# 第十一讲　约束优化的罚函数法

* 序列无约束最小化技术
* 罚函数法
* 外点罚函数法
* 二次罚函数法
* 精确L1罚函数法
* 内点罚函数法
* 对数障碍函数法

本讲开始学习约束优化问题的优化算法。约束优化的一些重要方法常用用一系列子问题代替原始问题，其中约束由添加到目标的项表示。在这里，我们首先讲解约束优化的罚函数法，包括外点罚函数法、内点罚函数法和乘子罚函数法。其基本思想将约束最小化问题转化为序列无约束最小化问题。例如，二次罚函数将每个约束违反的平方的倍数加到目标上。由于其简单性和直观的吸引力，这种方法在实践中经常使用，尽管它有一些重要的缺点。非光滑精确罚方法用单个无约束问题（而不是序列）代替原始约束问题。使用这些惩罚函数，我们经常可以通过执行单个无约束最小化来找到解决方案，但是非光滑性可能产生复杂性。这种类型的一个流行的函数是L1罚函数。另一种精确罚方法是乘子法或增广拉格朗日法，其中使用显式拉格朗日乘子估计来避免二次罚函数固有的病态条件。另外，在对数障碍罚函数方法中使用了一种相关的方法，其中对数项防止可行迭代点过于靠近可行区域的边界，其是非线性规划内点法的基础。

## **一、内点罚函数法**

**基本思想**：迭代总是从内点出发，并保持在可行域内部进行搜索。

**模型：**

是连续函数

**障碍函数**：

其中是很小的正数，定义在可行域内部，它满足两个条件：

1. 是连续函数；
2. 当点趋向于可行域边界时，

**两种重要的形式：**

倒数障碍函数：

对数障碍函数：**

**其中为严格单调减且趋于0的障碍因子数列。**

**算法步骤：**

1. 给定初始点，允许误差，初始参数，缩小系数，置。
2. 以为初始点，求解下列问题

设其极小点为。

1. **，则停止计算，得到点；否则，令，置，（注：用得到的作为新的初始点）返回2。

**收敛性分析（引理1、引理2 定理）**

**引理1**：对于由内点法所产生的序列,总有

**证明：**

（1）由和知

**

因为是的极小点，所以对，

有

（2）和分别使，取极小

所以

**

（3）由，得

**引理2：**

是问题（*A*）的一个最优解，则对，有

其中问题（*A*）为

**证明：**

由的定义及

所以

**定理**：

设是由内点法产生的一个序列，则的任何收敛子序列的极限都是原问题的最优解。

**证明**：

设是的任一收敛子序列，且有极限，

因为在上连续所以

令为原问题的最优值。

由引理1,2知是非增的，且以为下界的数列。

由*f*的连续性知，对，，当与充分接近时有

由*G*的定义知，对，有

特别的，

因为所以对同一个，，有

所以

由**引理2，**。因为*f*连续，所以

**求初始内点的迭代步骤**：

1、任取（如取），置。

2、令

3、若，停止计算；否则，转4。

4、构造函数

记

5、 以为初始点，在域内，求障碍函数的极小点：

得，转6。

6、令0<，置，转2。

**内点罚函数法优点：**

迭代总在可行域内进行，每一个中间结果都是可行解，可以作为近似解。

**内点罚函数法缺点：**

选择初始可行解较困难，且**只适用于含不等式约束非线性规划问题**。

（注：否则可行点都是边界点，都加上无穷大惩罚，惩罚也就失去意义了）

## 二、障碍罚函数法

**模型**

凸优化问题

**转化**

令，我们可以重新写一下上面的问题如下：

s.t.



**思想：**用*C*的障碍函数逼近，这样可以避免*C*的边界，使问题满足牛顿法。

**对数障碍函数**

假定是凸函数并且二阶可导，那么下面函数

对于非空集合被称为对数障碍函数。

**近似的原始问题为**：

这里。

**定义：**

**中心路径**

设，考虑上面的障碍问题

令为上述问题的解，中心路径为集合。

在合适的条件下，当时，这个集合在上是平滑的路径，我们有**,其中是原问题的一个解**。

**障碍函数的KKT条件是**

**

**原始问题的KKT条件是**

**

其中

前面两组KKT条件服从

**（对偶间隙）**

这是一个**有用的终止条件**。

**算法（V.0→V.1→V.2一步步的改进）**

**障碍方法V.0**

对于，取并且解决下列问题

从而可以得到

这不是一个好的方法，因为障碍函数太难了以至于很难解出来。

上面这个方法旨在于找到一个靠近中心路径末尾的点。一个更好的方法是产生一些点沿着中心路径。

**障碍方法V.1**

解决一系列障碍问题

s.t.

