实验14 数值最优化方法

参考答案

1. 首先画出函数图形如下，可见该函数在给定区间[1,5]上是单峰函数，适合使用精确线搜索算法求最小值。



黄金分割搜索法：function [S,E,G]=golden(f,a,b,delta,epsilon)（见代码文件）

Fibonacci搜索法：function [X,E,G]=fibonacci(f,a,b,tol,e) （见代码文件）

其中，需要调用生成Fibonacci序列的程序：function [y,n]=fib(n) （见代码文件）

1. 抛物线搜索法：function [p,yp,dp,dy,P] = quadmin(f,a,b,delta,epsilon) （见代码文件）

取横坐标容许误差10-9，纵坐标容许误差10-11，Fibonacci搜索法中的区别常数为0.001，三种算法的求解结果比较如下表：

|  |  |  |  |
| --- | --- | --- | --- |
| 算法 | 黄金分割搜索法 | Fibonacci搜索法 | 抛物线搜索法 |
| 最小值点 | (3.33333335727376, 4.48148148148148) | (3.33333333435288, 4.48148148148147) | (3.3333333358177, 4.48148148148147) |
| 误差 | (3.720947994168e-10, 0) | (1.73874248332595e-11, 0) | (3.47860563603882e-09, 7.105427357601e-15) |
| 迭代次数 | 41 | 46 | 6 |
| 收敛阶 | 线性 | 线性 | 超线性 |

1. （1）画图程序见代码文件FigBanana.m，函数图形和等高线图如图1-图3所示：



图1 函数图 图2 等高线图



图3 3D等高线图 图4 搜索初值点和最小值点



图5 由BFGS方法求得最小值点的搜索过程



图6 由DFP方法求得最小值点的搜索过程



图7 由最速下降法求得最小值点的搜索过程



图8 由Levenberg-Marquardt方法求得最小值点的搜索过程

当使用迭代起始点(-1.9,2)，（2）-（4）利用fminunc、fminsearch和lsqnonlin函数求解的程序见代码文件MinBanana.m和LsqBanana.m，结果如下表及图4-图8所示。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 方法 | | 最小值点 | 最小值 | 迭代  次数 | 函数求值次数 | 是否收敛 |
| fminunc函数(拟牛顿法) | BFGS法 | (0.999815490884581, 0.999628681574425) | 3.45884805405539e-08 | 34 | 50 | 是 |
| DFP法 | (1.00025505217543, 1.00051206177227) | 6.54097185349984e-08 | 277 | 321 | 是 |
| 最速下降法 | (0.99820418747436,  0.996405448665216) | 3.22872638553737e-06 | 1280 | 4539 | 是 |
| fminsearch函数  (单纯形法) | | (1.0000166688948, 1.00003447386277) | 4.06855153506342e-10 | 114 | 210 | 是 |
| lsqnonlin函数  (Levenberg-Marquardt法) | | (0.999999993685003 0.999999987215632) | 4.22623125444993e-17 | 28 | 42 | 是 |

Resenbrock函数为两个平方数的和，当*x*1=*x*2=1时，整个目标函数有最小值0。可以看出，在相同的设置下，最速下降法得到的结果最差。这是因为最速下降法特别不适合于从一狭长通道到达最优解的情况。另一方面，使用默认的BFGS方法的fminunc函数的效率明显高于fminsearch函数，因为对目标函数调用的次数与迭代的步数都明显少于后者，但由前者得出的解的精度没有后者高。而在这个求平方和最小的具体问题中，使用lsqnonlin函数把问题当作最小二乘优化问题来求解显然比上述两种方法更合适。

从三维等高线图（图3）中可以看出，函数的最小值点在图中的一个很窄的白色带状区域内，在这个区域内函数值的变化较为平缓，形如香蕉，故Rosenbrock函数又称为香蕉函数。这种在最值点附近变化缓慢的特性会造成算法在解点附近迭代过慢的问题。所以这个函数经常用来测试最优化算法的优劣。