University of Alberta

THE EFFECTIVENESS OF OPPONENT MODELLING IN A SMALL IMPERFECT INFORMATION GAME

by

Bret Hoehn

©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Spring 2006

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-494-13824-6
Our file Notre référence
ISBN: 0-494-13824-6

Abstract

Opponent modelling is an important issue in games programming today. Programs which do not perform opponent modelling are unlikely to take full advantage of the mistakes made by an opponent. Additionally, programs which do not adapt over time become less of a challenge to players, causing these players to lose interest. While opponent modelling can be a difficult challenge in perfect information games, where the full state of the game is known to all players at all times, it becomes an even more difficult task in games of imperfect information, where players are not always able to observe the actual state of the game. This thesis studies the problem of opponent modelling in Kuhn Poker, a small imperfect information game that contains several properties that make real-world poker games interesting. Two basic types of opponent modelling are studied, explicit modelling and implicit modelling, and the effectiveness is compared.

Acknowledgements

I would first like to thank my supervisor Dr. Robert Holte for his guidance in this project; whenever I ran out of ideas on how to proceed, Rob always helped me come up with new ways of attacking the problem. Rob has been a tremendous help in keeping this research on the right track while also encouraging me to explore many of the interesting side-tracks.

I would also like to thank Finnegan Southey, who has also provided a great deal of insight into this research. Finnegan has been very helpful in keeping this research in perspective and in motivating further research. I am especially thankful for Finnegan's help in presenting the findings of this study in a clear and concise manner.

Many other people have provided important insights at various stages of this research, including Valeriy Bulitko, Neil Burch, Dale Schuurmans, Michael Bowling, Martin Zinkevich, and the entire University of Alberta Computer Poker Research Group.

My family has been very helpful throughout my education, providing support when I've needed it and backing off when I've needed space to work. I am eternally thankful to Keri, who has been extremely supportive and patient during the writing of this thesis, doing whatever she could to make this process easier for me.

Finally, I would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC), Alberta's Informatics Circle of Research Excellence (iCORE), the Alberta Ingenuity Center for Machine Learning (AICML), and the University of Alberta for their financial support, without which this research would not have been possible.

Contents

List of Tables

List of Figures

List of Symbols

Symbol	Definition
α	Kuhn Poker parameter specifying the probability that Player 1 bets in Round One with the Jack
β	Kuhn Poker parameter specifying the probability that Player 1 bets in Round Three with the Queen
γ	Kuhn Poker parameter specifying the probability that Player 1 bets in Round One with the King
η	Kuhn Poker parameter specifying the probability that Player 2 bets in Round Two with the Queen after a P1 bet
ξ	Kuhn Poker parameter specifying the probability that Player 2 bets in Round Two with the Jack after a P1 pass
ϕ	The empty sequence, representing the state of a game where the deal has occurred but no player has performed an action.
ρ	Parameter for the Hedge algorithm, controlling the emphasis of play towards the highest-rated action
ψ	Parameter for the Exp3 algorithm, specifying the amount of uniform exploration among actions.

Chapter 1

Introduction

1.1 Problem Definition

Opponent modelling is currently a major issue in games programming. Many games lose their appeal to players as the players find weaknesses in the game AI and the game becomes less of a challenge. Game programs which do opponent modelling can tailor the play of the game to make it harder for the user, which gives the program more of a lasting appeal as it continues to challenge the user. However, the problem of performing effective opponent modelling, in terms of quickly generating a model that accurately predicts what the opponent will do, is extremely difficult in most applications. This problem can be challenging in games of perfect information, where the full state of the game is known to all players at all times. This problem becomes even more difficult when the domain is a game of imperfect information, where players make decisions without knowing the precise state of the game. For example, poker is an imperfect information game, where players are not informed of what their opponents' private cards are.

Poker presents a very interesting challenge in artificial intelligence research. While world-class computer players have been developed for perfect-information games such as checkers and chess, computer programs for poker have not been as successful. Some of the strongest current poker-playing programs are based on game-theoretic techniques; unfortunately, game-theoretic approaches are quickly reaching their computational bounds, as full-scale poker games such as Texas Hold'em are simply too big to solve at this time. In addition, game-theoretic solutions have a tendency to limit the winnings as well as the losses of the player using them. Since the goal is to defeat opponents and not just break even, being adaptable to different opponents is a key component of current poker programs in development.

In poker, hands often end without the players' private cards revealed, which is one of the major issues that makes opponent modelling in this setting difficult. When a hand ends with one player folding, the modeller is left to wonder which of the large number of potential hands his opponent held and based his decisions on. There is also a great deal of variance in the game, stemming from the size of the deck and from players using stochastic strategies. Another challenge is that against human opponents it is likely that most matches will only last a short time (maybe only 50 or 100 hands), meaning there is very little time to learn an opponent model. To top it off, if all of these challenges are overcome and a good opponent model is developed, an opponent may change his strategy, making the current model useless or even harmful.

The research in this thesis studies the effectiveness of different modelling techniques in an ideal setting; the context is the small two-player game of Kuhn Poker, and the opponent being modelled plays a fixed strategy. Two diametrically opposite types of opponent modelling are compared: explicit modelling and implicit modelling. Explicit modelling involves identifying the opponent's strategy in order to discover weaknesses, and using this model to develop an effective counter-strategy. Implicit modelling involves using different counter-strategies against the opponent and finding one which is effective, without worrying about the exact nature of the opponent's weaknesses. The problem has been reduced from a real-world game to a simpler setting to prevent the results from being obscured by the variance in the game, as well as the sparseness of data.

Opponents are restricted to fixed strategies because the techniques are being evaluated on how quickly the opponent model matches the opponent; evaluation becomes much trickier when both the model and the target are simultaneously changing. Being able to quickly and effectively model a stationary opponent is a logical first step towards being able to model a dynamic opponent.

If the ultimate goal is to be able to do effective modelling in a real-world game, then the ability to first do effective modelling in an ideal setting is a necessity. One of the key insights of this research is that even in the ideal situation, opponent modelling is quite difficult. In the small game being studied, the best counter-strategy to the opponent is often not discovered. However, the fact that the problem has been significantly simplified here allows for extensive analysis of precisely where the difficulties lie, and how they may be addressed. Difficulties in opponent modelling found in this ideal setting will surely be present in a full-scale poker game.

1.2 Approach to the Problem

The technique of explicit modelling discussed in this thesis assumes that the opponent is playing a fixed stochastic strategy. Such strategies are defined by a set of parameters, with each parameter specifying the probability of taking a particular action when faced with a certain situation. The goal of explicit modelling is to estimate the opponent's parameters; once the parameters have been estimated, a suitable counter-strategy can be computed.

The technique of implicit modelling is given several counter-strategies to use against the opponent. The modeller samples from this set of strategies as he plays against the opponent, attempting to determine which is the best counter-strategy. The sampling of strategies is done in accordance with the Exp3 algorithm [2], which has excellent long-term performance guarantees. However, since this thesis is concerned with winning in short matches, modifications are made to improve the short-term performance while not damaging the long-term guarantees.

The opponent modelling techniques are evaluated with two primary measures. Both measures involve the modeller collecting data about the opponent from hand 1 until hand t, deciding which counter-strategy appears to be the best against the opponent at that time, and then playing this counter-strategy for the remainder of the match. The set of hands from 1 to t is called the *exploration phase*, while the set of hands from $t + 1$ and onwards is called the *exploitation phase.* Hand t is known as the *switching hand* as this accurately describes the modeller's complete shift from exploration to exploitation at that hand. The first measure for evaluating the opponent modelling techniques is the winning rate of the counter-strategy suggested by the model at the switching hand. Plotting this metric for each possible switching hand generates payoff-rate graphs, which trace the progress of the model over the course of the match. Figure 1.1 is an example of a payoff-rate plot.

The second measure is the expected total winnings of the model, which is calculated as the sum of the winnings achieved during the exploration phase plus the expected winnings of the exploitation phase, assuming a specific total number of hands is to be played. Plotting this measure over each possible switching hand results in total winnings graphs, such as the example shown in Figure 1.2.

Both of these measures are important, as the first measure indicates how quickly the model is converging to the correct one, while the second measure indicates whether opponent modelling is worth the cost required to develop the model.

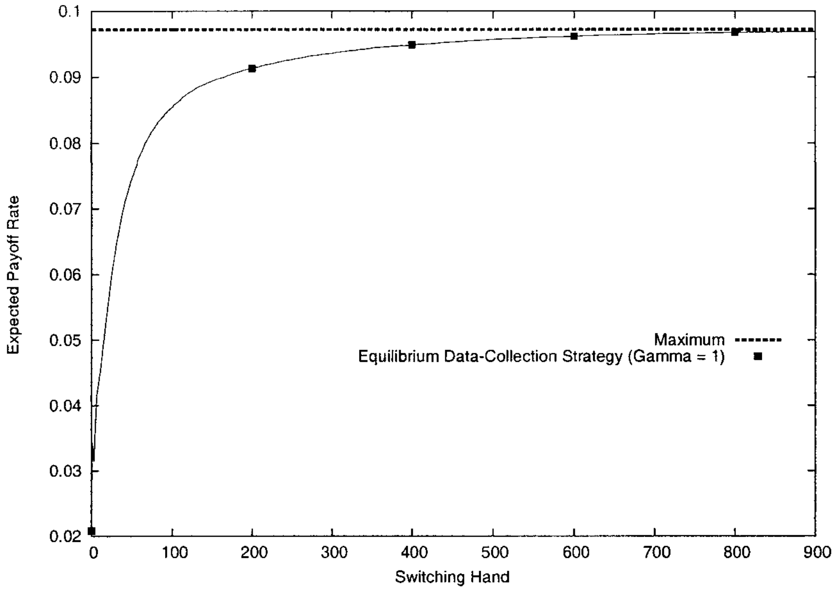


Figure 1.1: Sample Payoff-Rate Plot

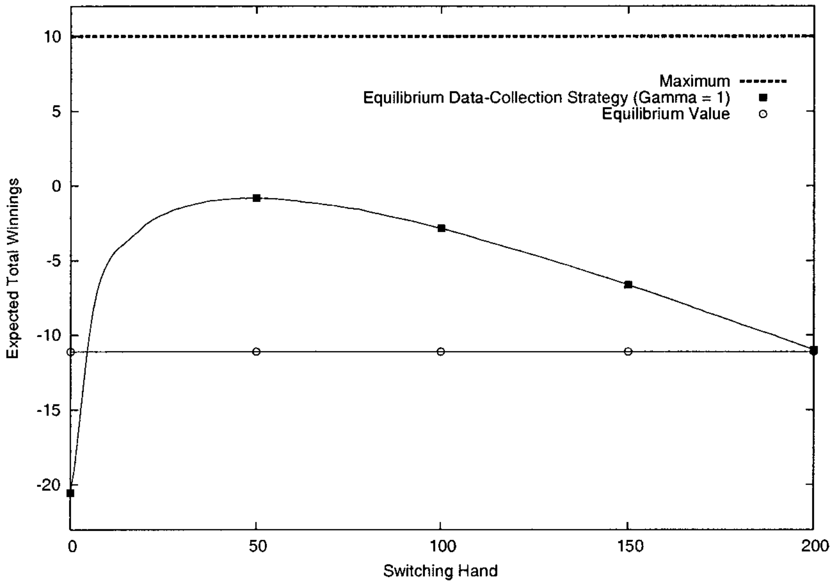


Figure 1.2: Sample Total winnings Plot

1.3 Contributions of this Research

The first major contribution of this thesis is the development of parameter estimation methods in imperfect information games, a problem which has not previously received much attention. Many different types of situations are identified, where different types of information are available; methods to make use of the available data are described for each of these settings.

A second major contribution is the adaptation of the well-known Exp3 algorithm for use in a multi-action game, as it is designed for a slightly different problem. Additionally, several enhancements are made to improve short-term performance of the algorithm without hurting the long-term guarantees.

This thesis provides insight into many of the problems faced by researchers doing modelling in larger systems, which may not be as easy to observe in the large systems. First and foremost, this thesis shows that even in a highly idealized setting, opponent modelling is a difficult problem. This suggests that the issues of partial observability and variance in the game are major contributors to opponent modelling difficulties; not all of the difficulties encountered by opponent modelling systems in larger settings are due to huge models and sparse data. Another insight, which has been known to game theorists for years but may not be commonly known among AI researchers, is that when a game player uses an equilibrium strategy (a game-theoretic solution to the game), the player often limits his own winnings as well as the winnings of his opponent. Developing programs that will win requires taking advantage of an opponent, which means straying from equilibrium solutions and risking defeat. Effective opponent modelling methods must be developed to guide the departure from equilibrium strategies and minimize the risk of losing.

The experiments shown in this thesis demonstrate many interesting conclusions. While it is not possible to always find the best counter-strategy in a short match, improving upon the equilibrium rate is almost always possible. In this small game where data is readily available, the impact of bad initial parameter estimates is quickly eliminated. Another interesting result is that the interval of time for which switching from exploration to exploitation makes big gains seems to be relatively insensitive to the type of opponent being played and to the length of the match. A very interesting result is the fact that strategies which are identically valued under one standard measure (worst-case winning rate) can have very different exploration values. Finally, in this small game explicit modelling is superior to implicit modelling; however, this does not mean that implicit modelling research should be abandoned, as there are indications that implicit modelling might be a better choice for large games.

1.4 Outline

The thesis proceeds as follows. Chapter 2 is focused on providing background material, which includes defining terminology essential to the thesis. This chapter also describes the game which is the testbed of the methods described, Kuhn Poker, and some of the properties which make this game interesting.

Chapter 3 describes how an explicit model of an opponent can be created, and Chapter 4 demonstrates the effectiveness of the explicit modelling techniques with different methods of collecting data about the opponent. Chapter 5 examines the technique of implicitly modelling an opponent, and shows how existing algorithms can be adapted and improved.

Chapter 6 describes other research activities which are strongly related to the research described here. This includes other studies done on poker and other imperfect information games, opponent modelling in a variety of settings, and other related work.

Chapter 7 concludes the thesis, summarizing the findings and possible future directions of this research.

Chapter 2

Essential Background

2.1 Game Theory Definitions

The purpose of this thesis is to investigate the usefulness of different opponent modelling methods. One may wonder why opponent modelling is useful at all if game theoretic solutions exist for the games under consideration (ie. what's left to do after solving the game?). The fundamentals of game theory determined that game theoretic solutions will achieve the highest expected payoff rate *that can be guaranteed* [40]; what is not as well known is the fact that if a player uses game theoretic solutions then often he will not exploit mistakes made by his opponent, and will limit his potential winnings as well as his losses.

Before going into how the concepts of game theory are very useful in this research, a few key terms must be defined. First of all, a *game* is a process which involves two or more participants (called *players*) who make decisions based on the information available to them, and these decisions affect the outcome of the game. Chance may be a factor both in determining what decisions players face, and in determining the outcome of the game. When a game ends, each player receives a reward (which could be positive or negative), depending on the outcome; generally it is assumed that each player wants to maximize his reward.

This research is primarily concentrated on *two-person zero-sum games.* Zero-sum games are games in which the sum of the rewards given to the players at the end of the game is zero. Since the rewards sum to zero, one player's loss is another's gain, and the players must compete against each other to get higher rewards. In games that aren't zero-sum, a player may be indifferent to what rewards his opponents are achieving as long as he is satisfied with his reward. The reason the research is restricted to two-player games is that as more players are added there is an exponential increase in the number of situations that can occur, as well as the fact that more complex opponent models are needed to describe some players cooperating rather than just competing against each other. When describing two-player games the players will be denoted as P1 and P2.

A key feature of many of the games studied here is that each player will have private information (like hole cards in Texas Hold'em) that is not available to the other player during gameplay. The games may also have public information (like the betting sequence or community cards in Texas Hold'em) that is available to all players. This leads to the concept of information sets: an *information set* for P1 [P2] is the set of all possible P1 [P2] decision nodes for which P1 [P2] receives the same information (both public and private), but P2's [P1's] private information is different. P1 [P2] cannot distinguish different elements of an information set during gameplay, and must use the same strategy for each distinct element of the set. Terminal nodes of a game tree for which a player has identical information are also of interest and will be said to be in an *information leaf-set.* Information sets will be denoted $\langle H_1, H_2 : D \rangle$, where H_i is the hand held by the ith player and D is the public information for the game. If a quantity is unknown (most of the time a player knows only his hand and not his opponent's hand) then it will be replaced by a question mark, while the empty sequence (which is usually the situation immediately after the deal before either player has acted) will be represented by ϕ. The terms information set and situation will be used interchangeably in this thesis.

A *strategy* for a player is a complete description of how to play a game; it describes how to choose actions in every information set that could possibly arise. A *pure strategy* is a strategy in which all of the choices are deterministic; every time a specific situation is reached, the action taken is always the same. A *mixed strategy* is a strategy that is a mixture of one or more pure strategies, each played with some nonzero probability. For example, the strategy of playing pure strategy S_A 50% of the time, playing pure strategy S_B 30% of the time and playing pure strategy S_C 20% of the time is a mixed strategy. The use of mixed strategies allows different decisions to be made when a situation is reached repeatedly. The *support* of a mixed strategy S_m is the set of pure strategies which are chosen from with nonzero probability when S_m is used [31].

The *expected value* or *expected payoff-rate* of a strategy S_P used by P against an opponent O using strategy S_O is the average reward P can expect to receive, averaged over the possible outcomes of the game given the two player's strategies and the chance elements of the game. If Z is the set of possible game outcomes and $x(z)$ is the reward given to P if the outcome of the game is z, then the expected reward can be explicitly stated as

$$EV[x|S_P, S_O] = \sum_{z \in Z} P(z|S_P, S_O)x(z).$$

If S_P is a mixed strategy with the pure strategies $S_1, S_2, \ldots, S_n$ being played with probabilities $p_1, p_2, \ldots, p_n$, then

$$EV[x|S_P, S_O] = \sum_{i=1}^{n} p_i EV[x|S_i, S_O].$$

A strategy S_P^* is said to be a *best-response strategy* to the strategy S_O if

$$EV[x|S_P^*, S_O] = \max_S EV[x|S, S_O]$$

where the maximization is over the set of all possible strategies that P could use. The expected value achieved by a best-response strategy is defined to be the *exploitability* of S_O. Suppose that S_O has an exploitability of V_O and a best-response strategy S_P^* is a mixed strategy with the pure strategies being played with probabilities $p_1^*, p_2^*, \ldots, p_n^*$. Then

$$\begin{aligned} EV[x|S_P^*, S_O] &= \sum_{i=1}^{n} p_i^* EV[x|S_i, S_O] \\ &\leq \sum_{i=1}^{n} p_i^* V_O \qquad (V_O \text{ is the maximum payoff-rate against } S_O) \\ &= V_O \qquad\qquad (\text{probabilities sum to } 1). \end{aligned}$$

In order for equality to hold in the second line it must be the case that each pure strategy in the support of S_P^* achieves the payoff-rate of V_O against S_O. Equality must hold since S_P^* is a best-response strategy. Thus each of the supporting pure strategies are best-response strategies to S_O themselves. A consequence of this result is that to find the exploitability of a strategy S_O and a corresponding best-response strategy, it is sufficient to determine the payoff-rate of each of P's pure strategies.

A strategy S_1 is *dominated* by another strategy S_2, if the expected payoff rate of S_2 is at least as high as the expected payoff rate of S_1 for every possible strategy the opponent could use, and is higher for some opponent strategies. It is said that S_2 strongly dominates S_1 if S_2 has a strictly greater expected payoff against every opponent strategy, and S_2 weakly dominates S_1 if there are opponent strategies against which both have the same expected payoff. One of the first simplifying steps generally made when analyzing games is to assume that neither player will play any easily identified dominated strategies (since the dominating strategy has a higher expected payoff, the dominating strategy should always be substituted in place of the dominated strategy); thus, one of the first steps when analyzing a game is to identify dominated strategies and remove them from consideration. This process of removing dominated strategies is iterative: when player P considers his opponent's complete strategy space, P's strategy S_A may not be better than another strategy S_B against all opponent strategies, but when P eliminates his opponent's dominated strategies S_A may dominate S_B

over this smaller set of opponent strategies. Similarly, after P removes dominated strategies from his set of possible strategies, P's opponent may find some strategies are dominated against P's reduced set of strategies.

One of the fundamental contributions to game theory is the *Minimax Theorem*, introduced by John von Neumann [39, 40]. The theorem states that for each two-player zero-sum game there exists a value V (known as the *value of the game*, which is unique) and strategies S_1 and S_2 (which may not be unique) such that if P1 plays the strategy S_1 then no matter what strategy P2 plays, P1's expected reward is at least V; conversely, if P2 plays the strategy S_2, then no matter what strategy P1 employs, P1's expected reward is no more than V. The strategies S_1 and S_2 are often referred to as *optimal* strategies in the literature. However, in order to reduce the confusion that this term may cause, these strategies will be referred to henceforth as *equilibrium strategies.* In zero-sum games the value V is often nonzero, as is the case in the game which will be introduced later in this chapter, Kuhn Poker, whose value is -1/18. This means that in Kuhn Poker P2 has an advantage and that by playing an equilibrium strategy he will win in the long run. To negate this advantage in repeated games, players alternate positions which means each player plays half of the games with an advantage and half of the games at a disadvantage.

It should be noted that in two-player games that are not zero-sum, equilibrium points are defined by a pair of strategies (one for each player) and not all equilibrium points necessarily achieve the same rewards for both players. Furthermore, if (S_1, S_2) and (S_1', S_2') are two equilibrium points, (S_1, S_2') and (S_1', S_2) may not be equilibrium points. In contrast, if (S_1, S_2) and (S_1', S_2') are any two equilibrium points in a zero-sum game, then (S_1, S_2') and (S_1', S_2) must also be equilibrium points; this characteristic of zero-sum games allows equilibrium strategies for each player to be identified independently of the opponent's strategy, rather than as a component of a specific equilibrium point.

If a player uses a strategy that is not an equilibrium strategy, it is said that he is *exploitable* or that he is playing *suboptimally.* As stated above, players in repeated games usually alternate positions so that neither player is always at an advantage brought on by his seat position. This research treats the situations where a player is in P1 position as being independent of the situations where the player is in P2 position, as the opponent being modelled may play very differently when in P2 position than when he is in P1 position. The objective of a player P is to take advantage of his opponent O's mistakes for one or both of the following subproblems: (i) when O is in P1 position, and (ii) when O is in P2 position.

All pure strategies can be partitioned into two categories: *essential strategies* (strategies which are in the support of at least one equilibrium strategy) and *superfluous strategies*

(strategies which are not in the support of any equilibrium strategy) [19]. Dominated strategies make up a subset of the set of superfluous strategies. An important result from the pioneering game theory research done by von Neumann and Morgenstern [40] is that in any two-player zero-sum game, if one player plays an equilibrium strategy and the other player plays any essential strategy (or any mixture of essential strategies), then the expected reward for P1 is V, the value of the game. This interesting result can be seen in the simple zero-sum game of Roshambo, where two players simultaneously choose one of three actions: Rock, Paper or Scissors. If the two players choose the same action they tie (and both receive a reward of 0). Otherwise Rock defeats Scissors, Paper defeats Rock, and Scissors defeats Paper, and the player choosing the winning action receives a payoff of 1. The value of this game is zero, and both players have the same equilibrium strategy, which is to choose each of the three actions 1/3 of the time. Each of the three pure strategies (always-Rock, always-Paper, always-Scissors) are essential strategies as all are components of the equilibrium strategy. Consider what happens when P1 plays the equilibrium strategy and P2 plays the always-Rock pure strategy: 1/3 of the time the players tie (both choose Rock), 1/3 of the time P1 defeats P2 (P1 chooses Paper and P2 chooses Rock), and 1/3 of the time P2 defeats P1 (P1 chooses Scissors and P2 chooses Rock), resulting in an expected winnings of zero for both players. Thus when one player plays the equilibrium strategy in Roshambo, the other player can play any essential strategy without reducing his expected winnings in the game.