对增加的值。

取且令

求解对的障碍问题得到

当时

取

在解决障碍问题，使用初始化牛顿方法生成结束。

共同更新对。

定心步骤：解决障碍问题的步骤。被称为对数障碍函数

**考虑事项**

的选择：如果太小，那么可能需要许多外部迭代；如果太大，那么牛顿方法（每个定心步骤）可能需要许多迭代来收敛。

的选择：如果太小，那么可能需要许多外部迭代；如果太大，那么第一牛顿解可能需要许多迭代来计算。

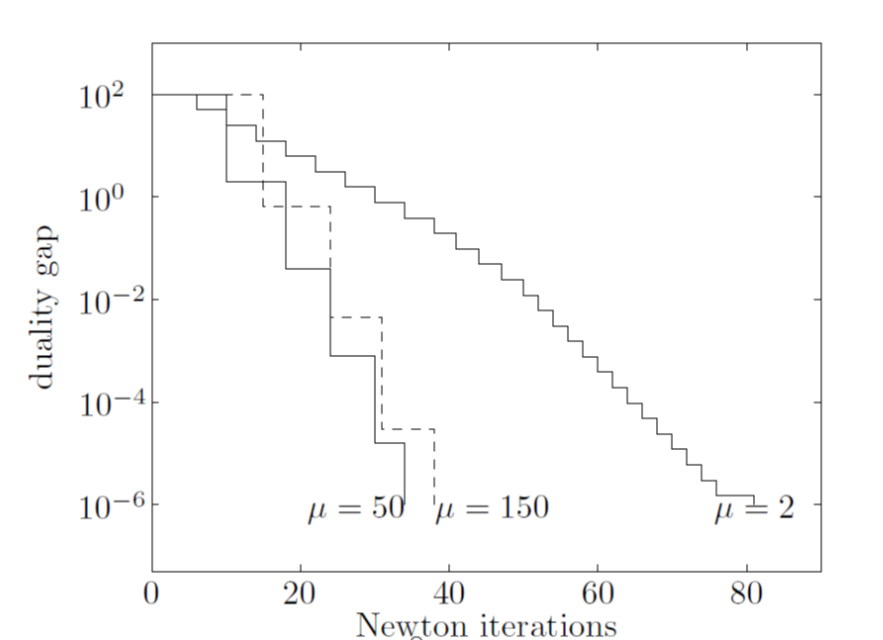
所幸的是，在实践中，屏障方法的性能常常是对和的选择。

这些参数的适当范围取决于尺度。

**例1：**

****

其中

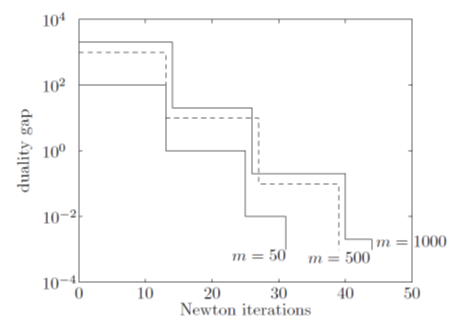


**例2**

****

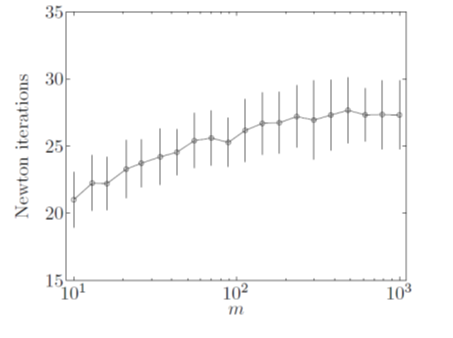
其中n=2m。

探讨：牛顿迭代次数和变量个数n以及等式约束数目m之间函数关系。



表明：在每种情况下都可以看到大致的线性收敛，

所需的牛顿阶数，随的增长非常缓慢。



尽管问题规模增加为初始规模100倍，所需要的牛顿迭代次数增量却很小，仅从21附近增加到27左右。

注意：单个牛顿步骤的成本取决于问题的大小。

**收敛性分析**

**定理**：K步中心步后的势垒法满足

换句话说，为了达到期望的级精度，我们需要

次包括初始点中心步骤在内的中心点步骤。

**障碍法V .2**

前面的算法生成的点正好在中心路径上。然而，中央路径只是一个末端。没有必要确切地解决每个问题。

取且设

求解对的障碍问题得到

当时

取

在解决障碍问题，使用初始化牛顿方法生成

**可行性方法(求可行点）**

**定义**：点满足

，称为严格可行点。

如何找到一个可行点？通过解

s.t.

.

我们可以障碍方法应用到上述问题，因为它很容易找到一个严格可行的起点。

（说明：取）

注意，我们不需要高精度地解决这个问题。一旦我们找到一个可行的,其中s<0，我们就可以早点终止。

**另一种方法是求解问题**

s.t.

, .

以前的s是所有不等式的最大不可行点, 现在每个不等式都有自己的不可行变量

**优点**：当原始系统不可行时，上述问题的解决将是有益的。s的非零项将告诉我们哪些约束不能满足。

## 三、外点罚函数法

**思想：**考虑一个序列无约束优化问题其目标函数由约束优化问题的目标函数加上每个约束的一个附加项，如当前点x违反该约束时，该附加项为正，否则为零。大多数方法定义了这样的惩罚函数序列，其中违反约束的惩罚项乘以正系数。通过增大这个系数，我们更严厉地惩罚约束违反，从而迫使惩罚函数的最小值更接近约束问题的可行区域。（迭代点在可行域外，对不在可行域内，加大惩罚）

**模型：**

（*A*）

