蔡沅恆 311551136

1. 程式介紹

Fictious function 介紹:

此次作業 Q1 至 Q9 各題的 code 是由呼叫一個叫作 fictious 的函數來完成,並且 print 到螢幕上。其有三個輸入:

1. matrix: payoff matrix
2. the belief of player1 to player2
3. the belief of player2 to player1

```
void fictious(vector<vector<vector<double>>> &matrix,vector<double> &belief1,vector<double> &belief2,int Q){
```

首先是由 player1 選擇出他的 best respond。對其每個策略 i 計算在 belief1 to player2 下所到的 payoff ($\sum_{j=0}^{1} belief1[j] \times matrix[i][j][0]$),並且以 **MAX1** 來紀錄當下最大的 payoff,同時把對應的 strategy 記錄在 **bestrespond1** 中。另外當 payoff 與 **MAX1** 相等時用 rand()的方式來決定是否要新 best respond(兩者皆可以)。

```
double max1{-100};
int bestrespond1{-1};
for(int i{0};i<n;i++){
    double temp{0};
    for(int j{0};j<n;j++){
        temp=temp+belief1[j]*matrix[i][j][0];
    }
    if((temp>max1) or (temp==max1 and rand()%2)){
        max1=temp;
        bestrespond1=i;
    }
}
```

接著 Player2 也做相似的步驟來找出 best respond

```
double max2{-100};
int bestrespond2{-1};
for(int i{0};i<n;i++){
    double temp{0};
    for(int j{0};j<n;j++){
        temp=temp+belief2[j]*matrix[j][i][1];
    }
    if((temp>max2) or (temp==max2 and rand()%2)){
        max2=temp;
        bestrespond2=i;
    }
}
```

最後更新 belief1 與 belief2 把對應到的 strategy 加一

```
if(bestrespond2>-1) belief1[bestrespond2]++;
if(bestrespond1>-1) belief2[bestrespond1]++;
```

而此過程會執行 10000 次,並起每次紀錄。若最後結果趨近於一萬時則可以判定其結果接近 pure strategy,反之則為 mix strategy。(程式設定 initial 執行次數小於 2,因此只要是大於 9990 就可以說是趨近於 10000 以判斷是否為 mix strategy)

```
cout<<Q<<"--player1 ["<<initial11<<","<<initial12<<"] player2 ["<<initial21<<","<<initial22<<"]"<<endl;
cout<<"                strategy of player1:"<<"["<<belief2[0]<<"]["<<belief2[1]<<"]"<<endl;
cout<<"                strategy of player2:"<<"["<<belief1[0]<<"]["<<belief1[1]<<"]";
if((belief1[0]<9900.0 and belief1[1]<9900.0) or (belief2[0]<9900.0 and belief2[1]<9900.0)) cout<<"~~~~~~~~~~~~~~~~~~~~mixed strategy";
cout<<endl;
cout<<endl;
```

Initial state 的設定:

為了能夠找出以些特定情況下才會發生的 pure strategy,Q1 至 Q9 每個問題會執行 1*10*10 次。分別是在不同的”已經走幾步(i)”與各 player 在不同 i 下已經走過各 strategy 之不同組合。

```
cout<< Q1 ---------------------------------------------------------------------------- <<endl;
   for(double i{0};i<t;i++){
        vector<vector<vector<double>>> matrix2
        {
            {{-1,-1},{1,0}},
        {{0,1},{3,3}}
        };
        for(double m{0};m<i;m=m+0.1){
            for(double k{0};k<i;k=k+0.1){
                vector<double> belief21{i-m,m};
                if(i-m<0.001) belief21[0]=0;
                vector<double> belief22{i-k,k};
                if(i-k<0.001) belief22[0]=0;

                fictious(matrix2,belief21,belief22,1);
            }
        }
   }
```

2.問題回答

Q1.

yes,考以找到 pure strategy,例如:

當初始 belief2 為[0.9,0.1]與 belief1 為[1,0]時,則找到 pure strategy(右下)

```
1--player1 [0.9,0.1] player2 [1,0]
                strategy of player1:[0.9][10000.1]
                strategy of player2:[1][10000]

1--player1 [0.8,0.2] player2 [1,0]
                strategy of player1:[0.8][10000.2]
                strategy of player2:[1][10000]

1--player1 [0.7,0.3] player2 [1,0]
                strategy of player1:[0.7][10000.3]
                strategy of player2:[1][10000]

1--player1 [0.6,0.4] player2 [1,0]
                strategy of player1:[0.6][10000.4]
                strategy of player2:[1][10000]
```

