线性规划

2023年5月15日

目录

线性规划标准型及解的概念和性质

线性规划模型

线性规划模型：

（1）一组决策变量；（2）一个线性目标函数；

（3）一组线性的约束条件。

线性规划模型（*LP*）的一般形式：

$$\min(\max)\ \sum_{i=1}^{n} c_i x_i \quad s.t. \begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq (=, \leq) b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq (=, \leq) b_2 \\ \cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq (=, \leq) b_m \\ x_i \geq (=, \leq) 0, i = 1, 2, \cdots, n \end{cases}$$

线性规划标准型

标准型：

$$\min \quad \sum_{i=1}^{n} c_i x_i \qquad s.t. \begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \\ x_i \ge 0, i = 1, 2, \cdots, n \end{cases}$$

$$c = (c_1, c_2, \cdots, c_n)^T, b = (b_1, b_2, \cdots, b_m)^T \ge 0,$$

$$x = (x_1, x_2, \cdots, x_n)^T, A = (a_{ij})_{m\times n}$$

线性规划模型及标准型

线性规划标准型可化为：

$$\min c^T x \quad s.t. \begin{cases} Ax = b \\ x \geq 0 \end{cases}$$

线性规划模型化为标准型的步骤？

线性规划模型转化为标准型

化标准型：

（1）目标函数：

原问题目标函数: $\max \quad c^T x \Rightarrow \min -c^T x$

线性规划模型转化为标准型

化标准型：

（2）约束条件：（分为以下几种情况）

1）原问题条件：

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \le b_i$$

$$\Rightarrow \begin{cases} a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + \boxed{x_{n+i}} = b_i \\ x_{n+i} \ge 0 \end{cases}$$

x_{n+i} 称为松弛变量

线性规划模型转化为标准型

2）原问题条件:

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geq b_i$$

$$\Rightarrow \begin{cases} a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - \boxed{x_{n+i}} = b_i \\ x_{n+i} \geq 0 \end{cases}$$

x_{n+i} 称为剩余变量。

线性规划模型转化为标准型

3）原问题：x_i无非负约束

令
$$\begin{cases} x_i = u_i - v_i \\ u_i , v_i \geq 0 \end{cases}$$

线性规划解的概念及性质

线性规划解的概念：

线性规划模型 (LP)

$$\min \quad z = c^T x \quad (1)$$

$$s.t. \begin{cases} Ax = b & (2) \\ x \ge 0 & (3) \end{cases}$$

可行解：满足（2）、（3）式的解 $x = (x_1, x_2, \cdots, x_n)^T$ 称为（LP）的可行解。

可行域： $D = \{x \mid Ax = b, x \ge 0\}$。

线性规划问题的可行域D是凸集。

线性规划的基本可行解

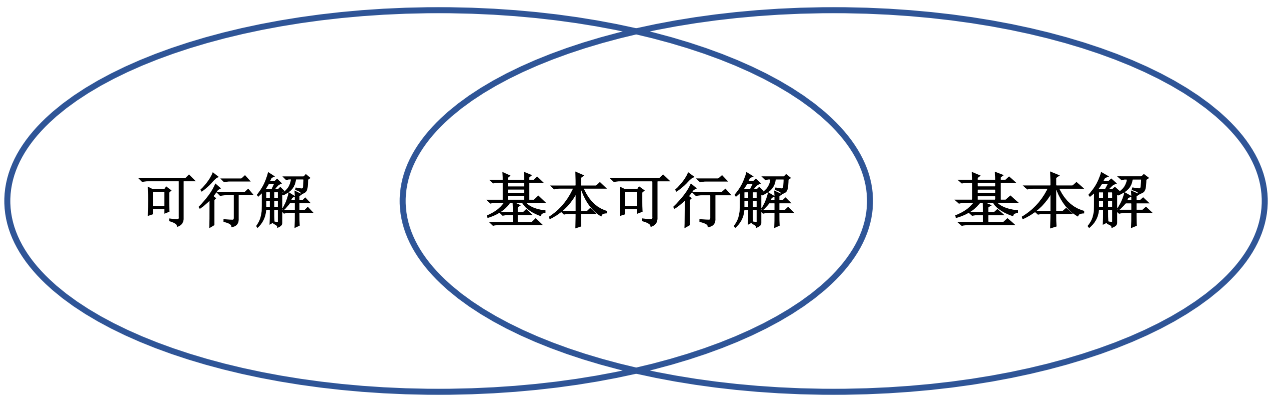
基：设A为$m \times n$的系数矩阵，秩为m。若B为A中$m \times m$阶的非退化子阵，则称B为A的（或（LP）问题）一个基。

基本解：取定线性规划问题的基B，令非基变量取零，求得基变量的取值 $B^{-1}b$ ，称解 $(B^{-1}b, 0)^T$ 为对应于基B的基本解。

基本可行解：**满足约束条件**$\mathrm{Ax} = \mathbf{0}, \boldsymbol{x} \geq \mathbf{0}$ (可行解）的基本解称为基本可行解。

线性规划解的性质

线性规划解的关系?



