等式约束优化问题的求解方法与牛顿法

摘要

等式约束优化问题在工程、经济和科学计算中广泛存在，其求解对于优化理论和实践具有重要意义。本文系统介绍了等式约束优化问题的主要求解方法，包括拉格朗日乘子法、消元法、Null-space 方法和序列二次规划（SQP）。随后，详细讨论了牛顿步和牛顿减量的定义、性质及其几何意义，并介绍了不可行初始点的牛顿法及其收敛性分析。最后，深入探讨了 SQP 方法的算法框架、理论基础及应用。通过这些方法的系统性阐述，为深入理解等式约束优化问题的求解提供了全面的视角。

目录

1 引言

在实际问题中，许多优化问题都存在约束条件，尤其是等式约束。例如，在资源分配、机械设计和经济规划中，都需要满足特定的平衡条件。如何有效地求解这些带有等式约束的优化问题，是优化理论中的重要课题。本文将探讨等式约束优化问题的主要求解方法，以及牛顿法在此类问题中的应用。

2 等式约束优化问题的求解方法

2.1 问题描述

一般形式的等式约束优化问题可以表示为：

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x), \\ \text{subject to} \quad & h_i(x) = 0, \quad i = 1, 2, \ldots, m, \end{aligned} \tag{1}$$

其中，$f : \mathbb{R}^n \to \mathbb{R}$ 是目标函数，$h_i : \mathbb{R}^n \to \mathbb{R}$ 是约束函数。

2.2 拉格朗日乘子法

拉格朗日乘子法通过引入拉格朗日乘子，将约束条件融入目标函数，从而将受约束问题转化为无约束问题。

定义 2.1 (拉格朗日函数)**.** 引入拉格朗日乘子 $\lambda \in \mathbb{R}^m$，构建拉格朗日函数：

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i=1}^{m} \lambda_i h_i(x).$$

通过对拉格朗日函数求偏导数，可以得到必要的最优性条件。

定理 2.1 (KKT 条件)**.** *若 x^* 是问题（1）的局部极小值，且满足一定的正则性条件（如约束条件的线性独立），则存在 $\lambda^* \in \mathbb{R}^m$，使得以下条件成立：*

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) - \sum_{i=1}^{m} \lambda_i^* \nabla h_i(x^*) = 0, \tag{2}$$

$$h_i(x^*) = 0, \quad i = 1, 2, \ldots, m. \tag{3}$$

理解

KKT 条件是等式约束优化问题的必要条件，它将约束条件与目标函数的梯度联系起来，提供了寻找最优解的途径。满足 KKT 条件的点被称为 KKT 点，可能是最优解。

2.3 消元法

消元法通过利用约束条件直接消去部分变量，将原始问题转化为一个较小规模的无约束优化问题。这种方法适用于约束条件可以显式表示为某些变量的函数。

示例 2.1. 考虑一个简单的等式约束优化问题：

$$\min_{x,y} \quad f(x,y) = x^2 + y^2, \quad \textit{subject to} \quad y = 1 - x.$$

通过约束条件，可以将 y 表示为 $1-x$，将问题转化为：

$$\min_{x} \quad f(x) = x^2 + (1-x)^2.$$

解得 $x = \frac{1}{2}$，$y = \frac{1}{2}$。

消元法的优势在于简化问题规模，但缺点是仅适用于可以明确消去变量的情况，对于复杂或隐式约束条件，消元法难以应用。

2.4 Null-space 方法

Null-space 方法通过将搜索方向限制在约束的 null-space 内，保持搜索过程中对约束的满足。

1. 计算约束的雅可比矩阵 $A = [\nabla h_1(x), \nabla h_2(x), \dots, \nabla h_m(x)]^\top$。
2. 求解 A 的 null-space，找到基矩阵 Z，使得 $AZ = 0$。
3. 表示步长 s 为 $s = Zd$，其中 d 是一个自由变量。
4. 将搜索方向限定在 null-space 中，通过优化 d 来找到步长 s。