Examples of the different types of strategies can be found in the simple matrix game shown below; Max, who chooses a row, wants to maximize his payoff, while Min, who chooses a column, wants to minimize the payoff received by Max. Both players choose their actions (which row and which column to play) simultaneously (thus they both make their decision before knowing the other's), and Max receives the corresponding payoff listed in the matrix.

$$\begin{array}{cccccc} & & & \text{Min} & & \\ & & c_1 & c_2 & c_3 & c_4 \\ \text{Max} & r_1 & 2 & 0 & 2 & 4 \\ & r_2 & 1 & 4 & 2 & 0 \end{array}$$

Min's pure strategy of playing c_3 is dominated by the pure strategy of playing c_1, so c_3 is identified as a dominated column and can be eliminated from consideration. Because Max only has two options, his equilibrium strategy can easily be found graphically. Any strategy for Max can be summarized by one parameter, p, the probability that Max plays row r_1, as he must then play r_2 with probability $1 - p$. Figure 2.1 shows how Min's pure strategies perform against Max for every value of p, as well as the minimum expected payoff that Min can force upon Max if Min knows Max's strategy.

Max wants to obtain the highest expected payoff that he can guarantee; ie. even if Min

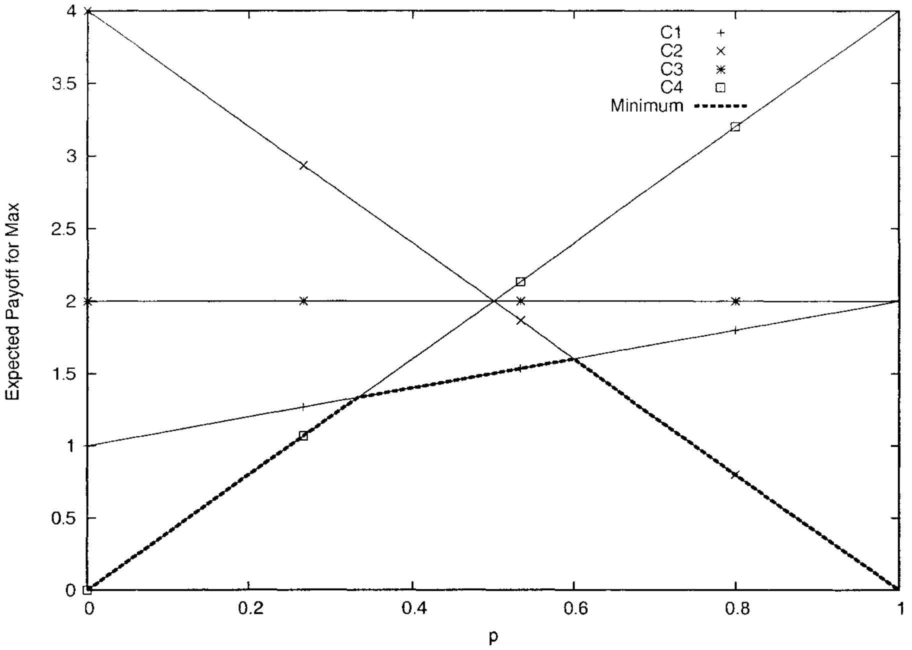


Figure 2.1: Expected Value of Each of Min's Pure Strategies vs. Max's Mixed Strategy with Parameter p

knows Max's strategy, Min cannot reduce the payoff that Max expects. This occurs at the point $p = 0.6$ where the "Minimum" plot peaks at 1.6. Thus Max's equilibrium strategy is to play row r_1 60% of the time and r_2 40% of the time and the value of the game is 1.6. By the Minimax Theorem, any equilibrium strategy for Min will hold Max to the value of 1.6. Note that if Min plays a mixed strategy with a nonzero probability y of playing c_4, then Max's equilibrium strategy of playing r_1 60% will achieve an expected payoff of $1.6 + 0.8y$ which is higher than the value of the game. Therefore, the strategy of playing c_4, which is not a dominated strategy because it is the best counter-strategy for all of Max's strategies on the interval $p < 1/3$, must be superfluous and can be eliminated from consideration when attempting to find Min's equilibrium strategy.

Now that there are only two pure strategies to consider in finding Min's equilibrium strategy, it can also be found graphically. Let q be the probability that Min plays column c_1 and $1 - q$ be the probability that Min plays c_2. Figure 2.2 shows how Max's pure strategies perform against Min for every value of q, as well as the maximum expected payoff that Max can attain if Max knows Min's strategy.

Min wants to limit Max to the lowest expected payoff that he can guarantee, which occurs at the point $q = 0.8$, where the "Maximum" plot is at it's lowest point. Thus there is a unique equilibrium strategy for Min in this game and it is to pick c_1 80% of the time

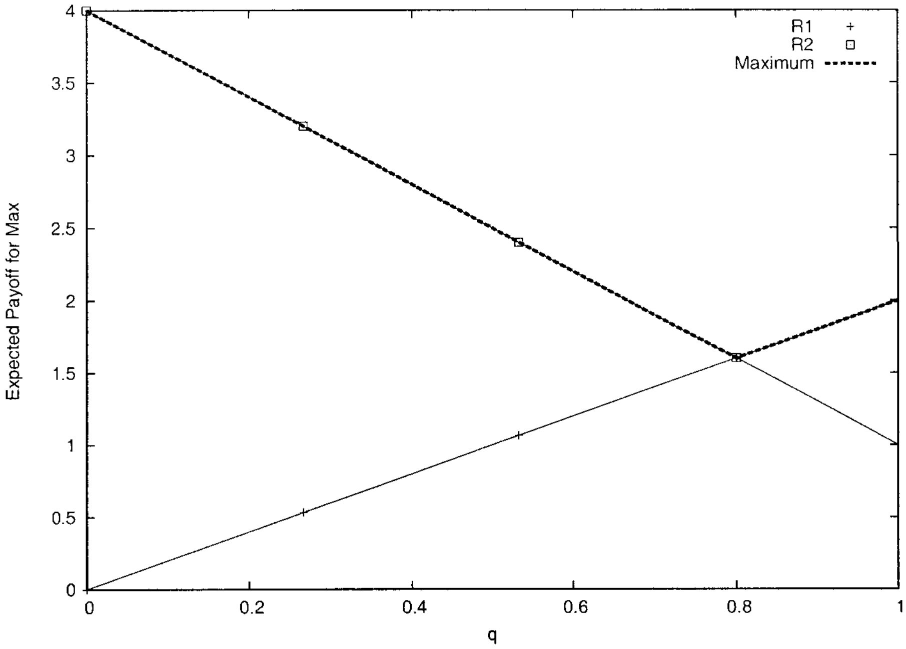


Figure 2.2: Expected Value of Each of Max's Pure Strategies vs. Min's Mixed Strategy with Parameter q .

and pick c_2 20% of the time, resulting in a mixed strategy with an expected payoff rate for Max of 1.6 against both of Max's pure strategies. This example game displays each of the different categories of pure strategies that have been described: the two pure strategies of just playing c_1 and just playing c_2 are essential strategies for Min, while playing c_3 is a dominated strategy, and playing c_4 is a superfluous strategy that is not dominated .

Given these terms, it can now succinctly be described how a player can be exploitable. There are three cases in which a player can play exploitably: (i) the player could use a dominated strategy; (ii) the player could use a non-dominated superfluous strategy; or (iii) the player could play only essential strategies, but in a mixture which does not make up an equilibrium strategy. If player P (in P1 position for the following) uses an equilibrium strategy against a type (i) player O_1, P would expect that O_1's dominated errors will usually allow P to gain a payoff rate higher than V; however, it is also possible that P's equilibrium strategy will never guide O_1 into a situation where he makes dominated errors. Similarly, if P uses an equilibrium strategy against a type (ii) player O_2, P may or may not exploit O_2's errors. Finally, if P uses an equilibrium strategy against a type (iii) player O_3, P will be guaranteed not to exploit O_3's errors and will be assured of receiving the expected payoff of V because of the fact that each essential strategy achieves the expected value V against an equilibrium player; thus any mix of essential strategies achieves the value V against an equilibrium player. Exploitable players can fall into more than one of the above categories.

2.2 Solving Small Games

For completeness, a brief discussion of how basic two-player zero-sum games can be solved is presented here (not all games reduce to two pure strategies for each player and can be solved graphically). First note that a game can be represented by an $m \times n$ payoff matrix A, where m (n) is the number of P1's (P2's) pure strategies, and matrix entry a_{ij} is P1's expected reward when he uses his ith pure strategy and P2 uses his jth pure strategy. A strategy for P1 can now be represented as a probability vector $\vec{x}$ with m non-negative entries that sum to 1 (the ith entry corresponds to the probability of playing the ith pure strategy); similarly, a strategy for P2 can be represented as a probability vector $\vec{y}$ with n non-negative entries that sum to 1. Let X be the set of all possible P1 strategy vectors and Y be the set of all possible P2 strategy vectors. The expected reward for P1 using strategy $\vec{x} \in X$ against P2 using strategy $\vec{y} \in Y$ is:

$$E[r|\vec{x}, \vec{y}] = \vec{x}^T A\vec{y}.$$

where the T represents the transpose operation and matrix multiplication is performed. For P1 to *guarantee* himself as high an expected reward as possible, he needs to find a strategy which gives him the maximum expected reward even if P2 was told P1's strategy in advance (and plays a good counter-strategy). This corresponds to finding a strategy $\vec{x}^*$ that maximizes

$$\min_{\vec{y} \in Y} (\vec{x}^*)^T A\vec{y}.$$

Similarly, an equilibrium strategy $\vec{y}^*$ for P2 is one that minimizes

$$\max_{\vec{x} \in X} \vec{x}^T A\vec{y}^*.$$

Since each of P2's (P1's) essential pure strategies limit P1 to V when P1 (P2) plays an equilibrium strategy, the problems simplify so that $\vec{x}^*$ maximizes

$$\min_{1 \leq j \leq n} (\vec{x}^*)^T A\vec{y}_j,$$

where $\vec{y}_j$ is the vector corresponding to the pure strategy of P2 playing the jth column; $\vec{y}^*$ minimizes

$$\max_{1 \leq i \leq m} \vec{x}_i^T A\vec{y},$$

where $\vec{x}_i$ is the vector corresponding to the pure strategy of P1 playing the ith row.

For sufficiently small games, these maximization/minimization problems can be solved with linear programming techniques, as discussed in [12].

2.3 Regret

The concept of *regret* is used to measure how well an adaptive playing algorithm A is performing against an opponent O, with respect to some set of alternative playing strategies. In contrast to the goal of maximizing the expected payoff against all opponents, which is the goal when solving a game, regret pertains to the particular opponent that is being played against. The term regret used in everyday language refers to a sense of loss that occurs when one takes an action and wishes an alternative action leading to a different outcome had been taken instead. In the context of measuring adaptive algorithms, two types of regret are usually discussed: external and internal regret [2, 18, 20].

External regret describes the maximum amount of reward that has been lost by playing the adaptive algorithm rather than the alternative of playing a fixed pure strategy. Suppose an adaptive algorithm A is used in T consecutive games against O, and the reward $x_A(t)$ is received after game t, for $t = 1, \ldots, T$. External regret at time T, ER_T, is defined as the total of the rewards obtained by A subtracted from the total of the rewards obtained by playing a best-response pure strategy to O:

$$ER_T(A, O) = \max_{s \in \mathcal{S}} \left(\sum_{t=1}^{T} x_s(t) \right) - \left(\sum_{t=1}^{T} x_A(t) \right),$$

where $\mathcal{S}$ is the set of pure strategies for the game being played, and $x_s(t)$ is the reward that would have been obtained if the pure strategy s had been played at time t.

Internal regret considers the alternative of playing a learning algorithm A' which acts identically to A except with regard to some pair of pure strategies, s and s'. Whenever A chooses the pure strategy s, A' instead chooses the pure strategy s'; whatever probability that A chooses to play s is shifted to s' for the algorithm A'. Internal regret at time T, IR_T, is defined as the total of the rewards obtained by A subtracted from the total of the rewards obtained by playing the best alternative algorithm A':

$$IR_T(A, O) = \max_{A'} \left(\sum_{t=1}^{T} x_{A'}(t) \right) - \left(\sum_{t=1}^{T} x_A(t) \right),$$

where the maximization is over all possible pairs of pure strategies s and s'.

This thesis will primarily be concerned with external regret, as test opponents will be using fixed strategies, which means that for each opponent there exists a pure strategy that is a best-response strategy. If an opponent does not use a stationary strategy, then the best-response strategy can change, and minimizing internal regret could be more important.

An algorithm's average regret is defined as

$$\text{Average}R_T(A) = \frac{1}{T} R_T(A).$$

A property that is desired for most adaptive algorithms is that the average regret converges to zero [9],

$$\lim_{T \to \infty} \frac{1}{T} R_T(A) = 0.$$

If an algorithm's average external regret converges to zero, then it must be the case that in the long term, the algorithm is almost always playing a best-response strategy to the opponent.

2.4 Kuhn Poker

The testbed used for the majority of this research is the tiny game of Kuhn Poker [26], as it is easily analyzed and is small enough that the effects of the opponent modelling methods being used are not blurred by the sheer size of the game. A discussion of how the methods can be applied to larger games is given in Chapter 7.

Kuhn Poker is a simple two-player poker game introduced and solved by H.W. Kuhn in 1950 [26]. The game is played with a three-card deck with the cards (in order from lowest to highest rank) Jack, Queen, and King. The structure of the game (also shown in Figure 2.3) is as follows:

- Each player pays an ante of $1.
- Each player is dealt a card and the remaining card is unseen by either player.
- P1 is now given the opportunity to bet $1 or pass.
  - If P1 passes in Round One, then in Round Two P2 can:
    - pass, in which case the game ends immediately in a showdown; or
    - bet, in which case there is a third round where P1 is given the option to bet and take the game to a showdown, or pass and forfeit the pot.
  - If P1 bets in Round One, then in Round Two P2 can:
    - pass (folding), in which case P1 wins the pot uncontested; or
    - bet (calling P1's bet) in which case the game ends immediately in a showdown.

In the event of a showdown, both players reveal their cards and the player with the highest card wins the pot. In this thesis the original notation presented by Kuhn will be followed in all figures and bet sequences pertaining to Kuhn Poker. Thus at every decision node the available actions will be to bet or pass. However, when discussing examples from Kuhn

Poker and other games, conventional poker terms will be used; a bet that matches a previous bet will be referred to as a *call* and a pass when facing a bet will be referred to as a *fold*.

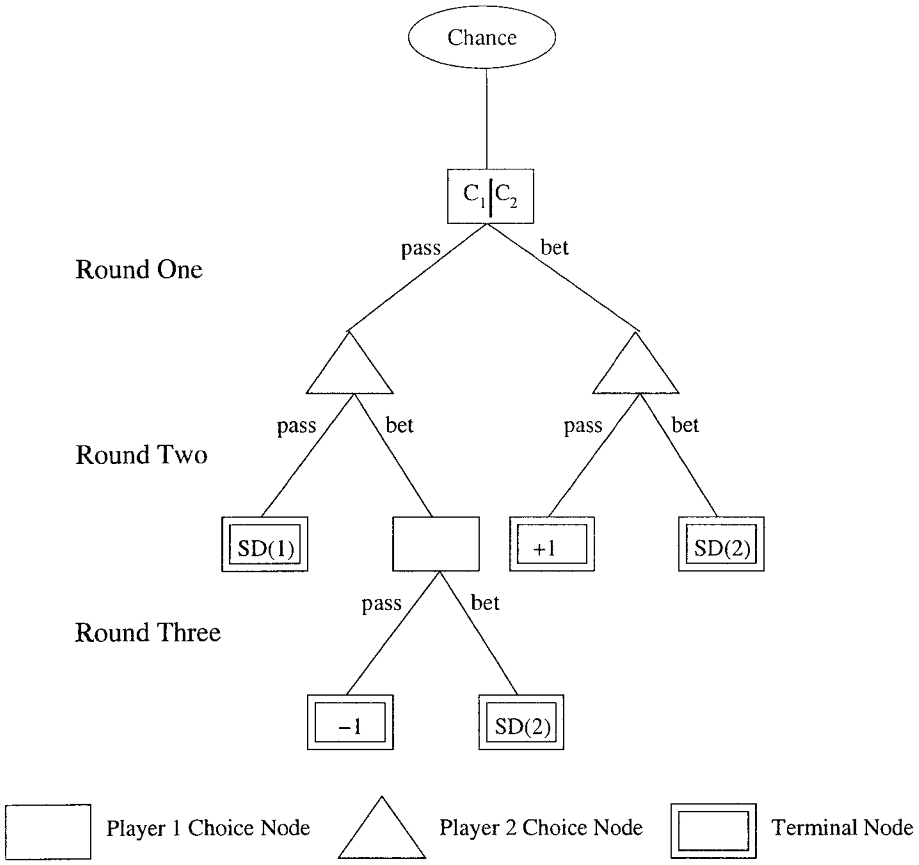


Figure 2.3: Game Structure of Kuhn Poker (SD(x) = Showdown, player with highest card wins x)

The complete P1 strategy space is defined by three possible pure strategies for each card: bet in Round One, pass in Round One and fold if P2 bets, or pass in Round One and call if P2 bets. This gives rise to 3 × 3 × 3 = 27 pure strategies for P1. For P2, there are four pure strategies for each card (two possible actions for each of the two information sets means there are four possible combinations of actions), creating 4 × 4 × 4 = 64 pure strategies in total. This is a large number of strategies for such a simple game, but fortunately there are several dominated strategies that can be removed. First of all, neither player should ever call a bet when holding the Jack, as they are sure to lose; conversely, both players should always call bets when holding the King, as they are sure to win. The P2 strategy of passing when holding the Queen and faced with a P1 pass weakly dominates the strategy of betting in the same scenario. Similarly, the P1 strategy of betting in the first round when holding the Queen is weakly dominated by the strategy to pass and call a P2 bet (if such a bet is made).

Once all dominated strategies are removed, P1's strategy can be summarized by three parameters, α, β, and γ, while P2's strategy can be summarized by the parameters η and ξ:

α = probability that P1 bets in Round One when holding the J

β = probability that P1 bets (calls) in Round Three when holding the Q

γ = probability that P1 bets in Round One when holding the K

η = probability that P2 bets (bluffing) in Round Two when holding the J and P1 passed in Round One

ξ = probability that P2 bets (calls) after a P1 bet, when P2 holds the Q

Figure 2.4 shows the game tree for the game with dominated strategies removed; P1's nontrivial information sets (sets that have more than one element) and information leaf-sets are marked with the letters A to F, while P2's nontrivial information sets and leaf-sets are marked with the letters R to W.

In later chapters, information sets in Kuhn Poker will be referred to by the card(s) known and the betting sequence. For example, the P1 information set labelled A in Figure 2.4 will be known as $\langle J, ? : \phi\rangle$ while the P1 information set F will be notated $\langle K, ? : bp\rangle$, and the P2 information set V (which should not be confused with P1's set F) will be notated $\langle ?, Q : bp\rangle$.

Assuming neither player plays dominated strategies, the expected payoff rate (in \$/hand) for P1 is

$$EV = \frac{1}{6}[\eta(-3\alpha + \gamma) + \xi(-1 + 3\beta - \gamma) + \alpha - \beta] \tag{2.1}$$

Kuhn determined that equilibrium strategies for P1 are of the form $(\alpha, \beta, \gamma) = (\gamma/3, (1+\gamma)/3, \gamma)$ for $0 \leq \gamma \leq 1$. There is one equilibrium strategy for P2: $(\eta, \xi) = (1/3, 1/3)$. The value of the game is $-1/18$ (the game is a loss for P1). One of the main reasons why Kuhn Poker is studied is that the equilibrium strategies contain both bluffing (betting a hand as if it is strong when it is actually weak) and underbidding (not betting a strong hand in order to possibly induce a bet from the other player), which are two interesting components of larger poker games.

Eight pure strategies can be generated from P1's three parameters (the corresponding

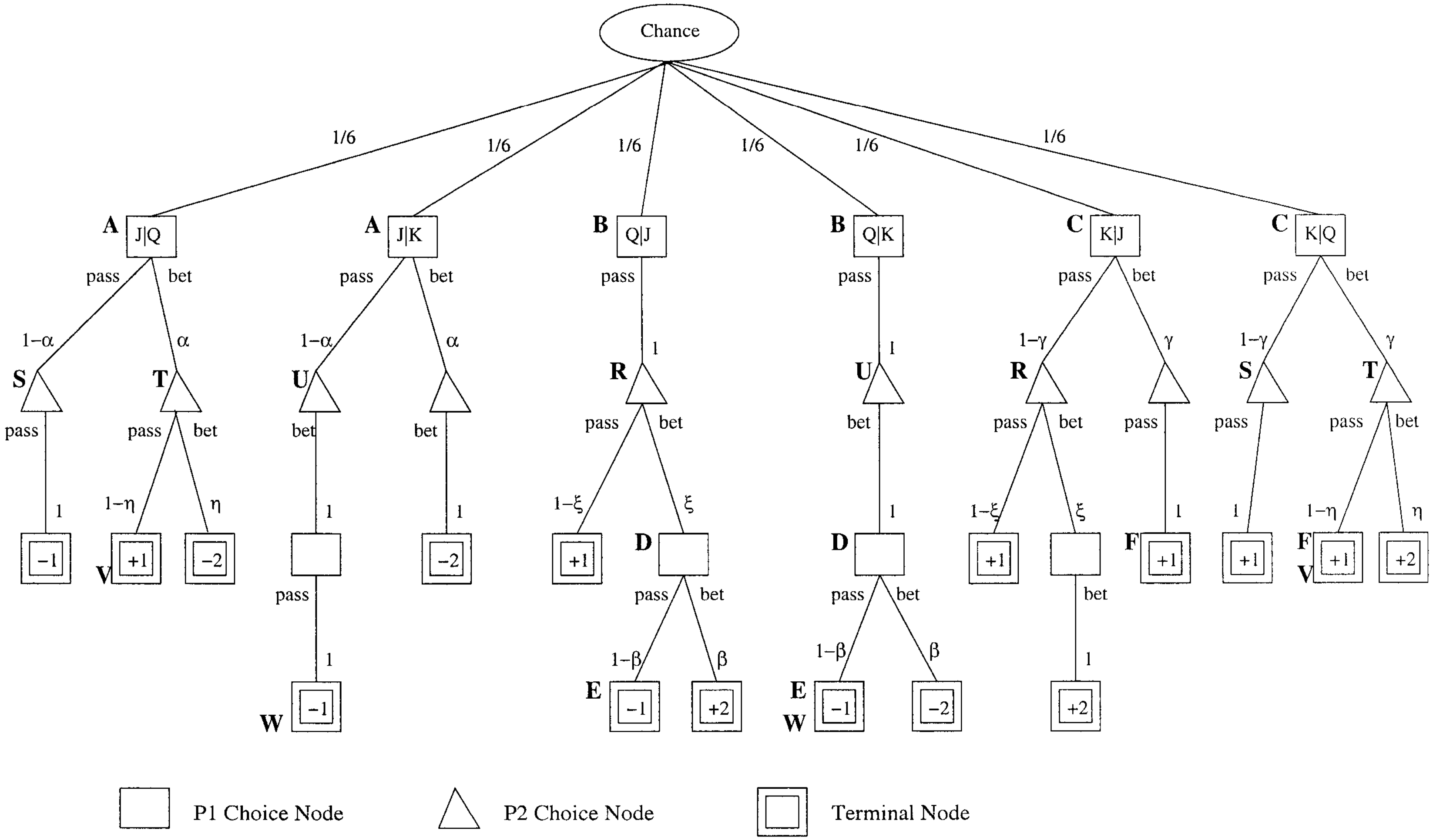


Figure 2.4: Kuhn Poker game tree with dominated strategies removed and information sets labelled

payoff functions are also listed here):

$$\begin{aligned}
S_0 = (0,0,0): &\quad EV_0 = \frac{1}{6}(-\xi) \\
S_1 = (0,0,1): &\quad EV_1 = \frac{1}{6}(-2\xi + \eta) \\
S_2 = (0,1,0): &\quad EV_2 = \frac{1}{6}(-1 + 2\xi) \\
S_3 = (0,1,1): &\quad EV_3 = \frac{1}{6}(-1 + \xi + \eta) \\
S_4 = (1,0,0): &\quad EV_4 = \frac{1}{6}(-3\eta - \xi + 1) \\
S_5 = (1,0,1): &\quad EV_5 = \frac{1}{6}(-2\eta - 2\xi + 1) \\
S_6 = (1,1,0): &\quad EV_6 = \frac{1}{6}(-3\eta + 2\xi) \\
S_7 = (1,1,1): &\quad EV_7 = \frac{1}{6}(-2\eta + \xi)
\end{aligned}$$

The strategy-space for P2 can be partitioned into 6 regions, as seen in Figure 2.5, within each of which a single P1 pure strategy is maximal (on the points which divide the regions, all bordering maximal strategies achieve the same value)[1].

