Chapter 3

Numerical Optimization

3.1 优化问题

优化问题来自科学和工程的各个领域，在商业和工业的中的应用也非常普遍。任何设计问题一般都要优化某些性能指标，如成本或效率。在数学上，优化问题可以表示称确定某些参数，使给定函数在某个定义域上取得极值，也就是说:

定义3.1.1 (优化问题). 给定一个函数f:$\mathbb{R}^n \to \mathbb{R}$和一个几何$S \in \mathbb{R}^n$,使$f$在$x^*$处的值是其在$S$上的最小值。

由于f最大值可以考虑-f的最小值，因此我们可以在这里只讨论最小值问题。如果不对目标函数f与集合S做某些假定，那么我们很难建立优化问题最优解的存在性和唯一性。因此，我们接下来会给出一些限制，在这些限制下，再来讨论具体的数值优化方法。

在所有假设中，最重要的假设就是凸性

定义3.1.2 (凸性). 称集合S是凸集，即对所有x,y∈ S，有

$$\{ax + (1-a)y : a \in [0,1]\} \in S$$

进一步，如果函数沿S中任意线段的图像都不超过连接线段端点函数值的弦，则称函数f在凸集S上是凸的。

3.1.1 一维优化

我们从一维优化方法开始讨论，这个问题不仅重要，并且是高维优化问题中许多算法的子问题。首先我们需要一个确定最小值点所在区间范围的方法，若在区间[a,b]上函数f只有一个峰值，则称函数是单峰的。

黄金分割搜索

假定f在[a,b]上是单峰的，$x_1, x_2 \in [a,b]$，且$x_1 < x_2$，比较两点的函数值，利用单峰的性质可以去掉$(x_2, b], [a, x_1)$中的一个，使最小值点位于剩下的子区间内。如果$f(x_1) < f(x_2)$,则最小值点肯定不会在$[a, x_1)$内，反之同理，这样就将最小值的区间缩小了一部分。我们为使包含最小值区间长度的降低规则变化，每个新区间两个端点的相对位置应该与上一段相同。由此我们得到黄金分割搜索的方法。

例3.1.1 (Golden section search algorithm)**.** *Using the golden section search algorithm to solve the minimum of function* $f(x) = 2 - x^{e^{-x^2}}$.

x_1	$f(x_1)$	x_2	$f(x_2)$
0.763932022500210	2	1.13948725299353	0.987223637414642
0.763932022500210	1.13948725299353	1.52786404500042	0.958084289228258
1.05572809000084	0.982049943239491	1.52786404500042	0.958084289228258
1.23606797749979	0.952936383585259	1.52786404500042	0.958084289228258
1.23606797749979	0.952936383585259	1.41640786499874	0.952064819561126
1.30495168499706	0.952936383585259	1.41640786499874	0.952064819561126
1.30495168499706	0.952936383585259	1.37383539249432	0.950719023262687
1.30495168499706	0.952936383585259	1.34752415750147	0.950273540207883
1.32121292250862	0.950180712996851	1.34752415750147	0.950273540207883
1.32121292250862	1.33747416002019	1.34752415750147	0.950193723852238
1.32121292250862	0.950180712996851	1.33126291998991	0.950171330706877
1.32505167995962	0.950170332256569	1.33126291998991	0.950171330706877
1.32505167995962	0.950170332256569	1.32889043741062	0.950168386761201
1.32651795483134	0.950168602406948	1.32889043741062	0.950168386761201
1.32742416253891	0.950168144654073	1.32833037024648	0.950168386761201
1.32742416253891	0.950168144654073	1.32889043741062	0.950168151342343

Table 3.1: Em:3.3.1

逐次抛物插值Successive parabolic interpolation

优化问题的黄金分割搜索除了对函数值进行比较外不再使用这些值，而逐次抛物插值的思想是，通过利用更多的这些函数值，加快方法的收敛速度。其具体流程是，首先用最小化函数在三个点上的值做二次多项式拟合，假定所做抛物线的最小值只有一个，以它作为函数最小值的新的近似，将前面三个点去掉一个，重复此过程直至收敛。计算与上一个方法相同的例子，我们有

例3.1.2 (Successive parabolic interpolation)**.** *Using the successive parabolic interpolation to solve the minimum of function* $f(x) = 2 - x^{e^{-x^2}}$.

k	x_k	$f(x_k)$
1	1.51294171057210	0.957134474135585
2	1.67421884889364	0.968262661681680
3	1.51740842355144	0.957415468387465
4	1.51924415770084	0.957531798557215
5	1.45418212901426	0.953777168082556
6	1.43030742004434	0.952645221040546
7	1.39803017069384	0.951398400342936
8	1.37914176450845	0.950847247204598
9	1.36276414357126	0.950491771217602
10	1.35205117833781	0.950326456371295
11	1.34412808713455	0.950240677909952
12	1.33884651075076	0.950201489255118
13	1.33275023777987	0.950174770438180
14	1.33110740438389	0.950171041721443
15	1.33001663049288	0.950169392634638
16	1.32928969056623	0.950168662111940

Table 3.2: Em:3.3.2

牛顿法

局部抛物近似的另一种方法是使用截断的泰勒级数展开，即：

$$f(x+h) \approx f(x) + f'(x)h + \frac{1}{2}f''(x)h^2.$$

则我们可以得到最小值为$h = -f'(x)/f''(x)$，其算法可以写作：
x_0 =初始值
for$k = 0, 1, 2, dots$
$x_{k+1} = x_k - f'(x_k)/f''(x_k)$
end

例3.1.3 (Newton method)**.** *Using the Newton method to solve the minimum of function* $f(x) = 2 - xe^{-x^2}$ß.

