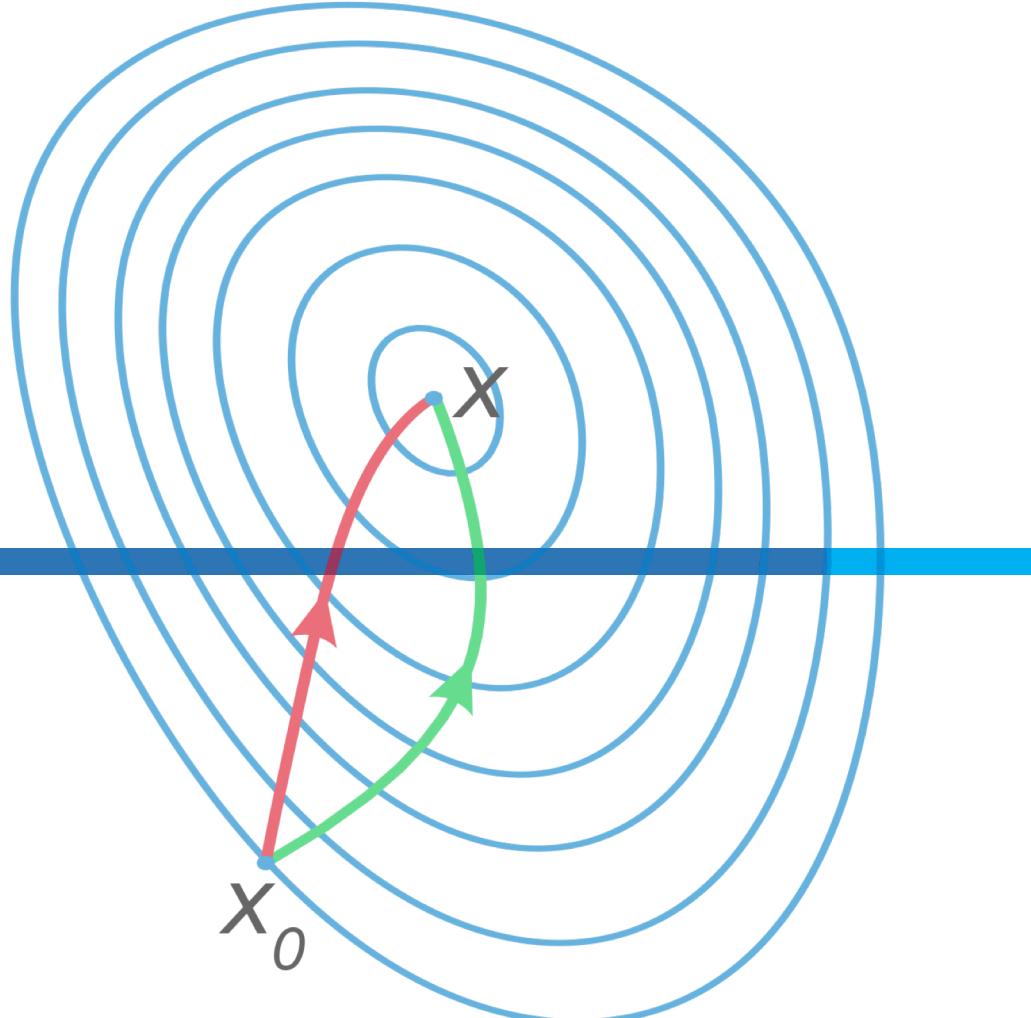


最优化方法

第一周

计算机学院
余皓然

2023/2/23



什么是最优化方法？

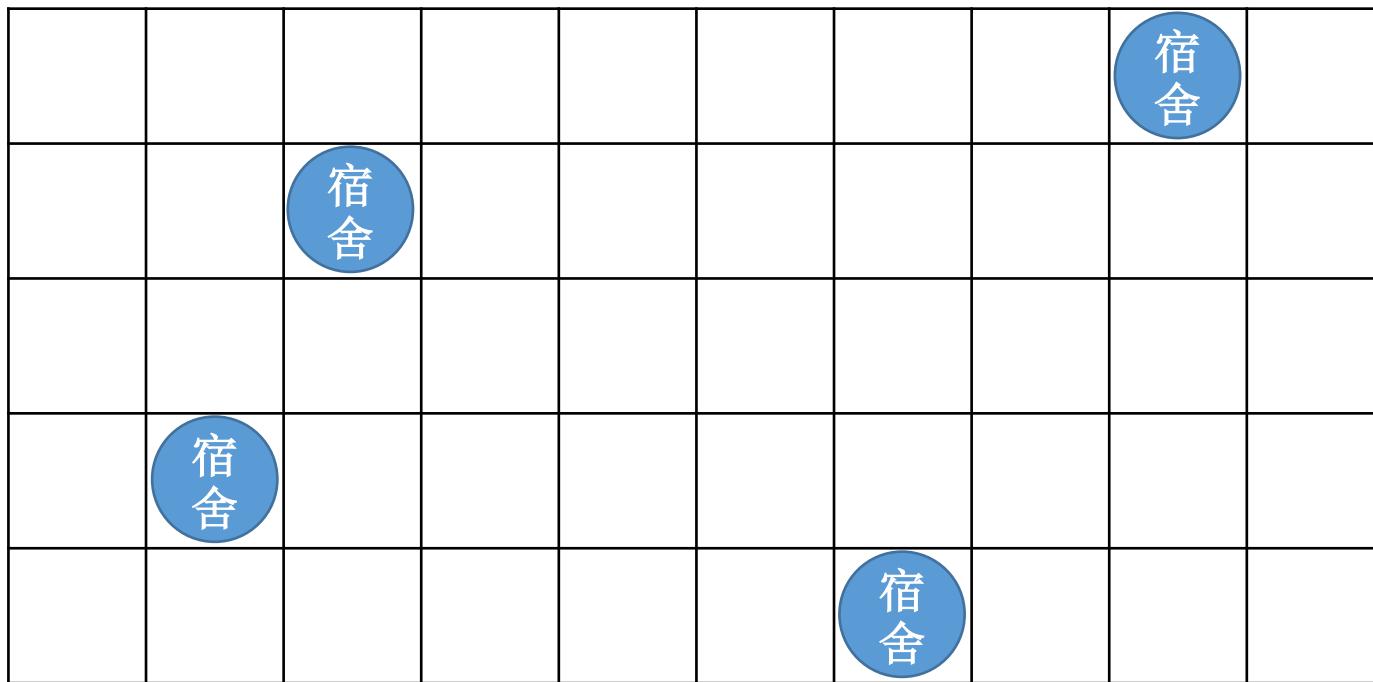


最优化方法

- ▶ 最优化方法：解决最优化问题的方法
- ▶ 最优化问题（Optimization Problem）：在一定约束条件下，如何选择变量，从而最大化/最小化预设目标

最优化问题示例

校区要新建两个超市，问如何选址可以最小化每个宿舍到离其最近超市的距离之和？



例子取自Harvard CS50's Introduction to Artificial Intelligence with Python课程

最优化问题示例

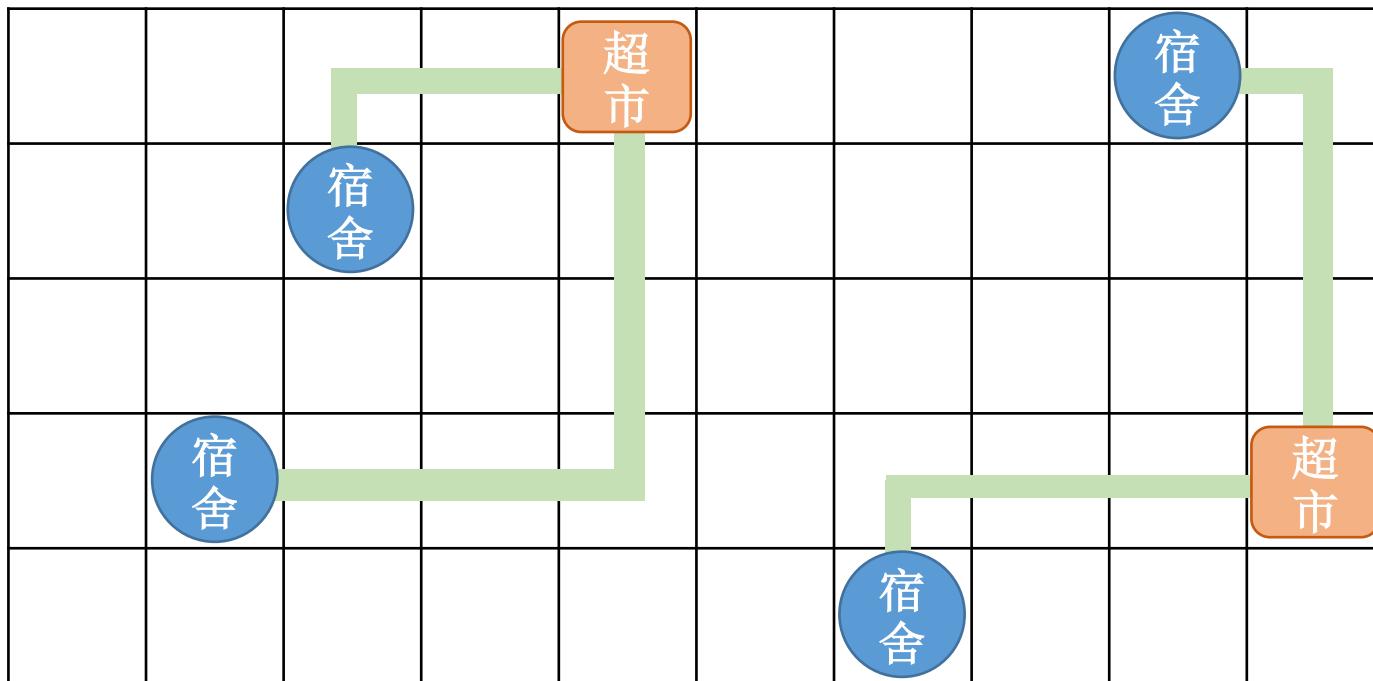
校区要新建两个超市，问如何选址可以最小化每个宿舍到离其最近超市的距离之和？



以上选址方案对应的每个宿舍到离其最近超市距离之和（目标值）为多少？

最优化问题示例

校区要新建两个超市，问如何选址可以最小化每个宿舍到离其最近超市的距离之和？

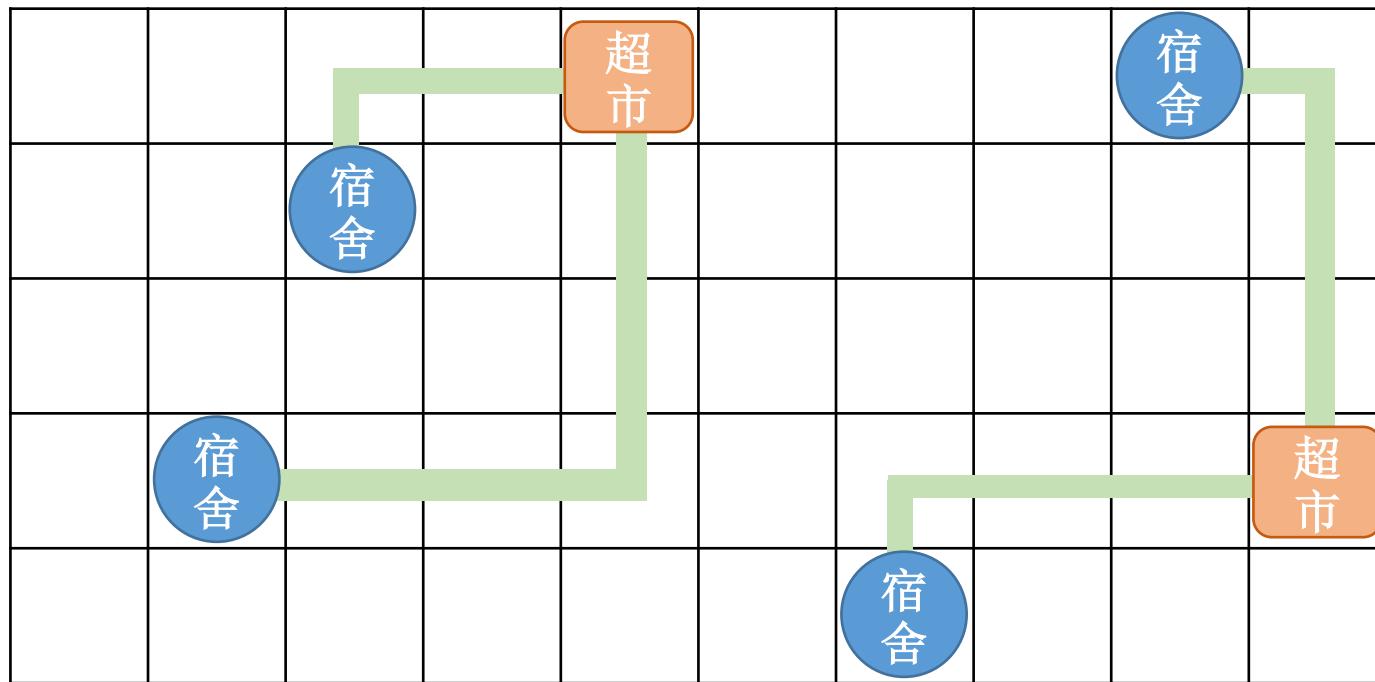


以上选址方案对应的目标值为17

最优化问题示例

最优化问题：在一定约束条件下，如何选择变量，从而最大化/最小化预设目标

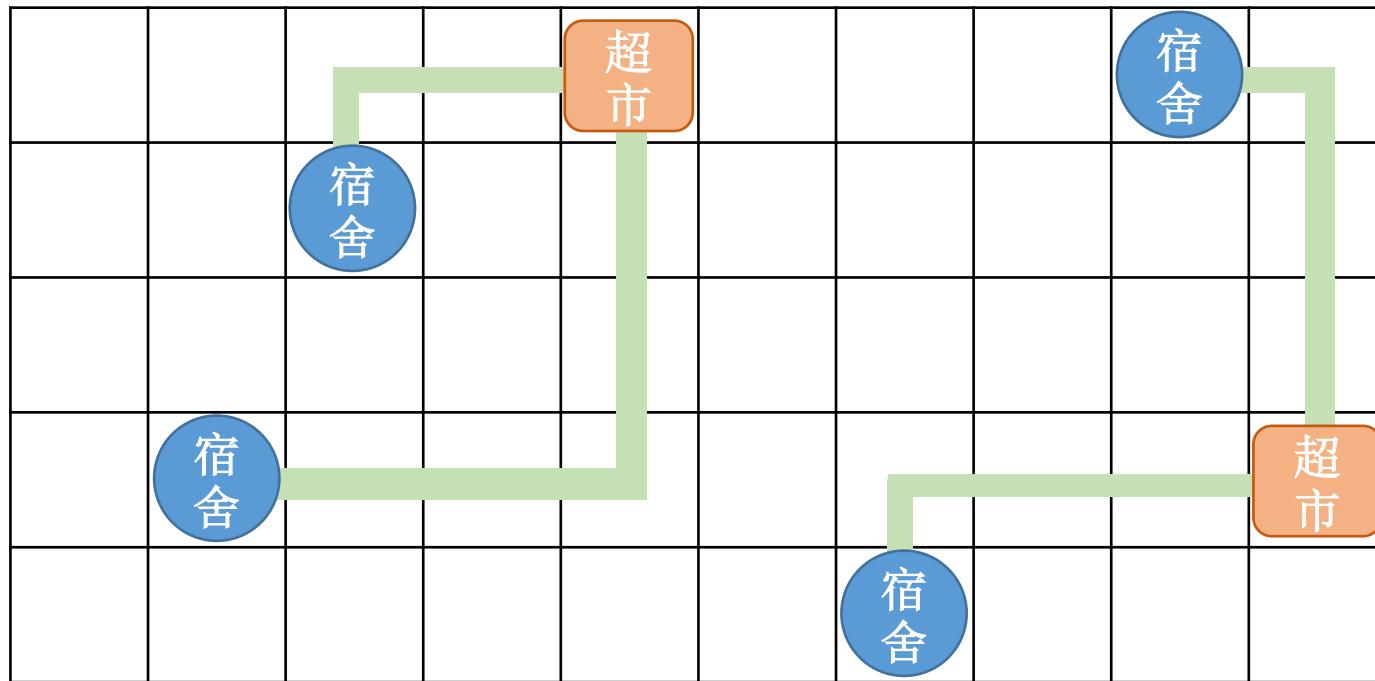
校区要新建两个超市，问如何选址可以最小化每个宿舍到离其最近超市的距离之和？



最优化方法示例



采用什么最优化方法求解?



最优化方法示例



采用什么最优化方法求解?
穷举法 (brute-force search/exhaustive search)?

