**深度学习方法与实践第一次作业**

姓名：杨玉雷

学号：18023040

**1.问题描述**

设计python程序，首先安装并导入opencv库：  
例如：conda install opencv  
import cv2  
  
然后使用cv2.imread()读取任意彩色图片为numpy矩阵，然后进行以下操作：  
(1) 将图片的三个通道顺序进行改变，由RGB变为BRG，并用imshow()或者matplotlib中的有关函数显示图片  
(2) 利用Numpy给改变通道顺序的图片中指定位置打上红框，其中红框左上角和右下角坐标定义方式为：假设学号为12069028，则左上角坐标为(12, 06), 右下角坐标为(12+90, 06+28).  (不可使用opencv中自带的画框工具）  
(3) 利用cv2.imwrite()函数保存加上红框的图片。

**2.实验结果**

（1）实验代码如下：

import cv2 as cv

import matplotlib.pyplot as plt

image\_arr = cv.imread('test01.jpg')

b,g,r=cv.split(image\_arr) #cv读取的时候按bgr的顺序排列

img2=cv.merge([b,r,g]) #改变通道为brg顺序排列

cv.imshow('input\_image', img2)

cv.waitKey(0)

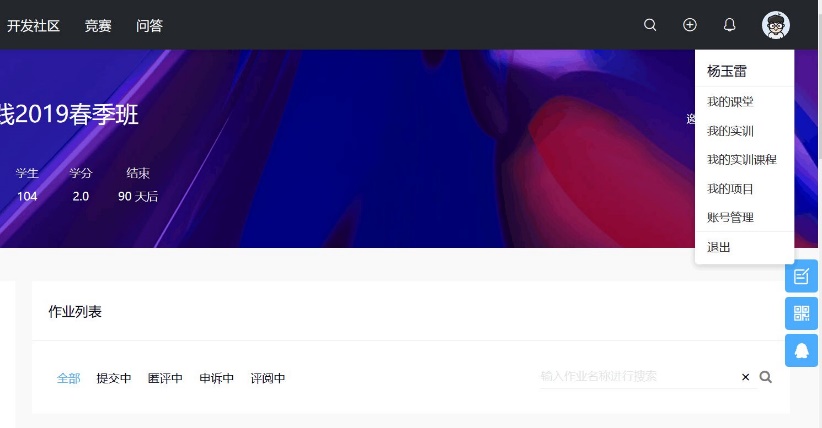
cv.destroyAllWindows() #删除建立的全部窗口，释放资源

plt.figure('brg')