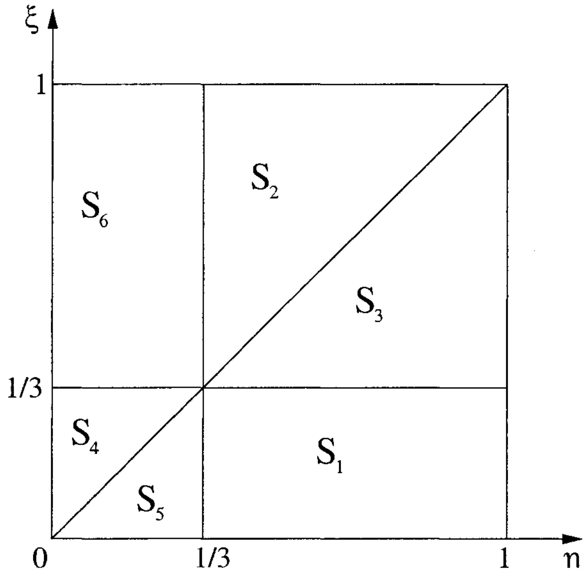


Figure 2.5: Partition of P2 Strategy-space by Maximal P1 Strategies

An example of how this analysis is performed is as follows. First, note that if $\eta > \xi$, then clearly $4\eta > 4\xi$; subtracting $2\xi + 3\eta$ from both sides of this inequality, one obtains

$$-2\xi + \eta > -3\eta + 2\xi,$$

[1] The author would like to thank Valeriy Bulitko for his initial analysis of the game which provided this partitioning.

which shows that $EV_1 > EV_6$ whenever $\eta > \xi$. Similarly, it can be shown over the same region that $EV_3 > EV_2$, $EV_5 > EV_4$ and $EV_1 > EV_0 > EV_7$. Thus the only three strategies that can possibly be best-response strategies in the region where $\eta > \xi$ are S_1, S_3 and S_5. If $\xi < 1/3$ then

$$\begin{aligned} -2\xi + \eta >& -\frac{2}{3} + \eta \\ >&(-1+\xi) + \eta. \end{aligned}$$

Therefore $EV_1 > EV_3$ whenever $\xi < 1/3$. If $\eta > 1/3$, then

$$\begin{aligned} -2\xi + \eta >& -2\xi + \frac{1}{3} \\ >& -2\xi + (1 - 2\eta). \end{aligned}$$

Therefore $EV_1 > EV_5$ whenever $\eta > 1/3$; putting all of this information together, S_1 must be the best counter-strategy for P2 strategies in the region where $1/3 < \eta < 1$ and $0 < \xi < 1/3$. Similar reasoning can be applied to show that the strategies given in Figure 2.5 achieve the highest expected payoff rates within their regions.

Note that there are no non-dominated superfluous strategies in this game; if player P assumes that his opponent O will not play dominated strategies (which are relatively easy to identify and avoid), then the only exploitable category O can fall into is the Type (iii) category (players in this category use essential strategies in a non-equilibrium mixture). Recall that playing an equilibrium strategy against a type (iii) player achieves only the value of the game, which means that P can only hope to exploit O's play if P deviates from playing equilibrium strategies.

In the next chapter, the standard parametrization of Kuhn Poker will be used for an opponent model and different ways of estimating the parameters will be attempted.

Chapter 3

Explicit Modelling in Kuhn Poker

3.1 Introduction

There are two basic types of opponent modelling that are studied in this thesis: explicit modelling and implicit modelling. The first, explicit modelling, is the situation where the modeller tries to infer his opponent's strategy, by observing the opponent's actions in different situations, and then computes a suitable counterstrategy. The second, implicit modelling, is the situation where the modeller simply tries to find a good counterstrategy against his opponent, without trying to identify the opponent's strategy. Explicit modelling is the focus of this chapter and the next chapter, while implicit modelling is studied in Chapter 5.

Two major issues that arise in explicit modelling are deciding how to gather observations about an opponent and deciding how to make use of observations when generating a model. This chapter deals with the second issue, exploring the problem of converting observations of an opponent's actions into estimates of that opponent's strategy-defining parameters; the next chapter discusses how these observations may be gathered and shows the effectiveness of different data-gathering methods.

For this thesis, the strategy used by the player being modelled will be stationary throughout each match. In this chapter explicit models will make single-point estimates of each of the parameters being estimated. Another approach which has been used by others is to maintain a probability distribution for each parameter which identifies the probability of each possible value of the parameter given the observed data [36].

The next section of this chapter will describe how a player can decompose the parameter estimation problem and generate parameter estimates given a set of game observations. The third section describes how to combine solutions of the subproblems to form more reliable estimates. The fourth section of the chapter will show an example of how estimates are computed in Kuhn Poker, and the final section summarizes the parameter estimation method and its limitations.

3.2 Generation of Parameter Estimates

In the preceding chapter, the term strategy was introduced, as a complete description of how to choose actions in every possible information set of the game in question. The parameter model assumes that there is a parameter for each action of each information set, where each parameter represents the probability of choosing the corresponding action when in that information set. The main idea of determining a parameter model for an opponent is to find the most likely set of parameters that would generate the decisions that the modeller has been able to observe.

The problem is defined as follows. Player P wants to estimate the parameters defining his opponent O's strategy after P has been able to observe several of O's decisions in several different situations. Although P may not know O's private information in all cases, P should be able to draw conclusions based on the information he can observe. The approach for estimation will be to simplify the problem as much as possible, by considering one opponent parameter at a time (single-parameter problems), and also to consider the occurrences of this parameter when the modeller's hand is different as separate problems. For example, the problem of modelling P2 in Kuhn Poker will be divided into two problems, estimating η and estimating ξ; the problem of estimating η is further divided into the problems of estimating η when P1 holds the Jack and estimating η when P1 holds the King (the problem of estimating ξ is similarly divided). The modeller will create single-holding parameter estimates in these simpler problems, and then combine these estimates in hopes of forming a better combined-holding estimate for each parameter. Once each parameter has been estimated, the modeller can compute an appropriate counter-strategy to use against the opponent.

It is assumed that the player doing the modelling has a perfect memory of everything that has occurred in his match against his opponent. This is the concept of *perfect recall*, as defined by Kuhn: "each player is allowed by the rules of the game to remember everything he knew at previous moves and all of his choices at those moves" [25]. Thus this research assumes that the modeller is able to use all of the information from his match with his opponent, without forgetting any of the decisions he has been able to observe.

Before further discussion of how explicit modelling is performed can proceed, some details of the game must be clarified. For simplicity, assume that each player has two possible options at each decision node $\mathcal{D}$, which will be denoted $L_{\mathcal{D}}$ and $R_{\mathcal{D}}$; any game G can be converted to another game G' with this binary property, as a decision in G where there are $n > 2$ possible actions $(a_1, \ldots, a_n)$ can be represented by a sequence of $n-1$ decisions in G': for $j < n-1$, the options for the jth decision are to (i) use action a_j, or (ii) proceed to decision $j+1$. The options for the last decision are to (i) use action a_{n-1}, or (ii) use action a_n. Since there are only two options at each decision, only one parameter needs to be estimated as the second parameter can be computed from the fact that parameters at each decision must sum to one.

Although O's parameters may be interdependent, P can estimate each of O's parameters individually; thus the problem discussed here will be how P can estimate a single parameter, $\alpha_{i_o,D}$, which denotes the probability that O will take action L when holding hand H_{i_o} and the sequence preceding the decision is D. For this research, H_{i_o} will represent all information in the game that is private to O, while D represents the information public to both P and O, consisting of L and R actions possibly interspersed with chance events. Suppose also that before P takes his observations into consideration, P has an initial estimate of $\alpha_{i_o,D} = \hat{b}_{i_o,D} \in [0,1]$. The strength of this estimate (which will determine how much influence this initial estimate has on later estimates of $\alpha_{i_o,D}$) will be denoted $w_{i_o,D} \geq 0$. This strength is essentially the amount of fictitious data that is incorporated into the final parameter estimate (eg. an initial estimate of 0.5 with a weight of 2 will have the effect of pretending that the modeller observed his opponent in the situation two extra times and that in one of those times the opponent took the L action).

Let N denote the number of times that O is in the information set $\langle ?, H_{i_o} : D\rangle$. In the long run, it is expected that the number of times that O takes action L in this situation, N_L, will be approximately $N * \alpha_{i_o,D}$; thus a good approximation of $\alpha_{i_o,D}$ is N_L/N. If P could identify O's hand every time that O holds the hand H_{i_o} and the sequence D occurs, then P's estimate of $\alpha_{i_o,D}$, incorporating in the initial estimate, would be

$$\hat{\alpha}_{i_o,D} = \frac{N_L + \hat{b}_{i_o,D}\, w_{i_o,D}}{N + w_{i_o,D}}$$

What makes the problem difficult is that P often does not know what O holds (for example, in poker one of the two players could fold, which means they do not get to see each others' hands). Another issue that should be considered is that the hand held by P may affect his observations, as he may play differently, making the sequence D and showdowns after O's action more or less likely. In addition, P knows that O does not hold the cards in P's hand. To address this issue, P can separately estimate $\alpha_{i_o,D}$ for each possible hand H_j that P can hold (suppose there are $\mathcal{J}$ such hands), and combine these estimates to form a more accurate estimate afterwards. These single-holding estimators, which will be denoted

$\hat{\alpha}_{i_o}^{(1)}, \hat{\alpha}_{i_o}^{(2)}, \ldots, \hat{\alpha}_{i_o}^{(\mathcal{J})}$ (the preceding sequence D will be assumed a constant for the remainder of this chapter) will fall into one of two categories: *complete-information estimators* or *partial-information estimators*.

In the following paragraphs, the notation $\text{COUNT}\langle H_j, H_i : D_1\rangle$ will represent the actual number of times that this situation has occurred in the match between P and O, although this quantity may be unknown to one or both players. The quantity $\widetilde{\text{COUNT}}\langle H_j, H_i : D_1\rangle$ is an estimate by the modeller of the actual number. Finally, the quantity $\text{count}\langle H_j, H_i : D_1\rangle$ will represent the number of times the modeller has observed the occurrence of the information set. In the event that the modeller is able to observe the information set every time it occurs, the upper-case notation will be used. Whenever these quantities are used in formulas in this chapter, the quantities are to be computed from the modeller's observations.

3.2.1 Complete-Information Estimators

A complete information estimator, $\hat{\alpha}_{i_o}^{(j_p)}$, is an estimator derived from an information set $\langle H_{j_p}, ? : D\rangle$, where P finds out O's hand every time that O holds the hand H_{i_o} (no matter what the ensuing bet sequence is). In this case, the single-hand estimate $\hat{\alpha}_{i_o}^{(j_p)}$ is

$$\hat{\alpha}_{i_o}^{(j_p)} = \frac{\text{COUNT}\langle H_{j_p}, H_{i_o} : DL\rangle \, + \, \hat{b}_{i_o} w_{i_o,D}/\mathcal{J}}{\text{COUNT}\langle H_{j_p}, H_{i_o} : DL\rangle \, + \, \text{COUNT}\langle H_{j_p}, H_{i_o} : DR\rangle \, + \, w_{i_o,D}/\mathcal{J}} \tag{3.1}$$

The notation Da (where a is either L or R) represents the sequence D followed by the action a. The strength of the initial estimate in each single-holding estimate has been scaled down (divided by $\mathcal{J}$) so that when the single-holding estimates are combined, the initial estimate will have the proper impact on the combined-holding estimates, as will be seen in Section 3.3.

An example of a complete-information estimator in Kuhn Poker occurs in the case when P1 estimates P2's η parameter when P1 holds the Jack. Figure 3.1 shows the portion of the Kuhn Poker game tree where P1 holds the Jack and bets in Round One. Terminal nodes are shaded to represent how transparent they are to the two players; terminal nodes that are unshaded on the left side are transparent to P1 (P1 knows what card P2 held), and those transparent on the right side are transparent to P2 (P2 knows what card P1 held). In this case each terminal node is transparent to P1 because if P1 bets in Round One with the Jack and P2 calls, then P1 will get to see P2's card (which may be the Queen or the King) in the showdown; if P2 folds after P1 bets, then P1 can deduce that P2 held the Queen since it is a dominated action to fold the King. Since each of the terminal nodes is transparent to P1, he has complete information in this case.

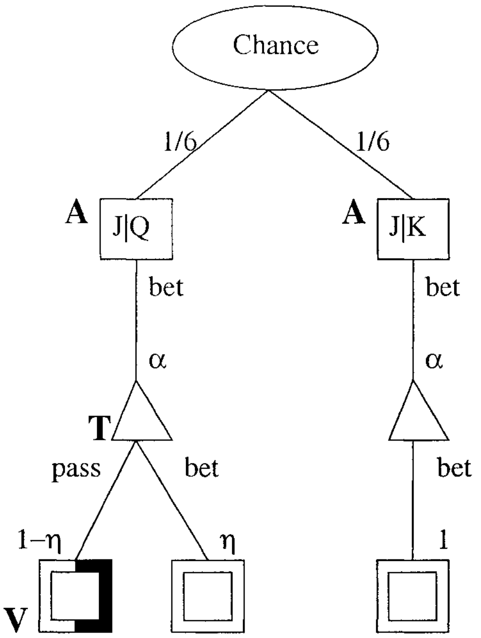


Figure 3.1: Kuhn Poker - P1 Holds the Jack and Bets in Round One

3.2.2 Partial-Information Estimators

In most poker games, many hands end with a fold, and a player is often not able to conclude what hand the other held. In Kuhn Poker, if P1 uses the equilibrium strategy corresponding to $\gamma = 0.5$ and P2 uses his equilibrium strategy, then the proportion of hands that end with a fold is $11/27 = 0.4074$. Despite the lack of complete information, a player should be able to use the available information (from hands that do have showdowns and from actions observed in other hands) to create useful parameter estimates. The general idea is to estimate the unknown quantities and then apply Equation 3.1. In the cases discussed below, it is often assumed that showdown information is known about the L action but not the R action; however, all of the methods can also be applied in the reverse cases, where showdown information is known about the R action and not the L action.

One of the assumptions that is common to several of the following cases is that P has an estimate of the probability that O holds H_{i_o} given that the preceding sequence is D and P holds the hand H_{j_o}. P can compute this probability estimate using a previous model of O's strategy, assuming P also knows the probabilities of all of the chance events involved. The computation is formed by expanding conditional probabilities, following the definition

$$\Pr(A|B) = \frac{\Pr(A,B)}{\Pr(B)},$$

where $\Pr(A|B)$ is the probability that A is true given that B is true, and $\Pr(A, B)$ is the probability that both A and B are true. This substitution is applied repeatedly in the following derivation:

$$\begin{aligned}\Pr(\text{O holds } H_{i_o}|D, \text{ P holds } H_{j_p}) &= \frac{\Pr(D, \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p})}{\Pr(D, \text{ P holds } H_{j_p})}\\ &= \frac{\Pr(D, \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p})}{\sum_{i=1}^{\mathcal{I}} \Pr(D, \text{ O holds } H_i, \text{ P holds } H_{j_p})}\\ &= \frac{\Pr(D| \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p}) \ \ \Pr(\text{O holds } H_{i_o}, \text{ P holds } H_{j_p})}{\sum_{i=1}^{\mathcal{I}} \Pr(D| \text{ O holds } H_i, \text{ P holds } H_{j_p}) \ \ \Pr(\text{O holds } H_i, \text{ P holds } H_{j_p})}.\end{aligned}$$

Note that in the numerator and the denominator in the last line above the quantities on the left-hand side are probabilities of sequences given the players hands and the quantities on the right-hand side are probabilities of chance events. A method to compute the left-hand quantities will be discussed in the following paragraphs, while the right-hand quantities can be computed based on the rules of the game.

Let a_ℓ represent an action chosen by P, b_ℓ represent an action chosen by O, and c_ℓ represent a chance event. Then the probability of the sequence $D = a_1 b_1 c_1 a_2 b_2 c_2 \cdots a_k b_k c_k$ given that the two players' hands are H_j and H_i is the product of the conditional probabilities of each event in the sequence:

$$\begin{aligned}&\Pr(D| \text{ O holds } H_i, \text{ P holds } H_j)\\ &\quad = \prod_{\ell=1}^{k} \Big(\Pr(\text{P takes action } a_\ell|D_{\ell-1}, \text{ P holds } H_j)\\ &\qquad \times \Pr(\text{O takes action } b_\ell|D_{\ell-1}a_\ell, \text{ O holds } H_i)\\ &\qquad \times \Pr(c_\ell|D_{\ell-1}a_\ell b_\ell, \text{ P holds } H_j, \text{ O holds } H_i)\Big)\end{aligned} \tag{3.2}$$

where $D_\ell = a_1 b_1 c_1 \cdots a_\ell b_\ell c_\ell$. Each of the quantities on the right-hand side above are either known (P should know his own action probabilities, as well as the probability of chance events), or can be estimated by P from a previous model of O.

To simplify the notation, there will be no null events shown in the sequences in the case when there is no chance event after a pair of player actions or in the event that one player gets fewer actions than the other. For example, in Kuhn Poker the sequence of a P1 pass followed by a P2 bet and a P1 calling bet will be represented by *pbb*.

Another estimate that is needed for some of the following cases is the probability of P eventually identifying O's hand when O holds H_{i_o} and P holds H_{j_p} and the bet sequence is Da. This estimate depends on the terminal nodes following Da where P can identify that O holds H_{i_o} and the probability of reaching these terminal nodes given the two players' hands and strategies. Let Z_{Da} be the set of terminal nodes following Da where P can identify that

O holds H_{i_o}, and for each terminal node $z \in Z_{Da}$ let D_z be the sequence that reaches z. Note that Da will be a subsequence of D_z for every $z \in Z_{Da}$, since Z_{Da} is a set of terminal nodes which descend from the sequence Da. The estimate in question is

$$\begin{aligned}
&\Pr(\text{P observes O holds } H_{i_o} | \text{ O holds } H_{i_o}, \text{P holds } H_{j_p},\ Da) \\
&\qquad = \sum_{z \in Z_{Da}} \Pr(D_z | \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p},\ Da) \\
&\qquad = \sum_{z \in Z_{Da}} \frac{\Pr(D_z | \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p})}{\Pr(Da | \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p})},
\end{aligned}$$

where these probabilities can be computed using Equation (3.2). The second equality holds because

$$\begin{aligned}
\Pr(D_z | \text{ O holds} H_{i_o}, \text{ P holds } H_{j_p}) &= \Pr(D_z, Da | \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p}) \\
&= \Pr(D_z | \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p},\ Da) \\
&\quad \times \Pr(Da | \text{ O holds } H_{i_o}, \text{ P holds } H_{j_p}).
\end{aligned}$$

Here the first equality follows from the fact that the presence of the sequence D_z implies that the subsequence Da must have occurred, and the second equality follows from conditional probability definition.

The following paragraphs discuss how to use the available information in several different partial-information situations.

Complete Information on One Action, No Information on Alternative

This case assumes that P (holding H_{j_p}) can identify O's hand every time that O takes action L after the sequence D, but P cannot identify O's hand when O takes action R when holding H_{i_o}. However, P may be able to observe or infer that O has a hand other than H_{i_o} some of the times that O takes action R with a different holding. Using the method just described, P can compute an estimate of the probability that O holds H_{i_o} given the preceding sequence is D and that P's hand is H_{j_p}.

Suppose P has been in the information set $\langle H_{j_p}, ? : D \rangle$ a total of $\text{COUNT}\langle H_{j_p}, ? : D \rangle$ times, and has observed O take action R a total of $\text{COUNT}\langle H_{j_p}, ? : DR \rangle \geq 0$ times. P has also been able to identify that O did not hold H_{i_o} when taking action R a certain number of times which will be denoted $\text{count}\langle H_{j_p}, \neg H_{i_o} : DR \rangle$. P estimates that the number of times O has held H_{i_o} in the set $\langle H_{j_p}, ? : D \rangle$ as

$$\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : D \rangle = \text{COUNT}\langle H_{j_p}, ? : D \rangle \times \Pr(\text{O holds } H_{i_o} \mid \text{P holds } H_{j_p},\ D). \quad (3.3)$$

P knows $\text{COUNT}\langle H_{j_p}, H_{i_o} : DL \rangle$ (this is not an estimate), since P can identify that O holds H_{i_o} every time that O takes action L with that hand. P then computes

$$c = \widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : D \rangle - \text{COUNT}\langle H_{j_p}, H_{i_o} : DL \rangle \quad (3.4)$$

and

$$u = \text{COUNT}\langle H_{j_p}, ? : DR\rangle \; - \; \text{count}\langle H_{j_p}, \neg H_{i_o} : DR\rangle \tag{3.5}$$

The quantity c is a candidate answer and u is an upper-bound for the final quantity needed for the parameter estimate, $\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : DR\rangle$. P sets

$$\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : DR\rangle = \begin{cases} 0 & \text{if } c \leq 0 \\ c & \text{if } 0 < c < u \\ u & \text{if } c \geq u \end{cases} \tag{3.6}$$

Now P can compute an estimate $\hat{\alpha}_{i_o}^{(j_p)}$ as in the complete-information case (Equation (3.1))

$$\hat{\alpha}_{i_o}^{(j_p)} = \frac{\text{COUNT}\langle H_{j_p}, H_{i_o} : DL\rangle \; + \; \hat{b}_{i_o,D}\, w_{i_o,D}/J}{\text{COUNT}\langle H_{j_p}, H_{i_o} : DL\rangle \; + \; \widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : DR\rangle \; + \; w_{i_o,D}/J}$$

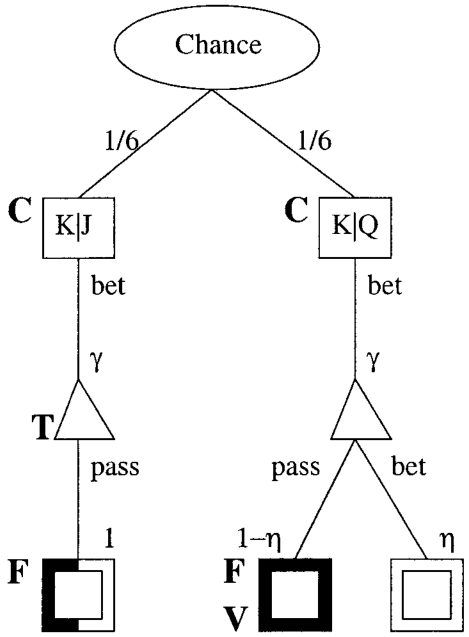


Figure 3.2: Kuhn Poker - P1 Holds the King and Bets in Round One

An example of this situation in Kuhn Poker occurs in the case when P1 estimates P2's η parameter when P1 holds the King; the relevant portion of the game tree is shown in Figure 3.2. If P1 bets in Round One with the King and P2 calls, then P1 will get to see P2's card in the showdown. If P2 folds after P1 bets, then P1 does not know whether P2 held the Queen or the Jack. However, P1 knows that it was equally likely that P2 could have held the Queen or the Jack after the deal and P1's bet in Round One does not change the probabilities. P1 can use this information to estimate the number of times that P2 held the Queen and then estimate the number of times P1 passed with the Queen in Round Two.

