ex5

第一部分 正则化线性回归

上半部分实现线性回归，利用水库水位的变化预测大坝的出水量；下半部分调试学习算法，测试偏差和方差所产生的影响

1.1 数据集可视化

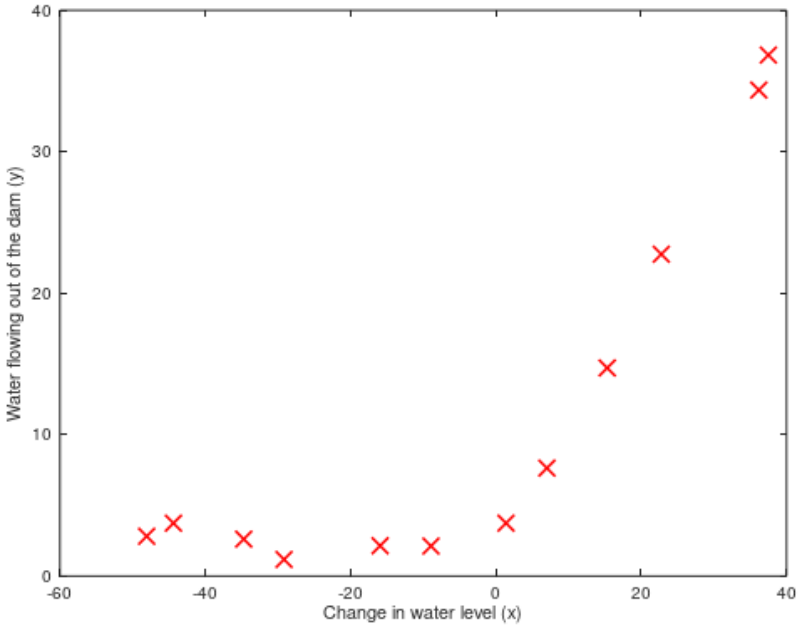
数据集包含水位变化和大坝出水量的历史数据，并被分割成三个部分：

（1）用来训练模型的训练集X、y

（2）用来选择正则化参数l的交叉验证集Xval、yval

（3）用来测试性能的测试集Xtest、ytest

运行ex5:



我们下一步想要实现一个线性回归用来生成一条能够拟合这些数据的直线

1.2 正则化线性回归的损失函数

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

λ是一个控制正则化程度的正则化参数，用来阻止过拟合，正则项对损失值整体进行了惩罚，随着模型参数量级的提高，惩罚也会逐步进行提升，注意不对θ0进行正则化，完成linearRegCostFunction.m文件，我们需要写一个计算正则化线性回归的损失的函数，运行ex5.m获得期望的损失函数值。

代码：

H=X*theta;

J=sum((H-y).^2)/(2*m)+lambda*theta(2:end).^2/(2*m)

运行结果：

```
J =  303.99
Cost at theta = [1 ; 1]: 303.993192
(this value should be about 303.993192)
```

第一次提交：

```
                                  Part Name |     Score | Feedback
                                  --------- |     ----- | --------
Regularized Linear Regression Cost Function |  25 /  25 | Nice work!
     Regularized Linear Regression Gradient |   0 /  25 |
                             Learning Curve |   0 /  20 |
                 Polynomial Feature Mapping |   0 /  10 |
                          Validation Curve |   0 /  20 |
                                  --------------------------
                                           |  25 / 100 |
```

