Least square problem for polynomial regression

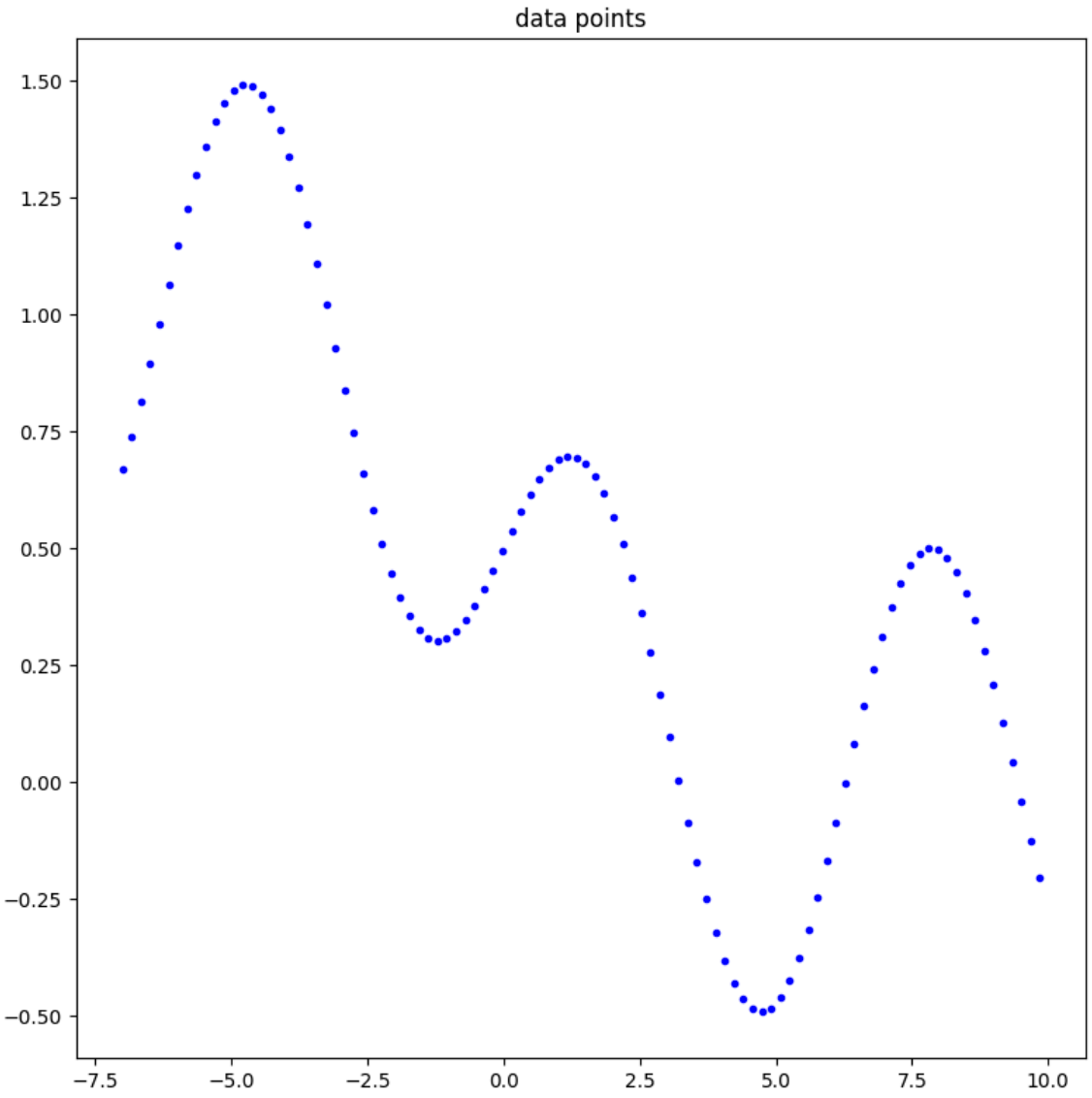
import library

In [ ]:
```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors as colors
import util
```

load a set of two dimensional point data $(x, y) \in \mathbb{R}^2$

In [ ]:
```
filename    = '06_data.csv'
data        = np.loadtxt(filename, delimiter = ',')
x           = data[0, ::10]    # independent variable
y           = data[1, ::10]    # dependent variable
```

In [ ]:
```
plt.figure(figsize=(8,8))
plt.plot(x, y, '.', color = 'blue')
plt.title('data points')
plt.tight_layout()
plt.show()
```



In [ ]:
```
print('num of x =', len(x))
print('num of y =', len(y))
```

```
num of x = 100
num of y = 100
```

construct matrix A as given by:

$$A = \begin{bmatrix} x_1^0 & x_1^1 & \cdots & x_1^{p-1} \\ x_2^0 & x_2^1 & \cdots & x_2^{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ x_n^0 & x_n^1 & \cdots & x_n^{p-1} \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

solve the polynomial regression problem using the numpy functions such as:

`np.matmul` , `np.linalg.inv` , `np.transpose` , `np.identity` , `np.power`

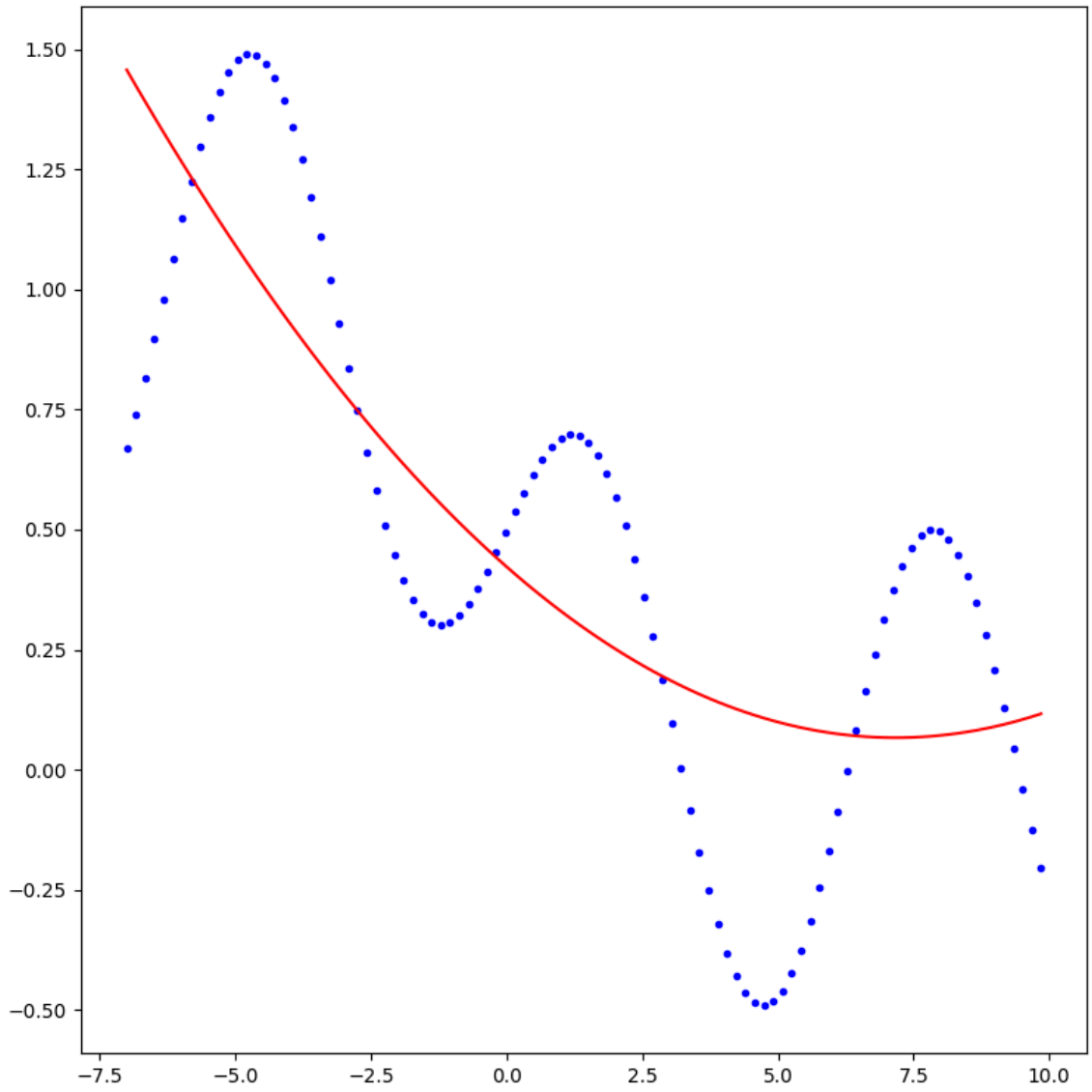
In [ ]:
```
p1          = 2
alpha1      = 0
A1          = util.get_matrix_A_regression_polynomial(x, p1)
f1, theta1  = util.compute_regression_polynomial(A1, y, alpha1)
loss1       = util.compute_loss_regression_polynomial(A1, y, theta
1, alpha1)
```

In [ ]:
```
print('size of A =', A1.shape)
print('size of f =', len(f1))
print('loss =', loss1)
```

```
size of A = (100, 3)
size of f = 100
loss = 0.0558576756688038
```

