Assignment12

December 20, 2018

1 Assignment12

2 Name : Junha Lee

3 Student ID : 2017220159

4 https://github.com/myosoo/Assignment12

5 Import packages

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from numpy.linalg import inv
from numpy import linalg as LA
```

6 A set of data $((x_1, y_1), (x_2, y_2), \cdots (x_n, y_n))$ is generated

In [2]:
```
num = 1001
std = 5

# x : x-coordinate data
# y1 : (clean) y-coordinate data
# y2 : (noisy) y-coordinate data

def fun(x):
    f = np.abs(x) * np.sin(x)
    return f

n = np.random.rand(num)
nn = n - np.mean(n)
x = np.linspace(-10, 10, num)
y1 = fun(x)
y2 = y1 + nn * std
```

7 Define vandemonde matrix :

$$\begin{bmatrix} x_0^0 & x_0^1 & x_0^2 & \cdots & x_0^p \\ x_1^0 & x_1^1 & x_1^2 & \cdots & x_1^p \\ x_2^0 & x_2^1 & x_2^2 & \cdots & x_2^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n^0 & x_n^1 & x_n^2 & \cdots & x_n^p \end{bmatrix}$$

```
In [3]: def Vandermonde_Matrix(x, p):
            return np.column_stack(x ** (i) for i in range(p + 1))
```

7.0.1 Find an optimal set of model parameters that provide the least square approximate solution :