理解

Null-space 方法通过分离变量，将问题转化为在自由变量空间内进行优化，确保所有的约束条件在每次迭代中都被满足。

2.5 序列二次规划（SQP）方法

定义 2.2 (序列二次规划（SQP）方法)**.** *序列二次规划（Sequential Quadratic Programming, SQP）是一种用于解决非线性约束优化问题的迭代方法。在每次迭代中，SQP 方法通过构建一个二次规划（Quadratic Programming, QP）子问题来近似原始问题，并求解该子问题以获得搜索方向。*

SQP 方法结合了牛顿法和拉格朗日乘子法的优点，通过逐步逼近优化问题的最优解。每次迭代中，SQP 构建一个局部二次模型和线性化约束条件，形成一个二次规划子问题。通过求解该子问题，获得步长和拉格朗日乘子，并更新迭代点。

2.5.1 SQP 方法的算法步骤

1. 初始化 x_0，设置容差 $\epsilon > 0$，选择初始的 Hessian 矩阵近似 B_0（通常选为目标函数的 Hessian 或其近似）。
2. 对于 $k = 0, 1, 2, \dots$：
   (a) 计算当前点的梯度 $\nabla f(x_k)$、约束函数值 $h(x_k)$ 和雅可比矩阵 $\nabla h(x_k)$。
   (b) 构建二次规划子问题：

$$\begin{aligned} \min_{s} \quad & \nabla f(x_k)^\top s + \frac{1}{2} s^\top B_k s, \\ \text{subject to} \quad & h_i(x_k) + \nabla h_i(x_k)^\top s = 0, \quad i = 1, 2, \dots, m. \end{aligned}$$

(c) 求解二次规划子问题，得到步长 s_k 和拉格朗日乘子 λ_k。

(d) 进行线搜索或信赖域调整，确定步长系数 α_k。

(e) 更新迭代点 $x_{k+1} = x_k + \alpha_k s_k$。

(f) 更新 Hessian 矩阵近似 B_{k+1}，可采用 BFGS 更新等方法：

$$B_{k+1} = B_k + \frac{y_k y_k^\top}{y_k^\top s_k} - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k},$$

其中，$y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$。

(g) 若满足收敛条件 $\|s_k\| < \epsilon$ 且 $\|h(x_{k+1})\| < \epsilon$，则停止迭代。

2.5.2 SQP 方法的理论基础

SQP 方法的理论基础源于泰勒展开和 KKT 条件。通过对目标函数进行二阶泰勒展开和对约束条件进行一阶近似，SQP 构建了一个二次规划子问题，逐步逼近原始问题的最优解。

2.5.3 SQP 方法的性质

- **快速收敛性**：在满足一定正则性条件下，SQP 方法具有二次收敛性。
- **处理约束能力强**：能够有效地处理复杂的非线性约束条件。
- **灵活性高**：可结合多种 Hessian 矩阵近似和步长调整策略，适应不同类型的优化问题。
- **鲁棒性**：在处理非线性和非凸问题时，表现出较好的稳定性和鲁棒性。

2.5.4 SQP 方法的实现细节

Hessian 矩阵的近似 在实际应用中，直接计算和存储 Hessian 矩阵可能代价较高，尤其在高维问题中。常用的 Hessian 矩阵近似方法包括：

- **BFGS 更新**：
  $$B_{k+1} = B_k + \frac{y_k y_k^\top}{y_k^\top s_k} - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k},$$
  其中，$y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$。
- **DFP 更新**：
  $$B_{k+1} = B_k + \frac{y_k y_k^\top}{y_k^\top s_k} - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k}.$$

这些更新公式保证了 Hessian 矩阵近似的正定性，增强了算法的稳定性和收敛性。

步长选择 步长选择是确保算法收敛性的关键步骤。常用的步长选择策略包括：

- **线搜索**：选择一个步长系数 α_k，使得目标函数在更新点处有足够的下降，满足某种下降条件（如 Armijo 条件）。
- **信赖域方法**：通过限制步长的大小，确保步长在模型近似可靠的范围内，同时调整信赖域的大小以平衡模型的可靠性和搜索的效率。

