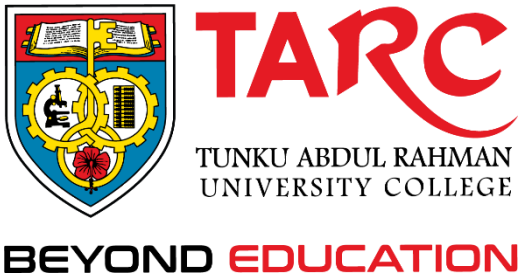
Game theory in baseball

By

Lee Cheng Zhan



FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

TUNKU ABDUL RAHMAN UNIVERSITY COLLEGE
KUALA LUMPUR

ACADEMIC YEAR
2021/2022

Game theory in baseball

By

Lee Cheng Zhan

Supervisor: Dr. Tey Siew Kian

A project report submitted to the
Faculty of Computing and Information Technology
in partial fulfilment of the requirement for the
Bachelor of Science (Hons.)
Management Mathematics with Computing
Tunku Abdul Rahman University College

Department of Mathematical and Data Science
Faculty of Computing and Information Technology
Tunku Abdul Rahman University College
Kuala Lumpur

2021/2022

DECLARATION

The project report submitted herewith is a result of my own efforts in totality and in every aspect of the works. All information that has been obtained from other sources had been fully acknowledged. I understand that any plagiarism, cheating or collusion of any sorts constitutes a breach of Tunku Abdul Rahman University College rules and regulations and would be subjected to disciplinary actions.

Signature : 

Name : Lee Cheng Zhan

ID No. : 19WMR05929

Date : 07/11/2021

Specially dedicated to

my beloved grandfather, grandmother, mother, father and my brother

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"GAME THEORY IN BASEBALL"** was prepared by **LEE CHENG ZHAN** and has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Hons.) Management Mathematics with Computing at Tunku Abdul Rahman University College.

Approved by,

Signature: ____ _____

Supervisor: _Dr. Tey Siew Kian_____

Date: _15/12/2021_____

Signature: ____ _____

Moderator: _Ms. Tan Li Yin_____

Date: _15/12/2021_____

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my supervisor, Dr. Tey Siew Kian for his invaluable advice, guidance and his enormous patience throughout the development of the research. Beside that, Dr. Tey Siew Kian also give a constant support and encouragement throughout the completion of this project

I would also like to personally thank my family, friends in Tunku Abdul Rahman University College and my fellow course-mates who have one way or another extended their assistance in completing this project.

Lastly, I would also like to express my gratitude to my loving parent and friends who had helped and given me encouragement.

ABSTRACT

The Objectives we focusing in this project is to decide the best strategy for each pitcher and hitter. The objective of this project is help to the team and player achieve better pitch and bat results by finding the best strategy. In this project, we will be modelling the situation by two-player games with finite games and the strategies for batter is swing and no-swing and the strategies for pitcher is pitch a goodball or badball.

In this project the payoff of each player is not same and we classify the player into weak and strong and consider the strong player can have high payoff when meet a weak player. After determine each player's payoff, we simulate the game in python by using the loop function. The purpose of the simulation is let each player can decide their strategy by guessing the opponent strategy.

The results of this project may not be the same as those in real life due to the payoff matrix of each player are hard to decide.

Keywords: Game Theory, Baseball, Prisoner's Dilemma, Mix Strategy, Zero-Sum Game

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

Chapter 1

INTRODUCTION

1.1 Background

1.1.1 What is baseball

Baseball is a bat-and-pitch game played between two opposite teams who take turns batting and pitching. The game proceeds when a player on the fielding team, called the pitcher, throws a ball and a player on the batting team tries to hit the ball. The purpose of the offensive team (batting team) is to hit the ball into the field of play to allowing its team players to run the bases, having them advance counter-clockwise around four bases to score what are called "runs". The purpose of the defensive team (fielding team) is to prevent the batter’s becoming runners, and to prevent runners advance around the bases. The batter has 3 chance to hit the ball when pitcher throw a ball into the strike zone (the space above home plate and between the batter's knees and the midpoint of their torso) also say as a goodball. But if the pitcher throws the ball 3 times outside the strike zone and batters can automatically go to the next base when them do not swing the ball. A run is scored when a runner legality advances around the four bases in order and touches the home plate (the place where the player started as a batter). The team that scores the most runs is the winner in the end of the games.

1.1.2 What is game theory

Game Theory objectives to help us understand the situations in which decision-makers interact. Like other sciences, game theory consists of a collection of models. A model is an abstraction

that we use to understand our observations and experiences in real life, “understanding” may not clear-cut. At least, it entails our perceiving relationships between situations and create a

new principle that applies to a series of problems, so that we can incorporate the new situations we encounter into our thinking. Game theory has been widely recognized as an important tool in many fields. As of 2014, 11 game theorists have been awarded the Nobel Prize in Economics with the awarding of the Nobel Memorial Prize in Economics to game theorist Jean Tirol. John Maynard Smith was awarded the Crawford Prize for his application of evolutionary game theory.

1.1.3 The theory of rational choice

Rational choice theory is an integral part of many models in game theory. In short, this theory holds that the decision maker chooses the best action among all available actions according to her preferences. There are no qualitative restrictions on the decision maker's preferences; her "rationality" lies in the consistency of her decisions when faced with different sets of available actions, not in the nature of her likes and dislikes.

1.1.4 The type of game theory

The type of game theory is classified as three types, which is single, two-player and multiplayer game. The essence of the single player game is the optimization problem of the individual. Two-player games are the most common, studied, and the most basic and useful type of the game. The prisoner's dilemma, coin guessing, and Tian Ji horse race are all two-player games. There are many possibilities for two players to play the game, and the interests of the game parties may or may not be in the same direction and the multiplayer games may be have spoilers: strategic choices that do not affect their own interests, but have a significant, sometimes decisive, impact on the interests of other parties to the game.

1.1.5 Limitations of Game Theory

The biggest problem with game theory is that, like most other economic models, it depends on the assumption that people are rational actors, self-interested, and utility maximizes. Of course, we are social beings and we do cooperate and do care about the welfare of others, often at our own expense. Game theory cannot explain the fact that in some cases we may get into a Nash equilibrium and other times not, depending on the social context and who the players are.

1.2 Objectives

While during this project, I start by understanding the rule and background of the baseball by social media and article. In the “understanding sabermetric”, I learn about the important of the data and their abbreviation e.g., OBP (on-base percentage), BA (batting average), SLG (slugging average) and how to calculate them. Base on it , I make a model to decide the ability of the batter. Beside that, in order to understand game theory, I have read some thesis and book that relevant to game theory like “an introduction to game theory” or “essentials of game theory” to understand what is game theory and their type and limitation. The Prisoner’s Dilemma is the method that help me to decide the best strategy. Since they are many different models in game theory learn the relevant knowledge in free learning sites like YouTube or coursera also a method that how I investigate to this project.

Sports games and tournaments have a lot of data, so it is a fruitful area to study and do betting. Baseball is a great game to analyse from a game theory perspective because the strategic and tactical decisions that are constantly being made on the field and between games are very complex and huge. In each case, there are countless participants (baseball players, coaches, team managers and owners) with different goals and payoff, making hundreds of pitch-by-pitch decisions over the course of an inning, a game and a season. Most baseball games are played in a one-on-one interaction. The pitcher throws the ball and the batter hits the ball. In that sense, in many cases it is fairly easy to judge the performance of batters and pitchers based on the outcome. Home runs, walks, strikeouts, all of these are based entirely on one-on-one matchups. All Major League Baseball (MLB) teams have their own departments dedicated to this type of analysis, and the time is ripe for analysis, as almost every aspect of the game is tracked with statistical information.

As time goes by, it has become common in sports to use mathematical methods to find the best strategy in a game. In this project, we try to use game theory to determine the best strategy in a baseball game. First, we will use the Nash equilibrium model to determine the best strategy for each pitcher and batter. In this statement, I will figure out what type of pitch the pitcher should throw, e.g., goodball or badball, and whether the batter should swing or not swing at the ball. I will try to figure this out through game theory. After that, I expect to use the data related to each player to get the payoff for each decision and make the model more accurate.

The Objectives we focusing in this project is to decide the best strategy for each pitcher and hitter. The objective of this project is help to the team and player achieve better pitch and bat results by finding the best strategy. In this project, we will be modelling the situation by two-player games with finite games and the strategies for batter is swing and no-swing and the strategies for pitcher is pitch a goodball or badball.

Chapter 2

LITERATURE REVIEW

A Nash equilibrium is a solution to a non-cooperative game involving two or more players. In a Nash equilibrium, each player is assumed to know the equilibrium strategies of the other players, and no player gains anything by simply changing his or her strategy. Essentially, it tells us what we believe rational actors will do in the game. In baseball, an example is batting (Micah Melling, 2018). A pure strategy Nash equilibrium is the fairest strategy such that each player's strategy is the best response to the other players' equilibrium strategies (leading to the highest available payoff).

The Prisoner's Dilemma is an example of a Nash equilibrium, which is a standard example of an analytic game in game theory that shows why two perfectly rational people may not cooperate, even if it seems to be in their best interests to do so. It was originally proposed by Merrill Flood and Melvin Dresher in 1950 while they were working at the RAND Corporation. Albert W. Tucker formalized this game with prison sentence rewards and named it the Prisoner's Dilemma, expressed as follows:
By examining the four possible pairs of actions in the Prisoner's Dilemma (reproduced in **Table 1**), we see that (Fink, Fink) is the only Nash equilibrium.

Player 1/Player 2	Quiet	Fink
Quiet	3 , 3	2 , 5
Fink	5 , 2	3 , 3

Table 1: Prisoner's Dilemma

The action pair (Fink, Fink) is a Nash equilibrium because (i) given that player 2 chooses Fink, player 1 is better off choosing Fink than Quiet (looking at the right column of the table, we see that Fink gives player 1 a payoff of 3, while Quiet gives her a payoff of 2). and (ii) given that Player 1 chooses Fink, Player 2 is better off choosing Fink than Quiet (looking at the bottom row of the table, we see that Fink gives Player 2 a payoff of 3, while Quiet gives her a payoff of 2).