$$\varepsilon(\theta,\lambda) = \|\mathbf{A}\cdot\theta - y\|^2 + \lambda\|\theta\|^2 = \left\| \begin{vmatrix} \mathbf{A} \\ \sqrt{\lambda} \end{vmatrix} \theta - \begin{vmatrix} y \\ 0 \end{vmatrix} \right\|^2 \Rightarrow \|Ax+b\|^2 form, \theta = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y_2}$$

```
In [4]: def Training_theta(A, lamb, y2):
            block_matrix = np.concatenate((A, np.sqrt(lamb) * np.identity(A.shape[1])), axis = 0
            b = np.concatenate((np.reshape(y2,(y2.size, 1)), np.zeros((A.shape[1], 1))), axis =
            return np.linalg.pinv(block_matrix).dot(b)
```

8 Define approximation model : $\hat{f}(x_n) = \mathbf{A}\cdot\theta$

```
In [5]: def Approximation_model(A, Theta):
            return np.matmul(A, Theta)
```

9 Define least square error

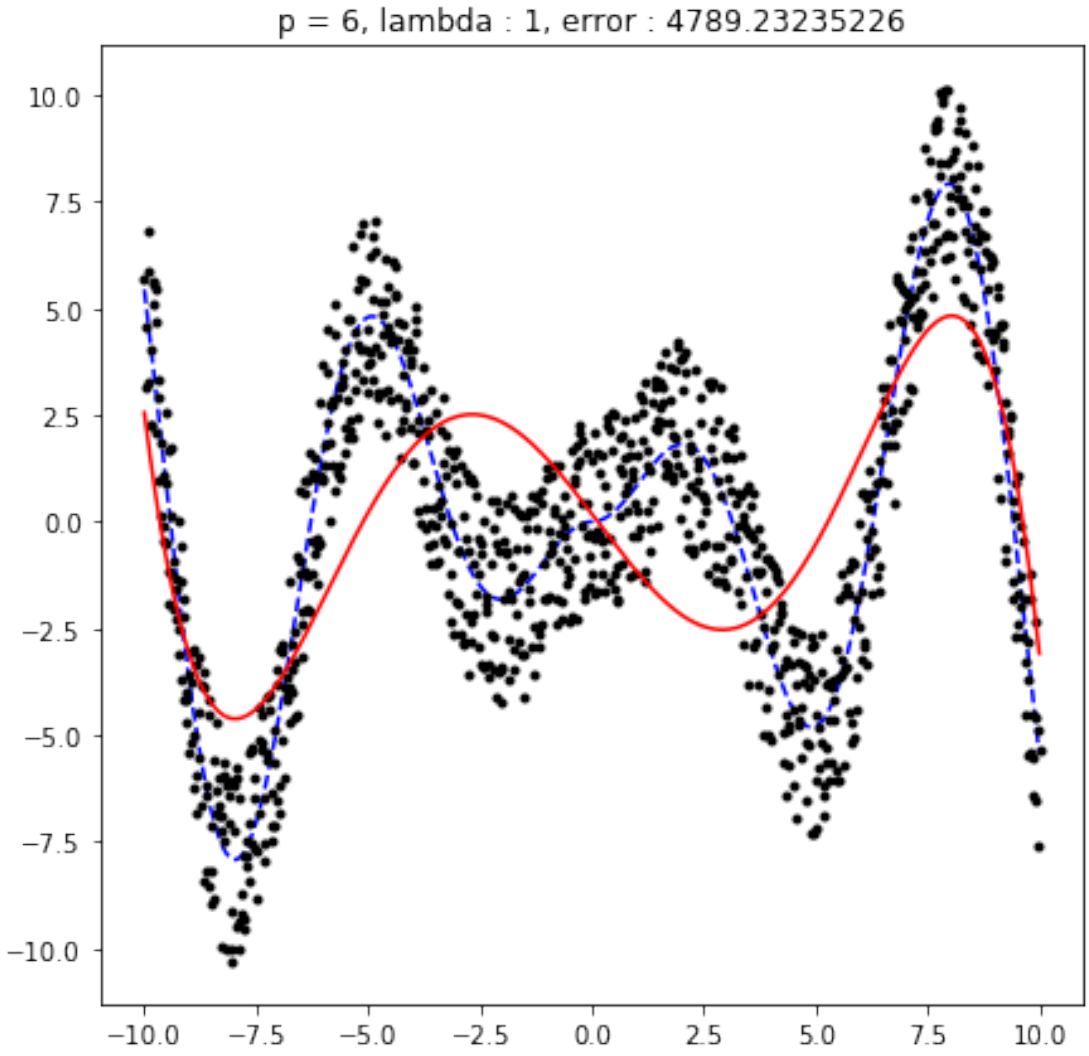
```
In [6]: def Least_square_error(app_model, y2, lamb, Theta):
            return LA.norm(app_model - y2) + LA.norm(lamb * Theta)
```

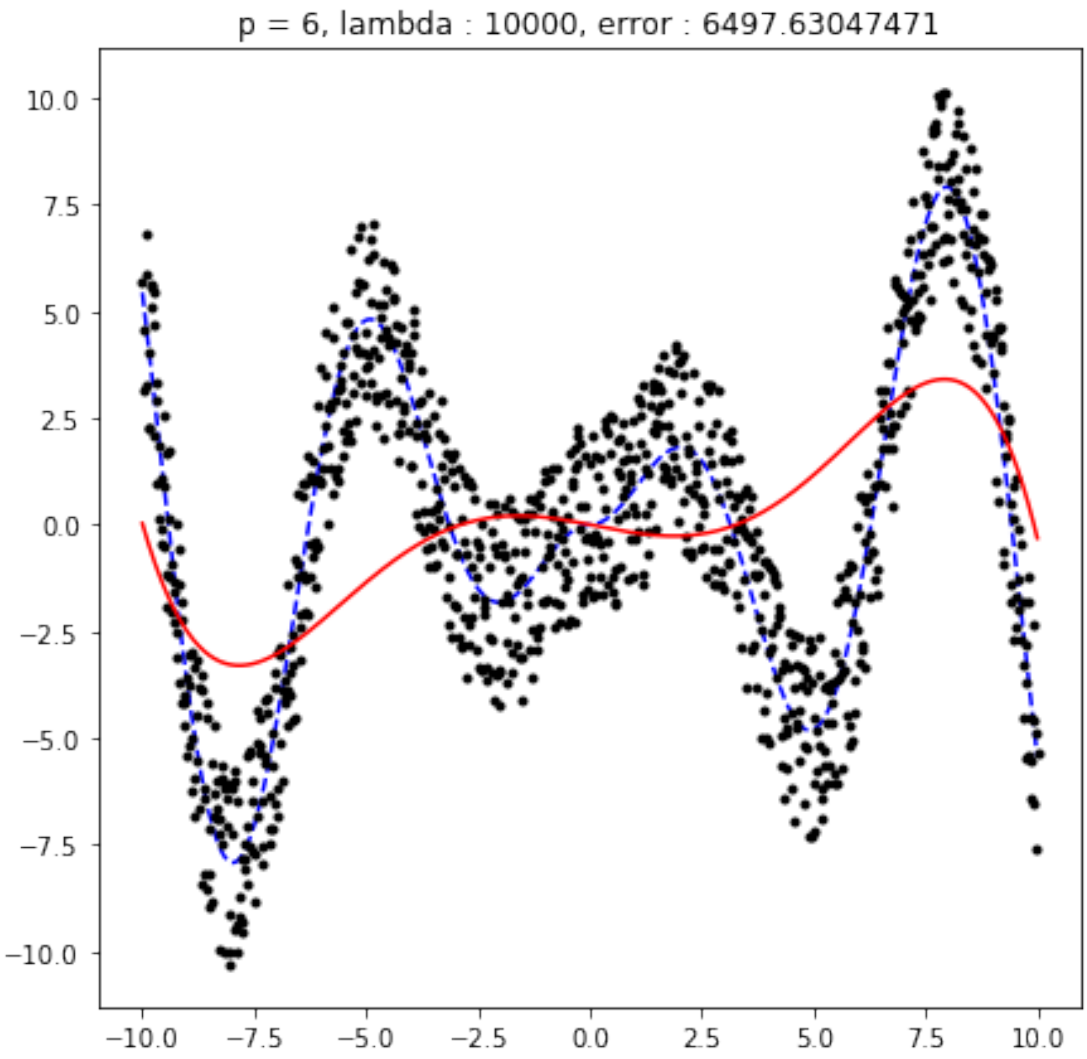
10 Plot the polynomial curves that t the noisy data by the least square error with varying $\lambda = 1, 10^4, 10^6$ with fixed each $p = 6,7,8,9,\cdots,15$