初始点选择 选择合适的初始点对于算法的收敛性和效率具有重要影响。可行的初始点能够加快收敛速度，而不可行的初始点需要算法具备从不可行状态逐步逼近可行域的能力。不可行初始点的牛顿法允许从不可行的起点开始迭代，提高了算法的适用性。

2.5.5 SQP 方法的几何解释

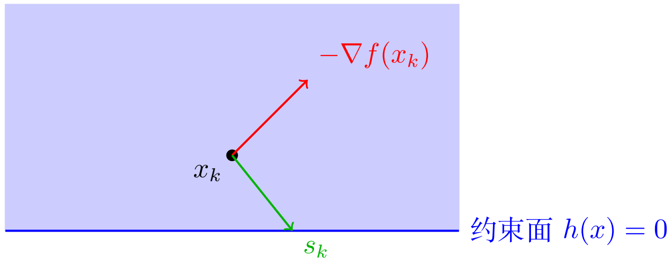

图 1: SQP 方法的几何解释

理解

在 SQP 方法中，每次迭代通过求解一个二次规划子问题，找到一个既满足线性化约束条件又能有效降低目标函数的步长 s_k。这种方法在可行域的切空间内，沿着目标函数下降的方向进行搜索，逐步逼近最优解。

2.5.6 SQP 方法的优缺点

- **优点**：
  - **快速收敛**：在满足一定条件下，具有二次收敛速度。
  - **强大的约束处理能力**：能够处理复杂的非线性约束条件。
  - **灵活性高**：可结合多种 Hessian 矩阵近似和步长调整策略，适应不同类型的优化问题。
  - **鲁棒性**：在处理非线性和非凸问题时，表现出较好的稳定性和鲁棒性。
- **缺点**：
  - **计算成本高**：每次迭代需要求解二次规划子问题，尤其在大规模问题中。
  - **需要良好的初始点**：不适合从远离可行域的初始点开始。
  - **Hessian 矩阵近似的选择和更新可能影响算法性能**：不当的近似可能导致收敛速度下降或收敛失败。

2.5.7 SQP 方法的收敛性

定理 2.2 (SQP 方法的局部收敛性). *在满足以下条件下，SQP 方法的迭代序列 $\{x_k\}$ 将以二次速度收敛到最优解 x^*：*

- *目标函数 $f(x)$ 和约束函数 $h(x)$ 在最优解附近具有二阶连续可微性。*
- *在最优解处，KKT 条件成立且满足二阶充分条件，即拉格朗日函数的 Hessian 在约束的切空间上是正定的。*
- *初始点 x_0 足够接近最优解 x^*。*

理解

该定理表明，SQP 方法在满足基本光滑性和正定性条件下，能够快速收敛到最优解。局部二次收敛性意味着误差随着迭代次数的增加而以平方级别减少，大大提高了算法的效率。

3 牛顿法在等式约束优化中的应用

3.1 牛顿法的基本思想

牛顿法是一种基于二阶导数信息的迭代优化方法，具有快速收敛的特点。其核心思想是利用目标函数的二阶泰勒展开，在当前点附近构建一个二次近似模型，并通过最小化该模型来确定更新步长。

