SIT718 Real World Analytics

Lecturer: Dr Ye Zhu

School of Information Technology
Deakin University

Week 10: Game Theory 2

Game Theory

For this week we have the following learning aims:

- Understand mix-strategies and Nash equilibrium
- To be able to solve mix-strategies with R programming

Recommended Textbooks

1. *Operations Research: Applications and Algorithms* by **Wayne L. Winston**
2. *Introduction to Operations Research* by **Frederick Hillier**
3. *Game Theory: An Introduction, 2nd Edition* by **E. N. Barron**

Mixed Strategies–Definition

- A **mixed strategy** for Player I is a vector $x = (x_1, \ldots, x_m)$ with $x_i \geq 0$, for all i, representing the probability (i.e., portion of time) that Strategy i is used; and

$$x_1 + x_2 + \cdots x_m = 1,$$

- E.g., Player I has 3 strategies, and

$$(x_1, x_2, x_3) = (0.5, 0.3, 0.2)$$

  means that he will spend 50 % of the time on Strategy 1, 30% of the time on Strategy 2, and 20% of the time on Strategy 3.

Mixed Strategies–Definition

- Similarly, a mixed strategy for Player II is a vector $y = (y_1, \ldots, y_n)$ with $y_j \geq 0$, for all j, representing the probability that Strategy j is used; and

$$y_1 + y_2 + \cdots y_n = 1.$$

- E.g., Player II has 4 strategies, and

$$(y_1, y_2, y_3, y_4) = (0.25, 0.25, 0.4, 0.1)$$

means that she will spend 25 % of the time on Strategy 1, 25% of the time on Strategy 2, 40% of the time on Strategy 3, and 10% of the time on Strategy 4.

Mixed Strategies–Definition

- A **pure strategy** is a vector $x = (x_1, \ldots, x_m)$, with one component 1 and all other components 0. i.e., $x = (0, \ldots, 0, 1, 0, \ldots, 0)$.
- So if a person uses a pure strategy they use the same strategy all the time (100% of the time).
  (N.B. This is what we do when there is a saddle point).

Mixed Strategies–Expected Payoffs

$$\begin{array}{cc} & \\ & \\ A_1 & x_1 \\ A_2 & x_2 \end{array} \begin{array}{c} \begin{array}{cc} B_1 & B_2 \\ y_1 & y_2 \end{array} \\ \begin{bmatrix} 1 & 5 \\ 6 & 2 \end{bmatrix} \end{array}$$

If Player I uses Strategy A_1 3/4 of the time and uses Strategy A_2 1/4 of the time,

- her expected payoff if Player II uses Strategy B_1 is given by:

$$x_1(1) + x_2(6) = 0.75(1) + 0.25(6) = 2.25$$

- her expected payoff if Player II uses Strategy B_2 is given by:

$$x_1(5) + x_2(2) = 0.75(5) + 0.25(2) = 4.25$$

Mixed Strategies–Expected Payoffs

$$\begin{array}{cc} & \\ & \\ A_1 & x_1 \\ A_2 & x_2 \end{array} \begin{array}{c} \begin{array}{cc} B_1 & B_2 \\ y_1 & y_2 \end{array} \\ \begin{bmatrix} 1 & 5 \\ 6 & 2 \end{bmatrix} \end{array}$$

Now, if Player II uses B_1 half of the time and uses B_2 the rest of the time, what is the overall expected payoff for Player I?

$$0.5(2.25) + 0.5(4.25) = 3.25$$

MIXED STRATEGIES–EXPECTED PAYOFFS

$$\begin{array}{cc} & \\ & \\ A_1 & x_1 \\ A_2 & x_2 \end{array} \begin{array}{c} \begin{array}{cc} B_1 & B_2 \\ y_1 & y_2 \end{array} \\ \begin{bmatrix} 1 & 5 \\ 6 & 2 \end{bmatrix} \end{array}$$

If Player II uses Strategy B_1 half of the time and uses Strategy B_2 the rest of the time,

