优化问题的描述及算法概述

主讲：喻孜

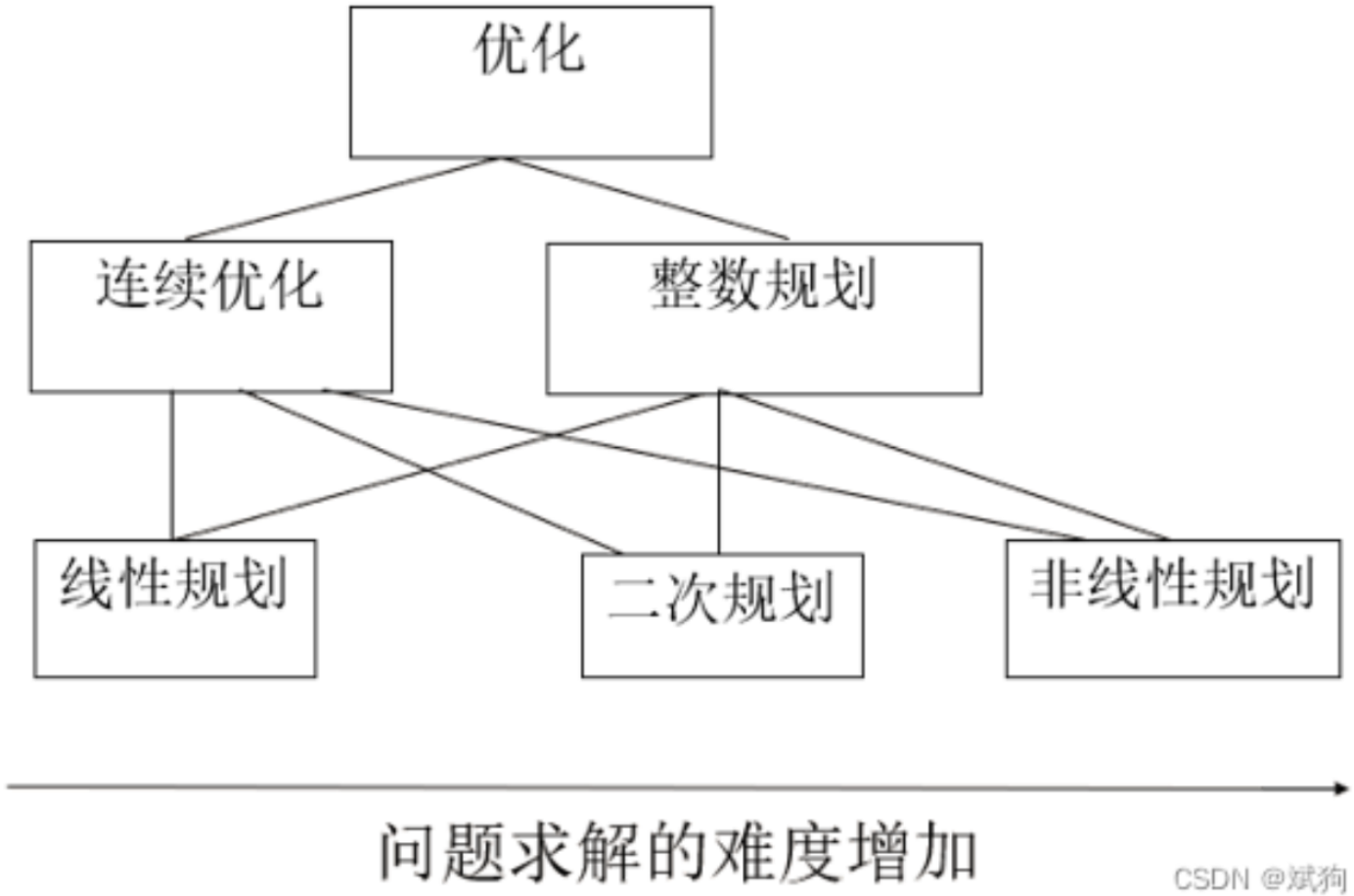
科学研究的最终目标：优化

- 寻找规律
- 确定规律
- 预测
- 优化

- 2000B 钢管订购和运输问题
- 2001B 公交车优化调度
- 2001C 基金使用的最优策略
- 2002B 彩票中的数学
- 2003B 露天矿生产的车辆安排问题
- 2004B 奥运会临时超市网点设计问题
- 2004D 公务员招聘工作中录用方案
- 2005B DVD在线租赁
- 2006B 出版社的资源配置问题
- 2007B 乘公交，看奥运
- 2008B 高等教育学费探讨
- 2009B 眼科病床的合理安排
- 2011B 交巡警服务平台设置与调度
- 2012B 太阳能小屋设计
- 2013B 碎纸片的拼接
- 2014B 创意折叠桌设计
- 2015B “互联网+”时代的出租车资源配置
- 2016B 小区开放对道路通行的影响
- 2017B “拍照赚钱”任务定价
- 2018B 智能RGV的动态调度策略
- 2019B “同心协力”策略研究
- 2020B 穿越沙漠
- 2021C 生产企业原材料的订购与运输

历年国赛优化问题：数学建模的问题中必然伴随着优化

优化模型的简单分类和求解难度



先来一个简单的例子

某工厂可生产甲、乙两种产品，需消耗煤、电、油三种资源，有关单耗数据如表，试拟定使总收入最大的生产计划。

资源＼产品	甲	乙	资源限制
煤(A)	9	4	360
电(B)	4	5	200
油(C)	3	10	300
单价	7	12	

解

设 x_1 为甲产品的产量，x_2 为乙产品的产量，

总收入可表示为：$z = 7x_1 + 12x_2$

A 资源的限制可表示为：$9x_1 + 4x_2 \le 360$

B 资源的限制可表示为：$4x_1 + 5x_2 \le 200$

C 资源的限制可表示为：$3x_1 + 10x_2 \le 300$

另外，产品的产量不能为负，可表示为：$x_1 \ge 0,\quad x_2 \ge 0$

最后的模型

$$\max\ z = 7x_1 + 12x_2$$

$$s.t.\begin{cases} 9x_1 + 4x_2 \le 360 \\ 4x_1 + 5x_2 \le 200 \\ 3x_1 + 10x_2 \le 300 \\ x_1 \ge 0,\quad x_2 \ge 0 \end{cases}$$

线性规划案例：

1.确定优化变量

2.确定约束函数

3.确定目标函数

4.求解

matlab求解

```
% 化成matlab求线性规划的标准形式
f = [-7 -12]'; % 目标函数的系数向量
A=[9 4;4 5;3 10]; % 不等式约束
b=[360 200 300]';
lb=[0 0]';
[x,val]=linprog(f,A,b,[],[],lb)
val=-val
```

（2）矩阵形式

主要用于在matlab中求解，用matlab求解时需要将涉及到的矩阵全部列出。策变量、约束条件比较少的时候完全可以使用。而当决策变量、约束条件成百上千时，写出其矩阵变得不太现实，这就需要用lingo求解

$$\max z = \mathbf{CX}$$

$$s.t.\begin{cases}\mathbf{AX} = \mathbf{b}\\ \mathbf{X} \geq \mathbf{0}\end{cases}$$