k	x_k	$f(x_k)$
1	0.50000000000000	1.61059960846430
2	0.700000000000000	1.57116152407091
3	0.707072135785007	1.57111805854924
4	0.707106780337929	1.57111805751965

Table 3.3: Em:3.3.3

3.2 优化计算方法

接下来考虑多维无约束优化，它的许多特点与一维优化和求解n维非线性方程组是相同的。

3.2.1 最速下降法

基于一维优化的黄金分割搜索算法提出的。但它不能像黄金分割搜索那样保证收敛。

最速下降法思想是，任意选取一初始点，以其负梯度方向作为其下降方向，走一段距离到下一个点，再重复这样的步骤，最终得到收敛的结果。其算法可以表示为：

$\mathbf{x}_0$ =初始值
for $k = 0, 1, 2, \ldots$
$s_k = -\nabla f(\mathbf{x}_k)$
选择α_k使$f(\mathbf{x}_k + \alpha_k s_k)$最小化
$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k s_k$
end

最速下降法是十分可靠的方法，只要负梯度不为0即可继续进行。但这种方法很难看出函数的性态。

3.2.2 随机梯度下降法

3.2.3 牛顿法

同样用局部抛物近似作为目标函数的近似，由截断泰勒级数展开，有：

$$f(x+s) \approx f(x) + \nabla f(x)^T s + \frac{1}{2} s^T H_f(x) s.$$

其中$H_f(x)$是f的二阶偏导数的黑塞矩阵，即$\{H_f(x)\}_{ij} = \partial^2 f(x)/\partial x_i \partial x_j$,由此我们可以得到无约束优化的牛顿法的算法为：

$\mathbf{x}_0$ =初始值
for $k = 0, 1, 2, \ldots$
解$H_f(\mathbf{x}_k)s_k = -\nabla f(\mathbf{x}_k) B s_k$
$\mathbf{x}_{k+1} = \mathbf{x}_k + s_k$
end 与一维情形类似，拟牛顿法和割线法对于无法计算二阶导数(Hessian矩阵)的情形

3.2.4 共轭梯度法

共轭梯度法师牛顿法的一个改进方案，这种方法不需要二阶导数，也不需要存储黑塞矩阵，对于大型问题来说是非常合适的，其算法可以表示为：

$\mathbf{x}_0$ =初始值
$\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$
$\mathbf{s}_0 = -\mathbf{g}_0$
for $k = 0, 1, 2, \ldots$
选择α_k使$f(\mathbf{x}_k + \alpha_k s_k)$最小化
$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k s_k$
$\mathbf{g}_{k+1} = \nabla f(x_{k+1})$
$\beta_{k+1} = (\mathbf{g}_{k+1}^T \mathbf{g}_{k+1}/(\mathbf{g}_k^T \mathbf{g}_k) \quad \mathbf{s}_{k+1} = -\mathbf{g}_{k+1} + \beta_{k+1}\mathbf{s}_k$
end

3.3 约束优化

接下来考虑有关约束优化的方法，首先考虑形如

$$\min_x f(\mathbf{x})$$

$$s.t.\ \mathbf{g}(\mathbf{x}) = \mathbf{0}$$

的等式优化问题。

3.3.1 罚函数与障碍函数法

这类方法的思想是将约束优化问题转化为无约束优化问题，再用之前的方法解决问题。罚函数法师通过目标函数和约束条件加权组合的最小化同步求得目标函数的最小值，例如对于之前提到的等式优化问题，可以等价提出一种无约束问题：

$$\min_x \phi_p(\mathbf{x}) = f(\mathbf{x}) + \frac{1}{2}\rho \mathbf{g}(\mathbf{x})^T \mathbf{g}(\mathbf{x})$$

的解，在适当的条件下可以证明此无约束问题的最小值是收敛到约束优化问题的。

例3.3.1 (Penalty method). *Using the penalty method to solve the following optimization problem:*

$$\begin{aligned} minJ(\mathbf{x}) &:= f(\mathbf{x}) = 0.5x_1^2 + 2.5x_2^2 \\ s.t.g(\mathbf{x}) &:= x_2 - x_1 + 1 = 0, \end{aligned} \tag{3.1}$$

ρ	x_1	x_2
1	0.454591922176405	-0.0908981213894332
10	0.768891319403512	-0.154124436139317
100	0.823110443717319	-0.168559043828027
1000	0.826860554167555	-0.172240529824000
10000	0.827234727128824	-0.172610821001843
100000	0.827270404929452	-0.172646391181139
100000	0.827278642591352	-0.172654625246885

Table 3.4: Em:3.3.4

3.3.2 线性规划

线性规划是一种最重要、最普遍的约束优化，在这类问题中目标函数和约束都是线性的，一种标准形式可以定义为：

$$\min_x f(\mathbf{x}) = \mathbf{c}^T \mathbf{x}$$

$$s.t.\ \mathbf{A}\mathbf{x} = \mathbf{b}; \mathbf{x} \geq 0$$

这类问题的可行域是$\mathbb{R}^n$中的凸多面体，其全局最小值一定在凸多面体的某个顶点出现。下面以图解法来示例线性规划的流程。假设求解问题：

例3.3.2 (linear program)**.** *Using the linear programming method to solve the following optimization problem:*

$$\begin{aligned} &\min J(\mathbf{x}) := f(\mathbf{x}) = -8x_1 - 11x_2 \\ &s.t. 5x_1 + 4x_2 \leq 40, \\ &\quad -x_1 + 3x_2 \leq 12 \\ &\quad x_1 \geq 0, x_2 \geq 0 \end{aligned} \tag{3.2}$$

软件包与参考材料

Exercise