In [ ]:
```
def plot_01():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f1, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss1)
```

In [ ]:
```
plot_01()
```




```
loss = 0.0558576756688038
```

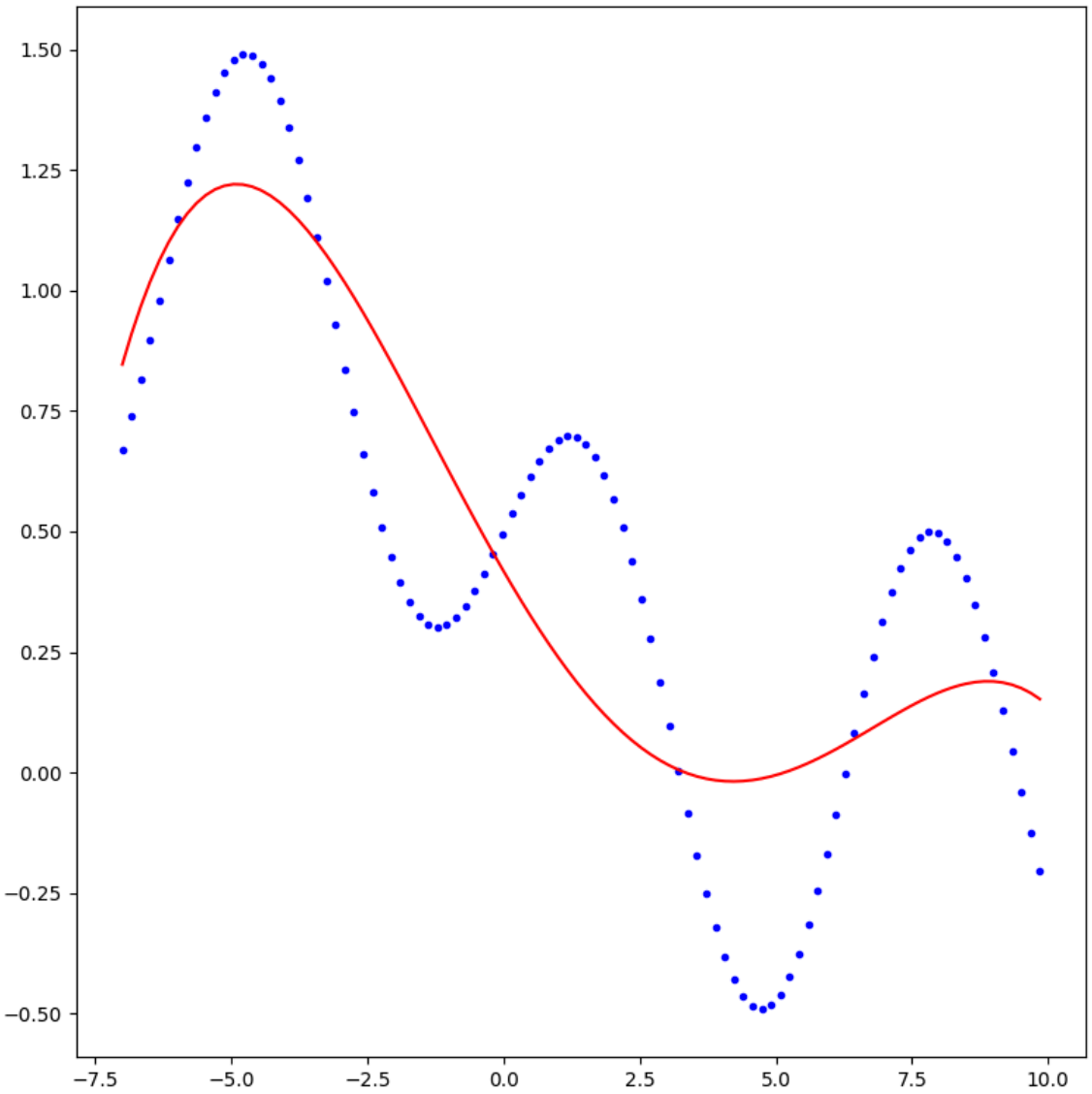
In [ ]:
```
p2          = 4
alpha2      = 0
A2          = util.get_matrix_A_regression_polynomial(x, p2)
f2, theta2  = util.compute_regression_polynomial(A2, y, alpha2)
loss2       = util.compute_loss_regression_polynomial(A2, y, theta
2, alpha2)
```

In [ ]:
```
print('size of A =', A2.shape)
print('size of f =', len(f2))
print('loss =', loss2)
```

```
size of A = (100, 5)
size of f = 100
loss = 0.04184288075723241
```

In [ ]:
```
def plot_02():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f2, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss2)
```

In [ ]:
```
plot_02()
```




```
loss = 0.04184288075723241
```

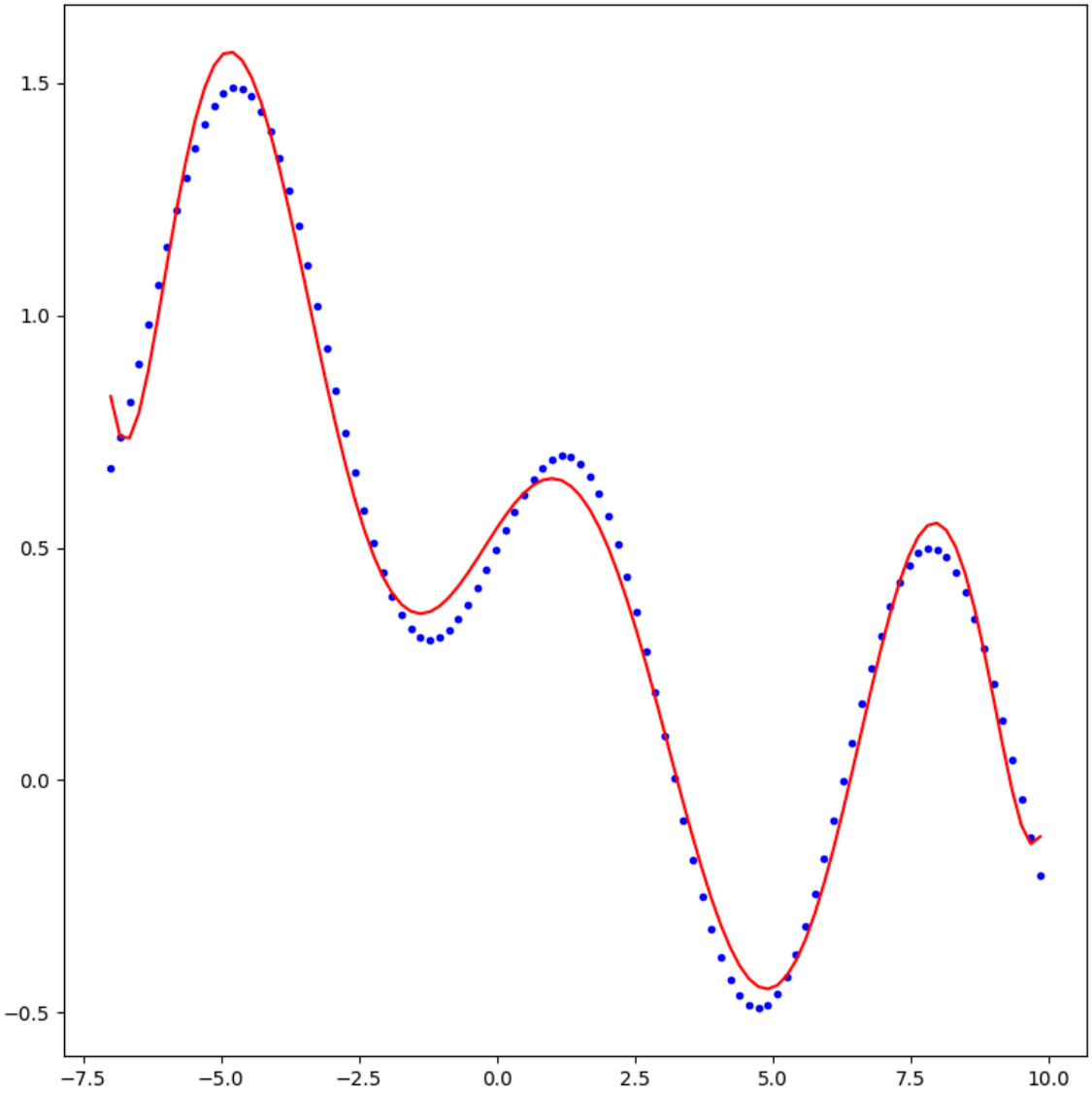
In [ ]:
```
p3          = 8
alpha3      = 0
A3          = util.get_matrix_A_regression_polynomial(x, p3)
f3, theta3  = util.compute_regression_polynomial(A3, y, alpha3)
loss3       = util.compute_loss_regression_polynomial(A3, y, theta
3, alpha3)
```

In [ ]:
```
print('size of A =', A3.shape)
print('size of f =', len(f3))
print('loss =', loss3)
```

```
size of A = (100, 9)
size of f = 100
loss = 0.00148517548440553
```

