**实验一 最小二乘法**

最小二乘法是一种数学优化方法，通过最小化误差的平方和寻找最佳的匹配函数。

**实验要求：**

使用最小二乘法对样本点进行拟合，实验只要求使用**直线**来拟合样本点。

①学会使用封装好的最小二乘法函数

②自己使用Python语言实现最小二乘法函数来拟合直线(可选）

**实验过程：**

1、使用封装好的最小二乘法函数

①最小二乘法函数封装在Scipy模块当中，使用前需要导入，导入代码为：  
 *from scipy.optimize import leasteq*

②样本点数据如下：

*import numpy as np*

*Xi=np.array([6.19, 2.51, 7.29, 7.01, 5.7, 2.66, 3.98, 2.5, 9.1, 4.2])*

*Yi=np.array([5.25, 2.83, 6.41, 6.71, 5.1, 4.23, 5.05, 1.98, 10.5, 6.3])*

③确定需要拟合的形状。

这里只要求使用直线，因此为*y = k\*x + b*的形式。注意：使用一个函数来定义该形状，函数的参数为k，b，x，函数的返回值为y。提示：用列表的形式将k和b组装成一个参数，如p=[k的值, b的值]，在函数内只需使用*k, b=p*即可读取相应的值(注意k和b的顺序)。(python中用中括号[]表示列表)

④定义误差函数。

误差应等于上一步中的返回值与要拟合的点的y值的差。提示：不需要平方。

⑤初始化k和b。利用第三步中提到的列表形式使用一个变量初始化k和b的值，如：

*p0=[1, 20]*

其中k初始化为1，b初始化20，这个 初始值可以随便设置。

⑥调用leasteq函数。该函数一般只用到三个参数：第一个参数为前面定义好的误差函数，只需传入函数的名字即可；第二个参数为需要估计的参数的初始值，在这里即为k和b的初始值，该参数要求为列表的形式；第三个参数为定义好的误差函数使用到的其它参数的值，即除了k和b之外的值，这里是x和y，需要用小括号()括起来，表示tuple类型的数据。使用示例如下：

*result=leasteq(error,p0,args=(Xi,Yi))*

⑦获取结果。设result为leasteq函数的返回变量，则需要的结果保存在result[0]中，也是一个列表的形式，如：

*k,b=result[0]*

可以将k和b值打印出来查看结果。

**\*\*\*注意\*\*\***

如果想将数据点和拟合的直线可视化，可以使用matplotlib.pyplot模块，代码如下：

*import matplotlib.pyplot as plt*

*plt.figure(figsize=(8,6))*

*plt.scatter(Xi,Yi,color="blue",label="Data Points",linewidth=2)*

*x=np.linspace(0,15,100)*

*y=k\*x+b*

*plt.plot(x,y,color="red",label="Fitting Result",linewidth=2)*

*plt.legend(loc='lower right')*

*plt.show()*

2、实现最小二乘法函数拟合直线

提示：

求解公式如下：(参考[维基百科](https://zh.wikipedia.org/wiki/%E6%9C%80%E5%B0%8F%E4%BA%8C%E4%B9%98%E6%B3%95))

其中n为样本点个数，和分别是x和y坐标的平均值。

实现完函数后可按照第1部分的实验检验一下是否跟直接调用所产生的结果一样。

**\*\*\*Python Tips\*\*\***

python中定义函数的语法如下：

def function\_name(arg0, arg1,...):

*函数体*

python中使用“#”来添加注释。要想在python代码中添加中文注释，需要在代码最开始的地方添加以下语句：  
 *# -\*- coding:utf-8 –\*-*