定理 线性规划基本定理 对于线性规划的标准型，有如下两个命题。

1、如果存在可行解，那么一定存在基本可行解；

2、如果存在最优可行解，那么一定存在最优基本可行解。

➡ 线性规划问题的求解→在有限数量的基本可行解上进行搜索

单纯形算法

单纯形算法

思路：从一个基本可行解开始，判断其是否为最优解。是则算法结束。不是，则转换到另一个更好的基本可行解，直到找到最优解，或判断出不存在最优解

- **单纯形法的基本步骤:**

1.根据初始基本可行解构造增广矩阵规范型；

2.计算非基变量的检验数；

3.如果对于所有j都有$r_j \geq 0$，则停止运算，当前基本可行解即是最优解；否则，进入下一步；

4.从小于零的检验数中选择一个检验数$r_q < 0$; q 列向量进入基矩阵；

5.如果不存在 $y_{iq} > 0$，则停止运算，问题有无界解；否则，计算 $p = argmin_i\{y_{i0}/y_{iq}:y_{iq} > 0\}$；如果求解得到多个满足条件的下标$i$，则令$p$等于最小的下标值。

6.以元素(p,q)为枢轴元素进行枢轴变换，更新增广矩阵规范型；

7.转到步骤2。

单纯形法的矩阵形式

- 令$\boldsymbol{A}$的前m列是基向量，组成了$m \times m$非奇异矩阵$\boldsymbol{B}$。$\boldsymbol{A}$的非基列向量组成了$m \times (n-m)$矩阵$\boldsymbol{D}$， 价值系数向量可相应地写为$\boldsymbol{c}^{\mathrm{T}} = [\boldsymbol{c}_B^{\mathrm{T}}, \boldsymbol{c}_D^{\mathrm{T}}]$。则

$$\textbf{minimize} \quad \boldsymbol{c}_B^T \boldsymbol{x}_B + \boldsymbol{c}_D^T \boldsymbol{x}_D$$

$$\textbf{subject to} \quad [\boldsymbol{B}, \boldsymbol{D}] \begin{bmatrix} \boldsymbol{x}_B \\ \boldsymbol{x}_D \end{bmatrix} = \boldsymbol{B}\boldsymbol{x}_B + \boldsymbol{D}\boldsymbol{x}_D = \boldsymbol{b}$$

$$\boldsymbol{x}_B \geq \boldsymbol{0}, \boldsymbol{x}_D \geq \boldsymbol{0}$$

- 如果$\boldsymbol{x}_D = 0$，那么$\boldsymbol{x} = [\boldsymbol{x}_B^{\mathrm{T}}, \boldsymbol{x}_D^{\mathrm{T}}]^{\mathrm{T}} = [\boldsymbol{x}_B^{\mathrm{T}}, \boldsymbol{0}]^{\mathrm{T}}$是关于基$\boldsymbol{B}$的基本可行解.基变量$\boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{b}$， 故基本可行解为 $\boldsymbol{x} = \begin{bmatrix} \boldsymbol{B}^{-1}\boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix}$,相应的目标函数值为

$$z_0 = \boldsymbol{c}_B^T \boldsymbol{B}^{-1} \boldsymbol{b}$$

单纯形法的矩阵形式

- 另一方面，如果$x_D \neq 0$，那么$x = [x_B^T, x_D^T]^T$不是基本解。此时,x_B可以表示为 $\boldsymbol{x_B = B^{-1}b - B^{-1}Dx_D}$,相应的目标函数值为

$$z = \boldsymbol{c_B^T B^{-1} b} + \left(\boldsymbol{c_D^T - c_B^T B^{-1} D}\right)\boldsymbol{x_D}$$

- 定义$\boldsymbol{r}_D^T = \boldsymbol{c}_D^T - \boldsymbol{c}_B^T\boldsymbol{B}^{-1}\mathbf{D}$，可得$z = z_0 + \boldsymbol{r}_D^T \boldsymbol{x}_D$，**向量$r_D$中的元素是非基变量的检验数。**

