章节 7.1 罚函数法

致谢：感谢北京大学文再文老师提供的《最优化方法》参考讲义

约束优化问题

考虑约束优化问题：

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & x \in \mathcal{X} \end{aligned}$$

其中，$\mathcal{X}$ 为 x 的可行域

相比于无约束问题的困难：

- 约束优化问题中 x 不能随便取值，梯度下降法所得点不一定在可行域内
- 最优解处目标函数的梯度不一定为零向量

为了解决这些困难，考虑使用**罚函数法**将约束优化问题转化为无约束优化问题处理

罚函数法

罚函数法的思想是将约束优化问题 (434) 转化为无约束优化问题来进行求解.

- 为了保证解的逼近质量，无约束优化问题的目标函数为原约束优化问题的目标函数加上与约束函数有关的惩罚项.
- 对于可行域外的点，惩罚项为正，即对该点进行惩罚；对于可行域内的点，惩罚项为 0，即不做任何惩罚. 因此，惩罚项会促使无约束优化问题的解落在可行域内. 并且只要落在可行域内，那么其就是原约束优化问题的解.
- 罚函数一般由约束部分乘正系数组成，通过增大该系数，我们可以更严厉地惩罚违反约束的行为，从而迫使惩罚函数的最小值更接近约束问题的可行区域.

等式约束的二次罚函数法

首先考虑简单情形：仅包含等式约束的约束优化问题

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & c_i(x) = 0, \quad i \in \mathcal{E} \end{aligned} \tag{141}$$

其中 $x \in \mathbb{R}^n$，$\mathcal{E}$ 为等式约束的指标集，$c_i(x)$ 为连续函数

定义该问题的二次罚函数为:

$$P_E(x, \sigma) = f(x) + \frac{1}{2}\sigma \sum_{i \in \mathcal{E}} c_i^2(x) \tag{142}$$

其中等式右端第二项称为**罚函数**，$\sigma > 0$ 称为**罚因子**

- 由于这种罚函数对不满足约束的点进行惩罚，在迭代过程中点列一般处于可行域之外，因此它也被称为**外点罚函数**.

例 1

为了直观理解罚函数的作用，我们给出一个例子:

考虑优化问题

$$\begin{aligned} \min \quad & x+\sqrt{3}y \\ \text{s.t.} \quad & x^2+y^2=1 \end{aligned}$$

容易求得最优解为 $\left(-\frac{1}{2},-\frac{\sqrt{3}}{2}\right)^{\mathrm{T}}$，考虑二次罚函数

$$P_E(x,y,\sigma)=x+\sqrt{3}y+\frac{\sigma}{2}\left(x^2+y^2-1\right)^2$$

并在下图中绘制出 $\sigma=1$ 和 $\sigma=10$ 对应的罚函数的等高线.

