e+00
Iteration    62 | Cost: 1.134154e+00
Iteration    63 | Cost: 1.134154e+00
Iteration    64 | Cost: 1.134154e+00
Iteration    65 | Cost: 1.134154e+00
Iteration    66 | Cost: 1.134154e+00
Iteration    67 | Cost: 1.134154e+00
Iteration    68 | Cost: 1.134154e+00
Iteration    69 | Cost: 1.134154e+00
Iteration    70 | Cost: 1.134154e+00
Iteration    71 | Cost: 1.134154e+00
Iteration    72 | Cost: 1.134154e+00
```

```
Iteration    73 | Cost: 1.134154e+00
Iteration    74 | Cost: 1.134154e+00
Iteration    75 | Cost: 1.134154e+00
Iteration    76 | Cost: 1.134154e+00
Iteration    77 | Cost: 1.134154e+00
Iteration    78 | Cost: 1.134154e+00
Iteration    79 | Cost: 1.134154e+00
Iteration    80 | Cost: 1.134154e+00
Iteration    81 | Cost: 1.134154e+00
Iteration    82 | Cost: 1.134154e+00
Iteration    83 | Cost: 1.134154e+00
Iteration    84 | Cost: 1.134154e+00
Iteration    85 | Cost: 1.134154e+00
Iteration    86 | Cost: 1.134154e+00
Iteration    87 | Cost: 1.134154e+00
Iteration    88 | Cost: 1.134154e+00
Iteration    89 | Cost: 1.134154e+00
Iteration    90 | Cost: 1.134154e+00
Iteration    91 | Cost: 1.134154e+00
Iteration    92 | Cost: 1.134154e+00
Iteration    93 | Cost: 1.134154e+00
Iteration    94 | Cost: 1.134154e+00
Iteration    95 | Cost: 1.134154e+00
Iteration    96 | Cost: 1.134154e+00
Iteration    97 | Cost: 1.134154e+00
Iteration    98 | Cost: 1.134154e+00
Iteration    99 | Cost: 1.134154e+00
Iteration   100 | Cost: 1.134154e+00
Iteration   101 | Cost: 1.134154e+00
Iteration   102 | Cost: 1.134154e+00
Iteration   103 | Cost: 1.134154e+00
Iteration   104 | Cost: 1.134154e+00
Iteration   105 | Cost: 1.134154e+00
Iteration   106 | Cost: 1.134154e+00
```

```
Iteration   107 | Cost: 1.134154e+00
Iteration   108 | Cost: 1.134154e+00
Iteration   109 | Cost: 1.134154e+00
Iteration   110 | Cost: 1.134154e+00
Iteration   111 | Cost: 1.134154e+00
Iteration   112 | Cost: 1.134154e+00
Iteration   113 | Cost: 1.134154e+00
Iteration   114 | Cost: 1.134154e+00
Iteration   115 | Cost: 1.134154e+00
Iteration   116 | Cost: 1.134154e+00
Iteration   117 | Cost: 1.134154e+00
Iteration   118 | Cost: 1.134154e+00
Iteration   119 | Cost: 1.134154e+00
Iteration   120 | Cost: 1.134154e+00
Iteration   121 | Cost: 1.134154e+00
Iteration   122 | Cost: 1.134154e+00
Iteration   123 | Cost: 1.134154e+00
Iteration   124 | Cost: 1.134154e+00
Iteration   125 | Cost: 1.134154e+00
Iteration   126 | Cost: 1.134154e+00
Iteration   127 | Cost: 1.134154e+00
Iteration   128 | Cost: 1.134154e+00
Iteration   129 | Cost: 1.134154e+00
Iteration     1 | Cost: 7.230457e+01
Iteration     2 | Cost: 1.079841e+01
Iteration     3 | Cost: 9.730897e+00
Iteration     4 | Cost: 5.312511e+00
Iteration     5 | Cost: 4.811858e+00
Iteration     6 | Cost: 4.122310e+00
Iteration     7 | Cost: 3.944377e+00
Iteration     8 | Cost: 3.816478e+00
Iteration     9 | Cost: 3.767234e+00
Iteration    10 | Cost: 3.703987e+00
Iteration    11 | Cost: 3.636496e+00
```