例如在上个例题中就需要列出以下矩阵

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \qquad \mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 360 \\ 200 \\ 300 \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} 9 & 4 & 1 & 0 & 0 \\ 4 & 5 & 0 & 1 & 0 \\ 3 & 10 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 7 & 12 & 0 & 0 & 0 \end{bmatrix}$$

两种写法

（3）和式形式

这种形式主要适用于lingo中，可以根据和式形式比较容易的编写出相应的程序

$$\max z = \sum_{j=1}^{n} c_j x_j$$

和式形式：

$$s.t.\begin{cases}\sum_{j=1}^{n} a_{ij} x_j = b_i & i = 1,2,\cdots,m \\ x_j \geq 0 & j = 1,2,\cdots,n\end{cases}$$

求解算法（matlab及很多软件已经内嵌函数无需自编代码）

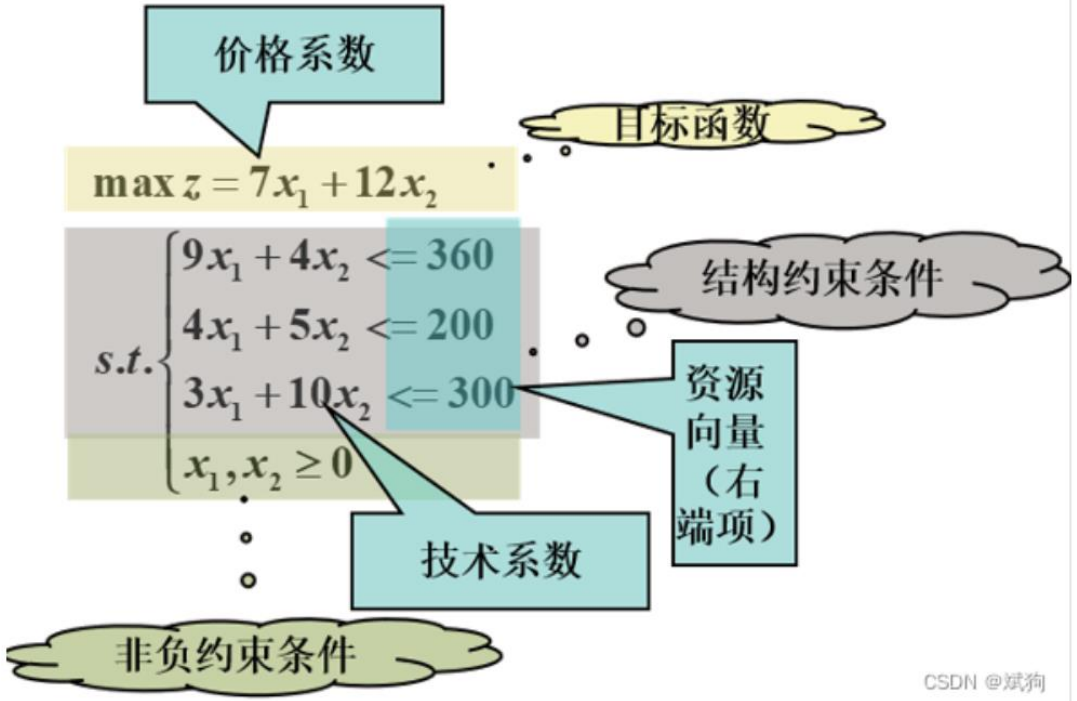
2.1.3 求解通用算法

- 单纯形算法

二维空间线性规划的最优解必然在凸多边形的顶点
三维空间线性规划的最优解必然在凸多边体的顶点

2.1.4 灵敏性分析

意义：现实问题中价格、技术、资源系数可能会变化（发生微小变动），生产方案也会受到影响
给定一个波动范围使得结果还是最优解不变



解决优化（规划）问题实际上分两个步骤

- 写出优化的数学方程式（决策变量、目标函数、约束条件）
- 寻找合适的算法

【例13.1-1】某厂生产 三种产品，每种产品生产需经过三道工序：选料、提纯和调配。根据现有的生产条件，可确定各工序有效工时、单位产品耗用工时及利润如表13.1-1所列。试问应如何安排各种产品的周产量，才能获得最大利润？

工 序	单位产品耗用工时（h/kg）			每周有效工时（h）
	A	*B*	*C*	
选料	1.1	1.2	1.4	4600
提纯	0.5	0.6	0.6	2100
调配	0.7	0.8	0.6	2500
利润（元/kg）	12	14	13	

该例数学模型如下：

$$\max \quad z = 12x_1 + 14x_2 + 13x_3$$

$$s.t. \begin{cases} 1.1x_1 + 1.2x_2 + 1.4x_3 \le 4600 \\ 0.5x_1 + 0.6x_2 + 0.6x_3 \le 2100 \\ 0.7x_1 + 0.8x_2 + 0.6x_3 \le 2500 \\ x_1 \ge 0,\ x_2 \ge 0,\ x_3 \ge 0 \end{cases}$$

线性规划的标准型

$$\min \quad z = f^T x$$

$$s.t \begin{cases} A \bullet x \le b \\ Aeq \bullet x = beq \\ lb \le x \le up \end{cases}$$

其中*f*为目标函数中决策变量的系数值向量，A为线性不等式约束的系数矩阵，b为线性不等式约束的右端常数向量，Aeq为线性等式约束的系数矩阵，beq为线性等式约束的右端常数向量，lb为决策变量 *x* 的下界值向量，up为决策变量x的上界值向量。

二、约束非线性优化

不失一般性，约束优化的一般形式：

$$\min f(x)$$

$$s.t\begin{cases} g_i(x) \le 0 & i = 1,2,\cdots,m \\ h_j(x) = 0 & j = 1,2,\cdots,l \end{cases}$$

其中，$f(x): \boldsymbol{R}^n \to \boldsymbol{R}$ 为非线性函数，$g_i(x)$ 为不等式约束，$h_j(x) = 0$ 为等式约束。

与无约束非线性最大化类似，对于约束非线性最大化可以通过转换，将其转化为标准的约束非线性优化的一般形式：

$$\max\ f(x)$$

$$s.t\begin{cases} g_i(x) \geq 0 \quad i = 1,2,\cdots,m \\ h_j(x) = 0 \quad j = 1,2,\cdots,l \end{cases}$$

$$\Rightarrow \min -\ f(x)$$

$$s.t\begin{cases} -g_i(x) \leq 0 \quad i = 1,2,\cdots,m \\ h_j(x) = 0 \quad j = 1,2,\cdots,l \end{cases}$$

三、 fmincon函数

fmincon是MATLAB最主要的求解约束最优化的函数，该函数要求的约束优化问题的标准形式为：

$$\min f(x)$$
$$s.t\begin{cases} c(x) \le 0 \\ ceq(x) = 0 \\ A \bullet x \le b \\ Aeq \cdot x = beq \\ lb \le x \le ub \end{cases}$$

其中， x, b, beq, lb, ub **为向量，** A **与** Aeq **为矩阵，** $f(x)$ **为目标函数，** $c(x), ceq(x)$ **为非线性约束，** $A \cdot x \le b, Aeq \cdot x = beq$ **为线性约束，** $lb \le x \le ub$ **为可行解的区间约束。**

fmincon调用格式主要如下：

```
x = fmincon(fun,x0,A,b)
x = fmincon(fun,x0,A,b,Aeq,beq)
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub)
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options)
[x,fval] = fmincon(...)
[x,fval,exitflag] = fmincon(...)
[x,fval,exitflag,output] = fmincon(...)
[x,fval,exitflag,output,lambda] = fmincon(...)
[x,fval,exitflag,output,lambda,grad] = fmincon(...)
[x,fval,exitflag,output,lambda,grad,hessian] = fmincon(...)
```