In [ ]:
```
def plot_03():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f3, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss3)
```

In [ ]:
```
plot_03()
```




```
loss = 0.00148517548440553
```

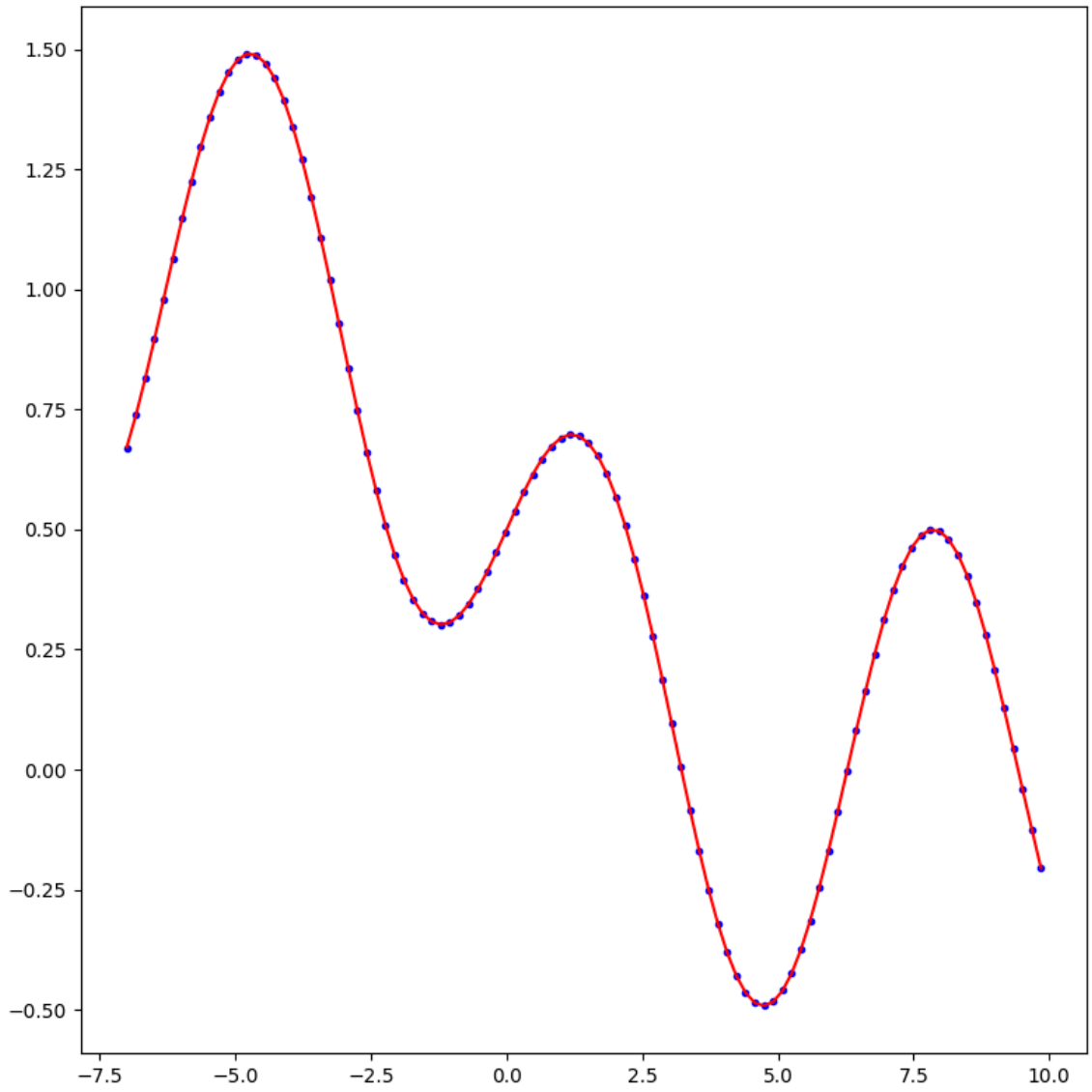
In [ ]:
```
p4          = 16
alpha4      = 0
A4          = util.get_matrix_A_regression_polynomial(x, p4)
f4, theta4  = util.compute_regression_polynomial(A4, y, alpha4)
loss4       = util.compute_loss_regression_polynomial(A4, y, theta
4, alpha4)
```

In [ ]:
```
print('size of A =', A4.shape)
print('size of f =', len(f4))
print('loss =', loss4)
```

```
size of A = (100, 17)
size of f = 100
loss = 2.590064888288907e-07
```

In [ ]:
```
def plot_04():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f4, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss4)
```

In [ ]:
```
plot_04()
```




```
loss = 2.590064888288907e-07
```

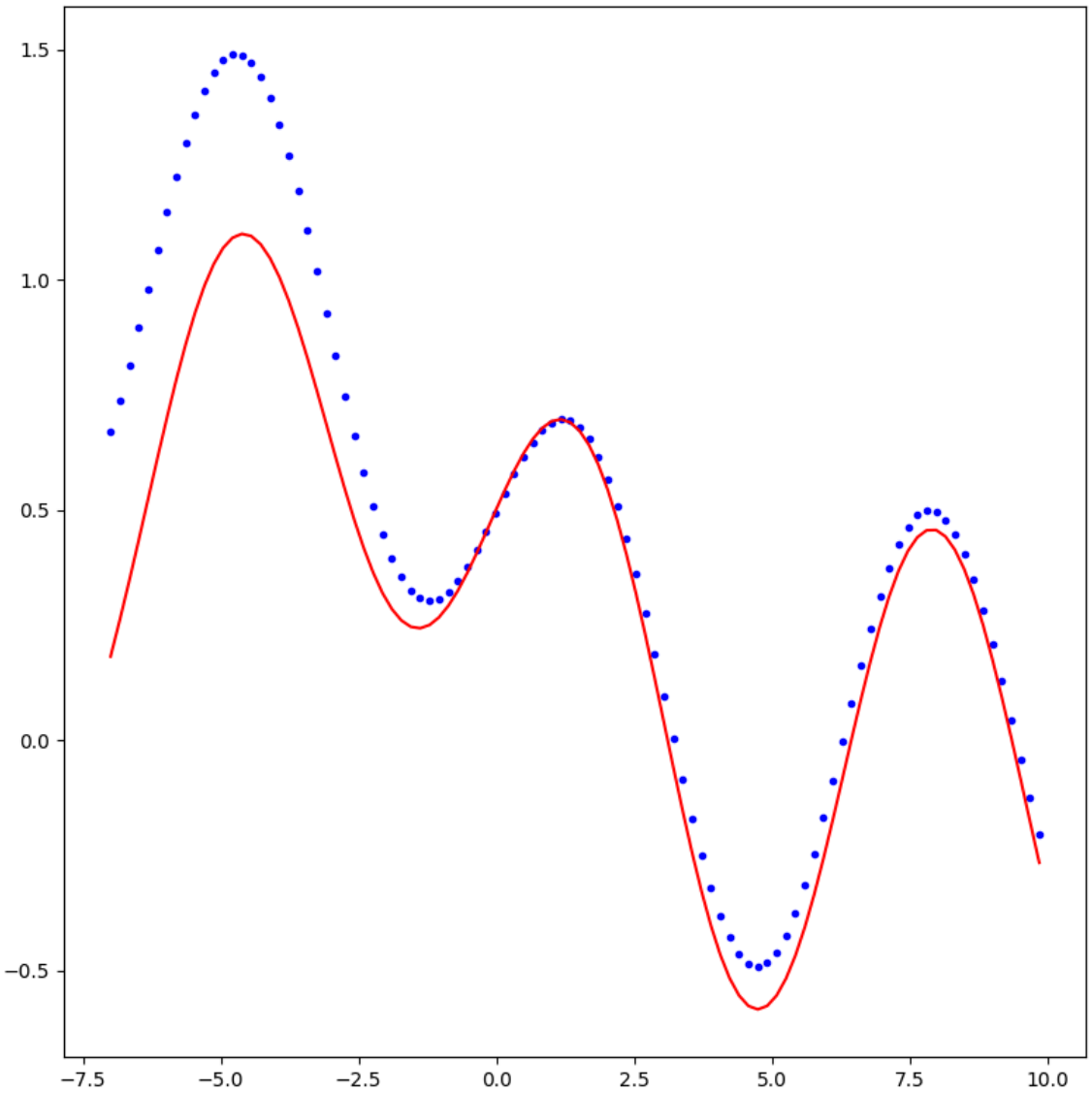
In [ ]:
```
p5          = 25
alpha5      = 0
A5          = util.get_matrix_A_regression_polynomial(x, p5)
f5, theta5  = util.compute_regression_polynomial(A5, y, alpha5)
loss5       = util.compute_loss_regression_polynomial(A5, y, theta
5, alpha5)
```

In [ ]:
```
print('size of A =', A5.shape)
print('size of f =', len(f5))
print('loss =', loss5)
```

```
size of A = (100, 26)
size of f = 100
loss = 0.021229768815313105
```

In [ ]:
```
def plot_05():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f5, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss5)
```

