1. 由所给出点计算相应的梯度和hessin矩阵



梯度函数为：

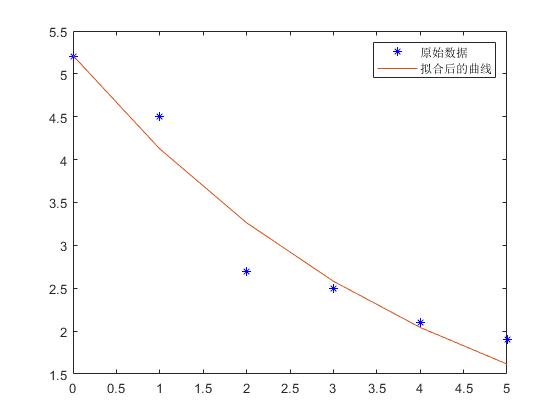


Hessin矩阵在此处为了简便对其进行化简：



此处忽略了高阶项。

最终拟合的结果为：





2，

1. KKT条件为：
2. 我们可以对原式进行一下变化，



所以原始KKT变成：因为D是对角矩阵，所以



假设不为零，所以可以得到：



可以得出是A的零空间向量记作Null(A)，则 

将 带入得到即， 。

假设为0，则：



将 带入得到即， 。

3，

1. 采用固定步长的梯度下降方法

* 首先求出函数的梯度，计算Euclidean norm，若符合要求则退出程序，若不符合则继续进行。

，





* 设置固定的步长为0.2，初始和分别为 和1。
* 计算搜索方向,
* 令，。



* 回到初始步骤



图 1 采用梯度下降Euclidean norm

CPU占用时间为：24.626秒，



迭代次数为1592次，



|  |  |  |
| --- | --- | --- |
| Euclidean norm | 迭代次数 | CPU占用时间/秒 |
| 10-2 | 19 | 0.749 |
| 10-3 | 167 | 2.791 |
| 10-4 | 1592 | 24.626 |

1. 采用固定步长的共轭梯度方法

* 设定初始的搜索方向为梯度下降方向，相应的初始位置，参数于方法一一致。
* 令，

其中，y为相应的w和c。

* 假如则停止计算，否则进行一维搜索。

，

* 返回继续进行迭代。



图 2 采用共轭梯度下降Euclidean norm

CPU占用时间：4.120秒



迭代次数为310次，



|  |  |  |
| --- | --- | --- |
| Euclidean norm | 迭代次数 | CPU占用时间/秒 |
| 10-2 | 34 | 0.659 |
| 10-3 | 101 | 1.504 |
| 10-4 | 310 | 4.120 |

可以看出在处理精度比较低时候，两种方法相差不大，但是在处理高精度的情况下，第二种方法比较好。