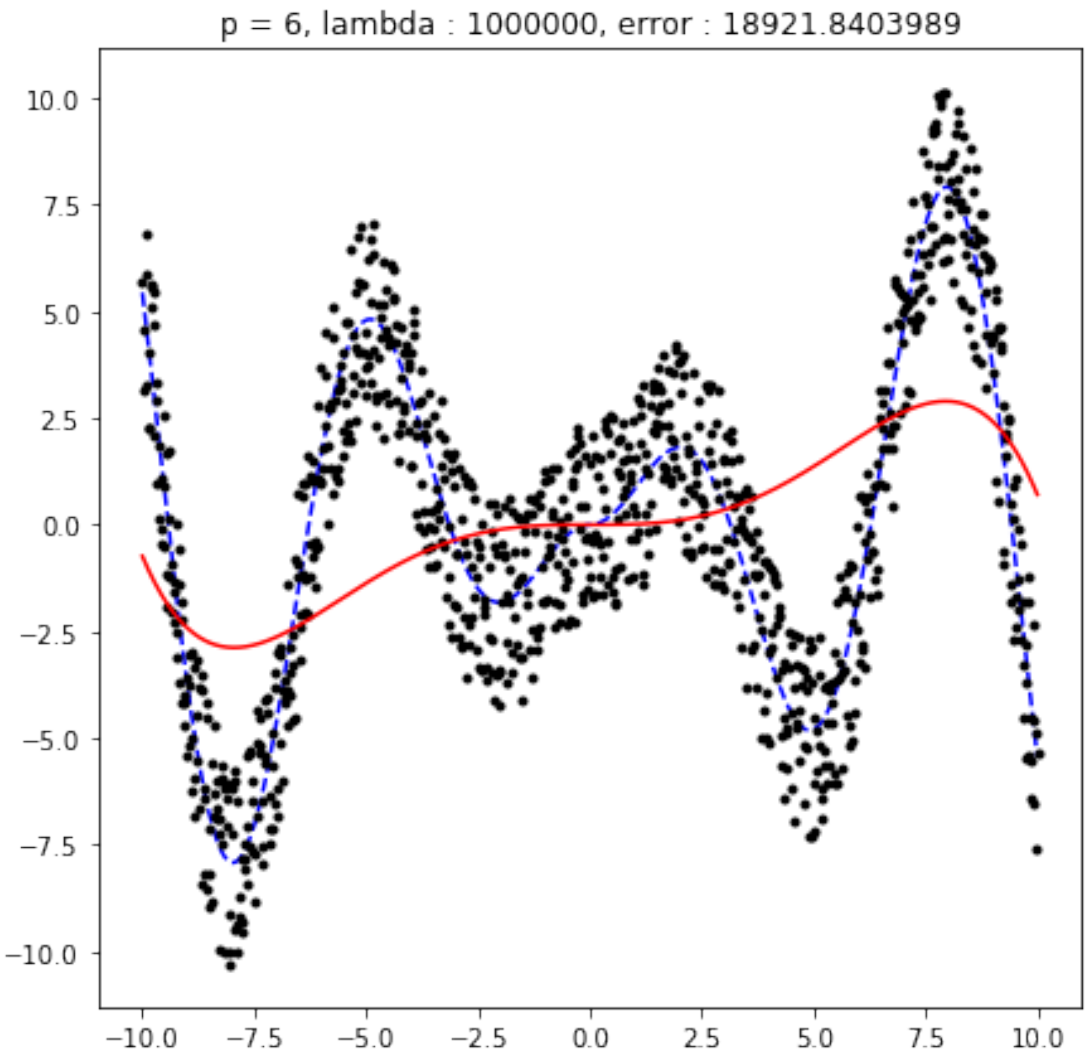
```
In [7]: lamb_list = [1, 10 ** 4, 10 ** 6]
        error_log = []

        for i in range(6, 16):
            for lamb in lamb_list:
                A = Vandermonde_Matrix(x, i)
                Theta = Training_theta(A, lamb, y2)
                app_model = Approximation_model(A, Theta)
                error = Least_square_error(app_model, y2, lamb, Theta)
                error_log.append(error)
```

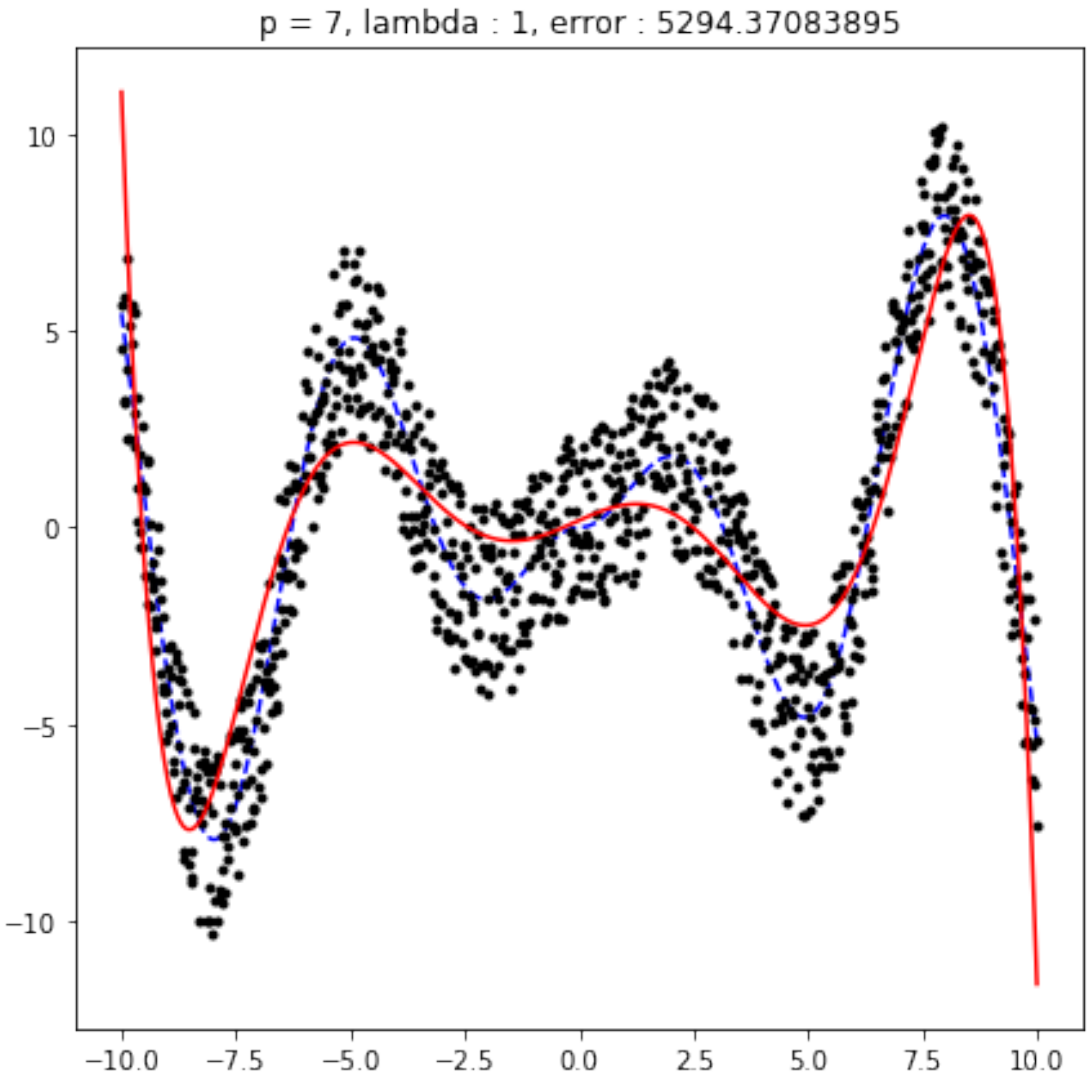
```
        plt.figure(figsize = (7, 7))
        plt.title('p = '+ str(i) + ', lambda : ' + str(lamb) + ', error : ' + str(error)
        plt.plot(x, y1, 'b--', x, y2, 'k.', x, app_model, 'r-')
```

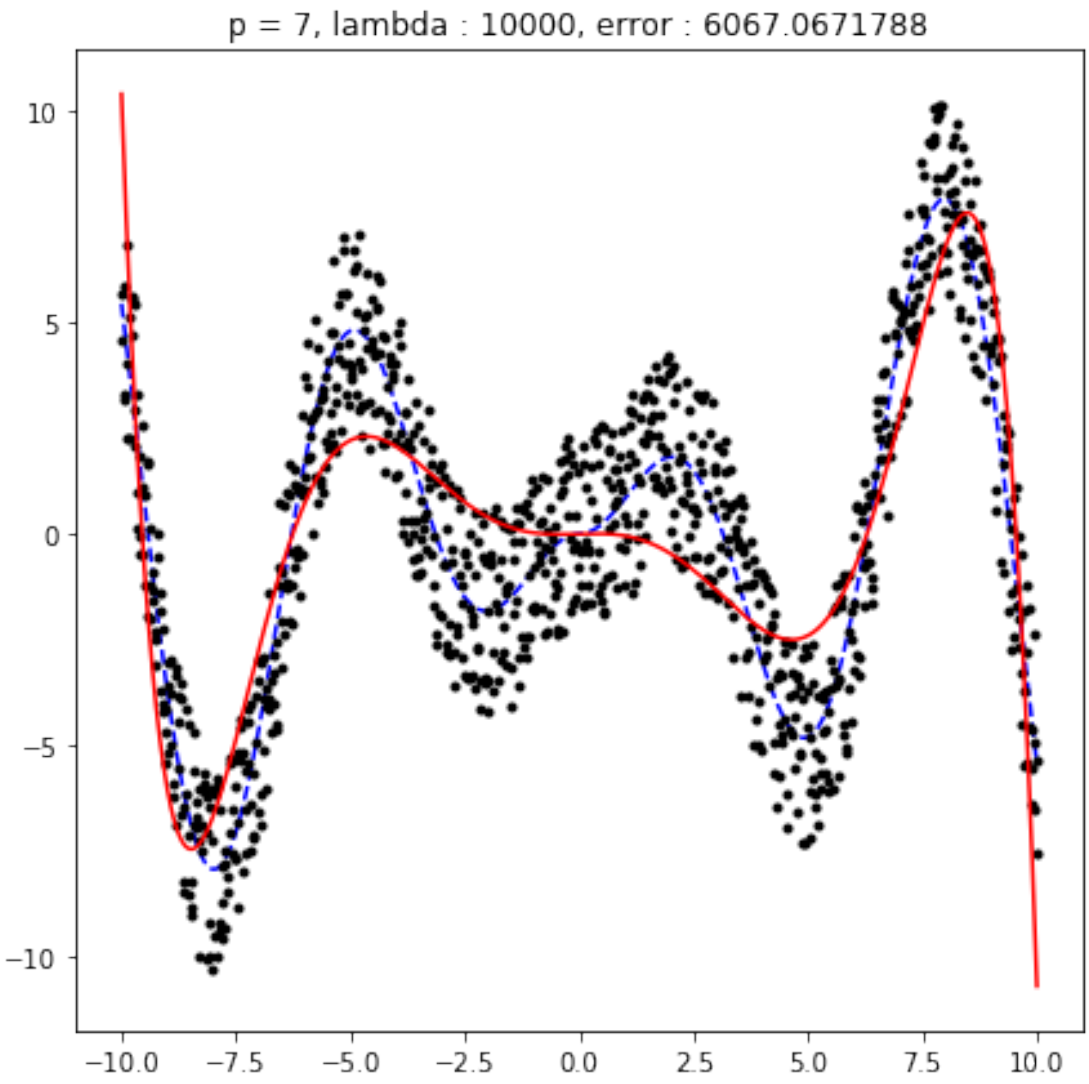
```
/usr/local/lib/python3.5/dist-packages/matplotlib/pyplot.py:524: RuntimeWarning: More than 20 fi
  max_open_warning, RuntimeWarning)
```

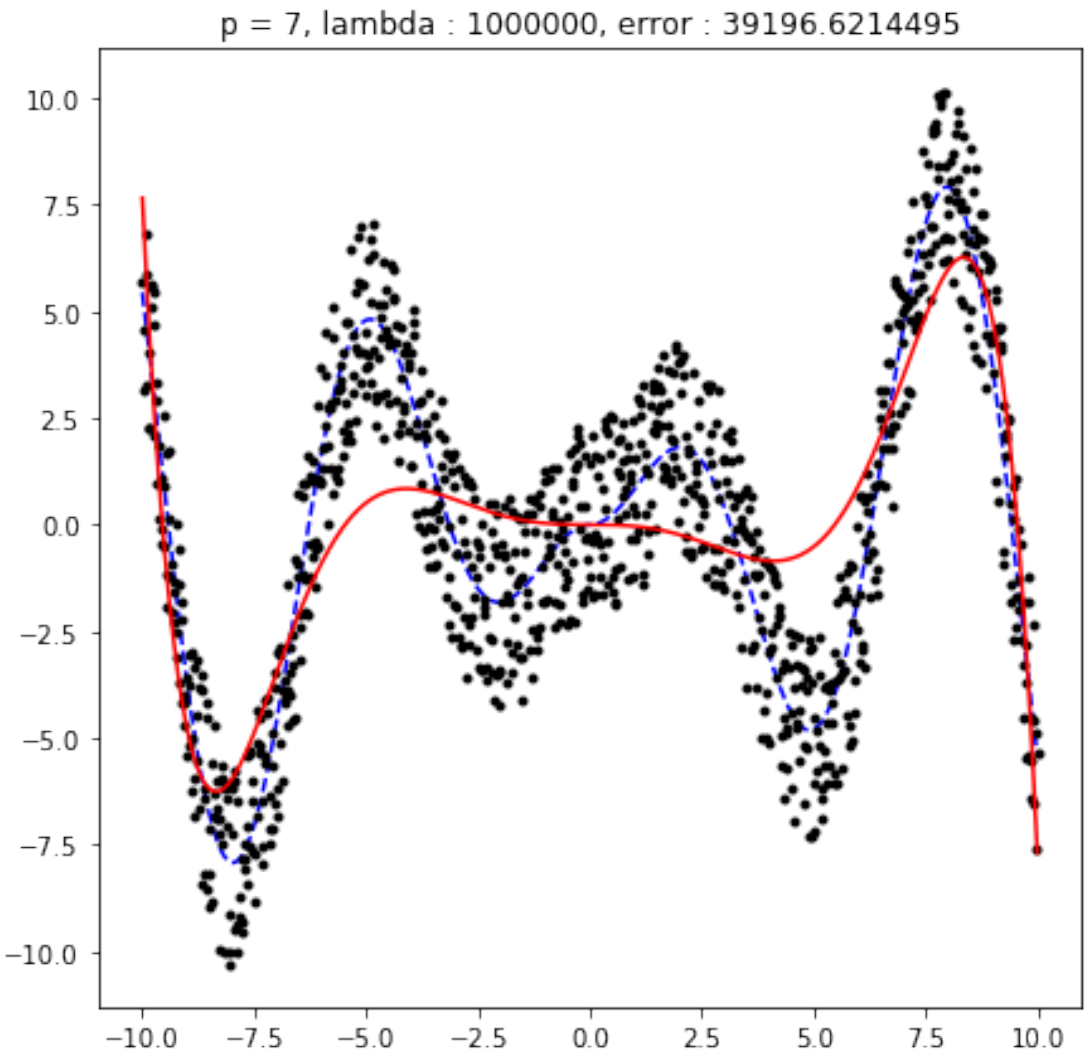


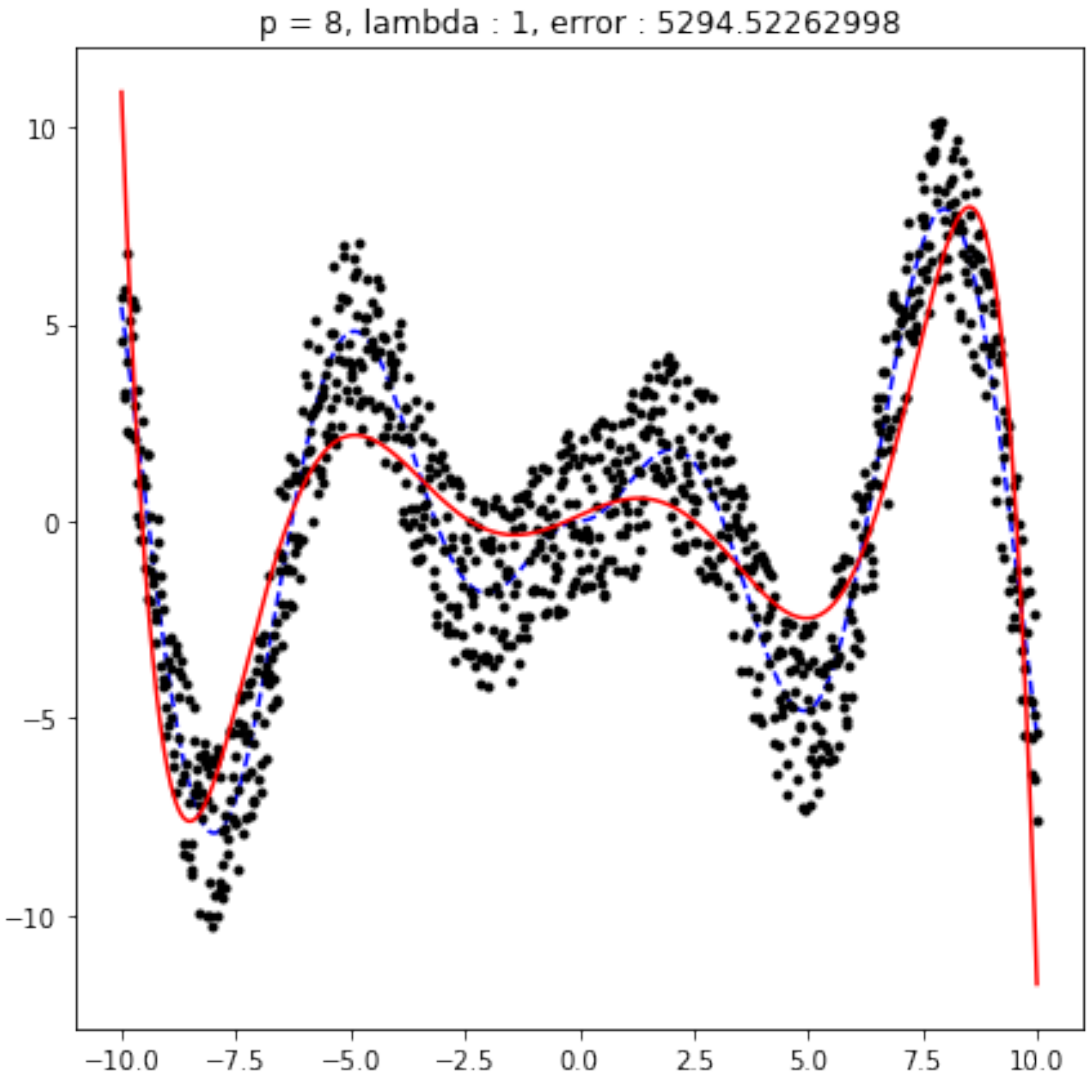

p = 6, lambda : 10000, error : 6497.63047471
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