```
Iteration    12 | Cost: 3.618625e+00
Iteration    13 | Cost: 3.541669e+00
Iteration    14 | Cost: 3.534525e+00
Iteration    15 | Cost: 3.516384e+00
Iteration    16 | Cost: 3.508978e+00
Iteration    17 | Cost: 3.478955e+00
Iteration    18 | Cost: 3.447048e+00
Iteration    19 | Cost: 3.443940e+00
Iteration    20 | Cost: 3.439502e+00
Iteration    21 | Cost: 3.437469e+00
Iteration    22 | Cost: 3.436898e+00
Iteration    23 | Cost: 3.431039e+00
Iteration    24 | Cost: 3.424016e+00
Iteration    25 | Cost: 3.421937e+00
Iteration    26 | Cost: 3.420699e+00
Iteration    27 | Cost: 3.418331e+00
Iteration    28 | Cost: 3.416990e+00
Iteration    29 | Cost: 3.416425e+00
Iteration    30 | Cost: 3.416320e+00
Iteration    31 | Cost: 3.416278e+00
Iteration    32 | Cost: 3.416260e+00
Iteration    33 | Cost: 3.416213e+00
Iteration    34 | Cost: 3.416146e+00
Iteration    35 | Cost: 3.416024e+00
Iteration    36 | Cost: 3.415984e+00
Iteration    37 | Cost: 3.415952e+00
Iteration    38 | Cost: 3.415948e+00
Iteration    39 | Cost: 3.415936e+00
Iteration    40 | Cost: 3.415935e+00
Iteration    41 | Cost: 3.415929e+00
Iteration    42 | Cost: 3.415927e+00
Iteration    43 | Cost: 3.415923e+00
Iteration    44 | Cost: 3.415922e+00
Iteration    45 | Cost: 3.415919e+00
```

```
Iteration    46 | Cost: 3.415918e+00
Iteration    47 | Cost: 3.415915e+00
Iteration    48 | Cost: 3.415912e+00
Iteration    49 | Cost: 3.415912e+00
Iteration    50 | Cost: 3.415911e+00
Iteration    51 | Cost: 3.415911e+00
Iteration    52 | Cost: 3.415911e+00
Iteration    53 | Cost: 3.415911e+00
Iteration    54 | Cost: 3.415911e+00
Iteration    55 | Cost: 3.415911e+00
Iteration    56 | Cost: 3.415911e+00
Iteration    57 | Cost: 3.415910e+00
Iteration    58 | Cost: 3.415910e+00
Iteration    59 | Cost: 3.415910e+00
Iteration    60 | Cost: 3.415910e+00
Iteration    61 | Cost: 3.415910e+00
Iteration    62 | Cost: 3.415910e+00
Iteration    63 | Cost: 3.415910e+00
Iteration    64 | Cost: 3.415910e+00
Iteration    65 | Cost: 3.415910e+00
Iteration    66 | Cost: 3.415910e+00
Iteration    67 | Cost: 3.415910e+00
Iteration    68 | Cost: 3.415910e+00
Iteration    69 | Cost: 3.415910e+00
Iteration    70 | Cost: 3.415910e+00
Iteration    71 | Cost: 3.415910e+00
Iteration    72 | Cost: 3.415910e+00
Iteration    73 | Cost: 3.415910e+00
Iteration    74 | Cost: 3.415910e+00
Iteration    75 | Cost: 3.415910e+00
Iteration    76 | Cost: 3.415910e+00
Iteration    77 | Cost: 3.415910e+00
Iteration    78 | Cost: 3.415910e+00
Iteration    79 | Cost: 3.415910e+00
```

