Practice – Week 06

Practice – 01

1. Write a function **my_ls_params(f, x, y)**, where **x** and **y** are arrays of the same size containing experimental data, and **f** is a list with each element a function object to a basis vector of the estimation function. The output argument, **beta**, should be an array of the parameters of the least squares regression for **x**, **y**, and **f**.

```
# Test case
x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))
```

```
beta = my_ls_params(func, x, y)
print(beta)
```

```
[[1.31459484]
 [1.05853804]]
```

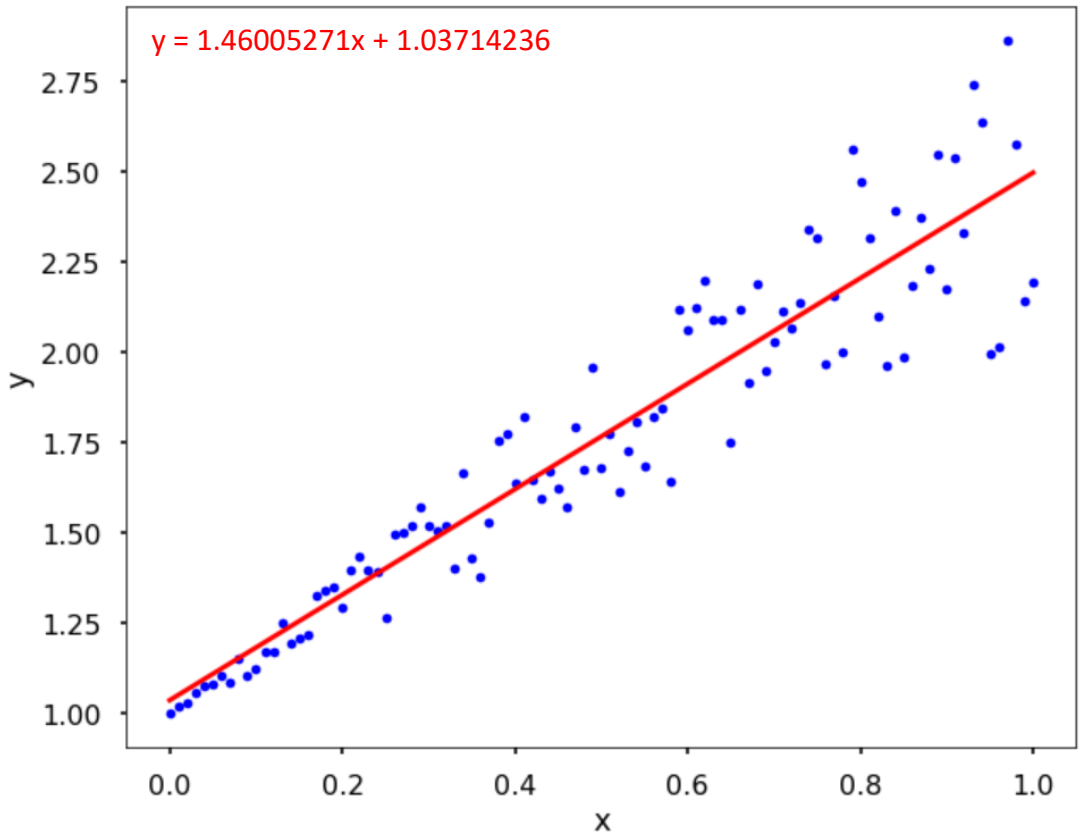
Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b):
    y = a*x + b
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*x + beta[1], 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b):
    y = a*x + b
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*x + beta[1], 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

```
[1.46005271 1.03714236]
```



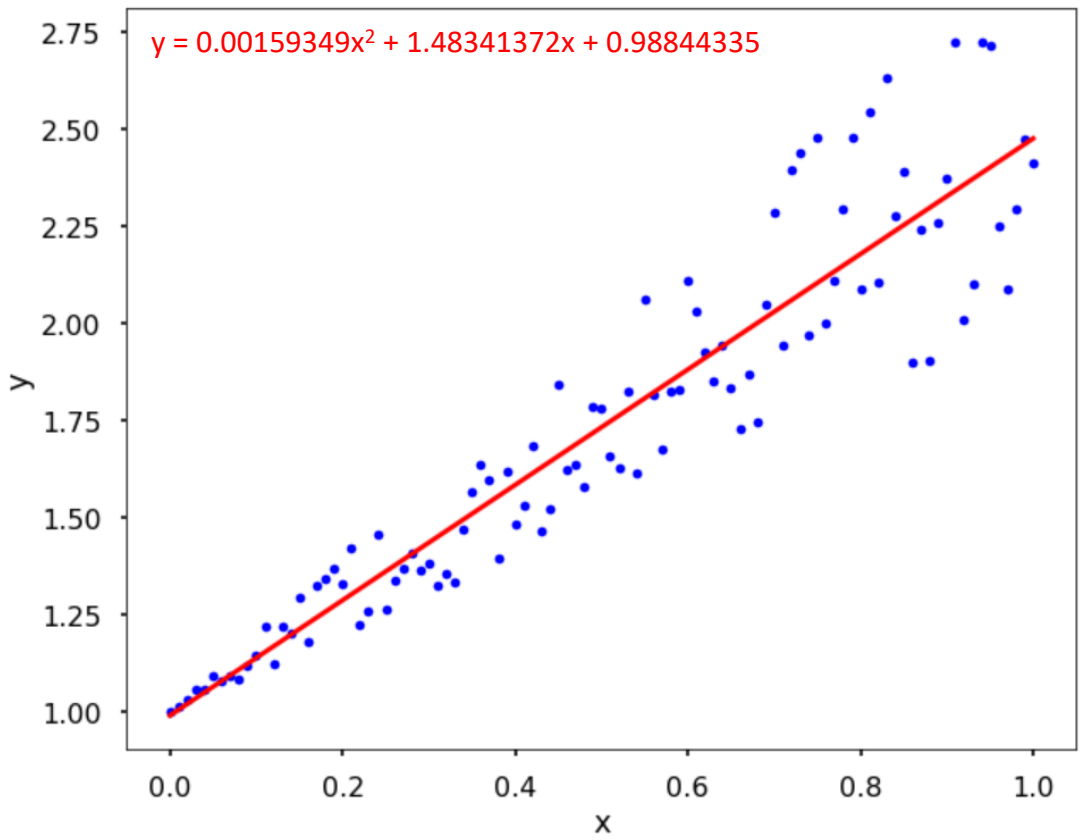
Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b, c):
    y = a*x*x + b*x + c
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*x*x + beta[1]*x + beta[2], 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b, c):
    y = a*x*x + b*x + c
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*x*x + beta[1]*x + beta[2], 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

```
[0.00159349 1.48341372 0.98844335]
```



