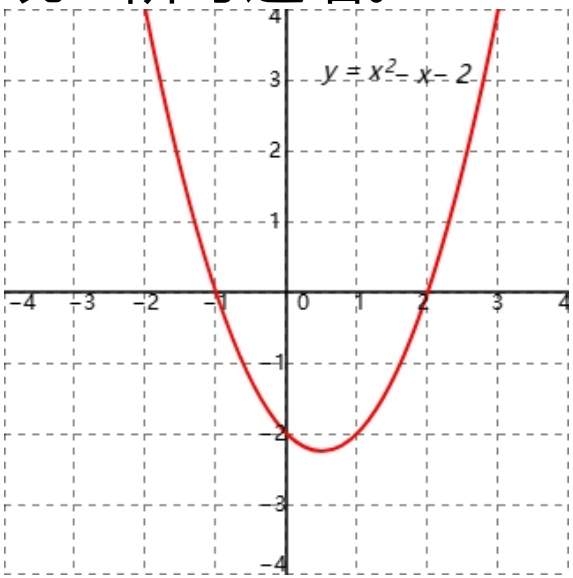
凸函数初探

谭富炫

2025 年 10 月 25 日

什么事凸性

回忆在数学中，我们称一个函数是凸的，如果其二阶导 $f''(x)>0$，或者说一阶导递增。



如图是一个典型的二次函数

注意函数的凸性描述的是「上境图」的性质（即函数上方点的集合）

约定：凸函数 = **下凸**函数，凹函数 = **上凸**函数。这里国内外的定义似乎相反，以通用定义为标准。

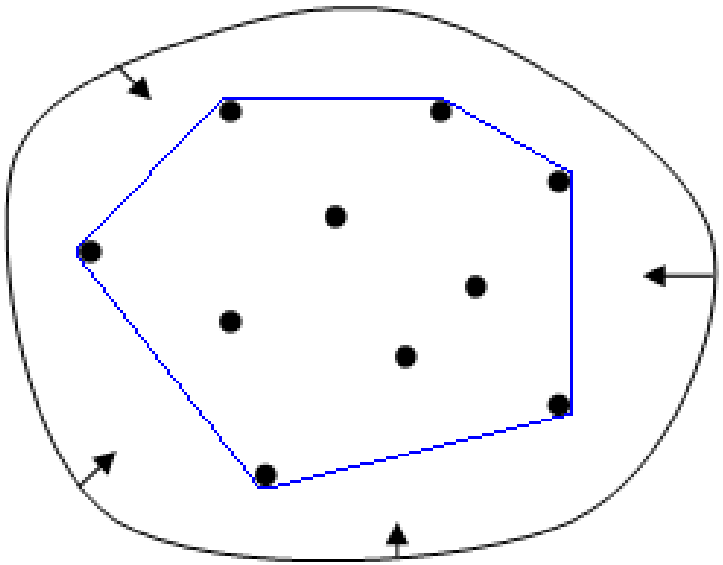
凸函数的性质

对于任意两个凸函数 f(x),g(x)，max(f(x),g(x)) 也是凸函数。
同理，对于任意两个凹函数，其 min 也是凹函数。
特别的，直线既是凸函数也是凹函数

离散情况

在 OI 中，更常见的是离散情况。
如果从数列的角度出发：我们称一个数列是凸的，如果其二阶差分 ≥ 0。
如果从函数的角度出发：则凸函数由若干条斜率递增的直线拼起来，形如分段函数。尽管在连接点处函数不可导，但我们认为其二阶导即为前后两段斜率之差。
离散情况满足连续情况的大部分性质

什么事凸包



凸包是一个点集 $\mathcal{C} = \{(x, y)\}$
满足任意两点的连线都在凸包内
$\forall p_1, p_2 \in \mathcal{C}, t \in (0, 1), p_1 + (p_2 - p_1)t \in \mathcal{C}$
找到最左边（x 最小）的点和最右边（x 最大）的点，可以把凸包分为上下两个凸包。
那么上凸包的边，从左往右斜率递减；下凸包的边，从左往右斜率递增。
如果我们连接定义域开头和结尾两个点，那么可以把凸函数也看成凸包。

维护凸包的算法

求凸包最常用的算法是 Graham 扫描法
OI-wiki 上有动图（我小时候都没有）
凸包最有用的一点是可以做 Mincowski 合并。
可以证明对于两个凸包 $\mathcal{C}_1, \mathcal{C}_2$，其 Mincowski 和 $\{p_1 + p_2 | p_1 \in \mathcal{C}_1, p_2 \in \mathcal{C}_2\}$ 也是一个凸包。
求 Mincowski 和最常用的算法是归并排序，可以证明两个凸包的 Mincowski 和的边恰为原两个凸包的边按顺序拼起来的。
如果边提前有序，那么时间复杂度是 $\mathcal{O}(n)$ 的，否则需要 $\mathcal{O}(n \log n)$ 排序。
OI-wiki 上有讲（我小时候都没有）。
板子：[JSOI2018] 战争
凸包还可以动态维护，参见 CF70D，不过要写平衡树，一般不会考这个。
二分凸包可以求出凸包与直线的切点。
半平面交得到的也是凸包。

维护凸函数的算法

李超树是一种经典的维护凸函数的算法。
OI-wiki 貌似没有提到它的凸性，其实若干条线段取 max/min 本身就是凸的了。
可以把凸函数转化为凸包进行维护，凸函数的 max-plus 卷积即为凸包的 Mincowski 和。
(max-plus 卷积指 $h(x) = \max_{0 \le y \le x}\{f(y) + g(x-y)\}$)