在等式约束优化中，牛顿法需要同时考虑目标函数和约束条件，确保每次迭代既能有效降低目标函数，又能保持在可行域内。

3.2 牛顿步和牛顿减量

定义 3.1 (牛顿步)**.** 在无约束优化中，牛顿步 s 定义为：

$$s = -[\nabla^2 f(x)]^{-1}\nabla f(x).$$

在等式约束优化中，牛顿步需要满足约束的线性化条件，即步长 s 不仅要沿着目标函数的下降方向，还要保持在约束的可行方向上。

定义 3.2 (牛顿减量)**.** 牛顿减量 $\lambda(x)$ 定义为：

$$\lambda(x) = \sqrt{\nabla f(x)^\top[\nabla^2 f(x)]^{-1}\nabla f(x)}.$$

它用于衡量当前迭代点与最优解的距离。

理解

牛顿减量反映了二次模型预测的目标函数下降量，与实际的下降量比较，可以评估模型的准确性和步长的有效性。较小的牛顿减量表明当前位置已经接近最优解，迭代可以终止。

3.3 等式约束下的牛顿法

在等式约束优化中，牛顿法通过构建并求解 KKT 系统，确保步长既能有效降低目标函数，又能满足约束条件。

3.3.1 线性化约束条件

在当前迭代点 x_k，对约束函数进行一阶近似：

$$h(x_k + s) \approx h(x_k) + \nabla h(x_k)^\top s = 0.$$

3.3.2 构建 KKT 系统

定理 3.1 (等式约束下的牛顿步). *构建以下线性方程组（KKT 系统）求解牛顿步 s 和拉格朗日乘子 λ:*

$$\begin{cases} \nabla^2 f(x_k)s + \nabla f(x_k) - \nabla h(x_k)\lambda = 0, \\ \nabla h(x_k)^\top s + h(x_k) = 0. \end{cases}$$

3.3.3 矩阵形式

将上述方程组写成矩阵形式，方便使用线性代数方法求解：

$$\begin{bmatrix} \nabla^2 f(x_k) & -\nabla h(x_k) \\ \nabla h(x_k)^\top & 0 \end{bmatrix} \begin{bmatrix} s \\ \lambda \end{bmatrix} = \begin{bmatrix} -\nabla f(x_k) \\ -h(x_k) \end{bmatrix}.$$

理解

该线性系统包含了优化目标的二阶信息和约束条件的线性化，是求解牛顿步的核心。通过求解 KKT 系统，可以同时获得步长 s 和拉格朗日乘子 λ，从而更新迭代点并维持约束条件。

3.4 牛顿法的算法步骤

1. 初始化 x_0，设置容差 $\epsilon > 0$。
2. 对于 $k = 0, 1, 2, \ldots$：
   (a) 计算梯度 $\nabla f(x_k)$ 和 Hessian 矩阵 $\nabla^2 f(x_k)$。
   (b) 计算约束函数值 $h(x_k)$ 和雅可比矩阵 $\nabla h(x_k)$。
   (c) 构建并求解 KKT 系统，得到牛顿步 s_k 和拉格朗日乘子 λ_k。
   (d) 更新迭代点 $x_{k+1} = x_k + s_k$。
   (e) 若 $\|s_k\| < \epsilon$ 且 $\|h(x_{k+1})\| < \epsilon$，则停止迭代。

3.5 几何解释

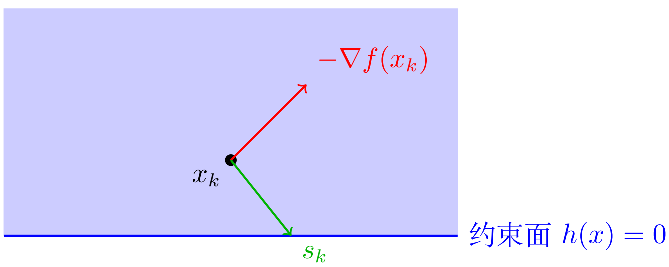

图 2: 等式约束优化的几何解释

理解

牛顿步 s_k 在可行域的切空间内，既满足线性化的约束条件，又沿着目标函数下降的方向，实现了在约束条件下的最优更新。几何上，步长 s_k 是沿着目标函数最速下降方向，同时保持在可行方向上的方向调整。

4 不可行初始点的牛顿法

4.1 动机

在实际应用中，找到一个满足约束条件的初始点可能并不容易。不可行初始点的牛顿法允许从不可行的起点开始迭代，逐步逼近可行域和最优解。这种方法增加了算法的适用性，特别是在无法轻易找到可行初始点的情况下。