其中

**

**惩罚项**：

其中

函数

其中。（需要满足在约束里为0，在约束外为正）

### 1 光滑罚函数

这种类型的最简单的惩罚函数是二次惩罚函数，其中惩罚项是约束违反的平方。

其中是很大的正数。

**算法步骤：**

1. 给定初始点,初始罚因子放大系数
2. 以为初始点，求解无约束问题

设其极小点为。

3.,则停止计算，得到点;否则，令，置,返回2。

**收敛性分析**

**引理1** 对于由外点法所产生的序列，总有

证明：（1）由和知

=

因为是的极小点，所以

（2）和分别使取极小

因为 （）

（3）由（），得

**引理2** 设是问题（*A*）的一个最优解，则对

其中（*A*）为

**证明：**因为是问题（*A*）的一个最优解，所以有

1. 因为是的极小点

所以

又因为

1. 所以

**定理** 设是由外点法产生的一个序列，则的任何收敛子序列的极限都是问题（*A*）的最优解，若，，具有连续的一阶偏导数，且在处

线性无关，则

**证明**：设是的任一收敛子序列，且有极限

因为在上连续，所以

令为问题（*A*）的最优值。

由引理1,2，知是非减的，且以为上界的数列

所以

因为

所以

又因为

所以

又因为为连续函数

所以是可行解

由引理2知，

所以为最优解。

因为是的最优解，所以有

当时，有，则存在*K*，当*k>K*时，有，因此

由于是问题（*A*）的最优解，所以有

所以

**外点罚函数的优点：**

函数是在整个空间内进行优化，初始点可以任意选择，且外点法也可用于非凸规划的最优化。

**缺点**：

1. 惩罚项的二阶偏导数一般不存在；
2. 外点法的中间结果不是可行解，不能作为近似解；
3. 当点接近最优解时，罚因子很大可能使罚函数性质变坏，使搜索产生极大困难。

**例1**：

其罚函数为

*Hesse*矩阵为

条件数=

**例2：**

若取，则，没有最小点。

若取，则，有极小点

**说明：根据目标函数的不同，可能需要不同的惩罚项。**

### 2．非光滑罚函数 NONSMOOTH PENALTY FUNCTIONS

有些罚函数是精确的，这意味着，对于它们的罚参数的某些选择，关于x的单个极小化可以得到非线性规划问题的精确解。这个属性是理想的，因为它使得罚函数法的性能较少依赖于更新惩罚参数的策略。但是，前面二次罚函数是不精确的，因为原问题的最小值一般不同于任何正的罚参数下的非线性规划的解。在这一节中，我们讨论非光滑精确罚函数，这在许多实际情况下被证明是精确的有用的。