1	2	3	4	5	6	7	8	宿舍	9
10	11	宿舍	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27	28
29	宿舍	30	31	32	33	34	35	36	37
38	39	40	41	42	43	宿舍	44	45	46

最优化方法示例



穷举法的缺点是什么？原因是什么？
能不能由此想到其它的方法？

1	2	3	4	5	6	7	8	宿舍	9
10	11	宿舍	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27	28
29	宿舍	30	31	32	33	34	35	36	37
38	39	40	41	42	43	宿舍	44	45	46

最优化方法示例



登山搜索：根据一定的方向在解空间进行搜索

1	2	3	4	5	6	7	8	宿舍	9
10	11	宿舍	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27	28
29	宿舍	30	31	32	33	34	35	36	37
38	39	40	41	42	43	宿舍	44	45	46

最优化方法示例

针对现有方案有很多种可能的更改方案，假设仅考虑将一个超市移动一格的更改方案，共6种该类更改方案

1	2	3	4超市	5超市	6超市	7	8	宿舍	9
10	11	宿舍	12	13超市	14	15	16	17	18
19	20	21	22	23	24	25	26	27	28超市
29	宿舍	30	31	32	33	34	35	36超市	37超市
38	39	40	41	42	43	宿舍	44	45	46超市

以上选址方案对应的目标函数值为17

最优化方法示例

从6种该类更改方案中选择对应目标函数最低的方案

1	2	3	4	5超市	6	7	8	宿舍	9
10	11	宿舍	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27	28
29	宿舍	30	31	32	33	34	35	36超市	37超市
38	39	40	41	42	43	宿舍	44	45	46

选址方案对应的目标函数值从17降低为15

最优化方法示例

针对新方案又有7种更改方案
(仍仅考虑将一个超市移动一格的更改方案)

1	2	3	4超市	5超市	6超市	7	8	宿舍	9
10	11	宿舍	12	13超市	14	15	16	17	18
19	20	21	22	23	24	25	26	27超市	28
29	宿舍	30	31	32	33	34	35超市	36超市	37超市
38	39	40	41	42	43	宿舍	44	45超市	46

最优化方法示例

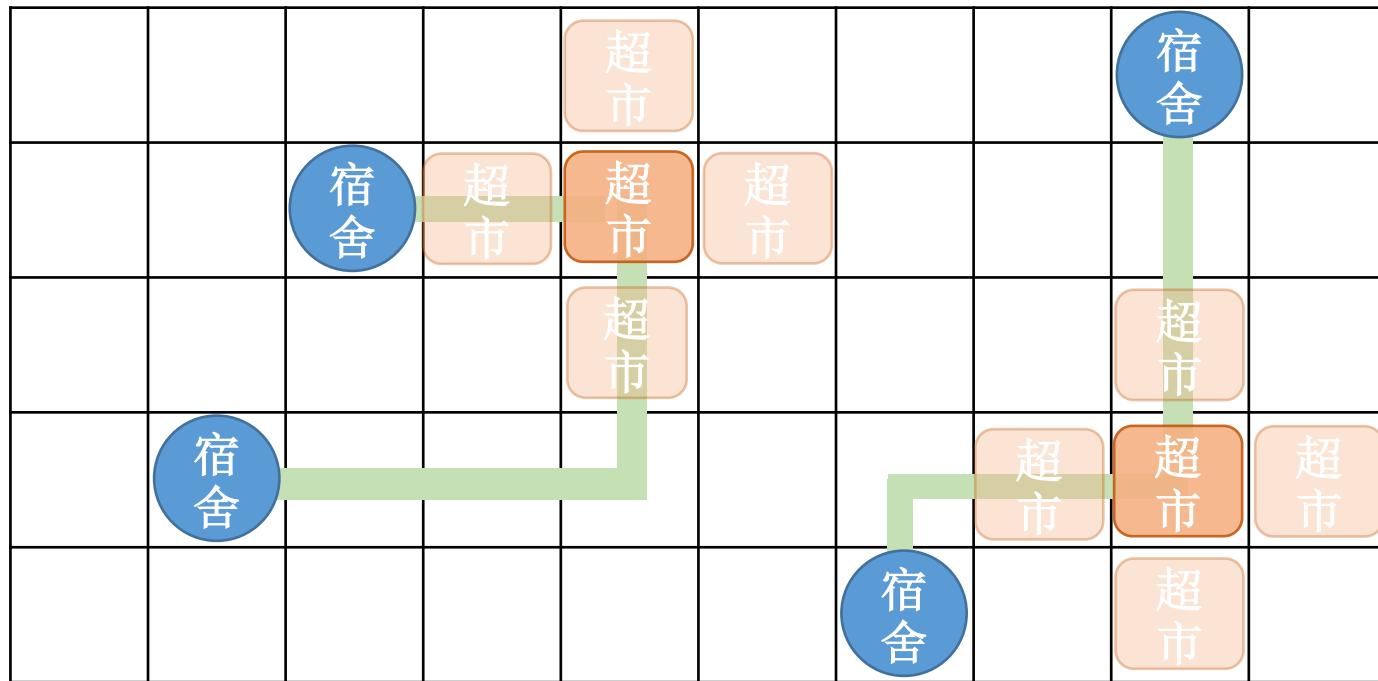
从7种该类更改方案中选择对应目标函数最低的方案

1	2	3	4	5 超 市	6	7	8	宿舍	9
10	11	宿舍	12	13 超 市	14	15	16	17	18
19	20	21	22	23	24	25	26	27	28
29	宿舍	30	31	32	33	34	35	36 超 市	37
38	39	40	41	42	43	宿舍	44	45	46

选址方案对应的目标函数值从15降低为13

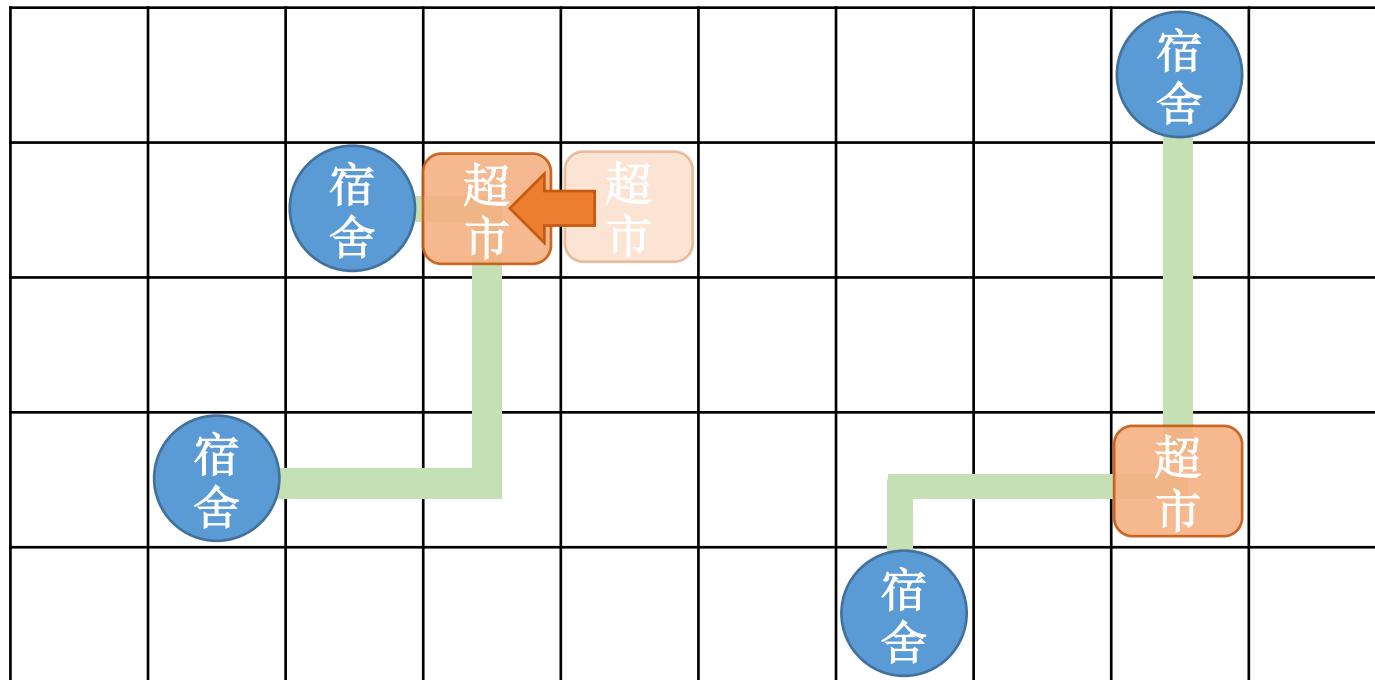
最优化方法示例

针对新方案又有8种更改方案
(仍仅考虑将一个超市移动一格的更改方案)



最优化方法示例

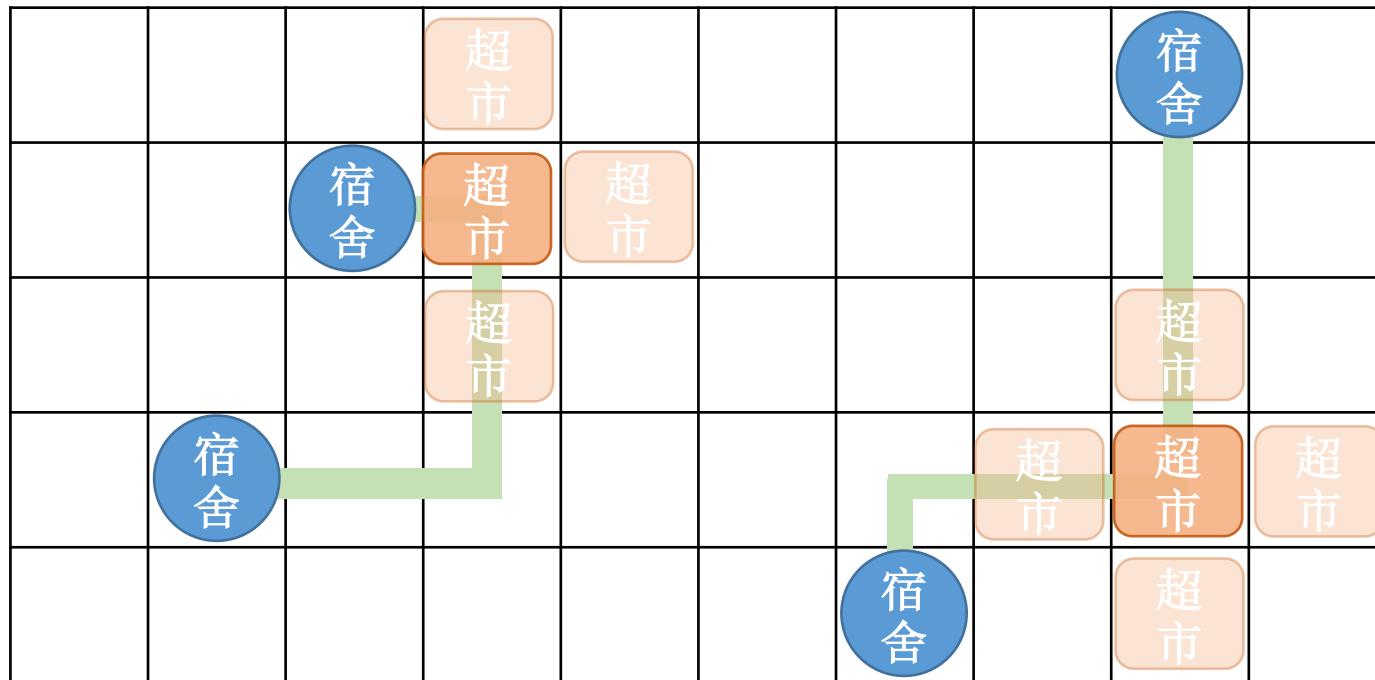
从8种该类更改方案中选择对应目标函数最低的方案



选址方案对应的目标函数值从13降低为11

最优化方法示例

针对新方案有7种更改方案，但都不能进一步降低目标函数

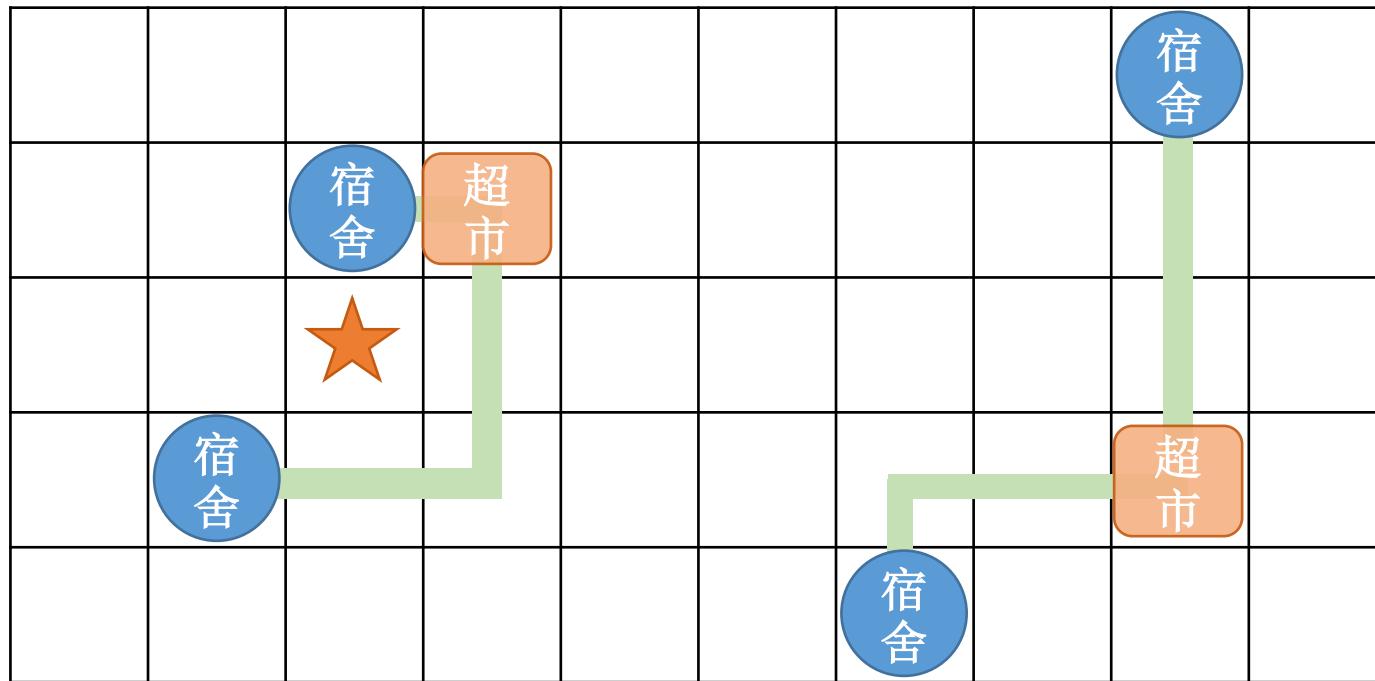


该方法停止搜索，输出结果，对应目标函数值为11

最优化方法示例



该算法是否找到了最优结果?
该算法优劣是什么? 是否存在更好的算法?



最优化问题



最优化问题广泛存在于各类应用场景



网约车APP

通过司乘匹配最小化总等待时长
(考虑优先级/…)

滴滴派单“全局最优”原则

派单方案①：

根据路况，两辆车行驶时间预计：

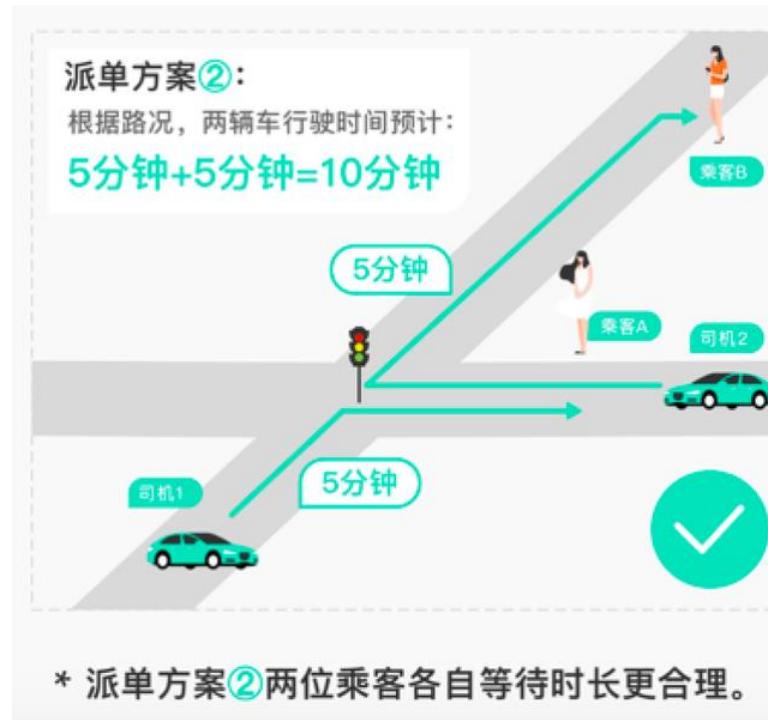
$$10\text{分钟} + 2\text{分钟} = 12\text{分钟}$$



派单方案②：

根据路况，两辆车行驶时间预计：

$$5\text{分钟} + 5\text{分钟} = 10\text{分钟}$$



* 派单方案②两位乘客各自等待时长更合理。

最优化问题

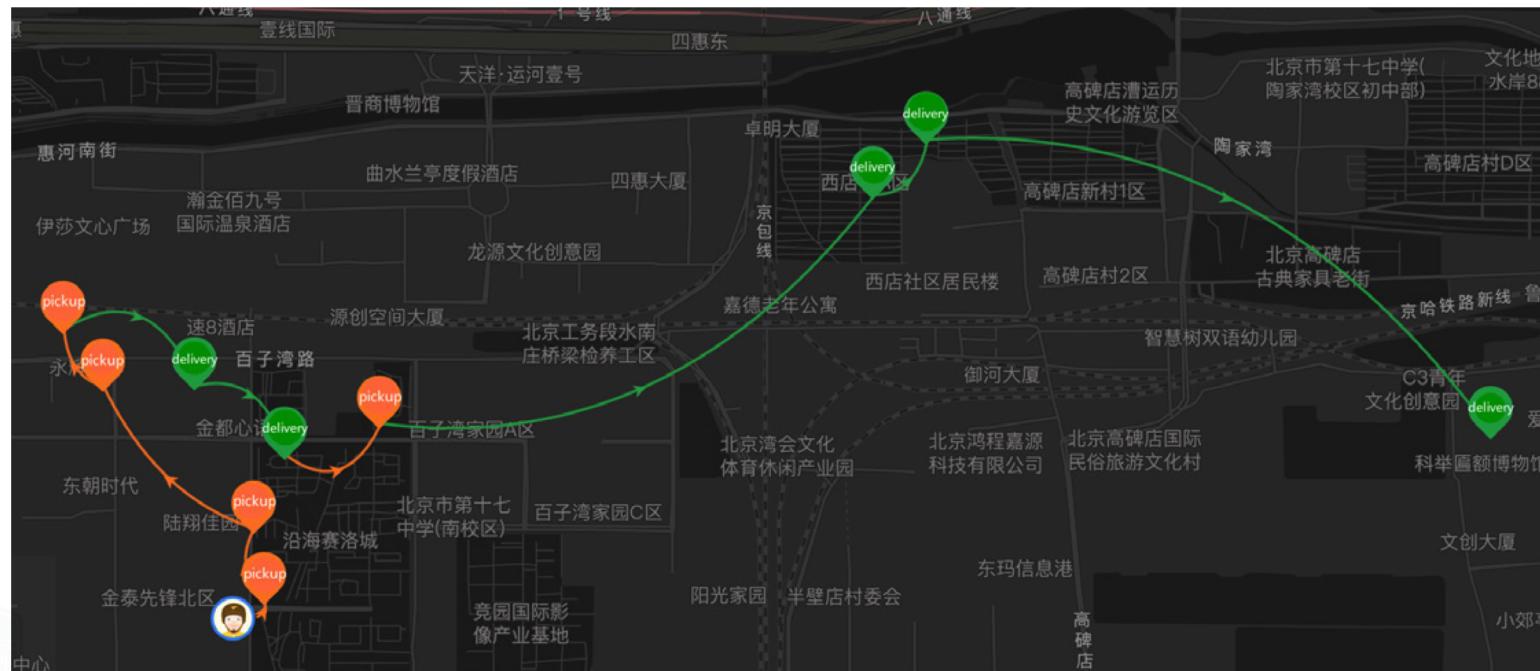


最优化问题广泛存在于各类应用场景



外卖APP

通过选择取餐/送餐顺序最小化总路程
(考虑顾客等待时间/出餐时间/…)



最优化问题



最优化问题广泛存在于各类应用场景



通讯APP



通过学习神经网络参数最小化损失
函数（翻译结果与真实标签的距离）



最优化问题



最优化问题广泛存在于各类应用场景



出行APP

通过动态定价最大化总利润
(考虑空房/空座量、未来需求量)

The screenshot shows the Qunar travel app interface for flight bookings. At the top, there are departure and arrival city inputs ('成都' to '澳门') and a date selector ('选择去程 03-05 周六'). Below this, a list of flight options is displayed:

出发地	到达地	时间	时长	价格
成都	澳门	10:05	2h45m	¥1197
成都	澳门	15:50	2h55m	¥1197
成都	澳门	15:50	2h55m	¥1752
成都	澳门	10:05	2h45m	¥1752
成都	澳门	15:30	2h55m	¥2220

Each flight row includes a small icon indicating the airline and flight number (e.g., '国航 CA5407'). The interface also features various filters and sorting options at the bottom.

