```
[xmin,fmin]=fmincon(f,x0,[],[],A,b)

error_cond=norm(A*xmin-b,2) % 计算是否符合约束条件，结果偏差很小
```

输出结果在最后一并展示并比较。

4.1.2 ADMM

我们的目标是求解

$$\min_x ||Bx||_1, \quad s.t. Ax = b$$

基于 ADMM 算法 [1]，将目标改写成

$$\min_x f(x) + g(z) = I_C(x) + ||z||_1, \quad s.t. Bx - z = 0 \tag{1}$$

$$C = \{x|Ax = b\}, \tag{2}$$

$$f(x) = I_C(x) = \begin{cases} 0, & x \in C \\ \infty, & x \notin C \end{cases} \tag{3}$$

迭代算法如下，其中 a^k 表示 a 的第 k 次迭代的值，ρ 为不满足条件 $Ax = b$ 的惩罚参数，即 ρ 越小则越不满足条件 $Ax = b$。

$$x^{k+1} = \arg\min_x f(x) + (\rho/2)||Bx - (z^k - u^k)||_2^2 \tag{4}$$

$$= argminLC(B, z^k - u^k, A, b) \tag{5}$$

$$z^{k+1} = \arg\min_z ||z||_1 + (rho/2)||z - (Bx^{k+1} + u^k)||_2^2 \tag{6}$$

$$= S_{1-\rho}(Bx^{k+1} - z^{k+1}) \tag{7}$$

$$u^{k+1} = u^k + Bx^{k+1} - z^{k+1} \tag{8}$$

其中，$S_k(a)$ 为向量函数，其中每个分量的计算为

$$S_k(a_i) = \begin{cases} a_i - k, & a > k \\ a_i + k, & a < -k \\ 0, & |a| \leq k \end{cases}$$

在 MATLAB 中可以很方便地实现，代码如下

Listing 12: Function S

```
function res=S(k,a)
agk=(a>k).*(a-k);% a>k term
alnk=(a<-k).*(a+k);% a<-k term
aink=(abs(a)≤k).*0;% |a|≤k term
res=agk+alnk+aink;
end
```

而函数 $argminLC$，是一个带线性约束 (Linearity Constraint) 的最优化问题，根据文献 [2]，对于这样一个满足线性约束的最优化问题

$$min||Ax-b|| \quad s.t. Cx=d$$

可以通过下式求解使其获得最优化的 x：

$$\begin{pmatrix} A^TA & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} \hat{x} \\ z \end{pmatrix} = \begin{pmatrix} A^Tb \\ d \end{pmatrix}$$

用 MATLAB 即可实现，代码如下：

Listing 13: Function argminLC

```
function xmin=argminLC(A,b,C,d)
[p,n]=size(C);
xz=[A'*A,C';C,zeros(p)]\[A'*b;d];
xmin=xz(1:n);
end
```

迭代算法已经介绍完，接下来明确迭代终止条件。

可以调参的是绝对误差 (absolute tolerance)ϵ^{abs} 和相对误差 (relative tolerance)ϵ^{rel}。

终止条件依赖于第 k 次迭代时的对偶残差 $s^k=-\rho B^T(z^k-z^{k-1})$ 和原始残差 $r^k=Bx^k-z^k$，而终止的 criteria 为

$$\epsilon^{dual} = \sqrt{n}\epsilon^{abs} + \epsilon^{rel}||B^Ty^k||_2 = \sqrt{n}\epsilon 6abs + \epsilon^{rel}||B^T\rho u^k||_2 \tag{9}$$

$$\epsilon^{pri} = \sqrt{p}\epsilon^{abs} + \epsilon^{rel}\max\{||Bx^k||_2, ||z^k||_2, 0\} \tag{10}$$