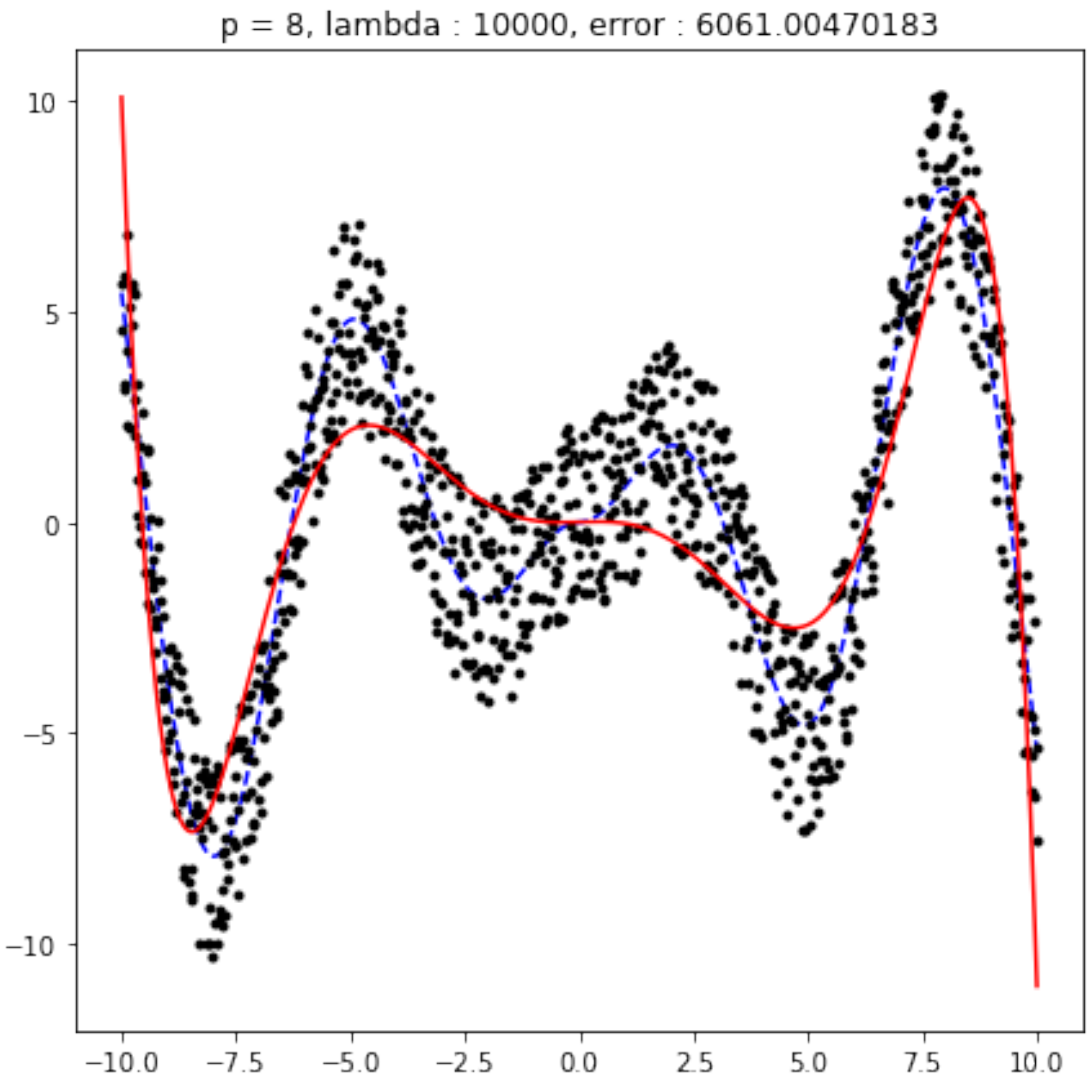

p = 6, lambda : 1000000, error : 18921.8403989
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

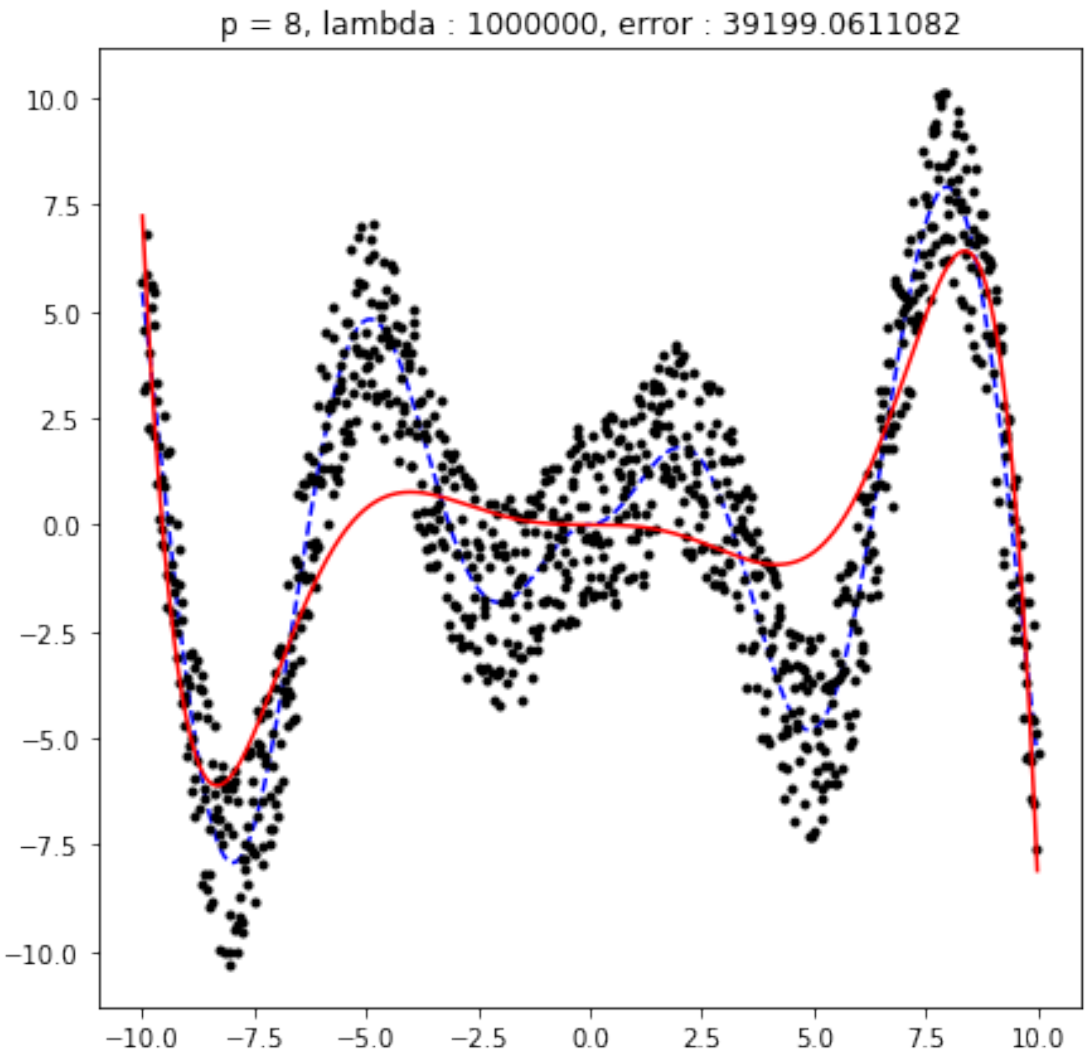

p = 7, lambda : 1, error : 5294.37083895
10
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

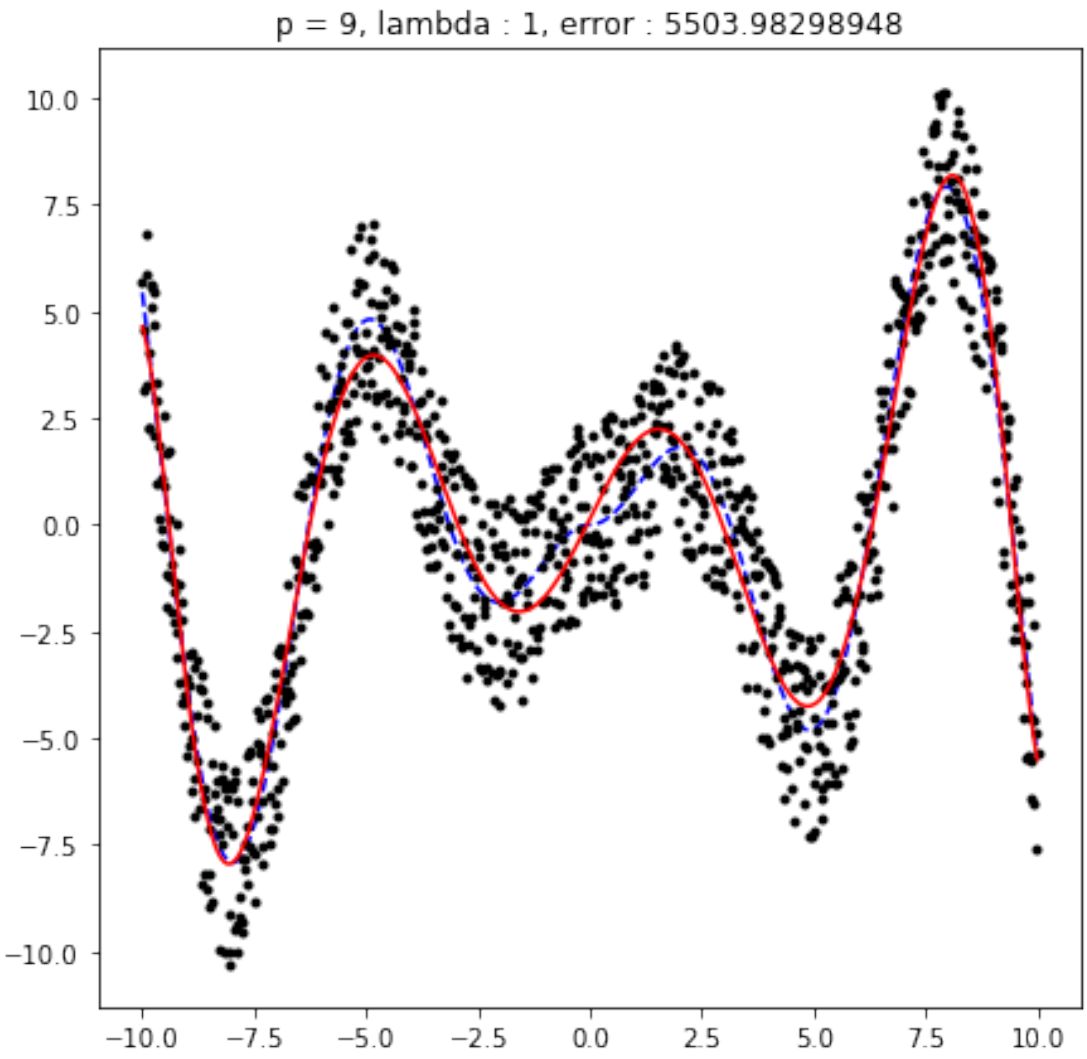

p = 7, lambda : 10000, error : 6067.0671788
10
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

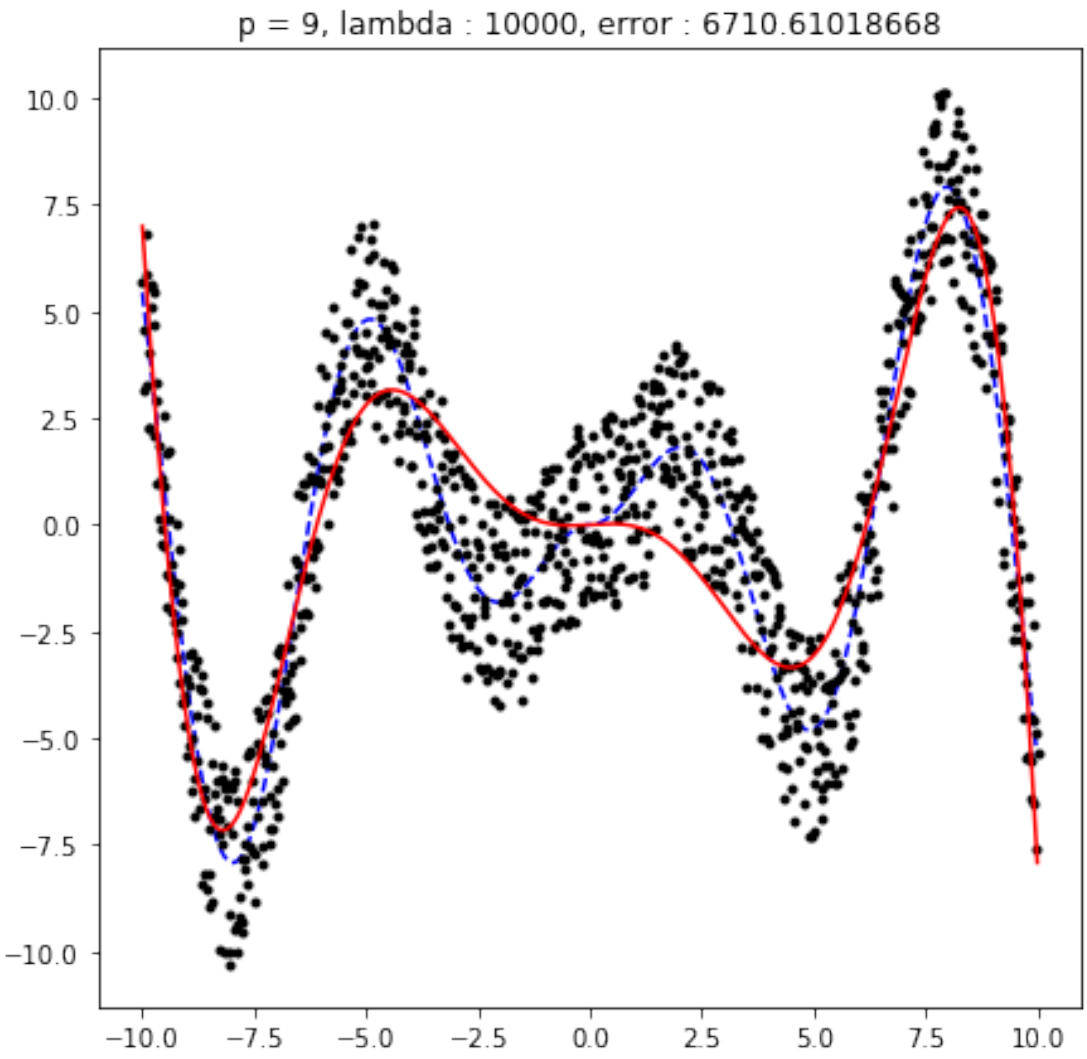

p = 7, lambda : 1000000, error : 39196.6214495
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

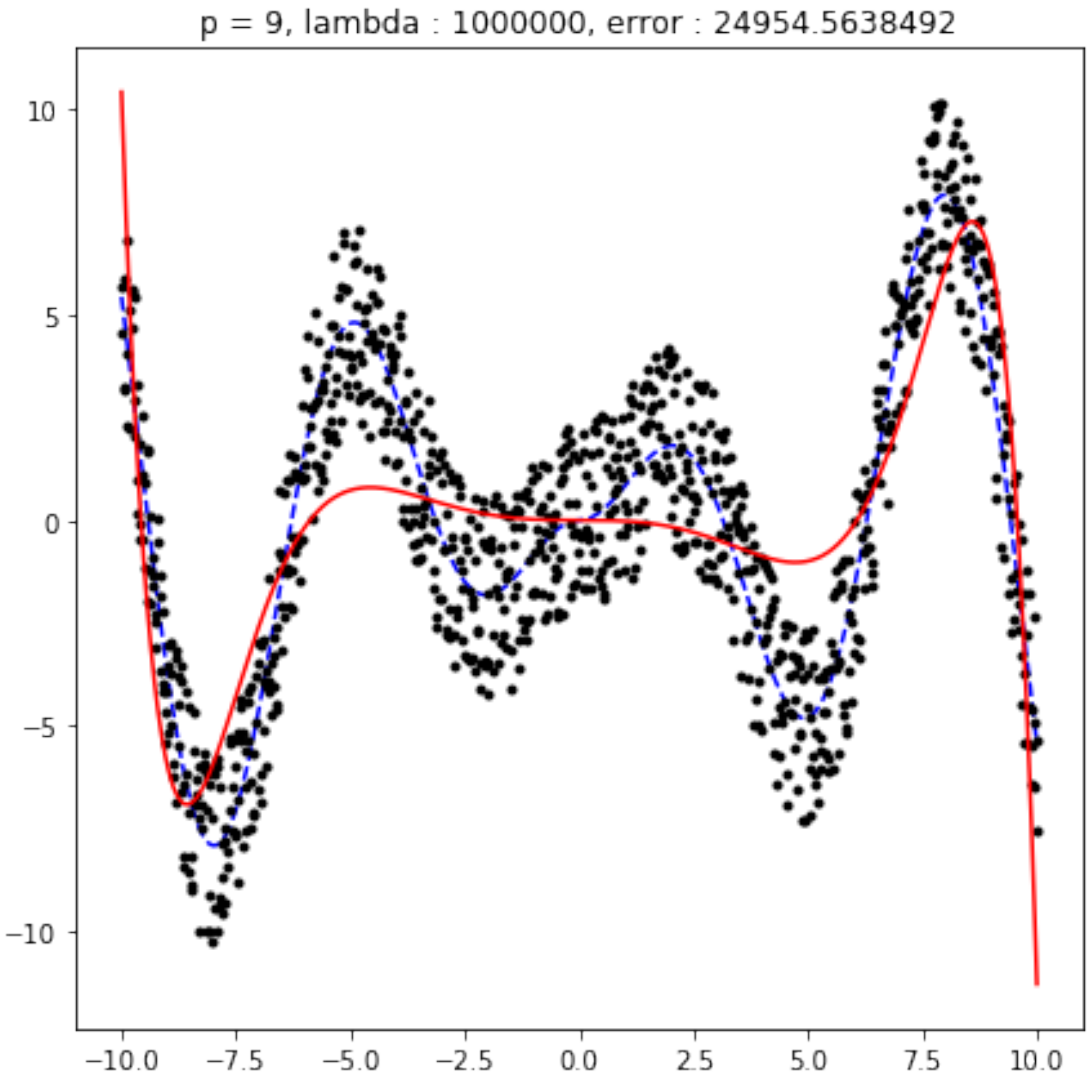

p = 8, lambda : 1, error : 5294.52262998
10
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

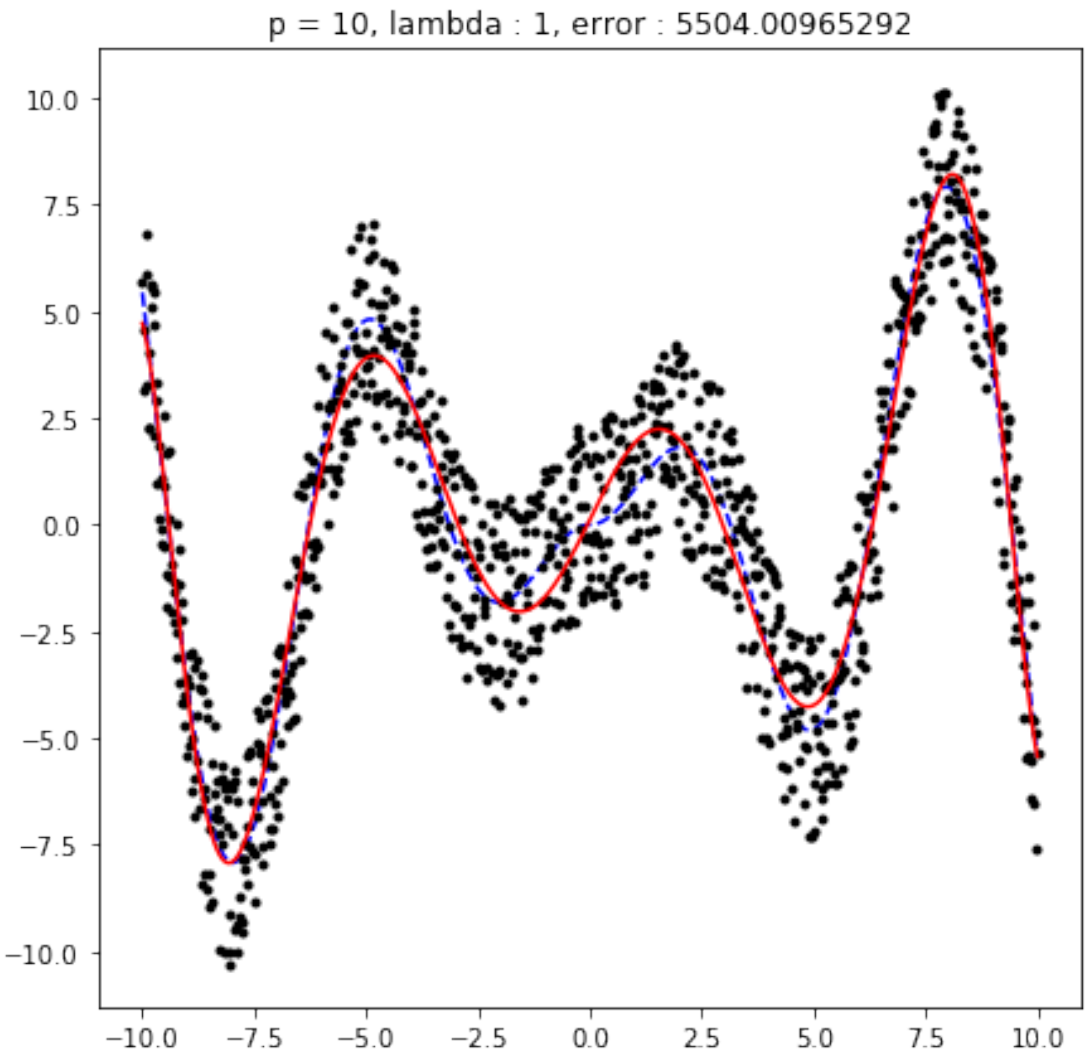

p = 8, lambda : 10000, error : 6061.00470183
10
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

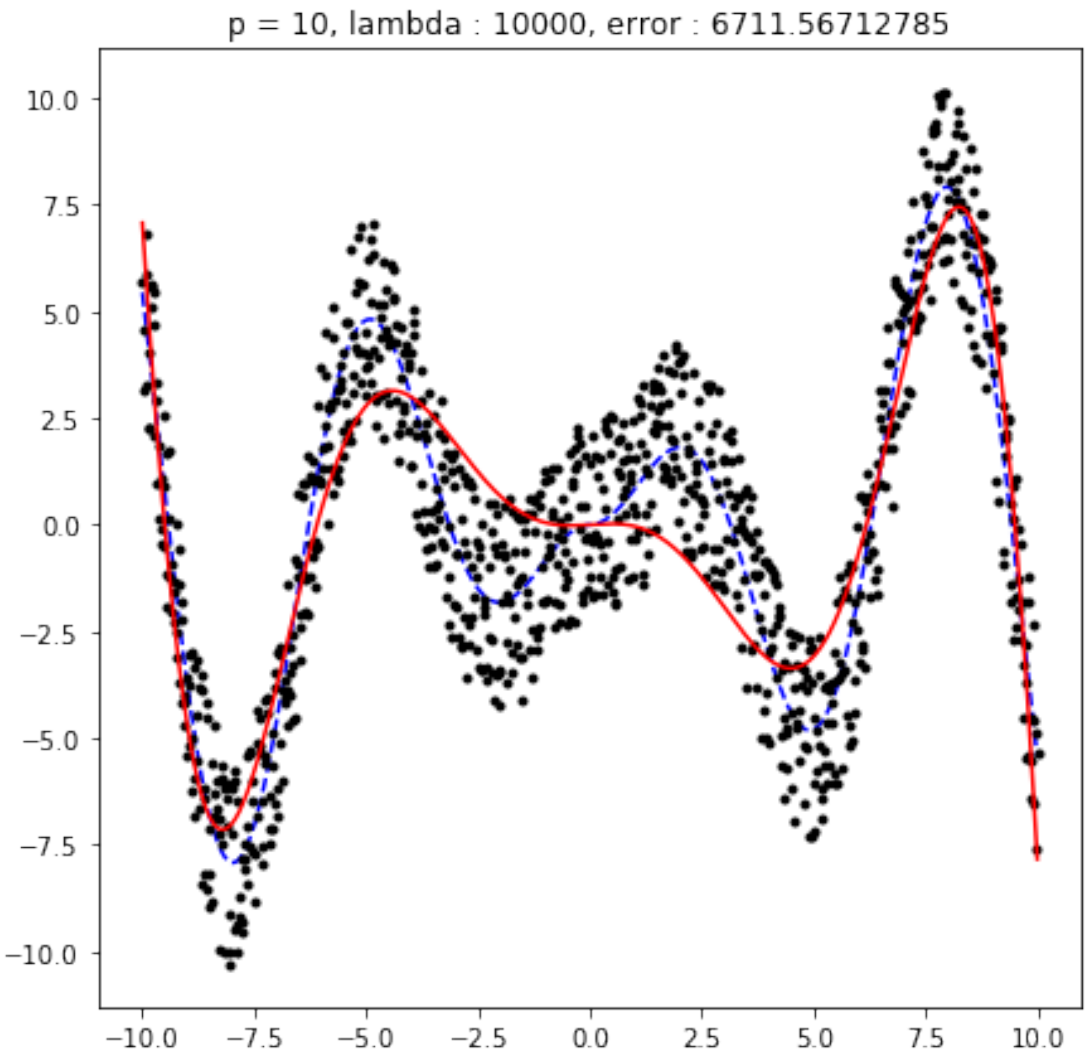

p = 8, lambda : 1000000, error : 39199.0611082
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

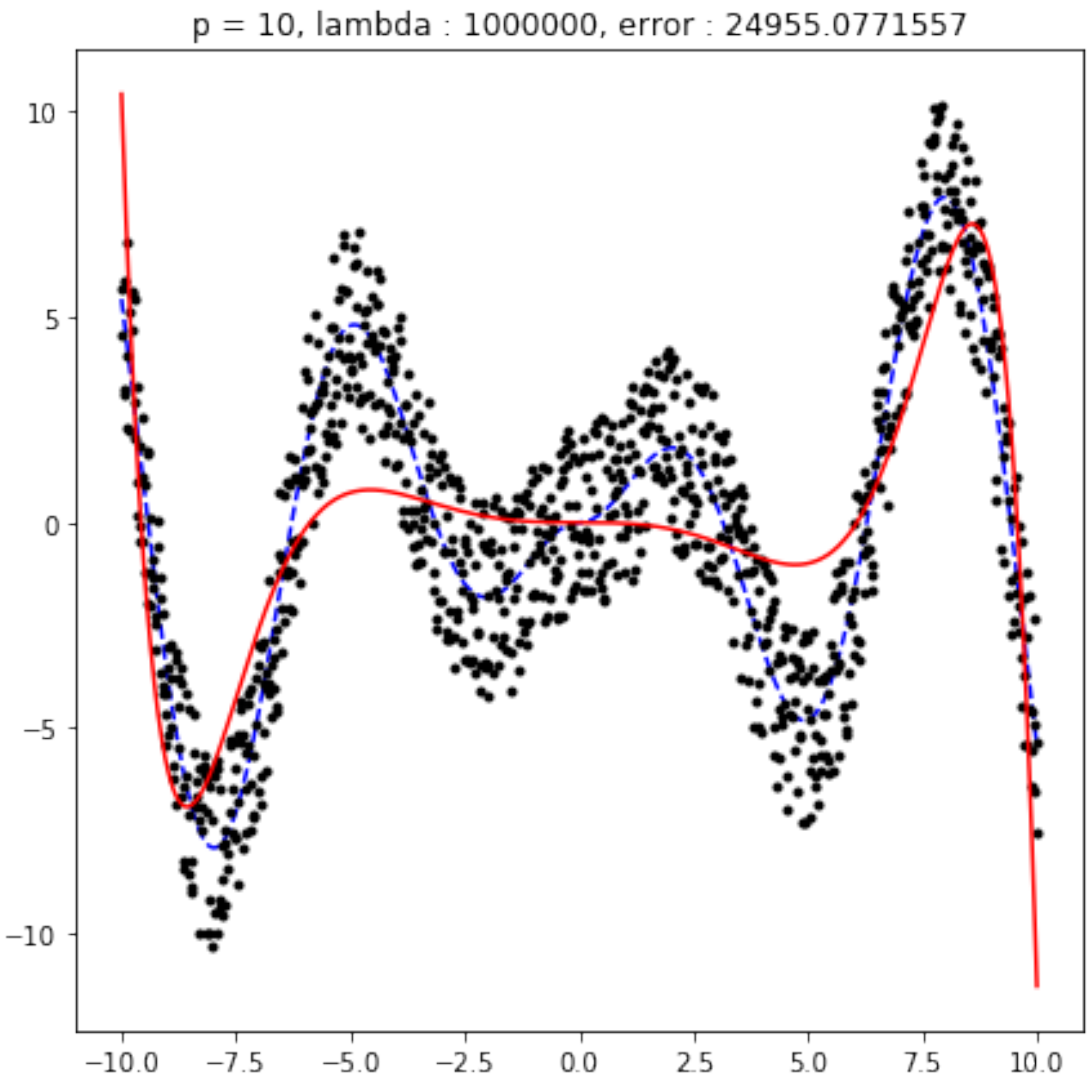

p = 9, lambda : 1, error : 5503.98298948
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