Since it is not obvious that the candidate answer c could be negative or larger than the upper-bound u, examples are presented here showing these possibilities. Consider the situation listed above, P1 estimating P2's η parameter when P1 holds the King, and suppose P1 has made the following observations:

$$\text{COUNT}\langle\text{K,?: b}\rangle = 4$$

$$\text{COUNT}\langle\text{K,Q: bb}\rangle = 3$$

$$\text{COUNT}\langle\text{K,?: bp}\rangle = 1.$$

Then

$$\begin{aligned}\widetilde{\text{COUNT}}\langle\text{K,Q: b}\rangle &= \text{COUNT}\langle\text{K,?: b}\rangle \times \Pr(\text{P2 holds Q} \mid \text{P1 holds K}, D = b)\\ &= 4 \times 0.5 = 2\end{aligned}$$

and P1 computes a negative c:

$$\begin{aligned}c &= \widetilde{\text{COUNT}}\langle\text{K,Q: b}\rangle - \text{COUNT}\langle\text{K,Q: bb}\rangle\\ &= 2 - 3 = -1.\end{aligned}$$

Another example in Kuhn Poker where the modeller has complete information on one action and no information on the alternative occurs when P2 is estimating P1's α parameter when P2 is holding the King. In this situation P2 has complete information when P1 bets in the first round as P2 will always call with the King, but P2 gains no information if P1 passes in Round One and folds in Round Three after P2 bets. Suppose P2 has made the following observations:

$$\text{COUNT}\langle\text{?,K: }\phi\rangle = 6$$

$$\text{COUNT}\langle\text{?,K: b}\rangle = 0$$

$$\text{COUNT}\langle\text{?,K: p}\rangle = 6$$

$$\text{count}\langle\neg\text{J,K: p}\rangle = \text{COUNT}\langle\text{Q,K: pbb}\rangle = 4$$

Then

$$\begin{aligned}\widetilde{\text{COUNT}}\langle\text{J, K: }\phi\rangle &= \text{COUNT}\langle\text{?,K: }\phi\rangle \times \Pr(\text{P1 holds J} \mid \text{P2 holds K}, D = \phi)\\ &= 6 \times 0.5 = 3\end{aligned}$$

and P finds that c is larger than the upper-bound u:

$$\begin{aligned} c &= \widetilde{\text{COUNT}}\langle \text{J,K: } \phi\rangle - \text{COUNT}\langle \text{J, K: b}\rangle \\ &= 3 - 0\ = 3 \end{aligned}$$

$$\begin{aligned} u &= \text{COUNT}\langle \text{?,K: p}\rangle - \text{count}\langle \neg\text{J, K: p}\rangle \\ &= 6 - 4\ = 2. \end{aligned}$$

Partial Information on One Action, No Information on Alternative

This case assumes that P (holding H_{j_p}) can identify O's hand some of the times that O takes action L after the sequence D, but P can never identify O's hand when O takes action R. P also has an estimate of the probability that O holds H_{i_o} given the preceding sequence is D and P's hand is H_{j_p}, and P has an estimate of the probability that he gets to observe that O holds H_{i_o} when O takes action L.

Given that P has been able to observe that O held the hand H_{i_o} and took action L a total of count$\langle H_{j_p}, H_{i_o} : DL\rangle$ times, P estimates the actual number of times that O held H_{i_o} and took action L as:

$$\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : DL\rangle = \frac{\text{count}\langle H_{j_p}, H_{i_o} : DL\rangle}{\Pr(\text{P observes O holds } H_{i_o} \mid \text{P holds } H_{j_p},\ DL)}. \tag{3.7}$$

P computes the estimates $\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : D\rangle$, $\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : DR\rangle$ and finally $\hat{\alpha}_{i_o}^{(j_p)}$ as was done for the previous case (complete information on one action, no information on alternative) with the above estimate replacing COUNT$\langle H_{j_p}, H_{i_o} : DL\rangle$ in all of the relevant formulas. There are no examples of this case in Kuhn Poker.

Partial/Complete Information on One Action, Partial Information on the Alternative

This case assumes that P (holding H_{j_p}) can identify O's hand some or all of the times that O takes action L and some of the times that O takes action R after the sequence D. P also has an estimate of the probability that O holds H_{i_o} given the preceding sequence is D and P's hand is H_{j_p}, and P has estimates of the probabilities that he gets to observe that O holds H_{i_o} when O takes action L and when O takes action R.

There are two methods to deal with this case; the first method is to estimate both quantities, COUNT$\langle H_{j_p}, H_{i_o} : DL\rangle$ and COUNT$\langle H_{j_p}, H_{i_o} : DR\rangle$ using Equation (3.7). This method should be used if the probability of observing O's hand is high for both of his possible actions.

The second method is to estimate one of the quantities, COUNT$\langle H_{j_p}, H_{i_o} : DL\rangle$ or COUNT$\langle H_{j_p}, H_{i_o} : DR\rangle$, using Equation (3.7) and estimate the other using Equations (3.3), (3.4), (3.5), and (3.6). This method should be used if the probability of observing O's hand is low for one or both of his possible actions. It is possible to switch between these two methods of estimating the quantities if P finds that the probability of observing O's hand is very different from what he initially anticipated.

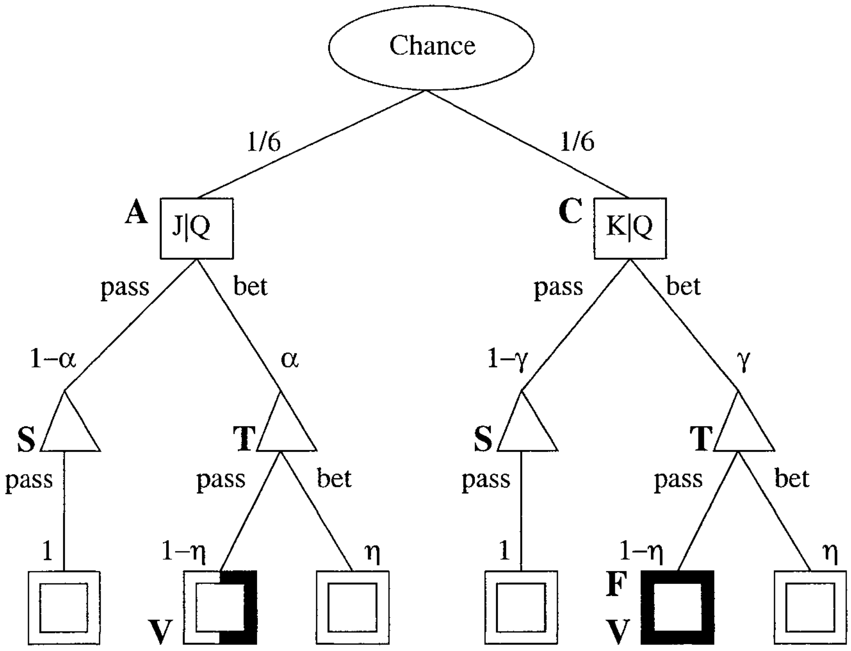


Figure 3.3: Kuhn Poker - P2 Holds the Queen

An example of this case in Kuhn Poker occurs when P2 estimates P1's α parameter when P2 holds the Queen, illustrated by Figure 3.3. If P1 passes in Round One, then P2 will also pass and see P1's card in the showdown. If P1 bets in Round One with the Jack, then P2 may observe it if he chooses to call (the safest strategy is to call 1/3 of the time), but will not observe it if he chooses to fold. Thus if P2 calls with frequency η, then P2 will estimate that the number of times P1 has actually bluffed with the Jack when P2 holds the Queen is $1/\eta$ times the number of times P2 has called the bet and observed the Jack. So in this case P2 has complete information on one of P1's actions (passing) and partial information on the other (betting).

No Information on Either Action

This case assumes that P (holding H_{j_p}) can never identify O's hand when O takes either action L or action R after the sequence D; P can only observe the number of times that O takes each action. P does have an estimate of the probability that O holds H_i for each possible i, given the preceding sequence is D and P's hand is H_{j_p}.

One possible way of dealing with the lack of information is to split the observed actions between each possible hand, proportional to the probabilities of O holding each hand; in this situation,

$$\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : DL\rangle = \text{COUNT}\langle H_{j_p}, ? : DL\rangle \times \Pr(\text{O holds } H_{i_o} \mid \text{P holds } H_{j_p},\, D)$$
$$\widetilde{\text{COUNT}}\langle H_{j_p}, H_{i_o} : DR\rangle = \text{COUNT}\langle H_{j_p}, ? : DR\rangle \times \Pr(\text{O holds } H_{i_o} \mid \text{P holds } H_{j_p},\, D).$$

For example, if the information set $\langle H_{j_p}, ? : DL\rangle$ has been observed 30 times, the information set $\langle H_{j_p}, ? : DR\rangle$ has been observed 20 times and the probability that O holds H_1 is 0.2, then P will estimate that O has taken action L with H_1 a total of $30 * 0.2 = 6$ times and taken action R with H_1 a total of $20 * 0.2 = 4$ times. If the weight of the initial estimate is low, then P's estimate $\hat{\alpha}_1^{(j_p)}$ will be close to $6/10 = 3/5$. Suppose further that the probability that O holds H_2 in the same information set is 0.8; then P will estimate that O has held H_2 and taken action L 24 times and taken action R 16 times, giving an estimate $\hat{\alpha}_2^{(j_p)}$ close to $24/40 = 3/5$.

This example illustrates the property of this method that each single-hand estimator $\hat{\alpha}_i^{(j_p)}$ converges to the same value for every i (as the effect of the initial estimates decrease as more observations are made):

$$\hat{\alpha}_i^{(j_p)} \rightarrow \frac{\text{COUNT}\langle H_{j_p}, ? : DL\rangle}{\text{COUNT}\langle H_{j_p}, ? : DL\rangle \,+\, \text{COUNT}\langle H_{j_p}, ? : DR\rangle} \qquad \forall\, i$$

This is problematic, because P expects that O plays strong hands differently from weak hands, but this method returns the same estimate for both types of hands. To counteract this problem of each of the estimates $\hat{\alpha}_i^{(j_p)}$ having the same value, P can use more sophisticated reasoning in splitting the observed actions between the various possible cards that O could have held. One such way is to assume that certain properties that hold in P's initial estimates about O's strategy also hold in the actual strategy that O is using; for example if P's initial estimate of α_{i_1} is five times larger than his initial estimate of α_{i_2}, then P could split the data in such a way that $\hat{\alpha}_{i_1}/\hat{\alpha}_{i_2} = 5$, preserving the ratio which the initial estimates satisfy.

Suppose O can hold one of the hands $H_1, H_2, \ldots, H_{\mathcal{I}}$; the first step is to compute ratios r_i such that P believes that $\alpha_i/\alpha_1 = r_i$:

$$r_i = \frac{\hat{b}_{i,D}}{\hat{b}_{1,D}} \qquad \text{for } i = 1, \ldots, I$$

P now estimates the number of times O has held each card, based on the probability of O holding each card given the sequence S:

$$\widetilde{\mathrm{COUNT}}\langle H_{j_p}, H_i : D\rangle = \mathrm{COUNT}\langle H_{j_p}, ? : D\rangle \times \Pr(\text{O holds } H_i \mid \text{P holds } H_{j_p},\, D) \qquad \forall i.$$

The number of times that P has observed O take action L is the sum of the times that O has taken action L with each card:

$$\begin{aligned}\mathrm{COUNT}\langle H_{j_p}, ? : DL\rangle &\approx \sum_{i=1}^{\mathcal{I}} \widetilde{\mathrm{COUNT}}\langle H_{j_p}, H_i : D\rangle \times \alpha_i \\ &\approx \sum_{i=1}^{\mathcal{I}} \widetilde{\mathrm{COUNT}}\langle H_{j_p}, H_i : D\rangle \times (r_i\,\alpha_1) \\ &= \alpha_1 \sum_{i=1}^{\mathcal{I}} \widetilde{\mathrm{COUNT}}\langle H_{j_p}, H_i : D\rangle \times r_i\end{aligned}$$

Finally P can estimate α_1 by

$$\hat{\alpha}_1 = \frac{\mathrm{COUNT}\langle H_{j_p}, ? : DL\rangle}{\sum_{i=1}^{\mathcal{I}} \widetilde{\mathrm{COUNT}}\langle H_{j_p}, H_i : D\rangle \times r_i}$$

and estimate α_{i_o} by the relationship $\hat{\alpha}_{i_o} = r_{i_o}\,\hat{\alpha}_1$.

This situation does not arise in Kuhn Poker where one player receives no information for both his opponent's actions, but one could conceive of this occurring in games where hand-mucking is allowed. Hand-mucking occurs when a showdown begins and one player shows his hand and the opponent who called sees that he is beaten and concedes the pot without showing his hand. In a game where mucking is allowed, a player with the best possible hand might never get to see his opponent's hand, whether the opponent folded or called the last bet.

3.3 Combining Single-Hand Estimates

The approach to the estimation problem in this chapter began by decomposing the problem, first into the problem of estimating just one of O's parameters, and then further decomposed into the problem of estimating the specific parameter when it occurs in one of P's information sets. It is possible that the parameter occurs in several of P's information sets (once for each hand that P could possibly hold and generate the sequence D). This section describes how P can combine these single-hand estimates to achieve an estimate that is better than each of the individual estimates. Specifically, P wants to find weights $q_{1,i_o}, q_{2,i_o}, \ldots, q_{\mathcal{J},i_o}$ to form an estimate

$$\hat{\alpha}_{i_o} = \sum_{j=1}^{\mathcal{J}} q_{j,i_o}\,\hat{\alpha}_{i_o}^{(j)},$$

where $q_{j,i_o} \geq 0\,\forall j$ and $\sum_j q_{j,i_o} = 1$. There are several possibilities for how P could arrive at these weights q_{j,i_o}. One possibility is that P could give each estimate equal weight, by setting $q_{j,i_o} = 1/\mathcal{J}\,\forall j$, but this does not adjust for the fact that some of the estimates may have been formed with little or no data yet are receiving weight equal to estimates based on much more data. Another possibility is to trust the estimate which has the most datapoints; if the ℓth estimate has the most data, then set $q_{\ell,i_o} = 1$ and $q_{j,i_o} = 0\,\forall j \neq \ell$. However, in this case P ignores all the data used for the other estimates. A third possibility is to give the estimates weights depending on how many datapoints they use; this can be achieved by setting

$$q_{j,i_o} = \frac{\text{COUNT}\langle H_j, H_{i_o} : D\rangle \,+\, w_{i_o,D}/J}{\text{COUNT}\langle *, H_{i_o} : D\rangle \,+\, w_{i_o,D}},$$

where

$$\text{COUNT}\langle *, H_{i_o} : D\rangle = \sum_{j=1}^{J} \text{COUNT}\langle H_j, H_{i_o} : D\rangle$$

This approach results in the following elegant formula for the combined estimate:

$$\hat{\alpha}_{i_o} = \frac{\sum_{j=1}^{J} \text{COUNT}\langle H_j, H_{i_o} : DL\rangle \,+\, b_{i_o,D} w_{i_o,D}}{\sum_{j=1}^{J} \text{COUNT}\langle H_j, H_{i_o} : D\rangle \,+\, w_{i_o,D}} \tag{3.8}$$

This method of combining the single-hand estimates will be the method used for all combined-estimator results shown in this thesis.

Another way to combine the single-hand estimates is to give each estimate a weight in relation to the confidence held in that estimate; in addition to giving estimators that use more datapoints more weight, estimates with complete information would receive greater weight than estimates with partial information. Studying the effectiveness of this approach is a topic for future work.

3.4 Computing Estimates in Kuhn Poker

Figure 3.4 shows hypothetical frequencies for the terminal nodes reached after several Kuhn Poker hands have been played between P1 and P2 (for example, the ⟨J,Q: pp⟩ terminal node has been reached five times). Naturally, neither player knows all of the numbers given; for example, when P1 holds the King and bets in Round One, he does not know that P2 passed four times each with the Queen and the Jack, but P1 does know that P2 passed eight times in total.

To slightly simplify the calculations, suppose that the initial estimates each have zero strength, meaning that they will not factor into the calculations. The estimation problem will first be considered from P1's point of view and then from P2's point of view.

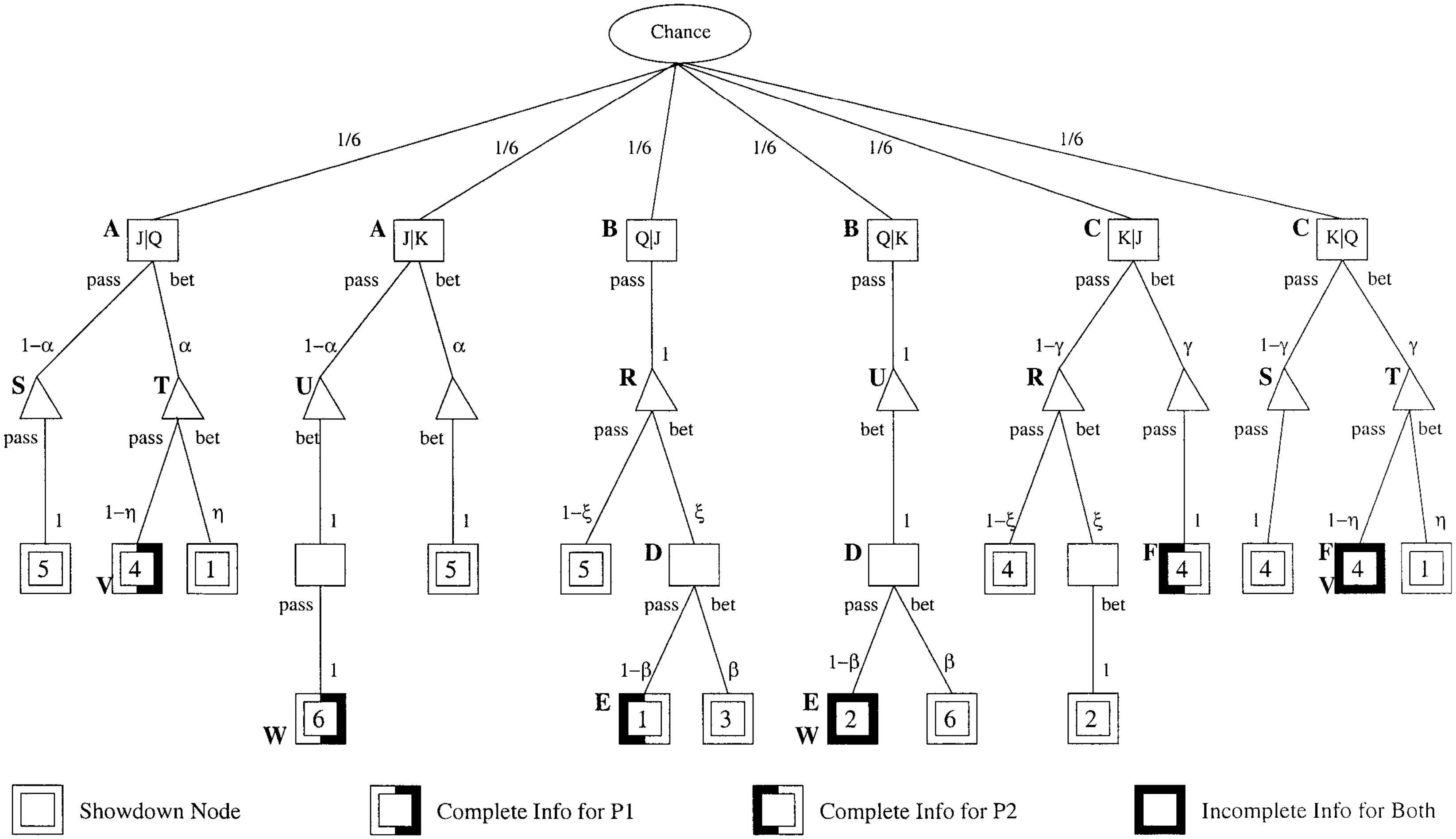


Figure 3.4: Kuhn Poker example (counts of visits to terminal nodes listed)

3.4.1 P1 Modelling P2

Estimating η

The P1 estimator $\hat{\eta}^{(J)}$ (the superscript (J) represents the case where P1 holds the Jack) is an example of a complete-information estimator :

$$\text{COUNT}\langle \text{J,Q: bb}\rangle = 1$$

$$\text{COUNT}\langle \text{J,Q: bp}\rangle = 4$$

and the estimate is

$$\hat{\eta}^{(J)} = \frac{\text{COUNT}\langle \text{J,Q: bb}\rangle}{\text{COUNT}\langle \text{J,Q: bb}\rangle + \text{COUNT}\langle \text{J,Q: bp}\rangle} = \frac{1}{1+4} = \frac{1}{5}$$

The estimator $\hat{\eta}^{(K)}$ is an example of a partial-information estimator for which P1 has complete information on one of P2's actions, as there is a showdown when P2 calls P1's bet, and P1 has no information when P2 folds:

$$\text{COUNT}\langle \text{K,Q: bb}\rangle = 1$$

and

$$\begin{aligned}\widetilde{\text{COUNT}}\langle \text{K,Q: b}\rangle &= \text{COUNT}\langle \text{K,?: b}\rangle \times \Pr(\text{P2 holds Q} \mid \text{P holds K, b})\\ &= 9 \times \left(\frac{1}{2}\right) \quad = \frac{9}{2}\end{aligned}$$

$$\begin{aligned}\widetilde{\text{COUNT}}\langle \text{K,Q: bp}\rangle &= \widetilde{\text{COUNT}}\langle \text{K,Q: b}\rangle - \text{COUNT}\langle \text{K,Q: bb}\rangle\\ &= \frac{9}{2} - 1 \quad = \frac{7}{2}\end{aligned}$$

which gives the estimate

$$\hat{\eta}^{(K)} = \frac{\text{COUNT}\langle \text{K,Q: bb}\rangle}{\text{COUNT}\langle \text{K,Q: bb}\rangle + \widetilde{\text{COUNT}}\langle \text{K,Q: bp}\rangle} = \frac{1}{1+(7/2)} = \frac{2}{9}$$

Combining the two single-hand η estimates gives

$$\hat{\eta} = \left(\frac{5}{5+(9/2)}\right)\frac{1}{5} + \left(\frac{9/2}{5+(9/2)}\right)\frac{2}{9} = \frac{2}{19/2} = \frac{4}{19}.$$

Estimating ξ

The estimator $\hat{\xi}^{(Q)}$ is an example of a partial-information estimator for which P1 has complete information when P2 passes and partial information when P2 bets, as P1 gets to see P2's card if P1 chooses to call in Round Three

$$\text{COUNT}\langle \text{Q,J: pp}\rangle = 5.$$

This particular P1 player has called frequently in Round Three when holding the Queen, which means this player should use Formula (3.7) to estimate the number of times P2 has held the Jack and bet:

$$\widetilde{\text{COUNT}}\langle \text{Q,J: pb}\rangle = \text{count}\langle \text{Q,J: pbb}\rangle / \Pr(\text{P1 bets in Round Three} \mid \text{P1 holds } Q, \text{pb}) = 3/\left(\frac{9}{12}\right) = 4,$$

which gives the estimate

$$\hat{\xi}^{(\text{Q})} = \frac{\widetilde{\text{COUNT}}\langle \text{Q,J: pb}\rangle}{\widetilde{\text{COUNT}}\langle \text{Q,J: pb}\rangle + \text{COUNT}\langle \text{Q,J: pp}\rangle} = \frac{4}{4+5} = \frac{4}{9}$$