According to (Hattie James, 2015), game theory uses mathematical models to analyse decisions. Most sports are zero-sum games in which a decision by one player (or team) will have a direct effect on the opposing player (or team). This creates an equilibrium known as the Nash equilibrium, named after mathematician John Forbes Nash. This means that if a team scores a run, it is usually at the expense of the opposing team - most likely based on an error by the outfielder or a hit by the pitcher. In the case of pitching, game theory - and in particular the use of Nash equilibria - can be used to predict the strategic purpose of pitching optimization. In this case, a mixed strategy is best - in game theory, when a player intends to keep his opponent guessing.

According to (Matt at el, 2012), it considers the structure of the following pair of games, where the batter is always better off when the action is fastball/swing or curveball/no swing, and the pitcher is always better off when the action is fastball/no swing and curveball/swing. The difference between these two examples is that a pitcher who has a good curveball has an even higher expected return on the ball/swing. In situation 1, the pitcher has a strike 50% of the time and the batter swings 50% of the time. In situation 2, the pitcher throws strike 56% of the time and the batter swings 44% of the time. A pitcher with a better out pitch will throw more fastballs to encourage the batter to actually swing at the fastball when they throw it. It also suggests that the batter observes a "signal" example after the pitch is thrown: the batter can consider the type of pitch that follows the pitcher's motion, and he uses reverse induction to solve this problem.

According to (Gelblum, Laucys, Saperstein, and Sodha, 2010), this article delves into the specific scenario that occurred in the 8th inning of the Dodgers vs. Giants game on April 18,

2010. The most interesting finding of the article talks about pitch selection and the expected balance of mixed strategies, and for simple reasons, does not include pitch location. Not surprisingly, the expected outcome of the batter's RAA (run average) was positive only when the batter correctly guessed the pitch being thrown, and the expected outcome of the pitcher was positive only when the batter incorrectly guessed the pitch about to be thrown. Based on the negative RAA payoff to the batter for guessing the wrong pitch, the batter is least negatively affected by not guessing the fastball than by not guessing the changeup or slider. Thus, mixed strategy Nash equilibrium occurred when batters Manny Ramirez guessed fastballs 6.25%, sliders 50.02%, and changeups 43.73% of the time, while pitchers threw fastballs 43.47%, sliders 34.90%, and changeups 21.63% of the time. Batters over-expect pitches that have worse returns for hitters. wRAA measures how many runs a hitter contributes, compared to an average player - so a player with a wRAA of 0 would be considered league average.

According to (Piper at el, 2012), Plate discipline statistics tell us how often a batter swings and makes contact on certain types of pitches, or how often a pitcher induces swings or contact on certain types of pitches. Some of the terminology and calculations for these data are shown below:

O-Swing% - Percentage of pitches a batter swings at outside the strike zone

Swings at pitches outside the zone / pitches outside the zone

Z-Swing% – Percentage of pitches a batter swings at inside the strike zone

Swings at pitches inside the zone / pitches inside the zone

Swing% – Percentage of pitches a batter swings at overall.

Swings / Pitches

This data is very useful in determining the type of batter or pitcher. Based on this data, we can more accurately determine the return of batters and pitchers in each game by understanding their pitching/swinging habits and frequency.

According to (Gabriel B.Costa, Micheal R. Huber and John T.Saccoman, 2009), Batting average (BA) is found by dividing the number of base hits (H) by the total number of at-bats (AB):

$$BA = \frac{H}{AB}$$

A batter's slugging percentage (SLG) is found by dividing the total number of bases (TB) of all base hits by the total number of times at bat:

$$SLG = \frac{TB}{AB}$$

A batter's on-base percentage (OBP) is found by dividing the total number of hits plus bases on balls (BB) plus hit by pitch (HBP) by at-bats plus bases on balls plus hit by pitch plus sacrifice flies (SF):

$$OBP = \frac{H + BB + HBP}{AB + BB + HBP + SF}$$

According to (Benjamin Baumer and Andrew Zimbalist, 2013), its shows about the correlation of the data with team scores. the corresponding correlations for OBP, BA and SLG are 0.885, 0.822 and 0.910 respectively. they show the autocorrelation of many commonly used statistics, as well as the correlation with team scores. However, they also tell us that the exact statistics (such as OPS and SLG) are not necessarily reliable in terms of the strength of their correlation with team scoring, as they are highly correlated with themselves in time. Thus, knowing a player's OPS can give us a good idea of how much he contributes to a team's offense in a given season, but it does not reveal what he might do the following season.

Chapter 3

METHODOLOGY

3.1 Data

In this project, we use official Major League Baseball (MLB) data from 2014 - 2018 to decide the pitcher and batter payoff.

3.2 Determine the best strategy

In most cases, pitch selection and hitter responses will be solved by two-player mixed strategies with finite games. That means a mixed strategy involved a player select two (or more but no unlimited) strategies with probabilities between 0 and 1, and the Nash Equilibrium requires the player to be indifferent between two (or more but no unlimited) strategies, conditional on his opponent's (potentially mixed) strategy.

Let's assume the pitcher is deciding whether to pitch or strike. If a player swings at a strike, he gets a hit and wins the game; if he swings at a pitch, he misses and loses the game. Therefore, the impact on each player's team record is reflected in the table below:

Pitcher\Batter	Swing	Take (No-swing)
Strike (Goodball)	$-2,2$	$2,-2$
Ball (Badball)	$2,-2$	$-2,2$

Table 2: the effect on each player's team's record

To solve this game, we consider the value a batter receives from swinging and not swinging, conditional on the pitcher's given strategy "p", i.e., the probability of throwing a strike. The value of a batter's swing will be the product of his payoff from Strike/Swing (2) multiplied by the probability of this outcome occurring when he swings (p), plus his payoff from Ball/Swing (-2) multiplied by the probability of this outcome occurring when he swings (1 - p).

Therefore, we define the batter's value from swinging as:

$$V(s) = (2)^*(P) + (-2)^*(1-p) = 4p - 2$$

The batter's value from taking is:

$$V(t) = (-2)^*(P) + (2)^*(1-P) = 2 - 4p$$

Conditional on the batter's strategy "q," his probability of swinging, the pitcher's value from throwing a strike is:

$$V(g) = (-2) * (q) + (2) * (1-q) = 2 - 4q$$

The pitcher's value of throwing a ball when the batter swings with probability "q" is:

$$V(b) = (2) * (q) + (-2) * (1-q) = 4q - 2$$

If 2p - 1 > 1 - 2p (i.e., when p (the probability of a pitcher throwing a strike) > 0.5), the batter will always tend to swing, so in this case his strategy will be to always swing (i.e., set q = 1). Similarly, when p < 0.5, the batter will always take, so his strategy will be to always take (i.e., set q = 0). However, when 1-2q>2q-1 (i.e., q<0.5), the pitcher will always tend to throw strikes, so in this case his strategy will be to always throw strikes (i.e., set p=1). Similarly, when q>0.5, the pitcher will always throw a pitch, so his strategy is to throw a pitch (i.e., set p=0).

The only Nash equilibrium that exists is when p=0.5 and q=0.5. In other words, the only possible outcome is for the pitcher to throw strikes in 50% of the full cases and the batter to swing at 50% of the cases. When this happens, the expected value for both the batter and the pitcher is equal to 0. However, if we can find the probabilities of p and q from the data, it means that we can decide the best strategy for the player.

However, this is just one of those situations where the payoff varies due to the circumstances, e.g. if a good hitter with a good swing meets a bad hitter, the hitter can see the type of ball by its movement or by the last pitch the hitter can swing at (if the hitter throws a pitch on three swings (bad pitch) he will be replaced).

3.3 Methodology / Model

3.3.1 Model for batter

Since everyone have different bat and pitch skill, the payoff can not be consistent and it is hard to determine the payoff for the strategy. So, we need to use the data to decide the payoff of each player. The equation of predicting the Strength of a batter can be:

$$B = B_0X_0 + B_1X_1 + B_2X_2$$

B = The ability of the batter
B_0 = Correlation between team run scored and OBP
X_0 = Batter's OBP
B_1 = Correlation between team run scored and BA
X_1 = Batter's BA
B_2 = Correlation between team run scored and SLG
X_2 = Batter's SLG

To decide the ability of the Batter is weak or strong we subsitute Batter's OBP, BA and SLG by average OBP, BA and SLG with 0.323, 0.252 and 0.435 respectively and subsitute the correlation with Benjamin Baumer and Andrew Zimbalist's data which OBP, BA and SLG are 0.885, 0.822 and 0.910 respectively :

Average B = 0.885*0.323+0.822*0.252+0.910*0.435 = 0.888849

So we can decided that if a batter ability below the average B which is 0.88849 can be consider weak and if above it can be consider strong. Now we use the Prisoners Dilemma to decide the strategy for batter that be consider strong, our payoff become :

Pitcher\Strong Batter	Swing	Take (No-swing)
Strike (Goodball)	$-3,3$	$2,-2$
Ball (Badball)	$2,-2$	$-2,2$

Table 3 : the effect on each player's team's record when strong batter is playing

we consider the value a batter receives from swinging and not swinging, conditional on the pitcher's given strategy "p", i.e., the probability of throwing a strike. The value of a batter's swing will be the product of his payoff from Strike/Swing (3) multiplied by the probability of this outcome occurring when he swings (p), plus his payoff from Ball/Swing (-2) multiplied by the probability of this outcome occurring when he swings (1 - p).

Therefore, we define the batter's value from swinging as:

$$V(s) = (3)^*(P) + (-2)^*(1-p) = 5p - 2$$

The batter's value from taking is:

$$V(t) = (-2)^*(P) + (2)^*(1-P) = 2 - 4p$$

Conditional on the batter's strategy "q," his probability of swinging, the pitcher's value from throwing a strike is:

$$V(g) = (-3) * (q) + (2) * (1-q) = 2 - 5q$$

The pitcher's value of throwing a ball when the batter swings with probability "q" is:

$$V(b) = (2) * (q) + (-2) * (1 - q) = 4q - 2$$

If 5p - 2 > 1 - 2p (i.e., when p (the probability of a pitcher throwing a strike) > 0.429), the batter will always tend to swing, so in this case his strategy will be to always swing (i.e., set q = 1). Similarly, when p < 0.429, the batter will always take, so his strategy will be to always take (i.e., set q = 0). However, when 2-5q>2q-1 (i.e., q<0.428), the pitcher will always tend to throw strikes, so in this case his strategy will be to always throw strikes (i.e., set p=1). Similarly, when q>0.428, the pitcher will always throw a pitch, so his strategy is to throw a pitch (i.e., set p=0).