Q2:

yes,考以找到兩個 pure strategy,例如:

當初始 belief2 為[0,1]與 belief1 為[0.3,0.7]時,則找到 pure strategy(右下)

當初始 belief2 為[1,0]與 belief1 為[0.2,0.8]時,則找到 pure strategy(左上)

補充:其也可以找到 mixed strategy((1/2,1/2),(1/2,1/2)),在初始 belief2 為[0.9,0.1]與 belief1 為[0.2,0.8]時

```
2--player1 [0,1] player2 [0.3,0.7]
                strategy of player1:[0][10001]
                strategy of player2:[0.3][10000.7]

2--player1 [1,0] player2 [0.2,0.8]
                strategy of player1:[10000][1]
                strategy of player2:[10000.2][0.8]

2--player1 [0.9,0.1] player2 [0.2,0.8]
                strategy of player1:[5000.9][5000.1]
                strategy of player2:[5000.2][5000.8]~~~~~~~~~~~~~~~~~~~~mixed strategy
```

Q3:

No, 他只能到達其中一個 NE(1,1)。原因是當其在右下(0,0)時還是能考慮到(1,1)此組策略,且只有此組策略非 0,因此度館怎樣最終會走向(1,1)此策略。而傳統方式則只會考慮到左下與右上兩組(0,0)策略,因此不會離開右下

```
3--player1 [0.2,0.8] player2 [0.4,0.6]
                strategy of player1:[10000.2][0.8]
                strategy of player2:[10000.4][0.6]

3--player1 [0.1,0.9] player2 [0.4,0.6]
                strategy of player1:[10000.1][0.9]
                strategy of player2:[10000.4][0.6]

3--player1 [0,1] player2 [0.4,0.6]
                strategy of player1:[10000][1]
                strategy of player2:[10000.4][0.6]

3--player1 [1,0] player2 [0.3,0.7]
                strategy of player1:[10001][0]
                strategy of player2:[10000.3][0.7]
```

Q4:

Yes,其會到達題目所給之 mixed strategy

Player1-[8000 左右,2000 左右],約是 4/5 與 1/5

Player2-[5000 左右,5000 左右],約是 1/2 與 1/2

```
4--player1 [0.7,0.3] player2 [0,1]
                strategy of player1:[8011.7][1989.3]
                strategy of player2:[4881][5120]~~~~~~~~~~~~~~~~~~~~mixed strategy

4--player1 [0.6,0.4] player2 [0,1]
                strategy of player1:[8023.6][1977.4]
                strategy of player2:[4916][5085]~~~~~~~~~~~~~~~~~~~~mixed strategy

4--player1 [0.5,0.5] player2 [0,1]
                strategy of player1:[8044.5][1956.5]
                strategy of player2:[4978][5023]~~~~~~~~~~~~~~~~~~~~mixed strategy

4--player1 [0.4,0.6] player2 [0,1]
                strategy of player1:[7946.4][2054.6]
                strategy of player2:[4990][5011]~~~~~~~~~~~~~~~~~~~~mixed strategy
```

Q5:

Yes,可以找到 mixed strategy

Player1-[5000 左右,5000 左右],約是 1/2 與 1/2

Player2-[5000 左右,5000 左右],約是 1/2 與 1/2

```
5--player1 [0.3,0.7] player2 [1,0]
                strategy of player1:[4966.3][5034.7]
                strategy of player2:[4963][5038]~~~~~~~~~~~~~~~~~~~~mixed strategy

5--player1 [0.2,0.8] player2 [1,0]
                strategy of player1:[4930.2][5070.8]
                strategy of player2:[5005][4996]~~~~~~~~~~~~~~~~~~~~mixed strategy

5--player1 [0.1,0.9] player2 [1,0]
                strategy of player1:[5026.1][4974.9]
                strategy of player2:[4956][5045]~~~~~~~~~~~~~~~~~~~~mixed strategy

5--player1 [0,1] player2 [1,0]
                strategy of player1:[5020][4981]
                strategy of player2:[4947][5054]~~~~~~~~~~~~~~~~~~~~mixed strategy
```