取不同的值时二次罚函数 $P_E(x, y, \sigma)$ 的等高线

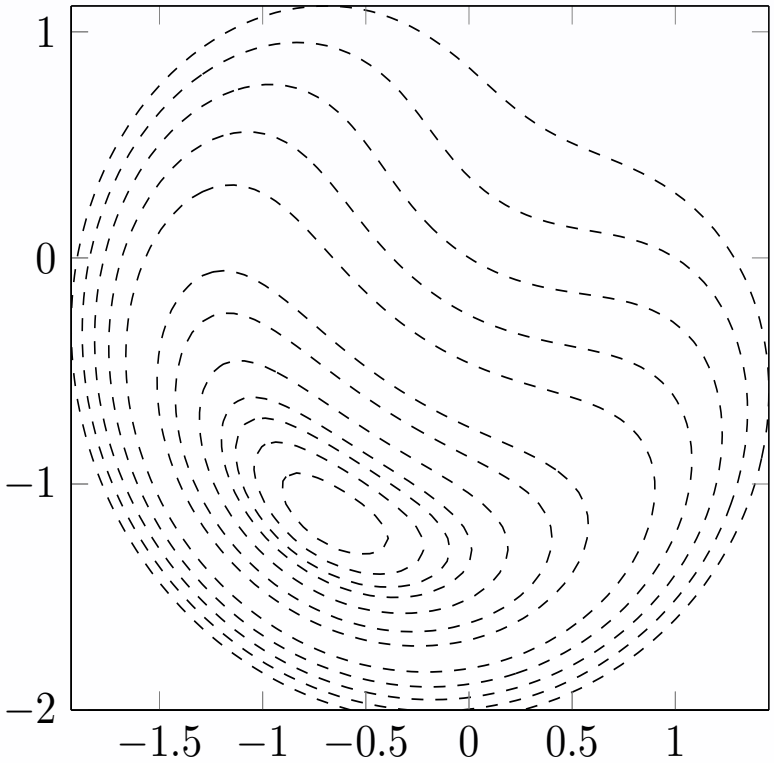

Figure: (a) $\sigma = 1$

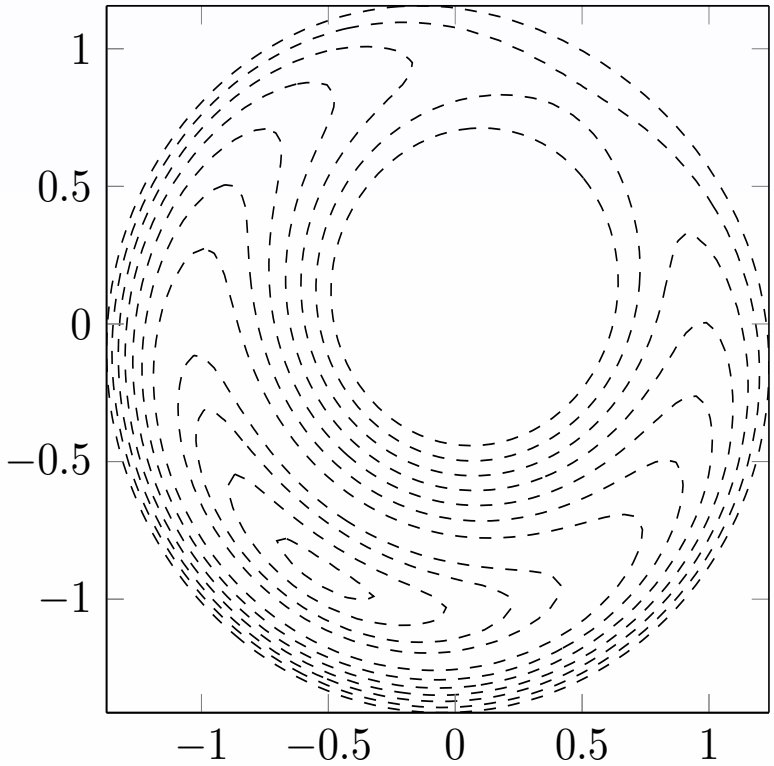

Figure: (b) $\sigma = 10$

例 2

下面这个例子表明，当 σ 选取过小时罚函数可能无下界.

考虑优化问题

$$\begin{aligned} \min \quad & -x^2 + 2y^2 \\ \text{s.t.} \quad & x = 1 \end{aligned}$$