还有一种小清新的维护凸函数的数据结构——slope trick。
考虑维护一个 set 或者其它的东西，包含 $\Delta^2 a_i$ 个 i。
这里 Δ^2 表示二阶差分，$\Delta^2 a_i = \Delta a_{i+1} - \Delta a_i = a_{i+2} - 2a_{i+1} - a_i$。
或者说每出现斜率变化就把这个变化的幅度插入进去。
不妨假设斜率从 0 开始
那么可以发现：

- 求 Δa_i，即为 $0 \sim i-1$ 的元素的数量
- 两个凸函数的 max-plus 卷积，即为它们的 slope trick 的可重并。
- 特别的，如果你想要跟一个线段做 max-plus 卷积，即插入一条边，相当于 set 插入元素。

凸性证明

除了用定义证明或者瞪眼，还可以用网络流的方式证明。
如果你发现答案可以转写成一个费用流的形式，那么它肯定关于流量是一个凹函数。
同时，打表也不失为一种好方法。

更高维度

前面提到半平面交得到的一定是凸包，即满足所有 $a_i x_i + b_i y_i \leq c_i$ 的点集。
如果推广到更高维度，在 n 维空间中，存在若干半超空间的限制 $\sum_{j=1}^{n} a_{i,j} x_j \leq b_i$，求其关于一个线性函数 $f(x) = \sum_{i=1}^{n} v_i x_i$ 的最值，即为所谓「线性规划」的问题。
算法竞赛中，线性规划的 practical 的算法是单纯形法（simplex），时间复杂度是 $\mathcal{O}$(玄学)，最坏是阶乘级别的，可以用来骗分。

WQS 二分

设想你有一个隐藏的凸函数 $f(x)$，你希望求出它的某个点的值 $f(x_0)$，但是你只有求出其全局最小值 $\min f(x)$ 的方法。
但是你还发现你还能求出 $\min\{f(x) - kx\}$，所以可以用二分凸包的方式求出 $f(x_0)$，只是此时我们可以改变的是 k 而非切点 x_0。
这就是所谓的 WQS 二分

QOJ9119

给你一张连通无向图，有重边无自环，请你判断是否存在一种给边赋权的方式 $w_i \in \{0, 1, \ldots, L\}$，使其**任意**生成树的权重恰为 W。
如果存在这样的权重分配，求所有满足条件的分配中平方和 $\sum w_i^2$ 的最小值。
给出 K, L，对于所有 $W = 0, 1, \ldots, K$ 求出答案。
$N, L, K \leq 10^5, M \leq 2 \times 10^5$

考虑建出原图的一棵生成树，然后我们发现对于任意一条非树边，应当要求其树上路径上任意一条边的权值都与其相等。
我们不断地缩下去，最终发现对于每个点双内部的边的权值都应当相同。
所以建出圆方树。记总共有 C 个点双，第 i 个点双有 a_i 条树边，b_i 条边，那么限制形如：

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{C} b_i x_i^2 \\ \text{s.t.} \quad & \sum_{i=1}^{C} a_i x_i = W, \\ & 0 \le x_i \le L, \forall i \end{aligned}$$

然后相当于 C 条 $(a_i x_i, b_i x_i^2), 0 \le x_i \le L$ 的线段做 min-plus 卷积，做 Mincowski 和即可。

QOJ6660

给你一棵带权树，根为 0，每个点 i 有一个 A_i 和 B_i，意味着如果你从 i 出发走距离为 d 的路程，需要 $A_i + d \times B_i$ 的代价，允许换乘任意多次，请计算从根开始出发走到其它点的最小代价。

$N \leq 10^5, 0 \leq A_i \leq 10^{12}, 0 \leq B_i \leq 10^6, 1 \leq W_i \leq 10^6$

首先显然如果我们要换乘，那么 B_i 一定是越来越小的。
所以考虑按照 B_i 排序，从大到小加入。
但是这个树上问题还是很难刻画距离，所以考虑套一层动态点分治。
然后对于每一层分治中心，可以维护一个凸包 $f(x)$，表示到分治中心的距离为 x 的情况下，走到这个点的最小代价。
然后更新形如插入一条线段，用李超树维护即可。由于是全局插入，所以时间复杂度是 $O(n \log^2 n)$。
也可以用一个单调栈维护凸包然后二分查询。

QOJ3272

给定长为 n 的序列 a，进行 q 次操作

1. 给定 v，全局 ckmin
2. 将所有 $a_i \leftarrow a_i + i$
3. 查询区间和

$n, q \leq 2 \times 10^5, 0 \leq a_i, v_i \leq 10^{12}$

直观地想，a 一定是越来越递增。
对于已经递增的段，我们可以二分找到一个分界点，然后把 ckmin 变成区间覆盖，操作 2 即为区间加等差数列。总之是好做的。
注意到一段只要被操作 1 影响过一次，那么由于后续操作只会使其递增，所以其内部一定是递增的。
那么我们可以把整个序列划为两部分：S 和 $[1,n]\backslash S$，其中 S 表示被操作 1 影响过至少一次的子序列。
那么我们只需要考虑一个元素何时加入 S，考虑二分。
记 b_i 表示第 i 次 1 操作前 2 操作的数量。
相当于查询是否 $\exists j\in[1,mid]$，使 $a_i+b_ji\geq v_j$。
转写为判断 $a_i\geq \min_{j\leq mid}\{v_j-b_ji\}$，可以维护一个 (b_j,v_j) 的凸包，每次拿一个斜率为 i 的线去切。
所以考虑整体二分，然后就做完了。时间复杂度 $O(n\log n)$。由于斜率单调，所以可以双指针维护。

鸣谢：[广告位招租]