函数输入：

Fun:　目标函数名称　　X0:　初始迭代点

A:　线性不等约束系数矩阵　B：　线性不等式约束的常数向量

Aeq：　线性等约束系数矩阵　Beq:　线性等式约束的常数向量

lb ：可行区域下届　　Ub ：　可行区域上界

Nonlcon：非线性约束　　Options：优化参数设置

函数输出：

X：　最优点（或者结束迭代点）

Fval:　最有点（或者结束迭代点对应的函数值

Exitflag：迭代停止标识

Output ：算法输出（算法计算信息等）

Ambda：拉格朗日乘子

Grad ：一阶导数向量

Hessian：二阶导数矩阵

用法参考书中：【例14.3-2】和【例14.3-3】

一、大规模优化问题

MATLAB求解大规模无约束最优化问题(或者其他大规模问题)时候，一般都会通过optimset函数对默认算法设置进行改变，使其使用专用的适合大规模问题的解算设置。

【例14.4-1】 求解如下优化问题（含200个变量）：

$$\min f = \sum_{i=1}^{n} (x(i) - \frac{1}{i})^2, n = 200$$

```
n = 200;
x0 = 10*ones(1,n); %初始迭代点
PTIONS =
optimset('LargeScale','on','display','iter','TolFun',1e-8);
[x,fval,exitflag,output] = fminunc(@LargObjFun,x0,PTIONS)
```

```
function f=LargObjFun(x)
f=0;
n = 200;
for ii=1:n
    f=f+(x(ii)-1/ii)^2;
end
```

$$\min f = \sum_{i=1}^{n} (x(i) - \frac{1}{i})^2, n = 200$$

一、最大最小问题的数学模型

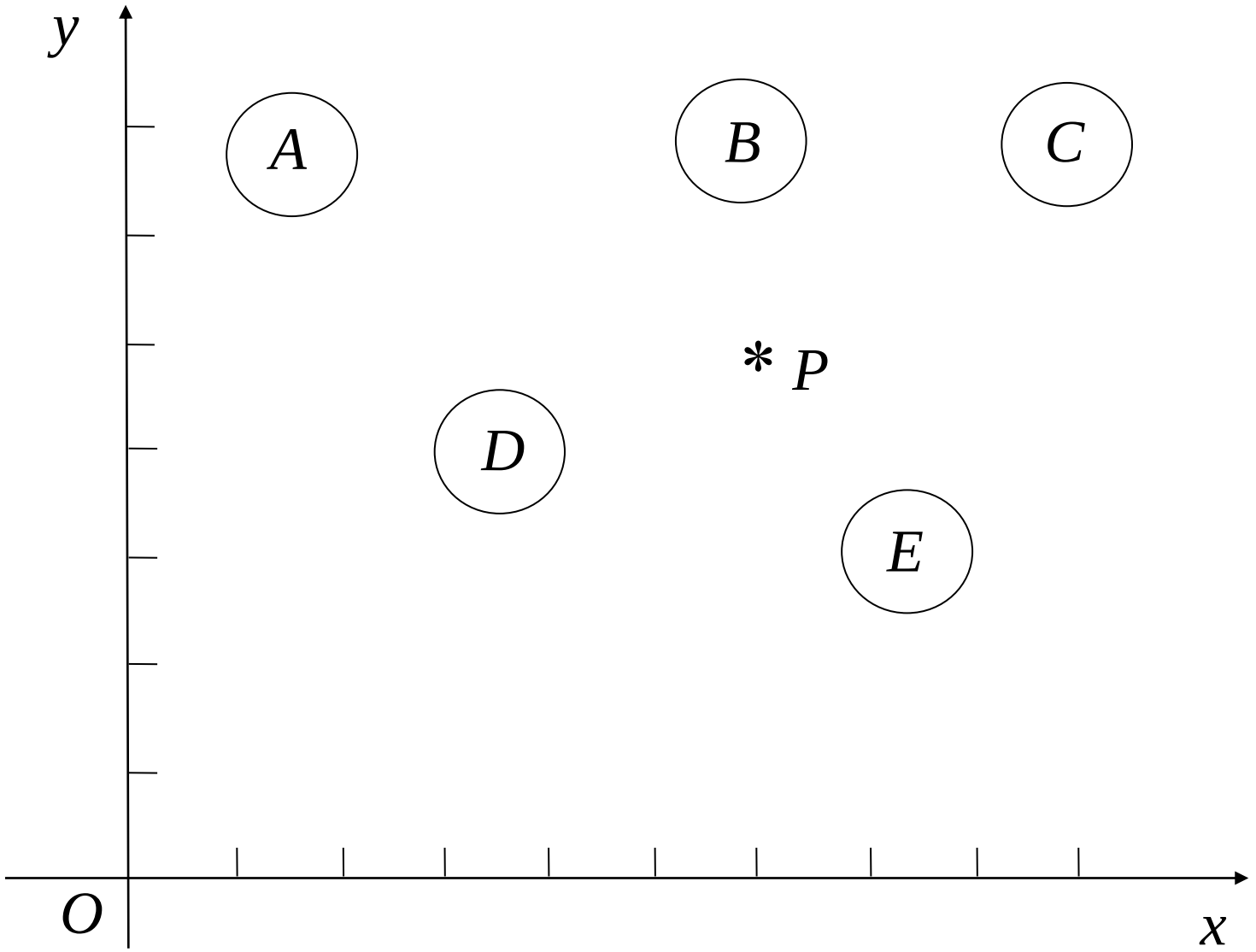
$$\min\{\max(f_1, f_2, \cdots, f_n)\}$$

$$s.t.\begin{cases} h_i(x) = 0, i = 1, 2, \cdots, m \\ g_j(x) \le 0, j = 1, 2, \cdots, n \end{cases}$$

二、垃圾场选址问题

【例15.1-1】已知A,B,C,D,E五个城市的位置分布如图15.1-1所示，坐标如表15.1-1所列。计划在A,B,C,D,E五个城市之间建造一个垃圾厂P，使得 五个城市将垃圾运往垃圾厂P的运输成本尽可能的相差不大。 由于运输成本主要由城市与垃圾厂之间的距离决定，所以垃圾处理厂P的选址目标是使得五个城市到垃圾厂P的距离尽量相近。

城市	X坐标	Y坐标
A	1.5	6.8
B	6.0	7.0
C	8.9	6.9
D	3.5	4.0
E	7.4	3.1


y
A
B
C
P
D
E
O
x

【例15.1-1续】该例数学模型如下：

$$\min\{\max(f_1, f_2, \cdots, f_5)\}$$

$$\begin{cases} f_1 = d_{A\to P} = \sqrt{(x-1.5)^2 + (y-6.8)^2} \\ f_2 = d_{B\to P} = \sqrt{(x-6.0)^2 + (y-7.0)^2} \\ f_3 = d_{C\to P} = \sqrt{(x-8.9)^2 + (y-6.9)^2} \\ f_4 = d_{D\to P} = \sqrt{(x-3.5)^2 + (y-4.0)^2} \\ f_5 = d_{E\to P} = \sqrt{(x-7.4)^2 + (y-3.1)^2} \end{cases}$$