容易求得最优解为 $(1,0)^T$，然而考虑罚函数

$$P_E(x, y, \sigma) = -x^2 + 2y^2 + \frac{\sigma}{2}(x-1)^2$$

对任意的 $\sigma \leq 2$，该罚函数无下界

分析 KKT 条件

从 KKT 条件角度分析：

- 原问题的 KKT 条件：

$$\nabla f(x^*) - \sum_{i \in \mathcal{E}} \lambda_i^* \nabla c_i(x^*) = 0$$

$$c_i(x^*) = 0, \quad \forall i \in \mathcal{E}$$

- 添加罚函数项问题的 KKT 条件：

$$\nabla f(x) + \sum_{i \in \mathcal{E}} \sigma c_i(x) \nabla c_i(x) = 0$$

假设两个问题收敛到同一点，对比 KKT 条件 (梯度式)，应有下式成立：

$$\sigma c_i(x) \approx -\lambda_i^*, \quad \forall i \in \mathcal{E}$$

最优点处乘子 λ^* 固定，为使约束 $c_i(x) = 0$ 成立，需要 $\sigma \to \infty$

二次罚函数法算法

算法 二次罚函数法

1: 给定 $\sigma_1 > 0, x_0, k \leftarrow 1$. 罚因子增长系数 $\rho > 1$.
2: **while** 未达到收敛准则 **do**
3: 以 x^{k-1} 为初始点，求解 $x^k = \arg\min_x P_E(x, \sigma_k)$
4: 选取 $\sigma^{k+1} = \rho\sigma_k$.
5: $k \leftarrow k + 1$
6: **end while**

分析数值困难

- 考虑罚函数 $P_E(x,\sigma)$ 的海瑟矩阵:

$$\nabla^2_{xx}P_E(x,\sigma) = \nabla^2 f(x) + \sum_{i\in\mathcal{E}} \sigma c_i(x)\nabla^2 c_i(x) + \sigma\nabla c(x)\nabla c(x)^{\mathrm{T}}$$

- 等号右边的前两项可以使用拉格朗日函数 $L(x,\lambda^*)$ 来近似，即:

$$\nabla^2_{xx}P_E(x,\sigma) \approx \nabla^2_{xx}L\left(x,\lambda^*\right) + \sigma\nabla c(x)\nabla c(x)^{\mathrm{T}}$$

- 右边为一个定值矩阵和一个最大特征值趋于正无穷的矩阵，这导致 $\nabla^2_{xx}P_E(x,\sigma)$ 条件数越来越大，求解子问题的难度也会相应地增加.
- 此时使用梯度类算法求解将会变得非常困难．若使用牛顿法，则求解牛顿方程本身就是一个非常困难的问题．因此在实际应用中，我们不可能令罚因子趋于正无穷．

注意事项:

- 选取合适的参数 ρ: σ_k 增长过快会使子问题求解困难，σ_k 增长过慢则会增加迭代次数. 另外，也可以自适应地调整 ρ
- 检测到迭代点发散就应该立即终止迭代并增大罚因子
- 为保证收敛，子问题求解误差需要趋于零

习题

证明如下 3 个结论：记 x_k 是 $P_E(x^k, \sigma^k)$ 最小值点。**结论 1：** 设 $\sigma_{k+1} > \sigma_k > 0$, 则有 $P_E(x^k, \sigma^k) \leq P_E(x^{k+1}, \sigma^{k+1})$,

$$\sum_{i\in\mathcal{E}} \|c_i(x^k)\|^2 \geq \sum_{i\in\mathcal{E}} \|c_i(x^{k+1})\|^2, \quad f(x^k) \leq f(x^{k+1}).$$

结论 2： 设令 $\bar{\mathbf{x}}$ 是原问题(141)的最优解，则对任意的 $\sigma^k > 0$ 成立

$$f(\bar{\mathbf{x}}) \geq P_E(x^k, \sigma^k) \geq f(x^k).$$

结论 3： 令 $\delta = \sum_{i\in\mathcal{E}} \|c_i(x^k)\|^2$, 则 x^k 也是约束问题

$$\begin{array}{ll} \min & f(\mathbf{x}) \\ \text{s.t.} & \sum_{i\in\mathcal{E}} \|c_i(x)\|^2 \leq \delta \end{array}$$

的最优解。

收敛性结论

下面的定理需要假设每个罚函数 $P_E(x,\sigma_k)$ 都有最小值，并且 $\{x^k\}$ 有极限点。

定理 5 (二次罚函数法的收敛性 1)

设 x^k 是 $P_E(x,\sigma_k)$ 的全局极小解，σ_k 单调上升趋于无穷，则 x^k 的每个极限点 x^* 都是原问题的全局极小解.

Proof.