In [ ]:
```
plot_05()
```




```
loss = 0.021229768815313105
```

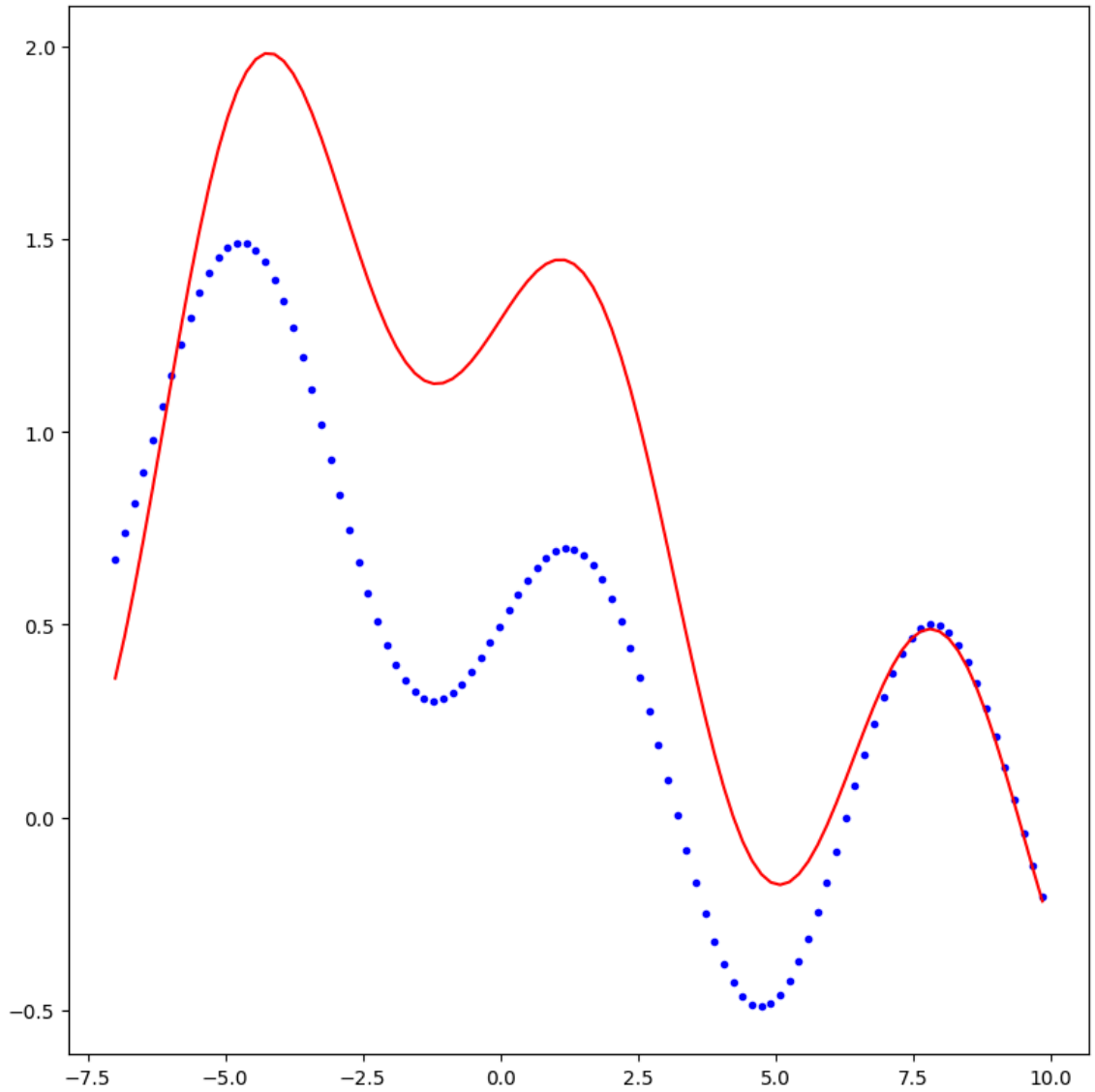
In [ ]:
```
p6          = 25
alpha6      = 0.01
A6          = util.get_matrix_A_regression_polynomial(x, p6)
f6, theta6  = util.compute_regression_polynomial(A6, y, alpha6)
loss6       = util.compute_loss_regression_polynomial(A6, y, theta
6, alpha6)
```

In [ ]:
```
print('size of A =', A6.shape)
print('size of f =', len(f6))
print('loss =', loss6)
```

```
size of A = (100, 26)
size of f = 100
loss = 0.15089854869114014
```

In [ ]:
```
def plot_06():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f6, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss6)
```

In [ ]:
```
plot_06()
```




```
loss = 0.15089854869114014
```

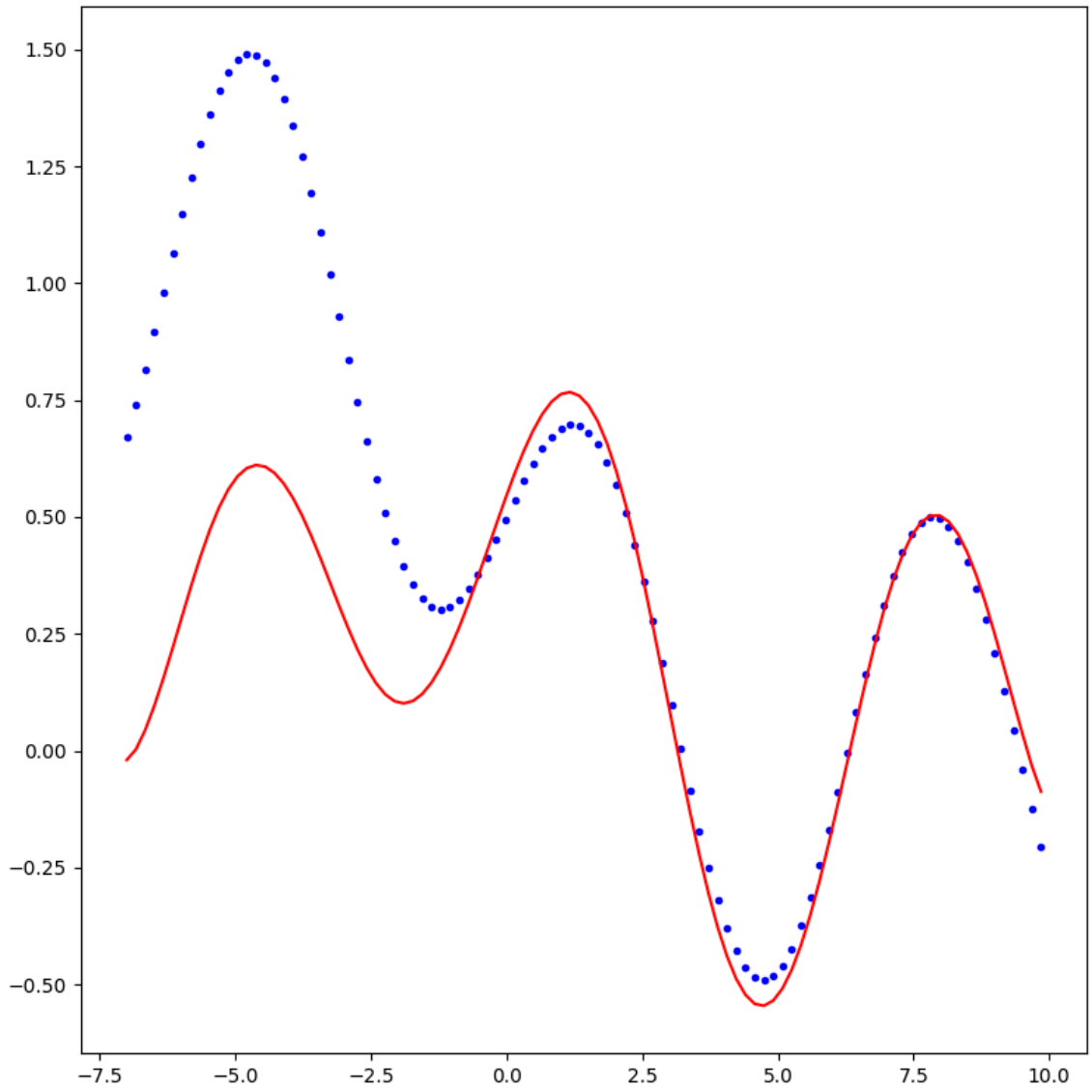
In [ ]:
```
p7          = 25
alpha7      = 0.1
A7          = util.get_matrix_A_regression_polynomial(x, p7)
f7, theta7  = util.compute_regression_polynomial(A7, y, alpha7)
loss7       = util.compute_loss_regression_polynomial(A7, y, theta
7, alpha7)
```

In [ ]:
```
print('size of A =', A7.shape)
print('size of f =', len(f7))
print('loss =', loss7)
```

```
size of A = (100, 26)
size of f = 100
loss = 0.10803262095665139
```

In [ ]:
```
def plot_07():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f7, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss7)
```

