实验3：无约束优化

PB19000196 晏瑞然

问题描述

实现基于 Wolfe-Powell 准则的非精确一维步长搜索算法，基于非精确一维步长搜索，从牛顿法、拟牛顿类方法(DFP/BFGS)、共轭梯度法中选择一种算法，手动实现。

需至少构造一个函数 (例如 Rosenbrock 函数)，应用算法在不同初始值下求解无约束最优化问题，并分析不同初值点对结果的影响.

算法原理

对于无约束优化问题，通过Wolfe-Powell非精确一维搜索搜索出步长，再用牛顿法迭代得到最优解。

基于 Wolfe-Powell 准则的非精确一维搜索算法

1. 给定初始一位搜索区间 $[0, \bar{\alpha}]$ ，以及 $\rho \in (0, 1/2), \sigma \in (\rho, 1)$ 计算 $\varphi_0 = \varphi(0) = f(x^{(k)}), \varphi_0' = \varphi_0'(0) = \nabla f(x^{(k)})^T d^{(k)}$. 并令 $a_1 = 0, a2 = \bar{\alpha}, \varphi_1 = \varphi_0, \varphi_1' = \varphi_0'$. 选取适当的 $\alpha \in (a_1 . a_2)$.
2. 计算 $\varphi = \varphi(\alpha) = f(x^{(k)} + \alpha d^{(k)})$. 若 $\varphi(\alpha) \le \varphi(0) + \rho\alpha\varphi'(0)$, 则转3.，否则，由 $\varphi_1, \varphi_1', \varphi$ 构造两点二次插值多项 $p^{(1)}(t)$, 并得到其最小点

$$\hat{\alpha} = a_1 + \frac{1}{2}\frac{(a_1 - \alpha)^2 \varphi_1'}{(\varphi_1 - \varphi) - (a_1 - \alpha)\varphi_1'}$$

   于是置 $a_2 = \alpha, \alpha = \hat{\alpha}$, 重复2.

3. 计算 $\varphi' = \varphi(\alpha)' = f(x^{(k)} + \alpha d^{(k)})$, 若 $\varphi'(\alpha) \ge \sigma\varphi'(0)$, 则输出 $\alpha_k = \alpha$ 并停止搜索。否则，由 $\varphi, \varphi', \varphi_1'$ 构造两点二次插值多项 $p^{(2)}(t)$ ，并求极小点

$$\hat{\alpha} = \alpha - \frac{(a_1 - \alpha)\varphi'}{\varphi_1' - \varphi'}$$

   置 $a_1 = \alpha, \alpha = \hat{\alpha}, \varphi_1 = \varphi, \varphi_1' = \varphi'$ ，返回2.

牛顿法一般迭代格式

对如下无约束优化问题

$$\min_x f(x)$$

牛顿法一般迭代格式如下:

(0) 初始化：选取适当的初始点 $x^{(0)}$，令 $k := 0$.
(1) 计算搜索方向：$d^{(k)} = -\nabla^2 f\left(x^{(k)}\right)^{-1} \nabla f\left(x^{(k)}\right)$
(2) 确定步长因子：采用非精确的一维搜索确定步长因子 α_k.
(3) 更新迭代点：令 $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$ 置 $k := k + 1$, 返回第 (1) 步.

数据集说明

可自行设定目标函数以及自定义参数进行测试，具体测试数据见程序测试结果。

程序说明

文件说明

代码一共3个文件分别为f.m, WolfePowell.m, NewtonMethod.m文件。

f.m是目标函数文件，可以自行修改。

WolfePowell.m为WolfePowell非精确一维搜索函数，会在NewtonMethod.m中被调用。

NewtonMethod.m即为牛顿法求解无约束最优化问题的脚本文件，直接运行该文件就能得到结果。

环境说明

使用MATLAB运行运行NewtonMethod.m文件，所使用的MATLAB必须带有Symbolic Math Toolbox，因为要用到里面的符号计算功能来计算符号梯度。

自定义输入

所有的参数输入都在NewtonMethod.m中，具体参数解释如下

```
eps %数值误差，两数的差小于这个误差时可以看作数值相等
alpha_ %初始搜索区间上界
rho %WolfePowell算法中参数rho
sigma %WolfePowell算法中参数sigma
x %初始x_0
```