- his expected payoff if Player I uses Strategy A_1 is given by:

$$y_1(1) + y_2(5) = 0.5(1) + 0.5(5) = 3$$

- his expected payoff if Player I uses Strategy A_2 is given by:

$$y_1(6) + y_2(2) = 0.5(6) + 0.5(2) = 4$$

Mixed Strategies–Expected Payoffs

$$\begin{array}{ccc} & & \begin{array}{cc} B_1 & B_2 \\ y_1 & y_2 \end{array} \\ \begin{array}{c} A_1 \\ A_2 \end{array} & \begin{array}{c} x_1 \\ x_2 \end{array} & \begin{bmatrix} 1 & 5 \\ 6 & 2 \end{bmatrix} \end{array}$$

Now, if Player I uses Strategy A_1 3/4 of the time and uses Strategy A_2 1/4 of the time, the expected payoff for Player II will be:

$$0.75(3) + 0.25(4) = 3.25$$

Now the question is, **what is the best combo**? (In other words, what is the best proportion for each of the players?)

Mixed Strategies–Using Linear Programming

Let v is the value of the game.

- Player I's game

$$
\begin{aligned}
\max\ z = v & \\
\text{s.t.}\quad & v - (a_{11}x_1 + a_{21}x_2 + \cdots + a_{m1}x_m) && \leq 0 \\
& v - (a_{12}x_1 + a_{22}x_2 + \cdots + a_{m2}x_m) && \leq 0 \\
& \vdots && \vdots \\
& v - (a_{1n}x_1 + a_{2n}x_2 + \cdots + a_{mn}x_m) && \leq 0 \\
& x_1 + x_2 + \cdots + x_m && = 1 \\
& x_i \geq 0,\ \forall i = 1, \ldots, m. \\
& v \text{ unrestricted in sign}
\end{aligned}
$$

Mixed Strategies–Using Linear Programming

- Player II's game

$$
\begin{aligned}
\min\ & w = v \\
\text{s.t.}\quad & v - (a_{11}y_1 + a_{12}y_2 + \cdots + a_{1n}y_n) \geq 0 \\
& v - (a_{21}y_1 + a_{22}y_2 + \cdots + a_{2n}y_n) \geq 0 \\
& \vdots \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
& v - (a_{m1}y_1 + a_{m2}y_2 + \cdots + a_{mn}y_n) \geq 0 \\
& y_1 + y_2 + \cdots + y_n = 1 \\
& y_i \geq 0,\ \forall j = 1, \ldots, n. \\
& v \text{ u.r.s.}
\end{aligned}
$$

Here, u.r.s means that v can be positive, negative or 0.

Example 6

Let the payoff matrix of a two-person zero-sum game be:

$$V = \begin{bmatrix} -2 & 1 & -3 \\ -1 & -1 & 2 \\ 3 & 0 & -1 \end{bmatrix}$$

As $L = -1$ and $U = 1$, the value of the game (v) is somewhere between $[-1, 1]$.

Mixed Strategies–Using Linear Programming

$$V = \begin{bmatrix} -2 & 1 & -3 \\ -1 & -1 & 2 \\ 3 & 0 & -1 \end{bmatrix}$$

Player I's game:

- Suppose player 1 chooses the mixed strategy (x_1, x_2, x_3).
  - If Player 2 choose strategy 1, the expected payoff is $(-2x_1 - x_2 + 3x_3)$
  - If Player 2 choose strategy 2, the expected payoff is $(x_1 - x_2)$
  - If Player 2 choose strategy 3, the expected payoff is $(-3x_1 + 2x_2 - x_3)$
- Player 2 wants to minimize player 1's payoff.
  - The player 2 will choose a strategy that makes the player 1's reward equal to $min(-2x_1 - x_2 + 3x_3,\ x_1 - x_2,\ -3x_1 + 2x_2 - x_3)$.
- Then the player 1 should choose (x_1, x_2, x_3) to make $min(-2x_1 - x_2 + 3x_3, x_1 - x_2, -3x_1 + 2x_2 - x_3)$ as *large as possible.*

