障碍内点法实现实验报告

内容：技术文档与算法文档
姓名：付乐豪
学号：PB20071456
时间：2023 年 6 月 30 日

目录

1 技术文档

1.1 运行环境

本程序使用 python3.9 版本实现，主要依赖于一下库：

- import pandas as pd
- import numpy as np
- import sympy

- from sympy import *
- from sympy import symbols, Matrix, diff
- from cvxopt import solvers, matrix
- import cvxopt
- import matplotlib.pyplot as plt

1.2 包含文件

- 内点法.py：主要实现障碍内点法算法的脚本文件。
- 迭代过程记录 1.csv：记录了测试函数 1 迭代过程中的迭代点坐标以及当前迭代点的目标函数值。
- 迭代过程记录 2.csv：记录了测试函数 2 迭代过程中的迭代点坐标以及当前迭代点的目标函数值。
- gap1.png：绘制了测试函数 1 迭代过程中目标函数的下降情况。
- gap2.png：绘制了测试函数 2 迭代过程中目标函数的下降情况。

1.3 参数配置

- input_fun()：用户可以通过是否注释此函数，选择在运行端手动输入目标函数还是直接修改代码运行。
- run(func, eq_cons, ineq_cons, start, gamma, t, iters=100, epsilon=1e-3)：运行内点法算法的主函数。其中 func 为目标函数，eq_cons 为等式约束，ineq_cons 为不等式约束，iters 为最大迭代次数，epsilon 为迭代过程中的容忍度。其中 iters 默认值为 100 次，如果需要更改必须为 int 类型，epsilon 默认值为 1e-3。

1.4 调用方法

本程序提供了两种使用办法（默认使用第二种办法，即用户在运行端输入）：

(1) 直接在程序的 218-225 行定义目标函数，约束，起始点和参数 γ, t，并注释 227 行后直接运行。如下图：

```
218  # 测试样例2
219  func = -log(x[0] + x[1])
220  ineq_cons = [-1 - x[1]]
221  eq_cons = [x[0] + 2*x[1] - 1]
222  x0 = 3; x1 = 0
223  start = Matrix([[x0, x1]]).T
224  gamma = 1.1
225  t = 1
226
227  # func, eq_cons, ineq_cons, start, gamma, t = input_fun()
```

图 1: 用户输入部分代码

(2) 取消 227 行的注释，在运行段手动输入目标函数，约束，起始点和参数 γ, t，其中目标函数和约束中 x 的两个分量分别以 x_0 和 x_1 表示，不等式约束默认为输入的函数 ≤ 0；如果已经输入了全部的不等式约束或者等式约束，请输入空格以结束，如下图：

```
请输入目标函数，其中分量用x_0和x_1表示：4 * x_0**2 + x_1**2
如果已经输入了所有不等式或者等式约束，请输入空格。
请逐个输入不等式约束(<=0)：x_1 - 1
请逐个输入不等式约束(<=0)：
请逐个输入等式约束：x_0 + x_1 - 1
请逐个输入等式约束：
请输入x0的初始值： 2
请输入x1的初始值： -1
请输入参数gamma的值(gamma>1)：5
请输入参数t的初始值(t>0)：1
```

图 2: 手动输入范例

1.5 输出结果

内点法脚本运行过程中，会输出每一步的迭代点和目标函数在当前点的取值。如果算法收敛，在运行结束后会输出最优值和最优值点，将记录的迭代过程保存在”迭代过程记录.csv”文件中，并自动绘制迭代曲线。如果在达到设置的最大迭代次数后，仍然没有满足终止条件，也会输出结果，并提示用户应该修改迭代次数和容忍度。如下图:

```
第0次迭代：当前迭代点  x0 = 2.0, x1 = -1.000 | 迭代点目标函数值: 17.000
第1次迭代：当前迭代点  x0 = 0.431, x1 = 0.569 | 迭代点目标函数值: 1.067
第2次迭代：当前迭代点  x0 = 0.273, x1 = 0.727 | 迭代点目标函数值: 0.827
第3次迭代：当前迭代点  x0 = 0.218, x1 = 0.782 | 迭代点目标函数值: 0.802
第4次迭代：当前迭代点  x0 = 0.204, x1 = 0.796 | 迭代点目标函数值: 0.800
第5次迭代：当前迭代点  x0 = 0.201, x1 = 0.799 | 迭代点目标函数值: 0.800
一共迭代了5次
取到最小值的点为: Matrix([[0.200159870597899], [0.799840129402100]])
目标函数的最小值为: 0.800000127793040
```