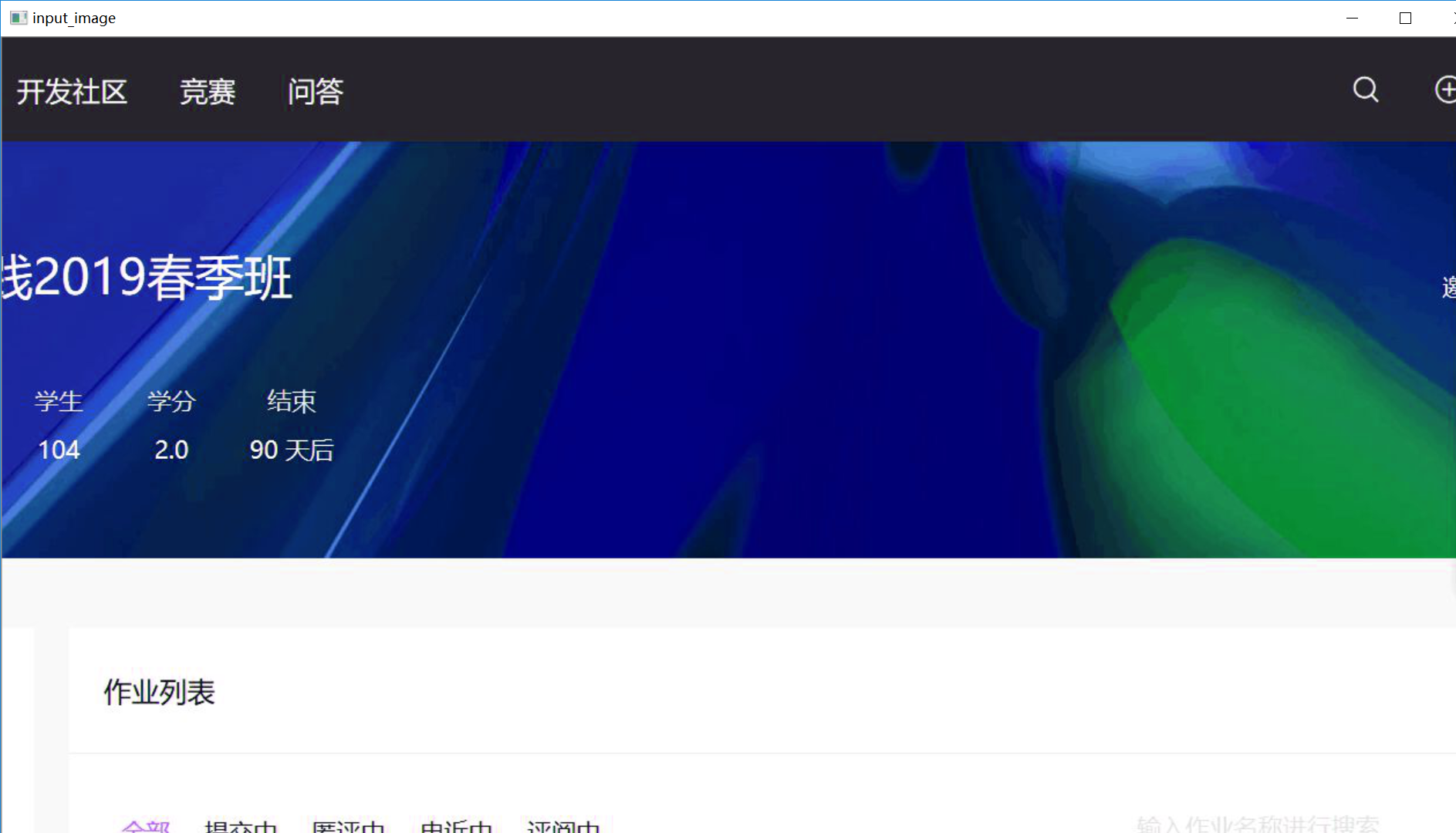
plt.imshow(img2)

plt.show()

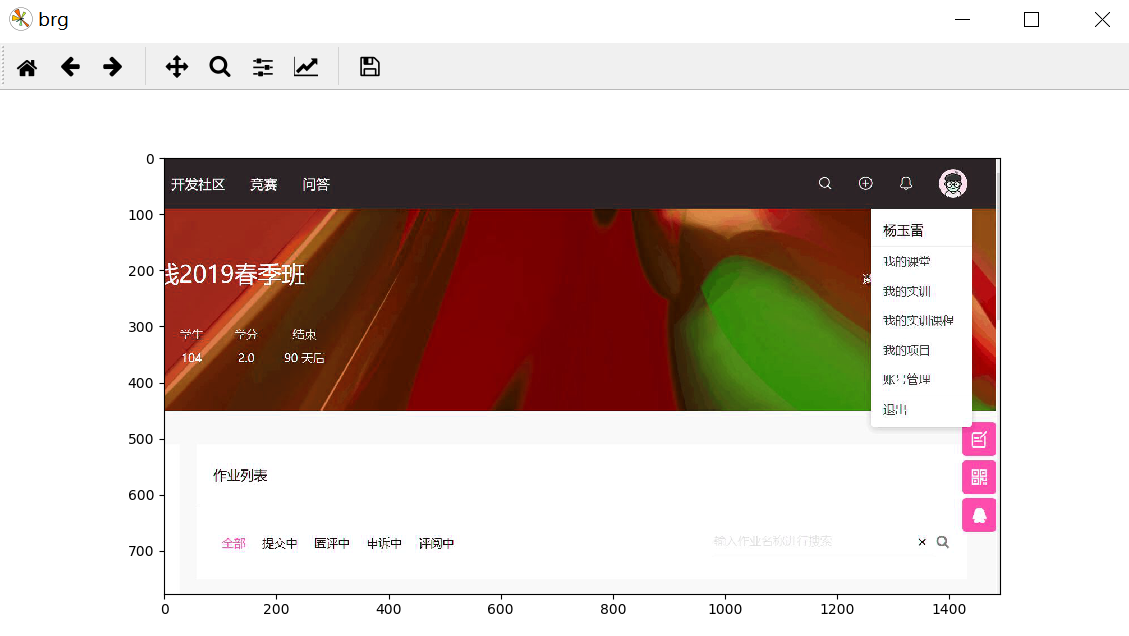
由于cv和matplotlib读取图片后的分别按照bgr和rgb不同的顺序排序，因此显示效果也不同：