The payoff for weak batter:

Pitcher\Weak Batter	Swing	Take (No-swing)
Strike (Goodball)	$-1,1$	$2,-2$
Ball (Badball)	$2,-2$	$-2,2$

Table 4 : the effect on each player's team's record when weak batter is playing

we consider the value a batter receives from swinging and not swinging, conditional on the pitcher's given strategy "p", i.e., the probability of throwing a strike. The value of a batter's swing will be the product of his payoff from Strike/Swing (1) multiplied by the probability of this outcome occurring when he swings (p), plus his payoff from Ball/Swing (-2) multiplied by the probability of this outcome occurring when he swings (1 - p).

Therefore, we define the batter's value from swinging as:

$$V(s) = (1)^*(P) + (-2)^*(1 - p) = 3p - 2$$

The batter's value from taking is:

$$V(t) = (-2)^*(P) + (2)^*(1 - P) = 2 - 4p$$

Conditional on the batter's strategy "q," his probability of swinging, the pitcher's value from throwing a strike is:

$$V(g) = (-1) * (q) + (2) * (1 - q) = 2 - 3q$$

The pitcher's value of throwing a ball when the batter swings with probability "q" is:

$$V(b) = (2) * (q) + (-2) * (1 - q) = 4q - 2$$

If 3p - 2 > 1 - 2p (i.e., when p (the probability of a pitcher throwing a strike) > 0.6), the batter will always tend to swing, so in this case his strategy will be to always swing (i.e., set q = 1). Similarly, when p < 0.6, the batter will always take, so his strategy will be to always take (i.e., set q = 0). However, when 2-3q>2q-1 (i.e., q<0.6), the pitcher will always tend to throw strikes, so in this case his strategy will be to always throw strikes (i.e., set p=1). Similarly, when q>0.6, the pitcher will always throw a pitch, so his strategy is to throw a pitch (i.e., set p=0).

3.3.2 Model for pitcher

Since everyone have different bat and pitch skill, the payoff can not be consistent and it is hard to determine the payoff for the strategy. So, we need to use the data to decide the payoff of each player. The equation of predicting the Strength of a pitcher can be:

$$P = B_0X_0 + B_1X_1 + B_2X_2$$

P = The ability of the pitcher
B_0 = Correlation between win game and WHIP
X_0 = Pticher's WHIP
B_1 = Correlation between team win game and ERA
X_1 = Pitcher's ERA
B_2 = Correlation between team run win game and OBA
X_2 = Pitcher's OBA

We chose WHIP, ERA and OBA as variables for this model because we found a high negative correlation between them and total winning games. So we can also imagine them has high positive correlation with total loses game. Therefore if this model have higher negative value then average P means that the pitcher has lower winrate and consider it is weak and have lower negative value then average P means that the pitcher has higher winrate and consider it is strong.

To decide the average P of the pitcher is weak or strong we subsitute pitcher's WHIP, ERA and OBA by the mean we found in *appendix 2.1 and figure 1* which WHIP, ERA and OBA with 1.308, 0.25266 and 4.076067 respectively and subsitute the correlation we found in *appendix 2.1 and figure 2* which WHIP, ERA and OBA are -0.755924, -0.743581 and -0.741970 respectively :

```
ERA     4.076067
OBA     0.252660
WHIP    1.308000
dtype: float64
```

Figure 1: Mean of ERA, OBA and WHIP

	GP	W
GP	1.000000	0.148107
W	0.148107	1.000000
L	-0.123670	-0.999645
ERA	-0.106815	-0.743581
SV	0.126572	0.618413
CG	-0.198420	0.190239
SHO	0.023655	0.478079
QS	0.136282	0.518454
ER	-0.093792	-0.739420
HR	-0.015619	-0.390623
BB	0.013301	-0.423417
OBA	-0.099907	-0.741970
WHIP	-0.074646	-0.755924

Figure 2: Correlation of win score between ERA, OBA and WHIP

Average P = -0.755924*1.308 -0.743581*0.25266 -0.741970*4.072067 = -4.191797

So we can decided that if a pitcher ability value smaller then the average P which is -4.191797 can be consider weak and if larger it can be consider strong. Now we use the Prisoners Dilemma to decide the strategy for pitcher that be consider strong, our payoff become :

Strong Pitcher\Batter	Swing	Take (No-swing)
Strike (Goodball)	$-1,1$	$2,-2$
Ball (Badball)	$2,-2$	$-2,2$

Table 5: the effect on each player's team's record when strong pitcher is playing

we consider the value a pitcher receives from strike and ball, conditional on the batter's given strategy "r", i.e., the probability of swing. The value of a pitcher strike will be the product of his payoff from Strike/Swing (-1) multiplied by the probability when batter swings (r), plus his payoff from strike/take (2) multiplied by the probability when batter take (1 - r).

Therefore, we define the pitcher's value from strike as:

$$V(g) = (-1)^*(r) + (2)^*(1-r) = 2 - 3\text{r}$$

The pitcher's value from ball is:

$$V(b) = (2)^*(r) + (-2)^*(1-\text{r}) = 4r - 2$$

Conditional on the pitcher's strategy "s," his probability of throwing a strike, the batter's value from swing is:

$$V(s) = (1) * (s) + (-2) * (1-s) = 3\text{s} - 2$$

The batter's value of take when the pitcher throws a strike with probability "s" is:

$$V(t) = (-2) * (s) + (2) * (1-s) = 2 - 4\text{s}$$

If 2-3r > 2r-1 (i.e., when r (the probability of batter swing) < 0.6), the pitcher will always tend to throw a strike, so in this case his strategy will be to always throw strikes (i.e., set s=1). Similarly, when p > 0.6, the pitcher will always tend to throw a ball (i.e., set s = 0). However, when 3s - 2 > 1 – 2s (i.e., when s (the probability of a pitcher throwing a strike) > 0.6), the batter will always tend to swing, so in this case his strategy will be to always swing (i.e., set r = 1).). Similarly, when s < 0.6, the batter will always take, so his strategy will be to always take (i.e., set r = 0).

The payoff for weak pitcher:

Weak Pitcher\Batter	Swing	Take (No-swing)
Strike (Goodball)	$-3,3$	$2,-2$
Ball (Badball)	$2,-2$	$-2,2$

Table 6: the effect on each player's team's record when weak pitcher is playing

we consider the value a pitcher receives from strike and ball, conditional on the batter's given strategy "r", i.e., the probability of swing. The value of a pitcher strike will be the product of his payoff from Strike/Swing (-3) multiplied by the probability when batter swings (r), plus his payoff from strike/take (3) multiplied by the probability when batter take (1 - r).

Therefore, we define the pitcher's value from strike as:

$$V(g) = (-3)^*(r) + (2)^*(1-r) = 2 - 5\mathrm{r}$$

The pitcher's value from ball is:

$$V(b) = (2)^*(r) + (-2)^*(1-r) = 4\mathrm{r} - 2$$

Conditional on the pitcher's strategy "s," his probability of throwing a strike, the batter's value from swing is:

$$V(s) = (3) * (s) + (-2) * (1-s) = 5\mathrm{s} - 2$$

The batter's value of take when the pitcher throws a strike with probability "s" is:

$$V(t) = (-2) * (s) + (2) * (1 - s) = 2 - 4s$$

If 2-5r > 2r-1 (i.e., r < 0.428), the pitcher will always tend to throw strikes, so in this case his strategy will be to always throw strikes (i.e., set s = 1). Similarly, when r > 0.428, the pitcher will always throw a ball, so his strategy is to throw a ball (i.e., set s = 0). However, when 5s - 2 > 1 – 2s (i.e., when p (the probability of a pitcher throwing a strike) > 0.429), the batter will always tend to swing, so in this case his strategy will be to always swing (i.e., set r = 1). Similarly, when s < 0.429, the batter will always take, so his strategy will be to always take (i.e., set r = 0).

Chapter 4

RESULTS

In this chapter, the results of each players strategy are been calculated by the program in *appendix 2.2 and 2.3*. Beside that, we simulate each interval 10000 times because we found that the graph will become smooth when the number of simulations become larger and it might become more accurate when number of simulations become larger. *Figure 3 and 4* shows the graph when the number of simulations is low (<100).