设 $\bar{x}$ 为原问题的极小解. 由 x^k 为 $P_E(x,\sigma_k)$ 的极小解，得 $P_E\left(x^k,\sigma_k\right) \leqslant P_E(\bar{x},\sigma_k)$，即

$$f\left(x^k\right)+\frac{\sigma_k}{2}\sum_{i\in\mathcal{E}}c_i^2\left(x^k\right) \leqslant f(\bar{x})+\frac{\sigma_k}{2}\sum_{i\in\mathcal{E}}c_i^2(\bar{x})=f(\bar{x}) \tag{143}$$

整理得:

$$\sum_{i\in\mathcal{E}}c_i^2\left(x^k\right) \leqslant \frac{2}{\sigma_k}\left(f(\bar{x})-f\left(x^k\right)\right) \tag{144}$$

设 x^* 是 x^k 的一个极限点，不妨设 $\{x^k\}$ 的子列 $x^{k_n}\to x^*$。在(144)式中令 $k_n\to\infty$，得 $\sum_{i\in\mathcal{E}}c_i^2(x^*)=0$. 由此易知，$x^*$ 为原问题的可行解，又由(143)式知 $f\left(x^k\right)\leqslant f(\bar{x})$，取极限得 $f(x^*)\leqslant f(\bar{x})$，故 x^* 为全局极小解。 □

收敛性结论

由于定理 1 需要每个罚函数 $P_E(x, \sigma_k)$ 解出全局最小值。这个要求比较高。下面的定理给出更弱的情况下的收敛结果。

定理 6 (二次罚函数法的收敛性 2)

设 $f(x)$ 与 $c_i(x)$ ($i \in \mathcal{E}$) 连续可微，正数序列 $\varepsilon_k \to 0$, $\sigma_k \to +\infty$
在算法13中，子问题的解 x^k 满足 $\left\|\nabla_x P_E\left(x^k, \sigma_k\right)\right\| \leq \varepsilon_k$，而对 x^k 的任何极限点 x^*，都有 $\left\{\nabla c_i\left(x^*\right), i \in \mathcal{E}\right\}$ 线性无关，则 x^* 是等式约束最优化问题(141) 的 KKT 点，且

$$\lim_{k \to \infty}\left(-\sigma_k c_i\left(x^k\right)\right) = \lambda_i^*, \quad \forall i \in \mathcal{E}$$

其中 λ_i^* 是约束 $c_i(x^*) = 0$ 对应的拉格朗日乘子.

收敛性分析的推论

- 不管 $\{\nabla c_i(x^*)\}$ 是否线性无关，通过算法13给出解 x^k 的聚点总是 $\phi(x) = \|c(x)\|^2$ 的一个稳定点．这说明即便没有找到可行解，我们也找到了使得约束 $c(x) = 0$ 违反度相对较小的一个解．
- 定理6虽然不要求每一个子问题精确求解，但要获得原问题的解，子问题解的精度需要越来越高．它并没有给出一个非渐进的误差估计，即没有说明当给定原问题解的目标精度时，子问题的求解精度 ε_k 应该如何选取．

一般约束问题的二次罚函数法

考虑不等式约束问题：

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & c_i(x) \leqslant 0, i \in \mathcal{I} \end{aligned}$$

定义该问题的二次罚函数为:

$$P_l(x, \sigma) = f(x) + \frac{1}{2}\sigma \sum_{i \in \mathcal{I}} \tilde{c}_i^2(x)$$

其中 $\tilde{c}_i(x)$ 定义为：

$$\tilde{c}_i(x) = \max\{c_i(x), 0\}$$

注：$h(t) = (\min\{t, 0\})^2$ 关于 t 可导，故 $P_l(x, \sigma)$ 梯度存在，所以可以使用梯度类算法求解

一般约束问题的二次罚函数法

现在考虑一般约束问题：

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & c_i(x) = 0, i \in \mathcal{E} \\ & c_i(x) \leqslant 0, i \in \mathcal{I} \end{aligned} \tag{145}$$

定义该问题的二次罚函数为:

$$P(x, \sigma) = f(x) + \frac{1}{2}\sigma \left[\sum_{i \in \mathcal{E}} c_i^2(x) + \sum_{i \in \mathcal{I}} \tilde{c}_i^2(x) \right]$$

其中等式右端第二项称为惩罚项，$\tilde{c}_i(x)$ 的定义如(145)式，常数 $\sigma > 0$ 称为罚因子.