非线性规划问题的一个常用的非光滑罚函数是由L1范数定义的L1罚函数，如下：

(17.22)

这里，[y]—=max { 0，-y}。它的名称来源于惩罚项是违反约束的L1范数的μ倍。注意，由于绝对值和Max函数的存在，L1罚函数是不可微的。下面的结果建立了L1罚函数的精确性。



**定理(17.3):**

假设 是非线性规划问题(17.6)的一个严格局部解，且满足（定理12.1）一阶必要条件，其中拉格朗日乘子为 那么对所有,其中来说 是的局部极小点。如果，二阶必要条件（定理12.6）成立且 ,那么是的严格局部极小点.

（定理12.1：)假设是问题12.6的局部解，函数连续可微，LICQ在成立。有拉格朗日乘子,由构成，其中，以下条件在处成立



(定理12.6)

假设对一些可行点，这有拉格朗日乘子，满足KKT条件，假设

那么对于问题17.6是严格局部解.

粗略地说，在非线性规划的最优解x\*处，对于任何进入不可行区域的移动都受到足够大的惩罚，使得惩罚函数增加到大于φ1(x\*;μ)=f(x\*)的值，从而迫使φ1(·μ)的最小值位于x\*。

例子（17.2）：

考虑下列单变量问题：

它的解为。我们有

正如我们看到的图17.3，当时，罚函数在处有最小值，但是当时，罚函数是单调递增函数。



由于罚函数法通过直接最小化罚函数，所以我们需要对φ1的稳定点进行刻画。尽管φ1是不可微的，但它具有沿任意方向的方向导数D(φ1(x;μ)；p)。下面给出其定义和实例。

**定义(17.1)：**

一点是罚函数的不动点，如果

 （17.26）

对于所有的。同样的，是不可行方法

（17.27）

的不动点,当对于所有的。如果一点对于问题17.6是不可行点但对于不可行方法*h*是不动点,我们称它为不可行不动点。

对于例17.2的方程，对于我们有

证明：

，



当时，我们有，对于所有的。

下面的结果是定理17.3的补充，证明了在一定的假设下，的稳定点对应于约束优化问题(17.6)的KKT点。

**定理(17.4):**

假设是罚函数的一个不动点，对所有的大于一个确定的极限。那么，如果对于非线性规划(17.6)是可行的，对于(17.6)它满足KKT条件。如果对于(17.6)是不可行的，它就是不可行不动点。

**证明** ：首先假设是可行的。我们由和的定义（17.22）有



这里的集合=

**（验证：）**

考虑定义12.3中的线性化可行方向集的任意方向。

由的性质我们有



因此由的稳定性假设，我们有

对所有的

我们现在运用**Farkas引理**：



其中，对所有的。正如我们之前注意的（看定理12.1和12.35），这个表达式能够推出KKT条件成立。

定理12.1：假设是问题12.6的局部解，函数连续可微，LICQ在成立。有拉格朗日乘子,由构成，其中，以下条件在处成立



特别的，非积极拉格朗日乘子为零，我们可以省略，重新写（12.34a）这个条件



**关于不可行点的证明：**

由已知条件，

****

**例17.3**

再次考虑问题17.3，它的罚函数是

图17.4刻画了函数,它的极小值是(17.3)的解。事实上，根据定理17.3，我们可以发现对于所有的，的极小值就是。轮廓上的尖角暗示着沿着圆圈边界的不平滑性，这个圆圈是被定义的。



在上述结果的启发下，我们下面提出基于L1罚函数的算法框架。

**框架17.2（经典的惩罚算法）**

给定，公差，起始点为;

找到的近似极小值点，*k*=0,1,2,…,起始点为；

如果,则在近似解处停止；

否则选择新的惩罚参数;

选择新的起始点;