Mixed Strategies–Using Linear Programming

In mathematics, we can write the following LP optimization to find ***minimum of three numbers , say*** a, b, c**.**

$$\begin{aligned} &\max\ v \\ &\text{s.t. } v \le a \implies v - a \le 0 \\ &\qquad v \le b \implies v - b \le 0 \\ &\qquad v \le c \implies v - c \le 0 \end{aligned}$$

$$V = \begin{bmatrix} -2 & 1 & -3 \\ -1 & -1 & 2 \\ 3 & 0 & -1 \end{bmatrix}$$

Hence, the **Player 1's game** can be written as follows:

$$\begin{aligned} &\max\ \ z = v \\ &\text{s.t. } v - (-2x_1 - x_2 + 3x_3) \le 0 \\ &\qquad v - (x_1 - x_2) \le 0 \\ &\qquad v - (-3x_1 + 2x_2 - x_3) \le 0 \\ &\qquad x_1 + x_2 + x_3 = 1 \\ &\qquad x_i \ge 0,\ \forall i = 1, 2, 3. \\ &\qquad v \text{ u.r.s. (means - unrestricted sign)} \end{aligned}$$

- Note the two additional constraints there in the above LP:
  - $x_1 + x_2 + x_3 = 1$
  - $x_i \ge 0,\ \forall i = 1, 2, 3.$

Mixed Strategies–Using Linear Programming

$$V = \begin{bmatrix} -2 & 1 & -3 \\ -1 & -1 & 2 \\ 3 & 0 & -1 \end{bmatrix}$$

- Player I's game

$$\begin{aligned} \max\ z &= v \\ \text{s.t. } v - (-2x_1 - x_2 + 3x_3) &\le 0 \\ v - (x_1 - x_2) &\le 0 \\ v - (-3x_1 + 2x_2 - x_3) &\le 0 \\ x_1 + x_2 + x_3 &= 1 \\ x_i \ge 0,\ \forall i &= 1, 2, 3. \\ v \text{ u.r.s.} & \end{aligned}$$

```
library(lpSolveAPI)
lprec <- make.lp(0, 4)
lp.control(lprec, sense= "maximize")
set.objfn(lprec, c(0, 0, 0, 1))
add.constraint(lprec, c(2, 1, -3, 1), "<=", 0)
add.constraint(lprec, c(-1, 1, 0, 1), "<=", 0)
add.constraint(lprec, c(3, -2, 1, 1), "<=", 0)
add.constraint(lprec, c(1,1,1,0), "=", 1)
set.bounds(lprec, lower = c(0, 0, 0, -Inf))
RowNames <- c("Row1", "Row2", "Row3","Row4")
ColNames <- c("x1", "x2", "x3", "v")
dimnames(lprec) <- list(RowNames, ColNames)
solve(lprec)
get.objective(lprec)
get.variables(lprec)
get.constraints(lprec)
```

Mixed Strategies–Using Linear Programming

$$V = \begin{bmatrix} -2 & 1 & -3 \\ -1 & -1 & 2 \\ 3 & 0 & -1 \end{bmatrix}$$

Player II's game:

- Suppose player 2 chooses the mixed strategy (y_1, y_2, y_3).
  - If Player 1 choose strategy 1, the expected reward is $(-2y_1 + y_2 - 3y_3)$
  - If Player 1 choose strategy 2, the expected reward is $(-y_1 - y_2 + 2y_3)$
  - If Player 1 choose strategy 3, the expected reward is $(3y_1 - y_3)$
- The player 1 will choose a strategy to ensure that she obtains an expected reward of $max(-2y_1 + y_2 - 3y_3, -y_1 - y_2 + 2y_3, 3y_1 - y_3)$.
- Then the player 2 should choose (y_1, y_2, y_3) to make $max(-2y_1 + y_2 - 3y_3, -y_1 - y_2 + 2y_3, 3y_1 - y_3)$ as *small as possible*.

Mixed Strategies–Using Linear Programming

In mathematics, we can write the following LP optimization to find ***Maximum of three numbers , say*** a, b, c**.**

$$V = \begin{bmatrix} -2 & 1 & -3 \\ -1 & -1 & 2 \\ 3 & 0 & -1 \end{bmatrix}$$

$$\begin{aligned} &\min\ v \\ &\text{s.t. } v \geq a \Longrightarrow v - a \geq 0 \\ &\qquad v \geq b \Longrightarrow v - b \geq 0 \\ &\qquad v \geq c \Longrightarrow v - c \geq 0 \end{aligned}$$

Hence, the **Player 2's game** can be written as follows:

$$\begin{aligned} &\min\ \ w = v \\ &\text{s.t. } v - (-2y_1 + y_2 - 3y_3) \geq 0 \\ &\qquad v - (-y_1 - y_2 + 2y_3) \geq 0 \\ &\qquad v - (3y_1 - y_3) \geq 0 \\ &\qquad y_1 + y_2 + y_3 = 1 \\ &\qquad y_i \geq 0,\ \forall i = 1, 2, 3. \\ &\qquad v \text{ u.r.s. (means - unrestricted sign)} \end{aligned}$$