① 如果$\boldsymbol{r}_D \geq 0$， 那么关于基$\boldsymbol{B}$的基本可行解就是最优解

② 如果$\boldsymbol{r}_D$中存在负数, 则可以通过将x_D中相应的值从零变为正数，使目标函数值变小，即变换基矩阵。

单纯形法的矩阵形式

- 在增广矩阵$[\boldsymbol{A}, \boldsymbol{b}]$的底部增加一行价值系数向量$\boldsymbol{c}^T$，如下式所示:

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{c}^{\mathrm{T}} & 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{D} & \boldsymbol{b} \\ \boldsymbol{c}_B^{\mathrm{T}} & \boldsymbol{c}_D^{\mathrm{T}} & 0 \end{bmatrix}$$

该矩阵称为线性规划的单纯形表，该单纯形表包含了线性规划的所有信息。

- 对单纯形表进行初等行变换，可得

$$\begin{bmatrix} \boldsymbol{I}_m & \boldsymbol{0} \\ -\boldsymbol{c}_B^T & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{B}^{-1} & \boldsymbol{0} \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{B} & \boldsymbol{D} & \boldsymbol{b} \\ \boldsymbol{c}_B^T & \boldsymbol{c}_D^T & 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_m & \boldsymbol{B}^{-1}\boldsymbol{D} & \boldsymbol{B}^{-1}\boldsymbol{b} \\ \boldsymbol{0}^{\mathrm{T}} & \boldsymbol{c}_D^{\mathrm{T}} - \boldsymbol{c}_B^{\mathrm{T}}\boldsymbol{B}^{-1}\boldsymbol{D} & -\boldsymbol{c}_B^T\boldsymbol{B}^{-1}\boldsymbol{b} \end{bmatrix}$$

获得关于基B 的标准单纯形表。

① 最后一列的前m 个元素$\mathbf{B}^{-1}\boldsymbol{b}$ **就是关于基B 的基变量**

② 最后一行中, $\boldsymbol{c}_D^T - \boldsymbol{c}_B^T\boldsymbol{B}^{-1}\boldsymbol{D}$**是检验数**

③ $-\boldsymbol{c}_B^T\boldsymbol{B}^{-1}\boldsymbol{D}$是当前基本可行解下目标函数值取负后的结果。

单纯形法的矩阵形式

- 如何在两个基矩阵对应的标准单纯形表之间相互转换?
- 可以采用初等行变换的方法来完成该变换，其中,y_{ij}和y'_{ij}分别表示原增广矩阵规范型和更新后的增广矩阵规范型中$(i\ ,j)$ 处的元素。

$$y'_{ij} = y_{ij} - \frac{y_{pj}}{y_{pq}} y_{iq}, i \neq p$$

$$y'_{pj} = \frac{y_{pj}}{y_{pq}}$$

- 原单纯形法的单纯形表是$(m+1)\times(n+1)$矩阵。在**修正单纯形法**中，并不计算 $B^{-1}D$ ，而是仅仅追踪基变量和修正的单纯形表$[B^{-1}, B^{-1}b]$，**该单纯形表是$m\times(m+1)$矩阵。**

修正单纯形算法

1，针对初始基本可行解构造修正的单纯形表$[B^{-1}, y_0]$，其中$y_0 = \boldsymbol{B}^{-1}\boldsymbol{b}$。

2，计算当前的检验数：$\boldsymbol{r}_D^T = \boldsymbol{c}_D^T - \boldsymbol{c}_B^T\boldsymbol{B}^{-1}\boldsymbol{D}$

3，如果对所有 j 都有 $r_j \geq 0$ 成立，则算法停止，当前基本可行解已是最优解；

4，从小于零的检验数中选择最小的检验数 $r_q < 0$，a_q 是进基向量并计算

$$y_q = B^{-1}a_q$$

5，如果不存在$y_{iq} > 0$，则算法停止，问题有无界解；否则，计算

$$p = \arg\min_i\{y_{i0}/y_{iq}:\ y_{iq} > 0\}$$

6，构造增广矩阵的修正单纯形表$[B^{-1}, y_0, y_q]$，以最后一列的第p个元素作为枢轴元素，开展枢轴变换，由新得到的增广修正单纯形表的前$m+1$列组成更新的修正单纯形表（删除增广修正单纯形表的最后一列）。