1.3 正则化线性回归梯度

正则化线性回归损失函数的偏导数：

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

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

在linearRegCostFunction.m文件中增加代码计算梯度，返回变量grad，运行ex5.m文件获得期望的梯度值。

新增代码：

grad=X'*(H-y)/m+lambda*[0;theta(2:end)]/m

运行结果：

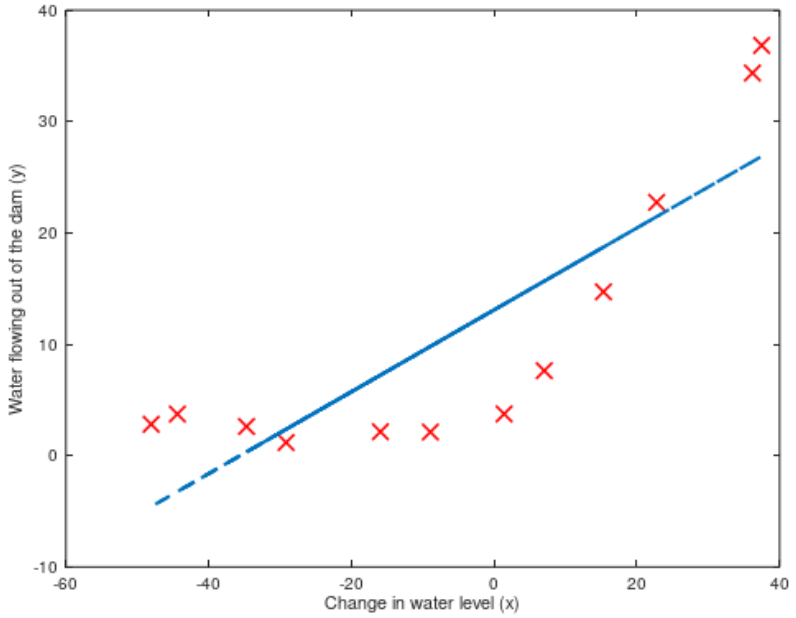
```
J =  303.99
grad =

   -15.303
   598.251

Gradient at theta = [1 ; 1]:  [-15.303016; 598.250744]
(this value should be about [-15.303016; 598.250744])
```

第二次提交：

```
                                  Part Name |     Score | Feedback
                                  --------- |     ----- | --------
Regularized Linear Regression Cost Function |  25 /  25 | Nice work!
     Regularized Linear Regression Gradient |  25 /  25 | Nice work!
                             Learning Curve |   0 /  20 |
                 Polynomial Feature Mapping |   0 /  10 |
                          Validation Curve |   0 /  20 |
                                  ---------------------------
                                           |  50 / 100 |
```

1.4 拟合线性回归

此时ex5.m将会运行trainLinearReg.m文件来计算theta的最优值，其使用了fmincg来求取最小的损失函数值。我们设置这则话参数lambda为0，由于现在实现的线性回归是通过拟合2维theta向量的，正则化对于求解如此低维度的theta毫无帮助，后面的多项式回归问题将会使用到正则化。ex5.m也会将最优的拟合直线显示出来，下一部分实现能够生成学习曲线的函数，以便对学习算法进行优化。



第二部分 偏差与方差

偏差与方差的平衡是机器学习中重要的概念，高偏差的模型对数据而言不够复杂，有欠拟合的趋势，而高方差的模型对训练数据来说是过拟合的。本练习中我们需要在一条学习曲线上显示训练和测试误差以对偏差和方差的问题进行诊断。、

2.1 学习曲线

用代码实现学习曲线将会对修正学习算法很有帮助，补充learningCurve.m文件，返回训练集和交叉验证集的误差向量。为了显示这条学习曲线，我们需要得

到针对不同大小训练集的训练集误差和交叉验证集误差，使用初始训练集X的不同子集来获取不同训练集的大小：X(1:i,:)and y(1:i)，依旧通过trainLinearReg来求得theta参数，lambda作为参数传递给learnigCurve函数，训练得到theta参数之后，计算训练集和交叉验证集的误差：

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

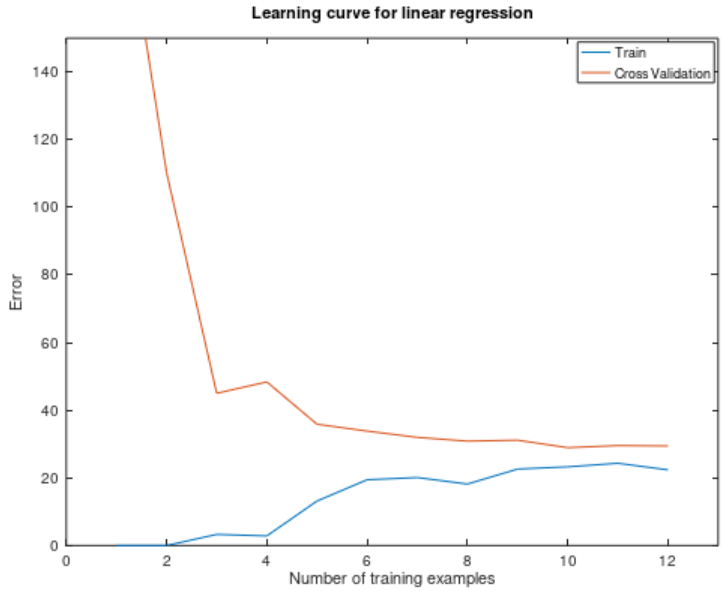
注意训练集误差不应该包含正则项，一种计算误差的方法是使用现有的损失函数，设置lambda为0，求训练集误差时要保证使用了训练子集，而求交叉验证集误差时使用整个交叉验证集，将两种误差存放到err_train和err_val两个变量中。