最优化问题



最优化问题广泛存在于各类应用场景



本课程：讲授解决各类最优化问题的各类方法

课程大纲



Part1 如何从复杂度的角度衡量算法好坏？

算法复杂度、P问题与NP问题与NP难问题

因与《数据结构与算法设计》
内容有重叠，略讲

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

无约束凸优化问题、梯度下降法、牛顿法；
线性规划问题、单纯形法、内点法；
有约束凸优化问题、KKT条件

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

多目标优化问题、NSGA-II算法

课程大纲



Part1 如何从复杂度的角度衡量算法好坏？

算法复杂度、P问题与NP问题与NP难问题

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

无约束凸优化问题、梯度下降法、牛顿法；

线性规划问题、单纯形法、内点法；

有约束凸优化问题、KKT条件

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

多目标优化问题、NSGA-II算法

* 是2022年新增内容， 是2023年拟增内容，部分原有课程内容移至附录

* 2023年重点增加 应用实例及算法代码 相关内容

算法复杂度

Part1 如何从复杂度的角度衡量算法好坏？

算法复杂度、P问题与NP问题与NP难问题

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

算法复杂度

- ▶ 从计算的角度衡量一个计算机算法的效率
- ▶ 包括时间和空间复杂度：
 - ① 时间复杂度 → 计算所需的步数或指令条数
 - ② 空间复杂度 → 计算所需的存储空间大小

算法复杂度

- ▶ 算法时间复杂度是关于问题规模 n （或算法输入的大小 n ）的函数，描述计算所需步数随 n 增长的速度
- ▶ 什么是问题规模/输入大小？

排序问题: 为 x 个数进行排序

旅行商问题: 确定访问 x 个地点的顺序

线性规划问题: 解决包含 x 个变量、 y 个约束条件的线性规划问题

- 将算法时间复杂度写成关于 x 和 y 的函数
- 或者考察当 x （或 y ）固定时，关于 y （或 x ）的变化情况

算法复杂度

算法复杂度用大O符号表示，描述当n趋近于无穷时，复杂度的增长情况

例如，当问题规模为n时，某算法最多需要 $5n^3+3n$ 的计算步数运行完毕，那么该算法的时间复杂度是O(n^3)



算法复杂度

算法复杂度用大O符号表示，描述当n趋近于无穷时，复杂度的增长情况

例如，当问题规模为n时，某算法最多需要 $5n^3+3n$ 的计算步数运行完毕，那么该算法的时间复杂度是 $O(n^3)$

可以直接写作： $5n^3+3n=O(n^3)$, 当n趋近于 ∞

具体定义

if there exists a positive real number M and a real number x_0

$$|f(x)| \leq Mg(x) \quad \text{for all } x \geq x_0.$$

then we can write: $f(x) = O(g(x))$ as $x \rightarrow \infty$

算法复杂度

算法复杂度用大O符号表示，描述当n趋近于无穷时，复杂度的增长情况

例如，当问题规模为n时，某算法最多需要 $5n^3+3n$ 的计算步数运行完毕，那么该算法的时间复杂度是 $O(n^3)$

可以直接写作： $5n^3+3n=O(n^3)$, 当n趋近于 ∞

具体定义

if there exists a positive real number M and a real number x_0

$$|f(x)| \leq Mg(x) \quad \text{for all } x \geq x_0.$$

then we can write: $f(x) = O(g(x))$ as $x \rightarrow \infty$

例如，可以写 $5n^2+n\log n=O(n^2)$, 当n趋近于 ∞

算法复杂度

Comparison sorts [edit]

Below is a table of comparison sorts. A comparison sort cannot perform better than $O(n \log n)$.^[4]

为什么有不同的时间复杂度？

空间复杂度

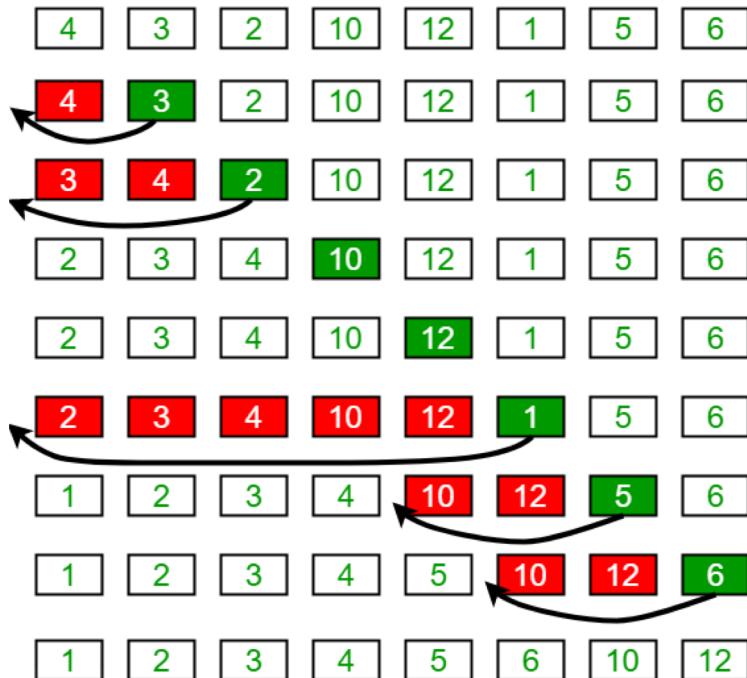
Comparison sorts

Name	Best	Average	Worst	Memory	Stable	Method
Quicksort	$n \log n$	$n \log n$	n^2	$\log n$	No	Partitioning
Merge sort	$n \log n$	$n \log n$	$n \log n$	n	Yes	Merging
In-place merge sort	—	—	$n \log^2 n$	1	Yes	Merging
Introsort	$n \log n$	$n \log n$	$n \log n$	$\log n$	No	Partitioning & Selection
Heapsort	$n \log n$	$n \log n$	$n \log n$	1	No	Selection
Insertion sort	n	n^2	n^2	1	Yes	Insertion
Block sort	n	$n \log n$	$n \log n$	1	Yes	Insertion & Merging
Quicksort	n	$n \log n$	$n \log n$	n	Yes	Merging
Timsort	n	$n \log n$	$n \log n$	n	Yes	Insertion & Merging
Selection sort	n^2	n^2	n^2	1	No	Selection

算法复杂度

插入排序的时间算法复杂度？

Insertion Sort Execution Example



1. for $i = 1$ to $n-1$
2. key = $A[i]$
3. $j = i - 1$
4. while $j \geq 0$ and $A[j] > key$
5. $A[j + 1] = A[j]$
6. $j = j - 1$
7. end while
8. $A[j + 1] = key$
9. end for

算法复杂度

插入排序的时间算法复杂度？

```
1. for i = 1 to n-1  
2.   key = A[i]  
3.   j = i - 1  
4.   while j >= 0 and A[j] > key  
5.     A[j + 1] = A[j]  
6.     j = j - 1  
7.   end while  
8. A[j + 1] = key  
9. end for
```

时间消耗	指令执行次数
C_1	$n - 1$
C_2	$n - 1$
C_3	$n - 1$
C_4	$\sum_{i=1}^{n-1} (1 + t_i)$
C_5	$\sum_{i=1}^{n-1} t_i$
C_6	$\sum_{i=1}^{n-1} t_i$
C_8	$n - 1$

t_i 是在外层循环的第*i*次循环中，内层循环的次数

算法复杂度

插入排序的时间算法复杂度？

```
1. for i = 1 to n-1  
2.   key = A[i]  
3.   j = i - 1  
4.   while j >= 0 and A[j] > key  
5.     A[j + 1] = A[j]  
6.     j = j - 1  
7.   end while  
8. A[j + 1] = key  
9. end for
```

时间消耗	指令执行次数
C_1	$n - 1$
C_2	$n - 1$
C_3	$n - 1$
C_4	$\sum_{i=1}^{n-1} (1 + t_i)$
C_5	$\sum_{i=1}^{n-1} t_i$
C_6	$\sum_{i=1}^{n-1} t_i$
C_8	$n - 1$

best case: $t_i=0$

即 $(C_1+C_2+C_3+C_4+C_8)(n-1)$

worst case: $t_i=i$

即 $(C_1+C_2+C_3+C_8)(n-1)+C_4(2+n)(n-1)/2+(C_5+C_6)(n-1)n/2$

t_i 是在外层循环的第*i*次循环中，内层循环的次数

算法复杂度

Name	Best	Average	Worst	Memory	Stable	Method
Insertion sort	n	n^2	n^2	1	Yes	Insertion

插入排序的时间算法复杂度？空间复杂度？

```
1. for i = 1 to n-1
2.   key = A[i]
3.   j = i - 1
4.   while j >= 0 and A[j] > key
5.     A[j + 1] = A[j]
6.     j = j - 1
7.   end while
8. A[j + 1] = key
9. end for
```

时间消耗	指令执行次数
C_1	$n - 1$
C_2	$n - 1$
C_3	$n - 1$
C_4	$\sum_{i=1}^{n-1} (1 + t_i)$
C_5	$\sum_{i=1}^{n-1} t_i$
C_6	$\sum_{i=1}^{n-1} t_i$
C_8	$n - 1$

best case: $t_i=0$

即 $(C_1+C_2+C_3+C_4+C_8)(n-1)$

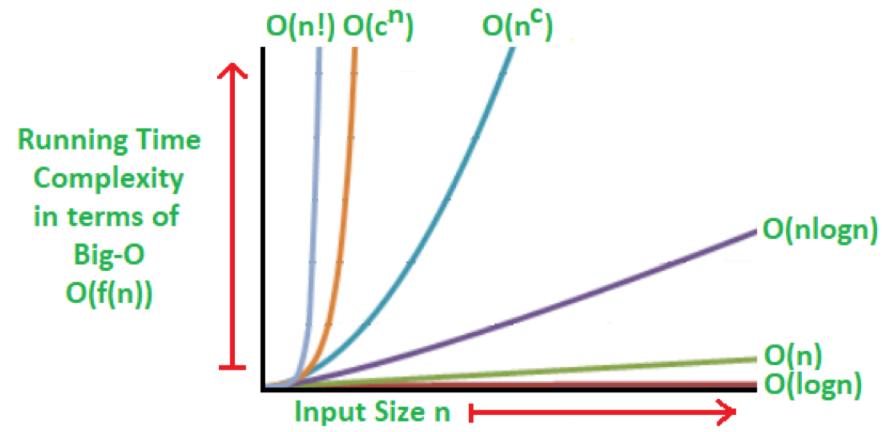
worst case: $t_i=i$

即 $(C_1+C_2+C_3+C_8)(n-1)+C_4(2+n)(n-1)/2+(C_5+C_6)(n-1)n/2$

t_i 是在外层循环的第*i*次循环中，内层循环的次数

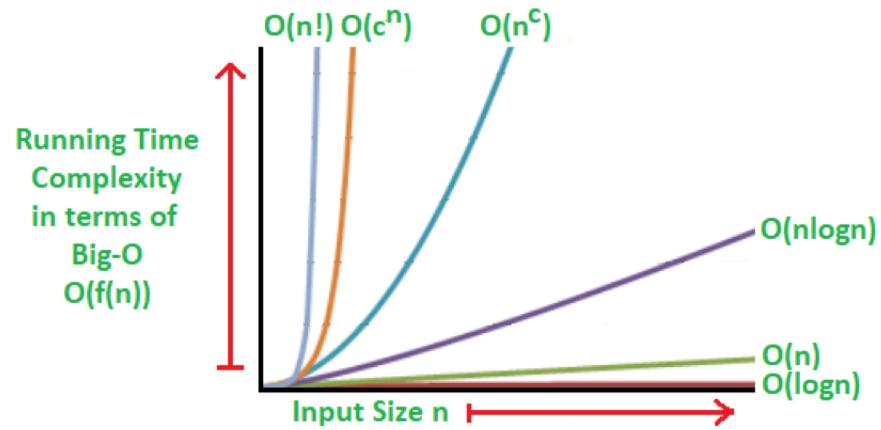
算法复杂度

$O(1)$	constant time
$O(\log n)$	logarithmic time
$O(n)$	linear time
$O(n \log n)$	linearithmic time
$O(n^2)$	quadratic time
$O(n^c)$	polynomial time
$O(c^n)$	exponential time
$O(n!)$	factorial time
$O(\infty)$	infinite time



算法复杂度

$O(1)$	constant time
$O(\log n)$	logarithmic time
$O(n)$	linear time
$O(n \log n)$	linearithmic time
$O(n^2)$	quadratic time
$O(n^c)$	polynomial time
$O(c^n)$	exponential time
$O(n!)$	factorial time
$O(\infty)$	infinite time



假设针对一个问题有四种算法，用每秒百万次的计算机运行

复杂性函数	问题规模n		
	10	30	60
n	0.01ms	0.03ms	0.06ms
n^3	1ms	27ms	216ms
n^5	100ms	24.3s	13min
2^n	1ms	17.9min	366世纪

P、NP、NP难问题

Part1 如何从复杂度的角度衡量算法好坏？

算法复杂度、**P问题与NP问题与NP难问题**

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

P问题

P: Polynomial time

可以找到一个在多项式时间之内（包含多项式时间）解决它的算法

$O(1)$	constant time
$O(\log n)$	logarithmic time
$O(n)$	linear time
$O(n \log n)$	linearithmic time
$O(n^2)$	quadratic time
$O(n^c)$	polynomial time
$O(c^n)$	exponential time
$O(n!)$	factorial time
$O(\infty)$	infinite time

P问题

例如：计算两个n位数的乘积

至少有一种算法的时间复杂度为 $O(n^2)$
(其实还存在其他更快的算法)

$$\begin{array}{r} 23958233 \\ \times \quad 5830 \\ \hline 00000000 \quad (= \quad 23,958,233 \times \quad 0) \\ 71874699 \quad (= \quad 23,958,233 \times \quad 30) \\ 191665864 \quad (= \quad 23,958,233 \times \quad 800) \\ + 119791165 \quad (= \quad 23,958,233 \times 5,000) \\ \hline 139676498390 \quad (= 139,676,498,390) \end{array}$$

```
multiply(a[1..p], b[1..q], base)
product = [1..p+q]
for b_i = 1 to q
    carry = 0
    for a_i = 1 to p
        product[a_i + b_i - 1] += carry + a[a_i] * b[b_i]
        carry = product[a_i + b_i - 1] / base
        product[a_i + b_i - 1] = product[a_i + b_i - 1] mod base
    product[b_i + p] = carry
return product
```

*// Operands containing rightmost digits at index 1
// Allocate space for result
// for all digits in b

// for all digits in a

// last digit comes from final carry*

P问题

例如：对n个数字排序

Comparison sorts [edit]

Below is a table of [comparison sorts](#). A comparison sort cannot perform better than $O(n \log n)$.^[4]

Comparison sorts

Name	Best	Average	Worst	Memory	Stable	Method
Quicksort	$n \log n$	$n \log n$	n^2	$\log n$	No	Partitioning
Merge sort	$n \log n$	$n \log n$	$n \log n$	n	Yes	Merging
In-place merge sort	—	—	$n \log^2 n$	1	Yes	Merging
Introsort	$n \log n$	$n \log n$	$n \log n$	$\log n$	No	Partitioning & Selection
Heapsort	$n \log n$	$n \log n$	$n \log n$	1	No	Selection
Insertion sort	n	n^2	n^2	1	Yes	Insertion
Block sort	n	$n \log n$	$n \log n$	1	Yes	Insertion & Merging
Quicksort	n	$n \log n$	$n \log n$	n	Yes	Merging
Timsort	n	$n \log n$	$n \log n$	n	Yes	Insertion & Merging
Selection sort	n^2	n^2	n^2	1	No	Selection

