课程专业实践报告

实验题目	实验五 scipy 模块使用

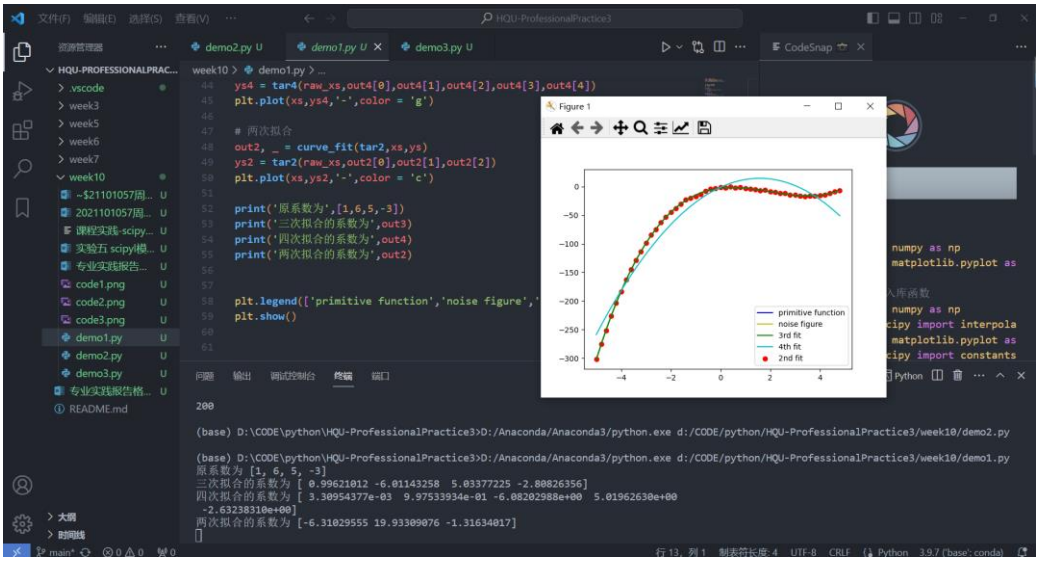
一．程序代码

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


# 设置随机数的种子
np.random.seed(10)

# 原函数
def fun(x):
    y = x**3 - 6*x**2 + 5*x - 3
    return y

# 3次拟合函数
def tar3(x,a,b,c,d):
    return a*x**3 +b*x**2 + c*x +d

# 4次拟合函数
def tar4(x,a,b,c,d,e):
    return a*x**4 +b*x**3 + c*x**2 +d*x +e

# 2次拟合函数
def tar2(x,a,b,c):
    return a*x**2 + b*x +c

# 原始数据
raw_xs = np.arange(-5,5,0.2)
raw_ys = fun(raw_xs)

plt.plot(raw_xs,raw_ys,color = 'b')

# 噪声数据
xs = raw_xs
ys = fun(raw_xs) + np.random.randn(50)
plt.scatter(xs,ys,color = 'r')

# 三次拟合
out3, _ = curve_fit(tar3,xs,ys)
ys3 = tar3(raw_xs,out3[0],out3[1],out3[2],out3[3])
plt.plot(xs,ys3,'-',color = 'y')

# 四次拟合
out4, _ = curve_fit(tar4,xs,ys)
ys4 = tar4(raw_xs,out4[0],out4[1],out4[2],out4[3],out4[4])
plt.plot(xs,ys4,'-',color = 'g')

# 两次拟合
out2, _ = curve_fit(tar2,xs,ys)
ys2 = tar2(raw_xs,out2[0],out2[1],out2[2])
plt.plot(xs,ys2,'-',color = 'c')

print('原系数为',[1,6,5,-3])
print('三次拟合的系数为',out3)
print('四次拟合的系数为',out4)
print('两次拟合的系数为',out2)


