DUAL SIMPLEX

c = [ -2 0 -1 0 0 0];

a = [-1 -1 1 1 0; -1 2 -4 0 1];

b = [ -5; -8];

A = [a b]

bv = [4 5]

zjcj = c(bv)\*A - c

simplex\_table = [A; zjcj]

array2table(simplex\_table, 'VariableNames',{'x1','x2','x3','s1','s2','sol'})

RUN = true;

while RUN

sol = A(:,end);

if any(sol<0)

[leaving\_var, pvt\_row] = min(sol);

zrow = A(pvt\_row, 1: end-1);

zc= zjcj(1:end-1);

for i = 1:size(A,2)-1

if(zrow(i)<0)

ratio(i) = abs(zc(i)/zrow(i));

else

ratio(i) = inf;

end

end

[enter\_var, pvt\_col] = min(ratio);

pvt\_key = A(pvt\_row, pvt\_col);

bv(pvt\_row)= pvt\_col;

A(pvt\_row,:)= A(pvt\_row,:)/pvt\_key;

for i = 1:size(A,1)

if i ~=pvt\_row

A(i,:)= A(i,:)-A(i,pvt\_col)\*A(pvt\_row,:);

end

end

zjcj= c(bv)\*A - c;

new\_table = [A; zjcj];

array2table(new\_table, 'VariableNames',{'x1','x2','x3','s1','s2','sol'})

else

RUN = false;

end

end

fprintf('the feasible solution is %f\n', zjcj(end))

LEAST COUNT

c=[6 4 1 5;8 9 2 7;4 3 6 4]

a=[14;16;5]

b=[6 10 15 4]

m=size(c,1);n=size(c,2)

z=0

if sum(a)==sum(b)

fprintf('TP is balanced')

else

fprintf('TP is unbalanced')

if sum(a)<sum(b)

c(end+1,:)=zeros(1,length(b))

a(end+1)=sum(b)-sum(a)

else

c(:,end+1)=zeros(length(a),1)

b(end+1)=sum(a)-sum(b)

end

end

X=zeros(m,n)

Initialc=c

for i=1:size(c,1)

for j=1:size(c,2)

cpq=min(c(:))

if cpq==Inf

break

end

[p1 q1]=find(cpq==c)

xpq=min(a(p1),b(q1))

[X1 ind]=max(xpq)

p=p1(ind);q=q1(ind)

X(p,q)=min(a(p),b(q))

if min(a(p),b(q))==a(p)

b(q)=b(q)-a(p)

a(p)=a(p)-X(p,q)

c(p,:)=Inf

else

a(p)=a(p)-b(q)

b(q)=b(q)-X(p,q)

c(:,q)=Inf

end

end

end

for i=1:size(c,1)

for j=1:size(c,2)

z=z+Initialc(i,j)\*X(i,j)

end

end

BIG M

clc;

clear;

%make table

a=[1 1 1 0 0 0;5 2 0 1 0 0;2 8 0 0 -1 1];

b=[2;10;12];

A=[a b];

M=1000

zfun=[5 3 0 0 0 -M 0];

bv=[3 4 6];

AV=[6];

zjcj=zfun(bv)\*A - zfun;

var={'x1','x2','s1','s2','s3','A1','sol'};

tab=[A;zjcj];

array2table(tab,'VariableNames',var)

while(any(zjcj(1:end-1)<0))

if any(zjcj(1:end-1)<0)

fprintf('No optimal solution');

zc=zjcj(1:end-1);

[entryVal,pvt\_col]=min(zc);

if all(A(:,pvt\_col)<=0)

error('unbounded solution')

else

col=A(:,pvt\_col);

sol=A(:,end);

for i=1:size(A,1)

if col(i)>0

ratio(i)=sol(i)/col(i);

else

ratio(i)=inf;

end

end

[min\_ratio,pvt\_row]=min(ratio);

pvt\_key=A(pvt\_row,pvt\_col);

A(pvt\_row,:)=A(pvt\_row,:)/pvt\_key;

for i=1:size(A,1)

if i~=pvt\_row

A(i,:)=A(i,:)-A(i,pvt\_col)\*A(pvt\_row,:);

end

end

bv(pvt\_row)=pvt\_col;

zjcj=zfun(bv)\*A-zfun;

newTab=[A;zjcj];

array2table(newTab,'VariableNames',var)

end

end

end

RUN=false

if any (bv==AV(1))

error('infeasible solution')

end

fprintf('Optimal Solution=%f',zjcj(end))

SIMPLEX

clc;

clear;

%make table

a=[1 2 1 0 0;1 1 0 1 0;1 -1 0 0 1];

b=[10;6;2];

A=[a b];

zfun=[2 1 0 0 0 0];

bv=[3 4 5];

zjcj=zfun(bv)\*A - zfun;

var={'x1','x2','s1','s2','s3','sol'};

tab=[A;zjcj];

array2table(tab,'VariableNames',var)

while(any(zjcj(1:end-1)<0))

if any(zjcj(1:end-1)<0)

fprintf('No optimal solution');

zc=zjcj(1:end-1);

[entryVal,pvt\_col]=min(zc);

if all(A(:,pvt\_col)<=0)

error('unbounded solution')

else

col=A(:,pvt\_col);

sol=A(:,end);

for i=1:size(A,1)

if col(i)>0

ratio(i)=sol(i)/col(i);

else

ratio(i)=inf;

end

end

[min\_ratio,pvt\_row]=min(ratio);

pvt\_key=A(pvt\_row,pvt\_col);

A(pvt\_row,:)=A(pvt\_row,:)/pvt\_key;

for i=1:size(A,1)

if i~=pvt\_row

A(i,:)=A(i,:)-A(i,pvt\_col)\*A(pvt\_row,:);

end

end

bv(pvt\_row)=pvt\_col;

zjcj=zfun(bv)\*A-zfun;

newTab=[A;zjcj];

array2table(newTab,'VariableNames',var)

end

end

end

fprintf('Optimal Solution=%f',zjcj(end))

TWO PHASE

clc

clear

a=[2 1;1 3;0 1];

b=[2;2;4];

c=[0 0 0 -1 0 0 0];

Id=[-1 1 0 0; 0 0 1 0; 0 0 0 1];

A=[a Id b];

n=3;

m=size(A,2)

run=1;

bv=[4 5 6];

while run==1

basic\_cost=c(bv);

zj=basic\_cost\*A;

zjcj=zj-c;

zc=zjcj(1:end-1);

[mz,ec]=min(zc);

pc=A(:,ec);

b=A(:,end);

ratio=b./pc;

for i=1:size(A,1)

if ratio(i)<0

ratio(i)=inf;

end

end

[leave,lindex]=min(ratio);

bv(lindex)=ec;

pivot\_row=lindex;

pivot\_column=ec;

pv=A(pivot\_row,pivot\_column);

pivot\_row=A(pivot\_row,:)/pv;

for i=1:size(A,1)

if i~=lindex

A(i,:)=A(i,:)-(pivot\_row)\*A(i,ec);

else

A(i,:)=pivot\_row;

end

end

basic\_cost=c(bv);

zj=basic\_cost\*A;

zjcj=zj-c;

zc=zjcj(1:end-1)

if(zc>=0)

run=0;

else

run=1;

end

end

A