7，返回步骤2。

单纯形算法 Matlab实验

实验示例

例：生产I、II两种产品，要占用A、B设备及调试时间，每件产品机时利润如表所示

	产品I	产品II	每天可用时间
占用A机时	0	5	15
占用B机时	6	2	24
调试时间	1	1	5
利润	2	1	

如何生产使每天利润最大？

实验示例

确定变量：　生产两种产品的数量 x_1, x_2

每天利润：　$2x_1 + x_2$

约束条件：　$5x_2 \le 15$　　A机时约束

$6x_1 + 2x_2 \le 24$　　B机时约束

$x_1 + x_2 \le 5$　　调试时间约束

$x_1 \ge 0, x_2 \ge 0$　　非负约束

实验示例的标准型

用matlab实现**简单、两阶段单纯形法**：

模型化为标准型：

$$\min -2x_1-x_2$$

$$s.t. 5x_2+x_3=15$$

$$6x_1+2x_2+x_4=24$$

$$x_1+x_2+x_5=5$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0,$$

Matlab实现简单单纯形法

```
function [x, v] = simplex( A, b, c ,v)
% 单纯形法的实现
% x: 目标函数的最优解
% A: 约束函数的系数矩阵
% b: 约束函数的常数列向量
% c: 目标函数的系数向量
% v: 选取的基向量序号

[m, n] = size(A);
% 构造增广矩阵规范式
cB=c(v(:));
r = c'-cB'*A; % 判别数
cost = -cB'*b;
tabl=[A b;r cost];

%OPTIONS(5) specifies how the pivot element is selected;
% 0=choose the most negative relative cost coefficient;
% 1=use Bland’s rule.
options(5) = 0;
```

```
while ones(1,n)*(r' >= zeros(n,1)) ~= n
% 选择出进基向量，以合适方式选取小于0的判别数
    if options(5) == 0
        [r_q,q] = min(r);
    else
        q=1;
        while r(q) >= 0
            q=q+1;
        end
    end
```

Matlab实现简单单纯形法

```
 min_ratio = inf;      p=0;
%如果不存在 y_iq > 0，  则停止运算，问题有无界解；
否则，计算p = argmin_i{y_i0/y_iq:y_iq > 0}
 for i=1:m
       if tabl(i,q)>0
           if tabl(i,n+1)/tabl(i,q) < min_ratio
               min_ratio = tabl(i,n+1)/tabl(i,q);
               p = i;
           end %if
       end %if
 end %for
       if p == 0
           disp('Problem unbounded');
           break;
       end %if

%以元素(p,q)为枢轴元素进行枢轴变换，更新增广
矩阵规范型；
tabl=pivot(tabl,p,q);
         v(p) = q;
         r = tabl(m+1,1:n); % 判别数
end %while
         x=zeros(n,1);
    x(v(:))=tabl(1:m,n+1);
end
```

```
function Mnew=pivot(M,p,q)
%Mnew=pivot(M,p,q)
%Returns the matrix Mnew resulting
from pivoting about the (p,q)th element of
the given matrix M.
for i=1:size(M,1),
if i==p
Mnew(p,:)=M(p,:)/M(p,q);
else
Mnew(i,:)=M(i,:)-M(p,:)*(M(i,q)/M(p,q));
end %if
end %for
```

实验示例

简单单纯形法：

```
A=[0 5 1 0 0; 6 2 0 1 0; 1 1 0 0 1];
b=[15; 24;5];
c=[-2;-1;0;0;0];
v=[3;4;5];
[ X, v ] = simplex( A, b, c,v )
```

```
X =

    3.5000
    1.5000
    7.5000
         0
         0


v =

     3
     1
     2
```

Matlab实现修正单纯形法

```
function [x,v,Binv]=revsimp(A,b,c,v,Binv)
%v: 选取的基向量序号
%Binv 为基B的逆
[m, n] = size(A);
% 由A选取的列向量组成的基矩阵
cB=c(v(:));
y0 = Binv*b;
r = c'- cB'*Binv *A; % 计算当前的检验数
options(5) = 1;
while ones(1,n)*(r' >= zeros(n,1)) ~= n
    % 从小于0的检验数中选择最小检验数
    if options(5) == 0
        [r_q,q] = min(r);
    else
        q=1;
        while r(q) >= 0
            q=q+1;
        end
    end
yq = Binv*A(:,q);
    min_ratio = inf;
    p=0;
    % 如果不存在yi大于0，则算法停止问题有无界解；否则计算
    for i=1:m
        if yq(i)>0
            if y0(i)/yq(i) < min_ratio
                min_ratio = y0(i)/yq(i);
                p = i;
            end
        end
    end %for
    if p == 0
        disp('Problem unbounded');
        break;
    end
```