二次罚函数法的优缺点

优点：

- 将约束优化问题转化为无约束优化问题，当 $c_i(x)$ 光滑时可以调用一般的无约束光滑优化问题算法求解.
- 二次罚函数形式简洁直观而在实际中广泛使用.

缺点：

- 需要 $\sigma \to \infty$，此时海瑟矩阵条件数过大，对于无约束优化问题的数值方法拟牛顿法与共轭梯度法存在数值困难，且需要多次迭代求解子问题.
- 对于存在不等式约束的 $P_E(x, \sigma)$ 可能不存在二次可微性质，光滑性降低.
- 不精确，与原问题最优解存在距离.(后面将介绍精确罚函数法)

低秩矩阵恢复

- 某视频网站提供了约 48 万用户对 1 万 7 千多部电影的上亿条评级数据，希望对用户的电影评级进行预测，从而改进用户电影推荐系统，为每个用户更有针对性地推荐影片.
- 显然每一个用户不可能看过所有的电影，每一部电影也不可能收集到全部用户的评级. 电影评级由用户打分 1 星到 5 星表示，记为取值 1- 5 的整数. 我们将电影评级放在一个矩阵 M 中，矩阵 M 的每一行表示不同用户，每一列表示不同电影. 由于用户只对看过的电影给出自己的评价，矩阵 M 中很多元素是未知的

	电影 1	电影 2	电影 3	电影 4	…	电影n
用户 1	4	?	?	3	⋯	?
用户 2	?	2	4	?	⋯	?
用户 3	3	?	?	?	⋯	?
用户 4	2	?	5	?	⋯	?
⋮	⋮	⋮	⋮	⋮		⋮
用户m	?	3	?	4	⋯	?

低秩矩阵恢复问题的性质

该问题在推荐系统、图像处理等方面有着广泛的应用。

- 由于用户对电影的偏好可进行分类，按年龄可分为：年轻人，中年人，老年人；且电影也能分为不同的题材：战争片，悬疑片，言情片等。故这类问题隐含的假设为补全后的矩阵应为低秩的。即矩阵的行与列会有“合作”的特性，故该问题具有别名“collaborative filtering”。
- 除此之外，由于低秩矩阵可分解为两个低秩矩阵的乘积，所以低秩限制下的矩阵补全问题是比较实用的，这样利于储存且有更好的诠释性。
- 有些用户的打分可能不为自身真实情况，对评分矩阵有影响，所以原矩阵是可能有噪声的。

低秩矩阵恢复

由上述分析可以引出该问题：

- 令 Ω 是矩阵 M 中所有已知评级元素的下标的集合，则该问题可以初步描述为构造一个矩阵 X，使得在给定位置的元素等于已知评级元素，即满足 $X_{ij} = M_{ij},\ (i,j) \in \Omega$.
- 低秩矩阵恢复（low rank matrix completion）

$$\begin{aligned} \min_{X \in \mathbb{R}^{m \times n}} \quad & \operatorname{rank}(X), \\ \text{s.t.} \quad & X_{ij} = M_{ij},\ (i,j) \in \Omega. \end{aligned} \tag{146}$$

  $\operatorname{rank}(X)$ 正好是矩阵 X 所有非零奇异值的个数

- 矩阵 X 的核范数（nuclear norm）为矩阵所有奇异值的和，即：$\|X\|_* = \sum_i \sigma_i(X)$:

$$\begin{aligned} \min_{X \in \mathbb{R}^{m \times n}} \quad & \|X\|_*, \\ \text{s.t.} \quad & X_{ij} = M_{ij},\ (i,j) \in \Omega. \end{aligned} \tag{147}$$