In [ ]:
```
plot_07()
```




```
loss = 0.10803262095665139
```

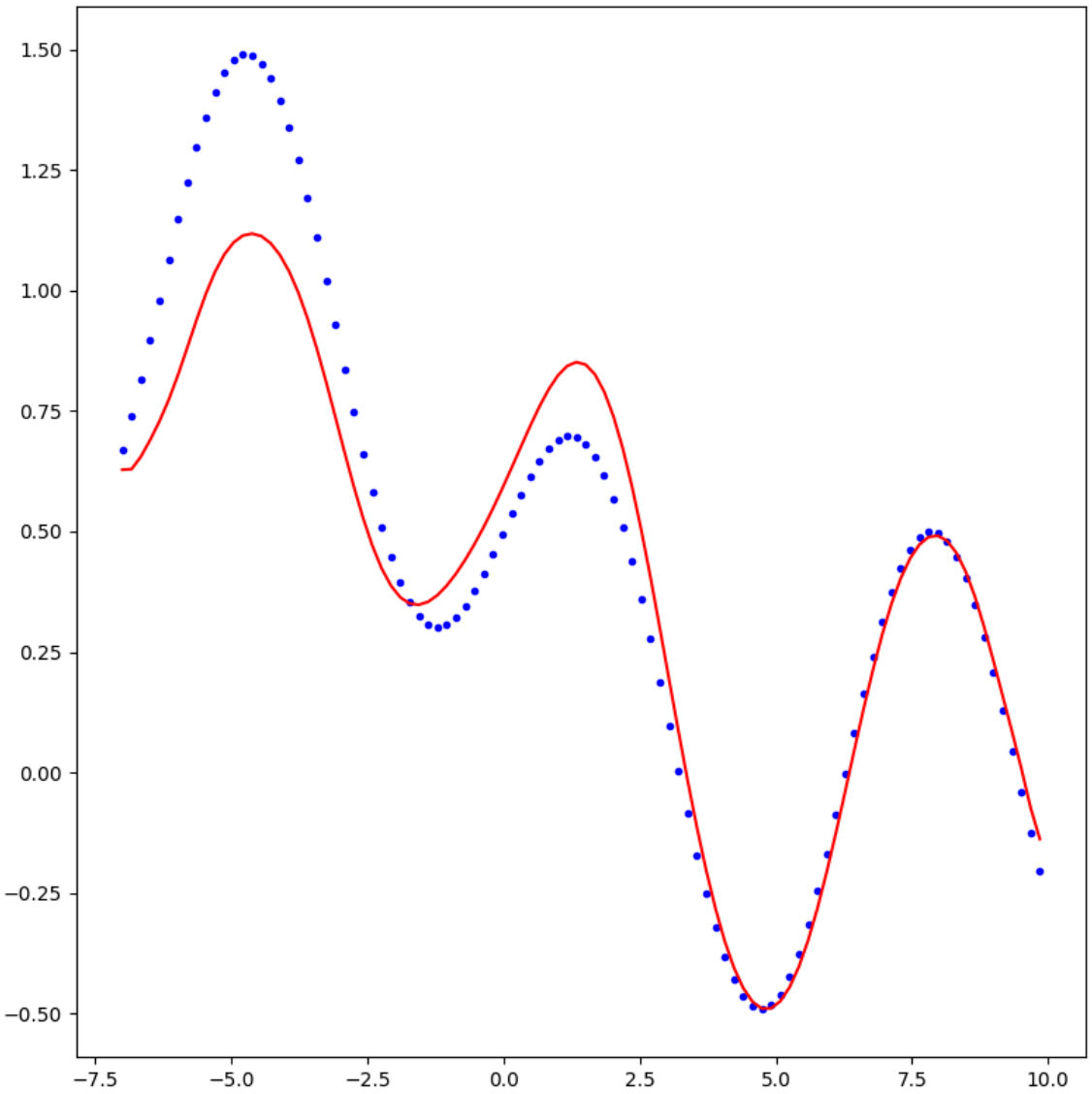
In [ ]:
```
p8          = 25
alpha8      = 1
A8          = util.get_matrix_A_regression_polynomial(x, p8)
f8, theta8  = util.compute_regression_polynomial(A8, y, alpha8)
loss8       = util.compute_loss_regression_polynomial(A8, y, theta
8, alpha8)
```

In [ ]:
```
print('size of A =', A8.shape)
print('size of f =', len(f8))
print('loss =', loss8)
```

```
size of A = (100, 26)
size of f = 100
loss = 0.22228097703870953
```

In [ ]:
```
def plot_08():
    plt.figure(figsize=(8,8))
    plt.plot(x, y, '.', color = 'blue')
    plt.plot(x, f8, '-', color = 'red')
    plt.tight_layout()
    plt.show()
    print('loss =', loss8)
```