巧用0-1变量描述优化问题

2.3 0-1规划规划

2.3.1 选址问题

题目：

某城市拟在东、西、南三区设立商业网点，备选位置有A_1~A_7共7个，如果选A_i，估计投资为b_i元，利润为c_i元，要求总投资不超过B元，规定

东区：A_1、A_2、A_3中至多选2个

西区：A_4、A_5中至少选一个

南区：A_6、A_7中至少选一个

问如何设点使总利润最大？

解决：

解：令 $x_i=\begin{cases}1, & A_i\text{被选中}\\ 0, & A_i\text{没被选中}\end{cases}$

$$\max\ z=\sum_{i=1}^{7}c_ix_i$$

$$\text{s.t.}\begin{cases}\sum_{i=1}^{7}b_ix_i\le B\\ x_1+x_2+x_3\le 2\\ x_4+x_5\ge 1\\ x_6+x_7\ge 1\\ x_i=0\text{或}1,\ i=1,\dots,7\end{cases}$$

2.3.2 指派问题

问题描述：n项任务可由n个人完成，由于专长不同，各人完成各任务的时间也不同，求最优安排。

要求：每人只能完成一项任务，每项任务只能由一人完成

题目：

例： 有一份中文说明书，需译成英、日、德、俄四种文字，分别记作任务E、J、G、R，现有甲、乙、丙、丁四人，他们将中文说明书翻译成不同语种说明书所需的时间如下表所示，问应指派何人去完成何项任务，使所需总时间最少？

	E	J	G	R
甲	2	15	13	4
乙	10	4	14	15
丙	9	14	16	13
丁	7	8	11	9

解：令

$$x_{ij}=\begin{cases}1, & \text{指派第i人去完成第j项任务}\\ 0, & \text{不指派第i人去完成第j项任务}\end{cases}$$

$$\begin{aligned}\min\ z=&2x_{11}+15x_{12}+13x_{13}+4x_{14}+10x_{21}+4x_{22}+14x_{23}+15x_{24}\\&+9x_{31}+14x_{32}+16x_{33}+13x_{34}+7x_{41}+8x_{42}+11x_{43}+9x_{44}\end{aligned}$$

$$x_{11}+x_{12}+x_{13}+x_{14}=1 \quad \text{(甲只能干一项工作)}$$

$$x_{21}+x_{22}+x_{23}+x_{24}=1 \quad \text{(乙只能干一项工作)}$$

$$x_{31}+x_{32}+x_{33}+x_{34}=1 \quad \text{(丙只能干一项工作)}$$

$$x_{41}+x_{42}+x_{43}+x_{44}=1 \quad \text{(丁只能干一项工作)}$$

$$x_{11}+x_{21}+x_{31}+x_{41}=1 \quad \text{(E任务只能一人干)}$$

$$x_{12}+x_{22}+x_{32}+x_{42}=1 \quad \text{(J任务只能一人干)}$$

$$x_{13}+x_{23}+x_{33}+x_{43}=1 \quad \text{(G任务只能一人干)}$$

$$x_{14}+x_{24}+x_{34}+x_{44}=1 \quad \text{(R任务只能一人干)}$$

$$x_{ij}=0 \text{ 或 } 1,\ i,j=1,2,3,4$$

2.3.4 0-1变量的其他用处

- 把相互约束的排斥条件转化为普通的约束条件
- 表示分段函数

2.4 多目标规划

多目标无法直接求解，需要转化为单目标规划

例2：如果有m个相互排斥的约束条件：

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \, , i = 1, 2, \cdots , m$$

使这m个约束条件仅有k个起作用，则引入m个0-1变量y_i 和一个充分大的常数M，将上述约束转化为：

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i + M \left(1 - y_i\right) \, , i = 1, 2, \cdots , m$$

$$\sum_{i=1}^{m} y_i = k$$

分析： $y_i = 1$时，$\sum_{j=1}^{n} a_{ij} x_j \leq b_i + M \left(1 - y_i\right) \Longrightarrow \sum_{j=1}^{n} a_{ij} x_j \leq b_i$ 则该约束条件起作用

$y_i = 0$时，$\sum_{j=1}^{n} a_{ij} x_j \leq b_i + M \left(1 - y_i\right) \Longrightarrow \sum_{j=1}^{n} a_{ij} x_j \leq +\infty$ 则该约束条件不起作用

$\sum_{i=1}^{m} y_i = k$ 限制了仅有k个变量可以取1，即仅有k个约束条件起作用

例3：固定成本问题

某工厂为了生产某种产品，有几种不同的生产方式可供选择，如选定的生产方式投资高，则由于产量大分配到每件产品的变动成本就低；反之，如选定的生产方式投资低，则将来分配到每件产品的变动成本可能增加。因此需要全面考虑。今设三种方式可供选择，令

j分别表示三种生产方式；

x_j表示采用第j种方式时的产量；

c_j表示采用第j种方式时每件产品的的变动成本；

k_j表示采用第j种方式时的固定成本。

分析：由题，第j种生产方式的总成本为

$$P_j = \begin{cases} k_j + c_j x_j \ , & x_j > 0 \\ 0 & x_j = 0 \end{cases} , j = 1,2,3$$

使三种生产方式成本之和最小，则目标函数写为

$$min \quad \sum_{j=1}^{3} P_j$$

这时候我们便注意到一个问题，由于固定成本k_j的存在，P_j是一个分段函数，因此目标函数的表达式不便于进一步明晰。故在决策变量x_j基础上再引入0-1变量y_j，则目标函数写为

$$min \quad \sum_{j=1}^{3} y_j \left(k_j + c_j x_j\right)$$

注意两套决策变量x_j、y_j 间是有联系的，$y_j = 1$表示采用第j种方式生产，则$x_j > 0$；$y_j = 0$表示不采用第j种方式生产，则$x_j = 0$，故需再添加约束条件描述这种联系：

$$\begin{cases} y_j \varepsilon \le x_j \le y_j M \ , & j = 1,2,3 \\ y_j = 0 \ \text{ or } 1 \ , & j = 1,2,3 \end{cases}$$