The estimator $\hat{\xi}^{(\text{K})}$ is an example of a complete-information estimator as P1 will always call with the King when P2 bets and the game goes directly to a showdown when P2 passes:

$$\text{COUNT}\langle \text{K,J: pb}\rangle = \text{COUNT}\langle \text{K,J: pbb}\rangle = 2$$

$$\text{COUNT}\langle \text{K,J: pp}\rangle = 4,$$

which gives the estimate

$$\hat{\xi}^{(\text{K})} = \frac{\text{COUNT}\langle \text{K,J: pb}\rangle}{\text{COUNT}\langle \text{K,J: pb}\rangle + \text{COUNT}\langle \text{K,J: pp}\rangle} = \frac{2}{2+4} = \frac{1}{3}$$

Combining the two single-hand ξ estimates gives

$$\hat{\xi} = \left(\frac{9}{9+6}\right)\frac{4}{9} + \left(\frac{6}{9+6}\right)\frac{1}{3} = \frac{6}{15} = \frac{2}{5}.$$

3.4.2 P2 Modelling P1

Estimating α

The estimator $\hat{\alpha}^{(\text{Q})}$ (since P2 is now the modeller, the superscript (Q) now represents P2 holding the Queen) is an example of a partial-information estimator for which P2 has complete information when P1 passes (P2 will pass with the Queen and the game goes to a showdown) and partial information when P1 bets, as P2 can choose whether or not to call:

$$\text{COUNT}\langle \text{J,Q: p}\rangle = \text{COUNT}\langle \text{J,Q: pp}\rangle = 5;$$

since this particular P2 has called with a frequency of 0.2 (twice in 10 opportunities) when holding the Queen, he uses the methods discussed for the case of having complete information on one action and no information on the alternative to estimate COUNT⟨J,Q: b⟩:

$$\widetilde{\text{COUNT}}\langle \text{J,Q: } \phi\rangle = \frac{1}{2}\text{COUNT}\langle \text{?,Q: } \phi\rangle = \frac{1}{2}(19)$$

$$\widetilde{\text{COUNT}}\langle \text{J,Q: b}\rangle = \widetilde{\text{COUNT}}\langle \text{J,Q: } \phi\rangle - \text{COUNT}\langle \text{J,Q: p}\rangle = \frac{19}{2} - 5 = \frac{9}{2}.$$

This gives the estimate

$$\hat{\alpha}^{(\mathrm{Q})} = \frac{\widetilde{\text{COUNT}}\langle\text{J,Q: b}\rangle}{\widetilde{\text{COUNT}}\langle\text{J,Q: b}\rangle\ +\ \text{COUNT}\langle\text{J,Q: p}\rangle} = \frac{9/2}{(9/2)+5} = \frac{9}{19}$$

The estimator $\hat{\alpha}^{(\mathrm{K})}$ is an example of a partial-information estimator for which P2 has complete information on one action (P2 always calls when P1 bets) and no information on the alternative, as P2 always bets in Round Two and P1 will then fold the Jack:

$$\text{COUNT}\langle\text{J,K: b}\rangle = \text{COUNT}\langle\text{J,K: bb}\rangle = 5$$

and

$$\widetilde{\text{COUNT}}\langle\text{J,K: }\phi\rangle = \frac{1}{2}\text{COUNT}\langle\text{?,K: }\phi\rangle = \frac{1}{2}(19)$$

$$\begin{aligned}\widetilde{\text{COUNT}}\langle\text{J,K: p}\rangle &= \widetilde{\text{COUNT}}\langle\text{J,K: }\phi\rangle - \text{COUNT}\langle\text{J,K: b}\rangle\\ &= \frac{19}{2} - 5 = \frac{9}{2},\end{aligned}$$

which gives the estimate

$$\hat{\alpha}^{(\mathrm{K})} = \frac{\text{COUNT}\langle\text{J,K: b}\rangle}{\text{COUNT}\langle\text{J,K: b}\rangle\ +\ \widetilde{\text{COUNT}}\langle\text{J,K: p}\rangle} = \frac{5}{5+(9/2)} = \frac{10}{19}.$$

Combining the single-hand estimates of α gives

$$\hat{\alpha} = \left(\frac{19/2}{(19/2)+(19/2)}\right)\frac{9}{19} + \left(\frac{19/2}{(19/2)+(19/2)}\right)\frac{10}{19} = \frac{19/2}{19} = \frac{1}{2}.$$

Estimating β

The estimator $\hat{\beta}^{(\mathrm{J})}$ is an example of a complete-information estimator as when P1 calls in Round Three there is a showdown, and when P1 folds in Round Three he must have held the Queen:

$$\text{COUNT}\langle\text{Q,J: pbb}\rangle = 3$$

$$\text{COUNT}\langle\text{Q,J: pbp}\rangle = 1,$$

which gives the estimate

$$\hat{\beta}^{(\mathrm{J})} = \frac{\text{COUNT}\langle\text{Q,J: pbb}\rangle}{\text{COUNT}\langle\text{Q,J: pbb}\rangle\ +\ \text{COUNT}\langle\text{Q,J: pbp}\rangle} = \frac{3}{3+1} = \frac{3}{4}$$

The estimator $\hat{\beta}^{(\mathrm{K})}$ is a partial-information estimator for which P2 receives complete information when P1 calls in Round Three and no information when P1 folds:

$$\text{COUNT}\langle\text{Q,K: pbb}\rangle = 6$$

and

$$\widetilde{\text{COUNT}}\langle\text{Q,K: pb}\rangle = \frac{1}{2}\text{COUNT}\langle\text{?,K: }\phi\rangle \quad = \frac{1}{2}(19)$$

$$\begin{aligned}\widetilde{\text{COUNT}}\langle\text{Q,K: pbp}\rangle &= \widetilde{\text{COUNT}}\langle\text{Q,K: pb}\rangle - \text{COUNT}\langle\text{Q,K: pbb}\rangle\\ &= \frac{19}{2} - 6 \quad = \frac{7}{2},\end{aligned}$$

which gives the estimate

$$\hat{\beta}^{(\text{K})} = \frac{\text{COUNT}\langle\text{Q,K: pbb}\rangle}{\text{COUNT}\langle\text{Q,K: pbb}\rangle\ +\ \widetilde{\text{COUNT}}\langle\text{Q,K: pbp}\rangle} \quad = \frac{6}{6+(7/2)} \quad = \frac{12}{19}.$$

Combining the single-hand estimators of β gives

$$\hat{\beta} = \left(\frac{4}{4+(19/2)}\right)\frac{3}{4} + \left(\frac{19/2}{4+(19/2)}\right)\frac{12}{19} \quad = \frac{9}{27/2} \quad = \frac{2}{3}.$$

Estimating γ

The estimator $\hat{\gamma}^{(\text{J})}$ is a complete-information estimator as P2 can deduce that P1 holds the King if he bets in Round One and will see the King in a showdown if P1 passes in Round One:

$$\text{COUNT}\langle\text{K,J: b}\rangle = \text{COUNT}\langle\text{K,J: bp}\rangle \quad = 4$$

$$\text{COUNT}\langle\text{K,J: p}\rangle = \text{COUNT}\langle\text{K,J: pp}\rangle\ +\ \text{COUNT}\langle\text{K,J: pbb}\rangle \quad = 4+2 \quad = 6,$$

which gives the estimate

$$\hat{\gamma}^{(\text{J})} = \frac{\text{COUNT}\langle\text{K,J: b}\rangle}{\text{COUNT}\langle\text{K,J: b}\rangle\ +\ \text{COUNT}\langle\text{K,J: p}\rangle} \quad = \frac{4}{4+6} \quad = \frac{2}{5}$$

The estimator $\hat{\gamma}^{(\text{Q})}$ is a partial-information estimator as P2 will see P1's King in a showdown if P1 passes in Round One, but will only see the King when P1 bets if P2 chooses to call:

$$\text{COUNT}\langle\text{K,Q: p}\rangle = \text{COUNT}\langle\text{K,Q: pp}\rangle \quad = 4;$$

since this particular P2 rarely calls (twice in 10 opportunities) with the Queen, he uses the methods discussed in Section 3.2.2 to estimate COUNT⟨J,Q: b⟩:

$$\widetilde{\text{COUNT}}\langle\text{K,Q: }\phi\rangle = \frac{1}{2}\text{COUNT}\langle\text{?,Q: }\phi\rangle \quad = \frac{1}{2}(19)$$

$$\begin{aligned}\widetilde{\text{COUNT}}\langle\text{K,Q: b}\rangle &= \widetilde{\text{COUNT}}\langle\text{K,Q: }\phi\rangle - \text{COUNT}\langle\text{K,Q: p}\rangle\\ &= \frac{19}{2} - 4 \quad = \frac{11}{2}.\end{aligned}$$

This gives the estimate

$$\hat{\gamma}^{(\mathrm{Q})} = \frac{\widetilde{\text{COUNT}}\langle \text{K,Q: b}\rangle}{\widetilde{\text{COUNT}}\langle \text{K,Q: b}\rangle \;+\; \text{COUNT}\langle \text{K,Q: p}\rangle} = \frac{11/2}{(11/2)+4} = \frac{11}{19}.$$

Combining the single-hand estimators of γ gives

$$\hat{\gamma} = \left(\frac{10}{10+(19/2)}\right)\frac{2}{5} + \left(\frac{19/2}{10+(19/2)}\right)\frac{11}{19} = \frac{19/2}{39/2} = \frac{19}{39}.$$

3.5 Summary

This chapter has given a detailed description of how to generate parameter estimates in order to create an opponent model. Although the techniques presented in the previous sections may seem complex, the basic idea is quite simple and is repeated here. The first step is to decompose the whole modelling problem into single-hand single-parameter problems. The next step is to use partial information and probabilities (from game properties and a previous model of the opponent) to create fictitious data when the actual data is not known. The third step is to create single-hand estimates of each of the parameters by substituting the data (fictitious if necessary) into the complete information formula, Equation (3.1). The final step is to combine single-hand estimates to achieve more adequate estimates that use all of the available data.

There are several limitations to the parameter estimation approach. The first is that when creating fictitious data, properties of the deal are assumed, such as if two hands are equally likely, then it is assumed that each has actually occurred exactly the same number of times. This assumption holds up in the long run (when equally likely hands have actually been dealt nearly the same amount of times relative to the total number of deals), but may cause inaccurate estimates in the short run. Another assumption that is needed is that the modeller can estimate the quantity $\Pr(D|$ opponent hand$)$. In large games with nontrivial sequences this probability will depend on the strategy used by the opponent, and thus knowledge of the opponent or good initial estimates are necessary for an accurate estimate of this probability. Also, in large games it is expected that data will be sparse, which means the parameter estimation method may not be practical (as most parameters will have no data to estimate them), unless abstractions are done to reduce the number of parameters. These abstractions would likely include unifying parameters (assuming that the opponent plays similar hands identically) and linking parameters (assuming parameters may not be independent).

Another assumption that is made is that the opponent does not play dominated strategies. This can lead to bad estimates and counterstrategies which do not take advantage of the dominated plays by the opponent. This could be counteracted by including the dominated strategies in the model of the opponent with heavily weighted initial estimates that are close to zero. However, this could greatly increase the complexity of the model; if dominated strategies are included in Kuhn Poker, the number of pure strategies available to P2 increases from four to 64 and the number of parameters increases from two to six.

Finally, the methods discussed here do not recognize when the opponent plays systematically in the repeated game (for example, if when put in a particular situation he bets every odd time and passes every even time) or if the opponent sometimes switches between strategies. This research is restricted to more basic opponents, as one of the objectives of this study is to determine how quickly one can accurately estimate a single fixed strategy. Recognizing when an opponent is dynamic and has changed his strategy could involve using pattern recognition ideas (to see if the opponent's recent actions match his previous pattern) and the introduction of history decay into the data (giving recent observations more emphasis in the estimates); this research is left for future work.

Although there are several limitations to the parameter estimation method, most do not seriously hinder its application in Kuhn Poker, and should not prevent the method from being applied in larger games, as long as the models for the opponent are kept simple. The issue of how the methods can be scaled will be discussed in Chapter 7.

Chapter 4

Data-Collection Methods for Explicit Modelling

4.1 Introduction

This chapter continues the study of explicit modelling. While the last chapter described what to do with the data that has been collected about an opponent O, this chapter focuses on the issue of how the modeller P's playing style affects the amount and quality of data collected. Certain strategies used by P will guide the gameplay into situations where he can learn more about one or more of O's parameters. A key result is that in a set of data-collection strategies that are equally exploitable (for example, the set of equilibrium strategies) elements can have vastly different exploration values.

The modeller has three basic options for his playing strategy while collecting data about the opponent being modelled: (i) the modeller can play within the space of equilibrium strategies; (ii) the modeller can play non-equilibrium mixtures of essential strategies (pure strategies which are part of some equilibrium strategy); or (iii) the modeller can play a mixture of both essential and superfluous strategies (dominated strategies and other pure strategies which aren't part of any equilibrium strategy). The first option ensures that the modeller will not leave himself open to exploitation during the data-collection process, while the second and third options may allow the modeller to learn more quickly than if he only used equilibrium strategies. One example of a dominated strategy that could be used to gain more information is calling a bet with the worst hand, to observe what hand the opponent holds in a showdown (even though the modeller is guaranteed to lose in the showdown). Another example is to simply check (pass when there is no bet to call) or call with the best hand rather than raising, also to see what the opponent has in a showdown rather than have the opponent fold and not be able to observe his holding.

This chapter will demonstrate the advantages and disadvantages of using different data-collection strategies in Kuhn Poker. Models will be evaluated assuming that the modeller uses a data-collection strategy for hands 1 to t, during the *exploration phase*, and then stops collecting data. The model is then used to compute a counter-strategy, and this counter-strategy is played from hand $t+1$ onwards, during the *exploitation phase.* Hand t is referred to as the *switching hand*, as it is the hand at which the modeller switches from the exploration phase to the exploitation phase.

This chapter will begin with the problem of P1 modelling P2 in Kuhn Poker, because P1 has the option of more than one equilibrium strategy to use, and the difference in exploration values within this set of strategies is revealing. The problem of P2 modelling P1 will then be discussed, and the idea of using a dominated strategy to learn more about an opponent will be studied.

4.2 P1 Modelling P2 in Kuhn Poker

In Kuhn Poker, if α or γ is large, P1 will bet more often in Round One and have more opportunities to observe whether P2 will call a bet when holding the Q, thus gaining information about η. If β is large or γ is small, P1 will have more opportunities to observe whether P2 bluffs with the J, gaining information about ξ. This leaves two options for P1 to gain information about P2: P1 can try to learn as much as possible about P2 while restricting himself to playing safe equilibrium strategies, or P1 can play exploitable strategies which do more exploration of P2's strategy and thus should learn faster (but possibly at a greater cost).

For this study, the exploration value of five P1 equilibrium strategies will be compared, for the settings $\gamma = \{0, 0.25, 0.5, 0.75, 1\}$ (recall that $\alpha = \gamma/3$ and $\beta = (\gamma + 1)/3$ for equilibrium strategies). In addition, three non-equilibrium "exploratory strategies" will also be evaluated: ExploreEta $= (\alpha = 1, \beta = 1, \gamma = 1)$, which forces P2 into the most situations where the η parameter is used; ExploreXi $= (1, 1, 0)$, which forces P2 into the most situations where ξ is used; and BalancedExplore $= (1, 1, 0.5)$, which will explore both of P2's parameters. These exploratory strategies are more exploitable than the equilibrium strategies. The ExploreEta strategy has a minimum winning rate of -0.333 dollars per hand, the ExploreXi strategy has a minimum winning rate of -0.5 \$/hand, and the BalancedExplore strategy has a minimum winning rate of -0.417 \$/hand. In comparison to the equilibrium data-collection strategies which ensure a winning rate of -0.0556 \$/hand, these exploratory strategies are risky to use, but the tradeoff in terms of information gained may prove to be worth the risk.

Using the minimum winning rates of the exploratory strategies to represent their riskiness is misleading as it is rare that the opponent being modelled would actually be playing a best-response strategy to the data-collection strategy. In such a case the opponent would likely be easy to model since best-response strategies are usually pure strategies, which are generally much easier to identify than mixed strategies. Thus some sense of the average winning rates of the exploratory methods might better describe how risky it is use these strategies in practice. Against randomly chosen opponents that have an exploitability of 0.0556 \$/hand (these opponents will be described in more detail in Section 4.2.1) the ExploreEta strategy achieved an average winning rate of −0.0850 \$/hand, the ExploreXi strategy achieved an average winning rate of −0.0966 \$/hand, and the BalancedExplore strategy achieved an average winning rate of −0.0909 \$/hand in experiments described in this chapter. Note also that the exploratory strategies may actually win more against certain opponents than a safe equilibrium strategy which limits the modeller's winnings as well as his losses.

In the Kuhn Poker experiments shown in this chapter, the players being modelled are all examples of the type (iii) exploitable player described in Chapter 2, meaning they play essential strategies in non-equilibrium mixtures and never play any superfluous strategies. Therefore the use of equilibrium data-collection strategies will each guarantee the equilibrium payoff rate of −0.0556 \$/hand to P1 against each of the players being modelled, while the exploratory strategies may have higher or lower payoff rates against the modelled players. Experiments will be shown illustrating both the risks and inadvertent rewards of using an exploratory strategy.

4.2.1 Experimental Setup

For the experiments in this thesis, the player being modelled will play a static strategy, and the modeller will collect data over the course of a match. For the experiments in this chapter, the modeller's data-collection strategy will also not change during the match. However, after each hand is completed, the modeller's opponent model will be updated and then evaluated. The opponent model is evaluated by computing the best-response strategy to the model and comparing this strategy to the true best-response strategy against the opponent.

Results will be averaged over many trials, where a single trial consists of a 900-hand match between the modeller (always in P1 position) and the opponent (always in P2 position). For each hand in a trial, one of the six possible holdings for the two players is randomly chosen according to the uniform distribution. At each decision node the action taken is randomly selected according to the distribution defined by the acting player's strategy. In every trial in this section, the modeller begins with initial estimates ($\eta = 0.5$, $\xi = 0.5$), each weighted by two fictitious datapoints.

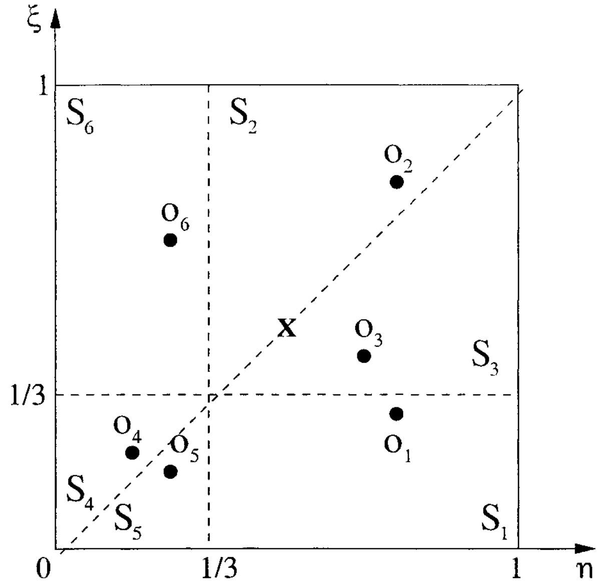


Figure 4.1: P2 Strategies used in Experiments

One type of experiment that is performed in this chapter is to have a modeller use each of the data-collection strategies against six different testpoints, where a testpoint is an ordered pair (η, ξ) that defines a P2 strategy. The testpoints, plotted in Figure 4.1, are $O_1 = (0.8, 0.29), O_2 = (0.75, 0.8), O_3 = (0.67, 0.4), O_4 = (0.17, 0.2), O_5 = (0.25, 0.17)$, and $O_6 = (0.25, 0.67)$. The X in figure 4.1 marks the strategy corresponding to P1's initial estimate $(0.5, 0.5)$. Results for each data-collection strategy and testpoint will be averaged over 30000 trials. Individual testpoints vary in exploitability and other properties which means results between the testpoints cannot be compared directly. Results from this study are used to demonstrate some of the interesting outcomes and side effects that can occur when explicitly modelling an opponent.

A second type of experiment is performed in this chapter in order to demonstrate the average performance of each data-collection method. In this experiment, the modeller faces a randomly generated opponent which has a fixed exploitability. Results are shown for the exploitability settings 0.0556 and 0, and results are averaged over 200000 trials for each data-collection strategy and exploitability setting. Prior to each trial, the opponent is randomly assigned a strategy that loses to its best-response strategy at an expected payoff rate of x \$/hand. Since there are multiple methods of randomly assigning a strategy to the opponent, the process used in this thesis is as follows. The first step is to choose which of

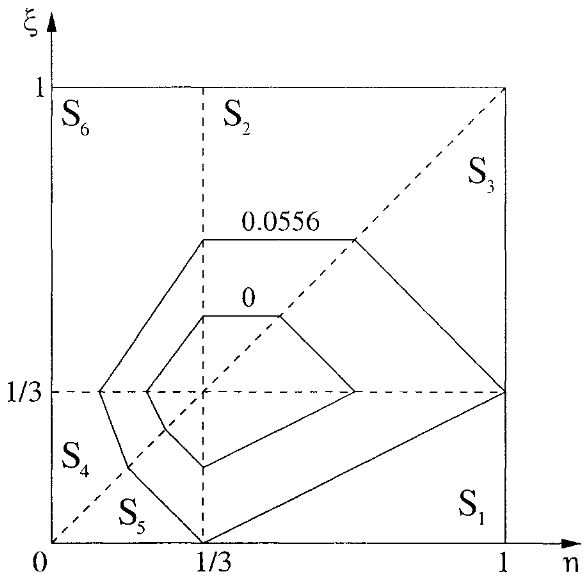


Figure 4.2: Contours of opponent strategies with fixed exploitability

the six regions (see Figure 4.2) the opponent's strategy will lie in, where each region has a probability of 1/6 of being chosen. The selection of region and the exploitability setting determine a line segment ℓ within the region that the opponent's strategy must lie on, where the equation for ℓ is $E(\alpha_{br}, \beta_{br}, \gamma_{br}, \eta, \xi) = x$, where E is the payoff-rate formula given in Equation (2.1) and $(\alpha_{br}, \beta_{br}, \gamma_{br})$ is P1's best-response strategy for the chosen region. A point is uniformly chosen from ℓ to be the strategy used by the opponent. Figure 4.2 shows the possible settings for the opponent strategies for both levels of exploitability. Results from this study illustrate general trends among the data-collection strategies, indicating which strategies should be used when little is known about the opponent.

Experimental results will be shown in three types of plots: *payoff-rate plots*, *total winnings plots*, and *proportion-above-equilibrium plots.* A payoff-rate plot shows the expected payoff-rate the modeller would achieve if he stopped collecting data after t hands (for each of these t hands the modeller is in P1 position) and began playing the best-response strategy to his model. In Kuhn Poker this payoff rate can be computed directly from Equation (2.1) given in Chapter 2. Figure 4.3 is an example of a payoff-rate plot, where the modeller (using the equilibrium data-collection strategy with $\gamma = 1$) tends to quickly learn a model that gives a good counter-strategy (after 200 hands against this opponent the counter-strategies chosen in the different trials achieve an average value of 0.09 \$/hand). However, finding the counter-strategy which gives the maximum value against this opponent in all trials, where the value is shown by the bold dotted horizontal line, can take a very long time. On all of the charts the horizontal axis is labelled "Switching Hand", as the evaluation at hand t assumes that the modeller switches from the exploration phase to the exploitation phase at that hand.