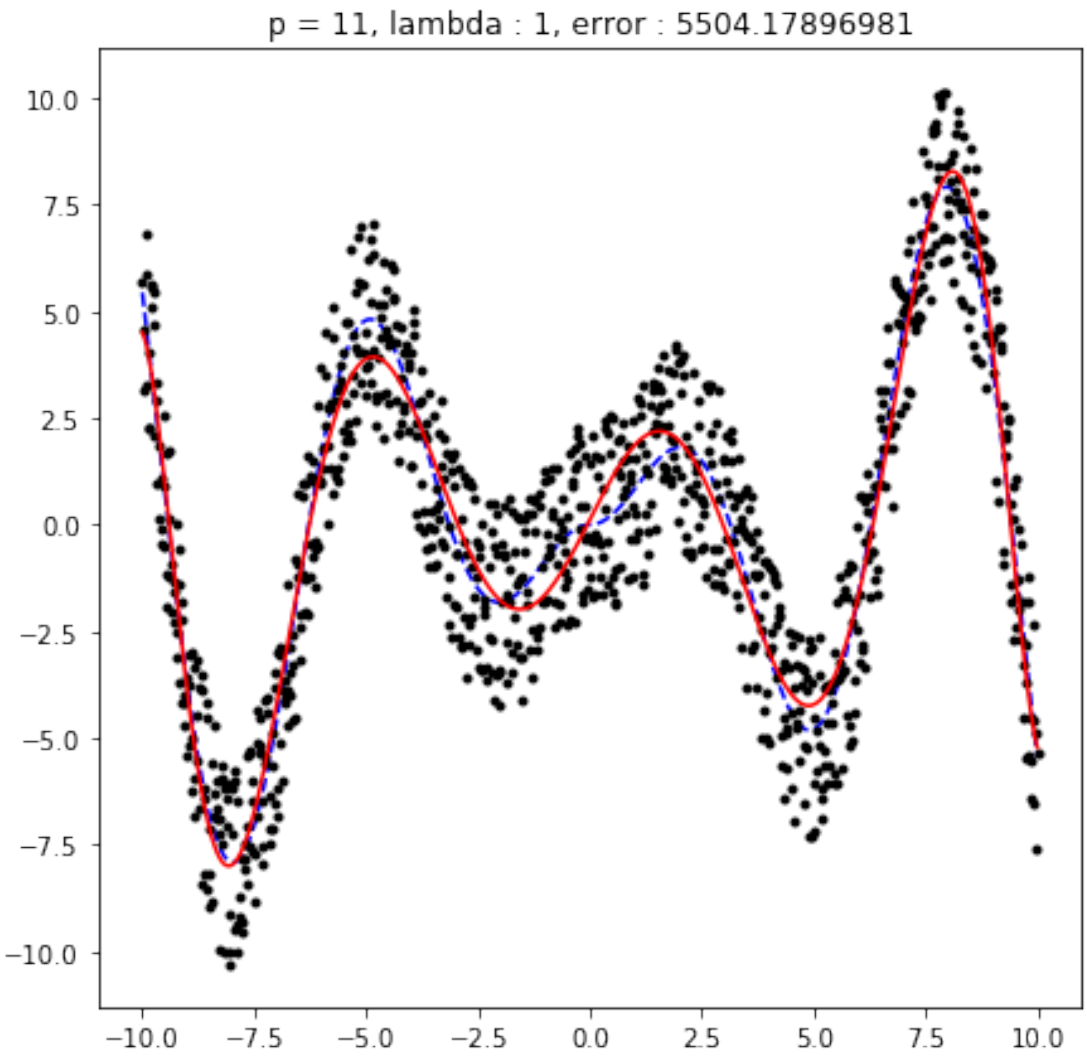

p = 9, lambda : 10000, error : 6710.61018668
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

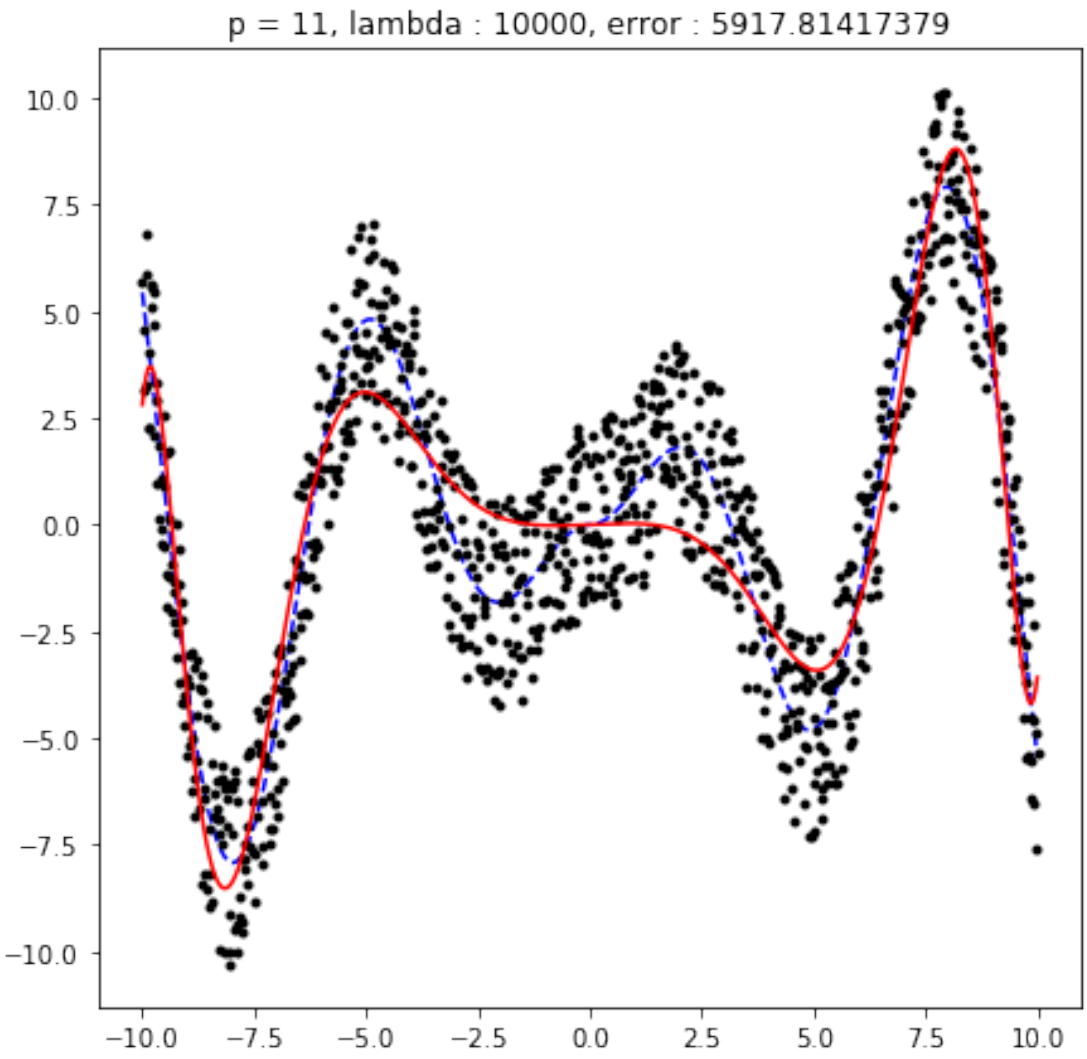

p = 9, lambda : 1000000, error : 24954.5638492
10
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

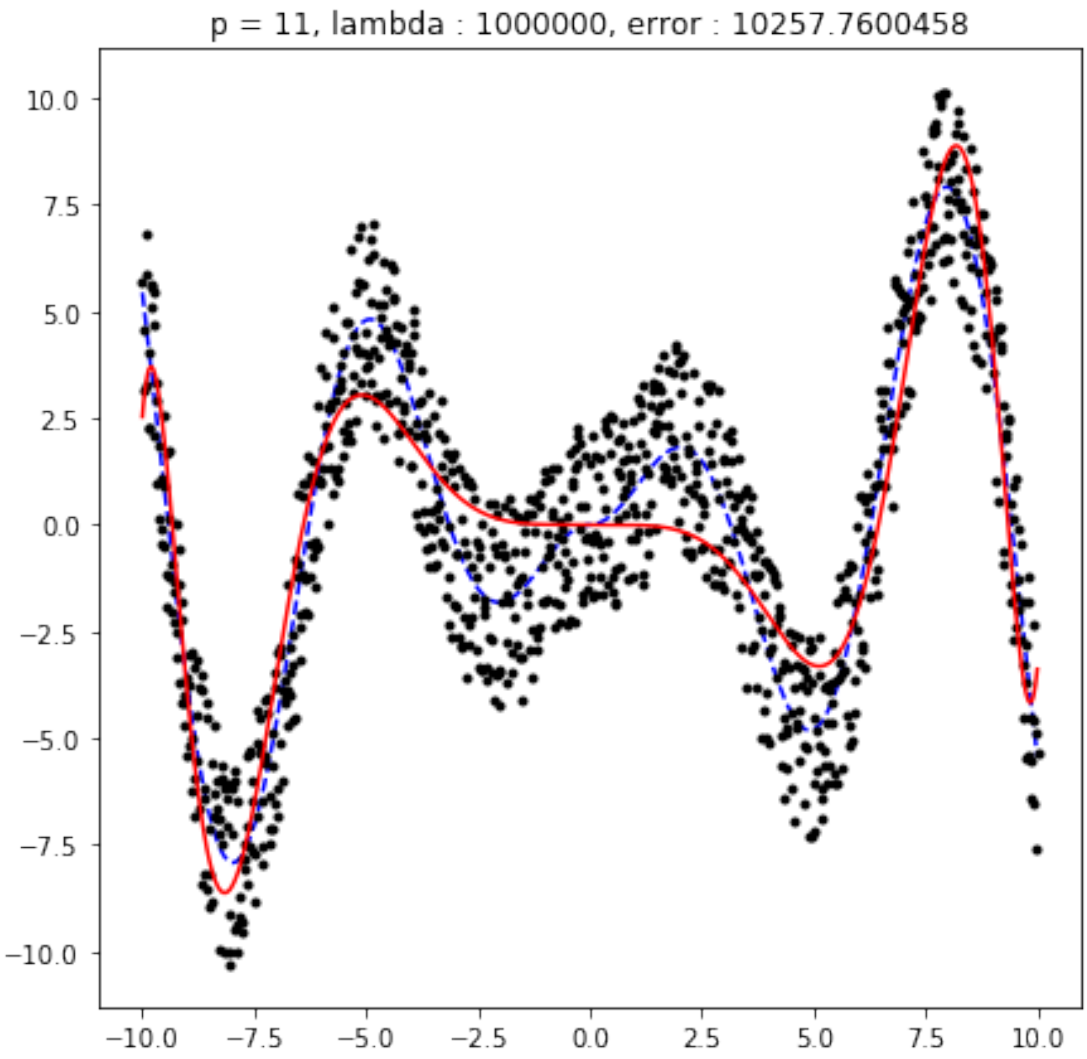

p = 10, lambda : 1, error : 5504.00965292
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

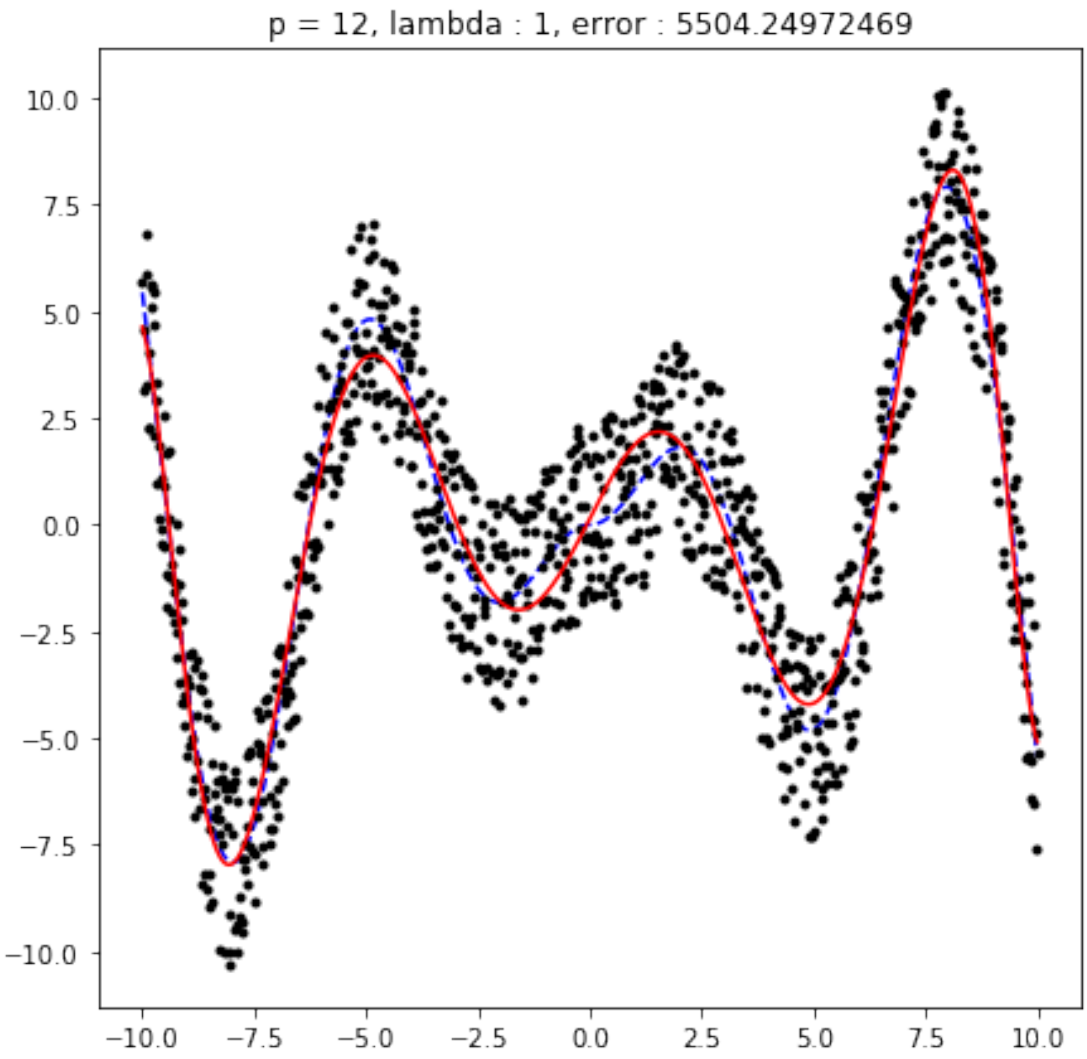

p = 10, lambda : 10000, error : 6711.56712785
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

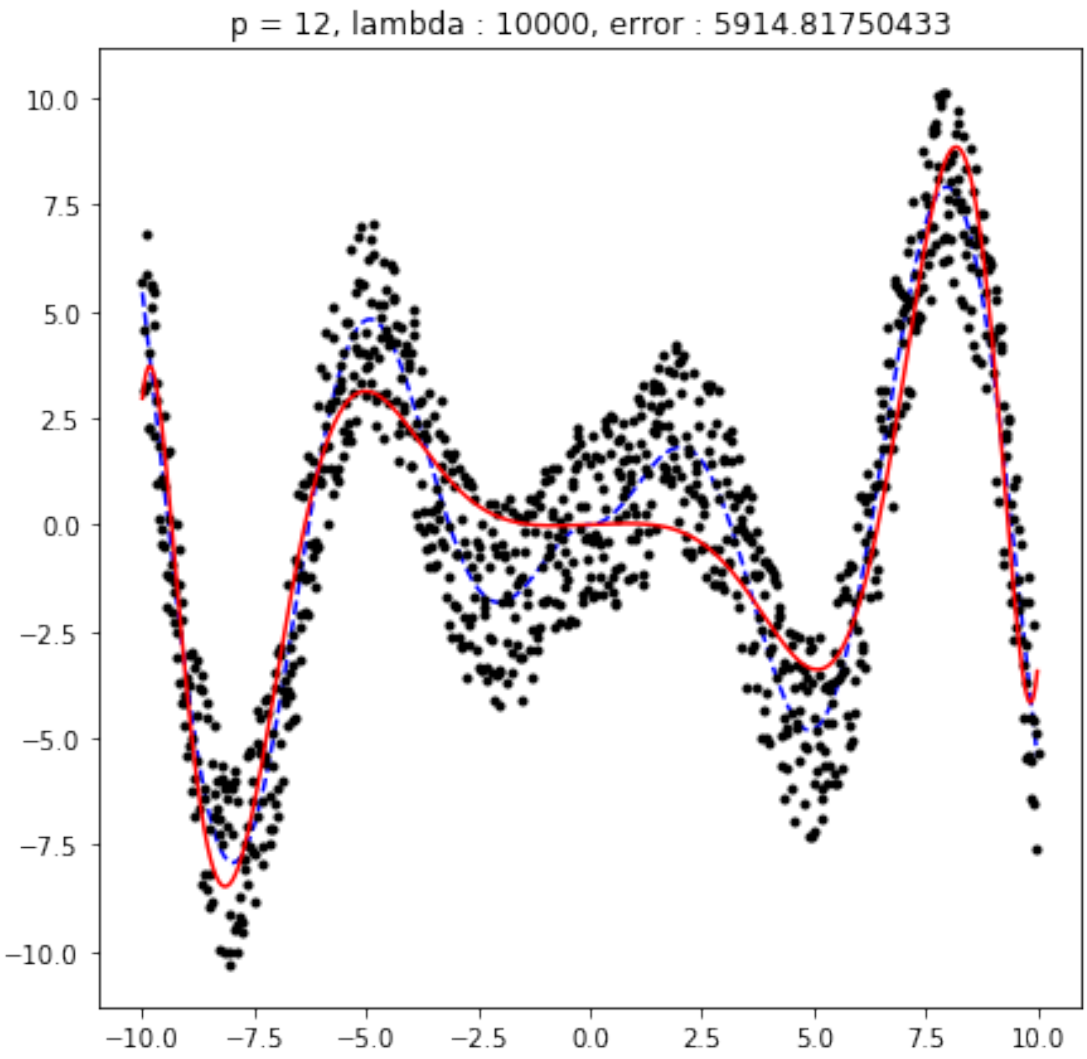

p = 10, lambda : 1000000, error : 24955.0771557
10
5
0
−5
−10
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

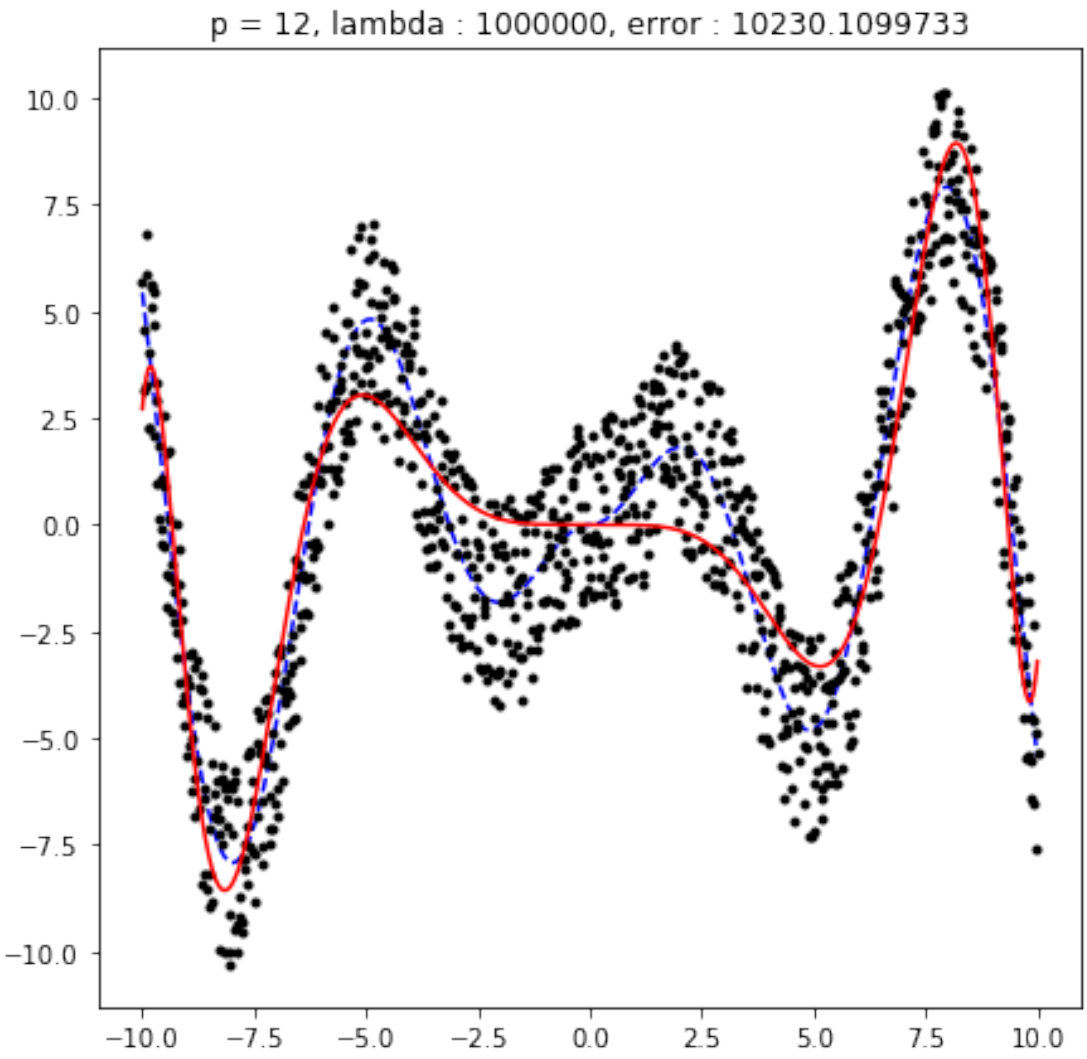

p = 11, lambda : 1, error : 5504.17896981
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

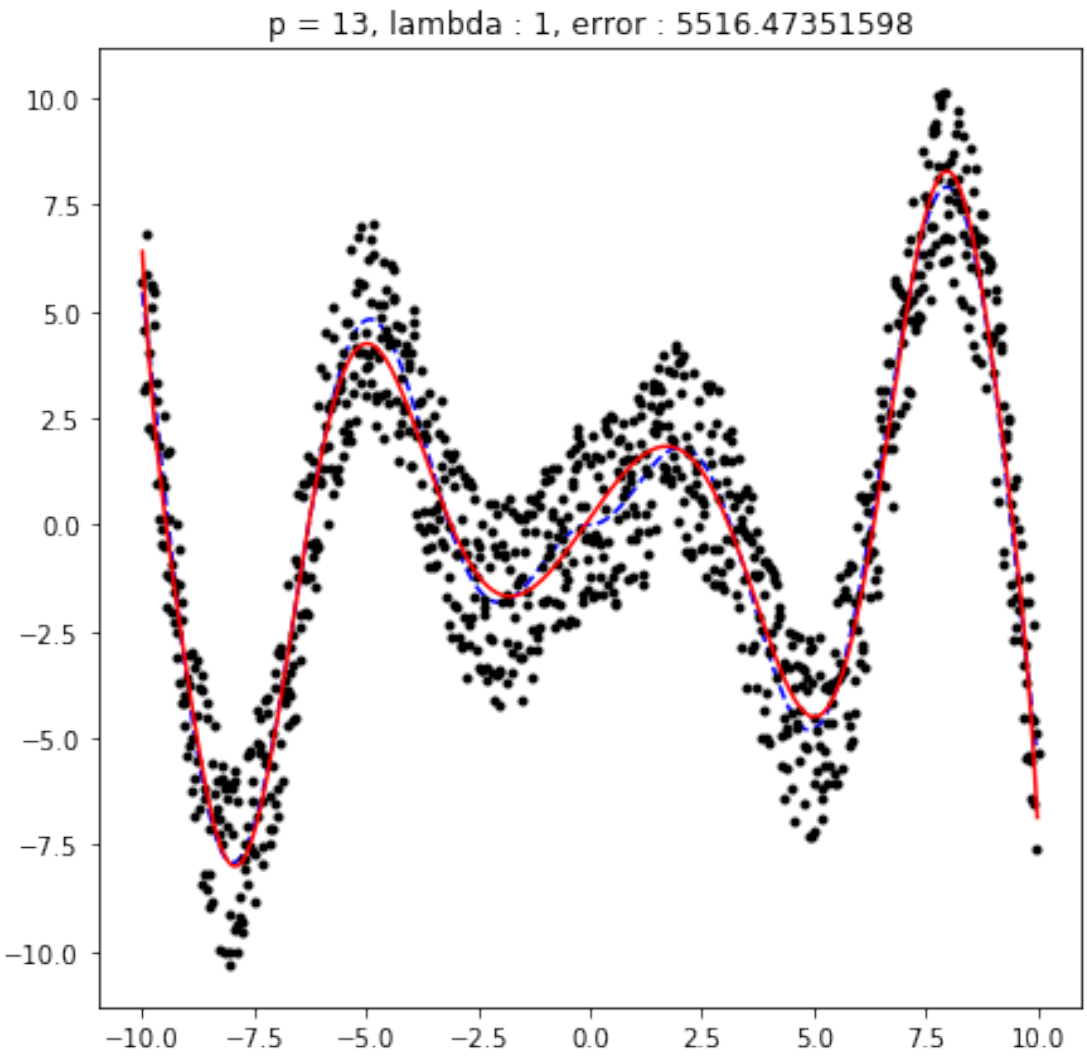

p = 11, lambda : 10000, error : 5917.81417379
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

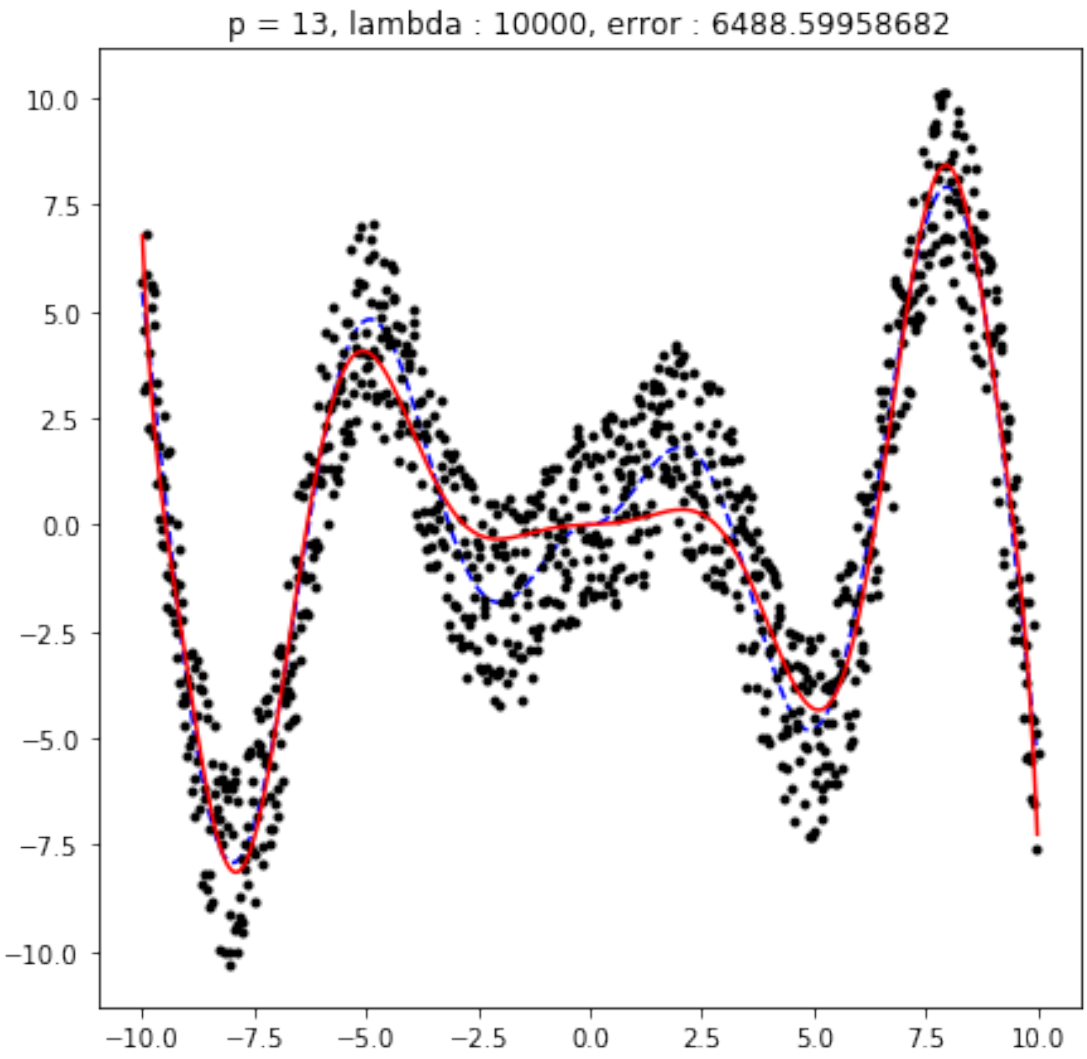

p = 11, lambda : 1000000, error : 10257.7600458