```
Iteration    80 | Cost: 3.415910e+00
Iteration    81 | Cost: 3.415910e+00
Iteration    82 | Cost: 3.415910e+00
Iteration    83 | Cost: 3.415910e+00
Iteration    84 | Cost: 3.415910e+00
Iteration    85 | Cost: 3.415910e+00
Iteration    86 | Cost: 3.415910e+00
Iteration    87 | Cost: 3.415910e+00
Iteration    88 | Cost: 3.415910e+00
Iteration    89 | Cost: 3.415910e+00
Iteration    90 | Cost: 3.415910e+00
Iteration    91 | Cost: 3.415910e+00
Iteration    92 | Cost: 3.415910e+00
Iteration    93 | Cost: 3.415910e+00
Iteration    94 | Cost: 3.415910e+00
Iteration    95 | Cost: 3.415910e+00
Iteration    96 | Cost: 3.415910e+00
Iteration    97 | Cost: 3.415910e+00
Iteration    98 | Cost: 3.415910e+00
Iteration    99 | Cost: 3.415910e+00
Iteration   100 | Cost: 3.415910e+00
Iteration   101 | Cost: 3.415910e+00
Iteration   102 | Cost: 3.415910e+00
Iteration   103 | Cost: 3.415910e+00
Iteration   104 | Cost: 3.415910e+00
Iteration   105 | Cost: 3.415910e+00
Iteration   106 | Cost: 3.415910e+00
Iteration   108 | Cost: 3.415910e+00
Iteration   109 | Cost: 3.415910e+00
Iteration     1 | Cost: 6.611004e+01
Iteration     2 | Cost: 1.227749e+01
Iteration     3 | Cost: 1.079462e+01
Iteration     4 | Cost: 8.838726e+00
Iteration     5 | Cost: 8.699382e+00
```

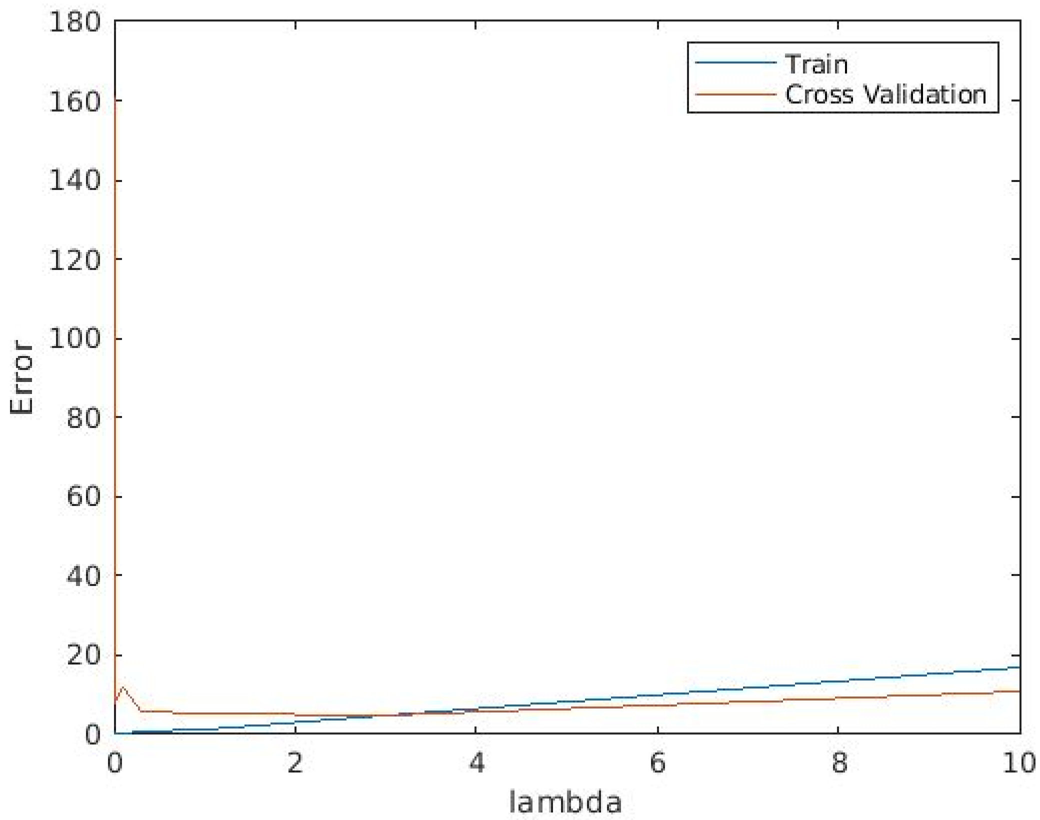
```
Iteration     6 | Cost: 8.507287e+00
Iteration     7 | Cost: 8.364305e+00
Iteration     8 | Cost: 8.333738e+00
Iteration     9 | Cost: 8.291157e+00
Iteration    10 | Cost: 8.288967e+00
Iteration    11 | Cost: 8.279162e+00
Iteration    12 | Cost: 8.274785e+00
Iteration    13 | Cost: 8.265375e+00
Iteration    14 | Cost: 8.258230e+00
Iteration    15 | Cost: 8.257980e+00
Iteration    16 | Cost: 8.257039e+00
Iteration    17 | Cost: 8.256708e+00
Iteration    18 | Cost: 8.256325e+00
Iteration    19 | Cost: 8.256301e+00
Iteration    20 | Cost: 8.256290e+00
Iteration    21 | Cost: 8.256253e+00
Iteration    22 | Cost: 8.256243e+00
Iteration    23 | Cost: 8.256231e+00
Iteration    24 | Cost: 8.256230e+00
Iteration    25 | Cost: 8.256227e+00
Iteration    26 | Cost: 8.256225e+00
Iteration    27 | Cost: 8.256225e+00
Iteration    28 | Cost: 8.256224e+00
Iteration    29 | Cost: 8.256224e+00
Iteration    30 | Cost: 8.256224e+00
Iteration    31 | Cost: 8.256224e+00
Iteration    32 | Cost: 8.256224e+00
Iteration    33 | Cost: 8.256224e+00
Iteration    34 | Cost: 8.256224e+00
Iteration    35 | Cost: 8.256224e+00
Iteration    36 | Cost: 8.256224e+00
Iteration    37 | Cost: 8.256224e+00
Iteration    38 | Cost: 8.256224e+00
Iteration    39 | Cost: 8.256224e+00
```