2.1 测试原图



2.2 改变通道后按cv方式显示



2.3 改变通道后按matplotlib方式显示

（2）本人学号为：18023040，故左上角坐标为（18,02），右下角坐标为（18+30,02+40），即（48,42）。要使得在此区间的颜色全为红色，代码如下：

import cv2 as cv

image\_arr = cv.imread('test01.jpg')

b,g,r=cv.split(image\_arr) #cv读取的时候按bgr的顺序排列

print(b.shape)

#红色矩形区域

for i in range(18,49):

for j in range(2,43):

r[i][j]=255

g[i][j]=0

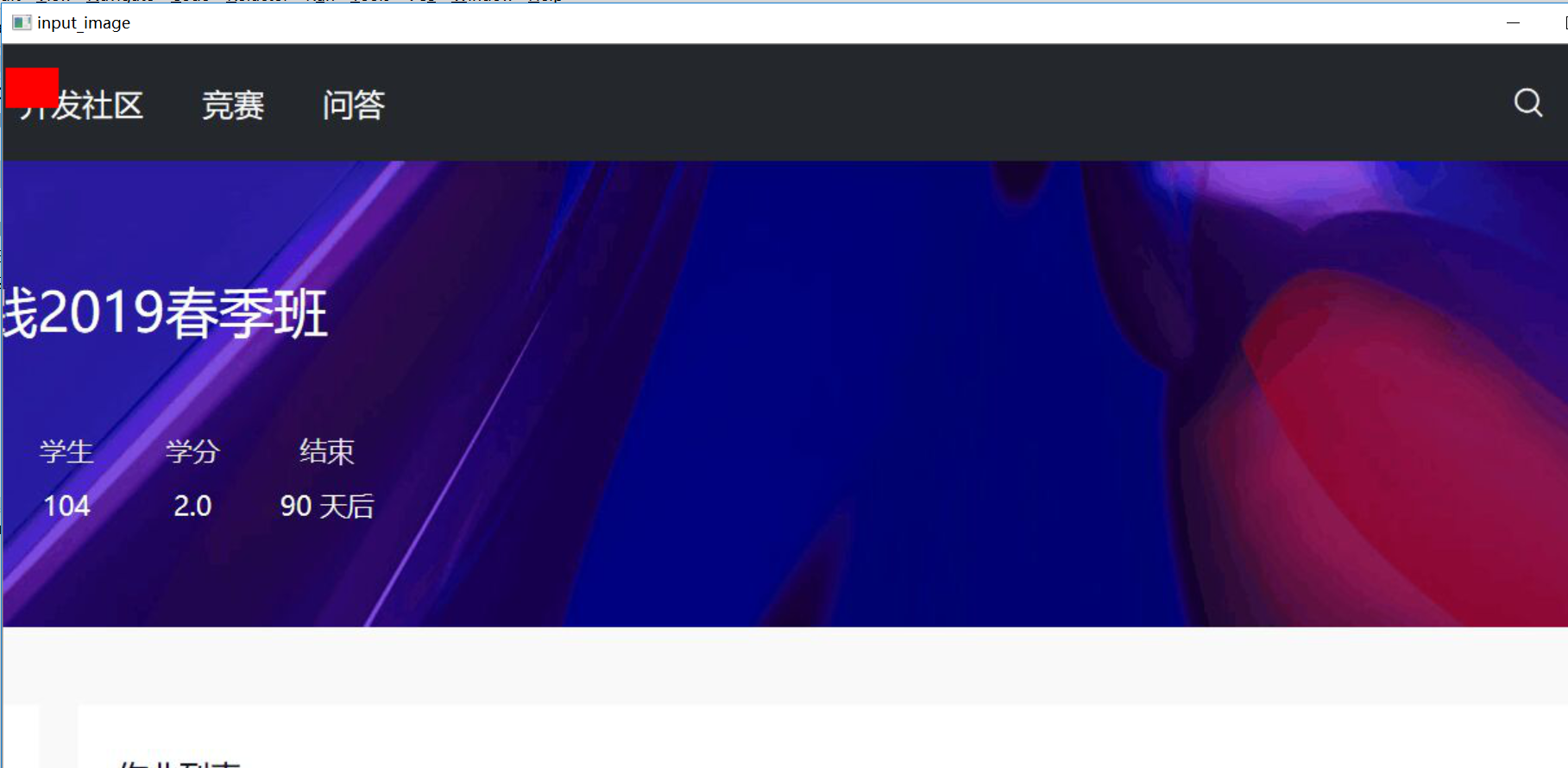
b[i][j]=0

img2=cv.merge([b,g,r]) #按brg顺序排列