Q6:

分別可以找到 pure strategy 與 mixed strategy。例如:

當初始 belief2 為[0.6,0.4]與 belief1 為[0.9,0.1]時,則找到 pure strategy

當初始 belief2 為[0.4,0.6]與 belief1 為[0.9,0.1]時,則找到 mixed strategy(在(1/2,1/2),(1/2,1/2))

```
6--player1 [0.6,0.4] player2 [0.9,0.1]
                strategy of player1:[10000.6][0.4]
                strategy of player2:[10000.9][0.1]

6--player1 [0.5,0.5] player2 [0.9,0.1]
                strategy of player1:[10000.5][0.5]
                strategy of player2:[10000.9][0.1]

6--player1 [0.4,0.6] player2 [0.9,0.1]
                strategy of player1:[5000.4][5000.6]
                strategy of player2:[5000.9][5000.1]~~~~~~~~~~~~~~~~~~~~mixed strategy

6--player1 [0.3,0.7] player2 [0.9,0.1]
                strategy of player1:[5000.3][5000.7]
                strategy of player2:[5000.9][5000.1]~~~~~~~~~~~~~~~~~~~~mixed strategy
```

Q7:

Yes,分別可以找到兩個 pure strategy 與 mixed strategy。例如:

當初始 belief2 為[0,1]與 belief1 為[1,0]時,則找到 pure strategy(左下)

當初始 belief2 為[1,0]與 belief1 為[0.9,0.1]時,則找到 pure strategy(右上)

當初始 belief2 為[0.9,0.1]與 belief1 為[0.9,0.1]時,則找到 mixed strategy(在(1/2,1/2),(1/2,1/2))

```
7--player1 [0,1] player2 [1,0]
                strategy of player1:[0][10001]
                strategy of player2:[10001][0]

7--player1 [1,0] player2 [0.9,0.1]
                strategy of player1:[10000][1]
                strategy of player2:[0.9][10000.1]

7--player1 [0.9,0.1] player2 [0.9,0.1]
                strategy of player1:[5000.9][5000.1]
                strategy of player2:[5000.9][5000.1]~~~~~~~~~~~~~~~~~~~~mixed strategy
```

Q8:

Yes,分別可以找到兩個 pure strategy 與 mixed strategy。例如:

當初始 belief2 為[0.2,0.8]與 belief1 為[0.9,0.1]時,則找到 pure strategy(左上)

當初始 belief2 為[0,1]與 belief1 為[0.9,0.1]時,則找到 pure strategy(右下)

當初始 belief2 為[0.1,0.9]與 belief1 為[0.9,0.1]時,則找到 mixed strategy(在(3/5,2/5),(2/5,3/5))

```
8--player1 [0.2,0.8] player2 [0.9,0.1]
                strategy of player1:[10000.2][0.8]
                strategy of player2:[9999.9][1.1]

8--player1 [0.1,0.9] player2 [0.9,0.1]
                strategy of player1:[6000.1][4000.9]
                strategy of player2:[4000.9][6000.1]~~~~~~~~~~~~~~~~~~~~mixed strategy

8--player1 [0,1] player2 [0.9,0.1]
                strategy of player1:[3][9998]
                strategy of player2:[1.9][9999.1]
```

Q9:

Yes,分別可以找到兩個 pure strategy 與 mixed strategy。例如:

當初始 belief2 為[1,0]與 belief1 為[0.8,0.2]時,則找到 pure strategy(左上)

當初始 belief2 為[0,1]與 belief1 為[0.9,0.1]時,則找到 pure strategy(右下)

當初始 belief2 為[0.1,0.9]與 belief1 為[0.9,0.1]時,則找到 mixed strategy(在(1/2,1/2),(1/2,1/2))

```
9--player1 [0.1,0.9] player2 [0.9,0.1]
                strategy of player1:[5000.1][5000.9]
                strategy of player2:[5000.9][5000.1]~~~~~~~~~~~~~~~~~~~~mixed strategy

9--player1 [0,1] player2 [0.9,0.1]
                strategy of player1:[1][10000]
                strategy of player2:[0.9][10000.1]

9--player1 [1,0] player2 [0.8,0.2]
                strategy of player1:[10001][0]
                strategy of player2:[10000.8][0.2]
```

Q10:

不適合,就像 Q3 的結果,其會少一組 NE。