```
Iteration    40 | Cost: 8.256224e+00
Iteration    41 | Cost: 8.256224e+00
Iteration    42 | Cost: 8.256224e+00
Iteration    43 | Cost: 8.256224e+00
Iteration    44 | Cost: 8.256224e+00
Iteration    45 | Cost: 8.256224e+00
Iteration    46 | Cost: 8.256224e+00
Iteration    47 | Cost: 8.256224e+00
Iteration    48 | Cost: 8.256224e+00
Iteration    49 | Cost: 8.256224e+00
Iteration    50 | Cost: 8.256224e+00
Iteration    51 | Cost: 8.256224e+00
Iteration    52 | Cost: 8.256224e+00
Iteration    53 | Cost: 8.256224e+00
Iteration    54 | Cost: 8.256224e+00
Iteration    55 | Cost: 8.256224e+00
Iteration     1 | Cost: 6.775954e+01
Iteration     2 | Cost: 2.385571e+01
Iteration     3 | Cost: 2.181636e+01
Iteration     4 | Cost: 1.941607e+01
Iteration     5 | Cost: 1.878074e+01
Iteration     6 | Cost: 1.857917e+01
Iteration     7 | Cost: 1.855924e+01
Iteration     8 | Cost: 1.853781e+01
Iteration     9 | Cost: 1.851653e+01
Iteration    10 | Cost: 1.851588e+01
Iteration    11 | Cost: 1.851509e+01
Iteration    12 | Cost: 1.851505e+01
Iteration    13 | Cost: 1.851494e+01
Iteration    14 | Cost: 1.851492e+01
Iteration    15 | Cost: 1.851491e+01
Iteration    16 | Cost: 1.851491e+01
Iteration    17 | Cost: 1.851490e+01
Iteration    18 | Cost: 1.851490e+01
```