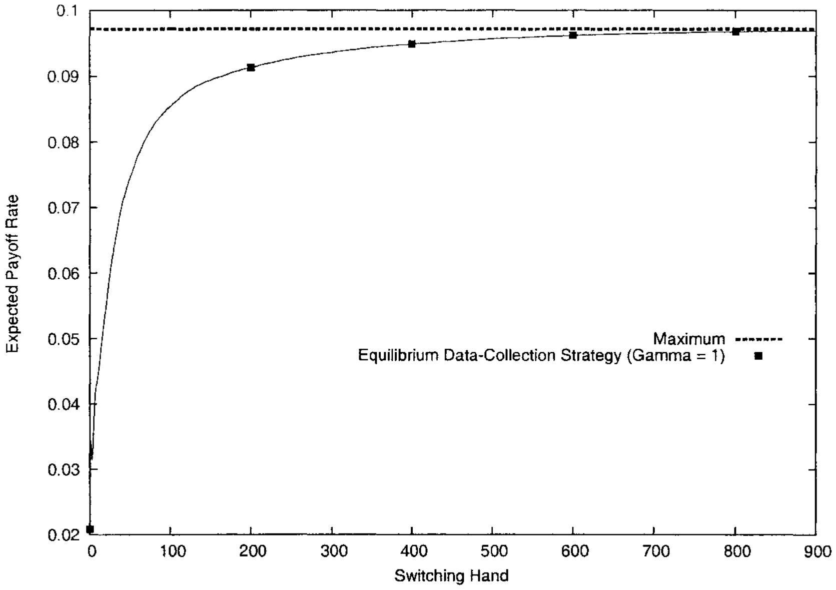


Figure 4.3: Sample Payoff-Rate Plot

An interesting thing to note in this sample payoff-rate plot is that against this opponent the payoff rate at hand 0 is 0.0208 \$/hand. This means that the counter-strategy to the modeller's initial model (created from fictitious data only) happens to be a good strategy against this opponent, as it achieves a much better payoff rate than the equilibrium rate of −0.0556 \$/hand. The counter-strategy to the initial model will be very good against some opponents (particularly opponents who play a strategy similar to the initial estimates) and very bad against others. This means the starting point of the payoff-rate graphs will vary greatly between testpoints.

A major concern of opponent modelling is how expensive the data-collection phase is, and whether anything is gained in the short term by doing opponent modelling. A total winnings plot assumes that the two players are playing a fixed-length match (of length H), and thus the total winnings if the modeller switches to the best-response strategy after hand t can be predicted as the sum of the winnings up until hand t plus the expected winnings of the best-response strategy over the remainder of the match ((payoff-rate)*$(H-t)$). The values on the horizontal axis start at 0, when the modeller has only his initial estimates (based on fictitious data), and end at H, when the modeller has used his data-collection strategy for the entire match.

This type of plot allows one to see if there is an advantage to doing opponent modelling as compared to just playing an equilibrium strategy, and also to identify the best time to switch to the perceived best-response strategy. Figure 4.4, which shows the same data-collection strategy and the same opponent as in Figure 4.3, is an example of a total winnings plot. The plot shows that opponent modelling can be advantageous in a short match, and also shows that the modeller may need to switch to best-response early in the match to make the most of his opponents' errors. Otherwise, as the end of the match gets closer, the modeller has less time to exploit the errors and win a significant amount of money. Even though the model of the opponent continues to get better (as shown by Figure 4.3), the benefits of improving the model are outweighed by the cost of continuing to explore. Total winnings plots tend to have a peak at around hand 40 or 50 when enough information has been learned to exploit the opponent and the plots then steadily decline as the data-collection costs more than the gains made by refining the model. The plots finish at the expected winnings for playing the data-collection strategy for the entire match, as that is exactly what has occurred if the modeller has not made the switch to best-response by then. As in the payoff-rate plots, the start points (the values at Switching Hand 0) of the total winnings plots will also vary greatly between testpoints, as the start point of the total winnings plot is precisely the start point of the corresponding payoff-rate plot multiplied by 200.

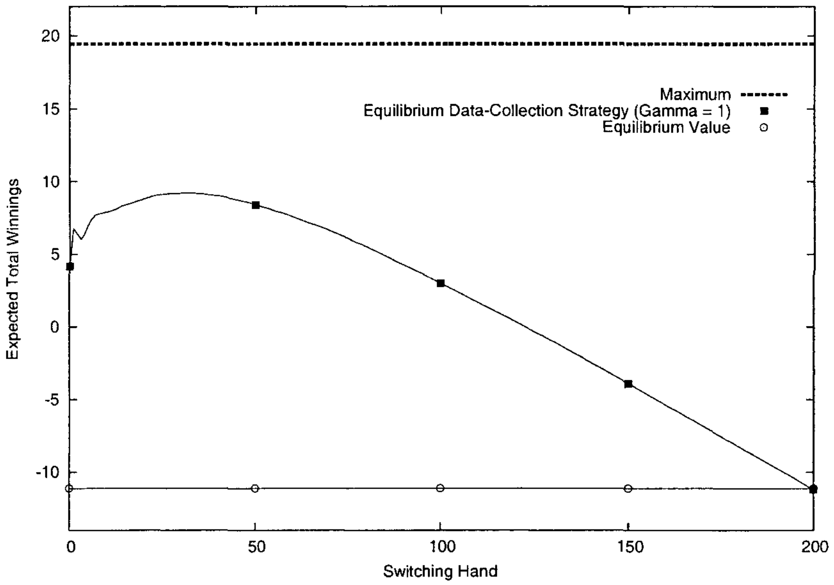


Figure 4.4: Sample Total Winnings Plot

The payoff-rate and total winnings plots show the average performance of the methods, but do not tell the whole story. It could be the case that the methods do not perform well in many of the trials, but a few very good trials are bringing up the average. A proportion-above-equilibrium plot shows what proportion of the trials at hand t have an expected total winnings (for 200 total hands) above the amount that would be won by receiving the equilibrium rate for the same number of hands. Figure 4.5 is an example of a proportion-above-equilibrium plot and shows that against this opponent 95% of the trials have an expected total winnings above the equilibrium value at hand 50. An interesting thing to note is that playing an equilibrium strategy for the entire 200-hand match does not guarantee the equilibrium rate. Equilibrium strategies have an expected winning rate of -0.0556 \$/hand, but equilibrium players don't receive this value on every hand. In practice, there is about a 50% chance of winning more than the equilibrium rate and a 50% chance of winning less than the equilibrium rate when playing a static equilibrium strategy over a 200-hand match. Thus the expected value for the equilibrium strategies is not only the average but also the median in the distribution of match winnings.

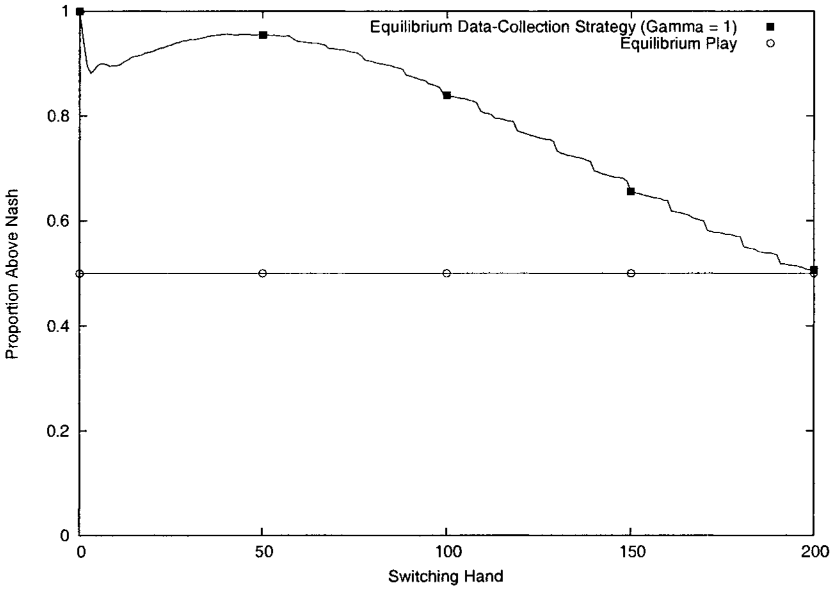


Figure 4.5: Sample Proportion-Above-Equilibrium Plot

Properties of the Testpoints

The first testpoint, $O_1 = (0.8, 0.29)$, is in the region where $S_1 = (\alpha = 0, \beta = 0, \gamma = 1)$ is the best-response strategy. This testpoint has a maximum exploitability of 0.0381 \$/hand. The exploratory data-collection strategies have low payoff rates against this testpoint, with all having rates less than -0.2 \$/hand, which is much lower than the equilibrium rate of -0.0556 \$/hand. Since the initial estimates $(\eta = 0.5, \xi = 0.5)$ for the opponent are not too distant from the correct model and the payoff rates for the exploratory data-collection strategies are much lower than the equilibrium rate, this testpoint will have the property that a modeller using one of the exploratory data-collection strategies should switch from exploration to exploitation very early in a match.

The second testpoint, $O_2 = (0.75, 0.8)$, is in the region where $S_2 = (0, 1, 0)$ is the best-response strategy. This opponent is highly exploitable with a potential expected payoff rate of 0.1 \$/hand for P1. In addition, this opponent is very close to the region where S_3 is maximal, and thus S_3 obtains the high payoff rate of 0.0917 \$/hand against this opponent. Thus it is expected that opponent modelling methods should usually find a good counter-strategy, as there are two very good options, but it is likely that the actual best counter-strategy will often not be identified in many of the trials, even after a large number of hands. The exploratory strategies do not have high payoff rates against this opponent, with each achieving a payoff rate less than -0.1 \$/hand. Finally, the initial estimates held by the modeller are that η and ξ are both 0.5, which suggest the counter-strategy of playing either S_2 or S_3. These last two factors suggest that for this opponent it is best to switch very early to the perceived best-response strategy when playing against him.

The third testpoint, $O_3 = (0.67, 0.4)$, is in the region where $S_3 = (0, 1, 1)$ is the best-response strategy. This opponent is much less exploitable than the previous two, with P1 obtaining an expected payoff rate of 0.0111 \$/hand when playing the best-response strategy. The exploratory strategies have poor payoff rates against this opponent as well, with all having rates less than -0.15 \$/hand. Since the initial estimates are close to the correct values, the results for this testpoint will suggest that the methods should switch from the data-collection strategy to the perceived best-response strategy very early in the trials. In Section 4.2.5 the effects of having different initial estimates and different weights on the initial estimates will be shown; some of the experiments will show the ill effect of switching early when the starting guess is not very close to the actual opponent strategy.

The fourth testpoint, $O_4 = (0.17, 0.2)$, is in the region where $S_4 = (1, 0, 0)$ is the best-response strategy. This testpoint has a maximum exploitability of 0.05 \$/hand and the exploratory strategies have slightly higher payoff rates than the equilibrium value, ranging from -0.0222 to -0.0167 \$/hand. This point nearly borders the region where S_5 is the best-response strategy, and thus S_5 achieves the high payoff rate of 0.0444 \$/hand against this opponent. Since the point is close to a border, it is expected that the modelling methods will usually find one of the two good counter-strategies after a large number of hands, but not necessarily the best counter-strategy.

The fifth test point, $O_5 = (0.25, 0.17)$, is in the region where $S_5 = (1, 0, 1)$ is the best-response strategy. This test point has a maximum exploitability of 0.0278 $/hand and the exploratory strategies have payoff rates comparable to the equilibrium value. Since this point is quite distant from its neighbouring regions, the modelling strategies should find the best counter-strategy in most of the trials, although possibly after many hands since the effect of the poor fictitious data must be overcome.

The final testpoint, $O_6 = (0.25, 0.67)$, is in the region where $S_6 = (1, 1, 0)$ is the best-response strategy. This counter-strategy is exactly the ExploreXi strategy, which means that the total winnings plot for this point will suggest that the modeller should never switch from the ExploreXi data-collection strategy to the perceived best-response strategy. In fact, all of the exploratory strategies have high payoff rates against this opponent in comparison to the equilibrium value of -0.0556 $/hand: ExploreXi obtains the maximum expected payoff rate of 0.0972 $/hand, ExploreEta has an expected payoff rate of 0.0278 $/hand, and BalancedExplore has an expected payoff rate of 0.0625 $/hand. This opponent is highly exploitable which means opponent modelling methods should have a good chance of winning more than the equilibrium value of the game. Furthermore, the test-point is located relatively distant from the nearest bordering region, which means that the exploratory strategies should identify the correct counter-strategy in a large percentage of the trials after 900 hands, because once the estimates are close to the correct values the point-estimate will be in the correct region.

Table 4.1 shows the expected payoff-rates of the six pure strategies that are potential best-response strategies against each of the six testpoints. The table headings are slightly abbreviated, as $EV[S, O_i]$ represents $EV[x|S, O_i]$. Table 4.2 shows the payoff-rates of the different data-collection strategies against each of the testpoints.

S	$EV[S, O_1]$	$EV[S, O_2]$	$EV[S, O_3]$	$EV[S, O_4]$	$EV[S, O_5]$	$EV[S, O_6]$
S_1	0.0381	-0.1417	-0.0222	-0.0389	-0.0139	-0.1806
S_2	-0.0714	0.1	-0.0333	-0.1	-0.1111	0.0556
S_3	0.0143	0.0917	0.0111	-0.1056	-0.0972	-0.0139
S_4	-0.2810	-0.3417	-0.2333	0.05	0.0139	-0.0694
S_5	-0.1952	-0.35	-0.1889	0.0444	0.0278	-0.1389
S_6	-0.3048	-0.1083	-0.2	-0.0167	-0.0694	0.0972

Table 4.1: Expected Payoff-Rates of Candidate Best-Response Strategies against Testpoints

4.2.2 Equilibrium Data-Collection Strategy Comparison

For this study, P1 uses five equilibrium data-collection strategies, corresponding to $\gamma = \{0, 0.25, 0.5, 0.75, 1\}$, and the combined-hand estimates described in Section 3.4.1 are used to compute the opponent model. The payoff-rate plots for the two fixed exploitability values

S	$EV[S,O_1]$	$EV[S,O_2]$	$EV[S,O_3]$	$EV[S,O_4]$	$EV[S,O_5]$	$EV[S,O_6]$
Equilibrium	-0.0556	-0.0556	-0.0556	-0.0556	-0.0556	-0.0556
ExploreXi	-0.3048	-0.1083	-0.2	-0.0167	-0.0694	0.0972
ExploreEta	-0.2190	-0.1167	-0.1556	-0.0222	-0.0556	0.0278
BalancedExplore	-0.2619	-0.1125	-0.1778	-0.0194	-0.0625	0.0625

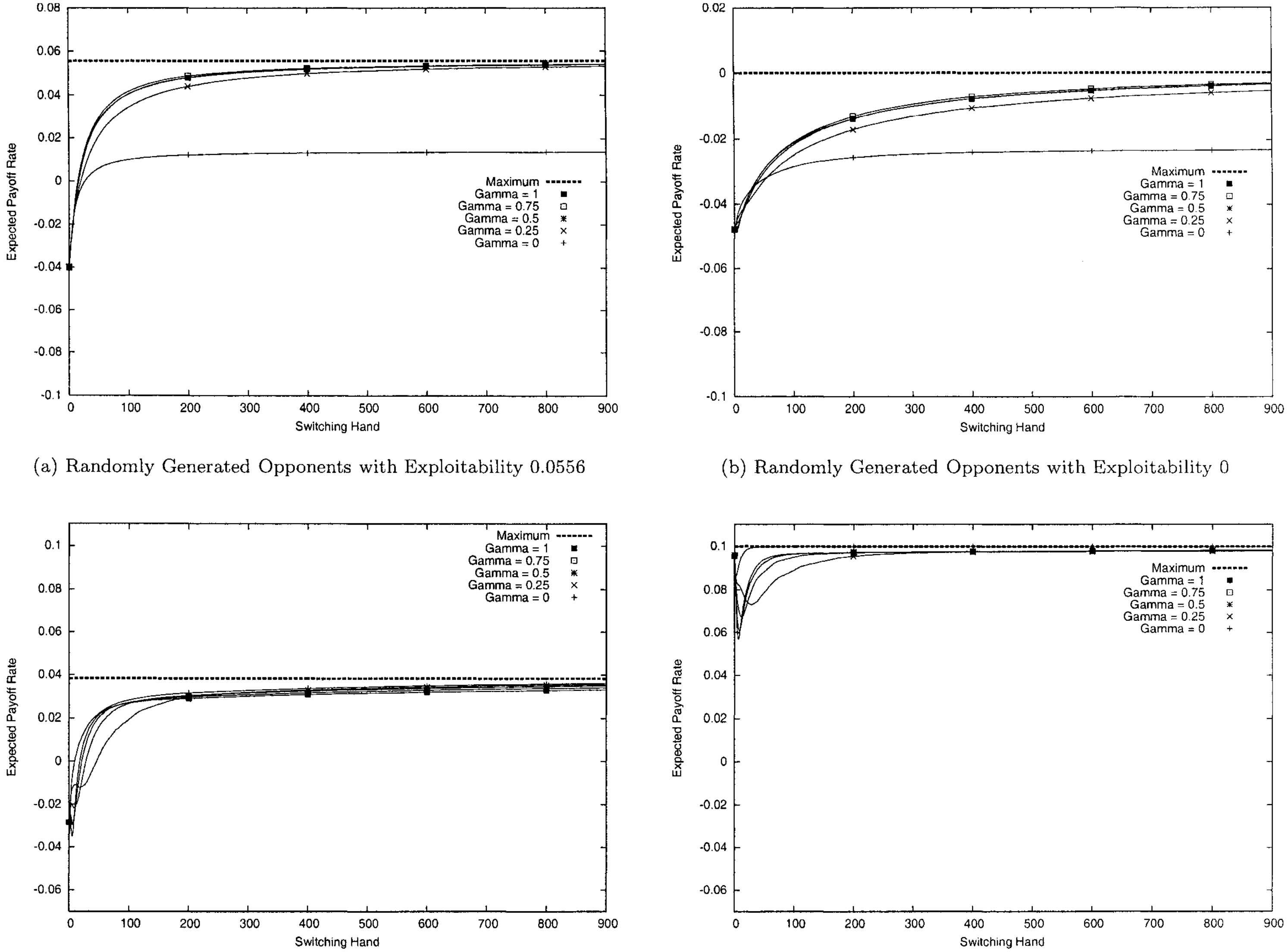
Table 4.2: Expected Payoff-Rates of Data-Collection Strategies against Testpoints

and the six P2 testpoints are shown in Figure 4.6 and Figure 4.7.

The plots showing results against opponents of fixed exploitability (Figures 4.6(a) and 4.6(b)) show that the average payoff-rate of the model discovered by the $\gamma = 0$ strategy is much lower than the other equilibrium data-collection strategies. Against opponents with exploitability 0.0556 \$/hand, the $\gamma = 0$ strategy is only reaching about 0.01 \$/hand on average. This is due to the fact that the $\gamma = 0$ strategy receives no information about P2's η parameter. The $\gamma = 0.25$ data-collection strategy achieves much better results, but the best results are obtained by the $\gamma = 0.5$, $\gamma = 0.75$ and $\gamma = 1.0$ data-collection strategies. These three strategies with the higher settings of γ all appear to have very similar average-case results. In general, convergence to the maximum payoff-rate is slower in Figure 4.6.(b) than in Figure 4.6.(a). The line segments making up the contour for opponents of exploitability 0 are half the length of the line segments making up the contour for the exploitability 0.0556 opponents, which results in the average distance of the exploitability 0 opponents to their nearest bordering region being half that of the exploitability 0.0556 opponents. Being closer to bordering regions makes it more difficult to identify the correct best-response strategy, which causes convergence to be slower.

The payoff-rate plots for O_2 and O_3 (Figures 4.6(d) and 4.7(a)) exhibit an interesting phenomenon that is present in many of the payoff-rate plots in this thesis; the average payoff-rate at switching hand 0 is close to the maximal payoff-rate, but the average payoff-rate decreases for about 10 hands before increasing back towards the maximum. This phenomenon will be examined in much greater detail in Section 4.2.4, but the underlying idea for why it occurs is that the initial model based on the initial estimates is "too good to be true"; as variance in the models increases from early observations, the average payoff-rate decreases as bad models are sometimes created.

The payoff-rate plots for O_4, O_5 and O_6 (Figures 4.7(b), 4.7(c) and 4.7(d)) are all similar in that the $\gamma = 0$ data-collection strategy does not converge to the maximum payoff-rate, while the other data-collection strategies do converge to the maximum. As in the payoff-rate plots for the opponents of fixed exploitability, the series corresponding to the three higher settings of γ ($\gamma = 0.5, 0.75$ and 1.0) are all grouped very tightly together, while the $\gamma = 0.25$



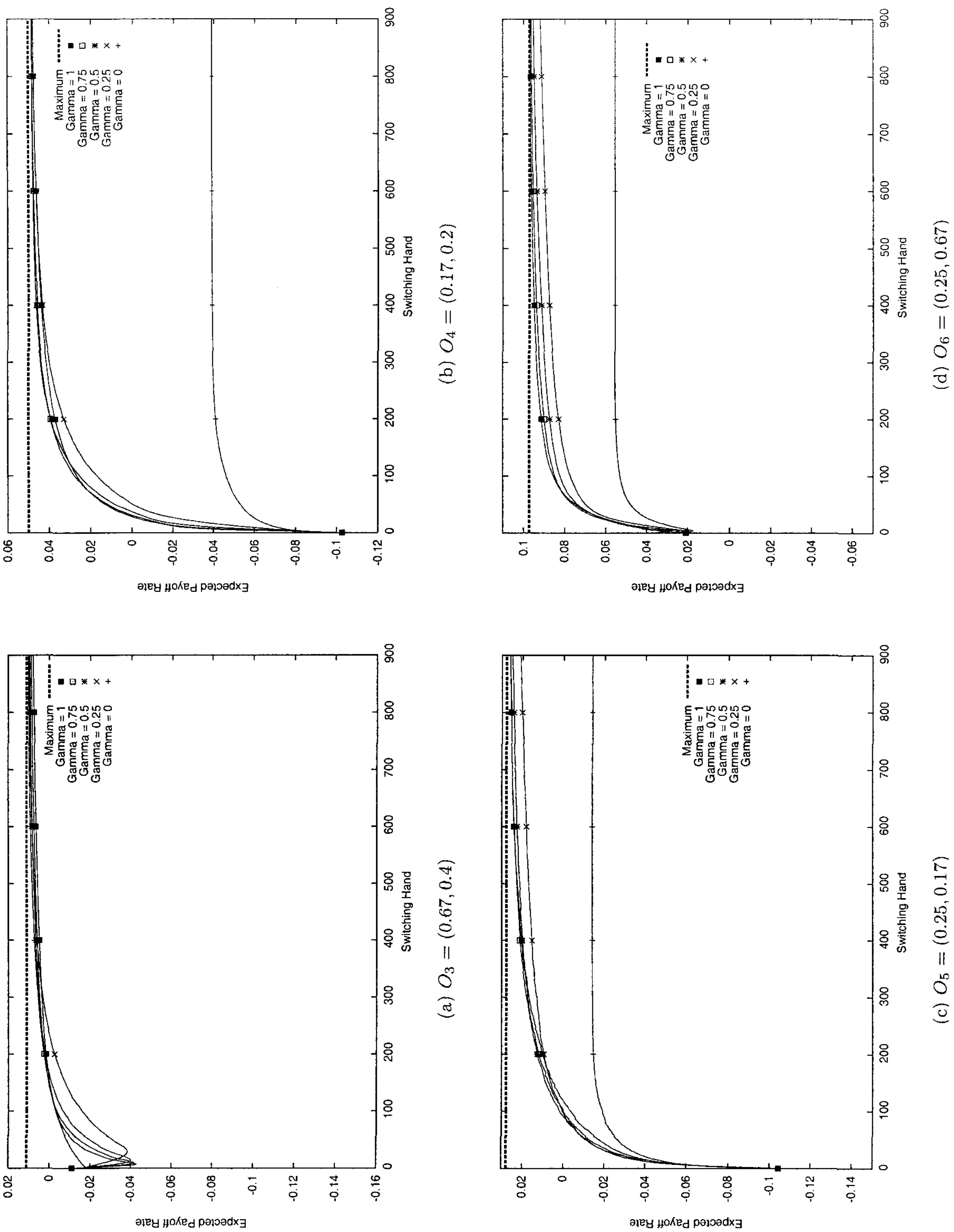
(a) Randomly Generated Opponents with Exploitability 0.0556

(b) Randomly Generated Opponents with Exploitability 0

(c) $O_1 = (0.8, 0.29)$

(d) $O_2 = (0.75, 0.8)$

Figure 4.6: Equilibrium Data-Collection Strategies Payoff-Rate Plots



(a) $O_3 = (0.67, 0.4)$

(b) $O_4 = (0.17, 0.2)$

(c) $O_5 = (0.25, 0.17)$

(d) $O_6 = (0.25, 0.67)$

Figure 4.7: Equilibrium Data-Collection Strategies Payoff-Rate Plots

data-collection strategy converges a little slower. The convergence to maximum is slightly slower for O_5 than for the other opponents because O_5 is closer to the equilibrium point $(1/3, 1/3)$ than the other testpoints in Figure 4.1; thus it is more likely to mistakenly infer that O_5 is in a region that has a bad counter-strategy against O_5 (such as S_2, S_3 or S_6) than it is the other testpoints.