应用举例——矩阵补全问题

引入等式约束的二次罚函数，

$$\min \quad \|X\|_* + \frac{\sigma}{2} \sum_{(i,j)\in\Omega} (X_{ij} - M_{ij})^2$$

令 $\sigma = \frac{1}{\mu}$，即有等价形式的优化问题：

$$\min \quad \mu\|X\|_* + \frac{1}{2} \sum_{(i,j)\in\Omega} (X_{ij} - M_{ij})^2 \tag{148}$$

矩阵补全问题罚函数法算法

算法 矩阵补全问题求解的罚函数法

给定初值 X^0，最终参数 μ，初始参数 μ_0，因子 $\gamma \in (0,1), k \leftarrow 1$
while $\mu_k \geq \mu$ **do**
  以 X^{k-1} 为初值，$\mu = \mu_k$ 为正则化参数求解问题(148)，得 X^k
  if $\mu_k = \mu$ **then**
    停止迭代，输出 X^k
  else
    更新罚因子 $\mu_k = \max\{\mu, \gamma\mu_k\}$
    $k \leftarrow k+1$
  end if
end while

其他类型的罚函数法——内点罚函数法

内点罚函数在迭代时始终要求自变量 x 不能违反约束，故主要用于不等式约束优化问题。对于不等式优化问题，定义**对数罚函数**：

$$P_I(x,\sigma) = f(x) - \sigma \sum_{i\in\mathcal{I}} \ln\left(-c_i(x)\right)$$

其中等式右端第二项称为惩罚项，$\sigma > 0$ 称为罚因子.

- $P_I(x,\sigma)$ 的定义为域 $\{x \mid c_i(x) < 0\}$ 因此在迭代过程中自变量 x 严格位于可行域内部.
- 当 x 趋于可行域边界时，由于对数罚函数的特点，$P_I(x,\sigma)$ 会趋于正无穷，这说明对数罚函数的极小值严格位于可行域内部. 但原问题最优解通常位于可行域边界，应减小惩罚效果，即调整罚因子 σ 使其趋于 0.

例 3

考虑优化问题

$$\begin{aligned} \min \quad & x^2 + 2xy + y^2 + 2x - 2y \\ \text{s.t.} \quad & x \geqslant 0, y \geqslant 0 \end{aligned}$$