实验示例

修正单纯形法：

```
A=[0 5 1 0 0; 6 2 0 1 0; 1 1 0 0 1];
b=[15; 24;5];
c=[-2;-1;0;0;0];
v=[3;4;5];
Binv=eye(3);
[x,v,Binv]=revsimp(A,b,c,v,Binv)
```

```
x =

    3.5000
    1.5000
    7.5000
         0
         0


v =

     3
     1
     2


Binv =

    1.0000    1.2500   -7.5000
         0    0.2500   -0.5000
         0   -0.2500    1.5000
```

两阶段单纯形法

对于标准形式的线性规划问题，可构造相应的人工问题：

$$\begin{aligned} \text{minimize} \quad & y_1 + y_2 + \cdots + y_m \\ \text{subject to} \quad & [A, I_m]\begin{bmatrix} x \\ y \end{bmatrix} = b \\ & \begin{bmatrix} x \\ y \end{bmatrix} \geq 0 \end{aligned}$$

其中，$y = [y_1, \cdots, y_m]^T$ 是由人工变量构成的人工向量。

人工问题有一个明显的初始基本可行解：$\begin{bmatrix} 0 \\ b \end{bmatrix}$

因此可以直接采用单纯形法求解该问题。

命题 16.1

原线性规划问题存在基本可行解，当且仅当相应的人工问题存在一个使目标函数值为0的最优解。

Matlab实现两阶段单纯形法

第一步：用单纯形法求解若含的线性规划问题，求解后所得可行解目标函数值为0表明原规划问题有基可行解。
转入第二步：去掉人工变量，得到第二阶段的单纯形表，继续用单纯形法求解。

```
function [ x,v ] = twophase( A, b, c )
n=length(c); m=length(b);
%选择人工问题的进基向量
v=n*ones(m,1);
for i=1:m
    v(i)=v(i)+i;
end
%单纯形法求解人工问题，获得初始可行解
[x,v]=simplex([A
eye(m)],b,[zeros(n,1);ones(m,1)],v);
%去掉人工变量求解原问题的最优解
Binv=inv(A(:,v));
A=Binv*A;
b=Binv*b;
[x,v]=simplex(A,b,c,v);
end
```

实验示例

两阶段单纯形法：

```
A=[0 5 1 0 0; 6 2 0 1 0; 1 1 0 0 1];
b=[15; 24;5];
c=[-2;-1;0;0;0];
[ x,v ] = twophase( A, b, c )
```

```
x =

    3.5000
    1.5000
    7.5000
         0
         0

v =

     2
     3
     1
```

仿射尺度法和Matlab实验

仿射尺度法

$$\begin{aligned} \text{minimize} \quad & \boldsymbol{c}^\top \boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geqslant \boldsymbol{0} \end{aligned}$$

- 假设可行解$\boldsymbol{x}^{(0)}$是一个严格内点（$\boldsymbol{x}^{(0)}$的所有元素都大于0）。希望沿着某个搜索方向$\boldsymbol{d}^{(0)}$寻找一个新的点$\boldsymbol{x}^{(1)}$，从而减少目标函数值，也就是说$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0 \boldsymbol{d}^{(0)}$，其中$\alpha_0$为步长
- 目标函数的负梯度方向是$-\boldsymbol{c}$，如果$\boldsymbol{d}^{(0)} = -\boldsymbol{c}$，那么点$\boldsymbol{x}^{(1)}$有可能不在可行集内。由于$\boldsymbol{x}^{(0)}$是可行点，所以有$\boldsymbol{A}\boldsymbol{x}^{(0)} = \boldsymbol{b}$；下一个迭代点$\boldsymbol{x}^{(1)}$应该满足$\boldsymbol{A}\boldsymbol{x}^{(1)} = \boldsymbol{b}$，因此$\boldsymbol{A}\left(\boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)}\right) = \alpha_0 \boldsymbol{A}\boldsymbol{d}^{(0)} = 0$
- 为了保证$\boldsymbol{x}^{(1)}$在可行集内，向量$\boldsymbol{d}^{(0)}$必须位于$\boldsymbol{A}$的零空间内。