```
Iteration    19 | Cost: 1.851490e+01
Iteration    20 | Cost: 1.851490e+01
Iteration    21 | Cost: 1.851490e+01
Iteration    22 | Cost: 1.851490e+01
Iteration    23 | Cost: 1.851490e+01
Iteration    24 | Cost: 1.851490e+01
Iteration    25 | Cost: 1.851490e+01
Iteration    26 | Cost: 1.851490e+01
Iteration    27 | Cost: 1.851490e+01
Iteration    28 | Cost: 1.851490e+01
Iteration    29 | Cost: 1.851490e+01
Iteration    30 | Cost: 1.851490e+01
Iteration     1 | Cost: 8.510592e+01
Iteration     2 | Cost: 6.030564e+01
Iteration     3 | Cost: 5.331152e+01
Iteration     4 | Cost: 4.305291e+01
Iteration     5 | Cost: 3.886919e+01
Iteration     6 | Cost: 3.697146e+01
Iteration     7 | Cost: 3.665382e+01
Iteration     8 | Cost: 3.661837e+01
Iteration     9 | Cost: 3.660956e+01
Iteration    10 | Cost: 3.660738e+01
Iteration    11 | Cost: 3.660677e+01
Iteration    12 | Cost: 3.660672e+01
Iteration    13 | Cost: 3.660669e+01
Iteration    14 | Cost: 3.660668e+01
Iteration    15 | Cost: 3.660667e+01
Iteration    16 | Cost: 3.660667e+01
Iteration    17 | Cost: 3.660667e+01
Iteration    18 | Cost: 3.660667e+01
Iteration    19 | Cost: 3.660667e+01
Iteration    20 | Cost: 3.660667e+01
Iteration    21 | Cost: 3.660667e+01
Iteration    22 | Cost: 3.660667e+01
```

```
Iteration    23 | Cost: 3.660667e+01
Iteration    25 | Cost: 3.660667e+01
Iteration    26 | Cost: 3.660667e+01
Iteration    28 | Cost: 3.660667e+01
Iteration    29 | Cost: 3.660667e+01
Iteration    30 | Cost: 3.660667e+01
```

```
plot(lambda_vec, error_train, lambda_vec, error_val);
legend('Train', 'Cross Validation');
xlabel('lambda');
ylabel('Error');
```




```
for i = 1:length(lambda_vec)
   if i == 1
       fprintf('lambda\t\tTrain Error\tValidation Error\n');
   end
   fprintf('%f\t%f\t%f\n',lambda_vec(i), error_train(i),
error_val(i));
```