4.2 算法步骤

1. 初始化 x_0（可能不可行），设置容差 $\epsilon > 0$。
2. 对于 $k = 0, 1, 2, \ldots$：
   (a) 计算梯度 $\nabla f(x_k)$、Hessian 矩阵 $\nabla^2 f(x_k)$、约束函数值 $h(x_k)$ 和雅可比矩阵 $\nabla h(x_k)$。
   (b) 构建并求解增广 KKT 系统：
   $$\begin{bmatrix} \nabla^2 f(x_k) & -\nabla h(x_k) \\ \nabla h(x_k)^\top & 0 \end{bmatrix} \begin{bmatrix} s_k \\ \lambda_k \end{bmatrix} = \begin{bmatrix} -\nabla f(x_k) \\ -h(x_k) \end{bmatrix}.$$
   (c) 计算步长系数 α_k，可采用线搜索或信赖域方法。
   (d) 更新迭代点 $x_{k+1} = x_k + \alpha_k s_k$。
   (e) 若 $\|\nabla f(x_k) + \nabla h(x_k)\lambda_k\| < \epsilon$ 且 $\|h(x_k)\| < \epsilon$，则停止迭代。

4.3 收敛性分析

定理 4.1 (收敛性)**.** *在以下条件下，不可行初始点的牛顿法具有局部二次收敛性：*

- *目标函数 $f(x)$ 和约束函数 $h(x)$ 在最优解附近具有二阶连续可微性。*
- *在最优解处，Hessian 矩阵 $\nabla^2 f(x^*)$ 在约束的切空间上是正定的（即满足二阶充分条件）。*
- *初始点 x_0 足够接近最优解 x^*。*

理解

当迭代点足够接近最优解时，算法的收敛速度将加快，实现二次收敛，即误差的平方级减少。这意味着在最优解附近，牛顿法能够迅速逼近最优解。

5 牛顿法的性质与解释

5.1 仿射不变性

定理 5.1 (仿射不变性)**.** *牛顿法对于变量的仿射变换是不变的，即在不同的坐标系下，牛顿法的迭代轨迹通过仿射变换联系，其收敛性质保持不变。*

证明. 考虑仿射变换 $x = Tz + c$，其中 T 是可逆矩阵，c 是常数向量。变换后的优化问题为：

$$\min_z \; f(Tz + c).$$

变换后的梯度和 Hessian 矩阵为：

$$\nabla_z \tilde{f}(z) = T^\top \nabla_x f(x), \quad \nabla_z^2 \tilde{f}(z) = T^\top \nabla_x^2 f(x) T.$$

牛顿步在新变量下为：

$$\tilde{s} = -[\nabla_z^2 \tilde{f}(z)]^{-1} \nabla_z \tilde{f}(z) = -(T^\top \nabla_x^2 f(x) T)^{-1} T^\top \nabla_x f(x) = T^{-1}(-[\nabla_x^2 f(x)]^{-1} \nabla_x f(x)) = T^{-1} s.$$

因此，新的牛顿步与原牛顿步通过仿射变换联系，故牛顿法具有仿射不变性。 □

理解

仿射不变性意味着牛顿法的收敛性质不依赖于变量的尺度和坐标系的选择，这使得牛顿法在不同尺度的优化问题中具有良好的鲁棒性。

5.2 收敛性

牛顿法的收敛性分析通常依赖于目标函数和约束函数的光滑性，以及初始点的选择。

定理 5.2 (牛顿法的收敛性). *在以下条件下，牛顿法的迭代序列 $\{x_k\}$ 将以二次速度收敛到最优解 x^*：*

- *目标函数 $f(x)$ 和约束函数 $h(x)$ 在最优解附近具有二阶连续可微性。*
- *在最优解处，满足 KKT 条件，并且拉格朗日函数的 Hessian 在约束的切空间上是正定的（即满足二阶充分条件）。*
- *初始点 x_0 足够接近最优解 x^*。*

此外，存在常数 $C > 0$，使得对于足够大的 k，

$$\|x_{k+1} - x^*\| \le C \|x_k - x^*\|^2.$$

理解

该定理表明，牛顿法在满足基本光滑性和正定性条件下，具有二次收敛速度。这意味着每次迭代后，误差的数量级将平方级减少，从而实现快速逼近最优解。

5.3 牛顿减量的意义

理解

牛顿减量 $\lambda(x)$ 可以视为当前点到最优解的“距离”测度。它衡量了在当前点沿着牛顿步方向的目标函数下降量，较小的牛顿减量表明当前位置已经接近最优解，迭代可以终止。