In [ ]:
```
plot_08()
```

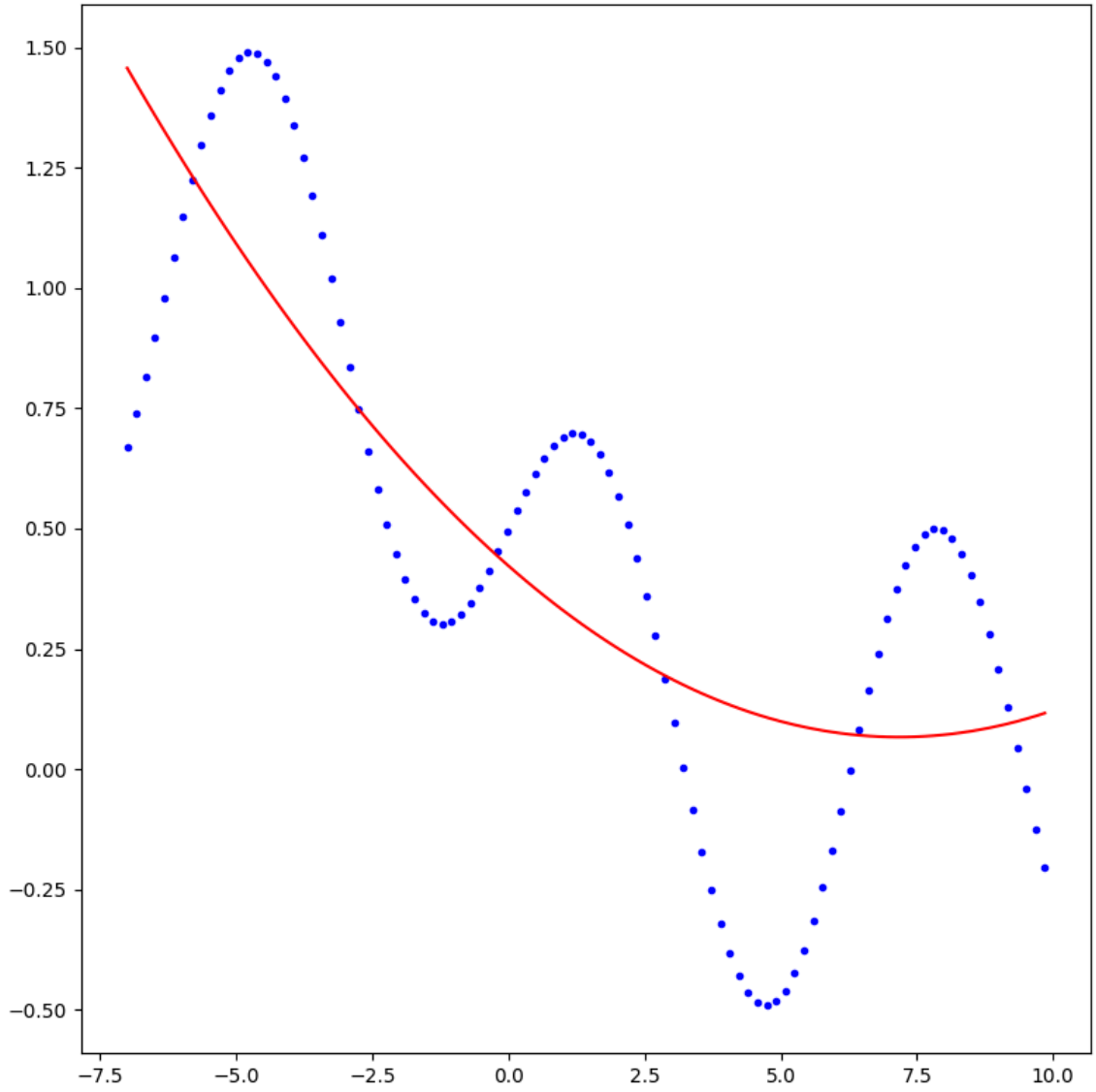



```
loss = 0.22228097703870953
```

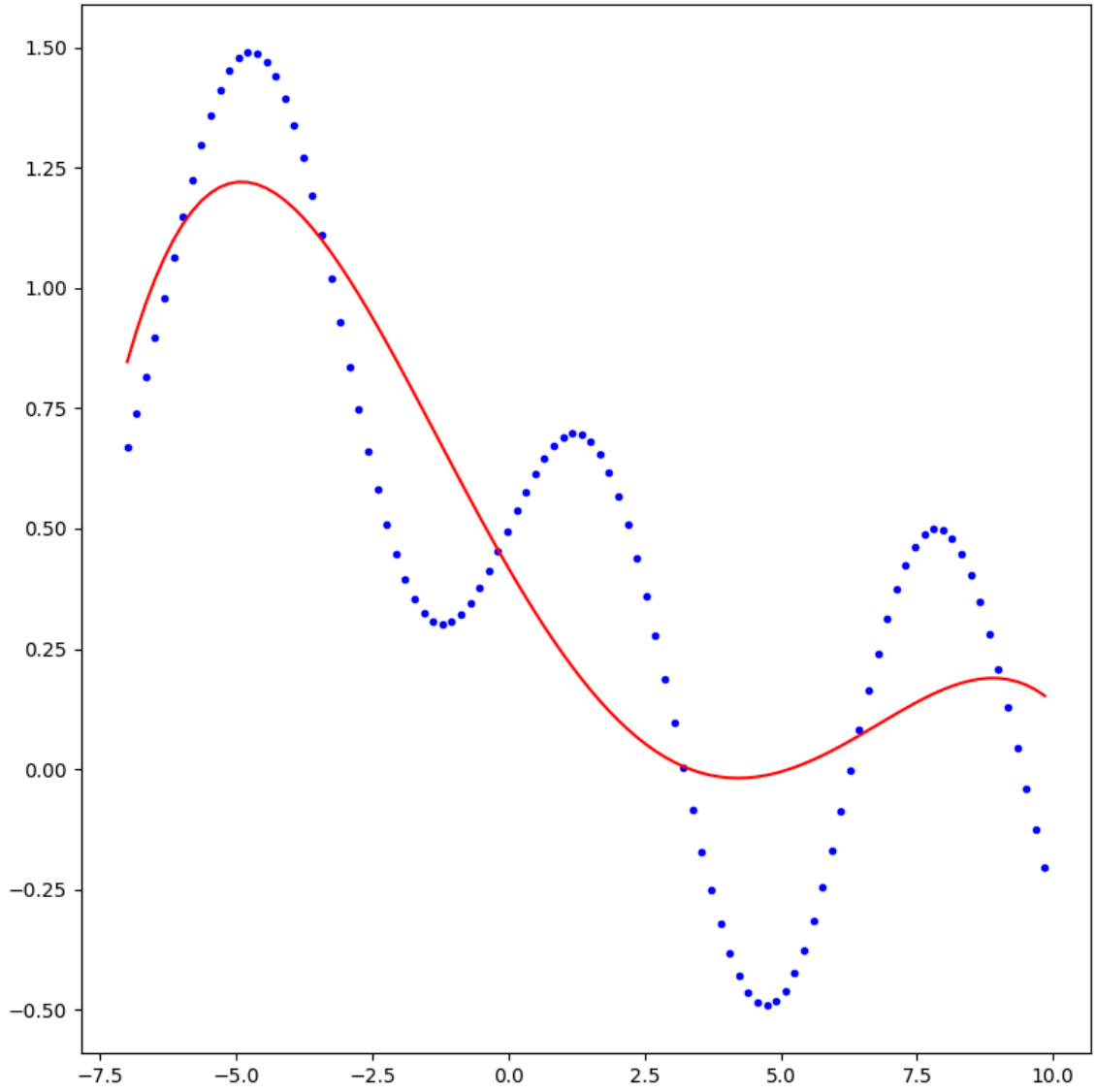
results

In [ ]:
```
number_result = 8

for i in range(number_result):
    title           = '# RESULT # {:02d}'.format(i+1)
    name_function   = 'plot_{:02d}()'.format(i+1)

    print('')
    print('##########################################################
########################')
    print(title)
    print('##########################################################
########################')
    print('')

    eval(name_function)
```

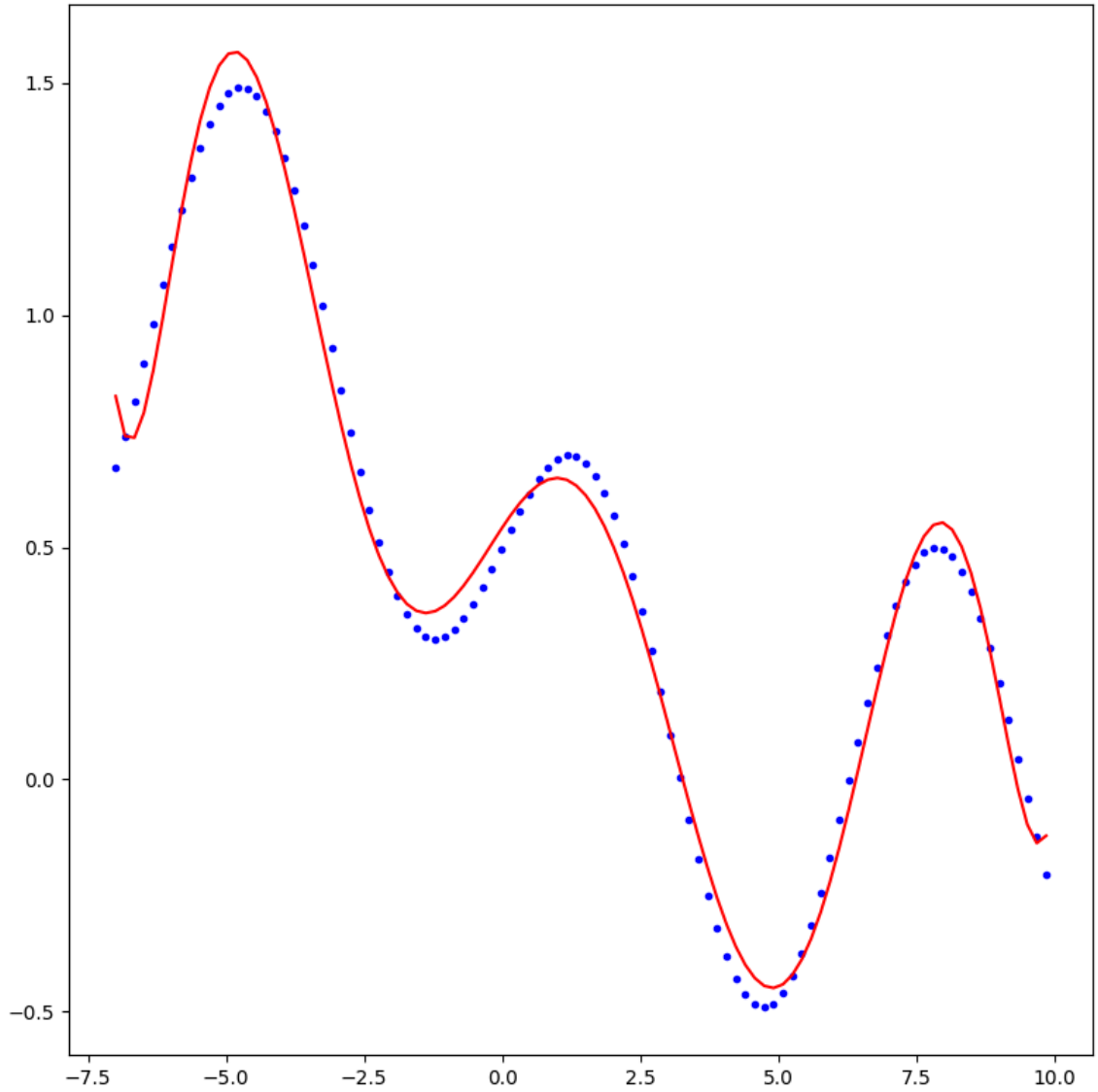
```
##########################################################################
#############
# RESULT # 01
##########################################################################
#############
```