代码：

```
for i=1:m
  mX=X(1:i,:)
  my=y(1:i)
  [theta]=trainLinearReg(mX,my,lambda)
  error_train(i)=linearRegCostFunction(mX,my,theta,0)
  error_val(i)=linearRegCostFunction(Xval,yval,theta,0)
endfor
```

运行结果：

Training Examples	Train Error	Cross Validation Error
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



第三次提交：

```
                                  Part Name |     Score | Feedback
                                  --------- |     ----- | --------
Regularized Linear Regression Cost Function |  25 /  25 | Nice work!
     Regularized Linear Regression Gradient |  25 /  25 | Nice work!
                             Learning Curve |  20 /  20 | Nice work!
                 Polynomial Feature Mapping |   0 /  10 |
                           Validation Curve |   0 /  20 |
                                  --------------------------------
                                           |  70 / 100 |
```

第三部分 多项式回归

上述线性模型对数据而言过于简单，导致欠拟合问题，这部分练习需要通过加入更多的特征（多项式特征）来解决高偏差的问题。多项式回归假设的形式：

$$h_\theta(x) = \theta_0 + \theta_1 * (\text{waterLevel}) + \theta_2 * (\text{waterLevel})^2 + \cdots + \theta_p * (\text{waterLevel})^p$$
$$= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_p x_p.$$

完成PolyFeatures.m文件，生成能将大小为m*1的原始训练集X映射成更高次项得到函数，当将X传入该方法中时需要返回一个m*p维度的矩阵X_poly,第一列为原始数据X，第二列为X.^2，第三列为X.^3，等等，无需计算0次方。运行ex5.m文件将多项式回归应用于训练集、交叉验证集以及测试集。