式中，ε是个充分小的正常数，M是个充分大的正常数。

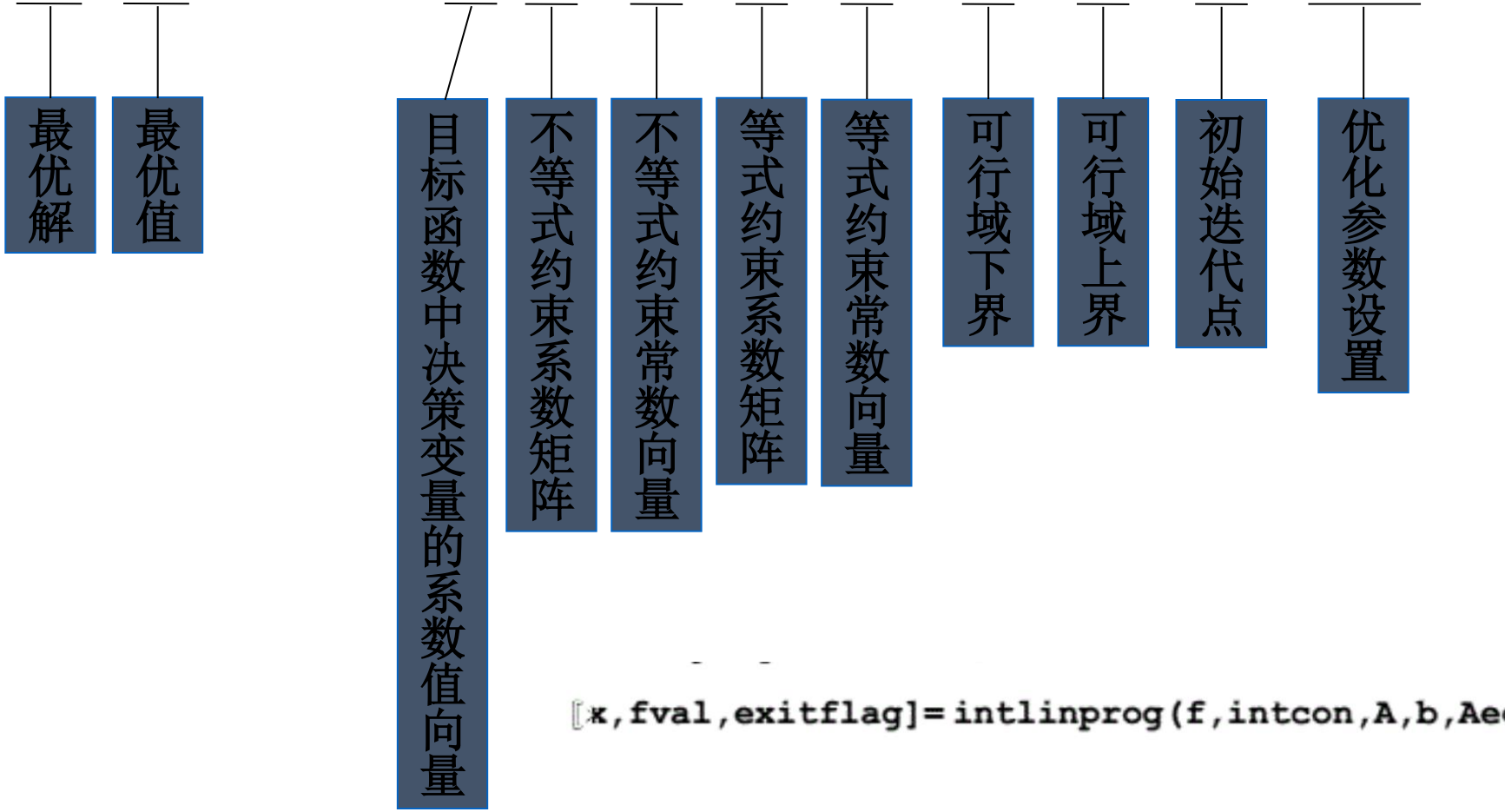
（解释一下这个约束条件，$y_j = 1$时，$y_j \varepsilon \le x_j \le y_j M \Longrightarrow 0^+ \le x_j \le +\infty \Longrightarrow x_j > 0$；$y_j = 0$时，$y_j \varepsilon \le x_j \le y_j M \Longrightarrow 0 \le x_j \le 0 \Longrightarrow x_j = 0$）

解决优化问题的方法

1. 线性优化问题（单纯形）
2. 非线性优化问题（梯度下降）
3. 启发式算法
4. 蒙特卡洛
5. 贪心

linprog函数的调用格式

- **[x, fval] = linprog(f, A, b, Aeq, beq, lb, ub, x0, options)**




```
[x,fval,exitflag]=intlinprog(f,intcon,A,b,Aeq,beq,lb,ub)
```

options参数的字段及说明

字 段	说 明
Diagnostics	显示目标函数的诊断信息，可选字段值如下： 'on' 显示诊断信息 'off' 不显示诊断信息（默认）
Display	结果显示方式，可选字段值如下： 'off' 不显示输出结果 'iter' 显示每一步迭代结果，只适用于大规模内点法和中等规模单纯形算法 'final' 只显示最终结果（默认）
LargeScale	设置迭代算法，可选字段值如下： 'on' 使用大规模内点法（默认） 'off' 使用中等规模算法
MaxIter	设置迭代步数，字段值为正整数
TolFun	设置目标函数的终止容限，对于大规模内点法，默认值为1e-8，对于单纯形法，默认值为1e-6
Simplex	设置中等规模算法，可选字段值如下： 'on' 使用单纯形算法，此时不需要用户指定初始迭代点x0 'off' 使用中等规模作用集算法（默认）

【例13.1-1】某厂生产 三种产品，每种产品生产需经过三道工序：选料、提纯和调配。根据现有的生产条件，可确定各工序有效工时、单位产品耗用工时及利润如表13.1-1所列。试问应如何安排各种产品的周产量，才能获得最大利润？

工 序	单位产品耗用工时（h/kg）			每周有效工时（h）
	A	*B*	*C*	
选料	1.1	1.2	1.4	4600
提纯	0.5	0.6	0.6	2100
调配	0.7	0.8	0.6	2500
利润（元/kg）	12	14	13	

该例数学模型如下：

$$\max \quad z = 12x_1 + 14x_2 + 13x_3$$

$$s.t. \begin{cases} 1.1x_1 + 1.2x_2 + 1.4x_3 \le 4600 \\ 0.5x_1 + 0.6x_2 + 0.6x_3 \le 2100 \\ 0.7x_1 + 0.8x_2 + 0.6x_3 \le 2500 \\ x_1 \ge 0,\ x_2 \ge 0,\ x_3 \ge 0 \end{cases}$$

大规模内点法求解

```
>> f = [-12,-14,-13];
>> A= [1.1, 1.2, 1.4; 0.5, 0.6, 0.6; 0.7, 0.8, 0.6];
>> b= [4600; 2100; 2500];
>> Aeq=[];
>> beq=[];
>> lb=[0; 0; 0];
>> ub=[];

>> [x,fval,exitflag,output,lambda] = linprog(f,A,b,Aeq,beq,lb,ub)
```

单纯形法求解

```
>> options = optimset('LargeScale', 'off', 'Simplex', 'on','Display','iter');
>> [x,fval,exitflag,output,lambda] = linprog(f,A,b,Aeq,beq,lb,ub,[],options)
% 最优解
x =
1.0e+003 *
    0.7500
    1.2500
    1.6250
% 最优值
fval =  -47625
```

```
>> f0 = [1,1,1];
>> a = [-1,-2,-3];
>> f = a.*f0;
>> A = [7, 3, 9; 8, 5, 4; 6, 9, 5];
>> b = [1; 1; 1];
>> Aeq = [];
>> beq = [];
>> lb = [0, 0, 0];
>> ub = [];
>> [x,fval,exitflag,output,lambda] = linprog(f,A,b,Aeq,beq,lb,ub)
```