结束。

由于函数的非光滑性，使的最小化变得困难。然而，正如下面讨论的，我们深刻理解如何使用类似于SQP方法的平滑化模型来计算最小化问题。

更新惩罚参数的最简单方案是，如果当前点产生的最小值不在允许误差内，则将其增加常数倍（例如5或10）。这个方案在实践中有时很好，但也可能是低效的。如果初始罚参数太小，则可能需要循环17.2算法的多个周期来达到适当的值。此外，在这些初始循环中，迭代点可能会远离最优解x\* 。在这种情况下，最小化应该提前终止，并且可能应该重置为先前的迭代点。另一方面，如果过大，则罚函数将难以最小化，可能需要大量的迭代。我们下面回到选择惩罚参数的问题。

**一个特定惩罚方法**

正如我们已经注意到的，是非光滑的，它的梯度在任何点*x*处都不能定义对于。与其使用不可微优化技术，我们更喜欢考虑该函数中不可微性特殊性质的技术。如同本书第一部分所讨论的无约束优化的算法一样，我们通过形成该函数的简化模型并寻求该模型的极小值来逐步获得的极小值。这里，该模型可以通过将线性约束和用二次函数代替非线性规划目标*f*来定义：

**

这里的是一个对称矩阵,它通常包含*f*和的二次导数，。模型是不光滑的，但是我们可以用公式表示的最小化问题作为一个平滑的二次规划问题，通过介绍人工变量和,如下面所示：

**(17.31)**

这个子问题可以用标准的二次规划求解器来求解。即使增加了p型无穷小的“盒形”信赖域约束，它仍然是一个二次规划。这种最小化φ1的方法与序列二次规划（SQP）密切相关。

惩罚参数的选择和更新对迭代的成功至关重要。我们提到一种简单（但不总是有效的）的方法是选择一个初始值并重复增加它，直到达到可行性。在该方法的一些变型中，在每次迭代中选择惩罚参数，使得>，其中是在处计算的拉格朗日乘子的估计。我们基于定理17.2，它表明在解*x*的邻域中，一个好的选择是设置适度大于。这种策略并不总是成功的，因为乘数估计可能不精确，并且在任何情况下都不能提供远离解的的良好适当值。

选择适当的值的困难导致非光滑惩罚方法在20世纪90年代不受支持，并刺激了滤波方法的发展，这不需要选择惩罚参数(参见第15.4节)。然而，近年来，惩罚方法重新引起了人们的兴趣，部分原因是因为它们处理退化问题的能力。用于更新惩罚参数的新方法似乎在很大程度上克服了与选择相关的困难，至少对于某些特定的实现方法是这样（参见算法18.5）。

对于的极小值，也应慎重考虑起始点的选择。如果当前周期的惩罚参数是适当的，那么在算法向可行性发展的意义上，我们可以将设置为在该周期上获得的的最小。否则，我们可能希望从较早的周期恢复初始点。

**一类非光滑罚函数方法**

精确非光滑罚函数可以用范数以外的范数来定义。我们可以写为

其中是任意向量范数，并且所有的等式约束和不等式约束分别被分组在向量函数和中。框架17.2适用于任何这些惩罚函数；我们简单地重新定义不可行性方法，如。 **(17.32)**在实际应用中最常用的范数是和（非平方），很容易找到类似于（17.31）范数的公式。

函数所描述的理论性质扩展到一般情况（17.32）。在定理17.3中，我们将不等式（17.23）替换为

这里的是的对偶范数，定义在（A.6）中。定理17.4在没有修改的情况下应用。

我们现在证明，在本章中迄今所考虑的类型的罚函数必须是非光滑的。为了简单起见，我们将注意力集中到当存在单一等式约束的情况下，并且考虑形式

的罚函数。是一个满足对于所有的且的函数，假设h连续可微是矛盾的。由于h的极小值为零，所以我们得到。如果是问题17.6的局部解，我们有，因此。如果是局部的极小元，我们因此有

然而，在求解约束优化问题时，f的梯度一般不会消失，所以我们原来的h是连续可微的假设一定是不正确的，从而φ(·;μ)不能是光滑的。

在通过其它机制来计算步骤的方法中，非光滑罚函数也可用作价值函数。有关进一步的细节，请参见第[NO2006]15.4节的一般讨论和第18章和第19章中给出的具体实现。