图 3: 算法收敛的输出结果

```
第0次迭代：当前迭代点  x0 = 2.0, x1 = -1.000 | 迭代点目标函数值: 17.000
第1次迭代：当前迭代点  x0 = 0.431, x1 = 0.569 | 迭代点目标函数值: 1.067
第2次迭代：当前迭代点  x0 = 0.375, x1 = 0.625 | 迭代点目标函数值: 0.953
第3次迭代：当前迭代点  x0 = 0.332, x1 = 0.668 | 迭代点目标函数值: 0.887
第4次迭代：当前迭代点  x0 = 0.298, x1 = 0.702 | 迭代点目标函数值: 0.848
一共迭代了5次, 未找到最优值，请尝试更改迭代次数iter和容忍度epsilon
```

图 4: 算法不收敛的输出结果

2 算法文档

2.1 算法流程

对于含有等式约束和不等式约束的凸优化问题，障碍内点法通过障碍函数将原始问题化为只含有等式约束的优化问题，然后对此问题使用牛顿法求解，算法流程如下图:

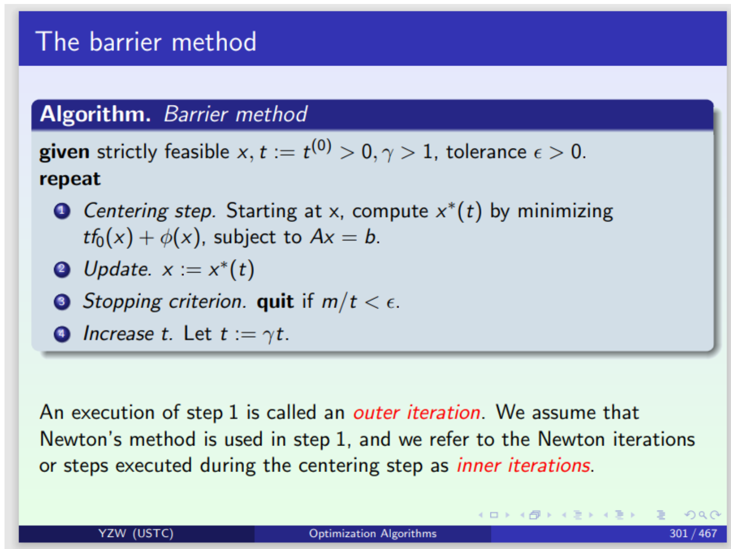

图 5: 障碍内点法算法流程（外层循环）

而对于 $\min tf_0(x) + \Phi(x)$ s.t. $Ax - b = 0$ 这一等式约束问题，我们使用牛顿法求解，如下图：

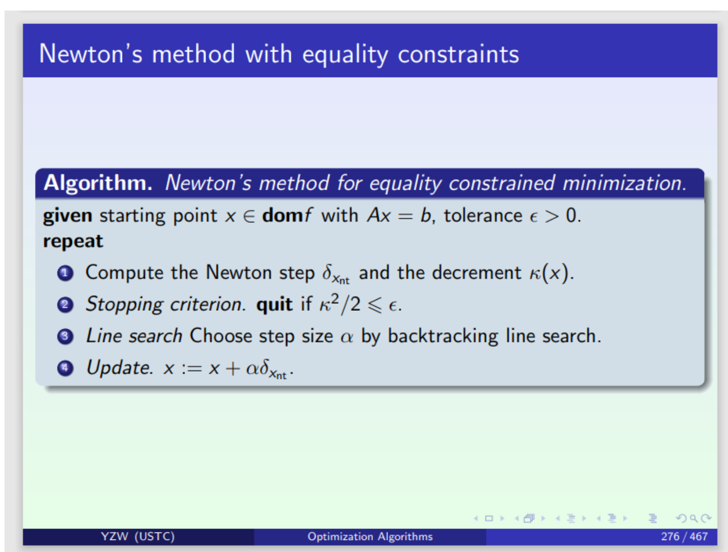

图 6: 障碍内点法算法流程（内层循环）

2.2 实验结果

2.2.1 实验设置

为了测试算法的效果，这里选择了两个优化问题来实现

问题 1:

$$\begin{aligned} \min_{x\in\mathbb{R}^{2\times 2}} \quad & 4x_0^2 + x_1^2 \\ s.t. \quad & x_1 - 1 \le 0 \\ & x_0 + x_1 - 1 = 0 \end{aligned} \tag{1}$$

问题 2:

$$\begin{aligned} \min_{x\in\mathbb{R}^{2\times 2}} \quad & -\log(x_0 + x_1) \\ s.t. \quad & -1 - x_1 \le 0 \\ & x_0 + 2x_1 = 1 \end{aligned} \tag{2}$$

2.2.2 测试结果

一下函数迭代图为选定某一组参数的情况下，两个测试函数的障碍内点法的收敛情况:

测试函数 1:

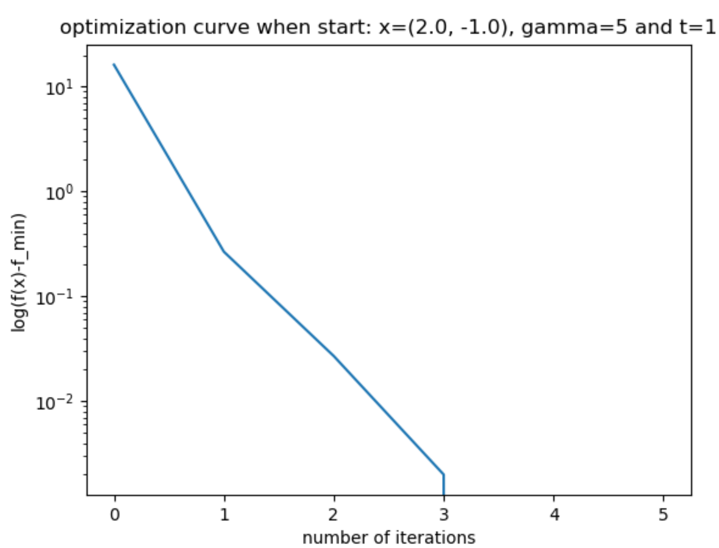

图 7: 测试函数 1 收敛曲线

且选取不同的超参数和初始点时，最终收敛点和收敛值均为：$x_{opt} = (x_0, x_1) = (0.2, 0.8), f_{min} = 0.8$

测试函数 2:

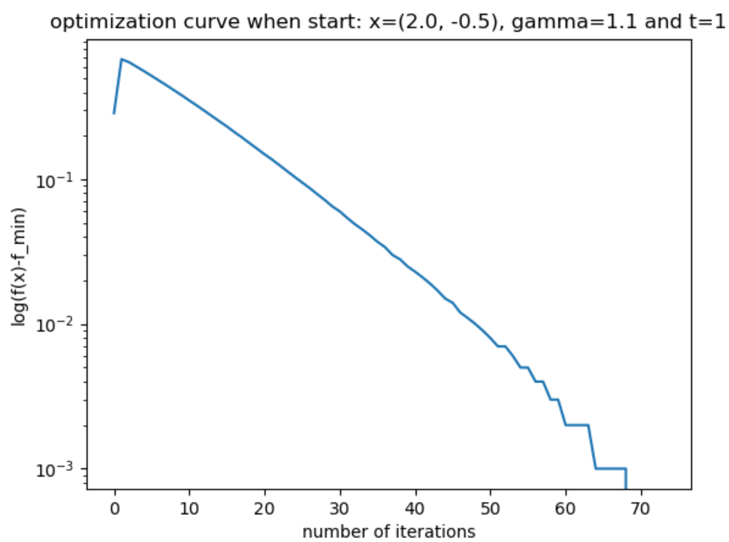

图 8: 测试函数 2 收敛曲线

且选取不同的超参数和初始点时，最终收敛点和收敛值均为：$x_{opt} = (x_0, x_1) = (3, -1), f_{min} = -\log(2)$

2.2.3 参数与算法收敛性

障碍内点法中，需要人为事先指定的超参数有 γ，t 和迭代初始点，这里仅以测试函数 1 作为例子研究这两个超参数对于算法收敛性的影响。

表 1: γ 和最终迭代函数值（固定 $start = (1, 0), t = 1$）

参数 γ	**最终函数值**	**迭代次数**
γ	$f(x)$	iterations
1.1	0.8	73
1.5	0.8	18
3	0.8	7
10	0.8	4

从上表可以看出，在满足 $\gamma > 1$ 的条件下，随着 γ 增大，迭代到收敛点的次数减少，迭代速度增加。

表 2: 初始点和最终迭代函数值（固定 $\gamma = 5, t = 1$）

第一坐标分量	**第二坐标分量**	**最终函数值**	**迭代次数**
x_0	x_1	$f(x)$	iterations
0.5	0.5	0.8	7
1	0	0.8	7
2	-1	0.8	7
100	-99	0.8	7

从上表可以看出，在满足初始点为可行域内点，即严格可行的情况下，初始点对于收敛速度的影响较小。

表 3: t 和最终迭代函数值（固定 $start = (1, 0), \gamma = 1.5$）

参数 t	**最终函数值**	**迭代次数**
t	$f(x)$	iterations
0.5	0.8	19
1	0.8	18
2	0.8	16
5	0.8	14
10	0.8	12

从上表可以看出，在满足 $t > 1$ 的条件下，随着 t 增大，迭代到收敛点

的次数减少，迭代速度增加。

总的来说，上述讨论的三个超参数（初始点，γ, t）中，对于收敛速度有影响的是 γ 和 t，二者越大时，收敛速度越快；而对比二者时，发现 γ 对于收敛速度的影响更大。

3 总结

本程序实现了障碍内点法算法，在内层循环时使用 Newton 算法处理含有等式约束的优化问题，实现了算法的基本功能，并测试了 γ, t 和迭代初始点这三个超参数对于收敛速度的影响。同时本程序还提供了一定的用户接口，可以让用户自主选择使用方式。