cv.imshow('input\_image', img2) #按cv方式显示改变通道后的图片

cv.waitKey(0)

cv.destroyAllWindows()#删除建立的全部窗口，释放资源



2.4 输出红色矩形区域

要使得矩形区域为3个像素的红色框，可改变换颜色的代码为：

#红色矩形框,边框3个像素

for i in range(18,49):

for j in range(2,6):

r[i][j]=255

g[i][j]=0

b[i][j]=0

for j in range(39,43):

r[i][j]=255

g[i][j]=0

b[i][j]=0

for i in range(18,21):

for j in range(2,43):

r[i][j]=255

g[i][j]=0

b[i][j]=0

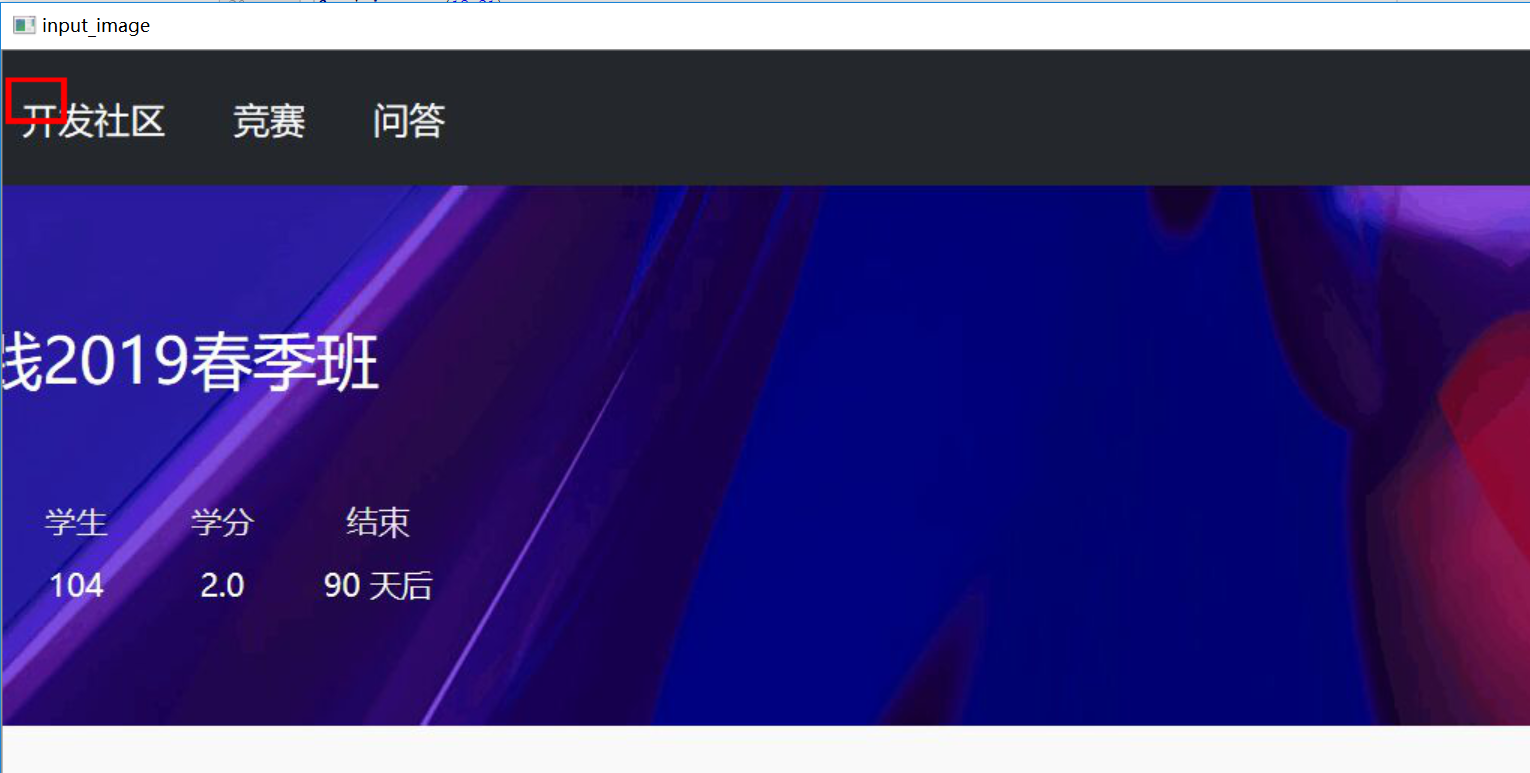
for i in range(45,49):