$$[p,n] = size(B) \tag{11}$$

当 $s^k \le \epsilon^{dual}$ 且 $r^k \le \epsilon^{pri}$ 时，终止迭代。

基于 ADMM 的算法在 MATLAB 中实现代码如下,其中函数 argminLC 和 S 的实现代码已在上文给出：

Listing 14: Function ImplementIt

```
function xmin=ImplementIt(B,A,b)
%ImplementIt Algorithem completed! Implement it!
% 推荐参数列表
% rho=2,errrel=1e-8,errabs=1e-8,uold=rand(p,1) 这组效果和fmincon差不多,
% 不过好像也没有比fmincon更好的参数，fmincon nb!

[p,n]=size(B);% substitute B as A is the same
rho=2;% rho avail 可调参数
errrel=1e-8;% errrel avail
errabs=1e-8;% errabs avail
x=A\b;zold=B*x;
uold=rand(p,1);% uold avail
iter=0;iterN=100000;
while iter<iterN % 人为控制迭代次数上限
    iter=iter+1;
    x=argminLC(B,zold-uold,A,b);
    znew=S(1./rho,B*x+uold);
    unew=uold+B*x-znew;
    snew=-rho.*B'*(znew-zold);
    rnew=B*x-znew;
    errdual=sqrt(n).*errabs+errrel.*norm(B'.*rho*unew,2);
    errpri=sqrt(p).*errabs+errrel.*max(norm(B*x,2),norm(znew,2));
    if norm(snew)<=errdual && norm(rnew)<=errpri % 终止条件
        break;
    end
    zold=znew;uold=unew;
    if iter==iterN % debug
        disp([norm(snew),errdual , norm(rnew),errpri]);
    end
end
fprintf('iter=%6.1f\n',iter);
xmin=x;
end
```

调用函数 ImplementIt 解决问题：

Listing 15: Call

```
disp('ImplementIt result')
xmin2=ImplementIt(B,A,b)
fmin2=f(xmin2)
error_cond2=norm(A*xmin2-b,2) % 计算是否符合约束条件
```

4.2 调用函数解题及其结果分析

完成 Q4 的全代码如下：

Listing 16: Q4

```
clear;clc
format compact
rng default
A = rand(5,8); B = rand(5,8); b = rand(5,1);

disp('fmincon result')
% fminbnd
f=@(x) norm(B*x,1);
x0=A\b;
tic;
[xmin,fmin]=fmincon(f,x0,[],[],A,b)
toc;
error_cond=norm(A*xmin-b,2) % 计算是否符合约束条件，结果偏差很小

disp(' ')

disp('ImplementIt result')
tic;
xmin2=ImplementIt(B,A,b)
fmin2=f(xmin2)
toc;
error_cond2=norm(A*xmin2-b,2) % 计算是否符合约束条件

disp(' ')

disp('comparison')% 可将rng default 改为 rng shuffle，结果相同
minc=abs(fmin-fmin2)
err_cond_comp=abs(error_cond-error_cond2)
```

输出结果为

Listing 17: outcomes of Q4

```
fmincon result

Local minimum possible. Constraints satisfied.

fmincon stopped because the size of the current step is less than
the value of the step size tolerance and constraints are
```

```
satisfied to within the value of the constraint tolerance.

<stopping criteria details>
xmin =
     1.3924
    -1.1354
     0.8642
    -0.7923
     1.8978
     0.7100
    -1.5551
    -1.6763
fmin =
     1.1473
时间已过 0.253831 秒。
error_cond =
    3.8459e-16

ImplementIt result
iter= 303.0
xmin2 =
     1.3924
    -1.1354
     0.8642
    -0.7923
     1.8978
     0.7100
    -1.5551
    -1.6763
fmin2 =
     1.1473
时间已过 0.036987 秒。
error_cond2 =
    5.1179e-16

comparison
minc =
    2.0993e-07
err_cond_comp =
    1.2719e-16
>>
```

结果分析：在给定参数下，并可以经过多次实验（即不设置 rng default），结果和 fmincon 差距很小，且所用时间比 fmincon 少一些。

参考文献

[1] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[2] L. Vandenberghe. Constrained least squares. Website, 2020. http://www.seas.ucla.edu/~vandenbe/133A/lectures/cls.pdf.