容易求得最优解为 $x = 0, y = 1$，考虑对数罚函数

$$P_l(x, y, \sigma) = x^2 + 2xy + y^2 + 2x - 2y - \sigma(\ln x + \ln y)$$

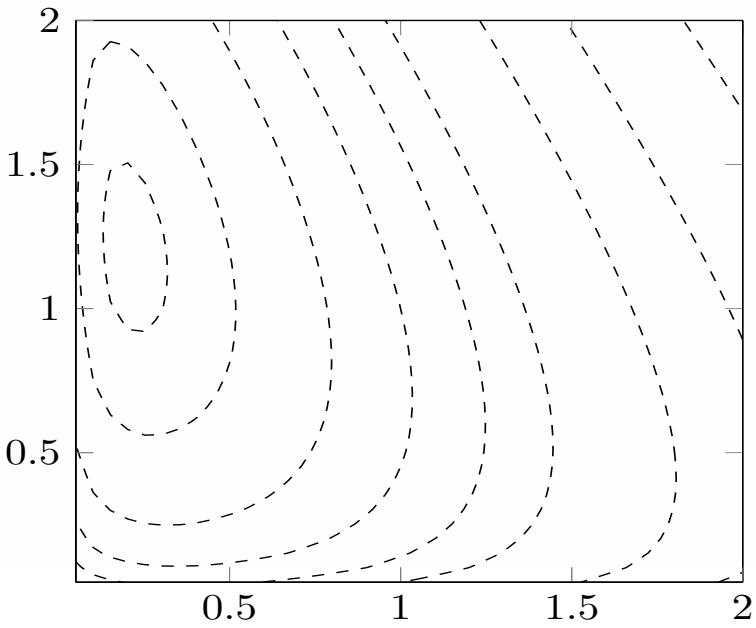

Figure: (a) $\sigma = 1$

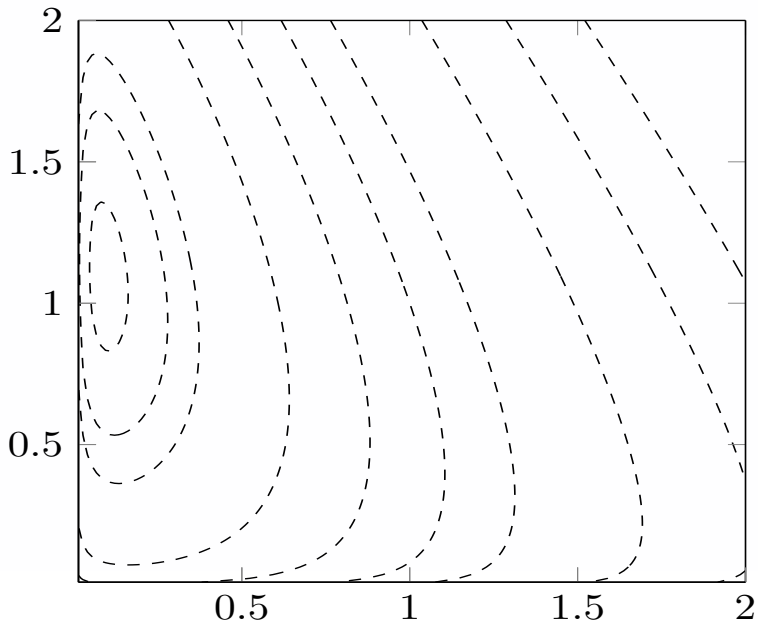

Figure: (b) $\sigma = 0.4$

对数罚函数法算法

算法 对数罚函数法

1: 给定 $\sigma_0 > 0$，可行解 x^0，$k \leftarrow 0$．罚因子缩小系数 $\rho \in (0,1)$．
2: **while** 未达到收敛准则 **do**
3: 以 x^{k-1} 为初始点，求解 $x^k = \arg\min_x P_I(x, \sigma_k)$
4: 选取 $\sigma_{k+1} = \rho\sigma_k$．
5: $k \leftarrow k+1$．
6: **end while**

- 初始点 x^0 必须是一个可行点
- 常用的收敛准则可以包含

$$\left| \sigma_k \sum_{i \in \mathcal{I}} \ln(-c_i(x^{k+1})) \right| \leq \varepsilon,$$

其中 $\varepsilon > 0$ 为给定的精度．

- 当 σ 趋于 0 的时候，同样存在数值困难

其他类型的罚函数法——精确罚函数法

- 由于二次罚函数存在数值困难，并且与原问题的解存在误差，故考虑精确罚函数.
- **精确罚函数**，是一种问题求解时不需要令罚因子趋于正无穷（或零）的罚函数. 常用的精确罚函数是 ℓ_1 罚函数.
- 二次罚函数对应的问题是光滑的,ℓ_1 罚函数对应的问题是非光滑的。

定义一般约束优化问题(145)的 ℓ_1 罚函数:

$$P(x,\sigma) = f(x) + \sigma\left[\sum_{i\in\mathcal{E}}|c_i(x)| + \sum_{i\in\mathcal{I}}\tilde{c}_i(x)\right]$$

这里用绝对值代替二次惩罚项，下面的定理揭示了 ℓ_1 罚函数的精确性

定理 7 (精确罚函数法的收敛性)

设 x^* 是一般约束优化问题(145)的一个严格局部极小解，且满足 *KKT* 条件，其对应的拉格朗日乘子为 $\lambda_i^*, i \in \mathcal{E} \cup \mathcal{I}$，则当罚因子 $\sigma > \sigma^*$ 时，x^* 也为 $P(x, \sigma)$ 的一个局部极小解，其中

$$\sigma^* = \|\lambda^*\|_\infty \overset{def}{=} \max_i |\lambda_i^*| .$$

另一方面，存在 $\hat{\sigma} > 0$, 对于 $\sigma \geq \hat{\sigma}$，如果 $\hat{x}$ 是罚函数 $P(x, \sigma)$ 的稳定点. 那么，如果 $\hat{x}$ 是一般约束优化问题(145)的可行点，则 $\hat{x}$ 也满足(145)的 *KKT* 条件。