一、 fminimax函数

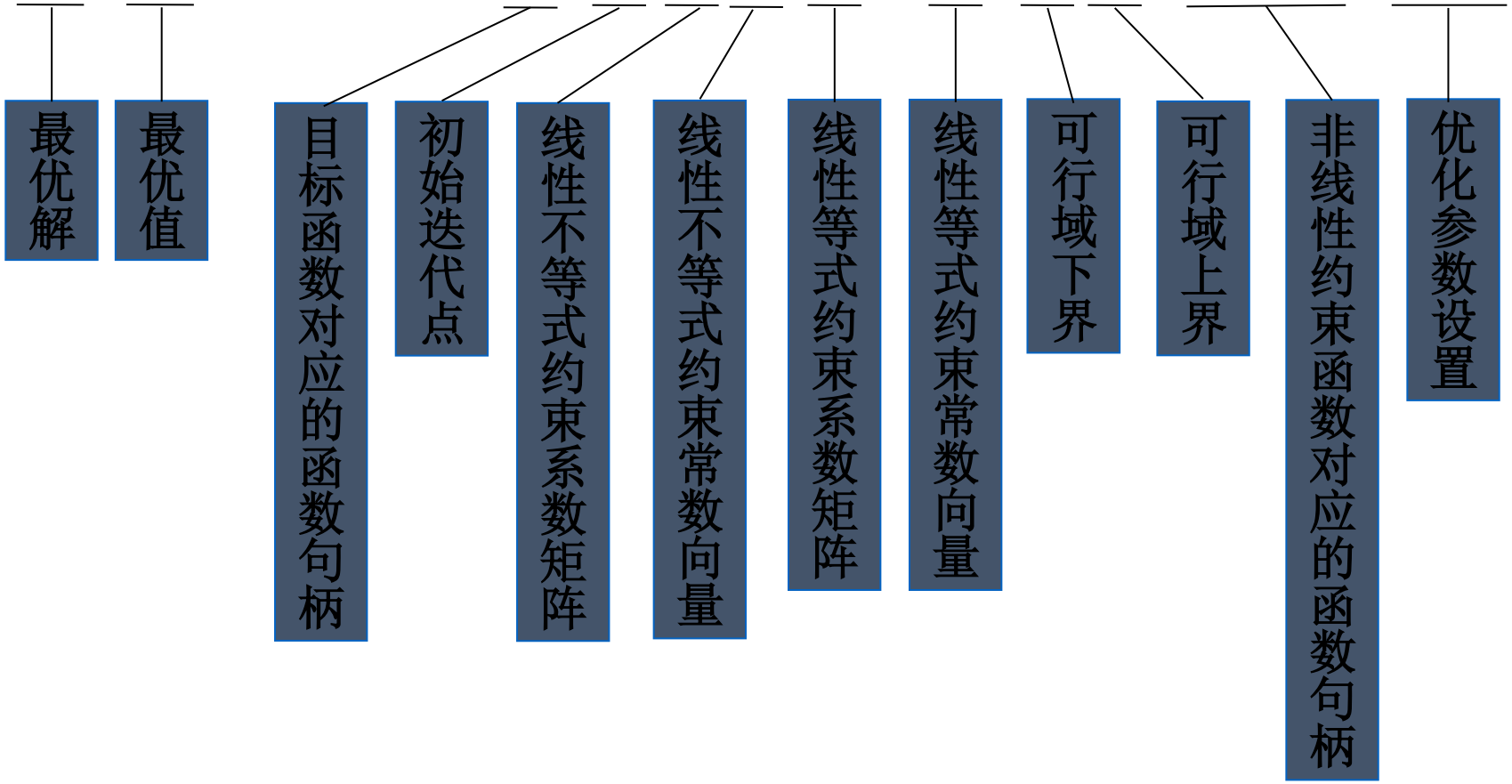
1. 最大最小问题的标准型

$$\min_{x}\{\max(f_1(x), f_2(x),\cdots, f_n(x))\}$$

$$s.t.\begin{cases} c(x)\le 0 \\ ceq(x)\le 0 \\ A\cdot x\le b \\ Aeq\cdot x = beq \\ lb\le x\le ub \end{cases}$$

2. fminimax函数的调用格式

- [x, fval]=fminimax (fun,x0,A,b, Aeq, beq,lb,ub,nonlcon,options)



二、垃圾场选址问题求解

1. 目标函数对应的匿名函数

```
>> minimaxMyfun = @(x)sqrt([(x(1)-1.5)^2+(x(2)-6.8)^2;
   (x(1)-6.0)^2+(x(2)-7.0)^2;
   (x(1)-8.9)^2+(x(2)-6.9)^2;
   (x(1)-3.5)^2+(x(2)-4.0)^2;
   (x(1)-7.4)^2+(x(2)-3.1)^2]);
```

2. 调用fminimax函数进行求解

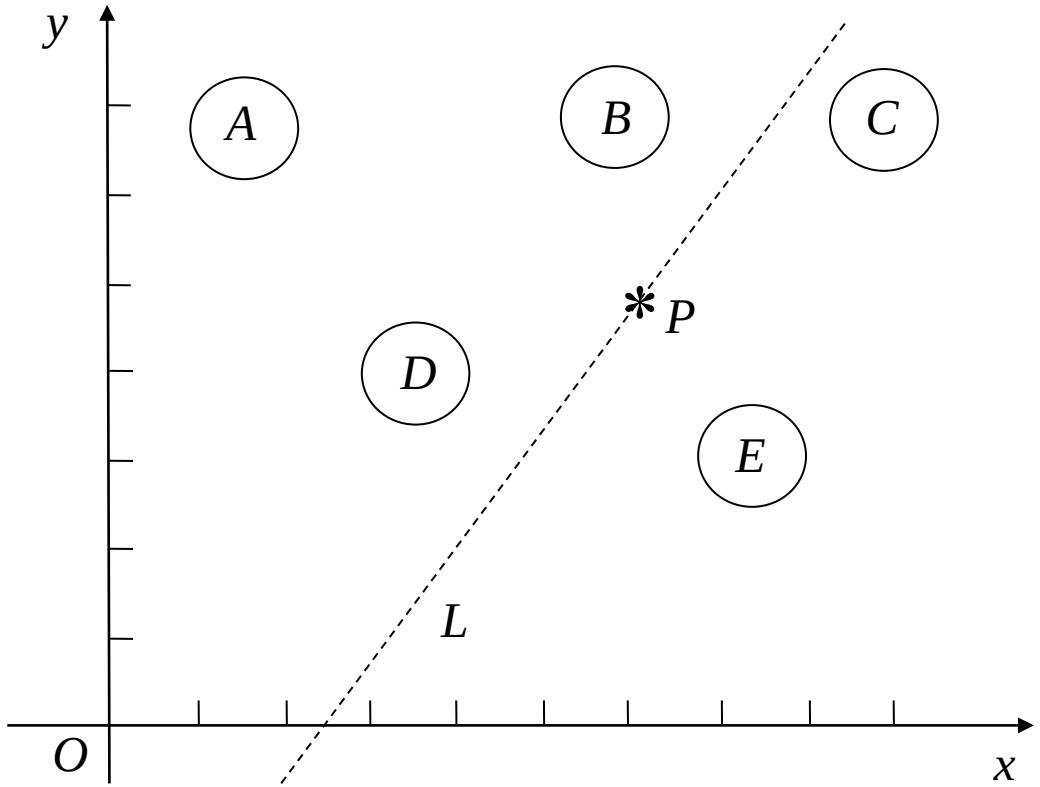
```
>> x0 = [0.0; 0.0];  % 设置初始迭代点
>> [x,fval] = fminimax(minimaxMyfun,x0)
% 最优解
x =
    5.2093
    6.1608
% 最优解对应的各目标函数值向量
fval =
    3.7640
    1.1530
    3.7640
    2.7551
    3.7640
```