代码：

```
for i = 1 : p,
    X_poly(:,i) = X.^i;
```

运行结果：

```
X_poly =

  -1.6747e+01   2.8045e+02  -4.6965e+03   7.8650e+04  -1.3171e+06   2.2057e+07  -3.6938e+08   6.1859e+09
  -1.4577e+01   2.1250e+02  -3.0978e+03   4.5157e+04  -6.5828e+05   9.5961e+06  -1.3989e+08   2.0392e+09
   3.4516e+01   1.1913e+03   4.1120e+04   1.4193e+06   4.8988e+07   1.6908e+09   5.8361e+10   2.0144e+12
  -4.7010e+01   2.2099e+03  -1.0389e+05   4.8839e+06  -2.2959e+08   1.0793e+10  -5.0738e+11   2.3852e+13
   3.6975e+01   1.3672e+03   5.0551e+04   1.8691e+06   6.9111e+07   2.5554e+09   9.4486e+10   3.4936e+12
  -4.0686e+01   1.6554e+03  -6.7350e+04   2.7402e+06  -1.1149e+08   4.5360e+09  -1.8455e+11   7.5088e+12
  -4.4720e+00   1.9999e+01  -8.9435e+01   3.9996e+02  -1.7886e+03   7.9987e+03  -3.5770e+04   1.5996e+05
   2.6534e+01   7.0403e+02   1.8681e+04   4.9566e+05   1.3152e+07   3.4896e+08   9.2593e+09   2.4568e+11
  -4.2798e+01   1.8316e+03  -7.8390e+04   3.3549e+06  -1.4358e+08   6.1450e+09  -2.6299e+11   1.1255e+13
   2.5374e+01   6.4384e+02   1.6337e+04   4.1454e+05   1.0518e+07   2.6690e+08   6.7723e+09   1.7184e+11
  -3.1110e+01   9.6780e+02  -3.0108e+04   9.3665e+05  -2.9139e+07   9.0649e+08  -2.8200e+10   8.7730e+11
   2.7312e+01   7.4593e+02   2.0373e+04   5.5642e+05   1.5197e+07   4.1505e+08   1.1336e+10   3.0960e+11
  -3.2639e+00   1.0653e+01  -3.4769e+01   1.1348e+02  -3.7039e+02   1.2089e+03  -3.9457e+03   1.2878e+04
  -1.8183e+00   3.3061e+00  -6.0115e+00   1.0930e+01  -1.9875e+01   3.6138e+01  -6.5708e+01   1.1948e+02
  -4.0720e+01   1.6581e+03  -6.7517e+04   2.7493e+06  -1.1195e+08   4.5585e+09  -1.8562e+11   7.5585e+12
  -5.0013e+01   2.5013e+03  -1.2510e+05   6.2566e+06  -3.1291e+08   1.5650e+10  -7.8270e+11   3.9145e+13
  -1.7412e+01   3.0317e+02  -5.2787e+03   9.1912e+04  -1.6003e+06   2.7865e+07  -4.8518e+08   8.4478e+09
   3.5882e+00   1.2875e+01   4.6198e+01   1.6577e+02   5.9481e+02   2.1343e+03   7.6583e+03   2.7479e+04
   7.0855e+00   5.0204e+01   3.5572e+02   2.5204e+03   1.7859e+04   1.2654e+05   8.9657e+05   6.3526e+06
   4.6282e+01   2.1421e+03   9.9140e+04   4.5884e+06   2.1236e+08   9.8286e+09   4.5489e+11   2.1054e+13
   1.4612e+01   2.1352e+02   3.1200e+03   4.5590e+04   6.6618e+05   9.7344e+06   1.4224e+08   2.0785e+09
```

第四次提交：

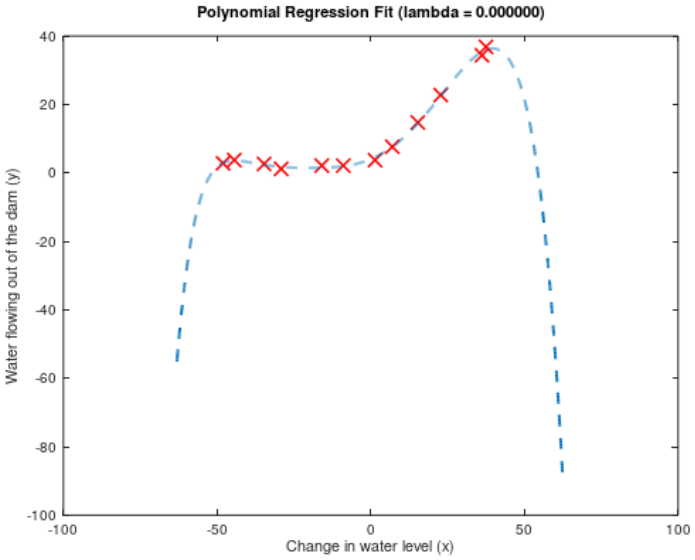
```
                                  Part Name |     Score | Feedback
                                  --------- |     ----- | --------
Regularized Linear Regression Cost Function |  25 /  25 | Nice work!
     Regularized Linear Regression Gradient |  25 /  25 | Nice work!
                             Learning Curve |  20 /  20 | Nice work!
                 Polynomial Feature Mapping |  10 /  10 | Nice work!
                           Validation Curve |   0 /  20 |
                                  ----------------------------------
                                           |  80 / 100 |
```