仿射尺度法

- 为了能够使$\boldsymbol{d}^{(0)}$既是$\boldsymbol{A}$的零空间向量又尽可能地“接近”$-\boldsymbol{c}$，通常将$-\boldsymbol{c}$正交投影到$\boldsymbol{A}$的零空间，并直接把投影作为$\boldsymbol{d}^{(0)}$，即$\boldsymbol{d}^{(0)}=-\boldsymbol{P}\boldsymbol{c}$，其中称为正交投影算子：

$$\boldsymbol{P}=\boldsymbol{I}_n-\boldsymbol{A}^{\top}(\boldsymbol{A}\boldsymbol{A}^{\top})^{-1}\boldsymbol{A}$$

- 通过下式得到一个新的可行点$\boldsymbol{x}^{(1)}$：

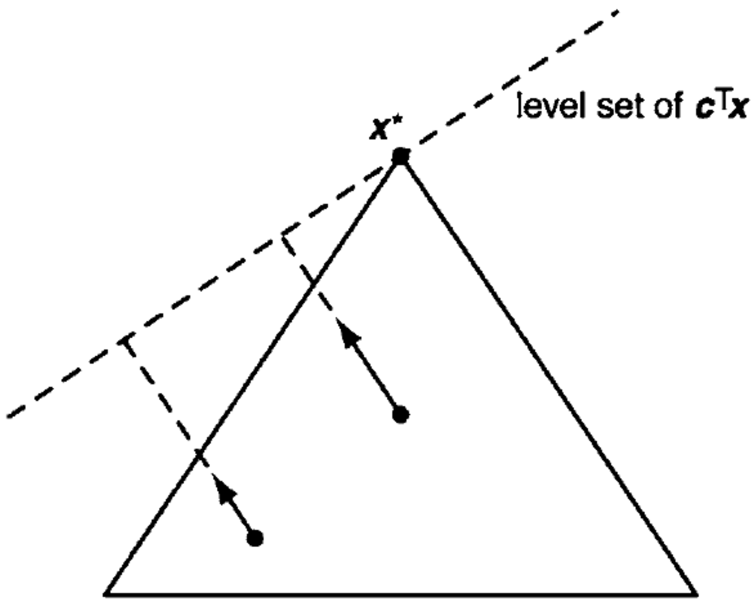
$$\boldsymbol{x}^{(1)}=\boldsymbol{x}^{(0)}-\alpha_0\boldsymbol{P}\boldsymbol{c}$$

步长α_0的选择方式将稍后进行介绍。$\boldsymbol{x}^{(1)}$的求解公式可视为梯度投影算法的一次迭代。

仿射尺度法

初始点$\boldsymbol{x}^{(0)}$应该选择靠近可行集中心的点。

- 如果从可行集的中心出发，可以在搜索方向上选取较大的步长；而从非中心点出发，只能选择较小的步长。
- 因此，从中心点出发进行一次大步长的迭代，能够使目标函数值下降地更多。



仿射尺度法

- 假设初始点$\boldsymbol{x}^{(0)}$是可行的，但不是中心点，则可以通过仿射尺度变换将其变换到中心。
- 为简化分析，假设$\boldsymbol{A}=[1,1,\cdots,1]/n, \boldsymbol{b}=[1]$。可行集的中心是$\boldsymbol{e}=[1,1,\cdots,1]^T$，为了将$\boldsymbol{x}^{(0)}$变换到$\boldsymbol{e}$，需要采用如下仿射尺度变换:$\boldsymbol{e}=\boldsymbol{D}_0^{-1}\boldsymbol{x}^{(0)}$

  $\boldsymbol{D}_0$是一个对角矩阵，其对角线上的值是向量$\boldsymbol{x}^{(0)}$中的元素:

$$\boldsymbol{D}_0=\mathrm{diag}[x_1^{(0)},\cdots,x_n^{(0)}]=\begin{bmatrix} x_1^{(0)} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & x_n^{(0)} \end{bmatrix}$$

  $\boldsymbol{x}^{(0)}$是严格内点，$\boldsymbol{D}_0$是可逆的。

- 对于一般形式的$\boldsymbol{A}$和**b**，也可以采用这种仿射变换进行处理，但可能无法精确地变换到可行集的中心。

仿射尺度法

经过仿射尺度变换后的初始点位于(或接近)可行集的中心

对初始点$\boldsymbol{x}^{(0)}$左乘矩阵$\boldsymbol{D}_0^{-1}$,坐标系也随之改变。因此，应该将原来的线性规划问题变换到新的坐标系上，变换坐标系后问题为:

$$
\begin{aligned}
\text{minimize} & \ \bar{c}_0^{\top}\overline{\boldsymbol{x}} \\
\text{subject to} & \ \overline{\boldsymbol{A}}_0\overline{\boldsymbol{x}} = \boldsymbol{b} \\
& \ \overline{\boldsymbol{x}} \geqslant \mathbf{0}
\end{aligned}
$$