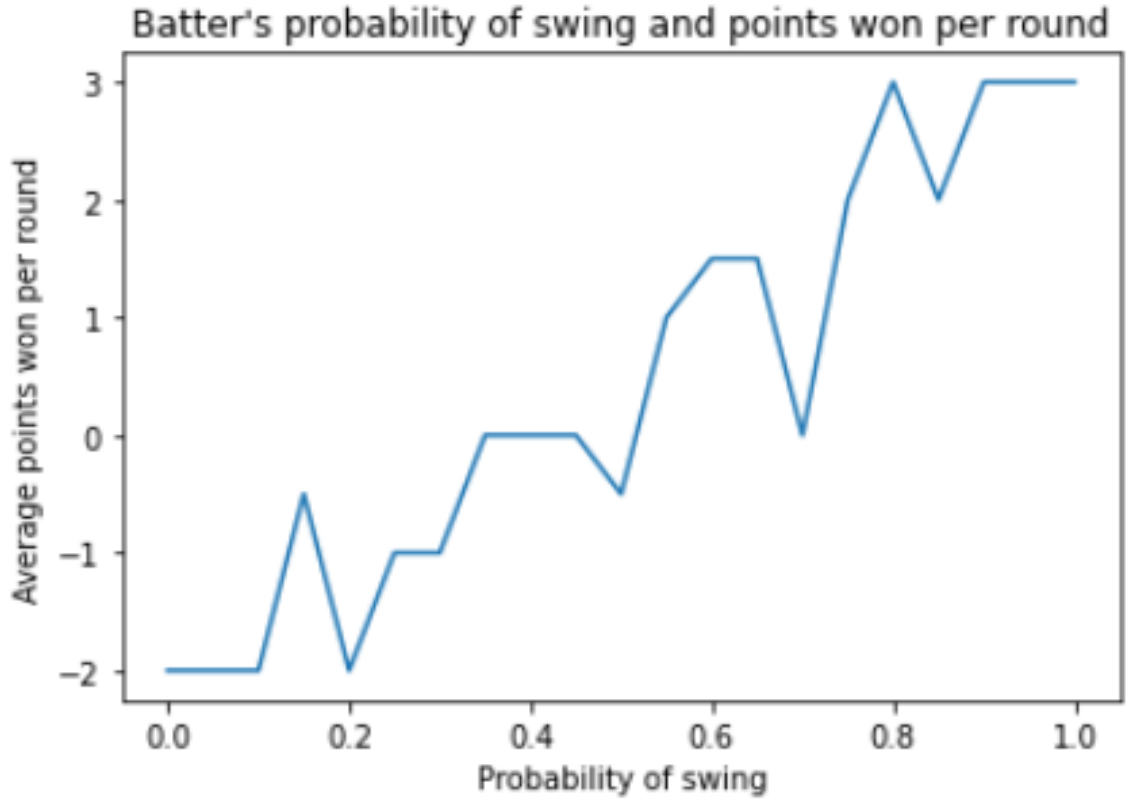


Figure 3: Simulate each interval 10 times

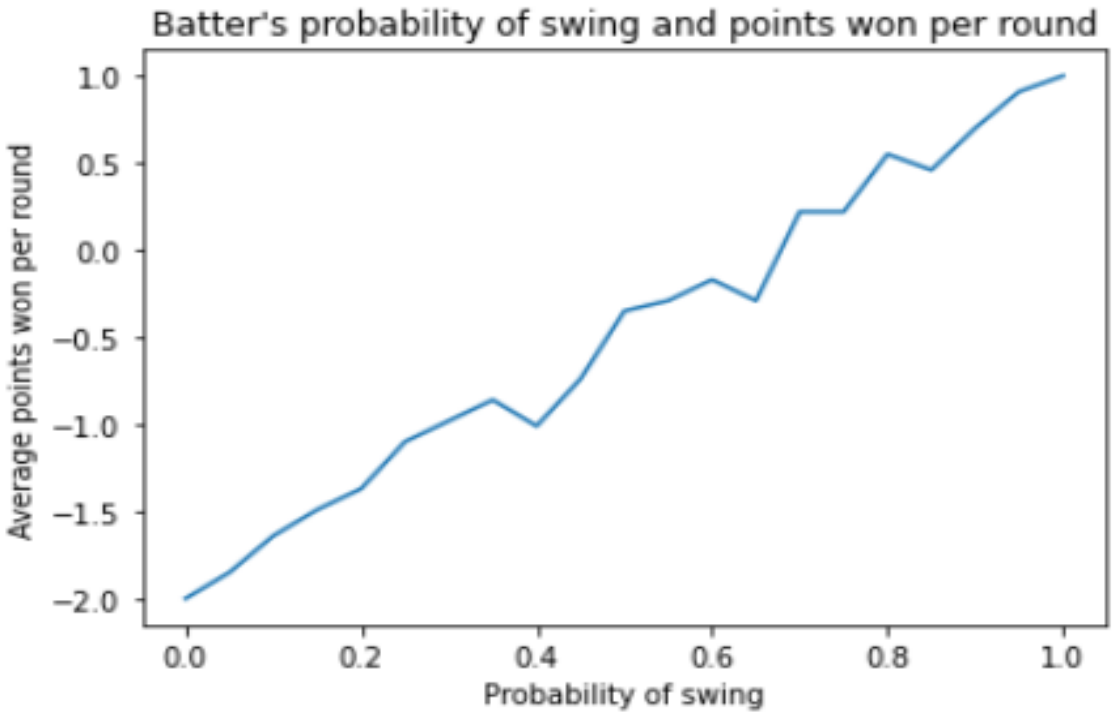


Figure 4: Simulate each interval 100 times

4.1 Only one strategy opponent will be used

4.1.1 Batter strategy

At first, we decide the optimal solution for batter by consider only one strategy pitcher will be used (pitch a goofball). We will use the program to looking for the batter's average expected point win per round when her probability of swing is 0 to 1 and the interval is 0.05. According to the calculated results, the average points won per round for the strong batter and weak batter is -2.0, -1.7545, -1.4845, -1.2325, -0.9965, -0.7525, -0.5775, -0.235, -0.028, 0.287, 0.5255, 0.771, 0.9965, 1.2505, 1.4785, 1.7645, 2.014, 2.245, 2.5205, 2.7495, 3.0 and -2.0, -1.8413, -1.6994, -1.5548, -1.3928, -1.2788, -1.0784, -0.9398, -0.788, -0.65, -0.5174, -0.3701, -0.1934, -0.0542, 0.0853, 0.2467, 0.4225, 0.5407, 0.7132, 0.844, 1.0 respectively.

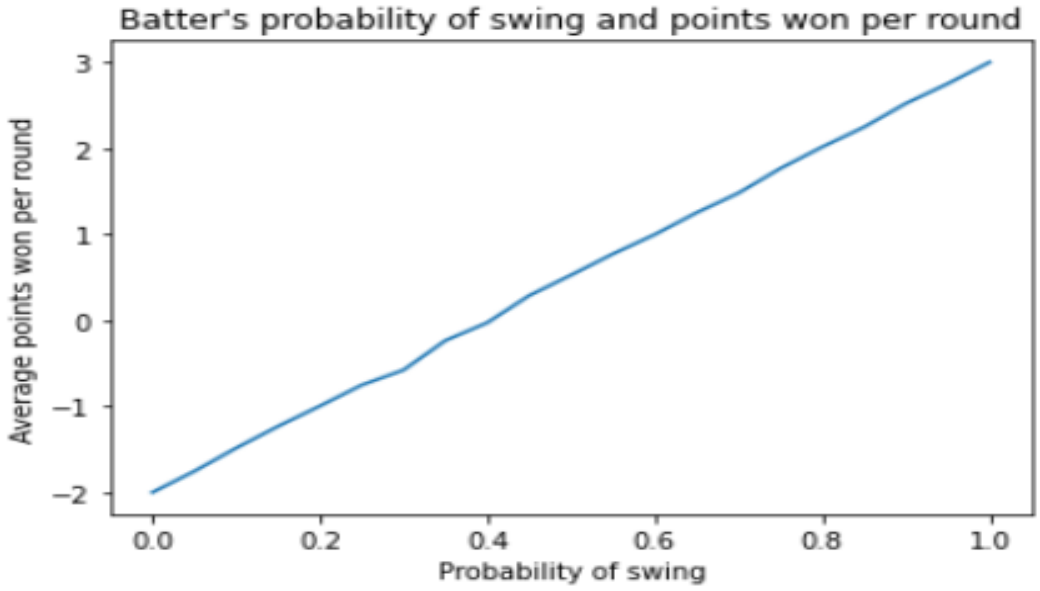


Figure 5: Strong batter's probability of swing and expected points won per round

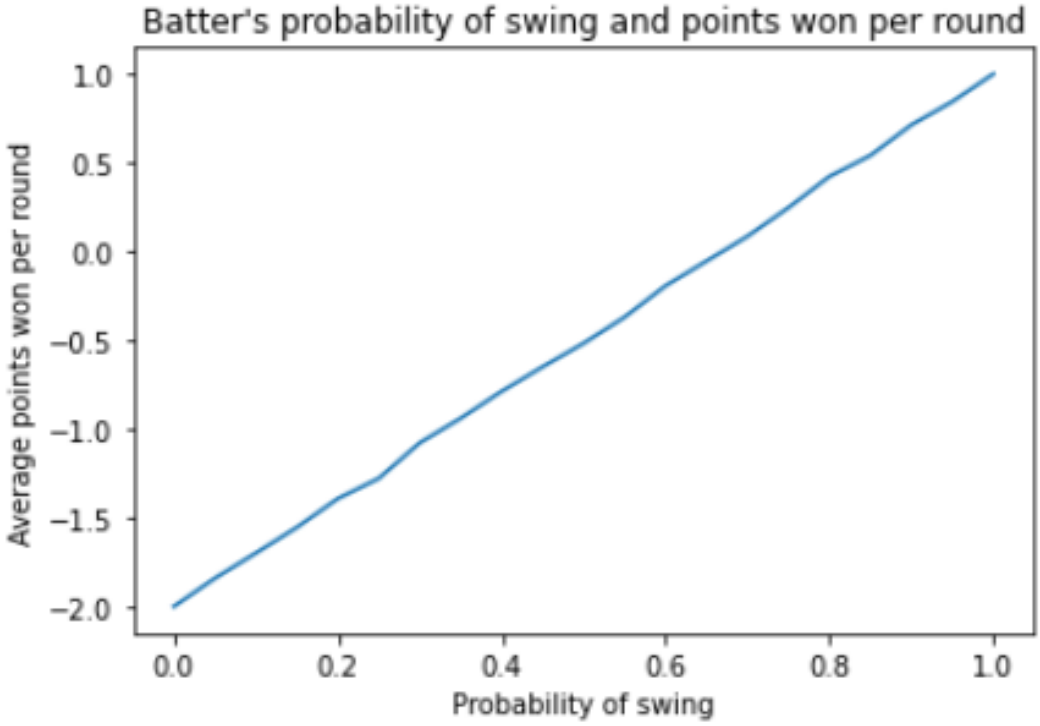


Figure 6: Weak batter's probability of swing and expected points won per round

From *figure 5 and 6*, we can find that if the pitcher always pitches a goodball the optimal strategy for strong batter and weak batter is maintain the probability of swing above 0.45 and 0.7 because the expected value for strong batter will become positive when her probability of swing is between 0.40 and 0.45 and the expected value for weak batter will become positive when her probability of swing is between 0.65 and 0.7. The optimal strategy for strong batter

is almost close to our model in chapter 3 (maintain probability of swing > 0.428) but the optimal strategy for weak batter is a bit far to our model in chapter 3 (maintain probability of swing > 0.6).