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

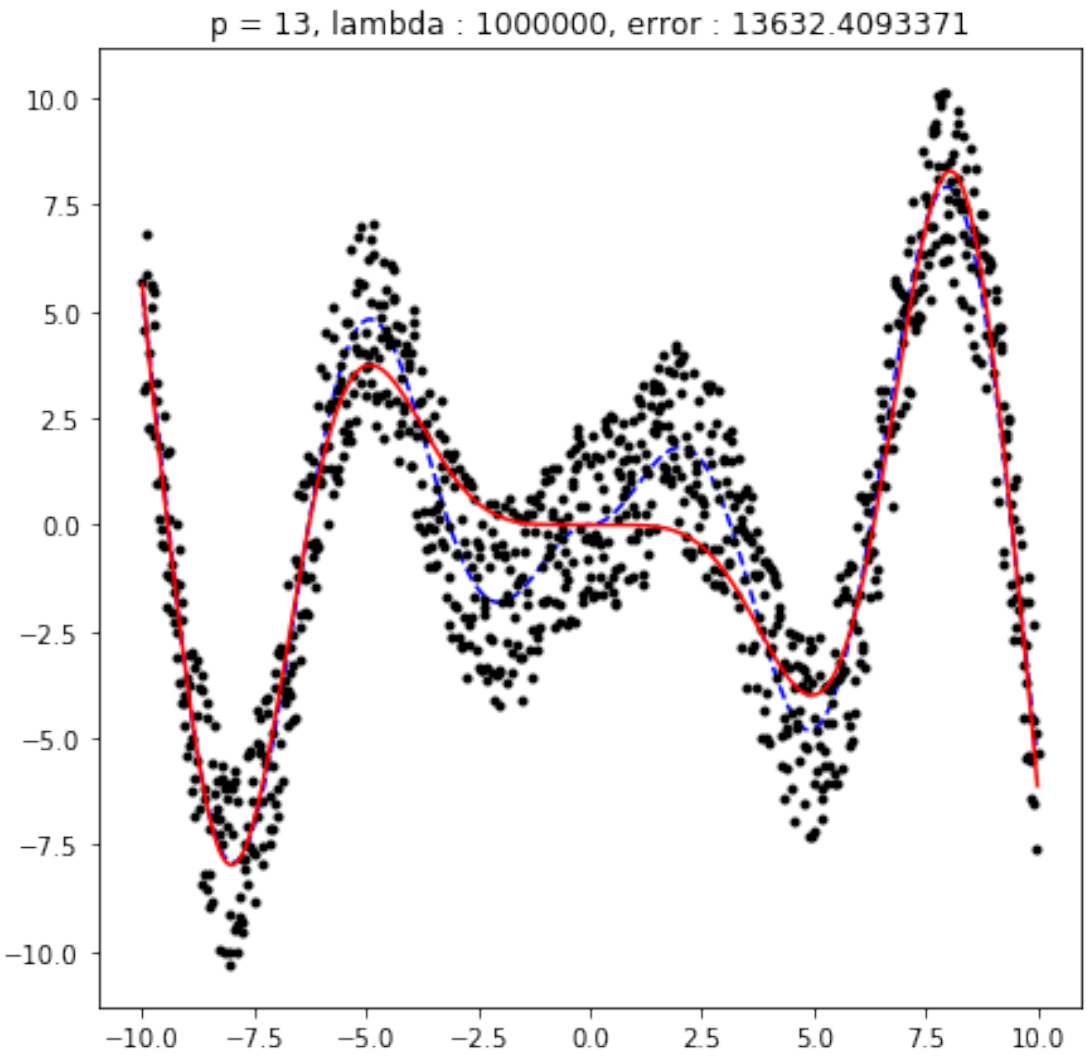

p = 12, lambda : 1, error : 5504.24972469
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

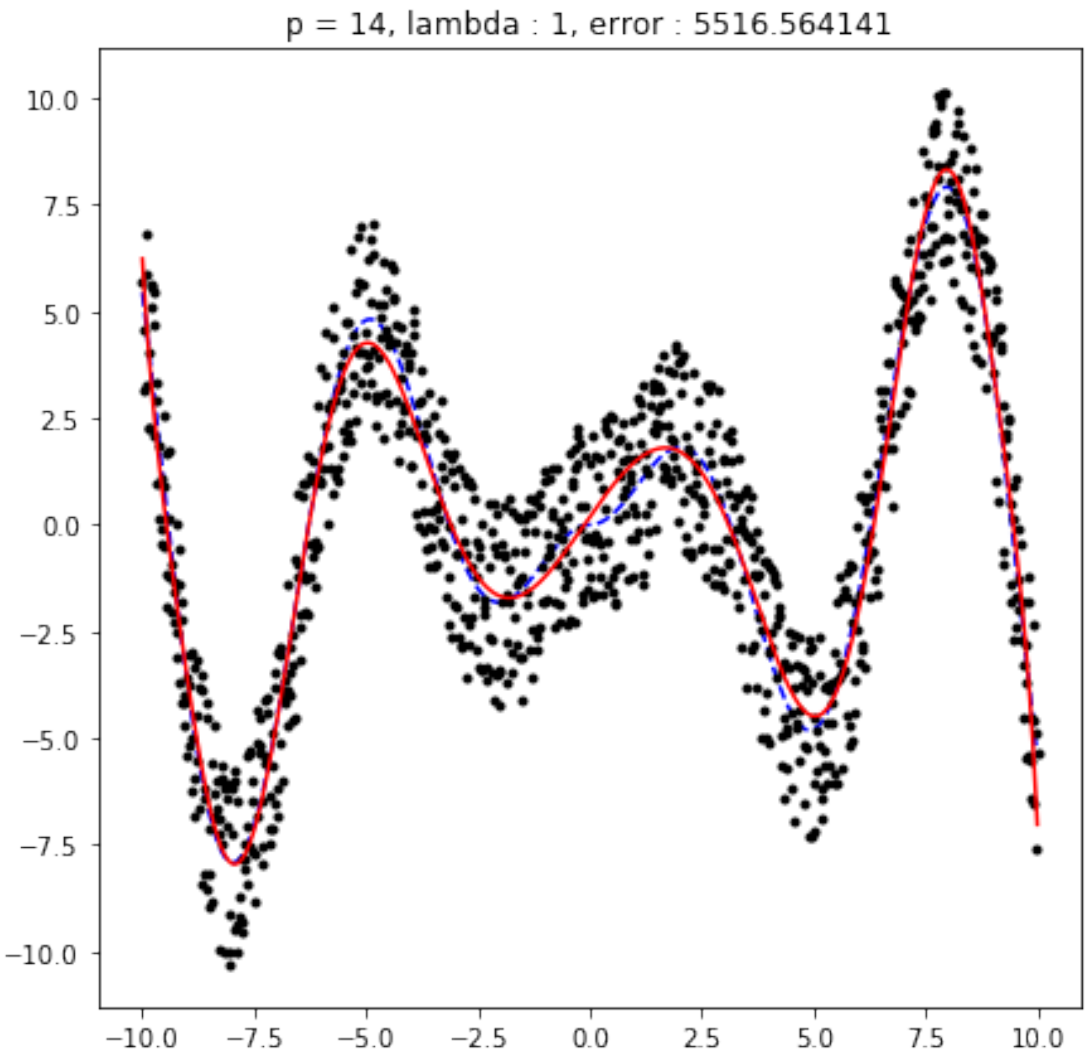

p = 12, lambda : 10000, error : 5914.81750433
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

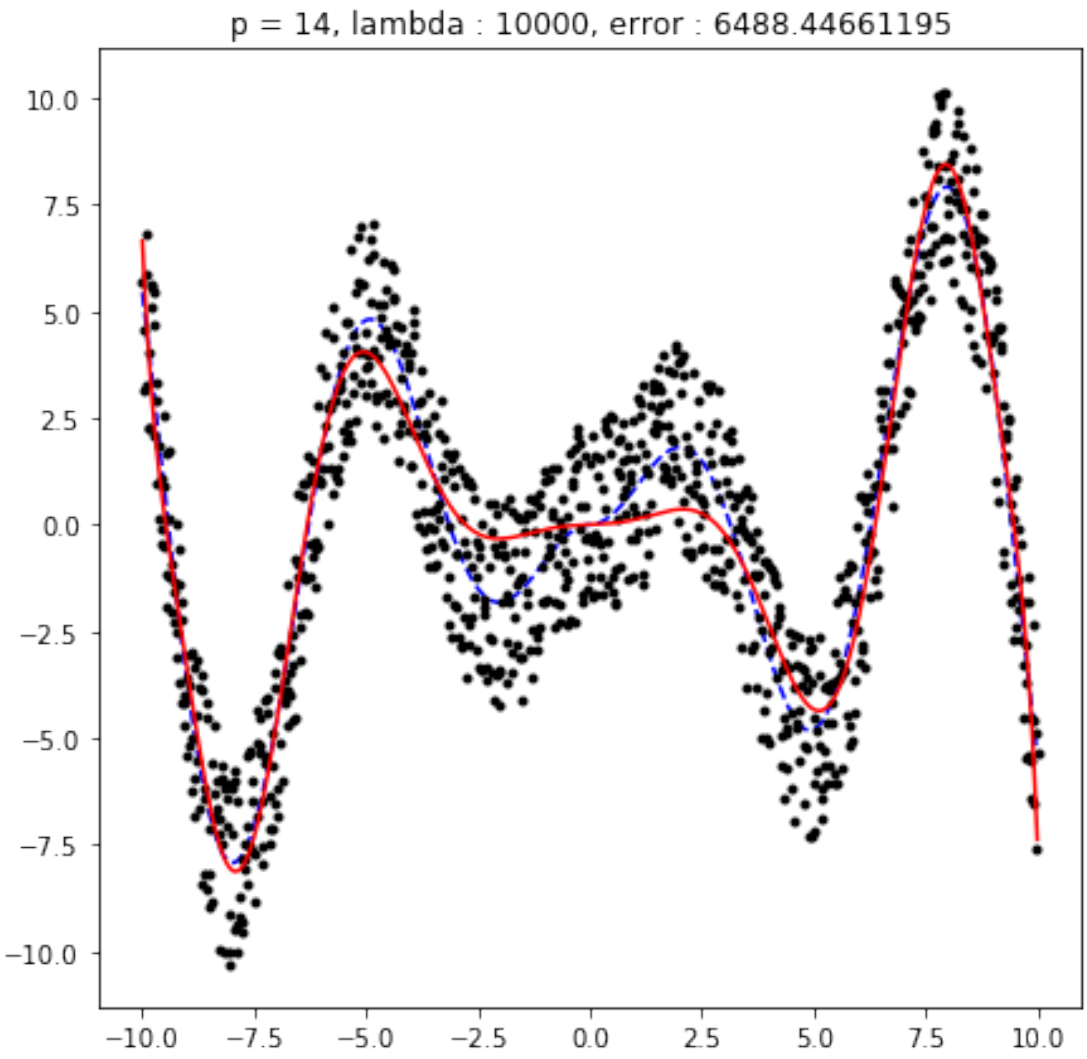

p = 12, lambda : 1000000, error : 10230.1099733
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

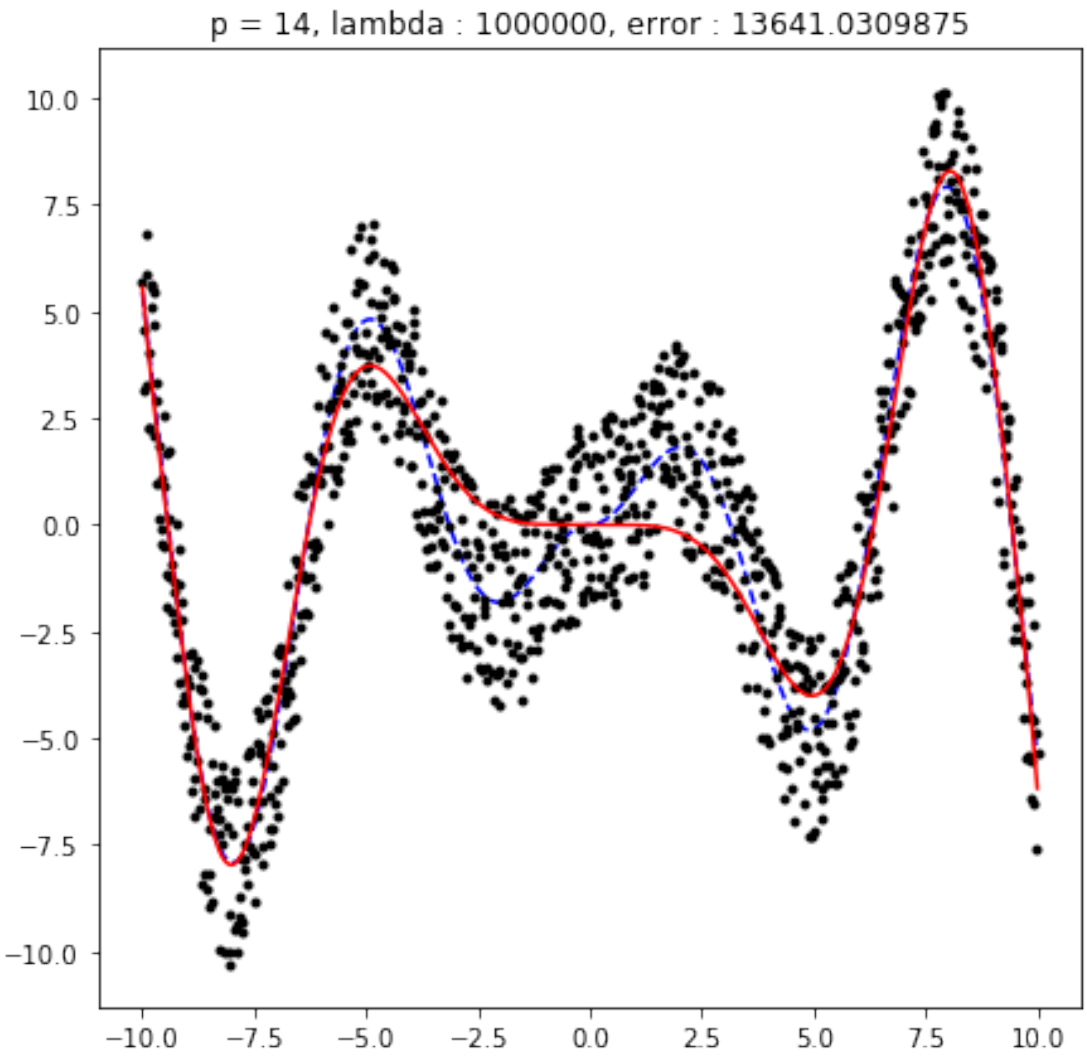

p = 13, lambda : 1, error : 5516.47351598
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

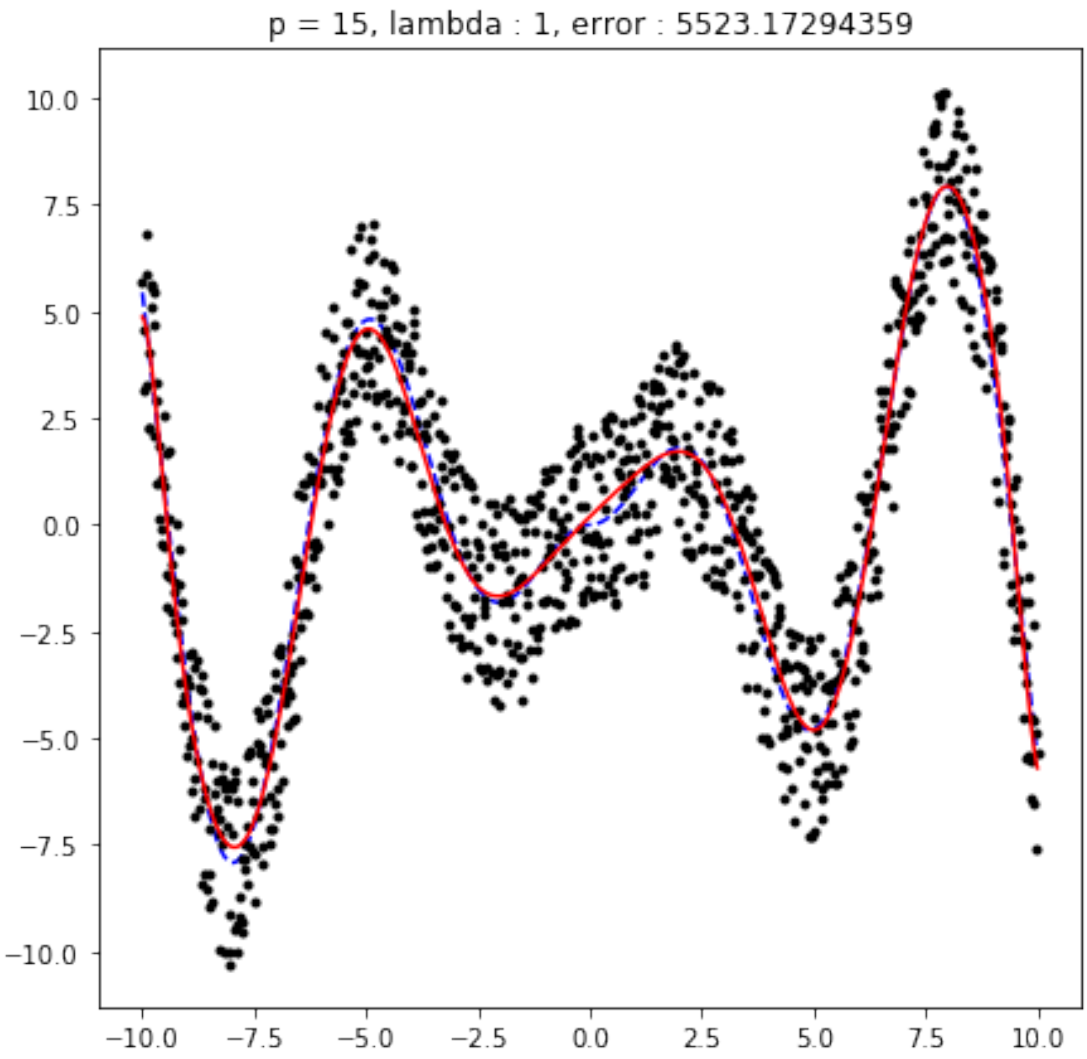

p = 13, lambda : 10000, error : 6488.59958682
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

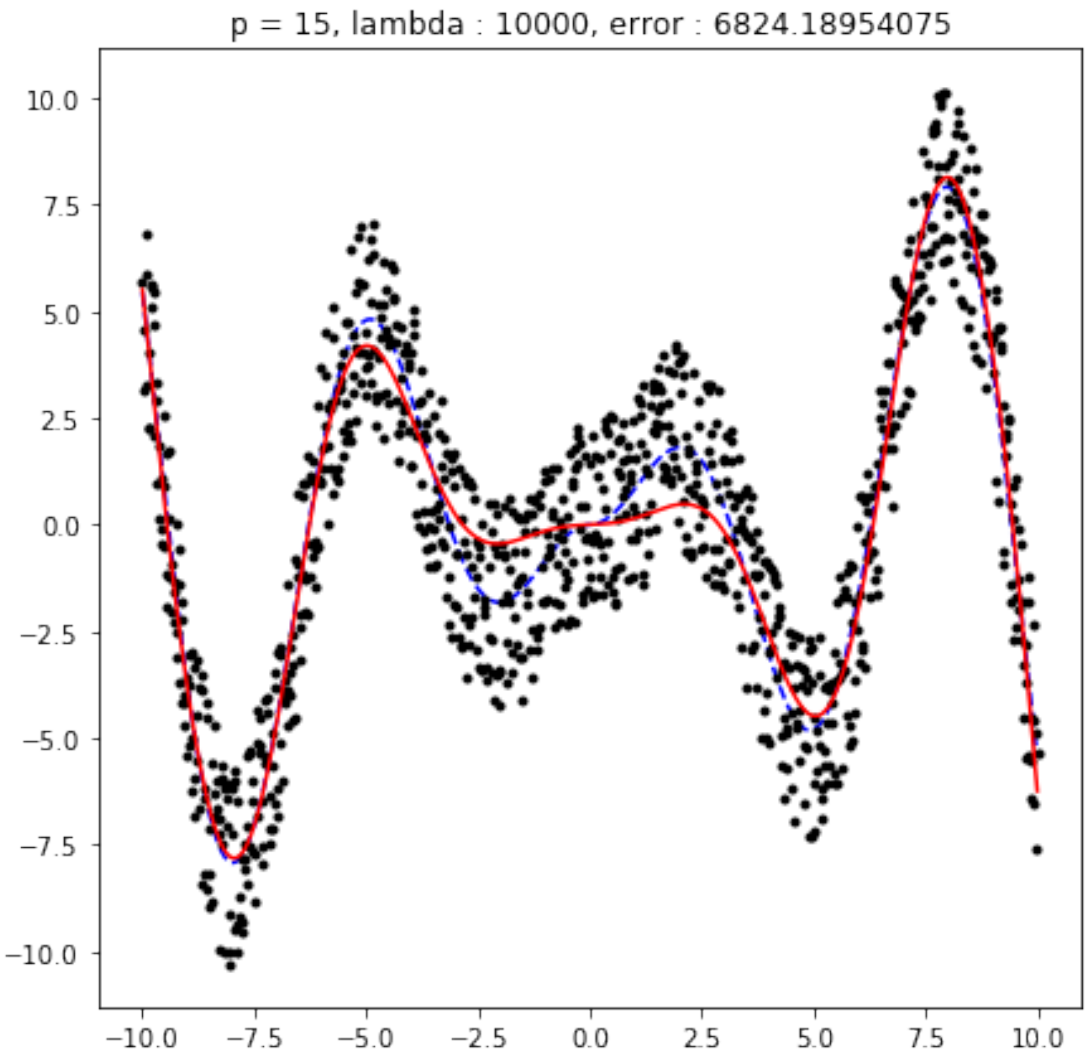

p = 13, lambda : 1000000, error : 13632.4093371
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

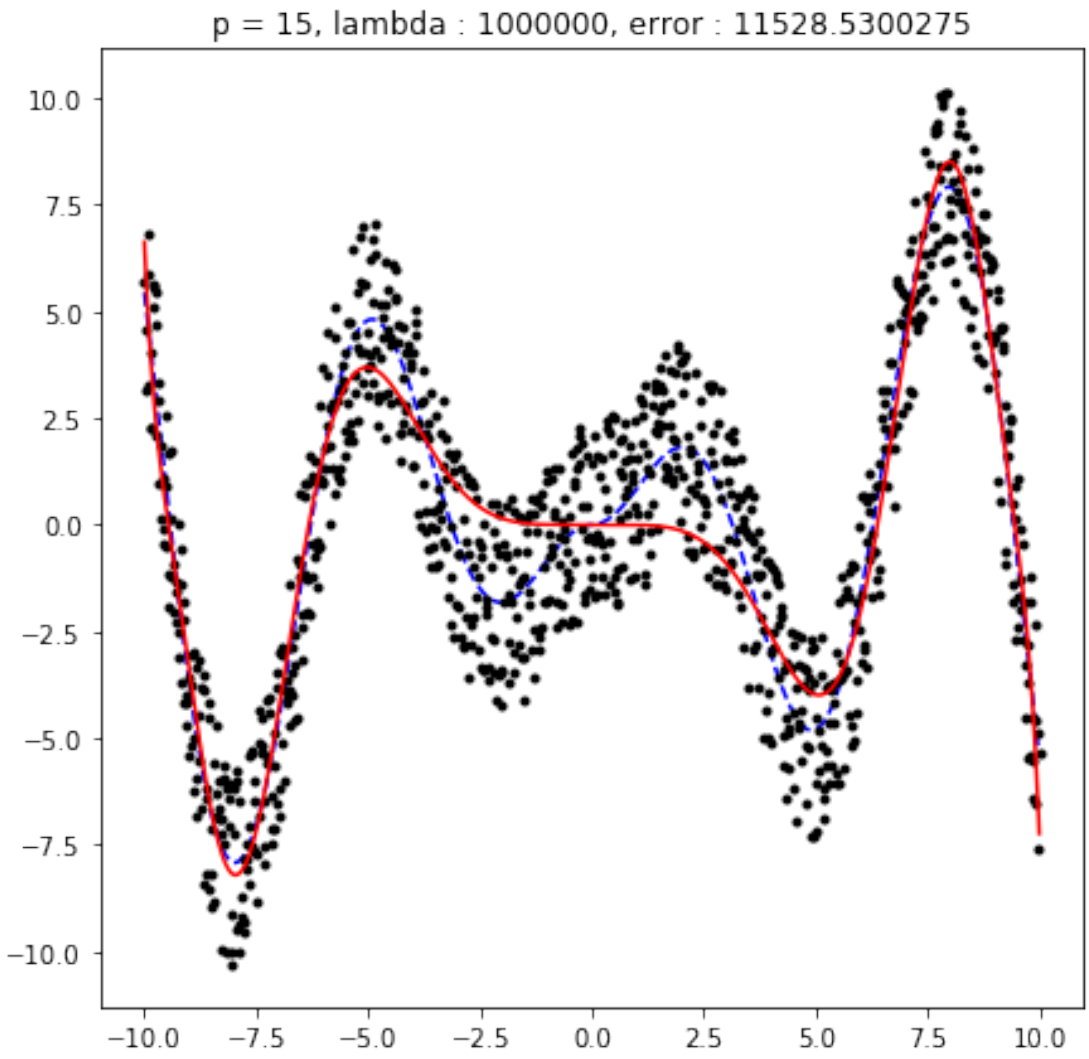

p = 14, lambda : 1, error : 5516.564141
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


p = 14, lambda : 10000, error : 6488.44661195
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0