```
end
```

lambda	Train Error	Validation Error
0.000000	0.000000	160.721900
0.001000	0.000000	143.551027
0.003000	0.000040	11.099141
0.010000	0.002759	7.815737
0.030000	0.007479	9.272094
0.100000	0.048222	11.852221
0.300000	0.551727	5.961587
1.000000	1.422968	5.516444
3.000000	4.641369	4.792851
10.000000	16.994847	10.957743

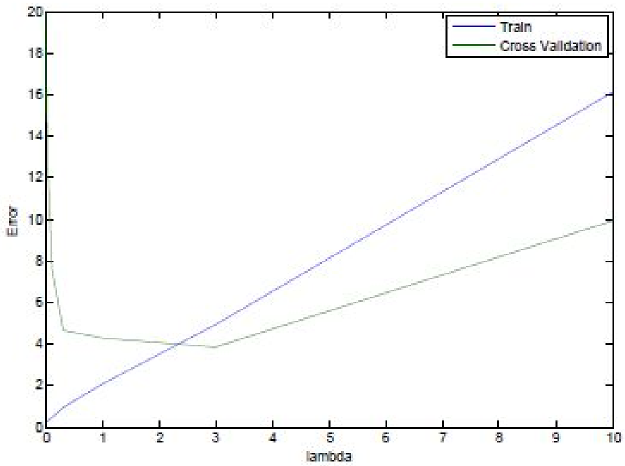


Figure 9: Selecting λ using a cross validation set

In this figure, we can see that the best value of λ is around 3. Due to randomness in the training and validation splits of the dataset, the cross validation error can sometimes be lower than the training error.

You should now submit your solutions. Enter **`submit`** *at the command prompt, then enter or confirm your login and token when prompted.*

3.4 Optional (ungraded) exercise: Computing test set error

In the previous part of the exercise, you implemented code to compute the cross validation error for various values of the regularization parameter λ. However, to get a better indication of the model's performance in the real world, it is important to evaluate the 'final' model on a test set that was not used in any part of training (that is, it was neither used to select the λ parameters, nor to learn the model parameters θ).

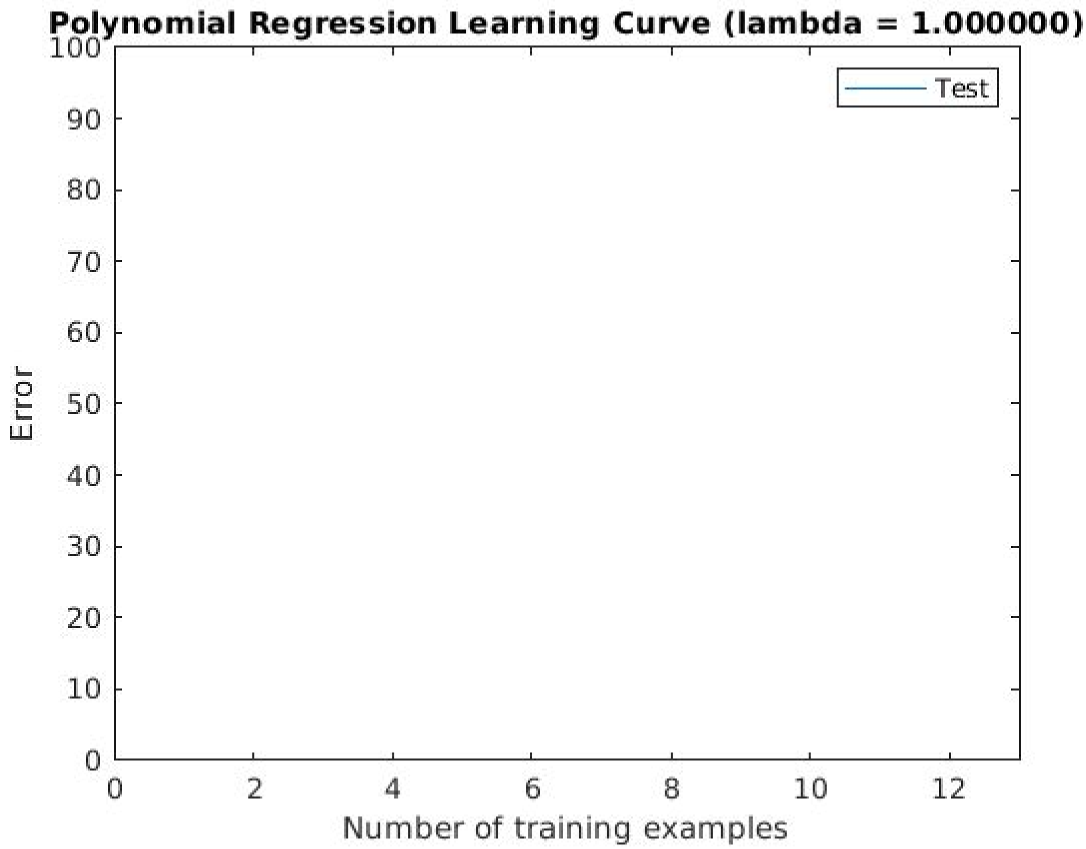
For this optional (ungraded) exercise, you should compute the test error using the best value of λ you found. In our cross validation, we obtained a test error of 3.8599 for $\lambda = 3$. You do not need to submit any solutions for this optional (ungraded) exercise.

```
%%%%%%%%% Add your code to compute the test error below %%%%%%%%%%
m = size(X, 1);
error_test = zeros(m, 1);

for i = 1:m
    theta = trainLinearReg(X(1:i, :),y(1:i),3);
    error_test(i) = linearRegCostFunction(Xtest,ytest,theta,0);
end
```