4.1.2 Pitcher strategy

We decide the optimal solution for pitcher by consider only one strategy batter will be used (swing). We will use the program to looking for the pitcher's average expected point win per round when her probability of pitch a goodball is 0 to 1 and the interval is 0.05. According to the calculated results, the average points won per round for the strong pitcher and weak pitcher is 2.0, 1.8512, 1.7048, 1.553, 1.406, 1.262, 1.1255, 0.9389, 0.8072, 0.6398, 0.5339, 0.3443, 0.1886, 0.0617, -0.0889, -0.2311, -0.4129, -0.547, -0.7024, -0.8539, -1.0 and 2.0, 1.7535, 1.5175, 1.2515, 1.004, 0.7535, 0.4975, 0.275, 0.007, -0.2435, -0.506, -0.7675, -0.983, -1.2395, -1.458, -1.7195, -2.006, -2.274, -2.475, -2.7575, -3.0 respectively.

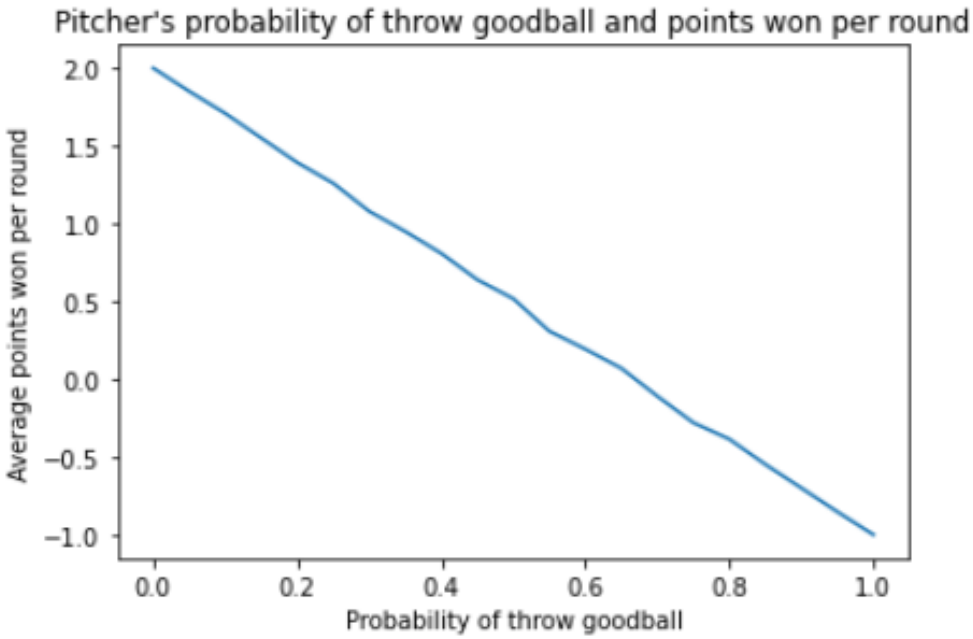


Figure 7: Strong pitcher's probability of pitch goodball and expected points won per round

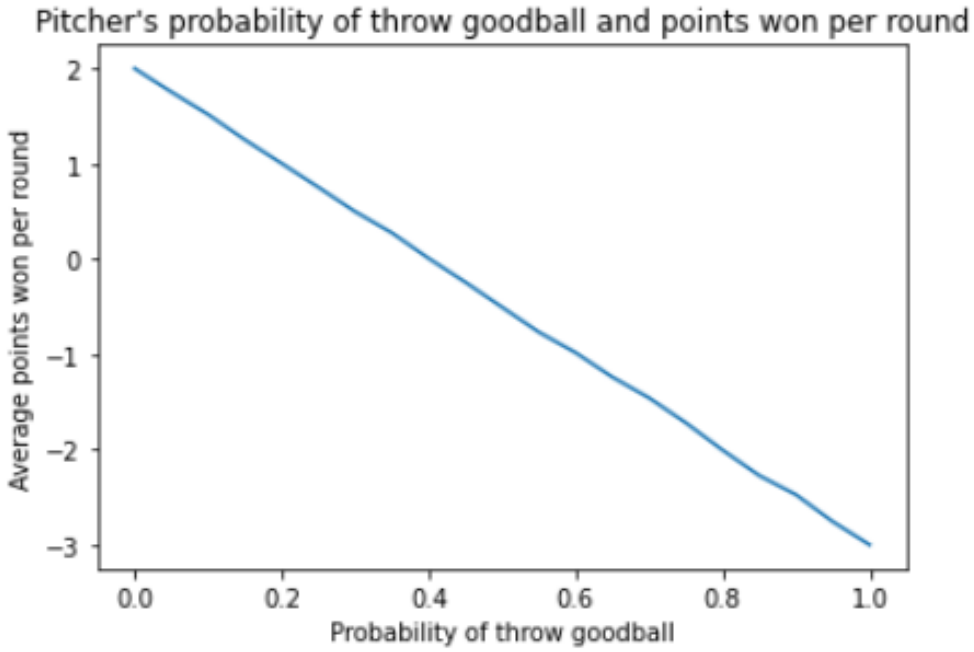


Figure 8: Weak pitcher's probability of pitch goodball and expected points won per round

From *figure 7 and 8*, we can find that if the batter always swings the optimal strategy for strong pitcher and weak pitcher is maintain the probability of pitch a goodball below 0.65 and 0.4 because the expected value for strong pitcher will become positive when her probability of pitch goodball is between 0.65 and 0.7 and the expected value for weak pitcher will become positive when her probability of swing is between 0.45 and 0.4. The optimal strategy for strong pitcher is almost close to our model in chapter 3 (maintain probability of pitch goodball < 0.6) but the optimal strategy for weak pitcher is also close to our model in chapter 3 (maintain probability of pitch goodball < 0.429).

4.2 Mix strategy opponent will be used

4.2.1 Batter strategy

Since the opponent will not only use a strategy, we need to find the optimal solution when opponent have mix strategy. At first, we consider pitcher's probability of pitch goodball is 0.5 and our ability is stronger than him. After putting the variable in the program in *appendix 2.3.* We can find that the batter's average expected point win every round is -0.0432, 0.0153, 0.0396, 0.0763, 0.1165, 0.1318, 0.1614, 0.1897, 0.2485, 0.2464, 0.1838, 0.2944, 0.2932,

0.3109, 0.3678, 0.3574, 0.4122, 0.4194, 0.4606, 0.4367, 0.4755 with a probability of swing 0, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, 0.95, 1 respectively.

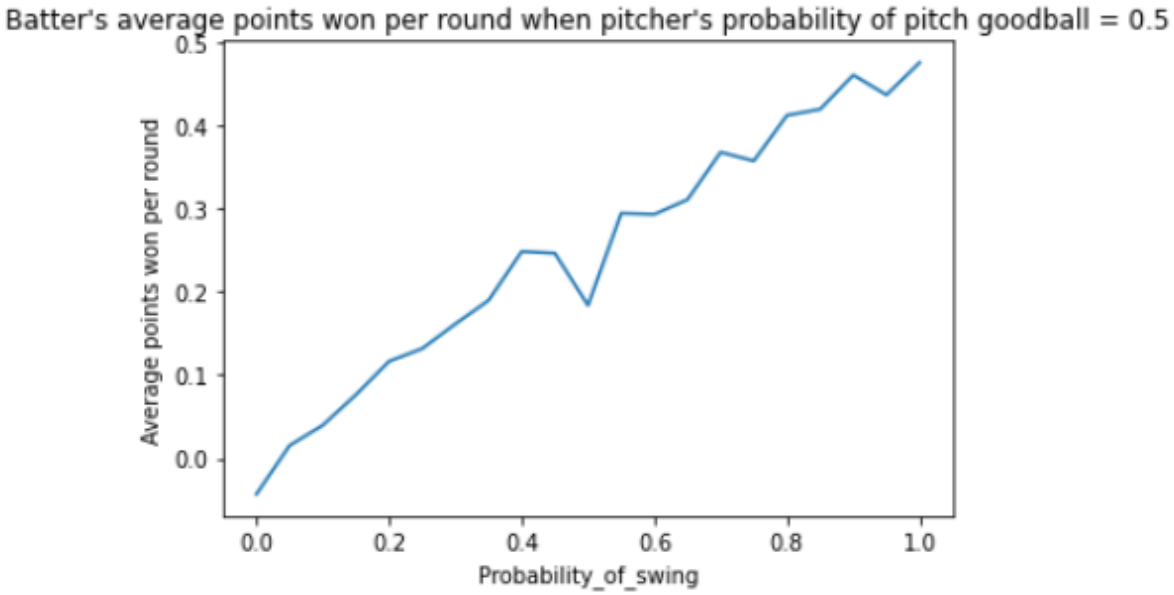


Figure 9: Strong batter average expected points won per round when pitcher's probability of pitch goodball = 0.5

From *figure 9*, we find that if pitcher's probability of pitch goodball is 0.5 and our ability is stronger than him, the optimal strategy for the batter is maintain probability of swing >0.05 and the average expected points won per round is at maximum when probability of swing = 1.

At second simulation for batter, we consider pitcher's probability of pitch goodball is 0.8 and our ability is weaker than him We found that the batter's average expected point win every round is -1.1732, -1.107, -1.0254, -0.9238, -0.866, -0.7941, -0.7064, -0.669, -0.5715, -0.4683, -0.4009, -0.3182, -0.2165, -0.1853, -0.0843, 0.0299, 0.0731, 0.1593, 0.2263, 0.3268, 0.3991 with probability of swing 0, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, 0.95, 1 respectively.