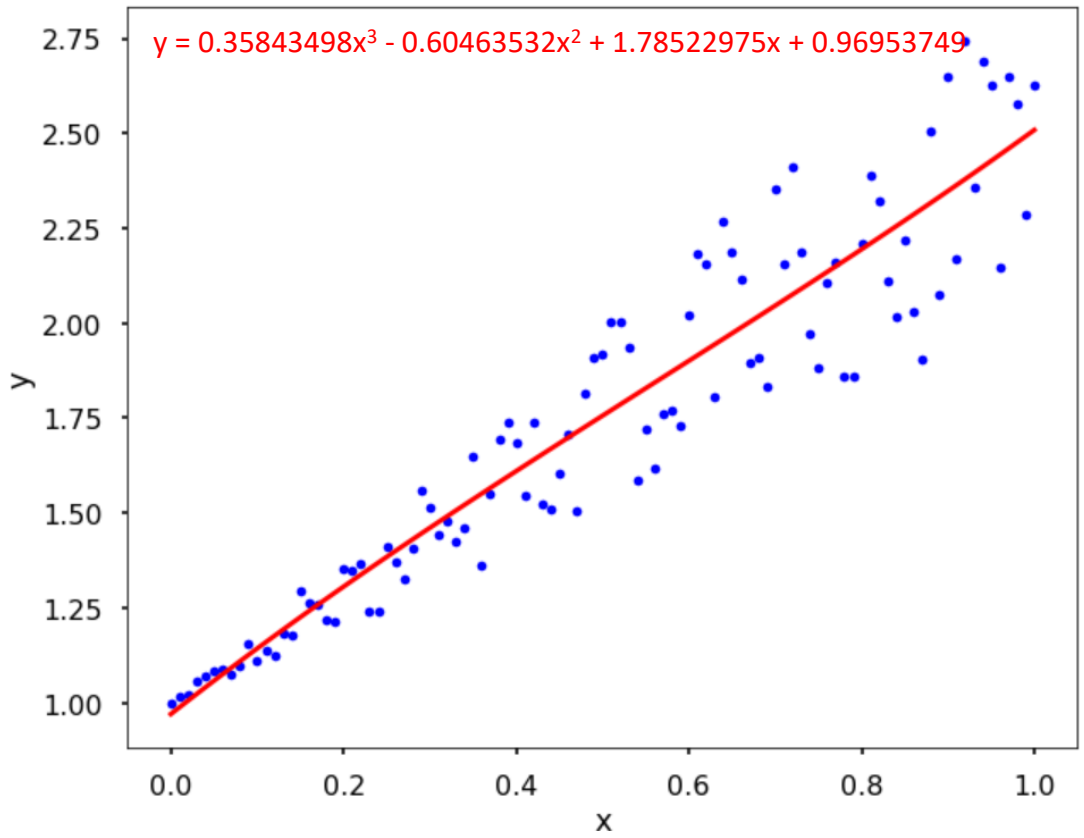
Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b, c, d):
    y = a*x*x*x + b*x*x + c*x + d
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*x*x*x + beta[1]*x*x + beta[2]*x + beta[3], 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b, c, d):
    y = a*x*x*x + b*x*x + c*x + d
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*x*x*x + beta[1]*x*x + beta[2]*x + beta[3], 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

```
[ 0.35843498 -0.60463532  1.78522975  0.96953749]
```



Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b):
    y = a*np.exp(b*x)
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*np.exp(beta[1]*x), 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

Practice – 01 – Answer

```
import numpy as np
from scipy import optimize
import matplotlib.pyplot as plt

plt.style.use('seaborn-poster')

def my_ls_params(f, x, y):
    return optimize.curve_fit(f, xdata = x, ydata = y)[0]

x = np.linspace(0, 1, 101)
y = 1 + x + x * np.random.random(len(x))

def f(x, a, b):
    y = a*np.exp(b*x)
    return y

beta = my_ls_params(f, x, y)
print(beta)

plt.figure(figsize = (10,8))
plt.plot(x, y, 'b.')
plt.plot(x, beta[0]*np.exp(beta[1]*x), 'r')
plt.xlabel('x')
plt.ylabel('y')
plt.show()
```

```
[1.11479849 0.83360932]
```