其中$\overline{\boldsymbol{c}}_0 = \boldsymbol{D}_0\boldsymbol{c}, \overline{\boldsymbol{A}}_0 = \boldsymbol{A}\boldsymbol{D}_0$

在新的坐标系($\overline{\boldsymbol{x}}$)下，构造正交投影算子:

$$
\overline{\boldsymbol{P}}_0 = \boldsymbol{I}_n - \overline{\boldsymbol{A}}_0^{\top}\left(\overline{\boldsymbol{A}}_0\overline{\boldsymbol{A}}_0^{\top}\right)^{-1}\overline{\boldsymbol{A}}_0
$$

仿射尺度法

令$\overline{\boldsymbol{d}}^{(0)}$为$-\bar{\boldsymbol{c}}_0$在矩阵$\overline{\boldsymbol{A}}_0$零空间上的正交投影，即

$$\overline{\boldsymbol{d}}^{(0)} = -\overline{\boldsymbol{P}}_0\bar{\boldsymbol{c}}_0$$

$\bar{\boldsymbol{x}}^{(1)}$的迭代公式为

$$\bar{\boldsymbol{x}}^{(1)} = \bar{\boldsymbol{x}}^{(0)} - \alpha_0\overline{\boldsymbol{P}}_0\bar{\boldsymbol{c}}_0$$

$\bar{\boldsymbol{x}}^{(0)} = \boldsymbol{D}_0^{-1}\boldsymbol{x}^{(0)}$，利用变换$\boldsymbol{x}^{(1)} = \boldsymbol{D}_0\bar{\boldsymbol{x}}^{(1)}$可得到原坐标系下的$\boldsymbol{x}^{(1)}$。

$$\boldsymbol{x}^{(1)} = \boldsymbol{x}^{(0)} + \alpha_0\boldsymbol{d}^{(0)}$$

其中，

$$\boldsymbol{d}^{(0)} = -\boldsymbol{D}_0\overline{\boldsymbol{P}}\boldsymbol{D}_0\boldsymbol{c}$$

不断重复上述迭代过程，将会得到一个序列$\{\boldsymbol{x}^{(k)}\}$，满足

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \alpha_k\boldsymbol{d}^{(k)}$$

仿射尺度法

选择α_k的主要原则是要保证步长尽可能大，但又不能大到使得$\boldsymbol{x}^{(k)}$中出现非正数的元素，即必须满足$x_i^{(k+1)} = x_i^{(k)} + \alpha_k d_i^{(k)} > 0, i = 1,2,3,\ldots,n$ 。定义

$$r_k = \min_{\{i:d_i^{(k)}<0\}} -\frac{x_i^{(k)}}{d_i^{(k)}}$$

r_k表示使$\boldsymbol{x}^{(k+1)}$的所有元素都非负的步长α_k的最大值。为了保证$\boldsymbol{x}^{(k+1)}$是严格内点，令步长为$\alpha_k = \alpha r_k, \alpha \in (0,1)$。$\alpha$的常用值为0.9或0.99。

仿射尺度法在有限次迭代内无法得到最优解。需要增加终止条件，如

$\frac{|\boldsymbol{cx}^{(k+1)} - \boldsymbol{cx}^{(k)}|}{max\{1,|\boldsymbol{cx}^{(k)}|\}} < \varepsilon$ 时，迭代停止，$\varepsilon > 0$是预先设定的阈值。

Matlab实现仿射尺度法

```
function [x,N] = affscale(A,b,c,u)
% 仿射尺度法
% u:初始点
xnew=u;
n=length(c);
% 设置迭代次数
max_iter=8;
% 设置合适的步长
alpha = 0.99;
for k = 1:max_iter
    xcurr=xnew;
    D = diag(xcurr);
    Abar = A*D;
    % 构造正交投影算子
    Pbar = eye(n) - Abar'*inv(Abar*Abar')*Abar;
    d = -D*Pbar*D*c;
    % 确定合适的步长
if d ~= zeros(n,1)
        nonzd = find(d<0);
        r = min(-xcurr(nonzd)./d(nonzd));
    else
        disp('Terminating: d = 0');
        break;
    end
    % 迭代序列
    xnew = xcurr + alpha*r*d;
end
    % 判断输出条件
if nargout >= 1
    x=xnew;
    if nargout == 2
        N=k;
    end
end
```