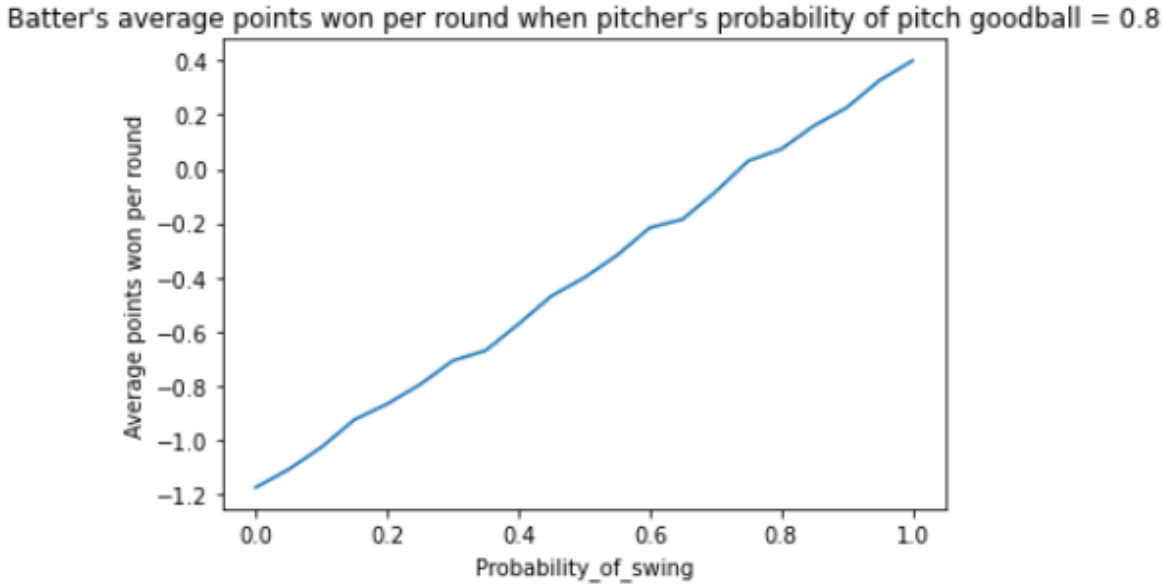


Figure 10: Weak batter average expected points won per round when pitcher's probability of pitch goodball = 0.8

From *figure 10*, we find that if pitcher's probability of pitch goodball is 0.8 and our ability is weaker than him, the optimal strategy for the batter is maintain probability of swing >0.75 and the average expected points won per round is at maximum when probability of swing = 1.

Of course, another people can use the program in *appendix 2.3* to determine their strategy by enter the probability of opponent strategy and your ability.

4.2.2 Pitcher strategy

In this part, we consider batter's probability of swing is 0.3 and our ability is stronger than him. We found that the batter's average expected point wins every round is -0.8216, -0.722, -0.5993, -0.5128, -0.4561, -0.3353, -0.2299, -0.1387, -0.0418, 0.0339, 0.1544, 0.2239, 0.3483, 0.4284, 0.5384, 0.6199, 0.7224, 0.7994, 0.9041, 1.0156, 1.0895 with a probability of pitch goodball = 0, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, 0.95, 1 respectively.

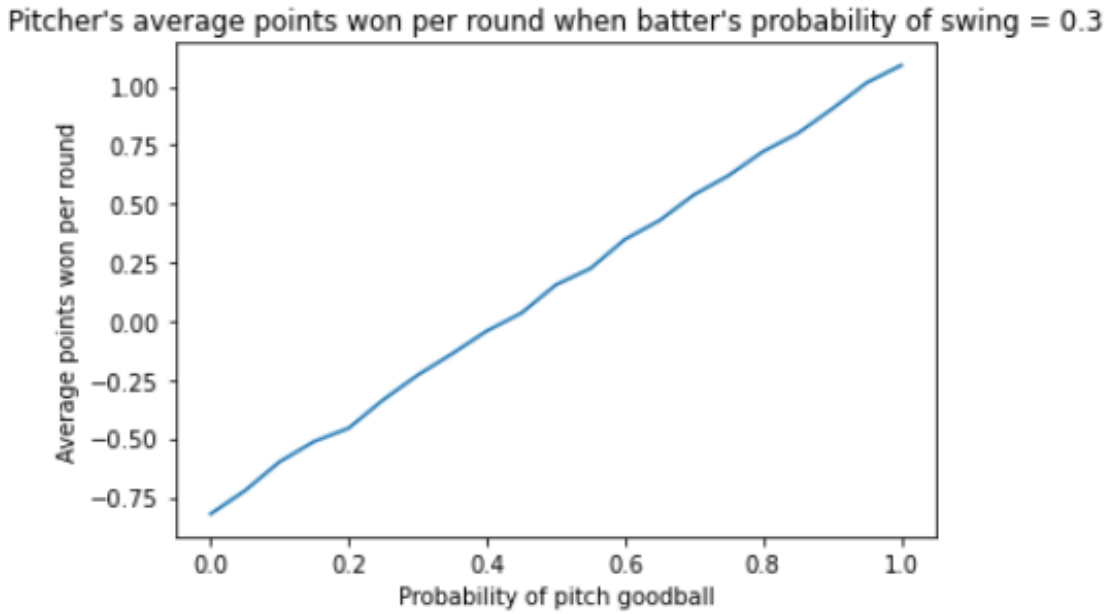


Figure 11: Strong pitcher average expected points won per round when batter's probability of swing= 0.3

From *figure 11*, we found that if batter's probability of swing is 0.3 and our ability is stronger than him, the optimal strategy for the pitcher is maintain probability of pitch goodball > 0.45 and the average expected points won per round is at maximum when probability of pitch goodball = 1.

At second simulation for pitcher, we consider batter's probability of swing is 0.8 and our ability is weaker than him We found that the pitcher's average expected point wins every round is 1.1872, 1.0744, 0.8975, 0.726, 0.5517, 0.3867, 0.2288, 0.1034, -0.0464, -0.2149, -0.377, -0.5728, -0.7249, -0.8728, -1.0243, -1.1691, -1.3706, -1.5083, -1.6907, -1.8551, -1.9885 with a probability of pitch goodball = 0, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, 0.95, 1 respectively.

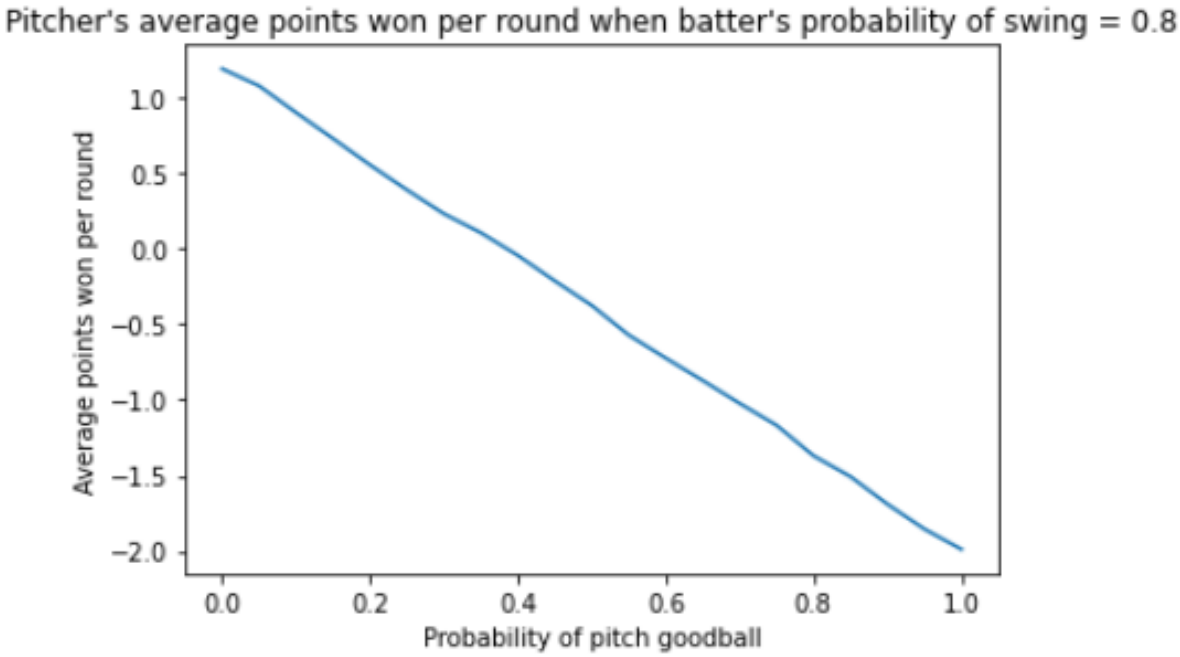


Figure 12: Weak pitcher average expected points won per round when batter's probability of swing= 0.8

From *figure 12*, we found that if batter's probability of swing is 0.8 and our ability is weaker than him, the optimal strategy for the pitcher is maintain probability of pitch goodball < 0.35 and the average expected points won per round is at maximum when probability of pitch goodball = 0.

Of course, another people can use the program in *appendix 2.3* to determine their strategy by enter the probability of opponent strategy and your ability.

Chapter 5

CONCLUSION

At first, we study about the type of game theory and consider that, pitch selection and batter responses will be solved by two-player mixed strategies with finite games. The strategy for the pitcher is throw a goodball or badball and the strategy for the batter is swing or do not swing. During the investigate of choosing the best strategy we found that we can decide the best strategy if know the probability of strategy that opponent take e.g., when pitcher have the higher probability to throw a goodball then the best strategy for batter is swing. But the payoff will not be constant while playing. So, we try to classified the player to strong and weak, strong player can get high payoff in the game and weak player get low payoff. While during this process we found that we do not have a method to decide strong player will have how many payoffs and I consider strong player add 1 and weak player -1.

When decide the batter is strong or weak, 3 variables (OBP, BA, SLG) been used because they have high correlation with team run scored and 3 variables (WHIP, ERA, OBA) been used for decide the pitcher is strong or weak because they have high correlation with team win scored but it may be have a high probability misjudgment due to less variable. Classified the player to two category only also might let the result been misjudgment.

When we simulate the game to determine the best strategy for eeach player, we simulate 10000 times for each interval of the probability that we use. At first, we consider the opponent have a constant strategy. We found that when the pitcher's probability of pitch goodball is 1, the strong batter and weak batter optimal strategy is maintain tthe probability of swing > 0.45 and 0.7

respectively. When the batter's probability of swing is 1, the strong pitcher and weak pitcher optimal strategy is maintain tthe probability of pitch goodball < 0..65 and 0.4 respectively.