NP问题



P问题：能在多项式的时间之内被解决

错误：NP问题指不能在多项式时间之内被解决的问题



NP问题：能在多项式的时间之内验证一个解是否正确的问题

NP: Nondeterministic Polynomial



NP问题



P问题：能在多项式的时间之内被解决



错误：NP问题指不能在多项式时间之内被解决的问题



NP问题：能在多项式的时间之内验证一个解是否正确的问题

1	5			8	4	9
	3	7		1	2	
8	9	6		3	7	
			3			4
4	3	1	6	2	7	9
	6					1
7				8		
9	8	2	5		1	4
5		8				

数独属于NP问题

NP问题



P问题：能在多项式的时间之内被解决



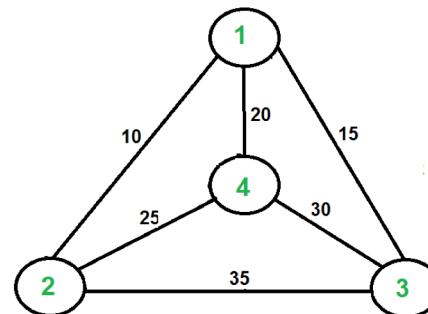
错误：NP问题指不能在多项式时间之内被解决的问题



NP问题：能在多项式的时间之内验证一个解是否正确的问题

旅行商问题的判定问题：给定n个城市，任意两个城市间有路相连且路程距离已知。找到一条总长度小于L的路径（从一个城市出发、不重复的遍历所有的城市并回到起点）

（旅行商问题的判定问题与旅行商问题不同）



NP问题



P问题：能在多项式的时间之内被解决



错误：NP问题指不能在多项式时间之内被解决的问题



NP问题：能在多项式的时间之内验证一个解是否正确的问题

两种问题之间的关系？

千禧年大奖难题

美国克雷研究所，每题100万美金

List of Unsolved Millennium Prize Problems

- P versus NP. **P与NP问题**
- Hodge conjecture. **霍奇猜想**
- Riemann hypothesis. **黎曼猜想**
- Yang–Mills existence and mass gap. **杨-米尔斯理论**
- Navier–Stokes existence and smoothness. **纳维叶-斯托克斯方程**
- Birch and Swinnerton-Dyer Conjecture. **贝赫和思维讷通-戴尔猜想**
- ~~Poincaré conjecture.~~ **庞加莱猜想**

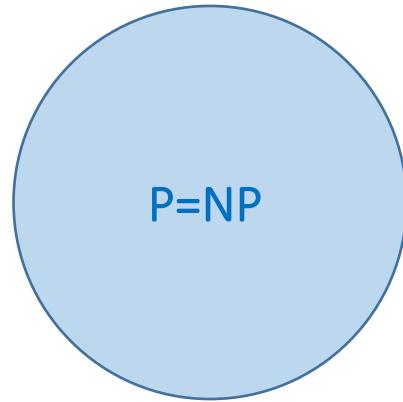
P vs NP

P问题：能在多项式时间内被解决

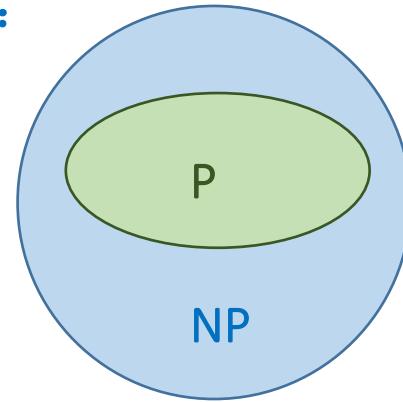
NP问题：能在多项式时间内验证解的有效性

易得所有的P问题都属于NP问题，是不是所有的NP问题也都属于P问题？

如果是：

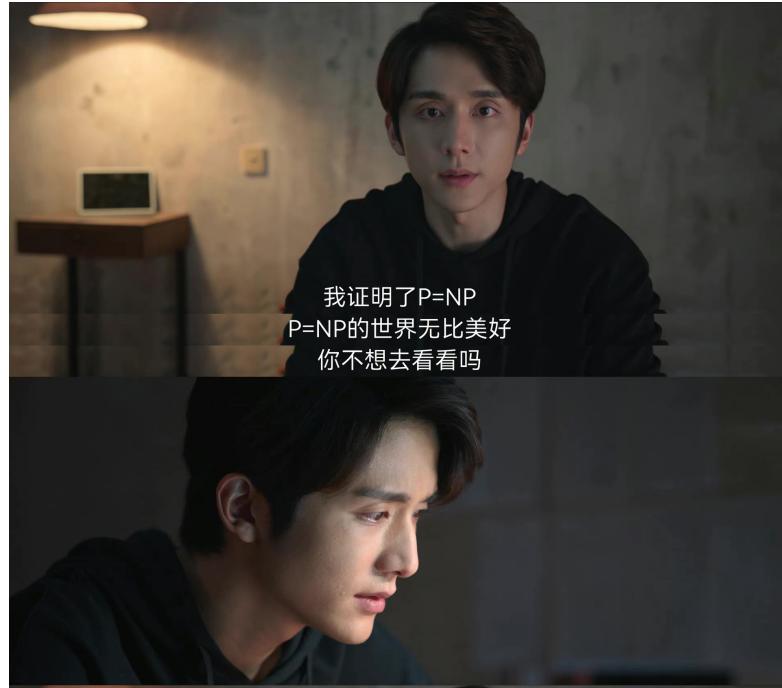


如果不是：



至今尚未被解决

P vs NP



P vs NP

是不是所有的NP问题也都属于P问题？

STEPHEN COOK

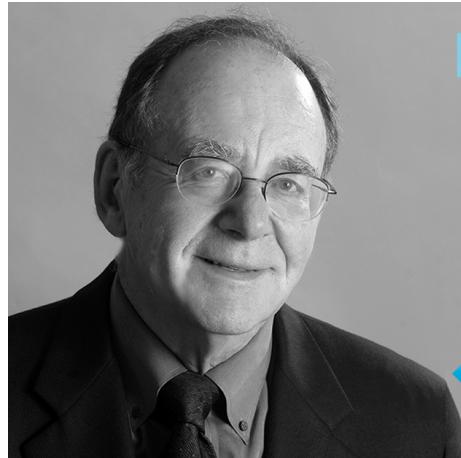


Laid the foundations
for the theory of
NP-Completeness

A.M.
TURING 1982
AWARD



斯蒂芬库克



RICHARD KARP

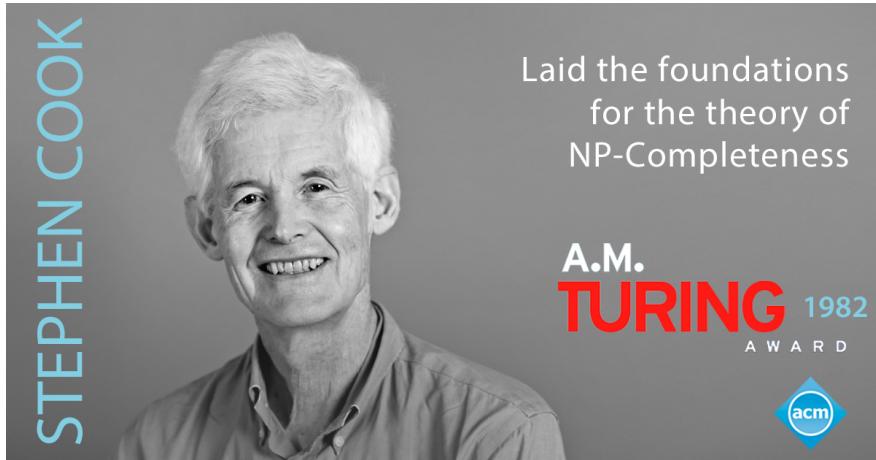
Developed efficient
algorithms for network
flow, introduced
NP-complete

A.M.
TURING 1985
AWARD

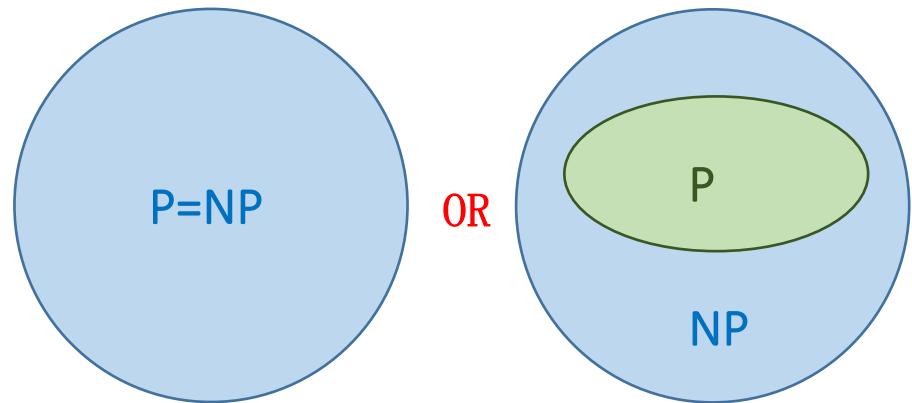
理查德卡普

NP完全问题

是不是所有的NP问题也都属于P问题？



斯蒂芬库克



思路：找到NP问题中最难的一类问题，试着证明它们是P问题，如果成立，那么可以说其余NP问题也是P问题

NP完全问题

思路：找到NP问题中最难的一类问题，试着证明它们是P问题，如果成立，那么可以说其余NP问题也是P问题

定义NP完全问题（NP-Complete Problems），即同时满足：

- (1) NP问题；
- (2) 所有NP问题可在多项式时间内归约（be reducible to）到该问题

NP完全问题

归约：若A问题可以归约到B问题，那么B问题至少和A问题一样难，任何可以解决B问题的算法可以用于解决A问题

例子：对一元一次方程的求解问题可以归约到对二元一次方程组的求解问题

对于任意一元一次方程 $a_1x + b_1 = 0$, 能构造出二元一次方程组
 $m_{11}x + n_{11}y + k_{11} = 0$ 和 $m_{21}x + n_{21}y + k_{21} = 0$,
令 $m_{11} - m_{21} = a_1, n_{11} - n_{21} = 0, k_{11} - k_{21} = b_1$ 。

当求解到该二元一次方程组的解时，该解也是对应一元一次方程的解。

如：对 $2x + 3 = 0$ 可以构造 $3x + y + 6 = 0$ 与 $x + y + 3 = 0$

NP完全问题

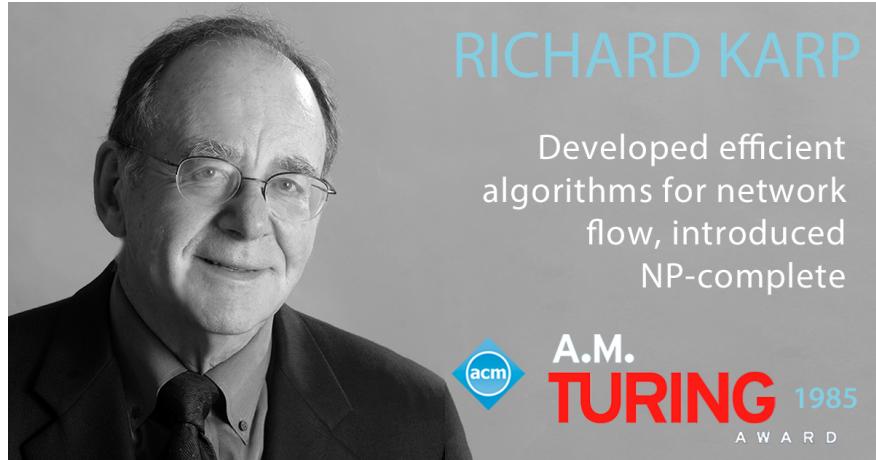
P问题：能在多项式时间内被解决

NP问题：能在多项式时间内验证解的有效性

NP完全问题：所有NP问题可于多项式时间内归约到的NP问题



NP完全问题



理查德卡普

1972年，在“Reducibility Among Combinatorial Problems”中，
提出21个NP完全问题



NP完全问题

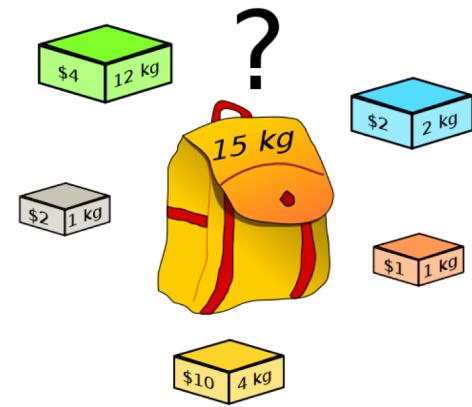
NP完全问题举例：

旅行商问题的判定问题：给定 n 个城市，任意两个城市间有路相连且路程距离已知。找到一条总长度小于 L 的路径（从一个城市出发、不重复的遍历所有的城市并回到起点）

NP完全问题

NP完全问题举例：

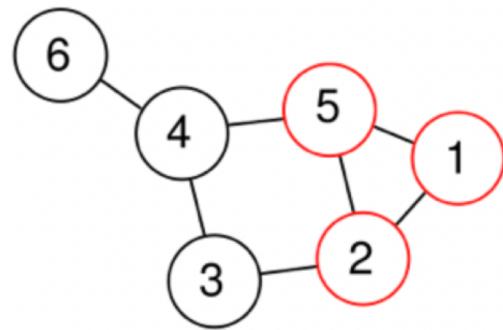
0-1背包问题的判定问题：给定一组物品，已知每个物品的重量和价值，问给定一个背包，是否有方案使得装进背包的物品的重量不超过W、价值超过V？



NP完全问题

NP完全问题举例：

分团问题的判定问题：给定一个图（包含若干顶点和边），是否有团使得团的顶点数目不小于K？



NP完全问题

迄今，还没有证明任何NP完全问题属于P问题（即能在多项式时间内被解决）

如果能证明某NP完全问题属于P问题：

任何能在多项式时间内验证解的有效性的问题都能在多项式时间内被解决

整数规划、组合优化等难题有望被高效解决。现实中，大到公路、铁路、城市规划，小到提升送餐APP配送满意度等问题都有望被妥善的解决。

NP难问题

P问题：能在多项式时间内被解决

NP问题：能在多项式时间内验证解的有效性

NP完全（NP-Complete）问题：所有NP问题可于多项式时间内归约到的**NP问题**

NP难（NP-Hard）问题：所有NP问题可于多项式时间内归约到的问题

NP难题与NP完全问题对比

- NP完全问题是NP问题；NP难题不用是NP问题
- NP完全问题一般是判定类问题，NP难题不用是判定类问题

NP难问题

NP完全问题

旅行商问题的判定问题：给定 n 个城市，任意两个城市间有路相连且路程距离已知。找到一条总长度小于 L 的路径（从一个城市出发、不重复的遍历所有的城市并回到起点）

NP难问题

旅行商问题：给定 n 个城市，任意两个城市间有路相连且路程距离已知。找到总长度最小的路径（从一个城市出发、不重复的遍历所有的城市并回到起点）

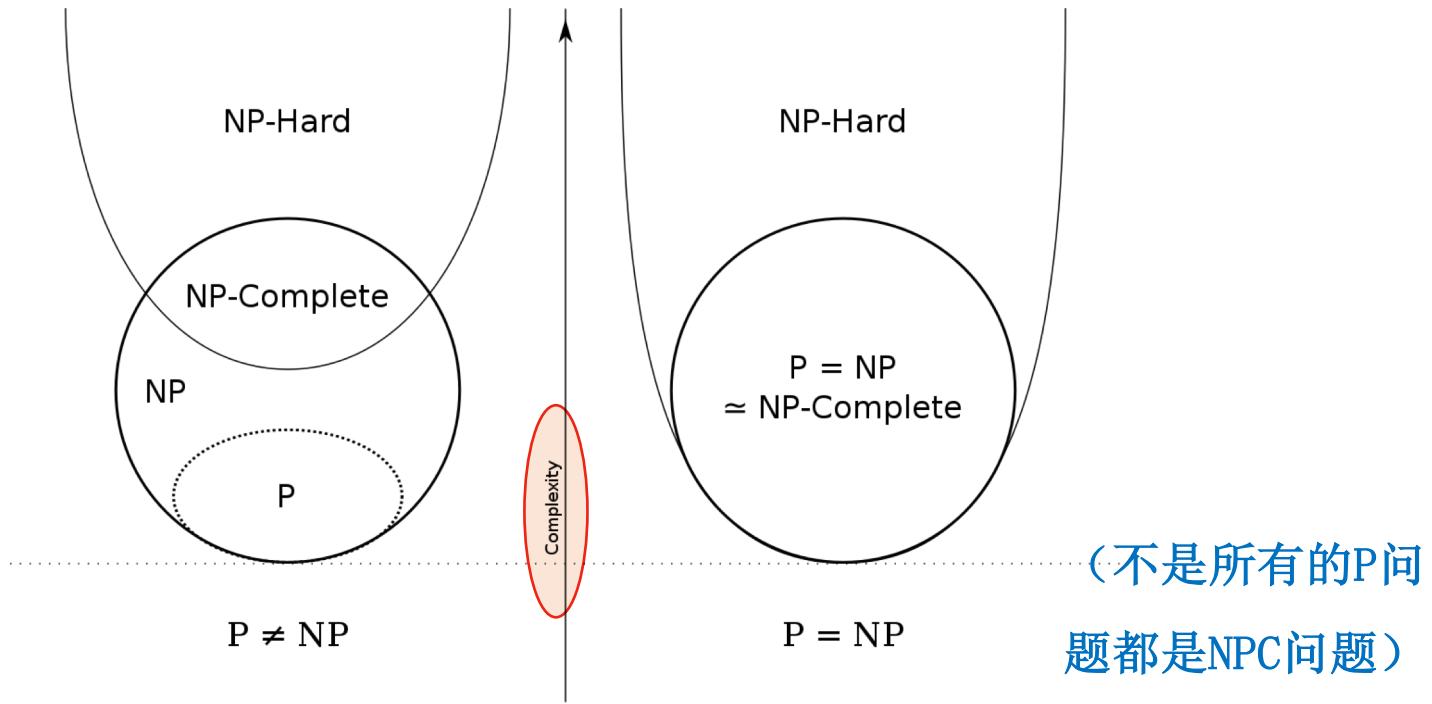
NP难问题

P问题：能在多项式时间内被解决

NP问题：能在多项式时间内验证解的有效性

NP完全问题：所有NP问题可于多项式时间内归约到的NP问题

NP难问题：所有NP问题可于多项式时间内归约到的问题



NP难问题

► 关于更多问题类别的定义



“世界七大数学难题之一: P与NP复杂度问题”

https://www.bilibili.com/video/av19085452?from=search&seid=13679117091167028644&spm_id_from=333.337.0.0