- Note the two additional constraints there in the above LP:
  - $y_1 + y_2 + y_3 = 1$
  - $y_i \geq 0,\ \forall i = 1, 2, 3.$

Mixed Strategies–Using Linear Programming

$$V = \begin{bmatrix} -2 & 1 & -3 \\ -1 & -1 & 2 \\ 3 & 0 & -1 \end{bmatrix}$$

- Player II's game

$$\begin{aligned} &\min\ w = v \\ &\text{s.t. } v - (-2y_1 + y_2 - 3y_3) \geq 0 \\ &\qquad v - (-y_1 - y_2 + 2y_3) \geq 0 \\ &\qquad v - (3y_1 - y_3) \geq 0 \\ &\qquad y_1 + y_2 + y_3 = 1 \\ &\qquad y_i \geq 0,\ \forall i = 1, 2, 3. \\ &\qquad v \text{ u.r.s.} \end{aligned}$$

```
lprec <- make.lp(0, 4)
lp.control(lprec, sense= "minimize")
set.objfn(lprec, c(0, 0, 0, 1))
add.constraint(lprec, c(2, -1, 3, 1), ">=", 0)
add.constraint(lprec, c(1, 1, -2, 1), ">=", 0)
add.constraint(lprec, c(-3, 0, 1, 1), ">=", 0)
add.constraint(lprec, c(1,1,1,0), "=", 1)
set.bounds(lprec, lower = c(0, 0, 0, -Inf))
RowNames <- c("Row1", "Row2", "Row3","Row4")
ColNames <- c("y1", "y2", "y3", "v")
dimnames(lprec) <- list(RowNames, ColNames)
solve(lprec)
get.objective(lprec)
get.variables(lprec)
get.constraints(lprec)
```

Example 6

$$v = -\frac{2}{11}$$

$$(x_1^*, x_2^*, x_3^*) = (\frac{3}{11}, \frac{5}{11}, \frac{3}{11})$$

$$(y_1^*, y_2^*, y_3^*) = (\frac{1}{33}, \frac{23}{33}, \frac{9}{33})$$

Classic non-zero sum games – Prisoner's Dilemma

Two prisoners awaiting trail are being kept in separate cells where they are not allowed to communicate. The prosecutor makes the same offer to both of them:

We have enough evidence to convict you both on a lesser charge. If you both plead innocent, you will be convicted and each will receive a 2-year sentence. However, if you help us, we will reward you: Admit guilt, then it'll be easier to convict your friend if he pleads innocent. He will get 5 years and we will let you go. If, however, you both plead guilty, you will both get 4 years.

Both prisoners are told the other has heard this offer.

Classic non-zero sum games – Prisoner's Dilemma

This two-person game can be represented as matrix of ordered pairs, the first component of the pair represents Player I's payoff, and the second component, Player II's payoff.

Table: The prisoner's dilemma payoff matrix

		Prisoner II	
		Don't confess	Confess
Prisoner I	Don't confess	(-2,-2)	(-5,0)
	Confess	(0,-5)	(-4,-4)

If the prisoners can communicate, and "co-operate", they can both plead innocent and receive a 2-year sentence. Otherwise they may both receive a 4-year sentence, or one receive 5 years whilst the other goes free.

2.8 – Nash equilibrium

A strategy profile is a Nash equilibrium if no player can do better by unilaterally changing his or her strategy. Each strategy in a Nash equilibrium is a best response to all other strategies in that equilibrium.

Nash equilibrium

Table: The prisoner's dilemma payoff matrix

		Prisoner II	
		Don't confess	Confess
Prisoner I	Don't confess	(-2,-2)	(-5,0)
	Confess	(0,-5)	(-4,-4)

(-4,-4) is a Nash equilibrium point, because if either prisoner changes his strategy, then his payoff decreases (from -4 to -5). However, (-2,-2) is not an equilibrium point, because if we are currently at (-2,-2), either prisoner can increase his payoff (from -2 to 0) by changing his strategy from "Don't confess" to "Confess" . Thus, (-2,-2) is not a stable solution if both playter double-cross each other without coopertation.