5.4 牛顿法的优势

- **收敛速度快**：在二次可微函数下，牛顿法具有二次收敛性。

- **利用二阶信息**：通过 Hessian 矩阵，牛顿法能够更准确地捕捉目标函数的曲率信息。
- **适用于等式约束**：通过构建 KKT 系统，牛顿法可以自然地处理等式约束优化问题。
- **仿射不变性**：牛顿法的仿射不变性使其在不同坐标系下具有一致的性能。

5.5 牛顿法的局限性

尽管牛顿法具有诸多优点，但在实际应用中也存在一些局限性：

- **计算成本高**：需要计算和存储 Hessian 矩阵，对于大规模问题，计算和内存需求可能过高。
- **Hessian 矩阵的正定性**：牛顿法要求 Hessian 矩阵在最优解附近是正定的，否则可能导致步长方向不正确。
- **敏感于初始点**：选择不合适的初始点可能导致收敛速度慢或陷入局部极小值。

6 KKT 系统的求解方法

求解 KKT 系统是等式约束优化中牛顿法的核心步骤。以下介绍几种常用的求解方法。

6.1 直接求解

对于小规模问题，可以直接使用线性代数方法求解 KKT 系统。常用的方法包括：

- **LU 分解**：将 KKT 系统的矩阵分解为下三角矩阵和上三角矩阵，进而求解线性方程组。
- **Cholesky 分解**：适用于对称正定矩阵，通过分解为下三角矩阵及其转置来简化求解过程。

然而，对于大规模问题，直接求解方法的计算成本和内存需求可能过高，因此需要更高效的方法。

6.2 分块消元法

利用 KKT 系统的结构，可以采用分块消元法进行求解，降低计算复杂度。具体步骤如下：

1. 将 KKT 系统表示为：
$$\begin{bmatrix} H & -A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} s \\ \lambda \end{bmatrix} = \begin{bmatrix} -g \\ -h \end{bmatrix},$$
其中，$H = \nabla^2 f(x_k)$，$A = \nabla h(x_k)$，$g = \nabla f(x_k)$，$h = h(x_k)$。
2. 通过消元法，将 λ 从方程组中消去。首先，从第二个方程解得：
$$\lambda = (AH^{-1}A^\top)^{-1}AH^{-1}g + (AH^{-1}A^\top)^{-1}h.$$
3. 将 λ 代入第一个方程，得到仅关于 s 的方程：
$$s = -H^{-1}g + H^{-1}A^\top(AH^{-1}A^\top)^{-1}(h + AH^{-1}g).$$

这种方法减少了求解系统的规模，但仍需计算逆矩阵，对于大规模问题依然不够高效。

6.3 迭代求解方法

对于大规模 KKT 系统，可以采用迭代求解方法，如共轭梯度法（Conjugate Gradient, CG），特别是其变种 Steihaug 方法。这些方法不需要显式存储整个矩阵，只需通过矩阵-向量乘法即可。

6.4 序列二次规划（SQP）方法中的 KKT 系统求解

在 SQP 方法中，每次迭代都需要构建并求解一个二次规划子问题的 KKT 系统。为了提高效率，常用以下技术：

- **矩阵分解与更新**：利用前一次迭代的矩阵分解结果，进行快速更新。
- **活跃集方法**：仅考虑当前活跃的约束，减少系统规模。
- **内点法**：通过引入障碍函数，避免边界问题，提高稳定性。

这些技术结合使用，可以显著提高大规模优化问题中 SQP 方法的求解效率。

理解

对于大规模问题，直接求解 KKT 系统可能不切实际，此时可以采用分布式方法或迭代求解器，如共轭梯度法，来高效地求解。

7 牛顿法的扩展

除了标准的牛顿法，针对不同类型的优化问题，还可以对牛顿法进行各种扩展和改进，以提高其适用性和效率。

7.1 信赖域牛顿法

信赖域牛顿法（Trust Region Newton Method）通过限制步长的大小，确保步长在模型近似可靠的范围内。其核心思想是在每次迭代中构建一个信赖域内的二次模型，并在该区域内求解最优化步长。