同时可以修改最优化目标函数，直接修改f.m的内容即可，默认函数为Rosenbrock 函数，即

$$f(x)=\sum_{i=1}^{d-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2],x\in R^d$$

输出说明

输出样例如下:

```
>> NewtonMethod

itr =

    22


x =

    1.0000    1.0000


ans =

   4.5665e-13

```

itr表示牛顿迭代次数，x表示找到的极值点，ans为目标函数极值。

程序测试结果

测试1:

f.m:

```
function y = f(x)
%Rosenbrock 函数
    n = length(x);
    y = 0;
    for i=1:n-1
        y = y + (100*(x(i+1)-x(i)^2)^2+(x(i)-1)^2);
    end
end
```

参数:

```
eps = 1e-6;
alpha_=1;
rho=0.25;
sigma=0.5;
x=[0,0];
```

结果:

```
>> NewtonMethod

itr =

    23


x =

    1.0000    1.0000


ans =

   3.0335e-14
```

可以看到极值点为 $x = (1, 1)$ ，目标极值为0(忽略数值误差)。

测试2:

f.m仍然是Rosenbrock 函数

参数:

```
eps = 1e-6;
alpha_=1;
rho=0.25;
sigma=0.5;
x=[0,0,0,0,0];
```

结果:

```
>> NewtonMethod

itr =

    17


x =

    1.0000    1.0000    1.0000    1.0000    1.0000


ans =

   2.7788e-16
```

可以看到极值点为 $x=(1,1,1,1,1)$，目标极值为0(忽略数值误差)，其也为全局最小值。

测试3:

f.m仍然是Rosenbrock 函数

参数:

```
eps = 1e-12;
alpha_=0.1;
rho=0.25;
sigma=0.5;
x=[-5,1,1,1,-5];
```

结果:

```
>> NewtonMethod

itr =

    35


x =

   -0.9621    0.9357    0.8807    0.7779    0.6051


ans =

    3.9308
```

可以看到极值点为 $x=(-0.9621,0.9357,0.8807,0.7779,0.6051)$，目标极值为3.9308，其为一个局部最小值(梯度为0点)。

测试4:

f.m:

```
function y = f(x)
% X^{T}X 函数
    n = length(x);
    y = 0;
    for i = 1:n
        y = y+x(i)^2;
    end
end
```

参数:

```
eps = 1e-6;
alpha_=1;
rho=0.25;
sigma=0.5;
x=[1,1];
```

结果:

```
>> NewtonMethod

itr =

    22


x =

   1.0e-06 *

    0.2384    0.2384


ans =

   1.1369e-13
```

可以看到极值点为 $x = (0, 0)$ (忽略数值误差)，目标极值为0(忽略数值误差)。

其他

对该程序还做了一系列其他不同初始值以及不同参数的实验，这里就不一一列出，在下面**分析总结**中会进行解释。

分析总结

本分析主要对Rosenbrock 函数为目标函数多的牛顿法进行分析，具体分为一下几个部分。

1. 初始值选取：Rosenbrock函数在维数小于4时，只有一个极值点，也是全局最小值点，即全1向量，但当维度较高时，会在 $(-1, 1, \cdots, 1)$ 处有另一个极值点，这并不是全局最优点，但因为也是极值点，仍会被牛顿法找到，可见**测试3**中的结果。

2. 对 $\bar{\alpha}$ 的选择：程序对 $\bar{\alpha}$ 非常敏感，可以尽量把 $\bar{\alpha}$ 取小，不然可能出现一直找不到grd=0的情况。原因是 $\bar{\alpha}$ 其实是我们人为估计的，我们并不知道，真实 $\bar{\alpha}_k$ 应该是使得 $f(x^{(k)}+\alpha d^{(k)})=f(x^{(k)})$ 的最小正数 α，否则我们就不能用二次函数去拟合 $\varphi-\alpha$ 曲线了，WolfePowell方法也就失效了。所以一个合适的参数alpha_的值是非常重要的。
3. 牛顿法缺陷：牛顿法本身存在缺陷，可能出现f(x)的hessian矩阵奇异的情况，比如初值选取 $x=[-1,0,0,0,0]$ 就会出现 "警告: 矩阵为奇异值、接近奇异值或缩放错误。结果可能不准确。RCOND = NaN" 的报错。，表示迭代过程中出现hessian矩阵奇异的情况。解决方法是可以采用课上所讲的方法，对牛顿法下降方向进行修正，就不会产生上述情况。