p = 14, lambda : 1000000, error : 13641.0309875
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

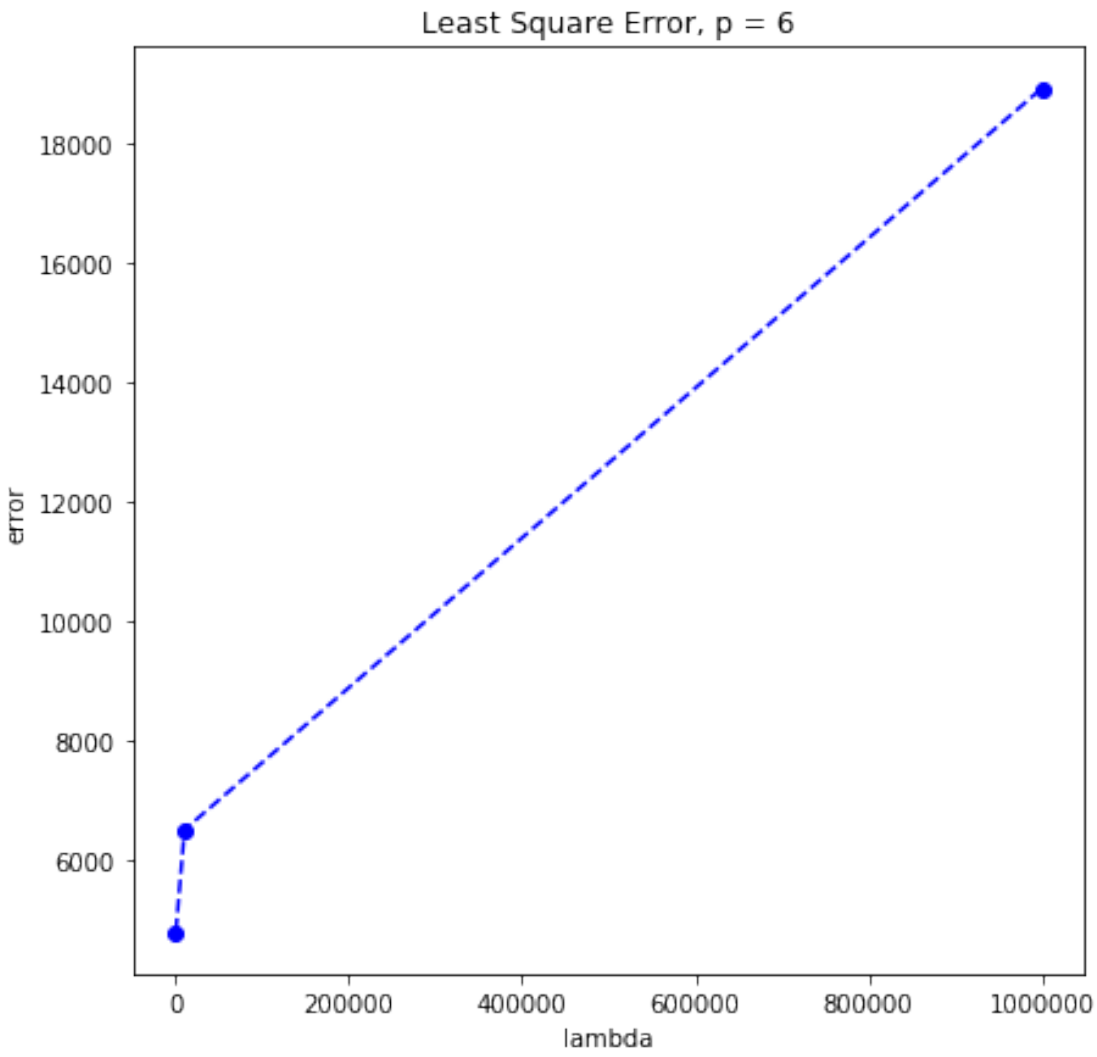

p = 15, lambda : 1, error : 5523.17294359
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

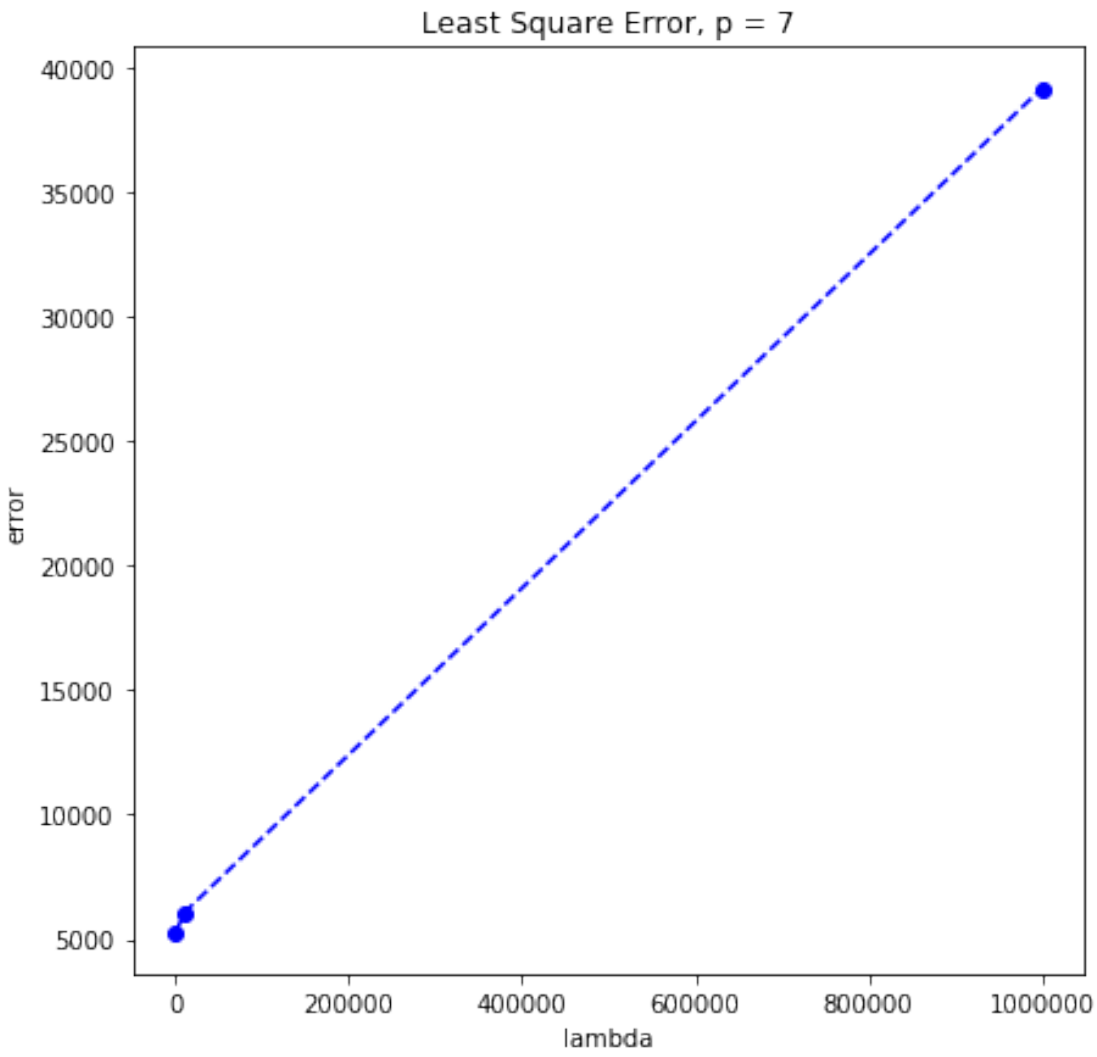

p = 15, lambda : 10000, error : 6824.18954075
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

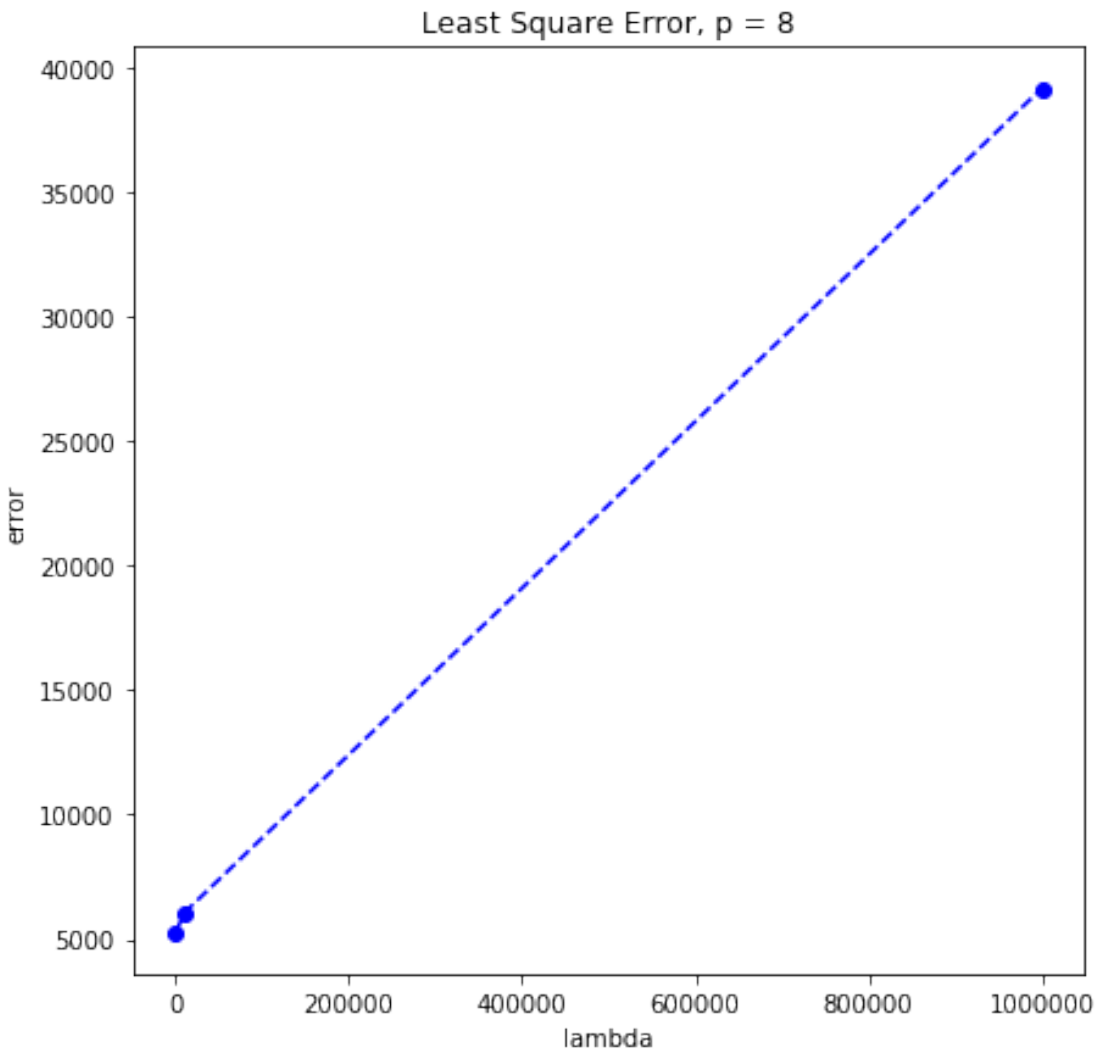


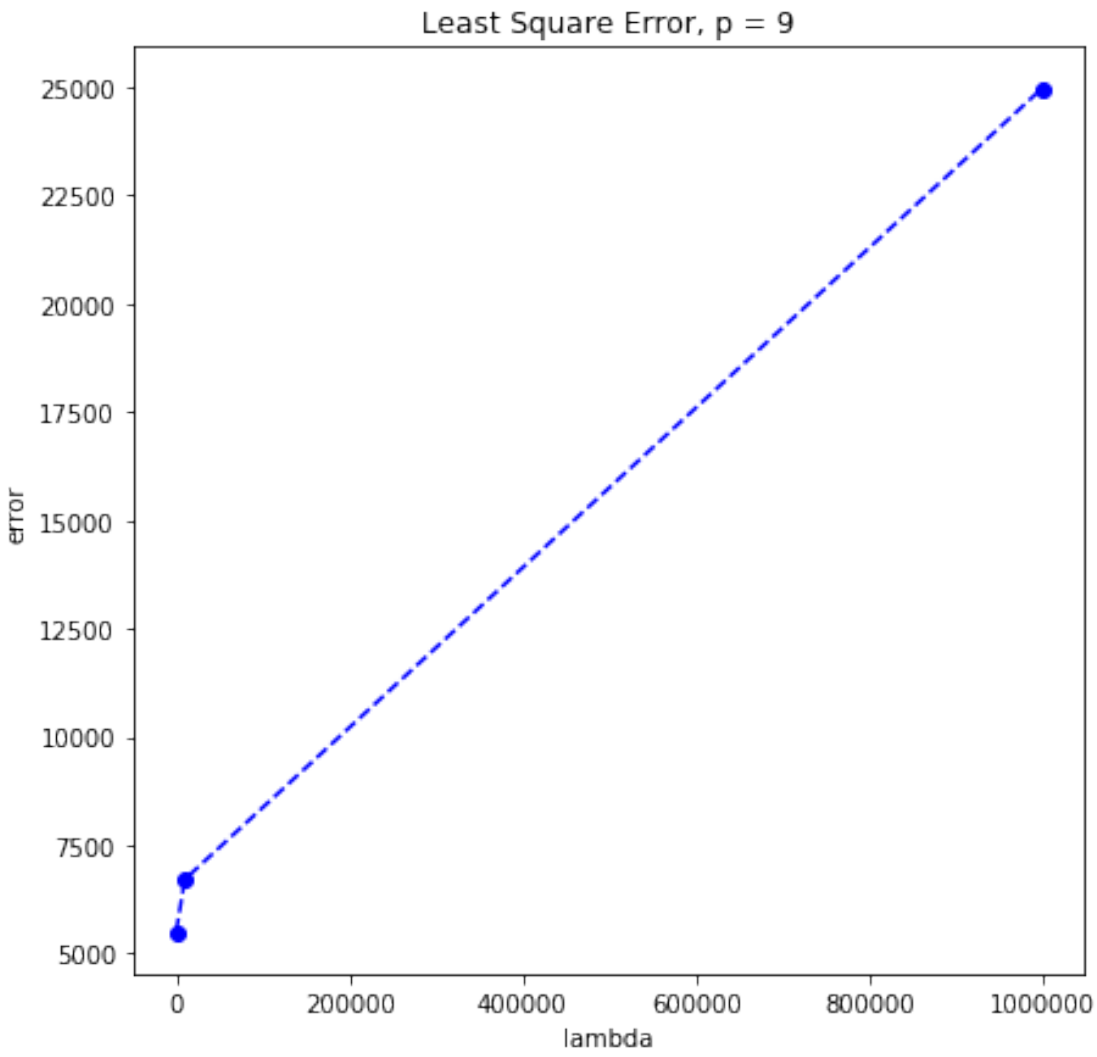
11 Plot the energy $\varepsilon(\theta, \lambda) = \|\mathbf{A} \cdot \theta - y\|^2 + \lambda\|\theta\|^2$ with varying λ

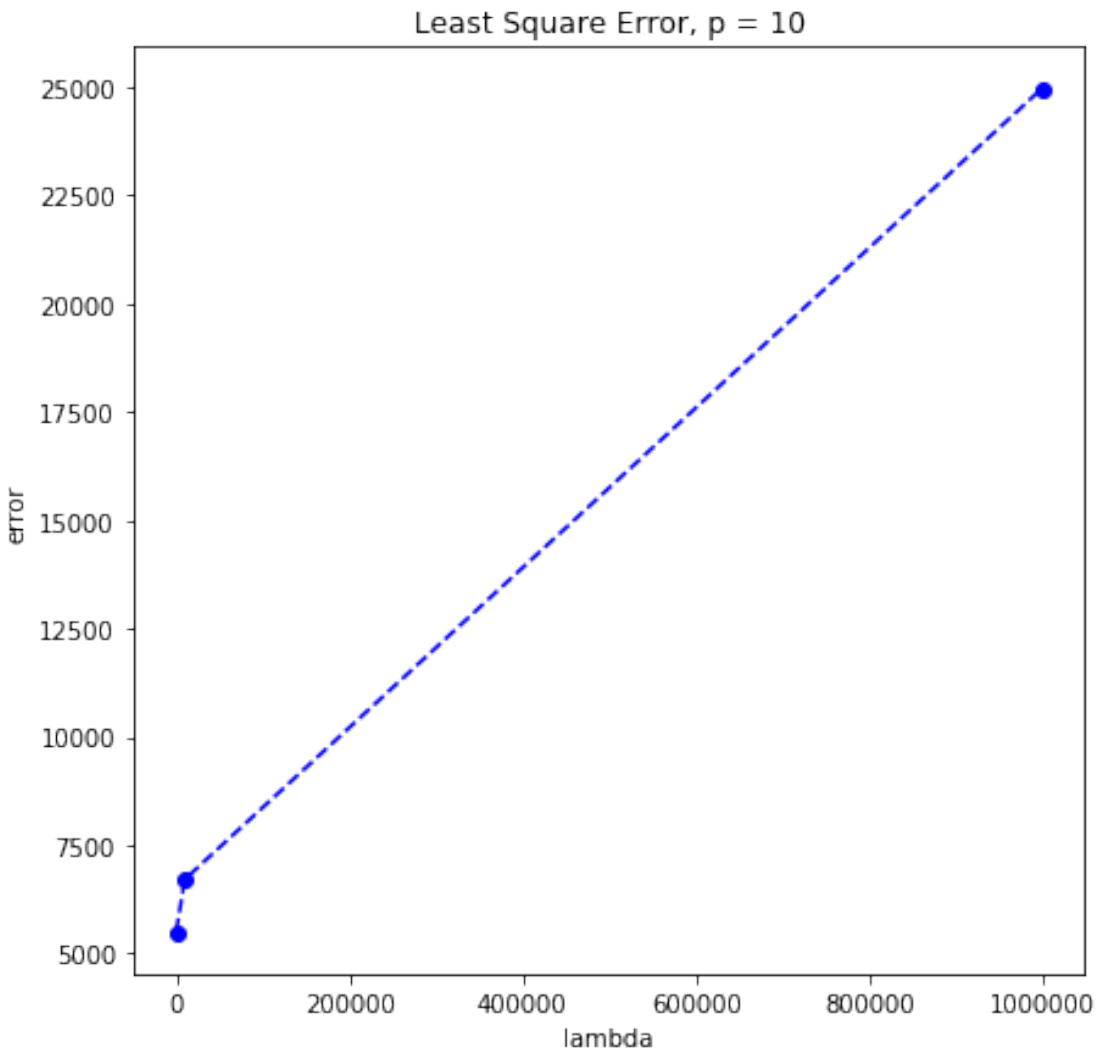
```
In [8]: for i in range(1, 10):
            plt.figure(figsize=(7,7))
            plt.plot(lamb_list, error_log[3*(i-1):3*i], color = 'b', marker = 'o', linestyle = '
            plt.title
            plt.xlabel('lambda')
            plt.ylabel('error')
            plt.title('Least Square Error' + ', p = '+ str(i + 5))
            plt.show()
```

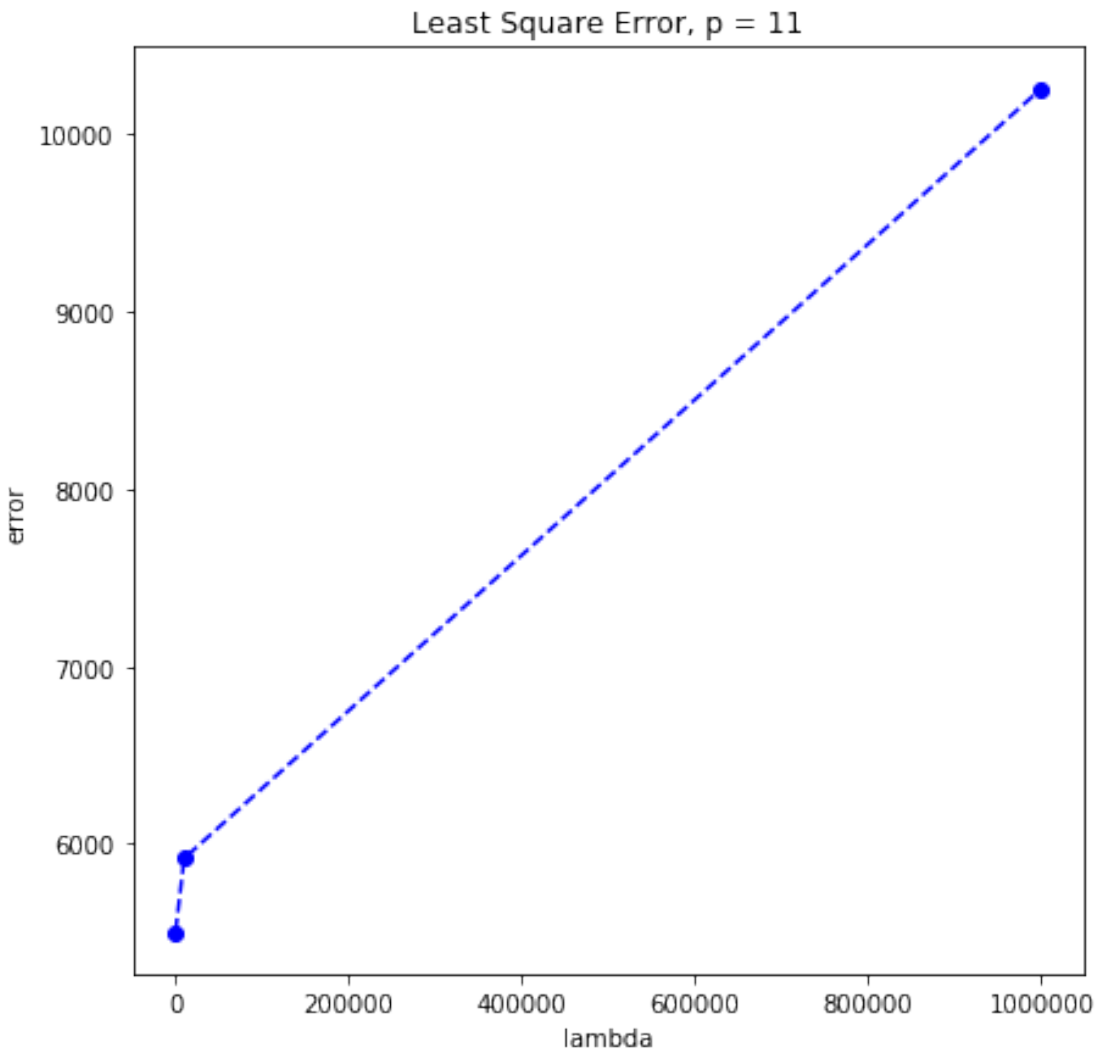

Least Square Error, p = 6
error
lambda
18000
16000
14000
12000
10000
8000
6000
0
200000
400000
600000
