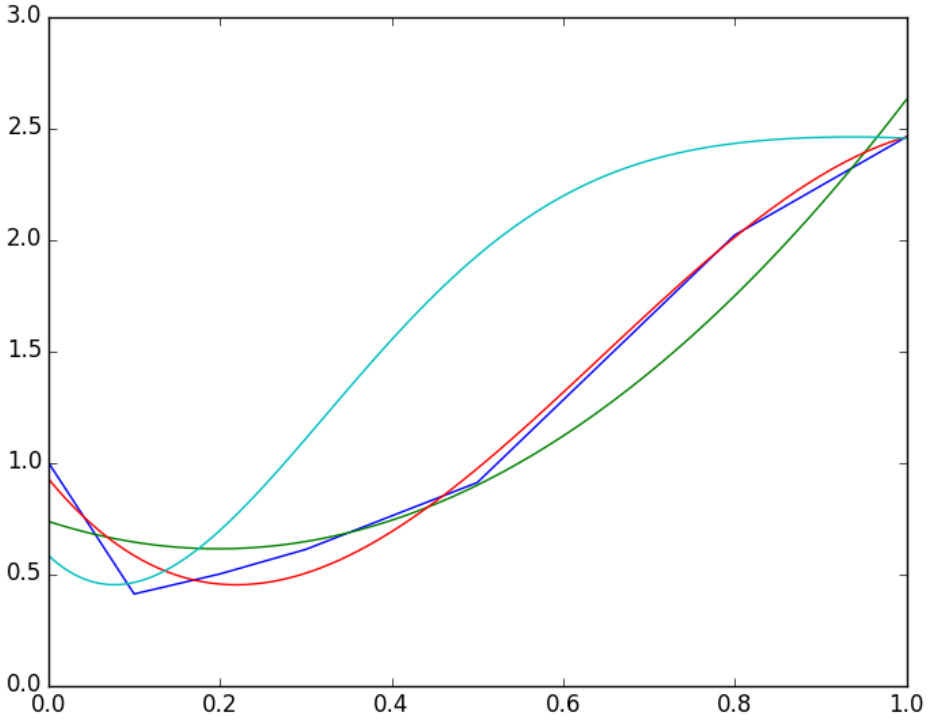
这是计算实习题 3.2 的代码和运行结果，其中代码需要在 Python3.4 环境中，加载 SciPy, Numpy, matplotlib, sympy 的条件下运行。

```
#Exercise 3.2
#15.10.10
#author: chuanlu

import scipy.linalg as lg
import numpy as np
import matplotlib.pyplot as plt
import sympy
from math import pi

sin = sympy.sin
cos = sympy.cos
x = sympy.Symbol('x')

def get_coefficient_matrix(x_values, n):
    matrix = np.zeros((n, n))
    temp_array = [0 for i in range(2*n)]
    for i in range(2*n):
        temp = 0
        for x in x_values:
            temp += x ** i
        temp_array[i] = temp

    for i in range(n):
        for j in range(n):
            matrix[i, j] = temp_array[i+j]
    return matrix

def get_constant_array(x_values, f_values, n):
    constant_array = [0 for i in range(n)]
    for i in range(n):
        temp = 0
        for j in range(len(x_values)):
            temp += f_values[j] * (x_values[j] ** i)
        constant_array[i] = temp
    return constant_array

def get_coefficient(x_values, f_values, n):
    coefficient_matrix = get_coefficient_matrix(x_values, n)
    constant_array = get_constant_array(x_values, f_values, n)
    return lg.solve(coefficient_matrix, constant_array)
```

```
def get_function(x_values, f_values, n):
    coefficient = get_coefficient(x_values, f_values, n)
    func = 0
    for i in range(len(coefficient)):
        func += coefficient[i] * (x ** i)
    return func

def get_value(func, value):
    return func.subs(x, value)

def plot(func):
    x_list = [0.01 * i for i in range(101)]
    y_list = [get_value(func, x) for x in x_list]
    plt.plot(x_list, y_list)


def main():
    x_values = [0.0, 0.1, 0.2, 0.3, 0.5, 0.8, 1.0]
    f_values = [1.0, 0.41, 0.50, 0.61, 0.91, 2.02, 2.46]
    n = 3
    func = get_function(x_values, f_values, n)
    plt.plot(x_values, f_values)
    print(func)
    plot(func)
    n = 4
    func = get_function(x_values, f_values, n)
    print(func)
    plot(func)
    func = get_sin_function(x_values, f_values, n)
    print(func)
    plot(func)
    plt.show()

def get_sin_coefficient_matrix(x_values):
    n = 2
    matrix = np.zeros((0, 0))
    for i in range(n):
        temp = 0
        for x in x_values:
            temp += sin(pi/2 * x) ** i
        temp_array = temp

    for i in range(n):
```

```
            for j in range(n):
                matrix[i, j] = temp_array[i+j]
        return matrix

def get_sin_constant_array(x_values, f_values, n):
        n = 2
        constant_array = [0 for i in range(n)]
        for i in range(n):
            temp = 0
            for j in range(len(x_values)):
                temp += f_values[j] * (sin(pi/2 * x_values[j]) ** i)
            constant_array[i] = temp
        return constant_array

def get_sin_coefficient(x_values, f_values, n):
        coefficient_matrix = get_sin_coefficient_matrix(x_values, n)
        constant_array = get_sin_constant_array(x_values, f_values, n)
        return lg.solve(coefficient_matrix, constant_array)

def get_sin_function(x_values, f_values, n):
        coefficient = get_coefficient(x_values, f_values, n)
        func = 0
        for i in range(len(coefficient)):
            func += coefficient[i] * (sin(pi/2 * x + 0.1) ** i)
        return func

if __name__ == '__main__':
        main()
```

运行结果如下：其中蓝色折线为数据点的连线， 绿色和红色的曲线为三次和四次多项式的最小二乘逼近，红色曲线为利用 sin(x)和 cos(x)的最小二乘逼近。


3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0