for j in range(2,43):

r[i][j]=255

g[i][j]=0

b[i][j]=0



2.5 输出3个像素的红色矩形边框

（3）保存代码如下：

#保存图片在当前目录下

cv.imwrite("output.jpg",img2)



2.6 cv保存图像结果

**3.进阶题**

假设有函数y = cos(ax + b), 其中a为学号前两位，b为学号最后两位。首先从此函数中以相同步长（点与点之间在x轴上距离相同），在0<(ax+b)<2pi范围内，采样出2000个点，然后利用采样的2000个点作为特征点进行三次函数拟合。请提交拟合的三次函数以及对应的图样（包括采样点及函数曲线）。

在此实验中，本人的a=18，b=40。故由0<(ax+b)<2pi得-b/a<x<(2pi-b)/a

由此可得步长计算为：step=((2\*math.pi-b)/a+b/a)/2000

设三次函数为y=a\*x\*x\*x+b\*x\*x+c\*x+d,则可根据样本点进行非线性最小二乘法拟合函数图像。

实验代码如下：

import math

import numpy as np

import matplotlib.pyplot as plt

from scipy.optimize import curve\_fit

#自定义3次函数形式

def func(x, a, b,c,d):

return a\*x\*x\*x+b\*x\*x+c\*x+d

a=18 #学号前两位

b=40 #学号后两位

step=((2\*math.pi-b)/a+b/a)/2000 #计算步长

x=np.arange(2001.)

y=np.arange(2001.)

for i in range(0,2001):

x[i]=-b/a+i\*step

y[i]=math.cos(a\*x[i]+b)

print("第",i,"个采样点：(",x[i],",",y[i],")")

plt.scatter(x,y)

#显示样本点所得到的图

plt.show()

#非线性最小二乘法拟合

popt, pcov = curve\_fit(func, x, y)

#获取popt里面是拟合系数

print(popt)

a = popt[0]

b = popt[1]

c = popt[2]

d = popt[3]

yvals = func(x,a,b,c,d) #拟合y值

print('popt:', popt)

print('系数a:', a)

print('系数b:', b)

print('系数c:', c)

print('系数d:', d)

print('系数pcov:', pcov)

print('系数yvals:', yvals)

#绘图

plot1 = plt.plot(x, y, 's',label='original values')

plot2 = plt.plot(x, yvals, 'r',label='polyfit values')

plt.xlabel('x')

plt.ylabel('y')

plt.legend(loc=4) #指定legend的位置右下角

plt.title('curve\_fit')

plt.show()

