**最优化HW4编程报告**

1. **罚函数法**

**1.1简要介绍**

罚函数法是一种用于处理带约束优化问题的方法，其中外点法（Exterior Point Method）是罚函数法的一种变体，用于求解线性规划问题。外点法通过在目标函数中引入罚函数，将原始问题转化为无约束的问题，并通过罚函数的增加来逼近原始问题的可行解集合。以下是外点法的简要介绍和算法流程：

1. 初始化： 选择初始点x₀，定义罚函数的初始参数λ₀（通常设为较小的正数），以及收敛容忍度ε。
2. 构建罚函数： 将原始问题的目标函数加上罚函数，形成新的目标函数。罚函数通常是对每个约束条件的惩罚项的总和。例如，对于线性规划问题，可以使用对数障碍罚函数或二次罚函数。
3. 对数障碍罚函数：将每个约束条件cᵢ(x)引入罚函数中，形成目标函数为f(x) + λ₀Σln(-cᵢ(x))。
4. 二次罚函数：形式为f(x) + λ₀Σmax(0, cᵢ(x))²。
5. 求解无约束问题： 使用某种优化算法（例如梯度下降法或牛顿法）求解加入罚函数的目标函数，得到新的解x₁。
6. 更新罚参数： 根据当前解x₁和罚函数的形式，更新罚函数的参数λ₁。
7. 检查停止条件： 如果罚参数λ收敛到零或达到预定的收敛容忍度，停止迭代。否则，返回步骤3。

外点法的核心思想是通过逐步增加罚函数的惩罚力度，将约束问题的可行域逼近为一个点。在迭代过程中，罚参数逐渐减小，使得罚函数的影响逐渐减弱。最终，通过不断迭代，算法收敛到原始问题的最优解。

**1.2 算例介绍**

目标函数：f=(x1-3)^2+(x2-2)^2;

约束：4-x1-x2>=0

初始点：x0=[0,0]

罚函数构建方法：外点法

计算最小值方法：最小牛顿法

**1.3 实验结果**

|  |  |
| --- | --- |
| 迭代次数： | 13 |
| 最优解： | [2.5000 1.5000] |
| 最小值： | 0.5000 |
| 消耗时间： | 0.1094 |

1. **增广拉格朗日函数法**

**2.1简要介绍**

增广拉格朗日函数法是一种用于处理等式和不等式约束的非线性规划问题的方法。它通过引入拉格朗日乘子来将原始问题转化为无约束问题，从而求解原问题的最优解。增广拉格朗日函数法也称为增广拉格朗日乘子法或增广拉格朗日对偶法。以下是增广拉格朗日函数法的简要介绍和算法流程：

1. 定义增广拉格朗日函数： 对于带等式约束的最小化问题，定义增广拉格朗日函数为：*L*(*x*,*λ,μ*)=*f*(*x*)+*λ\*g+1/2\*μ\*g2*
2. 构建拉格朗日对偶函数： 将增广拉格朗日函数关于变量x最小化，得到拉格朗日对偶函数：g(*λ,μ*)=inf x *L*(*x*,*λ,μ*)
3. 求解对偶问题： 构建拉格朗日对偶问题，即最大化拉格朗日对偶函数：max λ≥0​g(*λ,μ*)对偶问题的解即为原始问题的最优值的下界。
4. 迭代优化： 通过迭代优化求解原始问题和对偶问题：
5. 原始问题优化： 固定拉格朗日乘子λ，最小化增广拉格朗日函数关于变量x的部分。通常使用梯度下降法等优化算法。
6. 对偶问题优化： 固定优化后的x，最大化拉格朗日对偶函数关于λ的部分。通常使用梯度上升法等优化算法。
7. 更新拉格朗日乘子： 根据优化后的x和λ更新拉格朗日乘子。
8. 检查停止条件： 检查是否满足停止条件，例如目标函数的变化足够小或拉格朗日乘子的变化足够小。如果满足条件，则停止迭代。
9. 输出结果： 输出原始问题的最优解x和最优值，以及对偶问题的最优解λ 。

**2.2算例介绍**

目标函数：f = x1^2 + x2^2;

约束：x1+x2-1=0

初始点：x0=[10,10]

计算最小值方法：BFGS法

**2.3 实验结果**

|  |  |
| --- | --- |
| 计算最小值方法 | BFGS |
| 迭代次数： | 2 |
| 最优解： | [5.000000000000000e-01; 5.000000000000000e-01] |
| 最小值： | 0.50000000000 |
| 消耗时间： | 7.812500000000000e-02 |

1. **单调变分不等式求最短距离和问题**

**3.1简要介绍**

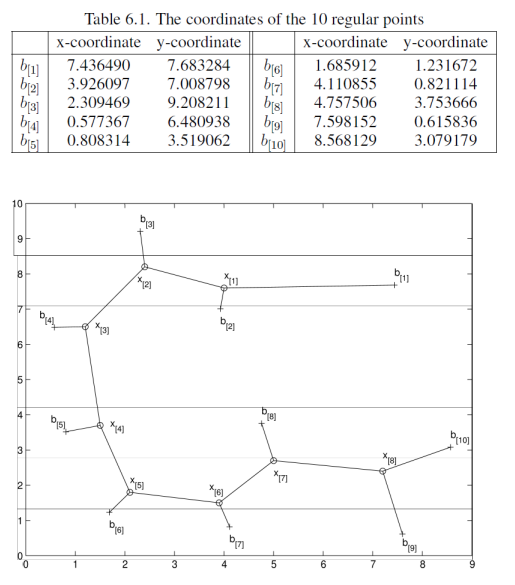
在数学和计算几何中，单调变分不等式（Monotone Variational Inequality）是一类重要的非线性不等式问题，而求解最短距离和问题是其中的一种特殊情况。最短距离和问题通常涉及到在一个特定的区域内找到两点之间的最短路径，而单调变分不等式提供了一种形式化的框架来解决这类问题。具体来说，单调变分不等式的一般形式如下：

⟨F(x), y−x ⟩≥0,∀y∈K

解决最短距离和问题的一种常见方法是使用单调变分不等式的迭代算法。算法的基本思想是从一个初始点开始，通过迭代的方式调整当前点，使得满足单调变分不等式，逐步逼近最优解。这类算法通常包含以下步骤：

1. 选择初始点： 选择一个初始点作为算法的起始点。
2. 迭代更新： 通过迭代更新当前点，使得满足单调变分不等式条件。更新规则通常涉及到目标函数的梯度或导数信息。
3. 收敛判断： 检查算法是否满足停止条件，例如目标函数的变化足够小或满足一定精度要求。如果满足条件，则停止迭代，输出最优解。
4. 输出结果： 输出最优解，即满足单调变分不等式条件的点。

**3.2算例介绍**



迭代方法：单调线性变分不等式的投影收缩算法

**3.3 实验结果**

|  |  |
| --- | --- |
| 迭代次数： | 143 |
| 最优解： | 3.92609689945923 7.62687688923009  2.30946894908847 7.62687689989858  1.13488202123319 6.48093980127285  1.13488202123319 3.51906191697979  1.68591197684740 1.44759788804638  4.11085500728153 1.44759788813656  4.75750619837140 1.64128276933863  7.59815010249336 1.64128276937353 |
| 最小值： | 2.866586217285111e+01 |
| 消耗时间： | 0 |