Overall, the payoff-rate plots seem to suggest that better modelling occurs when γ is higher and poor modelling occurs for very low values of γ. When $\gamma = 0$ the data-collection strategy only succeeds when the initial estimate of η is very good, such as for testpoints O_1, O_2, and O_3. This is a repercussion of the fact that when $\gamma = 0$, P1 never puts P2 into a situation where the η parameter is used, which means P1 can never learn about it. Thus when using the $\gamma = 0$ equilibrium data-collection strategy, P1's model can only shift up and down from the point $(\eta = 0.5, \xi = 0.5)$ in Figure 4.1.

The setting of the γ parameter allows P1 to focus the data-collection on either the η (high γ) or the ξ (low γ) parameter. Setting γ to give equal consideration to both parameters seems to result in the best equilibrium data-collection strategy against general opponents, as the region diagram (Figure 2.5) is symmetric with regards to P2's parameters, and it is also nearly symmetric in terms of costs as well. Thus for the case of P1 modelling P2 in Kuhn Poker it is not more important to be more accurate on the estimation of one parameter than the other. It is the case in other games that some parameters are more important than others, and thus the modeller can gain more by focusing learning on the important parameters.

To gain datapoints with certainty (complete-information) about P2's η parameter, the J|Q deal must occur and P1 must bet in Round One; this event happens with probability $\gamma/18$. To gain datapoints with certainty about the ξ parameter, the K|J deal must occur and P1 must pass in Round One; this event happens with probability $(1 - \gamma)/6$. Thus for P1 to expect equal numbers of datapoints with certainty in his estimates, the setting of γ must satisfy

$$\frac{\gamma}{18} = \frac{1-\gamma}{6}.$$

Solving this equation gives the setting $\gamma = 0.75$. Each hand has a probability of being a datapoint (certain or uncertain) which contributes to the η estimate of $(\gamma/18) + (\gamma/6)$. Each hand also has a probability of being a datapoint which contributes to the ξ estimate of $(1 - \gamma)/6 + 1/6$. Therefore, to expect equal numbers of total datapoints, γ must satisfy

$$\frac{\gamma}{18} + \frac{\gamma}{6} = \frac{1-\gamma}{6} + \frac{1}{6}$$

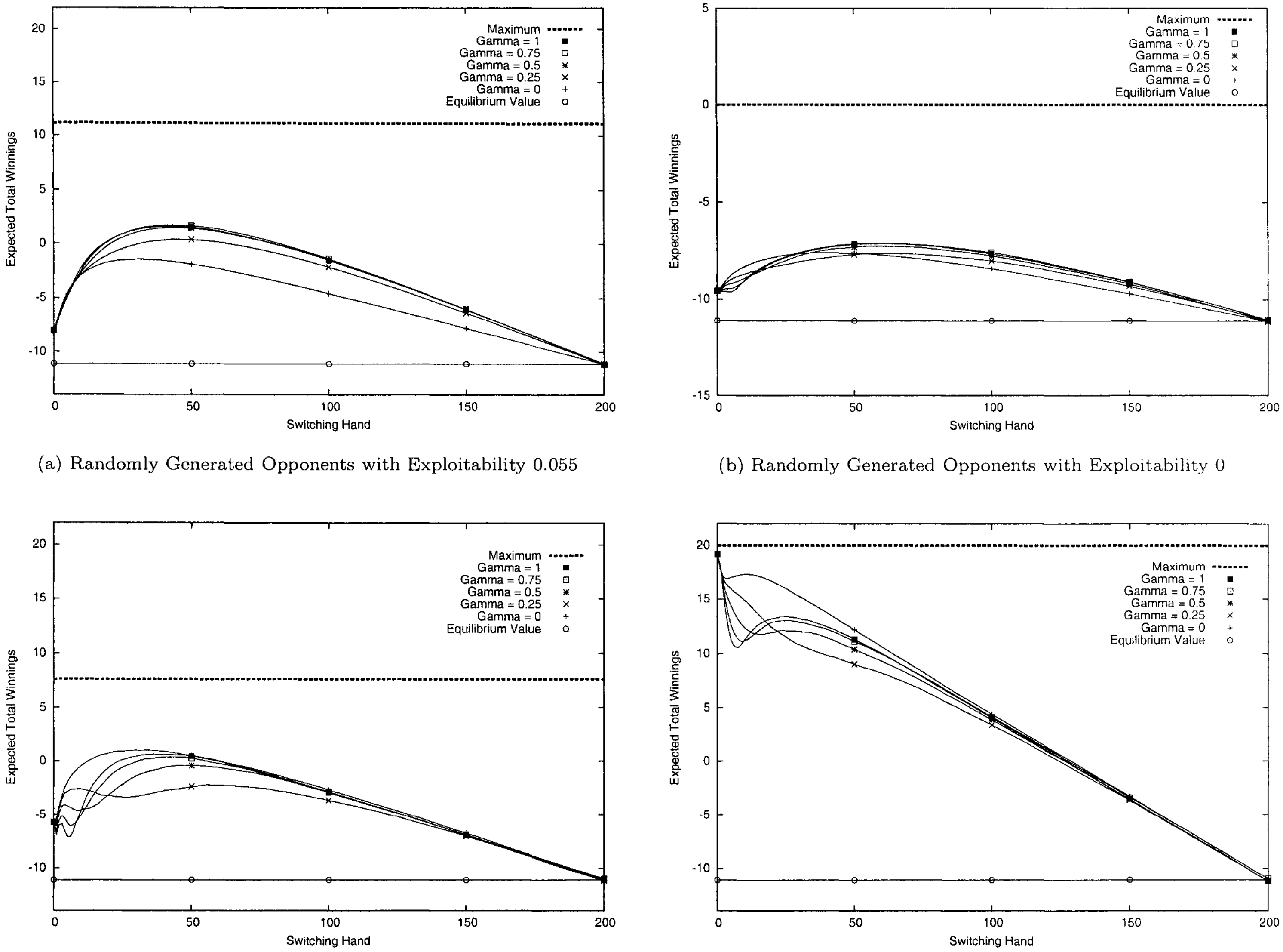
Solving this equation results in the setting $\gamma = 6/7 \approx 0.857$. The setting of γ which achieves the least squares error in the estimates of both parameters is likely between 0.75 and 0.857.

The total winnings plots, shown in Figure 4.8 and Figure 4.9, show the expected value of the various data-collection strategies for the horizon of 200 total hands. Since each equilibrium strategy has the same expected winning rate, these series should appear in the same order as in the payoff-rate plots (the winnings up to hand t should be the same for each equilibrium strategy for every t).

In the total winnings plot showing the equilibrium data-collection strategies used against opponents with a fixed exploitability of 0.0556, Figure 4.8(a), the best equilibrium data-collection strategies are achieving positive expected total winnings, with the peak being around hand 45. Thus against opponents of this exploitability, the use of opponent modelling to find a good counter-strategy appears to be much more favourable than the alternative of simply playing an equilibrium strategy throughout the match. Against opponents with the lower fixed exploitability of 0, shown in Figure 4.8(b), opponent modelling appears to be less useful, but is still a better alternative than settling for the equilibrium payoff-rate.

The total winnings plots for O_1, O_4, O_5 and O_6 (Figures 4.8(c), 4.9(b), 4.9(c) and 4.9(d)) all similarly show benefits of doing opponent modelling, with winnings higher than equilibrium and a peak at around hand 40. The plots for O_2 and O_3 (Figures 4.8(d) and 4.9(a)) are different in that the expected total winnings starts very high and then decreases over the course of the match. This is due to the fact that the counter-strategy to the initial estimates is very good against these opponents, so any time spent on collecting data is wasted; the model improves very little and a lot of winnings are sacrificed during the data-collection period. The results for O_2 and O_3 suggest the rash conclusion that data-collection is not useful and the modeller should just trust the initial estimates when determining a counter-strategy. However, testpoints such as O_4 and O_5 are counter-examples to this conclusion, as they are points where the initial model achieves a sub-equilibrium payoff-rate and data-collection pays large dividends.

The purpose of doing opponent modelling when game theoretic solutions are known is to attempt to win more than the value of the game. Figure 4.10 and Figure 4.11 show what proportion of the trials have a higher projected total winnings than the equilibrium value as a function of the switching hand. The results show that the modeller does better than the equilibrium value in about 70 to 95% of the trials if he uses an equilibrium data-collection strategy with a high setting of γ to learn and switches to a best-response strategy after a reasonable number of hands; the plots suggest that in a 200-hand match, the modeller should switch at about hand 50. Recall that playing an equilibrium strategy for the entire match will only achieve the equilibrium payoff-rate 50% of the time, so modelling has been



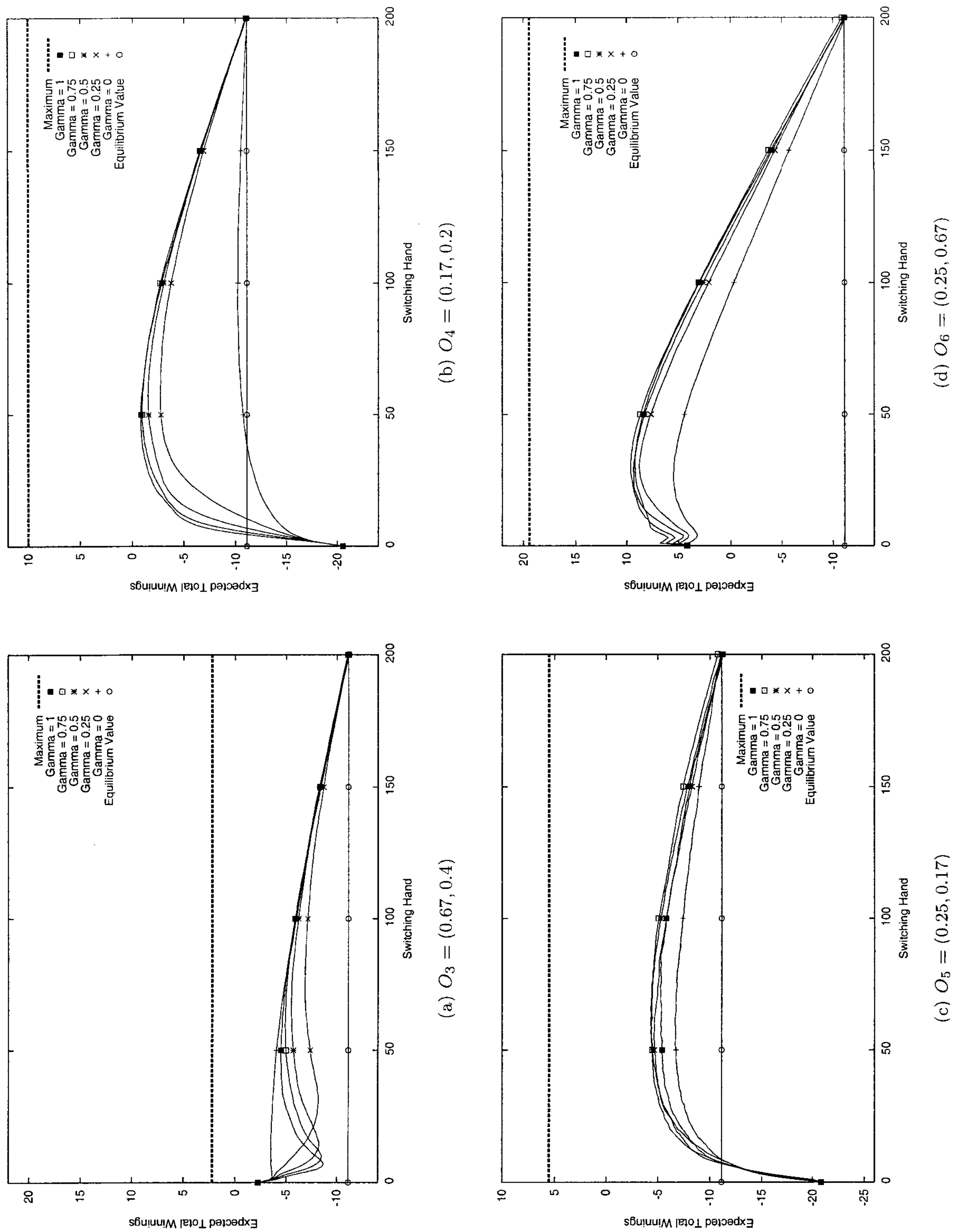
(a) Randomly Generated Opponents with Exploitability 0.055

(b) Randomly Generated Opponents with Exploitability 0

(c) $O_1 = (0.8, 0.29)$

(d) $O_2 = (0.75, 0.8)$

Figure 4.8: Equilibrium Data-Collection Strategies Total Winnings Plots



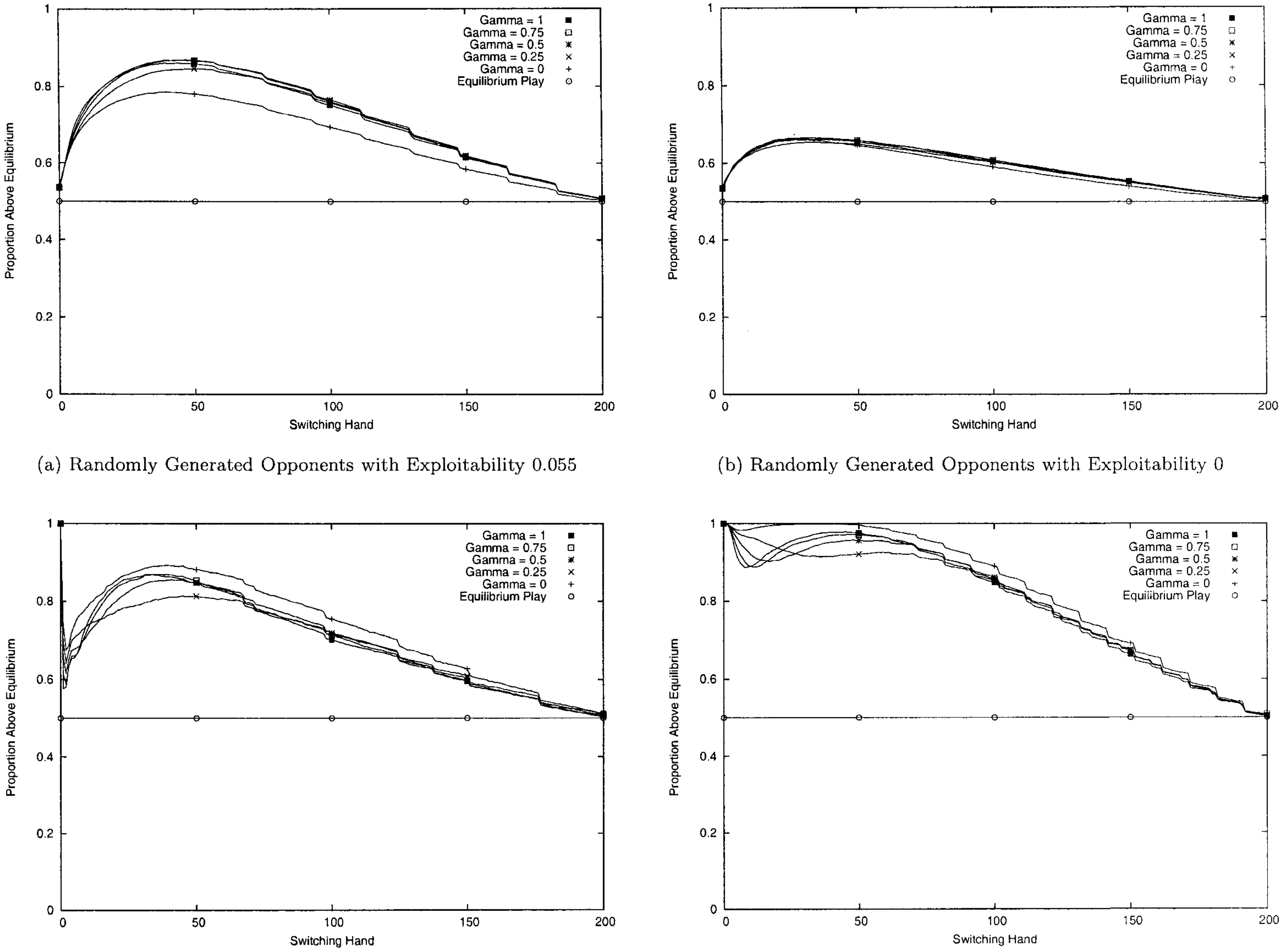
(a) $O_3 = (0.67, 0.4)$

(b) $O_4 = (0.17, 0.2)$

(c) $O_5 = (0.25, 0.17)$

(d) $O_6 = (0.25, 0.67)$

Figure 4.9: Equilibrium Data-Collection Strategies Total Winnings Plots



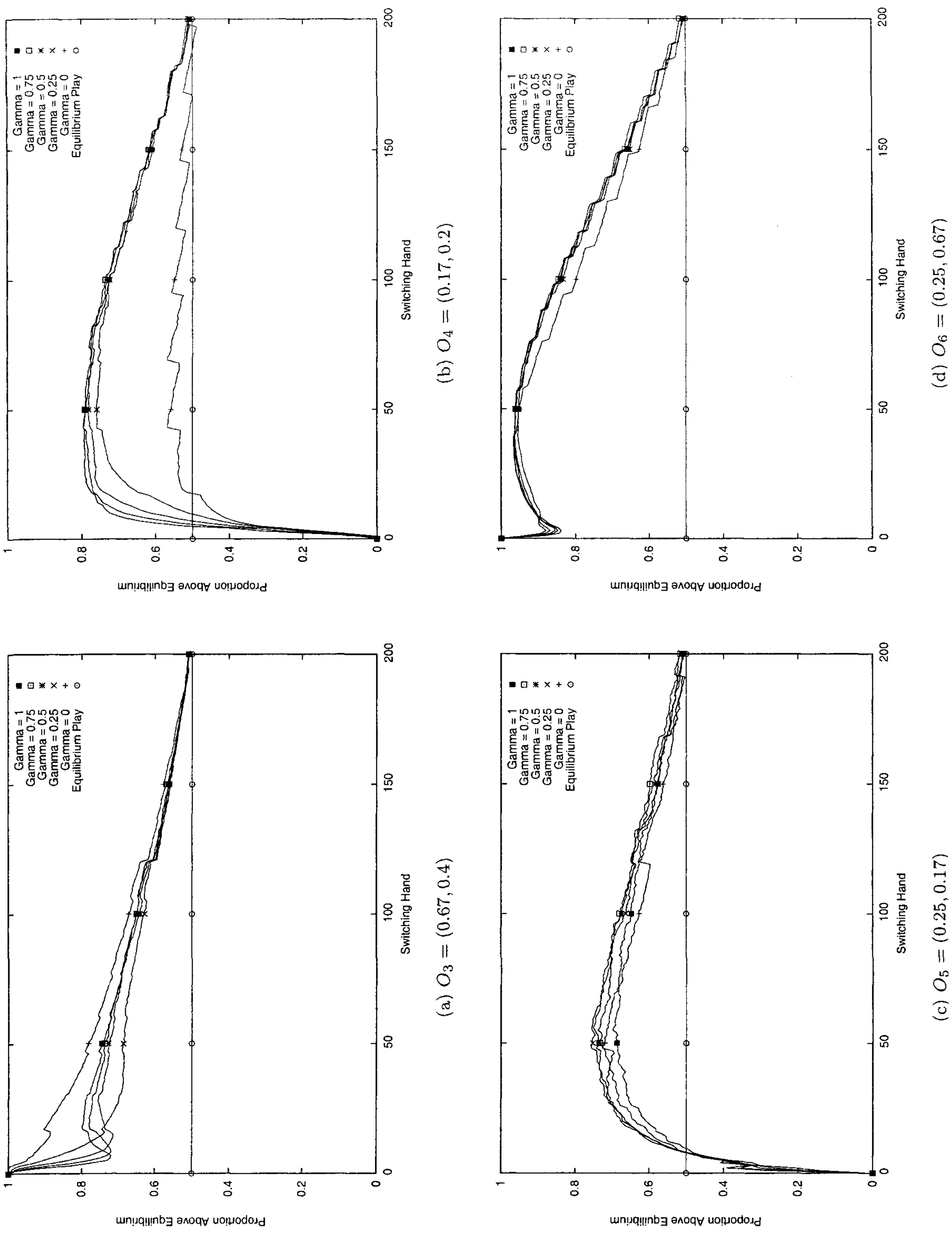
(a) Randomly Generated Opponents with Exploitability 0.055

(b) Randomly Generated Opponents with Exploitability 0

(c) $O_1 = (0.8, 0.29)$

(d) $O_2 = (0.75, 0.8)$

Figure 4.10: Equilibrium Data-Collection Strategies Proportion-Above-Equilibrium Plots



(a) $O_3 = (0.67, 0.4)$

(b) $O_4 = (0.17, 0.2)$

(c) $O_5 = (0.25, 0.17)$

(d) $O_6 = (0.25, 0.67)$

Figure 4.11: Equilibrium Data-Collection Strategies Proportion-Above-Equilibrium Plots

shown to be beneficial in each of the experiments.