结果如下：

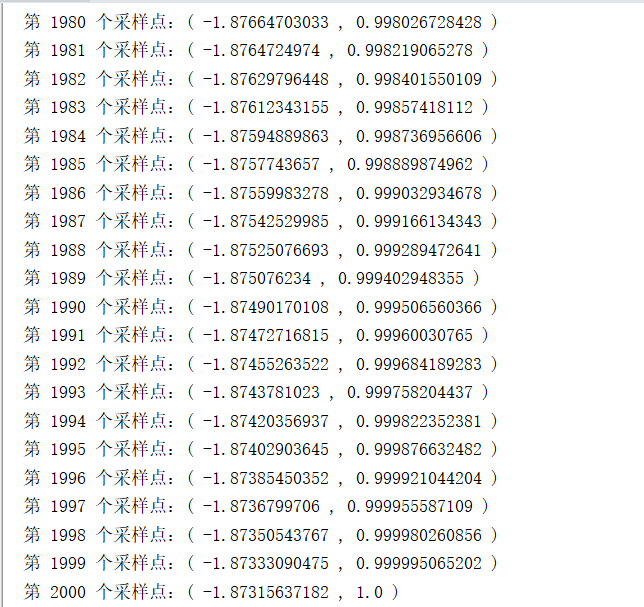
系数a: -0.000442334453412

系数b: 74.7723139817

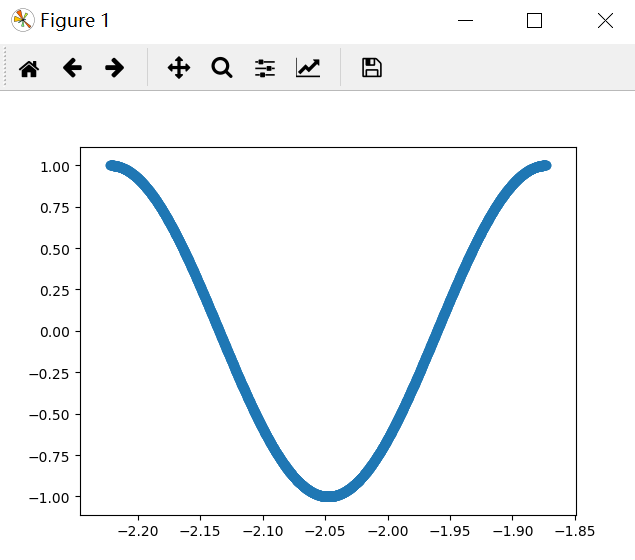
系数c: 306.226510224

系数d: 312.770766287

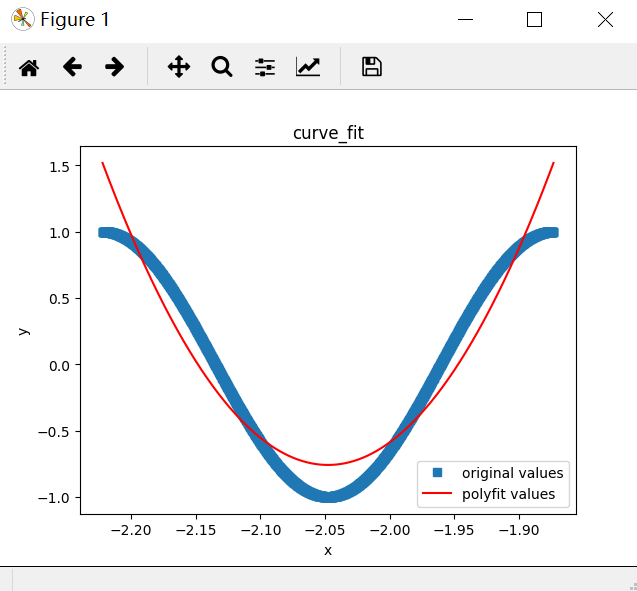
故带入上述系数即可得到三次函数公式：y=a\*x^3+b\*x^2+c\*x+d。



2.7 2000个样本点



2.8 样本点原始图像



2.9 由样本点拟合的三次函数图像（红色部分）