Aftet that, we consider that the opponent have a mix strategy and we have do some sample of simulation, we found that if pitcher's probability of pitch goodball is 0.5 and our ability is stronger than him, the optimal strategy for the batter is maintain probability of swing >0.05. and if pitcher's probability of pitch goodball is 0.8 and our ability is weaker than him, the optimal strategy for the batter is maintain probability of swing >0.75. Beside that, we found that if batter's probability of swing is 0.3 and our ability is stronger than him, the optimal strategy for the pitcher is maintain probability of pitch goodball > 0.45 and if batter's probability of swing is 0.8 and our ability is weaker than him, the optimal strategy for the pitcher is maintain probability of pitch goodball < 0.35. Other than that, anyone is welcome to use program in *appendix in 2.2 and 2.3* to determine their strategy.

Lastly, I would like to state that the results of this project may not be the same as those in real life due to the payoff matrix of each player are hard to decide.

REFERENCES

[1] Benjamin Baumer and Andrew Zimbalist, “The Sabermetrics Revolution”, *University of pennsylvania press,* 2013.

[2] Gabriel B.Costa, Micheal R. Huber and John T.Saccoman, “Understanding Sabermetrics”, *McFarland,* 2009.

[3] Gelblum, Laucys and Saperstein and Sodha, “ The Game Theory of Baseball”, *University library of Berkeley,* 2010.

[4] Hattie James, “How Game Theory Is Applied To Pitch Optimization”, *Fan Graphs Community Research,* 2015.

[5] Issah Musah and Douglas Kwasi Boah and Baba Seidu,” A Comprehensive Review of Solution Methods and Techniques for Solving Games in Game Theory”, *Journal of Game Theory,* 2020.

[6] Jackson, Matthew O, “A Brief Introduction to the Basics of Game ”. Available at SSRN: https://ssrn.com/abstract=1968579 or http://dx.doi.org/10.2139/ssrn.1968579, 2011.

[7] Jim Albert., “Sabermetrics: The Past, the Present, and the Future”, *ResearchGate, 10.5948/UPO9781614442004.002,* 2010.

[8] Kevin Leyton-Brown and Yoav Shoham., “Essentials of Game Theory: A Concise, Multidisciplinary Introduction”, *Morgan & Claypool,* 2008.

[9] Martin J.Osborne., “An Introduction to Game Theory”, *oxford university press,* 2000.

[10] Matt Swartz., “Game Theory and Baseball Part 1-5”, *Fan Graphs Community Research,* 2011.

[11] Micah Melling, “Game Theory Applications in Baseball”, *Baseball Data Science,* 2018.

[12] Piper Slowinski., Plate Discipline (O-swing%, Z-swing%, etc)”, *Fan Graphs Community Research,* 2012.

APPENDICES

APPENDIX 1: Glossary

GP: Games Played
AB: At Bats
R: Runs
H: Hits
2B: Doubles
3B: Triples
HR: Home Runs
RBI: Runs Batted In
TB: Total Bases
BB: Walks
SO: Strikeouts
SB: Stolen Bases
AVG: Batting Average
OBP: On Base Percentage
SLG: Slugging Percentage
OPS: OPB Pct + SLG Pct
Appendix 1.1 The glossary of batter data

GP: Games Played
W: Wins
L: Losses
ERA: Earned Run Average
SV: Saves
CG: Complete Games
SHO: Shutouts
QS: Quality Starts
IP: Innings pitched
H: Hits
ER: Earned runs
HR: Home Runs
BB: Walks
SO: Strikeouts
OBA: Opponent Batting Average
WHIP: Walks Plus Hits Per Inning Pitched
Appendix 1.2 The glossary in the mlbpitching2014-2018

APPENDIX 2: Computer Programme Listing

```
import pandas as pd
import numpy as np
file ="FYP\pit2020.csv"
df = pd.read_csv(file)
df.sample(10)
```

	GP	W	L	ERA	SV	CG	SHO	QS	IP	H	ER	HR	BB	SO	OBA	WHIP
139	162	96	66	3.03	45	5	19	106	1,470.20	1,351	495	110	352	1,288	0.244	1.16
45	162	92	70	3.95	38	2	8	77	1,447.10	1,294	636	194	554	1,439	0.238	1.28
101	162	93	69	3.80	34	7	10	84	1,441.00	1,353	609	173	397	1,117	0.248	1.21
106	162	92	70	3.44	47	6	21	95	1,445.20	1,317	553	145	395	1,396	0.242	1.18
149	162	77	85	3.56	37	3	22	84	1,463.20	1,292	579	145	482	1,437	0.234	1.21
62	162	74	88	4.28	29	4	12	64	1,421.10	1,480	676	208	498	1,136	0.269	1.39
89	161	79	82	4.05	55	0	12	63	1,435.00	1,358	646	152	595	1,379	0.251	1.36
116	162	79	83	4.04	44	1	12	69	1,466.20	1,450	659	182	500	1,215	0.258	1.33
136	162	76	86	3.59	44	5	13	103	1,446.00	1,282	576	163	507	1,290	0.237	1.24
54	162	71	91	4.67	45	2	12	67	1,430.20	1,417	742	226	554	1,325	0.259	1.38

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 16 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   GP      150 non-null    int64
 1   W       150 non-null    int64
 2   L       150 non-null    int64
 3   ERA     150 non-null    float64
 4   SV      150 non-null    int64
 5   CG      150 non-null    int64
 6   SHO     150 non-null    int64
 7   QS      150 non-null    int64
 8   IP      150 non-null    object
 9   H       150 non-null    object
 10  ER      150 non-null    int64
 11  HR      150 non-null    int64
 12  BB      150 non-null    int64
 13  SO      150 non-null    object
 14  OBA     150 non-null    float64
 15  WHIP    150 non-null    float64
dtypes: float64(3), int64(10), object(3)
memory usage: 18.9+ KB
```

corr_matrix = df.corr ('pearson')

corr_matrix.shape

print(corr_matrix)

```
             GP         W
GP     1.000000  0.148107 ·
W      0.148107  1.000000 ·
L     -0.123670 -0.999645
ERA   -0.106815 -0.743581
SV     0.126572  0.618413 ·
CG    -0.198420  0.190239 ·
SHO    0.023655  0.478079 ·
QS     0.136282  0.518454 ·
ER    -0.093792 -0.739420
HR    -0.015619 -0.390623
BB     0.013301 -0.423417
OBA   -0.099907 -0.741970
WHIP -0.074646 -0.755924
```

new_df = df.iloc[:, [3, 14, 15]]

np.mean(new_df, axis=0)

```
ERA     4.076067
OBA     0.252660
WHIP    1.308000
dtype: float64
```

Appendix 2.1 Find Pearson's correlation and mean for mlbpitching2014-2018.csv in python

```
import random
import matplotlib.pyplot as plt

payoff_sbtr = [
    [-3,3], [2, -2],
    [2, -2], [-2,2]]
payoff_wbtr = [
    [-1,1], [2, -2],
    [2, -2], [-2,2]]
payoff_sptr = [
    [-1,1],[2,-2],
    [2,-2],[-2,2]]
payoff_wptr = [
    [-3,3],[2,-2],
    [2,-2],[-2,2]]

def strategy_sbtr(probability_of_swing):
    probability_of_take = (1-probability_of_swing)
    strategy = [payoff_sbtr[0][1], payoff_sbtr[1][1]]
```

```
    return random.choices(strategy,  weights = [probability_of_swing, probability_of_take] , k
= 1)

def strategy_wbtr(probability_of_swing):
    probability_of_take = (1-probability_of_swing)
    strategy = [payoff_wbtr[0][1], payoff_wbtr[1][1]]
    return random.choices(strategy,  weights = [probability_of_swing, probability_of_take] , k
= 1)

def strategy_sptr(probability_of_tgoodball):
    probability_of_tbadball = (1-probability_of_tgoodball)
    strategy = [payoff_sptr[0][0], payoff_sptr[2][0]]
    return random.choices(strategy,  weights = [probability_of_tgoodball,
probability_of_tbadball] , k = 1)

def strategy_wptr(probability_of_tgoodball):
    probability_of_tbadball = (1-probability_of_tgoodball)
    strategy = [payoff_wptr[0][0], payoff_wptr[2][0]]
    return random.choices(strategy,  weights = [probability_of_tgoodball,
probability_of_tbadball] , k = 1)

print("btr = batter")
print("ptr = pitcher")
role = input("Please enter your role : ")
print("s = strong")
print("w = weak")
ability = input("Consider your ability : ")

probability_of_swing =
[0,0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,
                    0.85,0.90,0.95,1]
average_total_value_10 = []
if role == "btr":
    if ability == "s":
```

```
    j = 0
    while j <= 20:
        i = 0
        values = 0
        total_value = 0
        total_value_1 = 0
        probability_of_swing_lp = probability_of_swing[j]
        while i <= 9999:
            values = strategy_sbtr(probability_of_swing_lp)
            strings = [str(value) for value in values]
            a_string = "".join(strings)
            an_integer = int(a_string)
            total_value += an_integer
            i += 1
        total_value_1 += total_value
        average_total_value_10.append(total_value_1/10000)
        j += 1
elif ability == "w":
    j = 0
    total_value_1 = 0
    while j <= 20:
        i = 0
        values = 0
        total_value = 0
        total_value_1 = 0
        probability_of_swing_lp = probability_of_swing[j]
        while i <= 9999:
            values = strategy_wbtr(probability_of_swing_lp)
            strings = [str(value) for value in values]
            a_string = "".join(strings)
            an_integer = int(a_string)
            total_value += an_integer
            i += 1
        total_value_1 += total_value
```