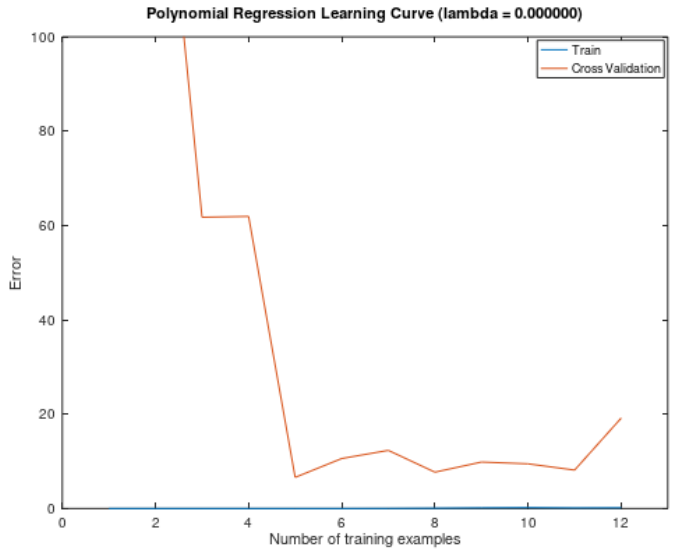
3.1 学习多项式回归

尽管我们在特征向量中有多项式项，我们依然在解决线性回归的优化问题，我们使用前面已经完成的损失函数和梯度。该部分练习中，多项式次数最高为8，如果直接在数据上进行训练将会得到较差的结果，因为特征值均被较差地放缩了，因此我们需要进行特征值标准化处理。在学习多项式回归的theta参数之前，ex5.m将会调用featureNormalize函数对训练集中的特征进行标准化处理，将mu、sigma参数分开放置。

```
Normalized Training Example 1:
  1.000000
  -0.362141
  -0.755087
  0.182226
  -0.706190
  0.306618
  -0.590878
  0.344516
  -0.508481
```

学习完参数theta之后你将看到两张多项式回归图（lambda=0，即未进行抗过拟合处理），



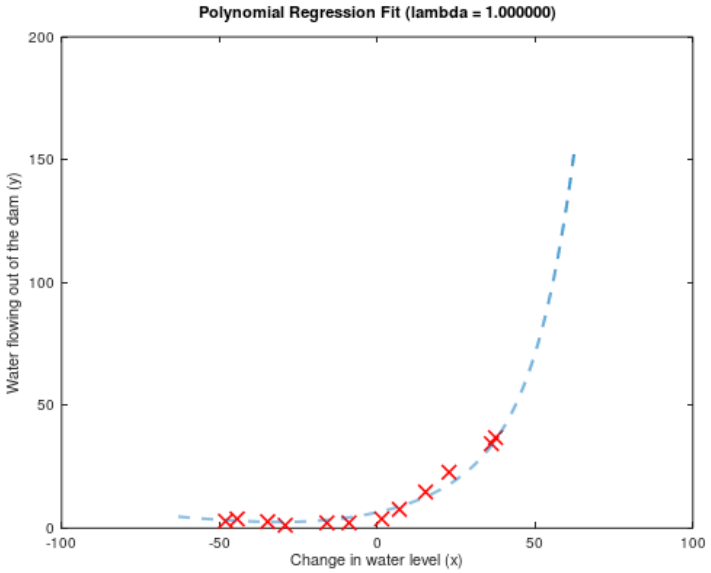


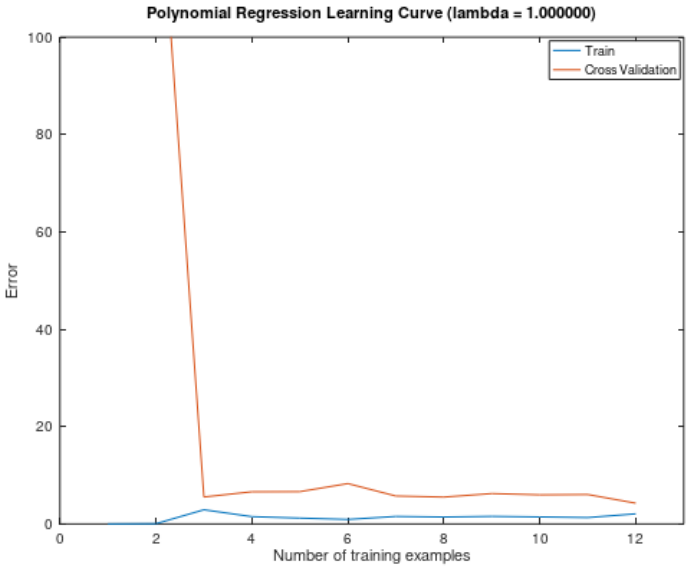
对抗过拟合的方法之一就是加入正则项，下一部分尝试不同的正则化参数λ生成更好的模型。