7.1.1 信赖域的定义

信赖域是一个以当前迭代点为中心的球体，其半径表示模型近似的可信程度。通过调整信赖域的大小，可以平衡模型的可靠性和搜索的效率。

7.1.2 信赖域牛顿法的算法步骤

1. 初始化 x_0，设置初始信赖域半径 $\Delta_0 > 0$，以及参数 $0 < \eta_1 < \eta_2 < 1$，$0 < \gamma_0 < 1 < \gamma_1$。
2. 对于 $k = 0, 1, 2, \ldots$：
   (a) 构建二次模型：
   $$m_k(s) = f(x_k) + \nabla f(x_k)^\top s + \frac{1}{2} s^\top B_k s.$$
   (b) 在信赖域内求解子问题：
   $$\min_s \ m_k(s), \quad \text{subject to } \|s\| \leq \Delta_k.$$
   (c) 计算实际下降量与预测下降量的比值 ρ_k：
   $$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(0) - m_k(s_k)}.$$
   (d) 根据 ρ_k 更新 x_{k+1} 和信赖域半径 Δ_{k+1}：
   - 如果 $\rho_k \geq \eta_2$，则接受步长：$x_{k+1} = x_k + s_k$，并增大信赖域半径：$\Delta_{k+1} = \gamma_1 \Delta_k$。
   - 如果 $\eta_1 \leq \rho_k < \eta_2$，则接受步长：$x_{k+1} = x_k + s_k$，并保持信赖域半径不变：$\Delta_{k+1} = \Delta_k$。
   - 如果 $\rho_k < \eta_1$，则拒绝步长：$x_{k+1} = x_k$，并减小信赖域半径：$\Delta_{k+1} = \gamma_0 \Delta_k$。

   (e) 若满足停止条件 $\|\nabla f(x_k)\| \leq \epsilon$，则停止迭代。

7.1.3 信赖域牛顿法的优势

- **提高稳定性**：通过限制步长，避免步长过大导致的模型失效。
- **自适应性强**：根据信赖域半径的调整，自动适应模型的近似程度。
- **收敛性保证**：在一定条件下，信赖域牛顿法能够保证全局收敛。

7.1.4 信赖域牛顿法的收敛性

定理 7.1 (信赖域牛顿法的全局收敛性)**.** *假设目标函数 $f(x)$ 连续可微，且其梯度 $\nabla f(x)$ 是 Lipschitz 连续的，即存在常数 $L>0$，使得对于任意 x, y：*

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|.$$

则信赖域牛顿法生成的迭代序列 $\{x_k\}$ 满足：

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0.$$

理解

该定理表明，信赖域牛顿法在满足基本光滑性条件下，能够确保迭代序列的梯度趋近于零，最终逼近最优解。

7.2 信赖域牛顿法的应用

信赖域牛顿法广泛应用于工程设计、经济规划和科学计算等领域，尤其适用于目标函数不可精确线性化或梯度信息不可靠的情况。其自适应的步长调整机制和对模型可信度的评估，使得算法在搜索过程中能够灵活调整步长，既保证了收敛性，又提高了效率。

8 数值实验

为了验证上述方法的有效性，本文通过一个简单的等式约束优化问题进行数值实验，并对比不同方法的表现。

8.1 问题示例

考虑以下等式约束优化问题：

$$\begin{aligned} \min_{x\in\mathbb{R}^2} \quad & f(x) = x_1^2 + x_2^2, \\ \text{subject to} \quad & h(x) = x_1 + x_2 - 1 = 0. \end{aligned}$$

8.2 求解步骤

1. 构建拉格朗日函数：

$$\mathcal{L}(x, \lambda) = x_1^2 + x_2^2 - \lambda(x_1 + x_2 - 1).$$

2. 计算梯度和 Hessian：

$$\nabla f(x) = \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix}, \quad \nabla^2 f(x) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}.$$