二、垃圾场选址问题

【例15.1-1】已知A,B,C,D,E五个城市的位置分布如图15.1-1所示，坐标如表15.1-1所列。计划在A,B,C,D,E五个城市之间建造一个垃圾厂P，使得 五个城市将垃圾运往垃圾厂P的运输成本尽可能的相差不大。 由于运输成本主要由城市与垃圾厂之间的距离决定，所以垃圾处理厂P的选址目标是使得五个城市到垃圾厂P的距离尽量相近。

城市	X坐标	Y坐标
A	1.5	6.8
B	6.0	7.0
C	8.9	6.9
D	3.5	4.0
E	7.4	3.1

【例15.3-1】在原问题上附加约束：A,B,C,D,E五个城市之间有一条高速公路，该公路的直线方程为$y = x-2.5$，为方便转运垃圾，垃圾处理厂需要紧邻公路。



【例15.3-1续】该例数学模型如下：

$$\min\{\max(f_1(x), f_2(x), \cdots, f_5(x))\}$$

$$s.t. \qquad x_1 - x_2 = 2.5$$

$$f_1 = d_{A\to P} = \sqrt{(x-1.5)^2 + (y-6.8)^2}$$

$$f_2 = d_{B\to P} = \sqrt{(x-6.0)^2 + (y-7.0)^2}$$

$$f_3 = d_{C\to P} = \sqrt{(x-8.9)^2 + (y-6.9)^2}$$

$$f_4 = d_{D\to P} = \sqrt{(x-3.5)^2 + (y-4.0)^2}$$

$$f_5 = d_{E\to P} = \sqrt{(x-7.4)^2 + (y-3.1)^2}$$

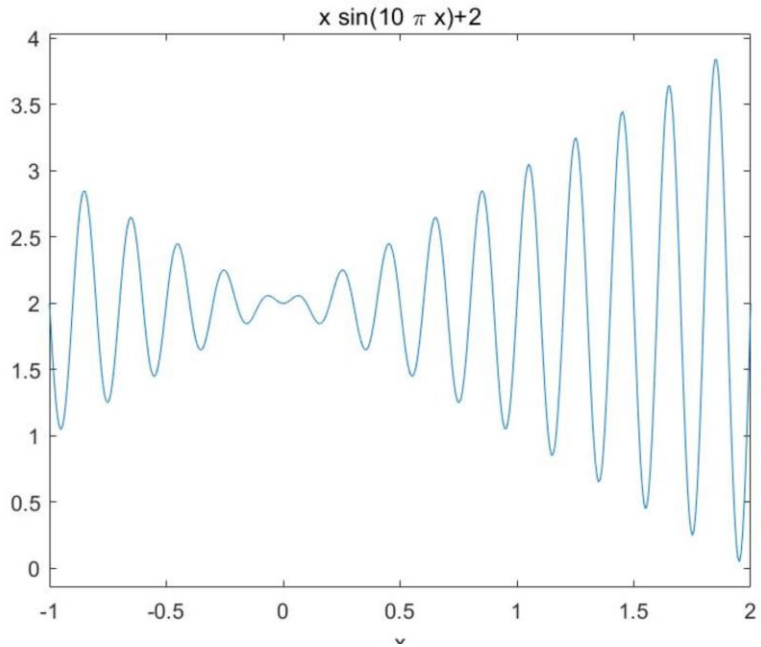
二、调用fminimax函数进行求解

```
>> minimaxMyfun = @(x)sqrt([(x(1)-1.5)^2+(x(2)-6.8)^2;
   (x(1)-6.0)^2+(x(2)-7.0)^2;
   (x(1)-8.9)^2+(x(2)-6.9)^2;
   (x(1)-3.5)^2+(x(2)-4.0)^2;
   (x(1)-7.4)^2+(x(2)-3.1)^2]);

>> x0 = [0.0; 0.0];
>> Aeq = [1,-1];
>> beq = 2.5;

% 调用fminimax函数求解模型（15.3-1）
>> [x,fval] = fminimax(minimaxMyfun,x0,[],[],Aeq,beq)
```

X = ga(FITNESSFCN,NVARS,A,b,Aeq,beq,lb,ub,NONLCON,options) minimizes
with the default optimization parameters replaced by values in OPTIONS.
OPTIONS can be created with the OPTIMOPTIONS function. See OPTIMOPTIONS
for details. For a list of options accepted by ga refer to the
documentation.
X = ga(PROBLEM) finds the minimum for PROBLEM. PROBLEM is a structure
that has the following fields:
fitnessfcn: <Fitness function>
nvars: <Number of design variables>
Aineq: <A matrix for inequality constraints>
bineq: <b vector for inequality constraints>
Aeq: <Aeq matrix for equality constraints>
beq: <beq vector for equality constraints>
lb: <Lower bound on X>
ub: <Upper bound on X>
nonlcon: <Nonlinear constraint function>
intcon: <Index vector for integer variables>
options: <Options created with optimoptions('ga',...)>
rngstate: <State of the random number generator>

例1 一元函数 $f(x)=x\sin 10\pi x+2,x\in(-1,2)$,求 $f(x)$ 最大值

```
fun1 = @(x)x.*sin(10*pi*x)+2;ezplot(fun1,[-1,2])
```



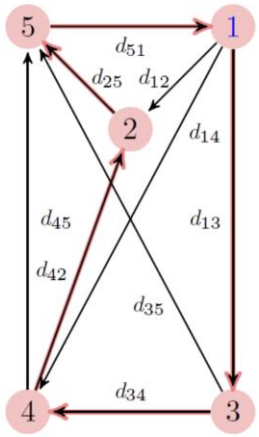
蒙特卡洛也是一种方法

```
>> f1= @(x)(-x*sin(10*pi*x)-2);[x,fval] = ga(f1,1,[],[],[],[],-1,2)
Optimization terminated: average change in the fitness value less than options.FunctionTolerance.

x =    1.4507
fval =   -3.4503
```

当 x = 1.4507，f(x)取最大值 fmax = 3.4503.

复杂问题举例

旅行商问题 (TSP): 假设有一个旅行商人要拜访 n 个城市, d_{ij} 表 示两城市间距离. x_{ij} 为 0,1 变量, 表示拜访路径是否包含路径 d_{ij}.