Nash equilibrium

Consider the following non-zero sum game.

	A_1	A_2
a_1	(8,0)	(3,3)
a_2	(3,3)	(0,8)

We see that (3,3) (a_1 vs A_2) is an equilibrium point. This is because either player will decrease his/her payoff by changing his/her strategy (from 3 to 0).

Solving non Zero-sum Two-player Games with Equations

Two-player games where each player has exactly two strategies:

Beth

Ann		Left	Right
	Up	1, 3	3, 2
	Down	4, 1	2, 4

Ann mixes if she plays a mixed strategy (x_1, x_2) with $x_2 = 1 - x_1$, and Beth mixes if she plays (y_1, y_2) with $y_2 = 1 - y_1$.

Nash equilibrium

		Beth	
		Left	Right
Ann	Up	1, 3	3, 2
	Down	4, 1	2, 4

Ann has the same expected payoffs when playing up or down provided Beth keeps mixing, i.e., no incentive to change strategy.

Ann plays up　　Ann plays down

Ann's payoff　$y_1 + 3(1 - y_1) = 4y_1 + 2(1 - y_1)$

Similarly, in a Nash equilibrium Beth has the same payoffs with each of her options when Ann mixes.

Beth plays left　　Beth plays right

Beth's payoff　$3x_1 + (1 - x_1) = 2x_1 + 4(1 - x_1)$

$x_1 = 0.75$ and $y_1 = 0.25$. $\frac{3}{4}$ up and $\frac{1}{4}$ down versus $\frac{1}{4}$ left and $\frac{3}{4}$ right.

Mixed Strategies–Using R Programming

- Player I's payoff

  $$\max z = v$$
  $$\text{s.t.} \quad v - (y_1 + 3y_2) = 0$$
  $$v - (4y_1 + 2y_2) = 0$$
  $$y_1 + y_2 = 1$$
  $$y_i \geq 0, \forall i = 1, 2.$$
  $$v \text{ u.r.s.}$$

- Player II's payoff

  $$\max z = v$$
  $$\text{s.t.} \quad v - (3x_1 + x_2) = 0$$
  $$v - (2x_1 + 4x_2) = 0$$
  $$x_1 + x_2 = 1$$
  $$x_i \geq 0, \forall i = 1, 2.$$
  $$v \text{ u.r.s.}$$

```
# Player I's payoff #

library(lpSolveAPI)
lprec <- make.lp(0, 3)
lp.control(lprec, sense= "maximize")
set.objfn(lprec, c(0, 0, 1))
add.constraint(lprec, c(-1, -3, 1), "=", 0)
add.constraint(lprec, c(-4, -2, 1), "=", 0)
add.constraint(lprec, c(1,1,0), "=", 1)
set.bounds(lprec, lower = c(0, 0, -Inf))
RowNames <- c("Row1", "Row2", "Row3")
ColNames <- c("y1", "y2", "v")
dimnames(lprec) <- list(RowNames, ColNames)
solve(lprec)
get.objective(lprec)
get.variables(lprec)

# Player II's payoff #

library(lpSolveAPI)
lprec <- make.lp(0, 3)
lp.control(lprec, sense= "maximize")
set.objfn(lprec, c(0, 0, 1))
add.constraint(lprec, c(-3, -1, 1), "=", 0)
add.constraint(lprec, c(-2, -4, 1), "=", 0)
add.constraint(lprec, c(1,1,0), "=", 1)
set.bounds(lprec, lower = c(0, 0, -Inf))
RowNames <- c("Row1", "Row2", "Row3")
ColNames <- c("x1", "x2", "v")
dimnames(lprec) <- list(RowNames, ColNames)
solve(lprec)
get.objective(lprec)
get.variables(lprec)
```

Notes on Nash's Theorem

- Minimax Theorem by establishing the existence of a solution for both zero-sum and nonzero-sum games.
- Saddle points in a zero-sum game were equivalent and interchangeable.
- Solution theory for zero-sum games does not carry over to non zero-sum games
- In mixed-strategy Nash equilibrium that each player plays opponent's of the game. **They neglect their own payoffs**!
- More than one solution may exist and may not be Pareto optimal.
- Nash equilibrium essentially describe probabilities that rational player can assign to opponent; not what they should do but what they should believe.