plt.legend(['primitive function','noise figure','3rd fit','4th fit','2nd fit'])
plt.show()
```

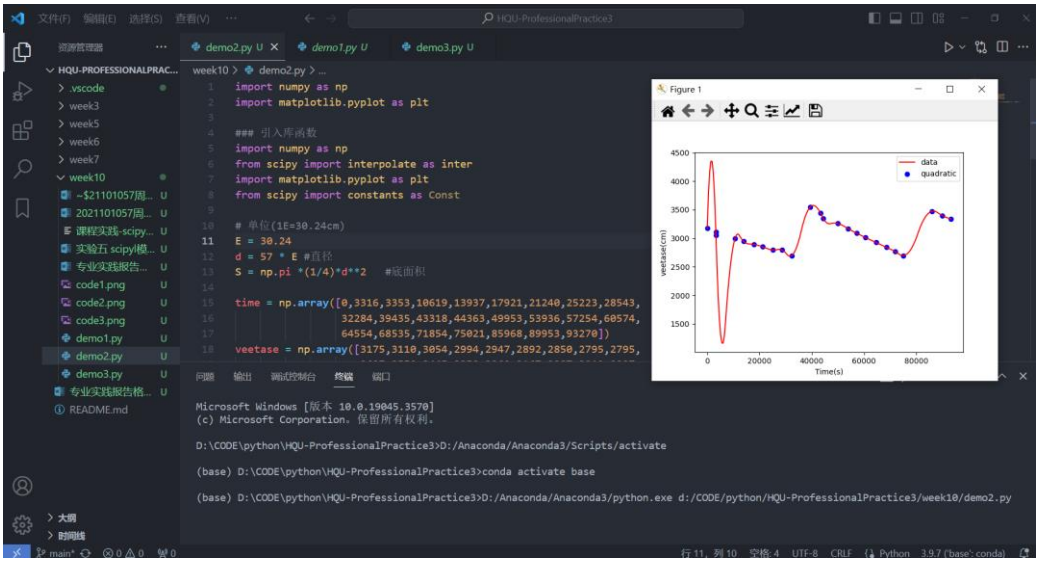
```python
import numpy as np
import matplotlib.pyplot as plt

### 引入库函数
import numpy as np
from scipy import interpolate as inter
import matplotlib.pyplot as plt
from scipy import constants as Const

# 单位(1E=30.24cm)
E = 30.24
d = 57 * E #直径
S = np.pi *(1/4)*d**2   #底面积

time = np.array([0,3316,3353,10619,13937,17921,21240,25223,28543,
                 32284,39435,43318,44363,49953,53936,57254,60574,
                 64554,68535,71854,75021,85968,89953,93270])
veetase = np.array([3175,3110,3054,2994,2947,2892,2850,2795,2795,
                    2697,3550,3445,3350,3260,3167,3087,3012,2927,
                    2842,2767,2697,3475,3397,3340])

f = inter.interp1d(time,veetase,kind ="quadratic",fill_value="extrapolate") #进行二次样条插值
xli = np.linspace(0,93270,200)
yli = f(xli)

plt.scatter(time,veetase,color = 'b')
plt.plot(xli,yli,'-',color = 'r')
plt.legend(['data','quadratic'], loc='best')
plt.xlabel('Time(s)')
plt.ylabel('veetase(cm)')
plt.show()

# 求导
def get_derivative(f, delta=0.001):
    def derivative(x):
        return (f(x+delta)-f(x))/delta
    return derivative

fd = get_derivative(f)
Flow_Rate = fd(yli * S)

plt.plot(xli,Flow_Rate,'-',color = 'b')
plt.legend(['predict'], loc='best')
plt.xlabel('Time(s)')
plt.ylabel('Flow Rate (cm^2)')
plt.show()
```

```python
from scipy.optimize import minimize
import numpy as np

def fun(xs):
    f = np.sin(xs[0]) + 0.05 * xs[0]**2 +np.sin(xs[1]) + 0.05 * xs[1]**2
    return f