- 限制: 每个城市必需且只能拜访一次, 最后要回到原来出发的城市, 即

$$\sum_{i=1}^{n} x_{ij} = \sum_{j=1}^{n} x_{ij} = 1$$

- 目标: 路径的选择目标是要求所得路径路程为所有路径之中的最小值, 即

$$\min \sum_{i \neq j} d_{ij} x_{ij}$$

```
%% 遗传算法工具箱解决TSP问题
%  author      YuZi      202-8-18
%% 概述
% 问题描述，14个城市旅行，从某个城市出
% 发访问每个城市一次且仅一次， 最后回到出发城市，如何安排才使其所走路线最短。
%% 问题分析
%     如果将该问题转为0，1规划，X_ij=1变量代表第i个次序经过第j号城市，则该问题
%     需要定义14*14个变量，如果是127个城市，则需要127*127个变量，求解量按照
%     城市数量的平方量级增长，工具箱是无法求解的。
%     换个思路，例如一条城市顺序表C={1,2,3,4,5,6,7,8,9}，
% 先将即将探访的城市在顺序表中的顺序记录在数列C中；
% 然后每走一个城市，将其在顺序表C中划去，
% 那么下一个要走的城市在新顺序表中的位置将被记录在p中。
% 例如，若实际的行走顺序T={1,2,3,4,5,6,7,8,9}
% 则对应的解x={1,1,1,1,1,1,1,1,1}；
% 若实际的行走顺序为T={2,1,5,6,7,9,3,4,8}，
% 则对应的染色体x={2,1,3,3,3,4,1,1,1}。
% 在这种情况下，只需要构建14个变量，就能构建出一个符合实际情况的路径。
% 标准答案的最优解约为29，需要多次运行，解集中在34-30附近
```

2. 模型建立的写法：对于大模型可以分成几个小标题，一步一步建立，每一个公式都要有一段话描述，解释清楚公式里第一次出现的符号是什么意思，代表什么，公式整体的含义。每一个公式都要有标号，。所有的都公式都建立好以后，如果涉及到优化模型，最后给出一个总括性的模型，目标函数，约束条件。如果没有大模型，则不需要重复写公式。算法的描述，建议不要出现在模型建立中。放在模型求解中。↓

3. 模型求解 的写法。分步写清楚，怎么求解模型建立中的公式。比如 第一步，将什么变量代入什么公式（公式编号）使用什么方法求解什么结果↓

第二步，。。。 第三步。。。，如果涉及到循环和判断都要在步骤中写清楚，代入到哪个编号的公式中求出来的，都要交代清楚。和模型建立要形成呼应。你们现在大部分都写的很孤立。如果在某一步中涉及到一个大的算法，比如你在第二步中要涉及到遗传算法。就在第二步下边再加小标号，写清楚遗传算法的编码，选择，交叉，变异，精英排序等等过程。 如果你们的算法除了常规步骤还有创新，想总括的介绍一下这个遗传算法。可以在整个模型求解的开头，先写一个总括性的话。比如，“在模型求解过程中，我们会用到 xx，xx 算法。接下来先介绍模型求解中涉及到的 xx 算法，然后给出整体求解步骤。”再下设几个小标题，1. 遗传算法 2.xx 算法 3. 求解步骤 第一步，第二步，第三步...↓

如果对于优化问题没有办法写成大的优化方程式，比如使用遗传算法解决路径规划问题。那么遗传算法如何套用在具体问题中，就是你的建模过程，这时遗传算法就写在模型建立中，在模型求解时只需写出求解的算法步骤即可。

1.1 游戏地图介绍

游戏地图由多个图层构成，图层栅格常被称为瓦片，每个瓦片都有自己的唯一坐标（图1）。在碰撞检测层中，通过定义瓦片的“键——值”对定义该瓦片作为可通过节点还是障碍物节点。

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)	(5,0)	(6,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)	(5,1)	(6,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)	(5,2)	(6,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)	(5,3)	(6,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)	(5,4)	(6,4)
(0,5)	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)	(6,5)

图1 游戏地图示意

2.1 确定种群规模与遗传算子的执行顺序

种群规模过大，会影响计算速度；过小，又会降低全局搜索能力。本文定义最佳种群规模 *bestpop* 的计算公式为

$$bestpop=[n/\min\{row,col\}] \tag{1}$$

式中 n 为地图的可通过节点数；*row* 为地图的行节点数；*col* 为地图的列节点数；$\min\{row,col\}$ 表示行、列较小者的节点数；[ ]为四舍五入取整运算。式(1)中用行或列的较小者中的节点数作为分子，而不用行、列的平均节点数，目的是首要考虑搜索成功率，而把计算速度放在次要位置。

不同于传统遗传算法，为使交叉算子和变异算子更好地配合，本文将遗传算子执行顺序设置为：选择、变异、交叉。

2.2 设计适应度函数

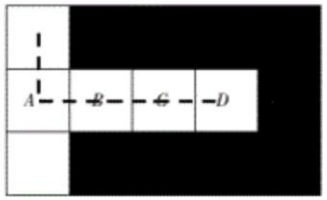
本文直接将染色体的长度定义为染色体的适应度，即

$$fitness(x_i)=distance(x_i),i=1,2,\ldots,M \tag{2}$$

式中 x_i 为第 i 个个体，$distance(x_i)$ 为第 i 个个体的长度，M 为种群规模。相应地调整进化规则：保留适应度小的染色体，淘汰适应度大的染色体。

2.3 生成初始种群

传统遗传算法在生成染色体时会产生环路，从而产生大量较差个体。初始群体的优秀程度是提高算法时间效率的关键[12]，为了避免产生环路，采用文献[13]的做法：凡是加入到染色体的节点，都做一个标记，只有没有被标记过的节点才能被选作新的节点。同时又产生一个新的问题：由于加入到染色体的节点都做了标记，如果生成染色体的过程中遇到了“死胡同”，程序将无法退出“死胡同”。因此，首先判断当前基因邻居节点中可行走的节点数是否为0。如果是，则从染色体中删除当前基因，再考察当前基因的上一个基因。如图2，生成染色体的过程中已经走进了“死胡同”，此时节点 D 位于染色体头部，节点 D 的邻居节点中可行走的节点数为0，因此把节点 D 从染色体头部删除。继而考察节点 C，因为之前考察过节点 D，已经对节点 D 做了标记，故节点 C 的邻居节点中可行走的节点数也为0，继续删除，直到考察节点 A，从而退出“死胡同”。



2.5 变异操作

由于传统的变异操作采用的是随机变异的形式，收敛速度慢，多位学者对其进行了改进。文献[7]在染色体中随机选择一个基因作为变异点，从起点到变异点的基因保持不变，由变异点开始，随机选择当前节点的可通过邻居节点加入到染色体，直到找到目标节点为止。文献[8]是在染色体中随机选择一个基因作为变异点，再从该变异点的邻居节点中随机选择一个节点作为新染色体的中间节点，用 A* 算法生成一条从该中间节点到起点的路径，再用 A* 算法生成一条从该中间节点到目标点的路径，连接两条路径作为新的路径。为满足游戏实时性的要求，设计了一种启发式深度优先搜索算法，用于变异操作，具体步骤如下：

1）从待变异父代路径 V_{old} 中随机选择变异节点 i（不包括起点和目标点）；

2）由节点 i 开始，选择其邻居可通过节点中距离起点最近的节点加入到染色体，直到找到起点为止，作为路径 V_{front}；

3）再由节点 i 开始，选择其邻居可通过节点中距离目标点最近的节点加入到染色体，直到找到目标点为止，作为路径 V_{back}；

4）连接 V_{front}、V_{back}，并将重复的节点 i 从中删除一个，形成新路径 V_{new} 作为子代路径。

此时，模型建立与模型求解实际上是写到一起了。这种复杂问题无法按照传统优化问题直接写出优化方程，然后套用标准形方程进行求解。建模过程必然伴随着设计算法的过程，为了把算法说清楚就必然要写出求解步骤。因此，多借用参考文献的模式展现自己的建模求解思路是关键，不能生搬硬套模板。必须先要在模型建立中把这些就算细节都写清楚，然后再在模型求解中写出笼统的大步骤即可。写清楚是关键！！！！