3.2 （可选部分）调整正则化参数

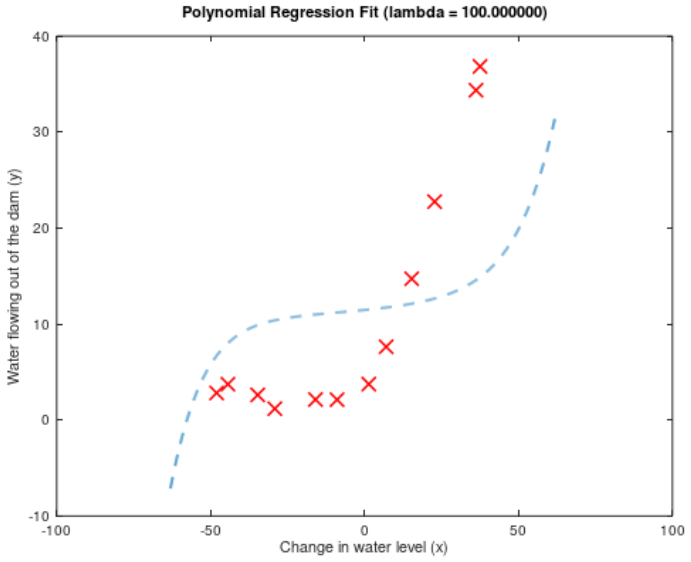
观察正则化参数是如何影响正则化多项式回归的，调整ex5.m中的正则化参数，ex5.m同样会产生拟合图和学习曲线图。