800000
1000000

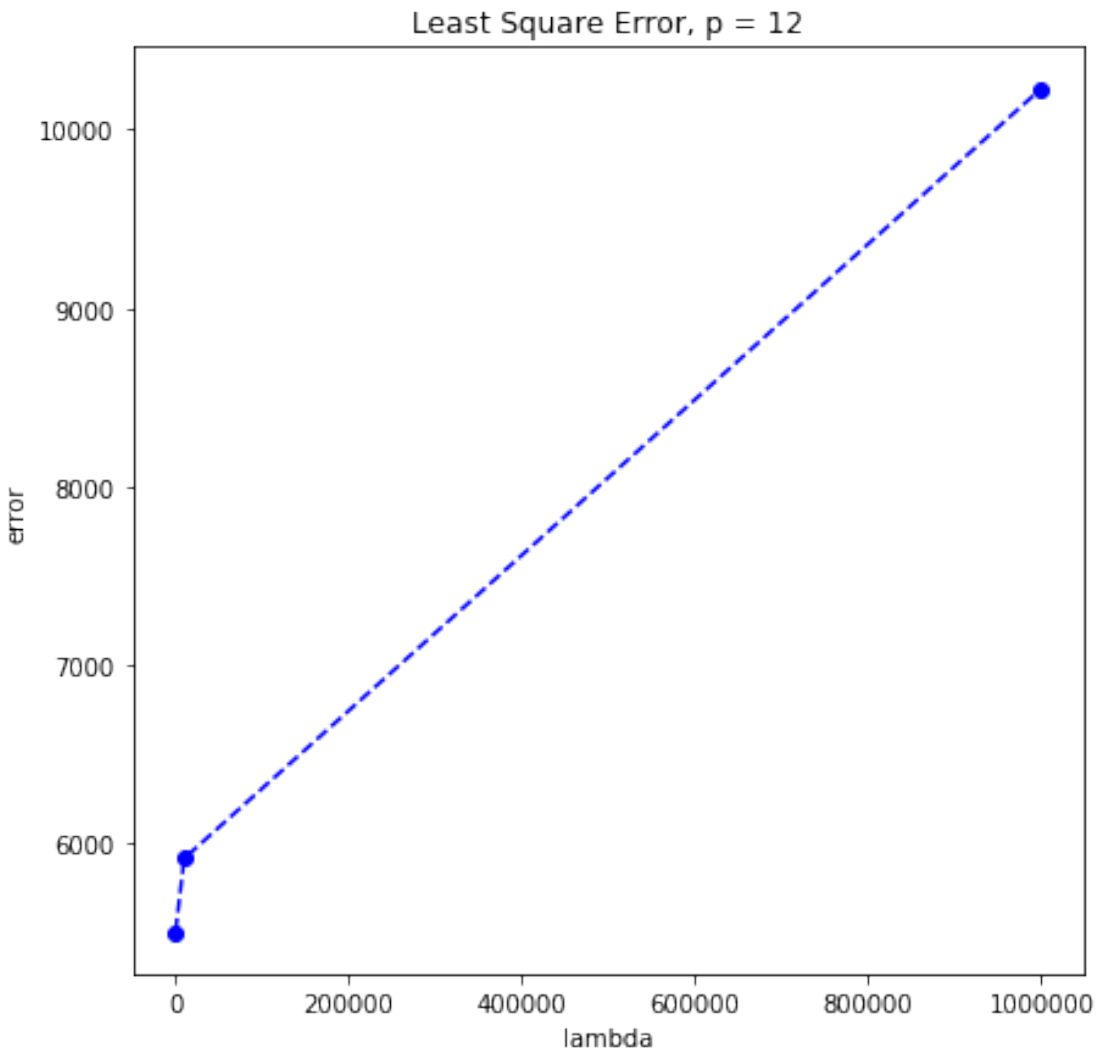

Least Square Error, p = 7
40000
35000
30000
25000
20000
15000
10000
5000
error
0
200000
400000
600000
800000
1000000
lambda

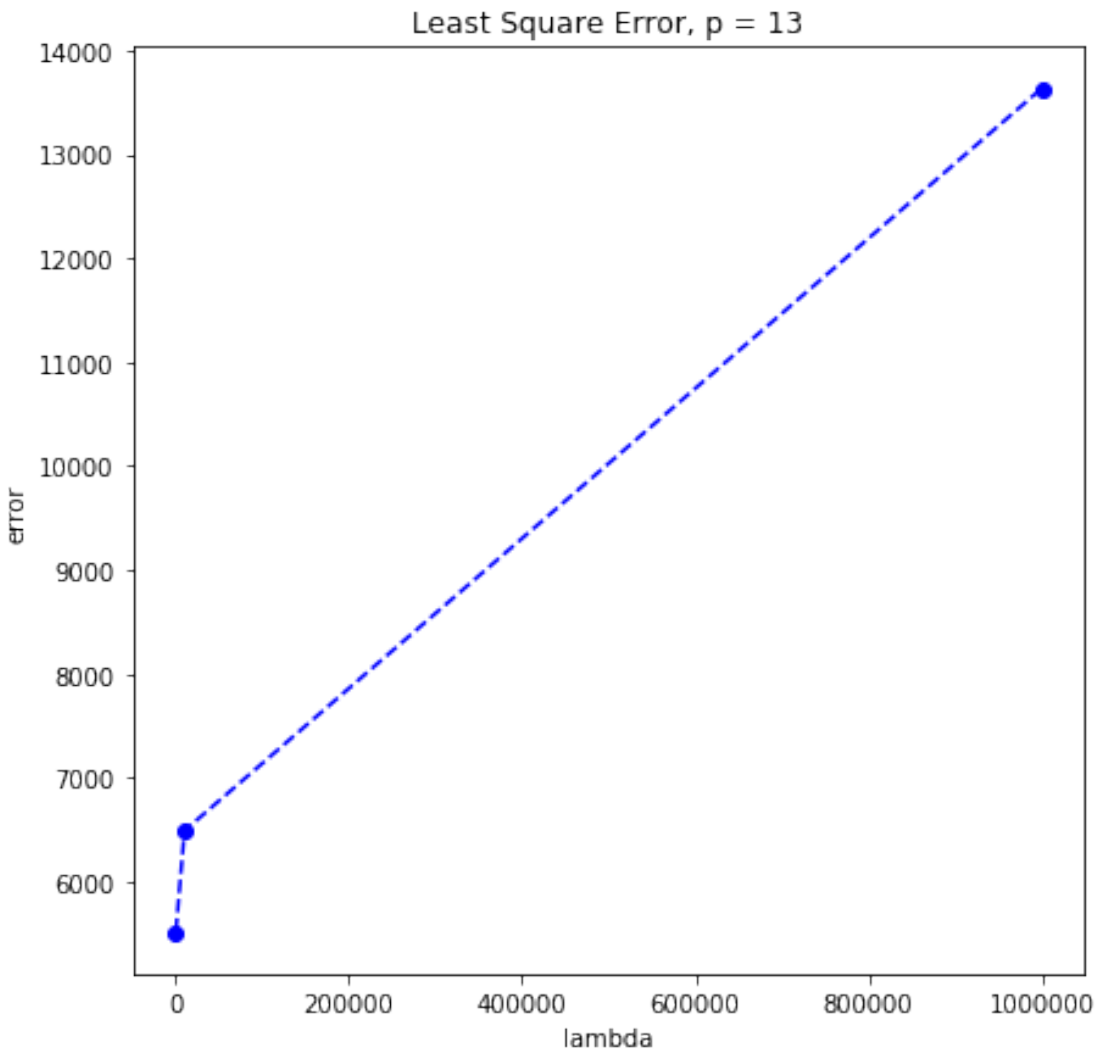

Least Square Error, p = 8
40000
35000
30000
25000
20000
15000
10000
5000
error
0
200000
400000
600000
800000
1000000
lambda

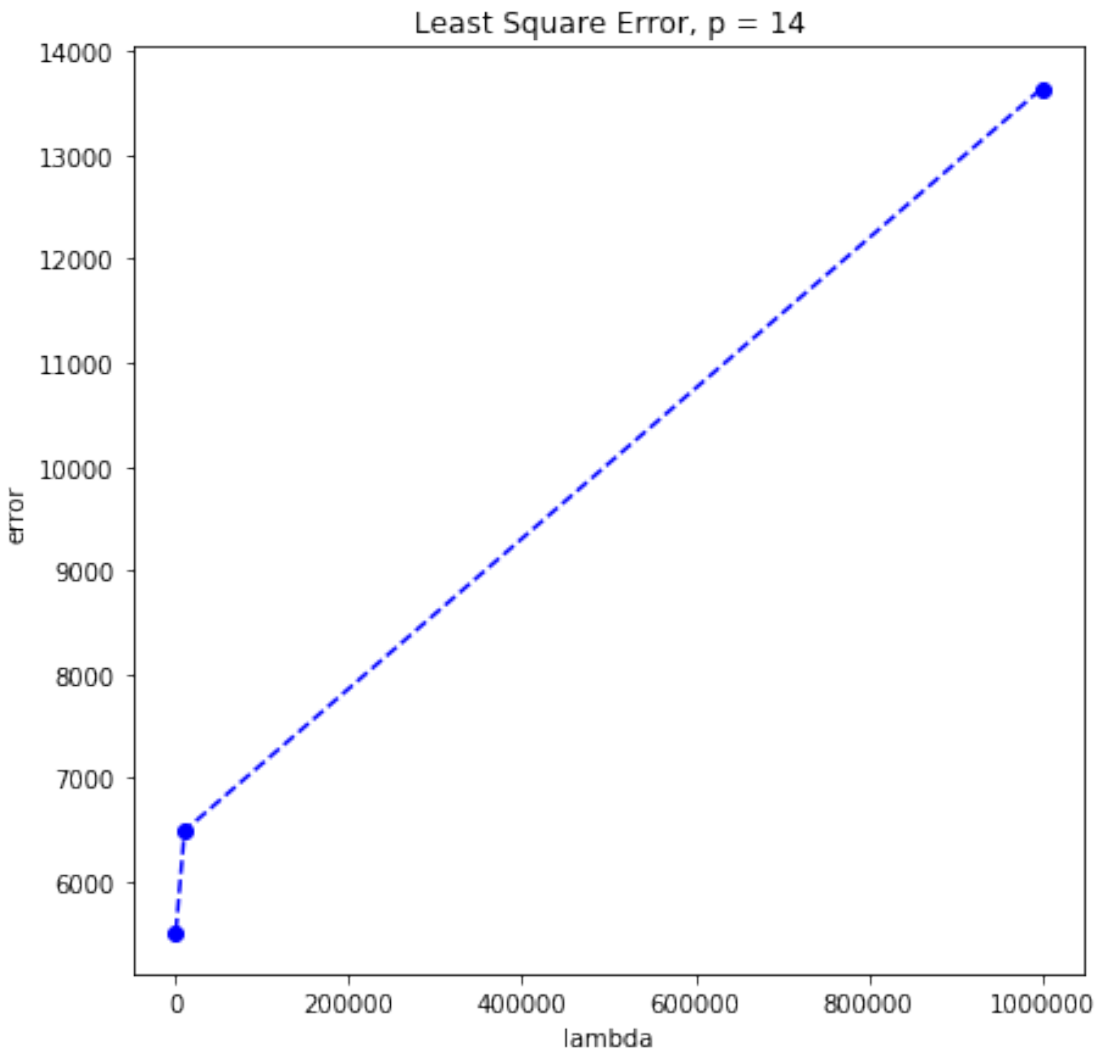

Least Square Error, p = 9
25000
22500
20000
17500
15000
12500
10000
7500
5000
error
0
200000
400000
600000
800000
1000000
lambda


Least Square Error, p = 10
25000
22500
20000
17500
15000
12500
10000
7500
5000
error
0
200000
400000
600000
800000
1000000
lambda


Least Square Error, p = 11
10000
9000
8000
7000
6000
error
0
200000
400000
600000
800000
1000000
lambda


Least Square Error, p = 12
10000
9000
8000
7000
6000
error
0
200000
400000
600000
800000
1000000
lambda


Least Square Error, p = 13
14000
13000
12000
11000
10000
9000
8000
7000
6000
error
0
200000
400000
600000
800000
1000000
lambda


Least Square Error, p = 14
14000
13000
12000
11000
10000
9000
8000
7000
6000
error
0
200000
400000
600000
800000
1000000
lambda