```
            average_total_value_10.append(total_value_1/10000)
            j += 1

    print(f'Average points won per round = {average_total_value_10}')
    plt.title("Batter's probability of swing and points won per round")
    plt.xlabel("Probability of swing")
    plt.ylabel("Average points won per round")
    plt.plot(probability_of_swing,average_total_value_10)
    plt.show()

probability_of_tgoodball =
[0,0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,
                    0.85,0.90,0.95,1]
average_total_value_10 = []
if role == "ptr":
    if ability == "s":
        j = 0
        while j <= 20:
            i = 0
            values = 0
            total_value = 0
            total_value_1 = 0
            probability_of_tgoodball_lp = probability_of_tgoodball[j]
            while i <= 9999:
                values = strategy_sptr(probability_of_tgoodball_lp)
                strings = [str(value) for value in values]
                a_string = "".join(strings)
                an_integer = int(a_string)
                total_value += an_integer
                i += 1
            total_value_1 += total_value
            average_total_value_10.append(total_value_1/10000)
            j += 1
    elif ability == "w":
```

```
    j = 0
    while j <= 20:
        i = 0
        values = 0
        total_value = 0
        total_value_1 = 0
        probability_of_tgoodball_lp = probability_of_tgoodball[j]
        while i <= 9999:
            values = strategy_wptr(probability_of_tgoodball_lp)
            strings = [str(value) for value in values]
            a_string = "".join(strings)
            an_integer = int(a_string)
            total_value += an_integer
            i += 1
        total_value_1 += total_value
        average_total_value_10.append(total_value_1/10000)
        j += 1

print(f'Average points won per round = {average_total_value_10}')
plt.title("Pitcher's probability of throw goodball and points won per round")
plt.xlabel("Probability of throw goodball")
plt.ylabel("Average points won per round")
plt.plot(probability_of_tgoodball,average_total_value_10)
plt.show()
```

Appendix 2.2: Simulation when opponent only use one strategy

```
import random
import matplotlib.pyplot as plt

payoff_sbtr = [
    [-3,3],[2,-2],
    [2,-2],[-2,2]]
payoff_wbtr = [
    [-1,1],[2,-2],
    [2,-2],[-2,2]]
payoff_sptr = [
    [-1,1],[2,-2],
    [2,-2],[-2,2]]
payoff_wptr = [
    [-3,3],[2,-2],
    [2,-2],[-2,2]]

def strategy_sbtr(probability_of_swing, probability_of_tgoodball):

    probability_of_take = (1-probability_of_swing)
```

```
    probability_of_tbadball = (1-probability_of_tgoodball)
    batter_strategy = ["swing","take"]
    pitcher_strategy = ["goodball", "badball"]
    batter_strategy_d = random.choices(batter_strategy,  weights = [probability_of_swing,
                                                probability_of_take] , k = 1)
    pitcher_strategy_d = random.choices(pitcher_strategy,  weights =
[probability_of_tgoodball,
                                                probability_of_tbadball] , k = 1)

    for element in batter_strategy_d:
        batter_strategy_1 = ""
        batter_strategy_1 += element

    for element in pitcher_strategy_d:
        pitcher_strategy_1 = ""
        pitcher_strategy_1 += element

    if batter_strategy_1 == "swing":
        if pitcher_strategy_1 == "goodball":
            return  payoff_sbtr[0][1]
        elif pitcher_strategy_1 == "badball":
            return payoff_sbtr[2][1]
    elif batter_strategy_1 == "take":
        if pitcher_strategy_1 == "goodball":
            return payoff_sbtr[1][1]
        elif pitcher_strategy_1 == "badball":
            return payoff_sbtr[3][1]


def strategy_wbtr(probability_of_swing, probability_of_tgoodball):

    probability_of_take = (1-probability_of_swing)
    probability_of_tbadball = (1-probability_of_tgoodball)
    batter_strategy = ["swing","take"]
```

```
    pitcher_strategy = ["goodball", "badball"]
    batter_strategy_d = random.choices(batter_strategy,  weights = [probability_of_swing,
                                                    probability_of_take] , k = 1)
    pitcher_strategy_d = random.choices(pitcher_strategy,  weights =
[probability_of_tgoodball,
                                                    probability_of_tbadball] , k = 1)

    for element in batter_strategy_d:
        batter_strategy_1 = ""
        batter_strategy_1 += element

    for element in pitcher_strategy_d:
        pitcher_strategy_1 = ""
        pitcher_strategy_1 += element

    if batter_strategy_1 == "swing":
        if pitcher_strategy_1 == "goodball":
            return  payoff_wbtr[0][1]
        elif pitcher_strategy_1 == "badball":
            return payoff_wbtr[2][1]
    elif batter_strategy_1 == "take":
        if pitcher_strategy_1 == "goodball":
            return payoff_wbtr[1][1]
        elif pitcher_strategy_1 == "badball":
            return payoff_wbtr[3][1]


def strategy_sptr(probability_of_swing, probability_of_tgoodball):

    probability_of_take = (1-probability_of_swing)
    probability_of_tbadball = (1-probability_of_tgoodball)
    batter_strategy = ["swing","take"]
    pitcher_strategy = ["goodball", "badball"]
    batter_strategy_d = random.choices(batter_strategy,  weights = [probability_of_swing,
```

```
                                                probability_of_take] , k = 1)
    pitcher_strategy_d = random.choices(pitcher_strategy,  weights =
[probability_of_tgoodball,
                                                 probability_of_tbadball] , k = 1)

    for element in batter_strategy_d:
        batter_strategy_1 = ""
        batter_strategy_1 += element

    for element in pitcher_strategy_d:
        pitcher_strategy_1 = ""
        pitcher_strategy_1 += element



    if batter_strategy_1 == "swing":
        if pitcher_strategy_1 == "goodball":
            return  payoff_sptr[0][0]
        elif pitcher_strategy_1 == "badball":
            return payoff_sptr[2][0]
    elif batter_strategy_1 == "take":
        if pitcher_strategy_1 == "goodball":
            return payoff_sptr[1][0]
        elif pitcher_strategy_1 == "badball":
            return payoff_sptr[3][0]



def strategy_wptr(probability_of_swing, probability_of_tgoodball):

    probability_of_take = (1-probability_of_swing)
    probability_of_tbadball = (1-probability_of_tgoodball)
    batter_strategy = ["swing","take"]
    pitcher_strategy = ["goodball", "badball"]
    batter_strategy_d = random.choices(batter_strategy,  weights = [probability_of_swing,
                                                probability_of_take] , k = 1)
```

```
    pitcher_strategy_d = random.choices(pitcher_strategy,  weights =
[probability_of_tgoodball,
                                                probability_of_tbadball] , k = 1)

    for element in batter_strategy_d:
        batter_strategy_1 = ""
        batter_strategy_1 += element

    for element in pitcher_strategy_d:
        pitcher_strategy_1 = ""
        pitcher_strategy_1 += element



    if batter_strategy_1 == "swing":
        if pitcher_strategy_1 == "goodball":
            return  payoff_wptr[0][0]
        elif pitcher_strategy_1 == "badball":
            return payoff_wptr[2][0]
    elif batter_strategy_1 == "take":
        if pitcher_strategy_1 == "goodball":
            return payoff_wptr[1][0]
        elif pitcher_strategy_1 == "badball":
            return payoff_wptr[3][0]

print("btr = batter")
print("ptr = pitcher")
role = input("Please enter your role : ")
print("s = strong")
print("w = weak")
ability = input("Consider your ability : ")

probability_of_swing =
[0,0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,
                    0.85,0.90,0.95,1]
```

```
average_total_value_10 = []
if role == "btr":
    probability_of_tgoodball = float(input("Please enter opponent's probability of throw goodball : "))
    if ability == "s":
        j = 0
        while j <= 20:
            i = 0
            values = 0
            total_value = 0
            total_value_1 = 0
            probability_of_swing_lp = probability_of_swing[j]
            while i <= 9999:
                values = strategy_sbtr(probability_of_swing_lp, probability_of_tgoodball)
                total_value += values
                i += 1
            total_value_1 += total_value
            average_total_value_10.append(total_value_1/10000)
            j += 1
    elif ability == "w":
        j = 0
        while j <= 20:
            i = 0
            values = 0
            total_value = 0
            total_value_1 = 0
            probability_of_swing_lp = probability_of_swing[j]
            while i <= 9999:
                values = strategy_wbtr(probability_of_swing_lp, probability_of_tgoodball)
                total_value += values
                i += 1
            total_value_1 += total_value
            average_total_value_10.append(total_value_1/10000)
            j += 1
```

```
    print(f'Average points won per round = {average_total_value_10}')
    plt.title(f"Batter's average points won per round when pitcher's probability of pitch
goodball = {probability_of_tgoodball}")
    plt.xlabel("Probability_of_swing")
    plt.ylabel("Average points won per round")
    plt.plot(probability_of_swing,average_total_value_10)
    plt.show()

    probability_of_tgoodball =
[0,0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,
                        0.85,0.90,0.95,1]
    average_total_value_10 = []
    if role == "ptr":
        probability_of_swing = float(input("Please enter opponent's probability of swing : "))
        if ability == "s":
            j = 0
            while j <= 20:
                i = 0
                values = 0
                total_value = 0
                total_value_1 = 0
                probability_of_tgoodball_lp = probability_of_tgoodball[j]
                while i <= 9999:
                    values = strategy_sptr(probability_of_swing, probability_of_tgoodball_lp)
                    total_value += values
                    i += 1
                total_value_1 += total_value
                average_total_value_10.append(total_value_1/10000)
                j += 1
        elif ability == "w":
            j = 0
            while j <= 20:
                i = 0
```

```
            values = 0
            total_value = 0
            total_value_1 = 0
            probability_of_tgoodball_lp = probability_of_tgoodball[j]
            while i <= 9999:
                values = strategy_wptr(probability_of_swing, probability_of_tgoodball_lp)
                total_value += values
                i += 1
            total_value_1 += total_value
            average_total_value_10.append(total_value_1/10000)
            j += 1


    print(f'Average points won per round = {average_total_value_10}')
    plt.title(f"Pitcher's average points won per round when batter's probability of swing = {probability_of_swing}")
    plt.xlabel("Probability of pitch goodball")
    plt.ylabel("Average points won per round")
    plt.plot(probability_of_tgoodball,average_total_value_10)
    plt.show()
```

Appendix 2.3: Simulation when opponent use mix strategy