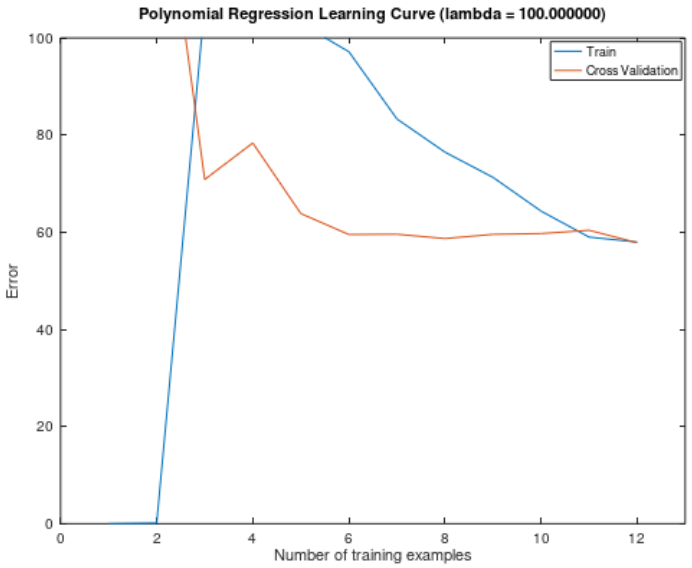
λ=1：在偏差与方差之间取得了较好的平衡





λ=100:正则化过多以至于无法正常拟合数据，属于欠拟合





3.3 通过交叉验证集来选择正则化参数

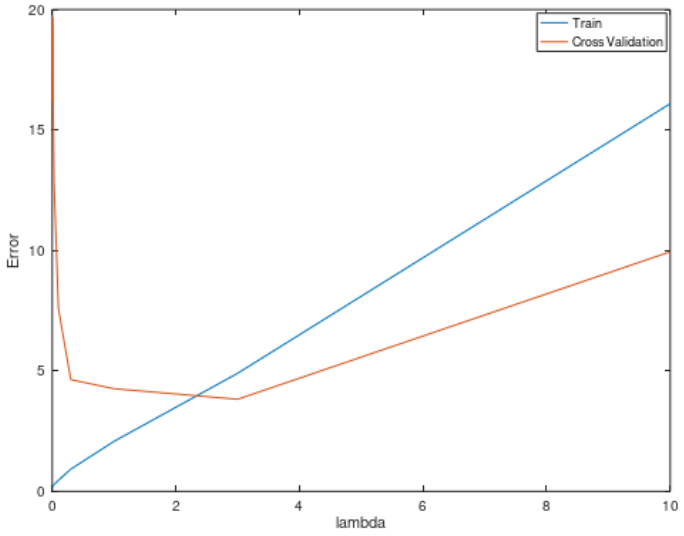
解决偏差与方差的平衡把控问题，一个未进行正则化的问题虽然能很好地拟合数据，却不具备好的泛化能力，一个正则化过度的模型不能很好地拟合训练集和测试集，因此需要一个好的lambda。该练习中你需要实现一个能够自动化选择正则化参数的函数。具体而言，需要通过交叉验证集来测试某一个正则化参数到底有多棒，然后我们就可以在从未见过的数据上测试我们的新模型。完成validationCurve.m文件，特别需要使用trainLinearReg函数来训练使用了不同正则化参数的模型，计算训练集误差和交叉验证集误差，在以下范围中尝试正则化参数：

{0,0.001,0.003,0.01,0.03,0.1,0.3,1,3,10}

代码：

```
for i=1:length(lambda_vec)
  lambda = lambda_vec(i)
  [theta]=trainLinearReg(X,y,lambda)
  error_train(i)=linearRegCostFunction(X,y,theta,0)
  error_val(i)=linearRegCostFunction(Xval,yval,theta,0)
endfor
```

测试结果：



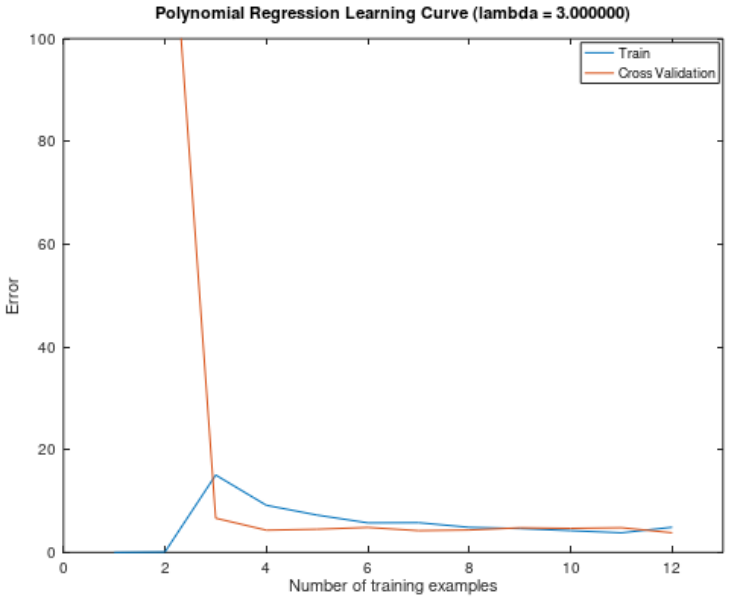
观察发现正则化参数的最优值在3附近

第五次提交：

```
                                  Part Name |     Score | Feedback
                                  --------- |     ----- | --------
Regularized Linear Regression Cost Function |  25 /  25 | Nice work!
    Regularized Linear Regression Gradient |  25 /  25 | Nice work!
                            Learning Curve |  20 /  20 | Nice work!
                Polynomial Feature Mapping |  10 /  10 | Nice work!
                        Validation Curve |  20 /  20 | Nice work!
                                  --------------------------------
                                            | 100 / 100 |
```

3.4 可选练习：计算测试集误差

为了证明在真实世界中，模型依旧具有良好的性能，需要让模型对从未见过的数据进行预测，我们使用lambda为3时抗过拟合训练的模型进行测试集检验，误差在3.8599左右。



3.5 可选练习：显示随机选择样本的学习曲线

在实践中，尤其是对小型数据集，在生成学习曲线图时最好使用随机选择的训练集样本进行theta参数的训练，使用随机选择的训练集样本和交叉验证集样本（共i个）对theta参数进

行评估，多次重复上述操作进行训练获取平均误差（训练集误差和交叉验证集误差）。下图为lambda=0.01时的学习曲线图：