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。