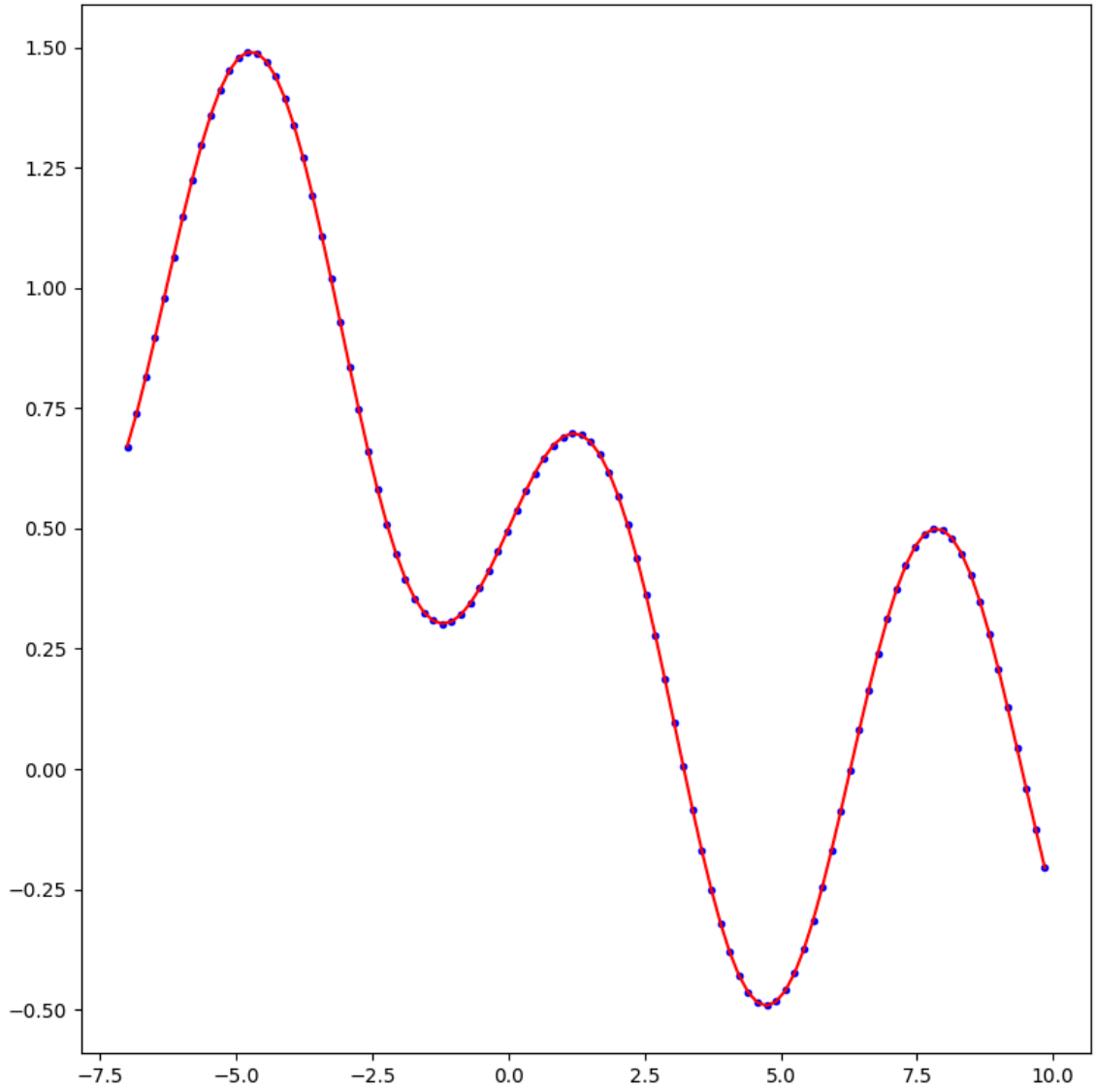

```
loss = 0.0558576756688038

##########################################################################
#############
# RESULT # 02
##########################################################################
#############
```



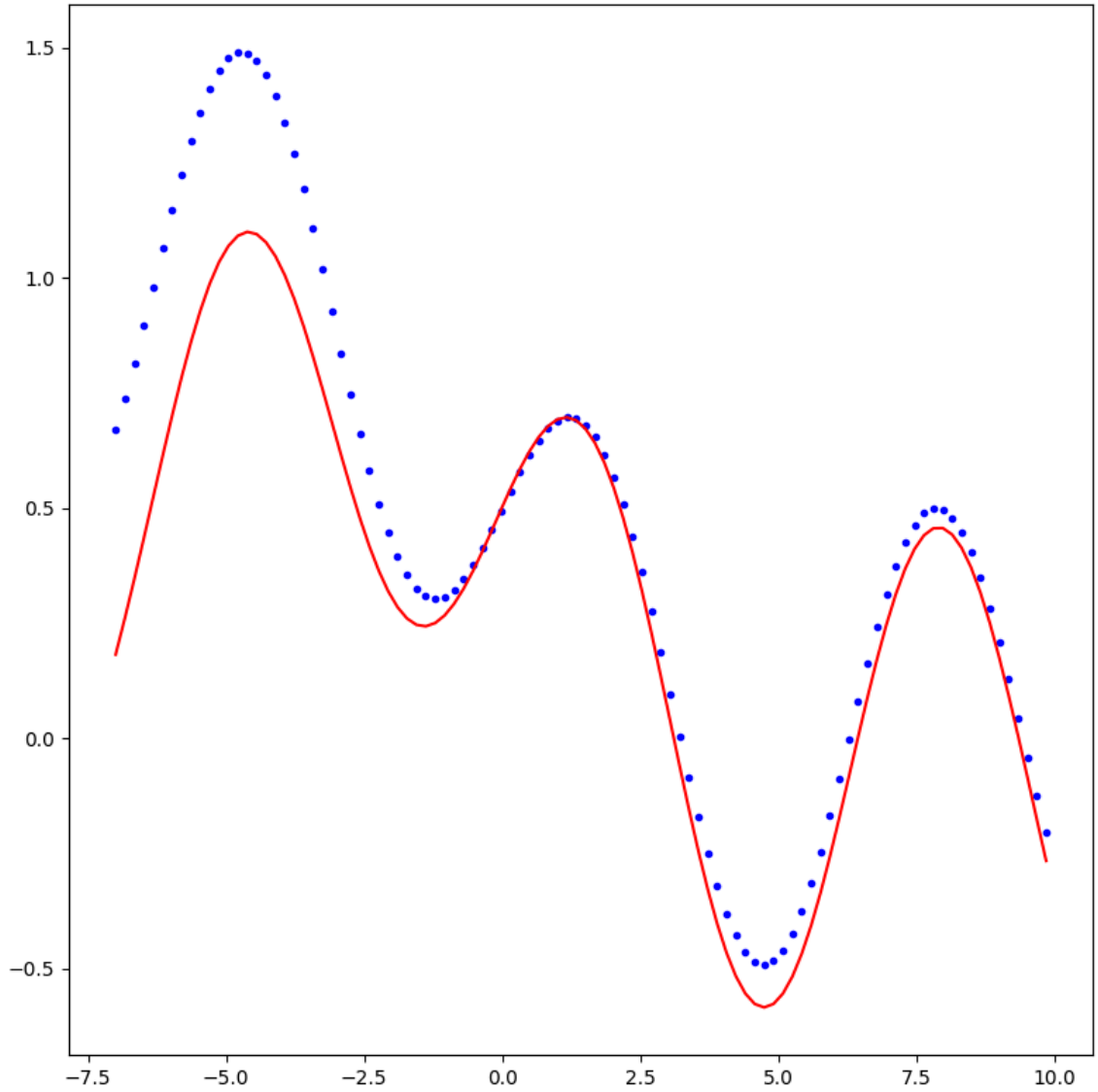

```
loss = 0.04184288075723241

#####################################################################
#############
# RESULT # 03
#####################################################################
#############
```



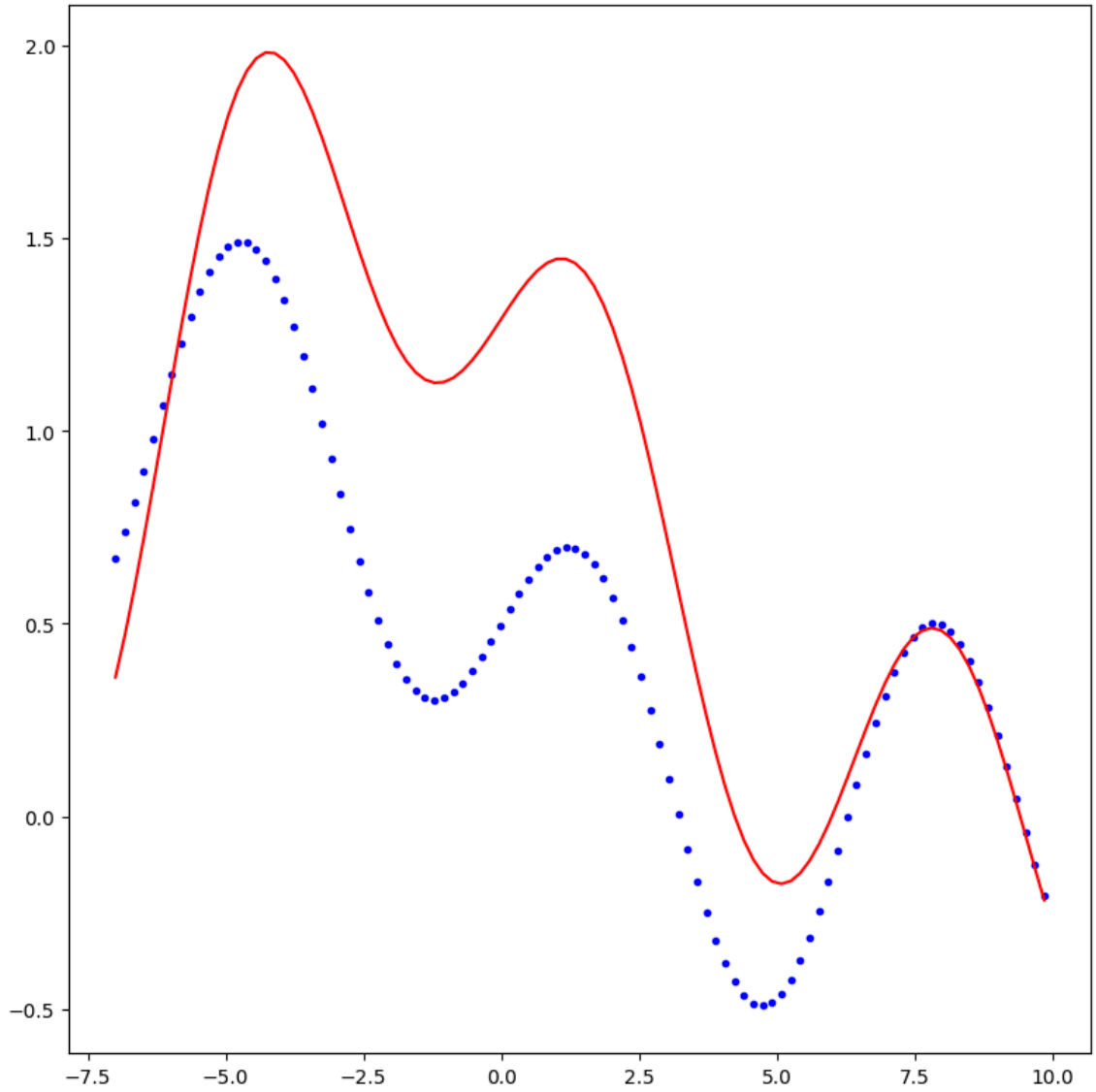

```
loss = 0.00148517548440553

##########################################################################
#############
# RESULT # 04
##########################################################################
#############
```