(7分55秒开始)

NP难问题



证明一个问题是否是NP-Complete或NP-Hard的目的是什么？

—— 公认接受通过智能优化算法找局部最优解来解决问题

最优化问题的一般数学形式

Part1 如何从复杂度的角度衡量算法好坏？

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

无约束凸优化问题、梯度下降法、牛顿法；
线性规划问题、单纯形法、内点法；
有约束凸优化问题、KKT条件

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

最优化问题



如何用数学符号表示最优化问题的一般形式

例如，一元二次方程的一般形式：

THE QUADRATIC FORMULA:

*For any quadratic equation in the form
 $ax^2 + bx + c = 0$, the solution is*

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

*where : a and b are the coefficients of the x^2 and x terms, respectively
 c is the constant*

例子

把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

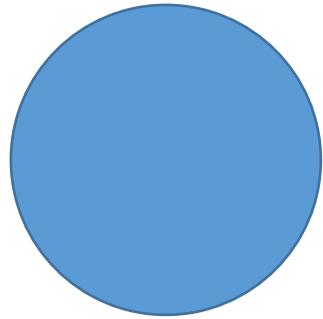


例子

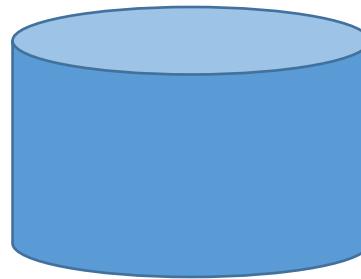
把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

用数学语言描述问题：定义参数和决策变量

实心金属球半径R



圆柱底半径 r、圆柱高h



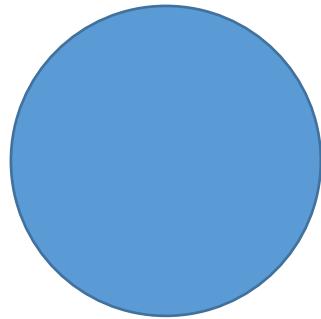
参数是固定的、用于描述问题： R

决策变量是可以调节的、用于描述对问题的解： r、 h

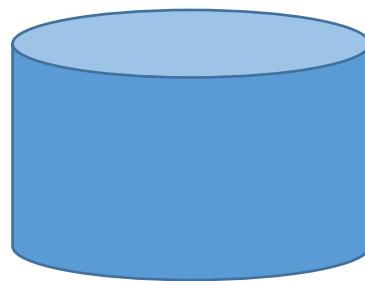
例子

把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

实心金属球半径R



圆柱底半径 r、圆柱高h



关键条件：金属球和圆柱体的质量相同 \Rightarrow 二者体积相同

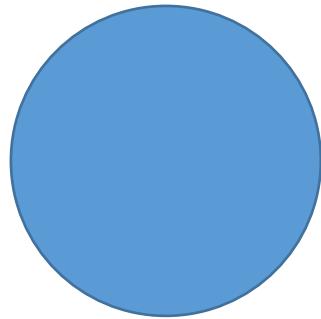
$$\frac{4}{3}\pi R^3 = \pi r^2 h$$



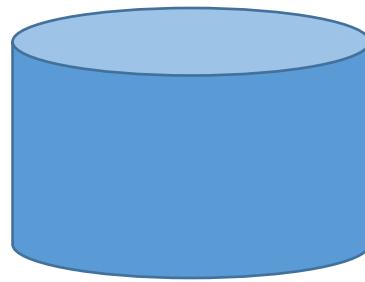
例子

把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

实心金属球半径R



圆柱底半径 r、圆柱高h



目标：最小化圆柱体的表面积 $2\pi r^2 + 2\pi r h$



例子

把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

对圆柱取尺寸的问题可以由如下数学问题表示：

minimize

$$\min 2\pi r^2 + 2\pi r h$$

subject to

$$\text{s.t. } \frac{4}{3}\pi R^3 = \pi r^2 h$$

variable(s)

$$\text{var. } r > 0, h > 0.$$

目标函数

约束条件

决策变量

例子

把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

$$\begin{aligned} \min \quad & 2\pi r^2 + 2\pi r h \\ \text{s.t.} \quad & \frac{4}{3}\pi R^3 = \pi r^2 h \\ \text{var.} \quad & r > 0, h > 0. \end{aligned}$$

把约束条件化简，可以得到r和h的关系

利用该关系将问题转化成一个关于单个决策变量的问题

通过求解得到答案

例子

把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

$$\begin{aligned} \min \quad & 2\pi r^2 + 2\pi r h \\ \text{s.t.} \quad & \frac{4}{3}\pi R^3 = \pi r^2 h \\ \text{var.} \quad & r > 0, h > 0. \end{aligned}$$

$$\begin{aligned} \min \quad & 2\pi r^2 + 2\pi \frac{4R^3}{3r} \\ \text{var.} \quad & r > 0. \end{aligned}$$

分析新问题中目标函数关于r单调性（一阶导、二阶导）

可得目标函数关于r先减后增，易得最优r值

代回原约束得最优h值

例子

把一个实心金属球熔化后，铸成一个实心圆柱体，问圆柱体取什么尺寸才能使它的表面积最小？

$$\begin{aligned} \min \quad & 2\pi r^2 + 2\pi r h \\ \text{s.t.} \quad & \frac{4}{3}\pi R^3 = \pi r^2 h \\ \text{var.} \quad & r > 0, h > 0. \end{aligned}$$



$$\begin{aligned} \min \quad & 2\pi r^2 + 2\pi \frac{4R^3}{3r} \\ \text{var.} \quad & r > 0. \end{aligned}$$

$$r^* = \sqrt[3]{\frac{2}{3}}R, h^* = 2 \cdot \sqrt[3]{\frac{2}{3}}R$$

最优化问题一般形式

最优化问题三要素

$$\begin{array}{ll}\min & 2\pi r^2 + 2\pi r h \\ \text{s.t.} & \frac{4}{3}\pi R^3 = \pi r^2 h \\ \text{var.} & r > 0, h > 0.\end{array}$$

目标函数

约束条件

决策变量

最优化问题一般形式

最优化问题三要素

$$\begin{array}{ll}\min & 2\pi r^2 + 2\pi r h \\ \text{s.t.} & \frac{4}{3}\pi R^3 = \pi r^2 h \\ \text{var.} & r > 0, h > 0.\end{array}$$

目标函数

约束条件

决策变量

$$\begin{array}{ll}\min & f(\boldsymbol{x}) \\ \text{s.t.} & g_i(\boldsymbol{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束} \\ & h_i(\boldsymbol{x}) = 0, j = 1, \dots, p, \text{ 等式约束} \\ \text{var.} & \boldsymbol{x} \in \mathcal{D}. \text{ 可行域} \\ & \boldsymbol{x} \text{ 是向量}\end{array}$$

一般（单目标）最优化问题都可整理成以上形式

最优化问题一般形式

目标函数

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

约束条件

$$\text{s.t. } g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束}$$

决策变量

$$h_i(\mathbf{x}) = 0, j = 1, \dots, p, \text{ 等式约束}$$

$$\text{var. } \mathbf{x} \in \mathcal{D}. \text{ 可行域}$$

注意1：可行域 (feasible region) 有时可以化入约束条件，反之亦然

$$\min x + y^2 + xy$$

$$\text{s.t. } x^3 + y \geq 0,$$

$$\text{var. } x \geq 0, y \leq 1.$$



$$\min x + y^2 + xy$$

$$\text{s.t. } x^3 + y \geq 0,$$

$$x \geq 0,$$

$$y \leq 1,$$

$$\text{var. } x, y.$$

最优化问题一般形式

目标函数

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

约束条件

$$\text{s.t. } g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束}$$

决策变量

$$h_i(\mathbf{x}) = 0, j = 1, \dots, p, \text{ 等式约束}$$

$$\text{var. } \mathbf{x} \in \mathcal{D}. \text{ 可行域}$$

注意1：可行域 (feasible region) 有时可以化入约束条件，反之亦然

$$\min x + y^2 + xy$$

$$\text{s.t. } x^3 + y \geq 0,$$

$$\text{var. } x \geq 0, y \leq 1.$$



$$\min x + y^2 + xy$$

$$\text{var. } (\mathbf{x}, y) \in \mathcal{D}_0.$$

$$\mathcal{D}_0 \triangleq \{(x, y) : x \geq 0, y \leq 1, x^3 + y \geq 0\}$$

最优化问题一般形式

目标函数

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

约束条件

$$\text{s.t. } g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束}$$

决策变量

$$h_i(\mathbf{x}) = 0, j = 1, \dots, p, \text{ 等式约束}$$

$$\text{var. } \mathbf{x} \in \mathcal{D}. \text{ 可行域}$$

注意2：最大化问题和带有 \geq 约束的问题都可以整理成一般形式

$$\max \log x + \frac{y}{z}$$

$$\text{s.t. } y^4 + z \geq x^2,$$



$$xy = x + z \sin y,$$

$$\text{var. } x, y, z.$$

最优化问题一般形式

目标函数

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

约束条件

$$\text{s.t. } g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束}$$

决策变量

$$h_i(\mathbf{x}) = 0, j = 1, \dots, p, \text{ 等式约束}$$

$$\text{var. } \mathbf{x} \in \mathcal{D}. \text{ 可行域}$$

注意2：最大化问题和带有 \geq 约束的问题都可以整理成一般形式

$$\max \log x + \frac{y}{z}$$

$$\text{s.t. } y^4 + z \geq x^2,$$

$$xy = x + z \sin y,$$

$$\text{var. } x, y, z.$$



$$f(\mathbf{x}) \triangleq -\log x_1 - \frac{x_2}{x_3}$$

$$g(\mathbf{x}) \triangleq x_1^2 - x_2^4 - x_3$$

$$h(\mathbf{x}) \triangleq x_1 x_2 - x_1 - x_3 \sin x_2$$

最优化问题一般形式

目标函数

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

约束条件

$$\text{s.t. } g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束}$$

决策变量

$$h_i(\mathbf{x}) = 0, j = 1, \dots, p, \text{ 等式约束}$$

$$\text{var. } \mathbf{x} \in \mathcal{D}, \text{ 可行域}$$



为什么要转化成一般形式？

方便判断最优化问题的类别（凸优化、线性规划、二次规划等）

方便选择求解方法、实施算法

最优化问题一般形式

目标函数

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

约束条件

$$\text{s.t. } g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束}$$

决策变量

$$h_i(\mathbf{x}) = 0, j = 1, \dots, p, \text{ 等式约束}$$

$$\text{var. } \mathbf{x} \in \mathcal{D}, \text{ 可行域}$$

常用概念

可行解 (**feasible solution**) : 满足约束条件和可行域的解

最优解 (**optimal solution**) : 令目标函数达到最小值的可行解

全局最优解 (**global optimal solution**) 、局部最优解 (**local optimal solution**)

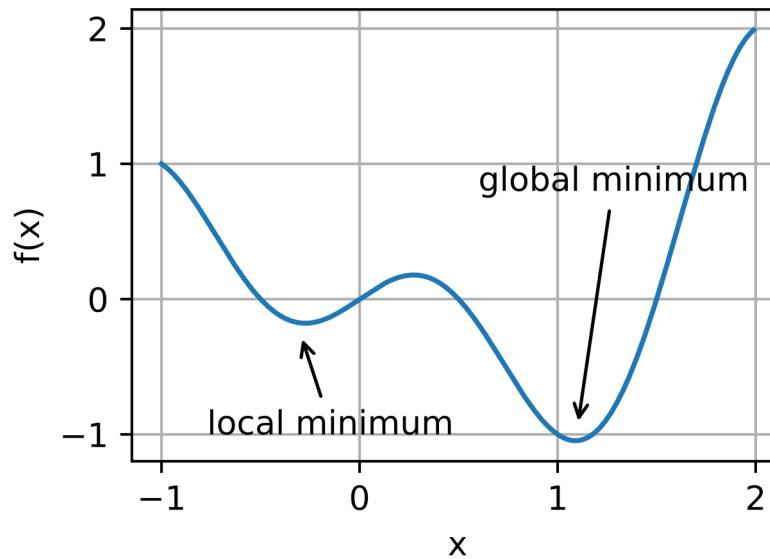
全局最优解 \mathbf{x}^* : 对所有可行解 \mathbf{x} , 有 $f(\mathbf{x}^*) \leq f(\mathbf{x})$

局部最优解 \mathbf{x}^* : 存在 $\delta > 0$, 使对所有满足 $\|\mathbf{x} - \mathbf{x}^*\| < \delta$ 的可行解 \mathbf{x} , 有 $f(\mathbf{x}^*) \leq f(\mathbf{x})$

最优化问题一般形式

全局最优解 x^* : 对所有可行解 x , 有 $f(x^*) \leq f(x)$

局部最优解 x^* : 存在 $\delta > 0$, 使对所有满足 $\|x - x^*\| < \delta$ 的可行解 x , 有 $f(x^*) \leq f(x)$



最优化问题一般形式

目标函数

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

约束条件

$$\text{s.t. } g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \text{ 不等式约束}$$

决策变量

$$h_i(\mathbf{x}) = 0, j = 1, \dots, p, \text{ 等式约束}$$

$$\text{var. } \mathbf{x} \in \mathcal{D}, \text{ 可行域}$$

常用概念

可行解 (**feasible solution**) : 满足约束条件和可行域的解

最优解 (**optimal solution**) : 令目标函数达到最小值的可行解

全局最优解 (**global optimal solution**) 、局部最优解 (**local optimal solution**)

一般情况下, “最优解”指的是“全局最优解”

注意: 最优解一定是可行解, 可行解不一定是最优解

无约束凸优化问题

Part1 如何从复杂度的角度衡量算法好坏？

Part2 针对一些有特定形式的最优化问题，如何找到最优解？

无约束凸优化问题、梯度下降法、牛顿法；

线性规划问题、单纯形法、内点法；

有约束凸优化问题、KKT条件

Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？

Part4 针对较复杂的多目标最优化问题，如何找到较好解？

无约束凸优化问题

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & g_i(\boldsymbol{x}) \leq 0, i = 1, \dots, m, \\ & h_i(\boldsymbol{x}) = 0, j = 1, \dots, p, \\ \text{var.} \quad & \boldsymbol{x} \in \mathcal{D}. \end{aligned}$$



$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{var.} \quad & \boldsymbol{x} \in \mathbb{R}^n. \end{aligned}$$

其中, $f(\boldsymbol{x})$ 是二阶连续可微的凸函数



无约束凸优化问题

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & g_i(\boldsymbol{x}) \leq 0, i = 1, \dots, m, \\ & h_i(\boldsymbol{x}) = 0, j = 1, \dots, p, \\ \text{var.} \quad & \boldsymbol{x} \in \mathcal{D}. \end{aligned}$$



$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{var.} \quad & \boldsymbol{x} \in \mathbb{R}^n. \end{aligned}$$

其中, $f(\boldsymbol{x})$ 是二阶连续可微的凸函数

什么是凸函数?

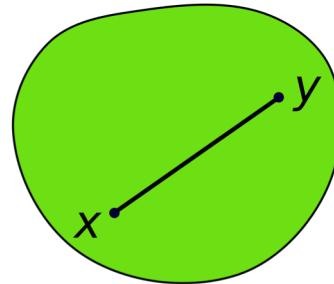
如何检验一个函数是否是凸函数?

例: $f(\boldsymbol{x}) = \boldsymbol{x}^2$

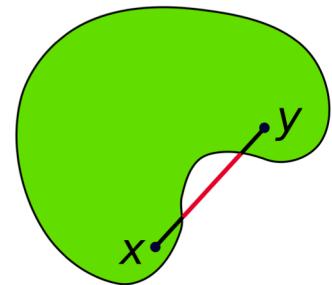
凸集与凸函数

凸集 (convex set)

集合中任取两点，两点间线段上的点也属于集合



凸集

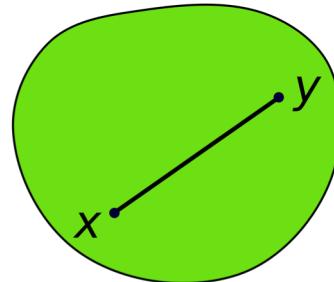


非凸集

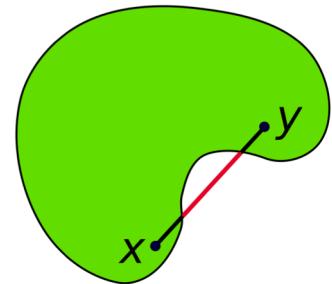
凸集与凸函数

凸集 (convex set)

集合中任取两点，两点间线段上的点也属于集合



凸集



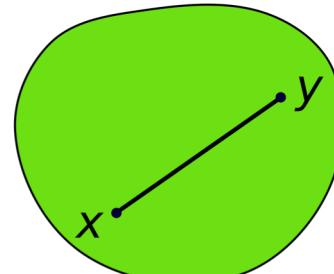
非凸集

定义：若对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$ ，有 $\lambda x + (1 - \lambda)y \in \mathcal{C}$ ，则集合 \mathcal{C} 为凸集。

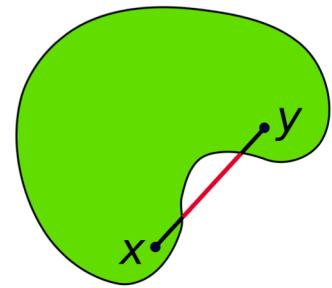
凸集与凸函数

凸集 (convex set)

集合中任取两点，两点间线段上的点也属于集合



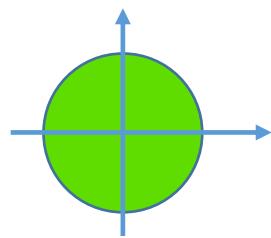
凸集



非凸集

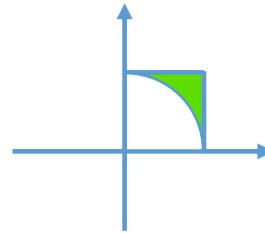
定义：若对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$ ，有 $\lambda x + (1 - \lambda)y \in \mathcal{C}$ ，则集合 \mathcal{C} 为凸集。

例：集合 $\{(x_1, x_2) | x_1^2 + x_2^2 \leq 1\}$ 是凸集



(先证 $x_1y_1 + x_2y_2 \leq 1$)

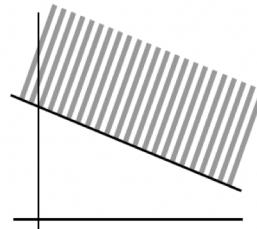
例：集合 $\{(x_1, x_2) | x_1^2 + x_2^2 \geq 1, 0 \leq x_1, x_2 \leq 1\}$ 是非凸集



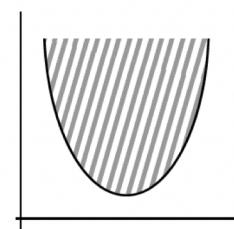
凸集与凸函数

凸函数

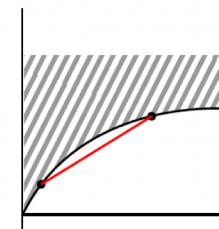
函数及其以上的区域是凸集



$$f(x) = ax + b$$



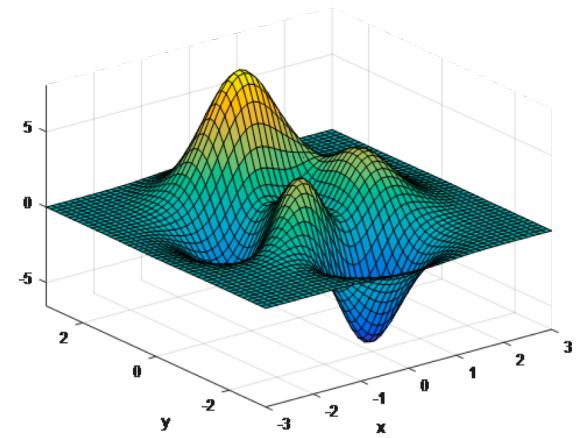
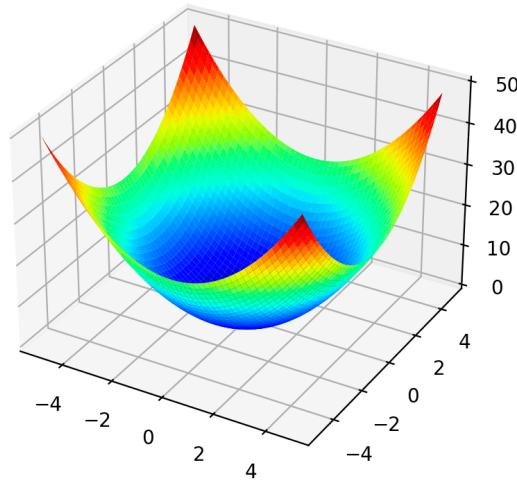
$$f(x) = \text{quadratic}$$



$$f(x) = \sqrt{x}$$

凸函数

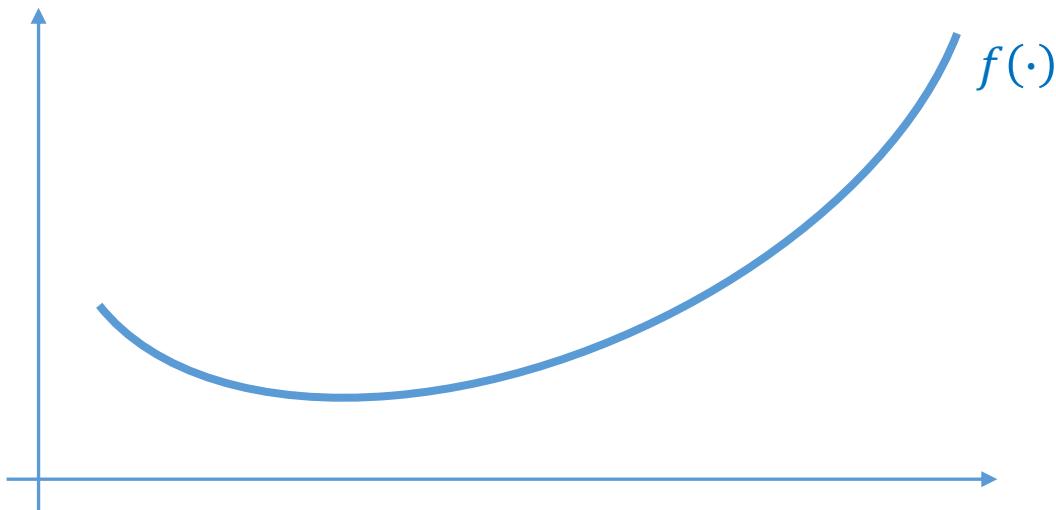
非凸函数



凸集与凸函数

凸函数定义：

- (1) 定义域 \mathcal{C} 是凸集；
- (2) 对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$, $f(\cdot)$ 满足 $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$ 。



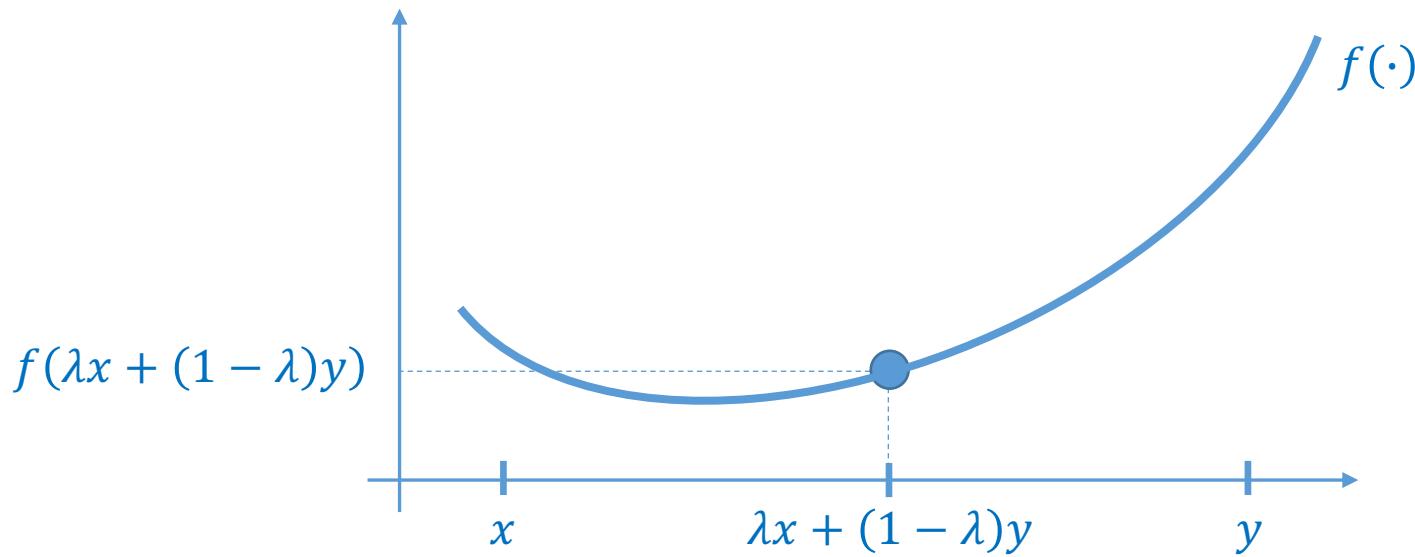
不等式在自变量是一维时的示意图



凸集与凸函数

凸函数定义：

- (1) 定义域 \mathcal{C} 是凸集；
- (2) 对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$, $f(\cdot)$ 满足 $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$ 。

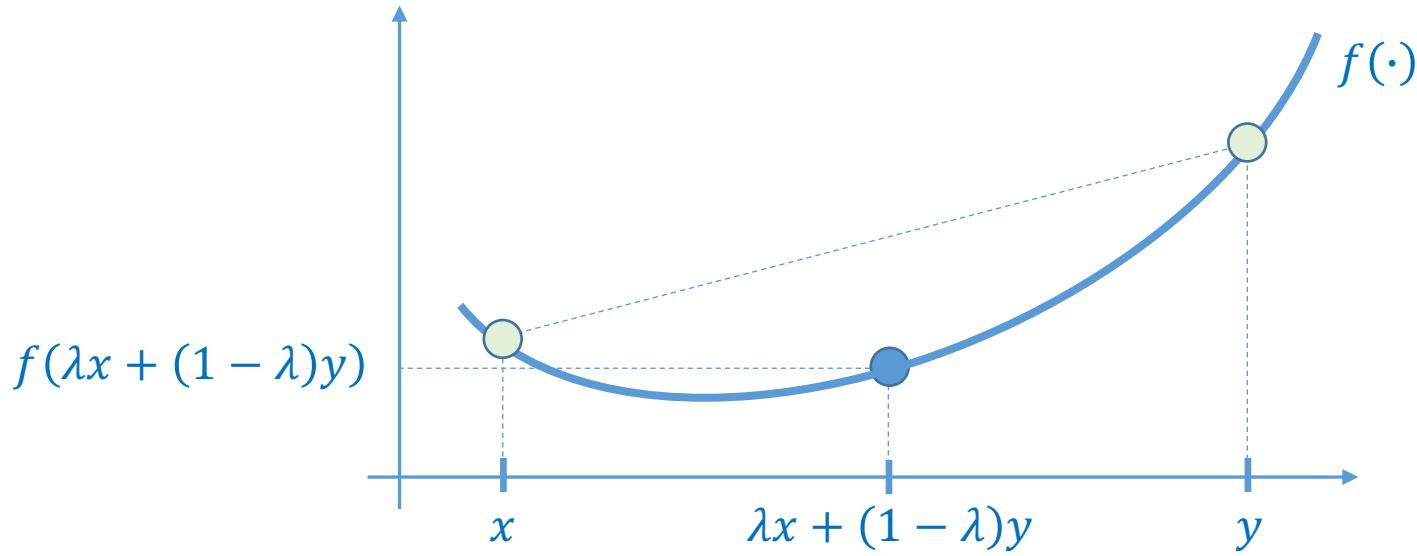


不等式在自变量是一维时的示意图

凸集与凸函数

凸函数定义：

- (1) 定义域 \mathcal{C} 是凸集；
- (2) 对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$, $f(\cdot)$ 满足 $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$ 。

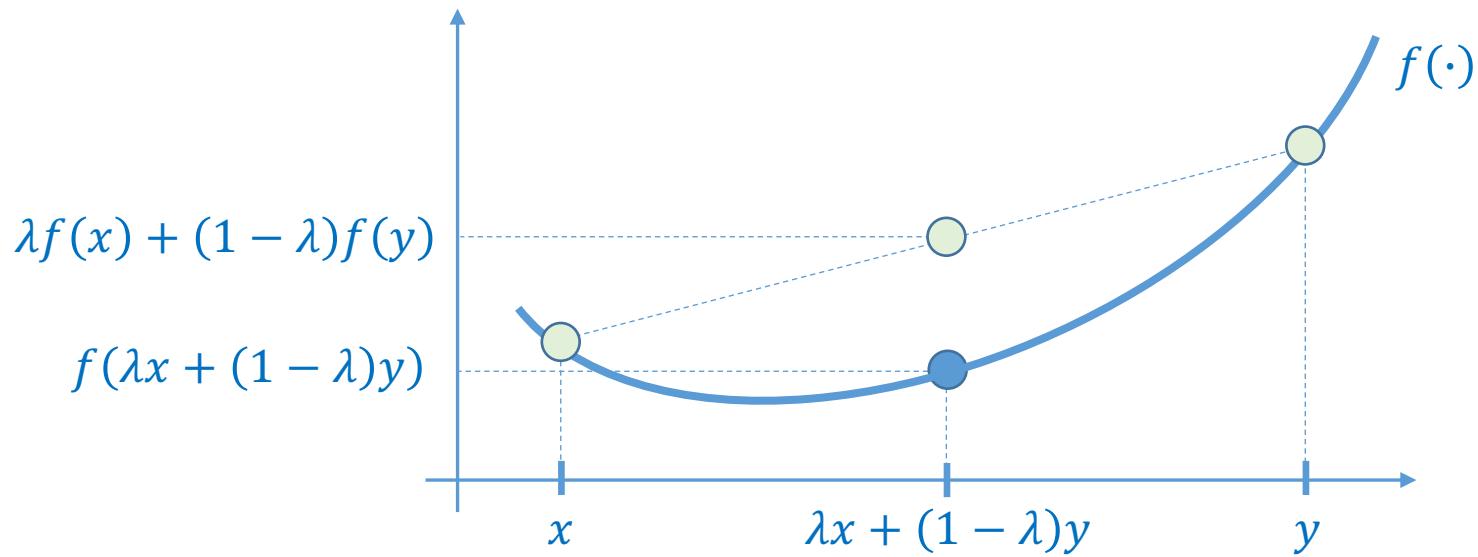


不等式在自变量是一维时的示意图

凸集与凸函数

凸函数定义：

- (1) 定义域 \mathcal{C} 是凸集；
- (2) 对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$, $f(\cdot)$ 满足 $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$ 。



不等式在自变量是一维时的示意图

凸集与凸函数

凸函数定义：

- (1) 定义域 \mathcal{C} 是凸集；
- (2) 对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$, $f(\cdot)$ 满足 $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$ 。

严格凸函数定义：

- (1) 定义域 \mathcal{C} 是凸集；
- (2) 对任意 $x, y \in \mathcal{C}$ 、 $x \neq y$ 及 $\lambda \in (0,1)$, 有 $f(\lambda x + (1 - \lambda)y) < \lambda f(x) + (1 - \lambda)f(y)$ 。

凸函数例子：

$$f(x) = x^2$$

如何快速检验一个函数是否是凸函数？

$$f(x) = e^x$$

$$f(x) = -\log x, x > 0$$

$$f(x) = x \log x, x > 0$$

凸集与凸函数

定理（凸函数的一阶条件）

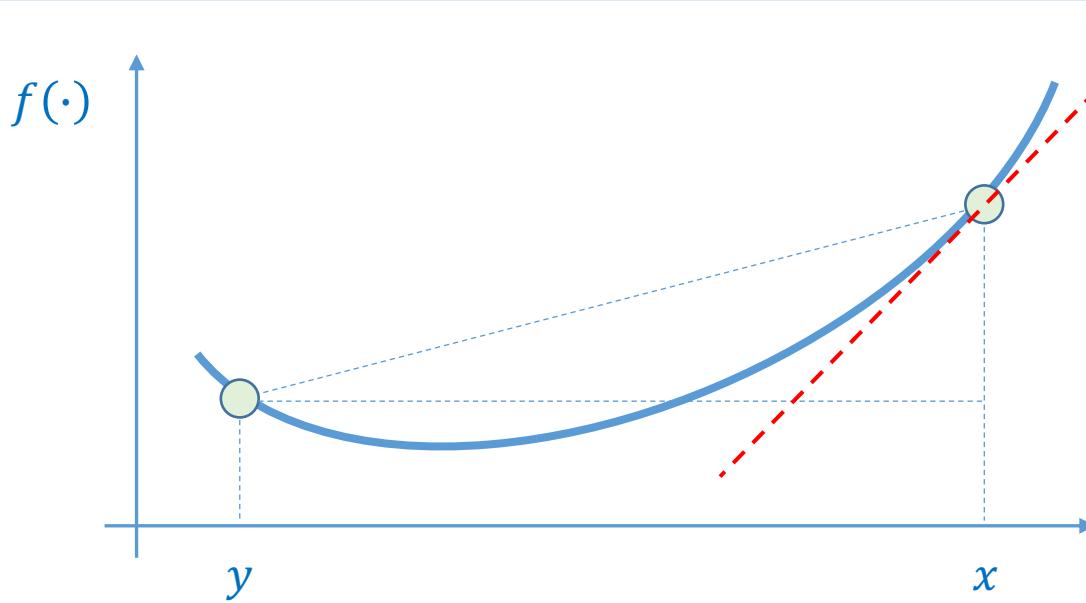
若 $f(x)$ 是定义在凸集 \mathcal{C} 上的一阶可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x, y \in \mathcal{C}$ ，有 $f(y) \geq f(x) + \nabla f(x)^T(y - x)$ 。



凸集与凸函数

定理（凸函数的一阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的一阶可微函数，那么 $f(x)$ 是凸函数的**充要条件**是：
对任意 $x, y \in \mathcal{C}$ ，有 $f(y) \geq f(x) + \nabla f(x)^T(y - x)$ 。



$$x, y \in \mathcal{R} \text{ 且 } x > y \text{ 时不等式可化简为 } \frac{f(x) - f(y)}{x - y} \leq f'(x)$$

凸集与凸函数

常用定理

定理（凸函数的二阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x \in \mathcal{C}$ ， $f(x)$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(x) \geq 0$ 。



凸集与凸函数

常用定理

定理（凸函数的二阶条件）

若 $f(\mathbf{x})$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(\mathbf{x})$ 是凸函数的充要条件是：对任意 $\mathbf{x} \in \mathcal{C}$ ， $f(\mathbf{x})$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(\mathbf{x}) \succeq \mathbf{0}$ 。

$$\text{海森矩阵 } \nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

例，对 $f(x_1, x_2)$

$$\nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f(x_1, x_2)}{\partial x_1^2} & \frac{\partial^2 f(x_1, x_2)}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f(x_1, x_2)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x_1, x_2)}{\partial x_2^2} \end{bmatrix}$$

凸集与凸函数

常用定理

定理 (凸函数的二阶条件)

若 $f(\mathbf{x})$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(\mathbf{x})$ 是凸函数的充要条件是：对任意 $\mathbf{x} \in \mathcal{C}$ ， $f(\mathbf{x})$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(\mathbf{x}) \succeq \mathbf{0}$ 。

$$\text{海森矩阵 } \nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

例，对 $f(x_1, x_2)$

$$\nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f(x_1, x_2)}{\partial x_1^2} & \frac{\partial^2 f(x_1, x_2)}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f(x_1, x_2)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x_1, x_2)}{\partial x_2^2} \end{bmatrix}$$

半正定矩阵 (高等代数知识)

例：单位矩阵 $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

对 $n \times n$ 实对称矩阵 \mathbf{A} ，若对任意向量 $\mathbf{u} \in \mathbb{R}^n$ 有 $\mathbf{u}^T \mathbf{A} \mathbf{u} \geq 0$ ， \mathbf{A} 为半正定矩阵。

凸集与凸函数

定理（凸函数的二阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x \in \mathcal{C}$ ， $f(x)$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(x) \geq 0$ 。

例，判断 $f(x_1, x_2) = 4x_1^2 - 4x_1x_2 + 2x_2^2$ 是否是凸函数



凸集与凸函数

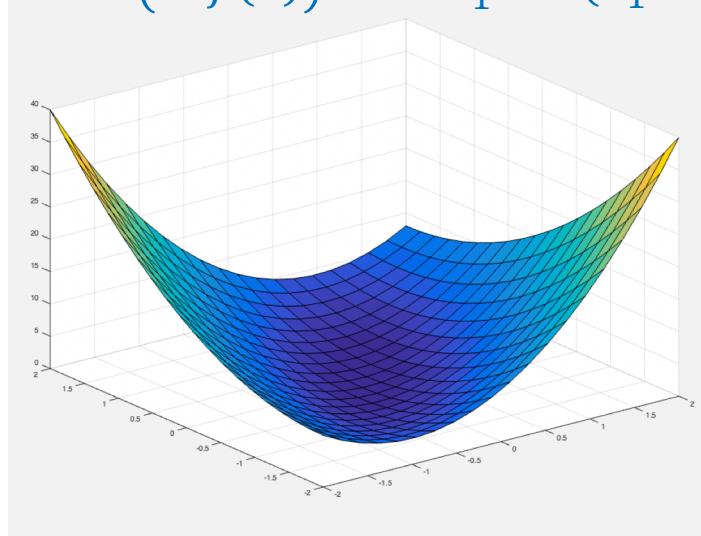
定理（凸函数的二阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x \in \mathcal{C}$ ， $f(x)$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(x) \geq 0$ 。

例，判断 $f(x_1, x_2) = 4x_1^2 - 4x_1x_2 + 2x_2^2$ 是否是凸函数

$$\nabla^2 f(x) = \begin{bmatrix} 8 & -4 \\ -4 & 4 \end{bmatrix}$$

$$\mathbf{u}^T (\nabla^2 f(x)) \mathbf{u} = 4u_1^2 + 4(u_1 - u_2)^2 \geq 0$$



凸集与凸函数

定理（凸函数的二阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x \in \mathcal{C}$ ， $f(x)$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(x) \geq 0$ 。

例，判断 $f(x_1, x_2) = 4x_1^2 - 4x_1x_2 + 2x_2^2$ 是否是凸函数

$$\nabla^2 f(x) = \begin{bmatrix} 8 & -4 \\ -4 & 4 \end{bmatrix} \quad \mathbf{u}^T (\nabla^2 f(x)) \mathbf{u} = 4u_1^2 + 4(u_1 - u_2)^2 \geq 0$$

当变量数大于2时如何判断是否是半正定矩阵？

高等代数知识：

对 $n \times n$ 实对称矩阵 A ，可以通过分析它的所有顺序主子式是否都非负，
判断 A 是否为半正定矩阵。

凸集与凸函数

定理（凸函数的二阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x \in \mathcal{C}$ ， $f(x)$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(x) \geq 0$ 。

例，判断 $f(x_1, x_2) = 4x_1^2 - 4x_1x_2 + 2x_2^2$ 是否是凸函数

$$\nabla^2 f(x) = \begin{bmatrix} 8 & -4 \\ -4 & 4 \end{bmatrix} \quad \mathbf{u}^T (\nabla^2 f(x)) \mathbf{u} = 4u_1^2 + 4(u_1 - u_2)^2 \geq 0$$

当变量数等于1时，条件 $\nabla^2 f(x) \geq 0$ 可化简为 $\frac{d^2 f(x)}{dx^2} \geq 0$

例： $f(x) = x^2$

凸集与凸函数

定理（凸函数的二阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x \in \mathcal{C}$ ， $f(x)$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(x) \geq 0$ 。

注意，该定理只适用于分析二阶连续可微函数

若 $f(x)$ 不满足二阶连续可微，也可以是凸函数



凸集与凸函数

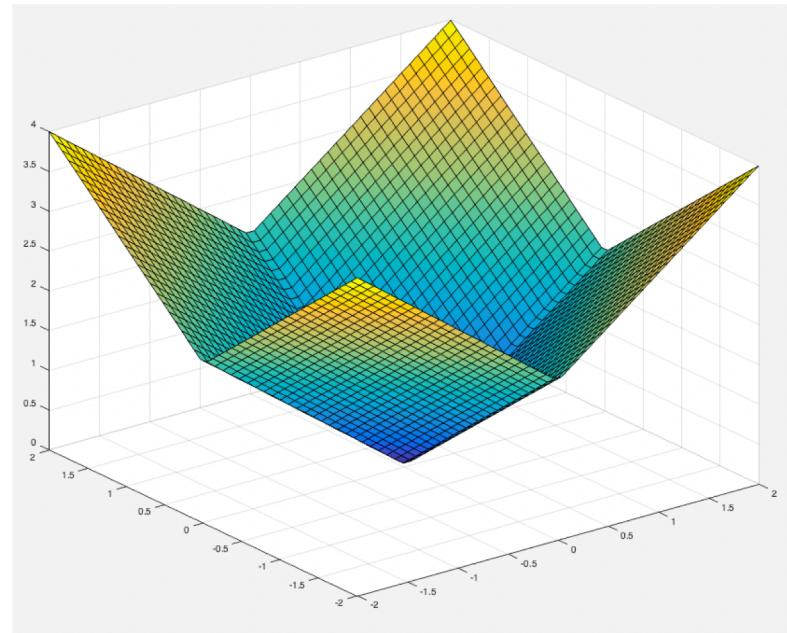
定理（凸函数的二阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的二阶连续可微函数，那么 $f(x)$ 是凸函数的充要条件是：
对任意 $x \in \mathcal{C}$ ， $f(x)$ 的海森矩阵是半正定矩阵，即 $\nabla^2 f(x) \geq 0$ 。

注意，该定理只适用于分析二阶连续可微函数

若 $f(x)$ 不满足二阶连续可微，也可以是凸函数

例： $f(x) = \|x\|_1 = |x_1| + |x_2|$

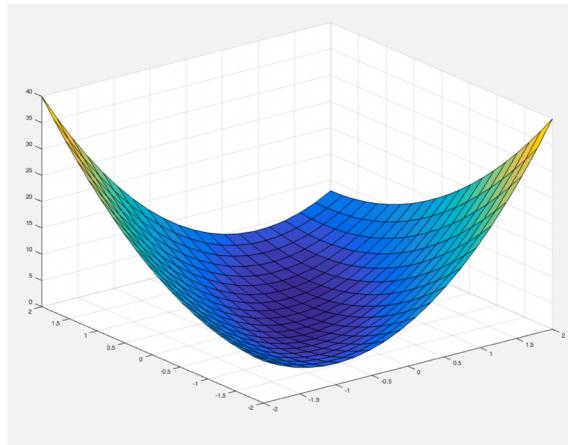
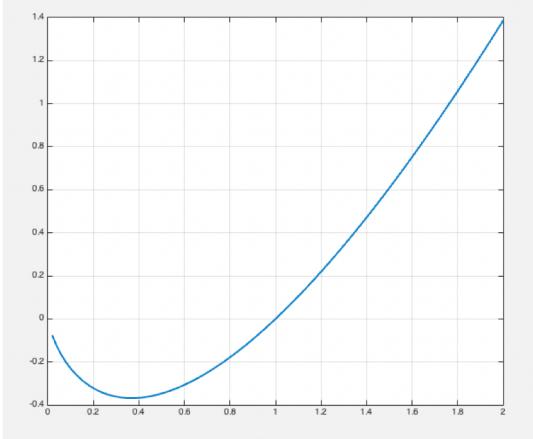


无约束凸优化问题

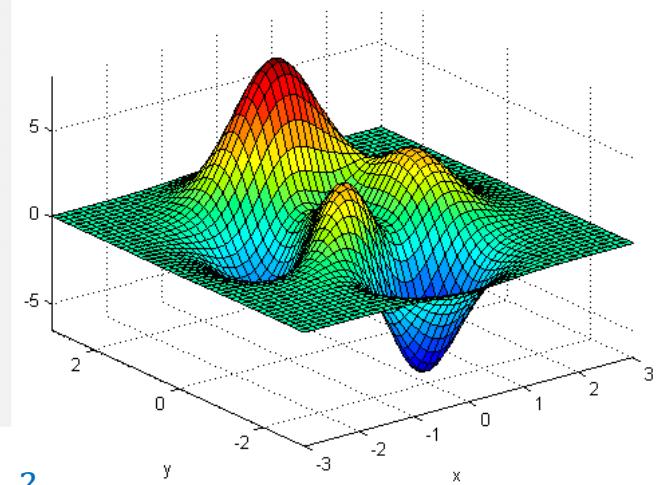
$$\begin{aligned} & \min f(x) \\ & \text{var. } x \in \mathbb{R}^n. \end{aligned}$$

其中, $f(x)$ 是二阶连续可微的凸函数

二阶连续可微凸函数
有什么性质可以简化问题求解?



$$f(x) = x \log x, x > 0 \quad f(x_1, x_2) = 4x_1^2 - 4x_1x_2 + 2x_2^2$$



非凸函数

无约束凸优化问题

定理

不要求二阶连续可微

若 $f(\mathbf{x})$ 是凸函数，那么任何使 $f(\mathbf{x})$ 达到局部极小值的局部最优解 $\mathbf{x}^{\text{local}}$ 也是使 $f(\mathbf{x})$ 达到全局最小值的全局最优解。

Proof. We prove the result by contradiction. Let \vec{x}_{loc} be a local minimum and \vec{x}_{glob} a global minimum such that $f(\vec{x}_{\text{glob}}) < f(\vec{x}_{\text{loc}})$. Since \vec{x}_{loc} is a local minimum, there exists $\gamma > 0$ for which $f(\vec{x}_{\text{loc}}) \leq f(\vec{x})$ for all $\vec{x} \in \mathbb{R}^n$ such that $\|\vec{x} - \vec{x}_{\text{loc}}\|_2 \leq \gamma$. If we choose $\theta \in (0, 1)$ ~~small~~ large enough, $\vec{x}_\theta := \theta \vec{x}_{\text{loc}} + (1 - \theta) \vec{x}_{\text{glob}}$ satisfies $\|\vec{x}_\theta - \vec{x}_{\text{loc}}\|_2 \leq \gamma$ and therefore

$$f(\vec{x}_{\text{loc}}) \leq f(\vec{x}_\theta) \tag{5}$$

$$\leq \theta f(\vec{x}_{\text{loc}}) + (1 - \theta) f(\vec{x}_{\text{glob}}) \quad \text{by convexity of } f \tag{6}$$

$$< f(\vec{x}_{\text{loc}}) \quad \text{because } f(\vec{x}_{\text{glob}}) < f(\vec{x}_{\text{loc}}). \tag{7}$$

无约束凸优化问题

定理

若 $f(x)$ 是一阶可微凸函数，那么 x^* 是最小化 $f(x)$ 的全局最优解的充要条件是：

$$\nabla f(x^*) = \mathbf{0} .$$



无约束凸优化问题

定理

若 $f(x)$ 是一阶可微凸函数，那么 x^* 是最小化 $f(x)$ 的全局最优解的充要条件是：

$$\nabla f(x^*) = \mathbf{0} .$$

当变量数等于1时，条件 $\nabla f(x^*) = \mathbf{0}$ 可化简为 $\frac{d^2 f(x)}{dx^2} \Big|_{x=x^*} = 0$

无约束凸优化问题

定理

若 $f(x)$ 是一阶可微凸函数，那么 x^* 是最小化 $f(x)$ 的全局最优解的充要条件是：

$$\nabla f(x^*) = \mathbf{0}.$$

可由凸函数的一阶条件证得：

定理（凸函数的一阶条件）

若 $f(x)$ 是定义在凸集 \mathcal{C} 上的一阶可微函数，那么 $f(x)$ 是凸函数的充要条件是：

对任意 $x, y \in \mathcal{C}$ ，有 $f(y) \geq f(x) + \nabla f(x)^T(y - x)$ 。

无约束凸优化问题

$$\begin{aligned} \min \quad & f(x) \\ \text{var. } \quad & x \in \mathcal{R}^n. \end{aligned}$$

其中, $f(x)$ 是二阶连续可微的凸函数

求解思路: 搜索满足 $\nabla f(x^*) = \mathbf{0}$ 的 x^*

例：线性回归

(理工睿府)

面积	室	厅	建成时间	楼高 (层)	售价
125.73	3	2	2014	20	540
85.25	2	1	2014	16	378
181.54	4	3	2014	20	620
...

假设 特征 与 售价 之间是线性函数关系，能否通过以上数据求出该线性函数？

例：线性回归



面积	室	厅	建成时间	楼高 (层)	售价
a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	b_1
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	b_2
a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	b_3
...

N条

希望找到一组系数 $w = (w_1, w_2, w_3, w_4, w_5)^T$ ，使得：

$$w_1 a_{11} + w_2 a_{12} + w_3 a_{13} + w_4 a_{14} + w_5 a_{15} \rightarrow b_1$$

$$w_1 a_{21} + w_2 a_{22} + w_3 a_{23} + w_4 a_{24} + w_5 a_{25} \rightarrow b_2$$

$$w_1 a_{31} + w_2 a_{32} + w_3 a_{33} + w_4 a_{34} + w_5 a_{35} \rightarrow b_3$$

...

在实际的线性回归中，还包含常数项 w_0

例：线性回归

希望找到一组系数 $\mathbf{w} = (w_1, w_2, w_3, w_4, w_5)^T$, 使得:

$$w_1 a_{11} + w_2 a_{12} + w_3 a_{13} + w_4 a_{14} + w_5 a_{15} \rightarrow b_1$$

$$w_1 a_{21} + w_2 a_{22} + w_3 a_{23} + w_4 a_{24} + w_5 a_{25} \rightarrow b_2$$

$$w_1 a_{31} + w_2 a_{32} + w_3 a_{33} + w_4 a_{34} + w_5 a_{35} \rightarrow b_3$$

...

$$\mathbf{A}\mathbf{w} \rightarrow \mathbf{b}$$

线性回归 (L2正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \\ \text{var.} \quad & \mathbf{w} \in \mathcal{R}^N. \end{aligned}$$

线性回归 (L1正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\| \\ \text{var.} \quad & \mathbf{w} \in \mathcal{R}^N. \end{aligned}$$

LASSO

为避免过拟合，在目标函数中加入正则项：

$$\|\mathbf{w}\|_2^2 = w_1^2 + w_2^2 + \cdots + w_N^2$$

$$\|\mathbf{w}\| = |w_1| + |w_2| + \cdots + |w_N|$$

(希望出现更多取值为零的系数)

例：线性回归

线性回归 (L2正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \\ \text{var.} \quad & \mathbf{w} \in \mathcal{R}^N. \end{aligned}$$

$$\|\mathbf{w}\|_2^2 = w_1^2 + w_2^2 + \cdots + w_N^2$$

线性回归 (L1正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\| \\ \text{var.} \quad & \mathbf{w} \in \mathcal{R}^N. \end{aligned}$$

$$\|\mathbf{w}\| = |w_1| + |w_2| + \cdots + |w_N|$$

目标函数是否是二阶连续可微凸函数？

LASSO

凸函数：对任意 $x, y \in \mathcal{C}$ 及 $\lambda \in [0,1]$ ，满足 $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$

例：线性回归

线性回归 (L2正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \\ \text{var.} \quad & \mathbf{w} \in \mathbb{R}^N. \end{aligned}$$

$$\|\mathbf{w}\|_2^2 = w_1^2 + w_2^2 + \cdots + w_N^2$$

目标函数是否是二阶连续可微凸函数？

是二阶连续可微凸函数

线性回归 (L1正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\| \\ \text{var.} \quad & \mathbf{w} \in \mathbb{R}^N. \end{aligned}$$

$$\|\mathbf{w}\| = |w_1| + |w_2| + \cdots + |w_N|$$

是凸函数

证明要点：

1. 若 $f_1(\mathbf{w})$ 和 $f_2(\mathbf{w})$ 都为凸函数，那么 $f_1(\mathbf{w}) + f_2(\mathbf{w})$ 也是凸函数（利用定义）
2. $\|\mathbf{w}\|_2^2$ 是凸函数（分析海森矩阵）
3. $\|\mathbf{w}\|$ 是凸函数（利用定义）
4. $\|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2$ 是凸函数

LASSO

半正定矩阵： $n \times n$ 实对称矩阵A，对任意向量 $\mathbf{u} \in \mathcal{R}^n$ 有 $\mathbf{u}^T \mathbf{A} \mathbf{u} \geq 0$

例：线性回归

线性回归 (L2正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \\ \text{var.} \quad & \mathbf{w} \in \mathcal{R}^N. \end{aligned}$$

$$\|\mathbf{w}\|_2^2 = w_1^2 + w_2^2 + \cdots + w_N^2$$

线性回归 (L1正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\| \\ \text{var.} \quad & \mathbf{w} \in \mathcal{R}^N. \end{aligned}$$

LASSO

$$\|\mathbf{w}\| = |w_1| + |w_2| + \cdots + |w_N|$$

目标函数是否是二阶连续可微凸函数？

$$\nabla(\|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2) = 2\mathbf{A}^T(\mathbf{A}\mathbf{w} - \mathbf{b})$$

$$\nabla^2(\|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2) = 2\mathbf{A}^T\mathbf{A}$$

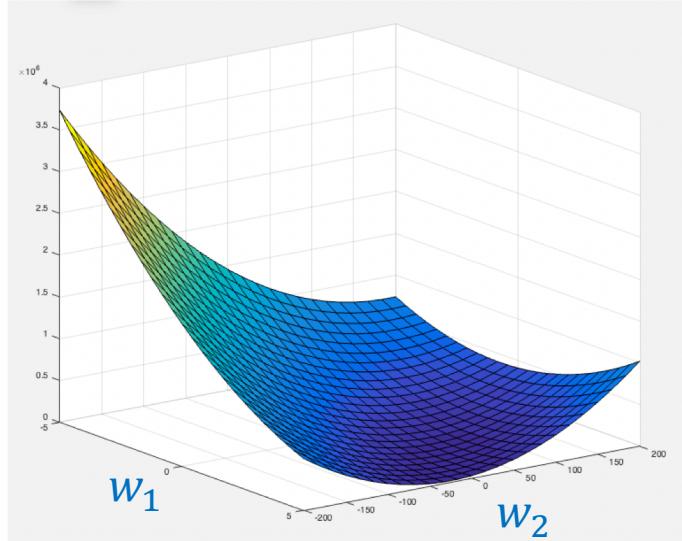
2 $\mathbf{A}^T\mathbf{A}$ 是半正定矩阵，因为它是实对称矩阵、且对任意向量 $\mathbf{u} \in \mathcal{R}^n$ 有 $\mathbf{u}^T(2\mathbf{A}^T\mathbf{A})\mathbf{u} = 2\|\mathbf{A}\mathbf{u}\|_2^2 \geq 0$

例：线性回归

线性回归 (L2正则化)

$$\begin{aligned} \min \quad & \frac{1}{N} \|\mathbf{A}\mathbf{w} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{w}\|_2^2 \\ \text{var. } \quad & \mathbf{w} \in \mathcal{R}^N. \end{aligned}$$

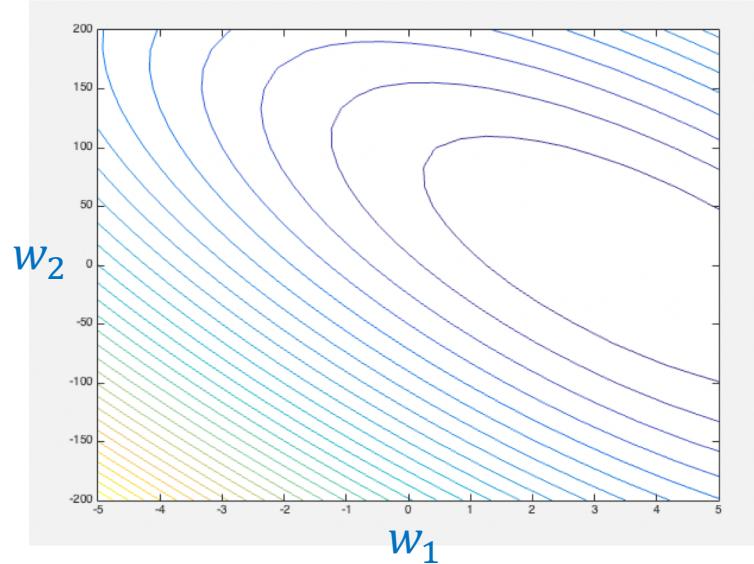
$$\|\mathbf{w}\|_2^2 = w_1^2 + w_2^2 + \cdots + w_N^2$$



$\lambda = 10$ 情况

考虑 $\mathbf{w} = (w_1, w_2)^T$ 的简化情况

面积	室	售价
125.73	3	540
85.25	2	378
181.54	4	620



等高线图

本讲小结



算法复杂度、P问题与NP问题



最优化问题的一般形式



凸集、凸函数、无约束凸优化问题

主要参考资料

Harvard CS50 <Introduction to Artificial Intelligence with Python> Slides

The OR Society 《运筹学起源故事》 视频

statistics.com “Safety in Numbers – Calculating Probabilities for Convoys” 网页

iq.opengenus.org “Time Complexity of Insertion Sort” 网页

David Evans CS3102 <Theory of Computation> Slides

Hackerdashery 《世界七大数学难题之一: P与NP复杂度问题》 视频

Wikipedia 关于Sorting algorithm、Gamma function等的词条

Carlos Fernandez-Granda <DS-GA1013/MATH-GA2821 Optimization-based Data Analysis> Lecture Notes

谢谢！



附录：最优化方法与运筹学的关系



运筹学 (Operations Research)

- ▶ 运用最优化理论和方法等解决生产生活中的问题：
在资源有限、受到约束的情况下，如何运用筹划各类资源，从而最大化/最小化指标
- ▶ 相关研究所一般划署在经管学院
- ▶ INFORMS运筹学与管理科学研究协会（类似ACM、IEEE）
(Institute for Operations Research and the Management Sciences)



运筹学 (Operations Research)

- ➡ 交通领域：路径优化问题
- ➡ 制造业领域：生产流程优化
- ➡ 电力领域：电网布局、用电分配、定价
- ➡ 能源领域：输油管道铺设
- ➡ 金融领域：资产配置、风险控制、保险分级定价
- ➡ 运输领域：火车、飞机航班调度
- ➡ 管理领域：酒店动态定价
- ➡



运筹学 (Operations Research)

► 运筹学的起源



“运筹学起源故事”

https://www.bilibili.com/video/BV1Y4411p7sq?from=search&seid=4798683364496178189&spm_id_from=333.337.0.0

运筹学 (Operations Research)

- ▶ 运筹学起源于二战时期
- ▶ 英国 海岸司令部运筹学处 (Coastal Command's Operational Research Section)

研究问题之一：如何躲避德军U型潜艇的袭击？



运筹学（Operations Research）

- ▶ 德军U型潜艇航速快，频繁袭击大西洋上英美等同盟国的运输船和商船
仅在1942年11月就击沉了盟军118艘船只



灰猎犬号的剧情简介 ······

影片讲述二战初期，由37支盟军船只组成的护航舰队在欧内斯特·克劳斯（汤姆·汉克斯饰）舰长率领的一艘美国驱逐舰指挥下，穿越险恶的北大西洋，同时还要与德国U型潜艇狼群的周旋。

运筹学 (Operations Research)

► 运筹学问题：应该让商船分散航行还是集中航行（减少因U型潜艇攻击的损失）？

例如有50支商船

分散航行：单支船单次航行被发现可能性低

集中航行：目标更大、单次航行被发现可能性高，但方便派军舰护送

需要哪些信息用于判断？

运筹学 (Operations Research)

► 运筹学问题：应该让商船分散航行还是集中航行（减少因U型潜艇攻击的损失）？

50支船集中航行被发现概率： $2/3$

单支船航行被发现概率： $1/4$

50支船集中航行 如果被发现 平均损失：5支船

单支船航行 如果被发现 平均损失：1 支船

运筹学 (Operations Research)

► 运筹学问题：应该让商船分散航行还是集中航行（减少因U型潜艇攻击的损失）？

50支船集中航行被发现概率： $2/3$

单支船航行被发现概率： $1/4$

50支船集中航行 如果被发现 平均损失：5支船

单支船航行 如果被发现 平均损失：1 支船

集中航行期望损失 3.33支

分散航行期望损失 12.5支

运筹学 (Operations Research)

► 运筹学问题：应该让商船分散航行还是集中航行（减少因U型潜艇攻击的损失）？

集中航行期望损失 3.33 支

分散航行期望损失 12.5 支

实际需要考虑的因素更多：集中航行需要花额外时间集结船队/开始航行后船队的航速受到最慢船只的限制/…

Nonetheless, the net effect of convoying was strongly positive. In the mid-Atlantic region, convoys lost only 4% of their tonnage, compared to 20% by independent ships.

运筹学（Operations Research）

- ▶ 二战之后，许多国家大学成立了运筹学系、管理科学系、工业工程系、系统科学系，继续运筹学的研究，将运筹学应用到生产生活的各个方面
- ▶ 现代计算机的发展促进了运筹学的发展，运筹学中复杂问题的求解通常需要大量的计算，计算机及相关软件的普及极大方便了求解过程
- ▶ 《最优化理论》是运筹学相关专业核心课程之一



附录：数学建模示例



数学建模

- ▶ 数学建模竞赛
- ▶ 针对实际问题建立数学模型，对模型求解，根据结果解决实际问题



数学建模

最优化类的赛题

2021年全国大学生数学建模竞赛题目之一

C 题 生产企业原材料的订购与运输

某建筑和装饰板材的生产企业所用原材料主要是木质纤维和其他植物素纤维材料，总体可分为 A, B, C 三种类型。该企业每年按 48 周安排生产，需要提前制定 24 周的原材料订购和转运计划，即根据产能要求确定需要订购的原材料供应商（称为“供应商”）和相应每周的原材料订购数量（称为“订货量”），确定第三方物流公司（称为“转运商”）并委托其将供应商每周的原材料供货数量（称为“供货量”）转运到企业仓库。

该企业每周的产能为 2.82 万立方米，每立方米产品需消耗 A 类原材料 0.6 立方米，或 B 类原材料 0.66 立方米，或 C 类原材料 0.72 立方米。由于原材料的特殊性，供应商不能保证严格按订货量供货，实际供货量可能多于或少于订货量。为了保证正常生产的需要，该企业要尽可能保持不少于满足两周生产需求的原材料库存量，为此该企业对供应商实际提供的原材料总是全部收购。

在实际转运过程中，原材料会有一定的损耗（损耗量占供货量的百分比称为“损耗率”），转运商实际运送到企业仓库的原材料数量称为“接收量”。每家转运商的运输能力为 6000 立方米/周。通常情况下，一家供应商每周供应的原材料尽量由一家转运商运输。

原材料的采购成本直接影响到企业的生产效益，实际中 A 类和 B 类原材料的采购单价分别比 C 类原材料高 20% 和 10%。三类原材料运输和储存的单位费用相同。

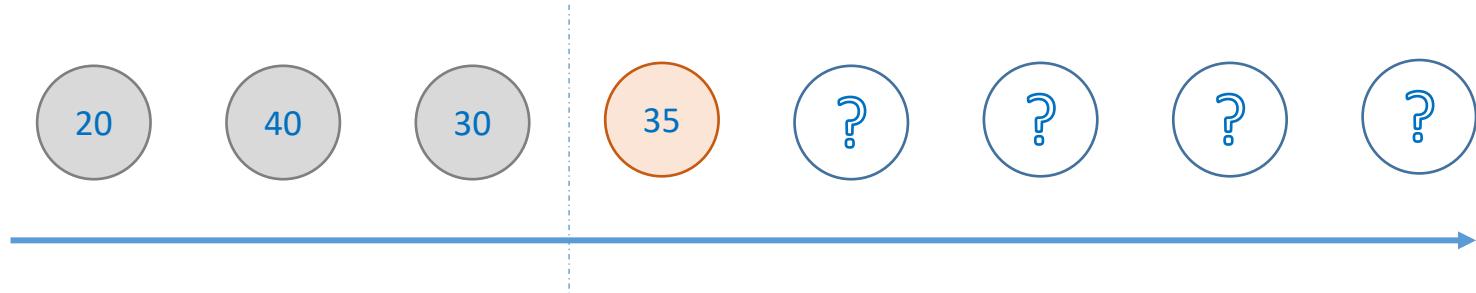
附件 1 给出了该企业近 5 年 402 家原材料供应商的订货量和供货量数据。附件 2 给出了 8 家转运商的运输损耗率数据。请你们团队结合实际情况，对相关数据进行深入分析，研究下列问题：

1. 根据附件 1，对 402 家供应商的供货特征进行量化分析，建立反映保障企业生产重要性的数学模型，在此基础上确定 50 家最重要的供应商，并在论文中列表给出结果。
2. 参考问题 1，该企业应至少选择多少家供应商供应原材料才可能满足生产的需求？针对这些供应商，为该企业制定未来 24 周每周最经济的原材料订购方案，并据此制定损耗最少的转运方案。试对订购方案和转运方案的实施效果进行分析。

考虑不同供应商的供货量、供货种类、价格等因素的不同

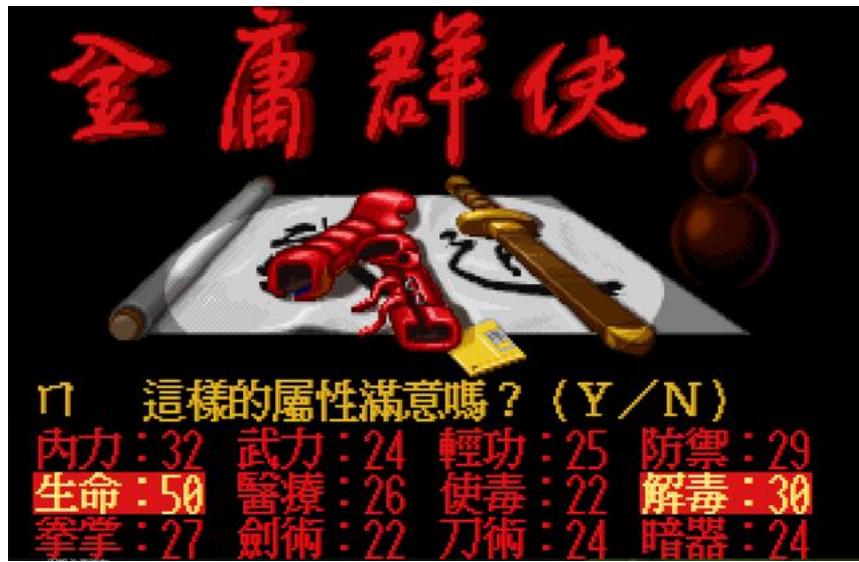
建模示例

小明将依次面见 n 个 ($n > 1$) 相亲对象。在每次相亲后要决定是否牵手，如果牵手将不能继续之后的相亲；如果不牵手则继续相亲但不能回头。问如何拟定牵手策略将最大化小明和最合适对象在一起的概率？【经典的相亲问题/秘书问题】



建模示例

小明将依次面见n个 ($n > 1$) 相亲对象。在每次相亲后要决定是否牵手，如果牵手将不能继续之后的相亲；如果不牵手则继续相亲但不能回头。问如何拟定牵手策略将最大化小明和最合适的对象在一起的概率？【经典的相亲问题/秘书问题】



游戏里选角色初始能力值

Y: 接受当前初始值开始游戏

N: 随机生成一组新的能力值再选择

建模示例

小明将依次面见 n 个 ($n > 1$) 相亲对象。在每次相亲后要决定是否牵手，如果牵手将不能继续之后的相亲；如果不牵手则继续相亲但不能回头。问如何拟定牵手策略将最大化小明和最合适对象在一起的概率？【经典的相亲问题/秘书问题】

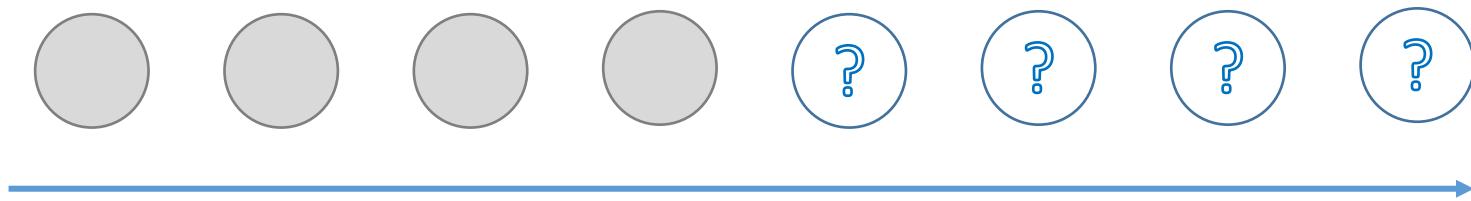
模型假设

- (1) 小明和所有对象之间的匹配程度可以由高到低排序
(即不会同时出现A比B好、B比C好、C比A好的情况)
- (2) 匹配程度均匀分布

示例一建立模型

已知最优选择策略满足如下形式（略过证明）：

小明先与前 $r-1$ 人相亲，无论多有好感都不牵手。从第 r 人开始，小明如果遇到比前 $r-1$ 人都更匹配的人就选择牵手。



求问最优的 r 取值是多少？

示例一模型求解

小明先与前 $r-1$ 人相亲，无论多有好感都不牵手。从第 r 人开始，小明如果遇到比前 $r-1$ 人都更匹配的人就选择牵手。问最优 r 值？

用 $P(r)$ 表示在给定 r 值下，选择的人是所有 n 个人中最佳人选的概率

$$\begin{aligned} P(r) &= \sum_{k=r}^n \Pr(\text{第 } k \text{ 人是最佳人选且被选择了}) \\ &= \sum_{k=r}^n \Pr(\text{第 } k \text{ 人是最佳人选}) \Pr(\text{第 } k \text{ 人被选择了} | \text{第 } k \text{ 人是最佳人选}) \\ &= \sum_{k=r}^n \frac{1}{n} \Pr(\text{第 } k \text{ 人被选择了} | \text{第 } k \text{ 人是最佳人选}) \\ &= \sum_{k=r}^n \frac{1}{n} \Pr(\text{前 } k-1 \text{ 个人中最匹配的人出现在前 } r-1 \text{ 个人中}) \\ &= \sum_{k=r}^n \frac{1}{n} \frac{r-1}{k-1} = \frac{r-1}{n} \sum_{k=r}^n \frac{1}{k-1} \end{aligned}$$

示例一模型求解

小明先与前r-1人相亲，无论多有好感都不牵手。从第r人开始，小明如果遇到比前r-1人都更匹配的人就选择牵手。问最优r值？

用P(r)表示在给定r值下，选择的人是所有n个人中最佳人选的概率

$$P(r) = \frac{r-1}{n} \sum_{k=r}^n \frac{1}{k-1}$$

可以用数值方法求到给定n值下令P(r)值最大的r

可以发现，当n较大时，最优r取值为 $n/e+1 \approx 0.368n+1$

先和前0.368n个人相亲，然后从第0.368n+1个人开始，如果遇到比之前所有人都更匹配的人就牵手。