```
Iteration     1 | Cost: 2.663903e-01
Iteration     2 | Cost: 9.860761e-32
Iteration     3 | Cost: 0.000000e+00
Iteration     1 | Cost: 4.290118e-01
Iteration     3 | Cost: 4.290118e-01
Iteration     1 | Cost: 1.023725e+02
Iteration     2 | Cost: 1.023725e+02
Iteration     3 | Cost: 1.023725e+02
Iteration     4 | Cost: 1.023725e+02
Iteration     1 | Cost: 1.439595e+02
Iteration     2 | Cost: 1.040554e+02
Iteration     1 | Cost: 1.594787e+02
Iteration     2 | Cost: 1.594787e+02
Iteration     3 | Cost: 1.594787e+02
Iteration     4 | Cost: 1.594787e+02
Iteration     5 | Cost: 1.594787e+02
Iteration     1 | Cost: 1.531667e+02
```

```
Iteration     2 | Cost: 1.355282e+02

Iteration     1 | Cost: 1.384516e+02

Iteration     2 | Cost: 1.214818e+02

Iteration     1 | Cost: 1.238826e+02

Iteration     2 | Cost: 1.216797e+02

Iteration     3 | Cost: 1.211012e+02

Iteration     1 | Cost: 1.091241e+02

Iteration     2 | Cost: 1.082722e+02

Iteration     3 | Cost: 1.082428e+02

Iteration     4 | Cost: 1.082428e+02

Iteration     1 | Cost: 1.109986e+02

Iteration     1 | Cost: 1.024794e+02

Iteration     2 | Cost: 1.024794e+02

Iteration     3 | Cost: 1.024794e+02

Iteration     4 | Cost: 1.024794e+02

Iteration     1 | Cost: 1.054113e+02
```

```
plot(1:m, error_test);

title(sprintf('Polynomial Regression Learning Curve (lambda = %f)',
lambda));

xlabel('Number of training examples')

ylabel('Error')

axis([0 13 0 100])

legend('Test')
```



3.5 Optional (ungraded) exercise: Plotting learning curves with randomly selected examples

In practice, especially for small training sets, when you plot learning curves to debug your algorithms, it is often helpful to average across multiple sets of randomly selected examples to determine the training error and cross validation error. Concretely, to determine the training error and cross validation error for i examples, you should first randomly select i examples from the training set and i examples from the cross validation set. You will then learn the parameters θ using the randomly chosen training set and evaluate the parameters θ on the randomly chosen training set and cross validation set. The above steps should then be repeated multiple times (say 50) and the averaged error should be used to determine the training error and cross validation error for i examples.

For this optional (ungraded) exercise, you should implement the above strategy for computing the learning curves in `learningCurve.m` and use the code below to call your modified function and generate the plot.

```
lambda = 0.01;

[error_train, error_val] = learningCurve(X_poly, y, X_poly_val, yval,
lambda);
```

```
plot(1:m, error_train, 1:m, error_val);

title(sprintf('Polynomial Regression Learning Curve (lambda = %f)', lambda));
xlabel('Number of training examples')
ylabel('Error')
axis([0 13 0 100])
legend('Train', 'Cross Validation')
```

For reference, Figure 10 shows the learning curve we obtained for polynomial regression with $\lambda = 0.01$.

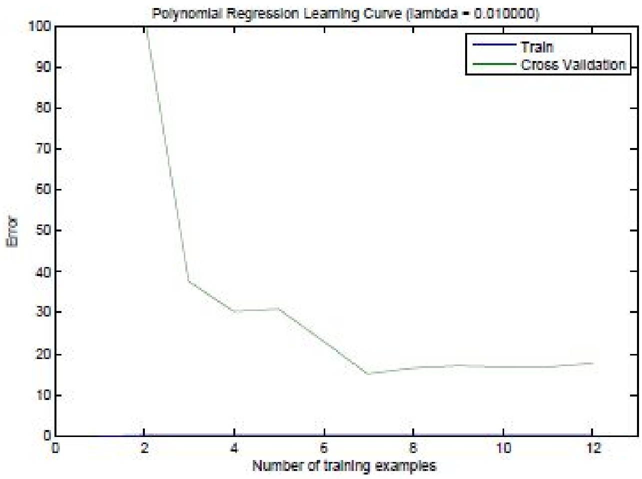


Figure 10: Optional (ungraded) exercise: Learning curve with randomly selected examples

Your figure may differ slightly due to the random selection of examples.