实验示例

仿射尺度法：

```
A = [1 0 1 0 0; 0 1 0 1 0; 1 1 0 0 1];
% A为约束方程组系数矩阵
b = [4;6;8];
%b为约束方程组常数项
c = [-2;-5;0;0;0];
u = [2;3;2;3;3];
[x,N] = affscale(A,b,c,u);
```

```
x =

    2.0000
    6.0000
    2.0000
    0.0000
    0.0000


N =

     8
```

linprog函数介绍

1.函数简介

在matlab中，linprog函数可以求解线性规划问题，用于寻找目标函数的最小值。matlab中，规划模型的标注写法如下

$$\min f \cdot x$$

$$s.t.\ A \cdot x \leq b$$

$$Aeq \cdot x = beq$$

$$lb \leq x \leq ub$$

f, x, b, beq, lb, ub是向量；A和Aeq是矩阵

linprog函数介绍

可以看出，linprog函数对应的线性规划模型与我们常见的线性规划规范模型略有区别，

规范类型：

$$\min f \cdot x$$

$$s.t.\ A \cdot x >= b$$

所以，在带入matlab中求解之前，把模型的约束条件转化为linprog函数对应的状态。

linprog函数介绍

2.语法

a. **x=linprog(f,A,b)**

用于求解

$$\min f \cdot x$$

$$\text{s.t. } A \cdot x <= b$$

b. **x=linprog(f,A,b,Aeq,beq)**

用于求解

$$\min f \cdot x$$

$$\text{s.t. } A \cdot x <= b$$

$$Aeq \cdot x = beq$$

linprog函数介绍

c. **x=linprog(f,A,b,Aeq,beq,lb,ub)**

用于求解

$$\min f \cdot x$$

$$\text{s.t. } A \cdot x \le b$$

$$Aeq \cdot x = beq$$

$$lb \le x \le ub$$

可以约束决策变量的范围在[lb,ub]内

d. **[x,fval]=linprog(f,A,b,Aeq,beq,lb,ub)**

用法和c一致，不同的是，这种写法会返回目标函数的值fval

实验示例

数学规划模型：

把模型的约束条件转化为linprog函数对应的状态！

$$
\begin{aligned}
\max\ & 2x_1 + x_2 \\
\text{s.t.}\ & 5x_2 \le 15 \\
& 6x_1 + 2x_2 \le 24 \\
& x_1 + x_2 \le 5 \\
& x_1 \ge 0, x_2 \ge 0
\end{aligned}
$$

$$
\begin{aligned}
& \min -2x_1 - x_2 \\
& s.t. 5x_2 \le 15 \\
& 6x_1 + 2x_2 \le 24 \\
& x_1 + x_2 \le 5 \\
& x_1 \ge 0, x_2 \ge 0,
\end{aligned}
$$

Linprog计算实验示例

根据目标函数和约束条件，可以得出目标函数系数矩阵f=[-2; -1]，不等式约束系数矩阵A =[0 5;6 2;1 1]，不等式约束常向量b=[15; 24; 5]，lb=zeros(2,1),如下所示:

```
f=[-2;-1];
A=[0 5;6 2;1 1];
b=[15;24;5] ;
lb=zeros(2,1);
[x,fval,exitflag,output,lambda]=linprog(f,A,b,[],[],lb)
```

Linprog计算实验示例

我们可以看到求出的最优解x,目标函数最优值fval，其中exitflag =1代表求解的结果是成功的，如果是其他数字代表失败。我们也可以看一下优化过程中的各种输出信息output，结构体，包含最优解处的拉格朗日乘子lambda，如下图所示：

```
>> f=[-2;-1];
A=[0 5;6 2;1 1];
b=[15;24;5] ;
lb=zeros(2,1);
[x,fval,exitflag,output,lambda]=linprog(f,A,b,[],[],lb)
Optimization terminated.
x =
       7/2
       3/2
fval =
     -17/2
exitflag =
       1
output =
         iterations: 7
          algorithm: 'interior-point-legacy'
       cgiterations: 0
            message: 'Optimization terminated.'
     constrviolation: 0
       firstorderopt: *
lambda =
    ineqlin: [3x1 double]
      eqlin: [0x1 double]
      upper: [2x1 double]
      lower: [2x1 double]
```