xs = np.arange(2)
res = minimize(fun, xs, method='TNC')
print('最小值为',res.fun)
```

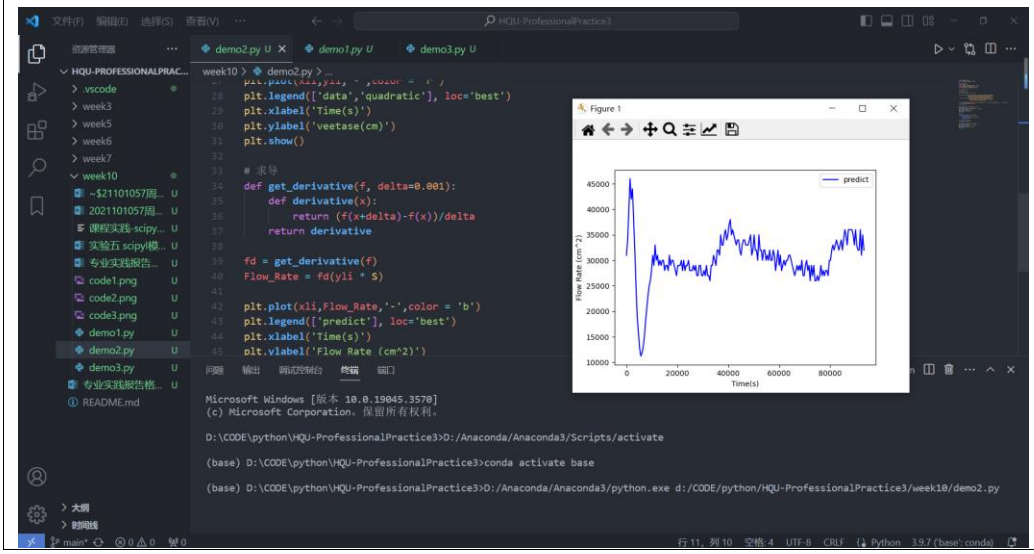
二．运行结果

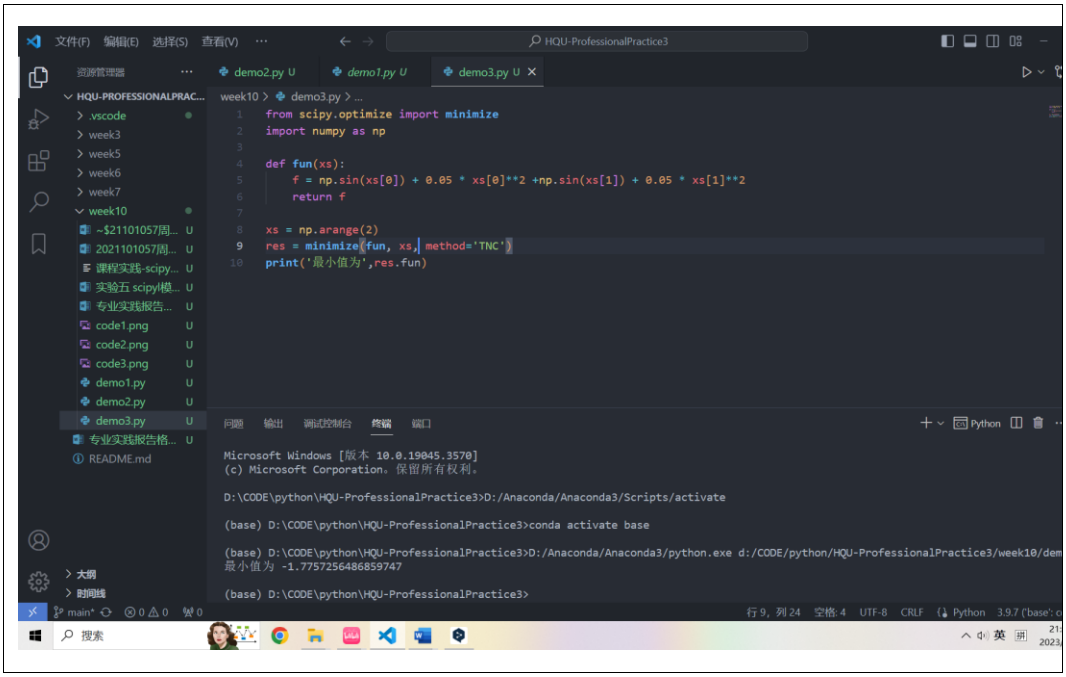


水位线预测：



流量预测：




demo2.py U
demo1.py U
demo3.py U
week10 > demo3.py > ...
from scipy.optimize import minimize
import numpy as np
def fun(xs):
f = np.sin(xs[0]) + 0.05 * xs[0]**2 +np.sin(xs[1]) + 0.05 * xs[1]**2
return f
xs = np.arange(2)
res = minimize(fun, xs, method='TNC')
print('最小值为',res.fun)
Microsoft Windows [版本 10.0.19045.3570]
(c) Microsoft Corporation。保留所有权利。
D:\CODE\python\HQU-ProfessionalPractice3>D:/Anaconda/Anaconda3/Scripts/activate
(base) D:\CODE\python\HQU-ProfessionalPractice3>conda activate base
(base) D:\CODE\python\HQU-ProfessionalPractice3>D:/Anaconda/Anaconda3/python.exe d:/CODE/python/HQU-ProfessionalPractice3/week10/dem
最小值为 -1.7757256486859747
(base) D:\CODE\python\HQU-ProfessionalPractice3>