3. 计算约束雅可比矩阵：

$$\nabla h(x) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

4. 构建 KKT 系统并求解：

$$\begin{bmatrix} 2 & 0 & -1 \\ 0 & 2 & -1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

5. 求解线性方程组，得到步长 $s = \left(\frac{1}{2}, \frac{1}{2}\right)$ 和拉格朗日乘子 $\lambda = 1$。

6. 更新迭代点 $x_1 = (0,0) + \left(\frac{1}{2}, \frac{1}{2}\right) = \left(\frac{1}{2}, \frac{1}{2}\right)$。

7. 检查收敛条件：

$$\|s\| = \sqrt{\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2} = \frac{\sqrt{2}}{2} \approx 0.707 > \epsilon.$$

8. 第二次迭代：

$$\nabla f(x_1) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad h(x_1) = 0.$$

构建 KKT 系统：

$$\begin{bmatrix} 2 & 0 & -1 \\ 0 & 2 & -1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \lambda \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ 0 \end{bmatrix}.$$

求解得到 $s = (0,0)$，$\lambda = 1$。更新迭代点 $x_2 = \left(\frac{1}{2}, \frac{1}{2}\right)$。

9. 检查收敛条件：

$$\|s\| = 0 < \epsilon, \quad \|h(x_2)\| = 0 < \epsilon.$$

因此，算法停止，得到最优解 $x^* = \left(\frac{1}{2}, \frac{1}{2}\right)$，对应的目标函数值为 $f(x^*) = \frac{1}{2}$。

8.3 结果分析

通过上述迭代过程，牛顿法成功找到最优解 $x^* = \left(\frac{1}{2}, \frac{1}{2}\right)$，满足约束条件 $x_1 + x_2 = 1$，且达到目标函数的最小值。这表明牛顿法在等式约束优化问题中的有效性。

8.4 与其他方法的对比

通过与消元法、Null-space 方法和 SQP 方法的对比，可以发现牛顿法在处理等式约束优化问题时，具备以下优势：

- **收敛速度快**：在满足条件下，牛顿法具有二次收敛性，收敛速度远快于线性收敛的梯度下降法。
- **无需显式消元**：与消元法不同，牛顿法无需显式地消去变量，适用于更广泛的约束条件。
- **处理复杂约束**：相比 Null-space 方法，牛顿法更适合处理复杂和高度非线性的约束条件。

9 结论

本文系统地介绍了等式约束优化问题的主要求解方法，深入阐述了牛顿法在此类问题中的应用。通过定义牛顿步和牛顿减量，解释了它们在优化过程中的作用和意义。不可行初始点的牛顿法为实际问题的求解提供了灵活性，使得算法具有更广泛的适用性。序列二次规划（SQP）方法作为一种强大的迭代优化算法，通过解决一系列二次规划子问题，结合二阶信息和约束条件，实现了高效的优化过程。牛顿法的二次收敛性和仿射不变性，使其在优化算法中占有重要地位。

未来的研究可以进一步探索 SQP 方法在大规模优化问题中的高效实现，以及结合现代计算技术提升其性能。此外，针对非等式约束（如不等式约束）的扩展方法也值得深入研究，以满足更多实际应用的需求。

参考文献

[1] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, 2006.

[2] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[3] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1999.

[4] D. G. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley, 1984.

[5] S. J. Wright and J. Nocedal, "Numerical optimization," *Springer Science*, vol. 35, 1999.

[6] R. Fletcher, *Practical Methods of Optimization*, John Wiley & Sons, 1987.

[7] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, 1951, pp. 481–492.

[8] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, Academic Press, 1981.

[9] A. Forsgren, P. E. Gill, and M. H. Wright, "Interior methods for nonlinear optimization," *SIAM Review*, vol. 44, no. 4, pp. 525–597, 2002.

[10] J. Nocedal, "Updating Quasi-Newton Matrices with Limited Storage," *Mathematics of Computation*, vol. 35, no. 151, pp. 773–782, 1980.