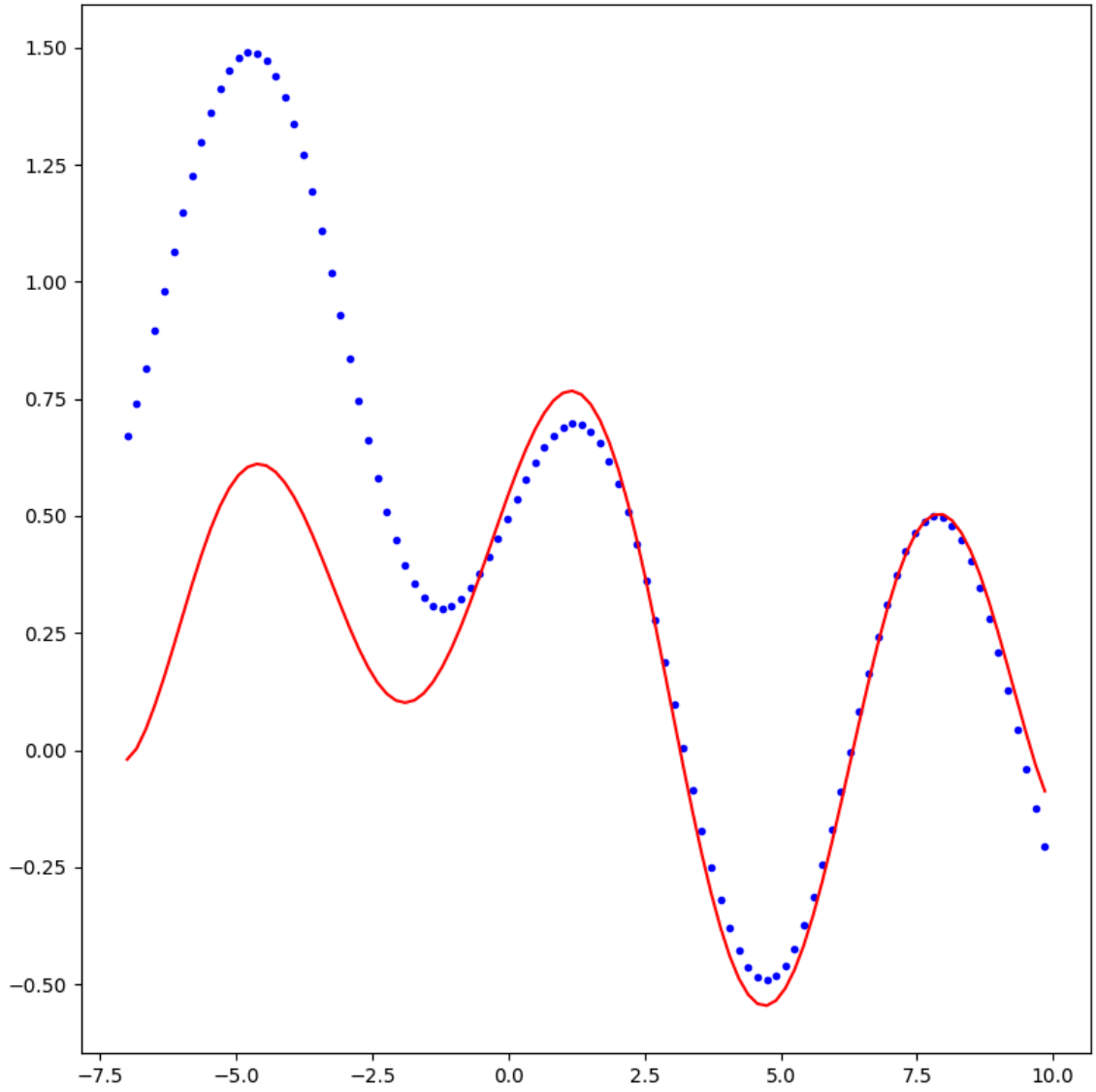

```
loss = 2.590064888288907e-07

##########################################################################
#############
# RESULT # 05
##########################################################################
#############
```



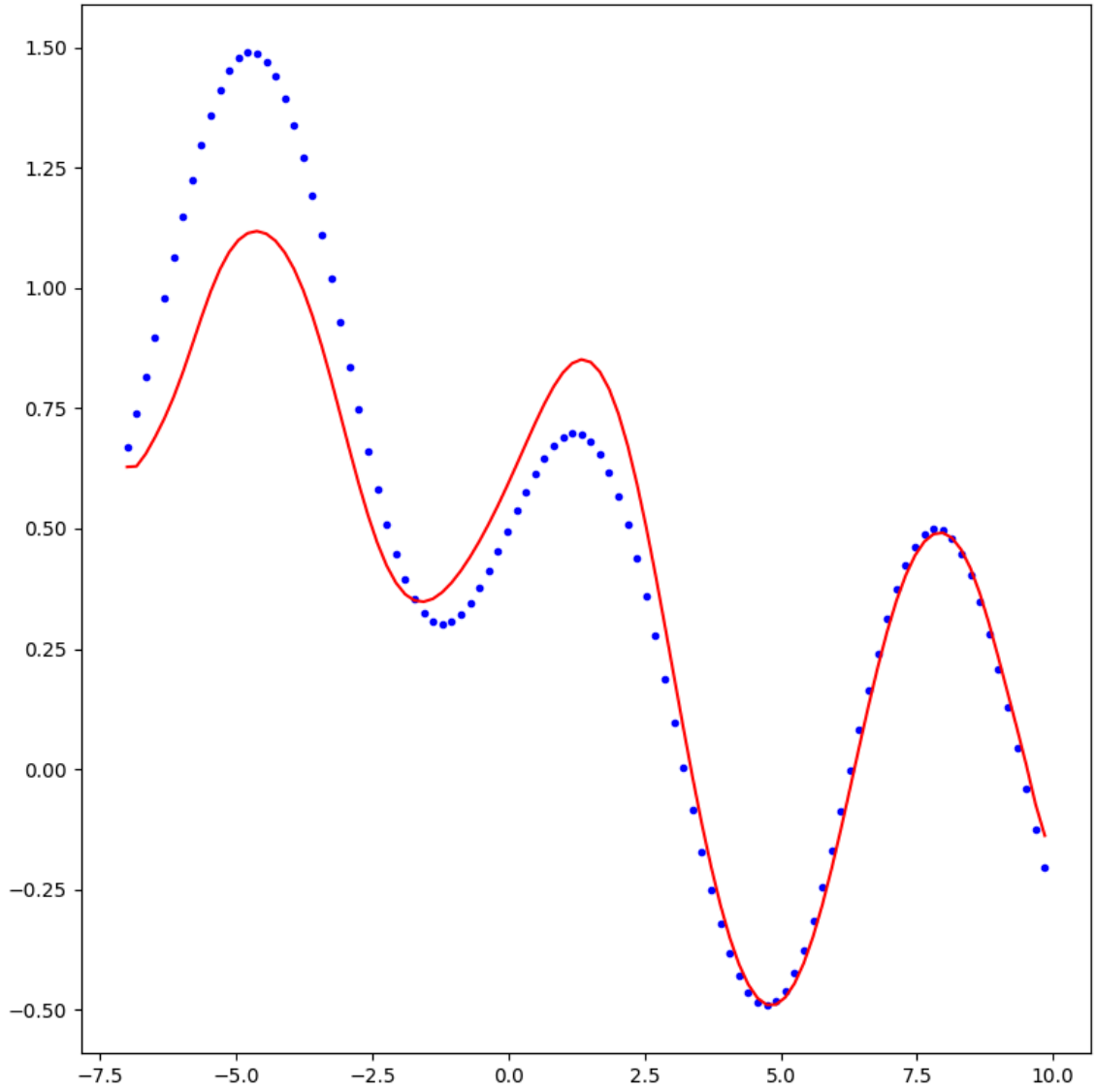

```
loss = 0.021229768815313105

##########################################################################
#############
# RESULT # 06
##########################################################################
#############
```




```
loss = 0.15089854869114014

##########################################################################
#############
# RESULT # 07
##########################################################################
#############
```




```
loss = 0.10803262095665139

##########################################################################
#############
# RESULT # 08
##########################################################################
#############
```




```
loss = 0.22228097703870953
```

In [ ]: