Contents


```
clear all;
close all;
clc;
warning('off')
```

This code is to compute the primal game LP of player 2

Initialization

```
load P.mat; %transition matrix of player 1
load Q.mat; %transition matrix of player 2
load M.mat; %payoff matrix
G=M;
T=2; %Number of stages in a game
A=2; %Number of player 1's actions
B=2; %Number of player 2's actions
k=3; %Number of states of player 1
l=2; %Number of states of player 2
lm=0.3; %discounted valuee
p=[0.5 0.3 0.2]; %player 1's initial probability for state
q=[0.5 0.5];     %player 2's initial probability for state

%information set of player 1 and 2
[is1,n_is1]=info_I(T,A,B,k);
[is2,n_is2]=info_J(T,A,B,l);
```

Contraint S_{J_t}(b_t)=Q_{a_{t-1},b_{t-1}}(I_{t-1},I_t)S_{J_{t-1}}(b_t-1);

```
%\forall t=1,...n, \forall \mathcal{J}_t
%Creating the equation as matrix multiplication [Aeq]*[variable]=[beq]. In
%this equation the variable is S_{J_t}

Aeq=zeros(sum(n_is2),sum(n_is2)*B);
beq=zeros(sum(n_is2),1);
```

```
row_index=0;
for t=1:T
    for j=1:length(is2{t})
        row_index=row_index+1;
        for b=1:B  %As there is summation b_t in S_{J_{t}}
            J=is2{t}(j,:); %for selecting each row from the information set of player 2
            [col_index_SJt] = SJt_col_index_P2(t,J,A,B,l,b,n_is2);
            Aeq(row_index,col_index_SJt)=1;
        end
        lt=J(end);
        if t==1
           beq(row_index,1)=q(lt); %As at t=1, Q_{a_0,b_0}(l_0,l_1)=1 and S_{J_0}(b_0)=Pr(
l1)
        else
            Jpre=is2{t}(j,1:(length(is2{t}(j,:))-3));
            aprev=is2{t}(j,(length(is2{t}(j,:))-2));
            bprev=is2{t}(j,(length(is2{t}(j,:))-1));
            [col_index_SJtprev] = SJt_col_index_P2(t-1,Jpre,A,B,l,bprev,n_is2);
            ltpre=J(end-3);
            Aeq(row_index,col_index_SJtprev)=-Q{aprev,bprev}(ltpre,lt);
        end
    end
end
```

for inequality contraint

```
%Creating the equation as matrix multiplication [Ain]*[variables]=[bin]. In
%this equation the variables are S_{J_t},(Z_{I}_{t}},Z_{\mathcal{I}_{t+1}})
[lset,n_lset] = Lset(T,l); %to get player 2's state information only

Ain=zeros(sum(n_is1)*A,(sum(n_is2)*B+sum(n_is1)));
bin=zeros(sum(n_is1)*A,1);

row_index=0;
for t=1:T
    for i=1: length (is1{t}) %for different I_t and different a_t there will be different
row
        for a=1:A
            row_index=row_index+1;
            for ln=1:length(lset{t})
                for b=1:B
                    [J] = construct_Is(i,ln,t,lset,is1);
                    [col_index_SJt] = SJt_col_index_P2(t,J,A,B,l,b,n_is2);
                    Ain(row_index,col_index_SJt)=lm^(t-1)*G{(is1{t}(i,(3*t-2))),J(3*t-2)}(
a,b); %S_{J_t} coefficient
                end
            end
            if t<T %As for Z_{I_{N+1}}=0
                for b1=1:B
                    for kplus=1:k
                        Iplus=[is1{t}(i,:) a b1 kplus];
                        [col_index_Isplus] = I_col_index(n_is1,A,B,k,Iplus,n_is2,t+1);
                         Ain(row_index,col_index_Isplus)=P{a,b1}(is1{t}(i,end),kplus); %Th
e coefficient for Z_{I_{t+1}}
                    end
                end
```

```
            end
            %To find the term for Z_{I_{t}}
            Ipre=is1{t}(i,:);
            [col_index_Ipre] = I_col_index(n_is1,A,B,k,Ipre,n_is2,t);
             Ain(row_index,col_index_Ipre)=-1;
        end
    end
end
```

Construct the objective function

```
f=[zeros(1,sum(n_is2)*B) p zeros(1,sum(n_is1(2:end)))]; %zeros(1,sum(n_is2)*B) is for S_{J_t} as there is no S_{J_t} in the objective function. zeros(1,sum(n_is1(2:end))) is for Z_{I_{2:T}} as in objective function there is only Z_{I_1}

%Rearrange every coefficient to use in linprog
f1=[zeros(1,size(Ain,2)-size(f,2)) f ]; % To ensure that matrix of objective function and Ain,Aeq are of same size we add zeroes
Aeq1=padarray(Aeq,[0, (size(Ain,2)-size(Aeq,2))],0,'post'); % to use linprog Ain and Aeq must have same column number as column number indicates variables. Both equation must have same variables
lb=[zeros(sum(n_is2)*B,1);-Inf((size(Ain,1)-sum(n_is2)*B),1)];
ub=+Inf;
options = optimoptions('linprog','Display','none');
[y,v2]=linprog(f1,Ain,bin,Aeq1,beq,lb,ub,options); %y=[S_{J_t} Z_{I_t}]
```

Game value

```
v2
```

```
v2 =

  103.6310
```

Finding the optimal strategy from the realization plan we got from previous linprog result

```
%tau=SJT/(Q_{a_{t-1},b_{t-1}}(l_{t-1},l_t)*SJ_{t-1})

tau=zeros(1,sum(n_is2));
tau_col=0;

for t=1:T
    for j=1:length(is2{t})
        tau_col=tau_col+1;
        J=is2{t}(j,:); %for selecting each row from the information set of player 2
        for b=1:B
            [row_index_SJt] = SJt_col_index_P2(t,J,A,B,l,b,n_is2);
            SJt=y(row_index_SJt,1);
        lt=J(end);
        if t==1
```

```
            SJtprev=q(lt);
            qvalue=1;
        else
            Jpre=is2{t}(j,1:(length(is2{t}(j,:))-3));
            aprev=is2{t}(j,(length(is2{t}(j,:))-2));
            bprev=is2{t}(j,(length(is2{t}(j,:))-1));
            [row_index_SJtprev] = SJt_col_index_P2(t-1,Jpre,A,B,l,bprev,n_is2);
            ltpre=J(end-3);
            SJtprev=y(row_index_SJtprev,1);
            qvalue=Q{aprev,bprev}(ltpre,lt);
        end
          tau(b,tau_col)=SJt/(SJtprev*qvalue);
       end
    end
end
```

Player 2's optimal strategy

```
tau %tau(1st row) for b=1 and tau(2nd row) for b=2. Each column indicates each information
 set at different stages. For example. 4th column is for T=2 (k_1=1, a_1=1, b_1=1, k_2=2)
```

```
tau =

  Columns 1 through 7

         0    1.0000       NaN       NaN         0    0.5610       NaN
    1.0000         0       NaN       NaN    1.0000    0.4390       NaN

  Columns 8 through 14

       NaN    0.3142    1.0000    0.0678    1.0000       NaN       NaN
       NaN    0.6858         0    0.9322         0       NaN       NaN

  Columns 15 through 18

    0.0678    1.0000       NaN       NaN
    0.9322         0       NaN       NaN
```

to find mu

```
%to find ZI1
for k_present=1:k
    t=1;
    info=k_present;
    [col_index_ZI1] = I_col_index(n_is1,A,B,k,info,n_is2,t);
    ZI1=y(col_index_ZI1,1);
    mu(k_present)=[ZI1];
end
```

mu is the initial vector payoff over player 1's state

```
mu=-mu %as mu=-U_{J_1}. mu[1] is for mu(k=1), mu[2] is for mu(k=2), mu[3] is for mu(k=3)
```

```
mu =

 -146.5664  -80.5188  -30.9606
```

Published with MATLAB® R2019b