- 定理7说明对于精确罚函数，罚因子充分大（不是正无穷），原问题的极小值点是 ℓ_1 罚函数的极小值点，这和定理5是有区别的. 反之，罚函数的稳定点若是可行点，在较弱的假设下，则也是约束问题的 KKT 点。

精确罚函数法算法

算法 精确罚函数法

1: 给定 $\sigma_1 > 0, x_0, k \leftarrow 0$．罚因子增长系数 $\rho > 1$．
2: **while** 未达到收敛准则 **do**
3: 以 x^{k-1} 为初始点，求解
$$x^k = \arg\min_x \left\{ f(x) + \sigma \left[\sum_{i\in\mathcal{E}} |c_i(x)| + \sum_{i\in\mathcal{I}} \tilde{c}_i(x) \right] \right\}$$
4: 选取 $\sigma^{k+1} = \rho\sigma_k$．
5: $k \leftarrow k + 1$
6: **end while**

- 取 ρ 为固定值是一种在实际中行之有效的方法，然而也可能出现：
  - 初始罚因子过小，迭代次数增加，且最优解可能远离原问题最优解
  - 罚因子过大时子问题求解困难，此时需要适当减小罚因子
- 子问题求解的初始点取法不唯一。一般取上一次子问题求解的最优值点作为下一次子问题求解的起点。

一般的精确罚函数法

除了 ℓ_1 范数，可以用更一般的范数定义精确罚函数法：

$$P(x,\sigma) = f(x) + \mu \left\| c_{\mathcal{E}}(x) \right\| + \mu \left\| [c_{\mathcal{I}}(x)]^+ \right\|$$

其中，$\|\cdot\|$ 可以是任意的向量范数，$[c_{\mathcal{I}}(x)]^+$ 为向量各分量取 $\max\{0, x\}$

则我们可以推广定理7，将 $\|\cdot\|_\infty$ 替换为 $\|\cdot\|_D$ ($\|\cdot\|$ 的对偶范数)
对偶范数的定义如下：

$$\|x\|_D = \max_{\|y\|=1} x^T y$$

常见的对偶范数：

- $\|\cdot\|_1$ 和 $\|\cdot\|_\infty$ 互为对偶
- ℓ_2 范数的对偶是它自身.

精确罚函数的非光滑性

下面说明，精确罚函数必然是非光滑的。

为简化讨论，假设仅有一条等式约束 $c_1(x) = 0$。设罚函数的形式为：

$$P(x, \sigma) = f(x) + \sigma h(c_1(x))$$

其中，函数 $h : \mathbb{R} \to \mathbb{R}$ 满足 $h(y) \geq 0,\ \forall y \in \mathbb{R}$ 且 $h(0) = 0$

若函数 h 连续可微，则有 $\nabla h(0) = 0$ 成立。故对于 $P(x, \sigma)$ 最优点 x^*，有

$$0 = \nabla P(x^*, \sigma) = \nabla f(x^*) + \sigma \nabla c_1\left(x^*\right) \nabla h\left(c_1\left(x^*\right)\right) = \nabla f(x^*)$$

然而，在约束优化问题中，f 取到最小值时，其梯度不一定为 0。这说明假设 h 连续可微是不正确的，即罚函数项必须是非光滑的。

另一方面，正是罚函数项的非光滑性，克服了原函数在最优点处的梯度，才能在充分大的罚因子下实现精确求解。

参考文献


Nocedal J, Wright S, Numerical optimazation[M]. 2nd ed. Berlin: Springer Science & Business Media, 2006


Han S P , Mangasarian O L . Exact Penalty Functions in Nonlinear Programming. , 1978.