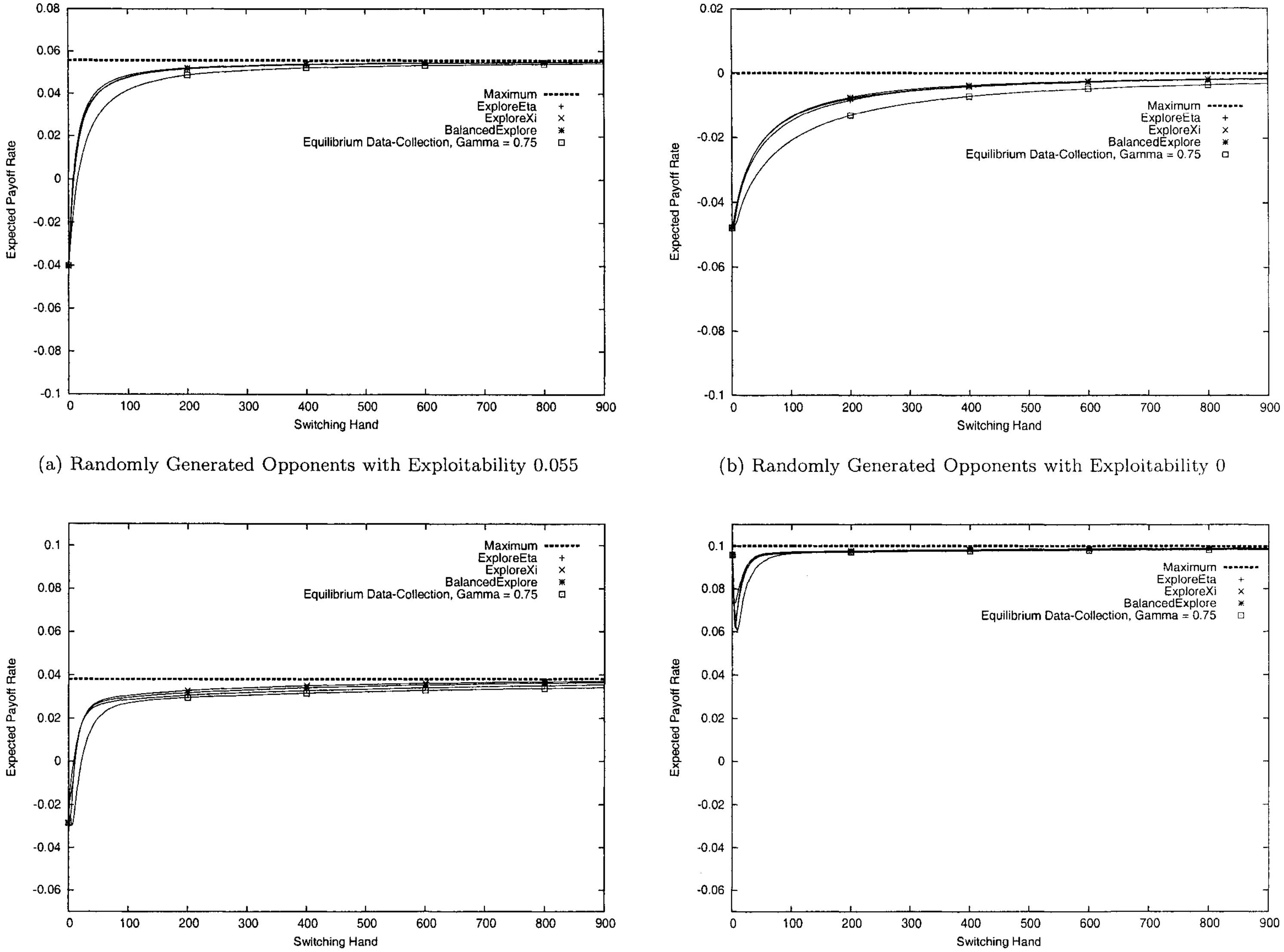
Many of the proportion-above-equilibrium charts in Figure 4.10 and Figure 4.11 exhibit an interesting sawtooth-shape, particularly evident in the $\gamma = 0$ series in Figure 4.11(b). An explanation for this sawtooth shape is forthcoming in Section 4.2.4.

4.2.3 Exploratory Data-Collection Strategy Comparison

The main problem with using equilibrium data-collection strategies is that they can take a very long time to collect useful data. P1's equilibrium strategies in Kuhn Poker restrict α to the interval $[0, 1/3]$, but when P1 passes with the Jack in Round One (which happens with probability $1 - \alpha$ in that situation), P1 cannot learn anything about P2's parameters. Similarly, when P1 folds with the Queen in Round Three (which happens with probability $1 - \beta$ in that situation), P1 does not get to observe what card P2 holds. This problem is removed by the "exploratory strategies" which each have α and β set equal to 1; the final parameter, γ, can be set to explore η (high γ) or ξ (low γ) more thoroughly. The three exploratory data-collection strategies used in these experiments are ExploreEta = (1, 1, 1), ExploreXi = (1, 1, 0), and BalancedExplore = (1, 1, 0.5). In the graphs shown in this section, the equilibrium data-collection strategy corresponding to $\gamma = 0.75$ has been plotted for comparison to the exploratory data-collection strategies.

It is evident from the payoff-rate plots in Figure 4.12 and Figure 4.13 that the exploratory data-collection strategies do learn more quickly than the equilibrium data-collection strategies. In each plot there is a gap between the exploratory strategies, which are typically grouped tightly together, and the $\gamma = 0.75$ equilibrium data-collection strategy. In particular, when the initial model achieves a low payoff-rate against the opponent, as for O_4 and O_5, the exploratory strategies recover much more quickly from the bad initial model. No single exploratory strategy is shown to be superior to the others, as each exploratory strategy has at least one testpoint for which it converges the fastest. Although the exploratory strategies learn faster than the equilibrium data-collection strategies, the use of an exploratory data-collection strategy is risky due to the fact that the value of the game is not assured. The modeller could lose a large amount of money while collecting data.

The total winnings plots for the opponents having fixed exploitabilities, Figures 4.14(a) and 4.14(b), show that although the exploratory strategies have lower average payoff-rates during data-collection than equilibrium against these opponents, the improved models allow for higher expected total winnings than the equilibrium data-collection strategy. However, to attain the higher peaks in the total winnings graphs, the modeller must switch from exploration to exploitation much earlier in the match, around hand 30.



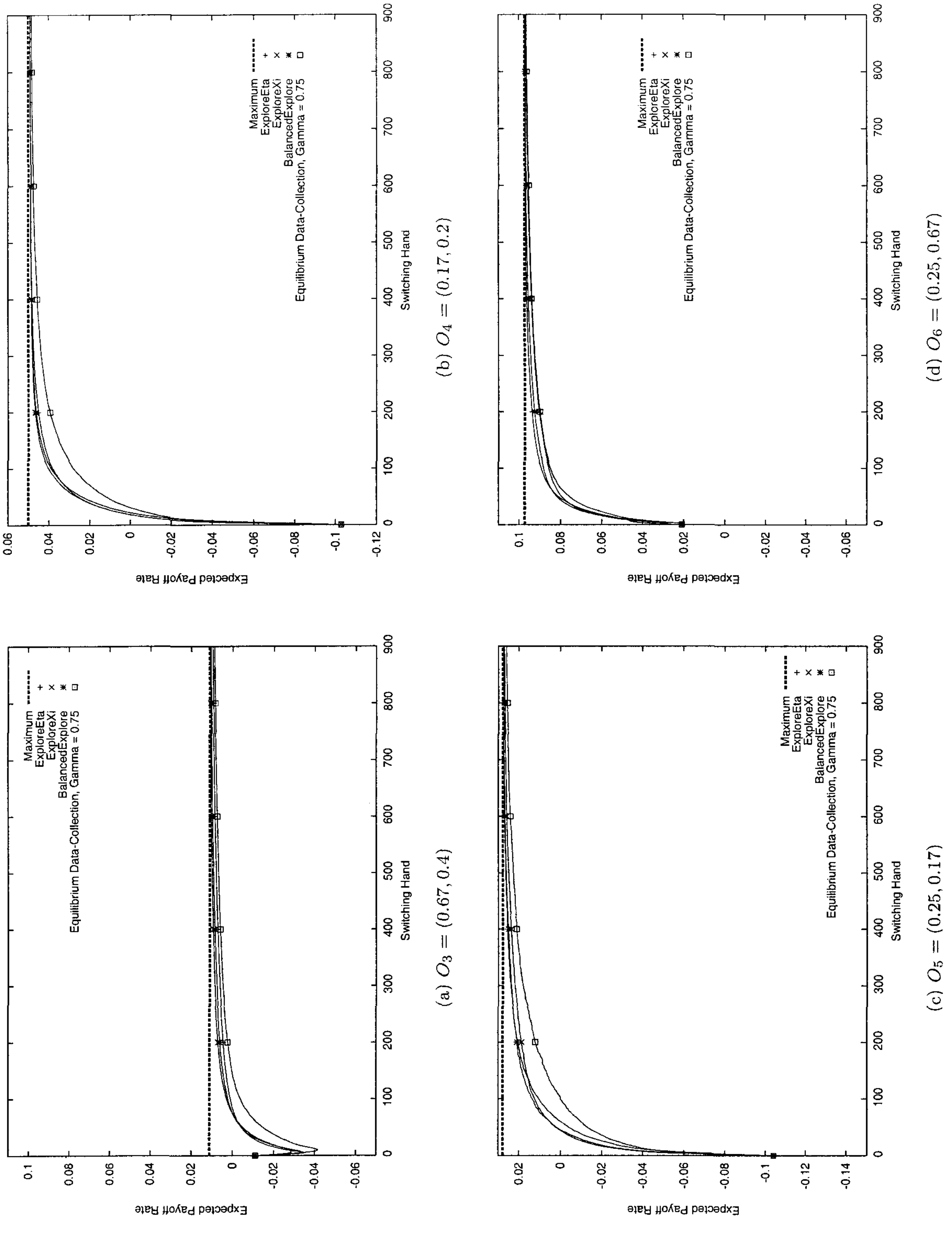
(a) Randomly Generated Opponents with Exploitability 0.055

(b) Randomly Generated Opponents with Exploitability 0

(c) $O_1 = (0.8, 0.29)$

(d) $O_2 = (0.75, 0.8)$

Figure 4.12: Exploratory Data-Collection Strategies Payoff-Rate Plots



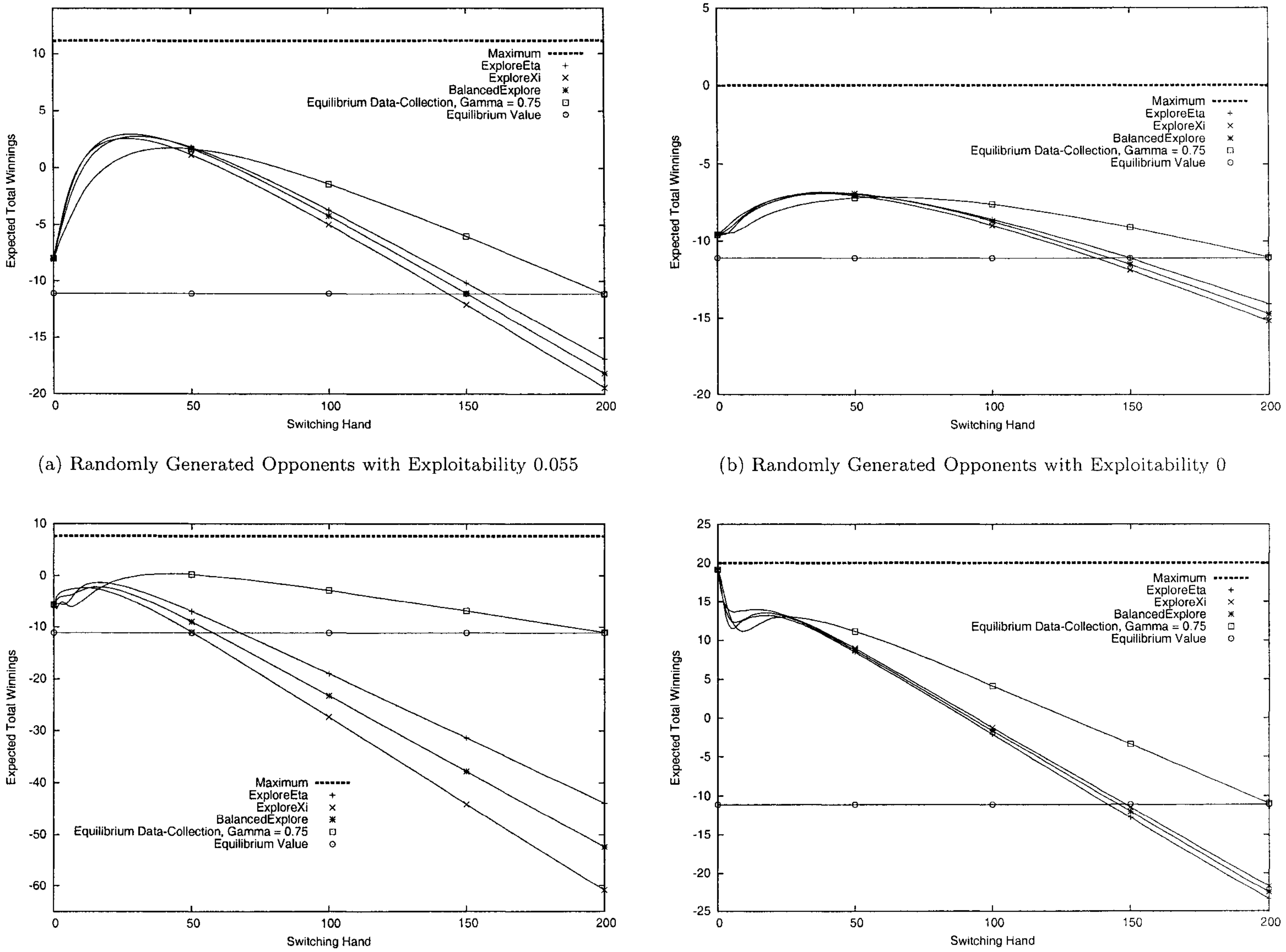
(a) $O_3 = (0.67, 0.4)$

(b) $O_4 = (0.17, 0.2)$

(c) $O_5 = (0.25, 0.17)$

(d) $O_6 = (0.25, 0.67)$

Figure 4.13: Exploratory Data-Collection Strategies Payoff-Rate Plots



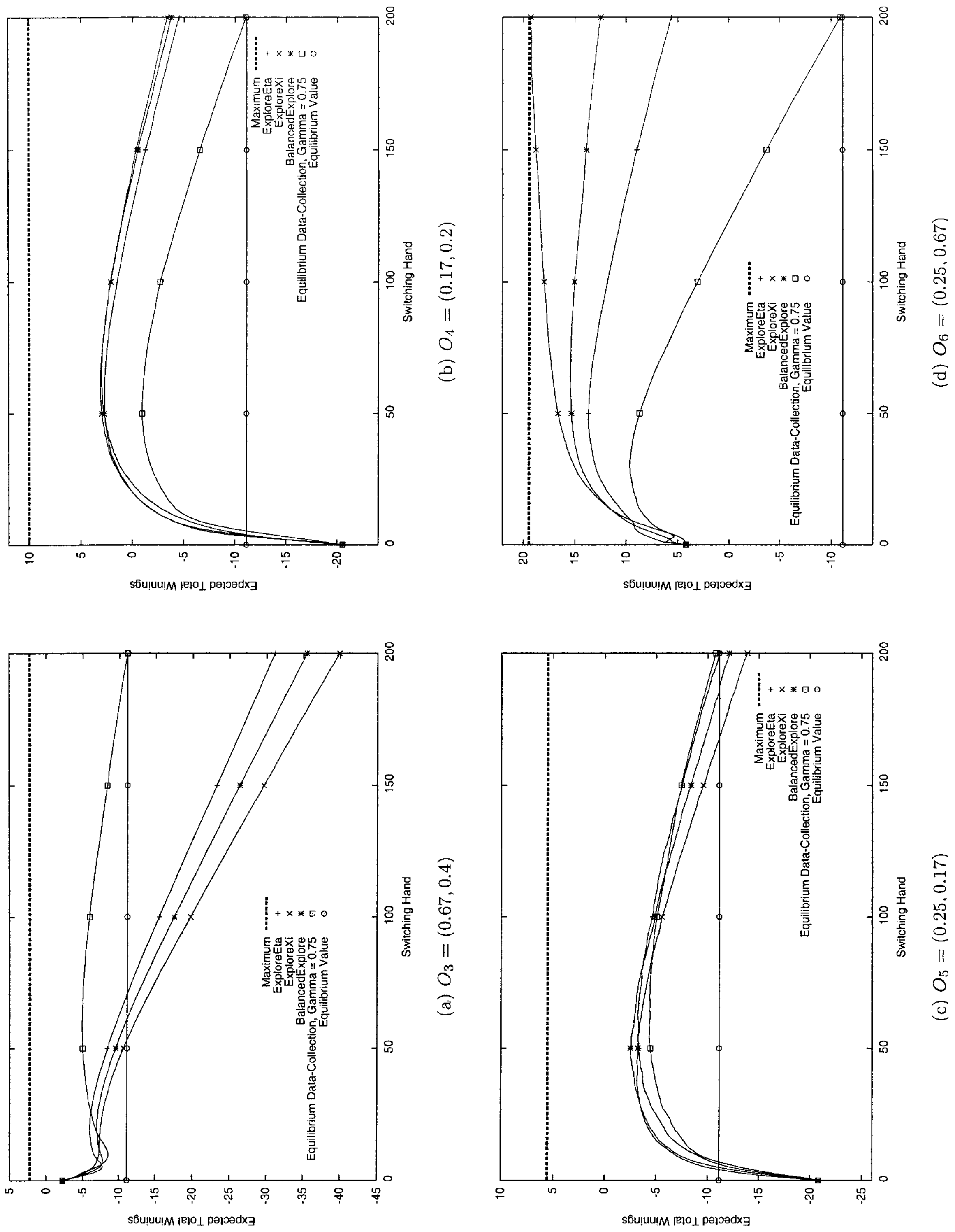
(a) Randomly Generated Opponents with Exploitability 0.055

(b) Randomly Generated Opponents with Exploitability 0

(c) $O_1 = (0.8, 0.29)$

(d) $O_2 = (0.75, 0.8)$

Figure 4.14: Exploratory Data-Collection Strategies Total Winnings Plots



(a) $O_3 = (0.67, 0.4)$

(b) $O_4 = (0.17, 0.2)$

(c) $O_5 = (0.25, 0.17)$

(d) $O_6 = (0.25, 0.67)$

Figure 4.15: Exploratory Data-Collection Strategies Total Winnings Plots

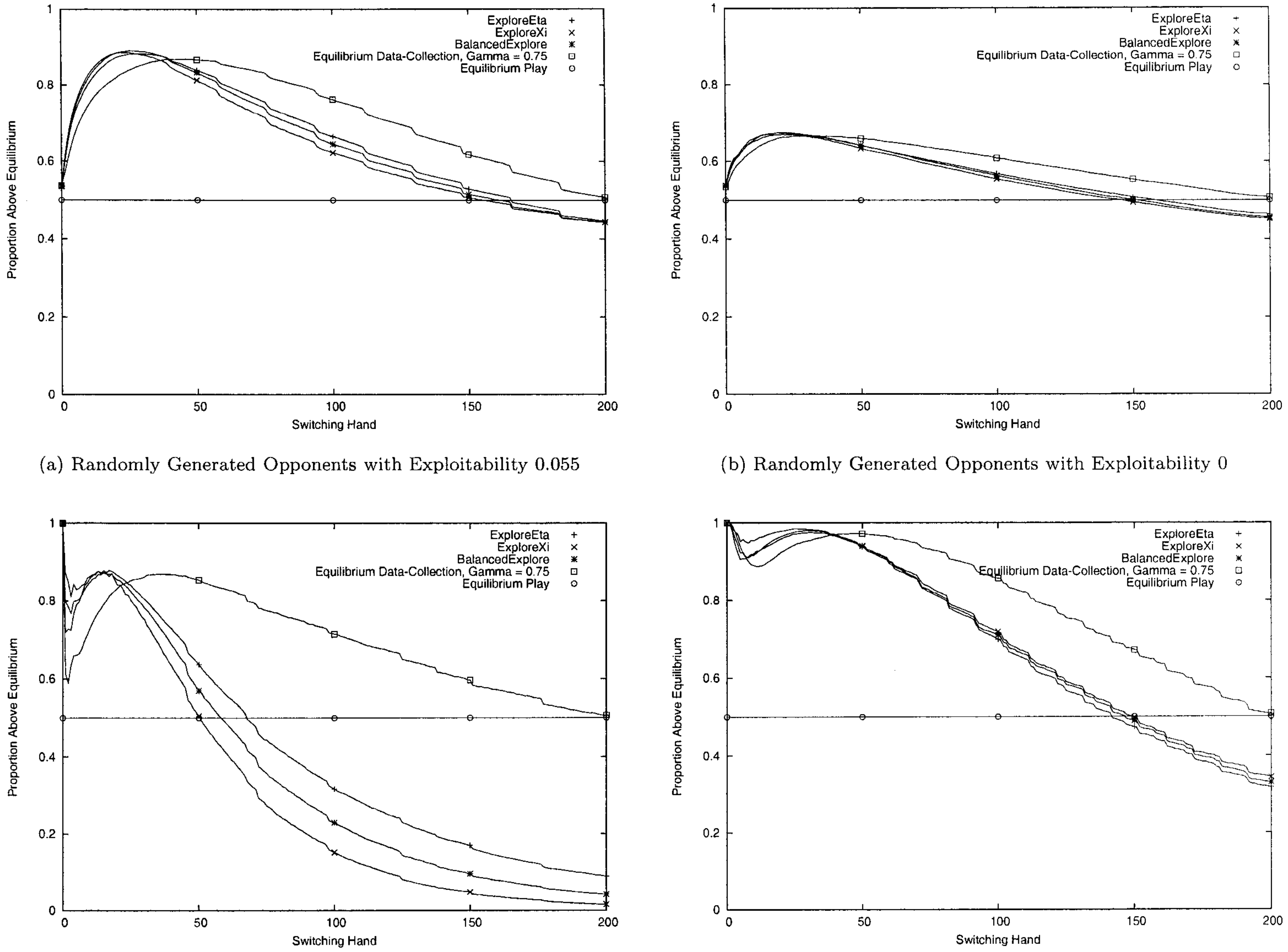
The risk of not being guaranteed the equilibrium payoff-rate is demonstrated particularly by the total winnings plots for O_1 and O_3, Figures 4.14(c) and 4.15(a). For these points the safe equilibrium data-collection strategy has a higher expected winnings than the exploratory data-collection strategies, because the exploratory strategies lose a lot of money while exploring. Conversely, the plots for O_4 and O_6, Figures 4.15(b) and 4.15(d), show the advantage held by the exploratory strategies when they have higher payoff-rates than the equilibrium value.

The plots showing the proportion of trials which achieve higher expected total winnings at each switching hand than the equilibrium value are given in Figure 4.16 and Figure 4.17. These plots show that if the data-collection strategy loses more than the equilibrium payoff rate against an opponent, then the modeller generally needs to switch to an exploitive strategy earlier in the match to recoup his losses than if he was using an equilibrium data-collection strategy. As before, each of the data-collection strategies have total winnings exceeding the equilibrium value in 70 to 95% of the trials regardless of the opponent. However, the appropriate switching point now differs for many of the test-points, and the proportion of exploratory strategies with total winnings above the equilibrium value can quickly drop below 50% for some opponents, such as opponents O_1 and O_3. On the other hand, opponents such as O_4 and O_6 can make the exploratory strategies look deceivingly preferrable to the equilibrium data-collection strategies. This occurs because the exploratory strategies have higher than equilibrium payoff-rates against these opponents, which results in a large proportion of trials finishing 200 hands with higher-than-equilibrium winnings.

4.2.4 Explanation of Graph Peculiarities

There are two peculiarities present in many graphs in this section that remain to be explained. The first peculiarity to explain is the dips, short periods in which the average payoff-rate achieved by the model decreases rather than increases, that are present at the beginning of several of the payoff-rate graphs. An example of such a dip occurs in Figure 4.6(d). The second peculiarity is the odd sawtooth shape exhibited in many of the proportion-above-equilibrium plots, possibly most notably displayed by the $\gamma = 0$ equilibrium data-collection method in Figure 4.11(b).

The first peculiarity is relatively easy to explain. Against certain opponents, such as O_2, the initial estimates held by the modeller of his opponent's parameters happen to result in a counter-strategy that is very good against the opponent. Thus at hand 0, every trial has a high expected payoff-rate against the opponent. Focusing on O_2, there are two pure strategies which have very high payoff-rates against this opponent, S_2 and S_3, which have



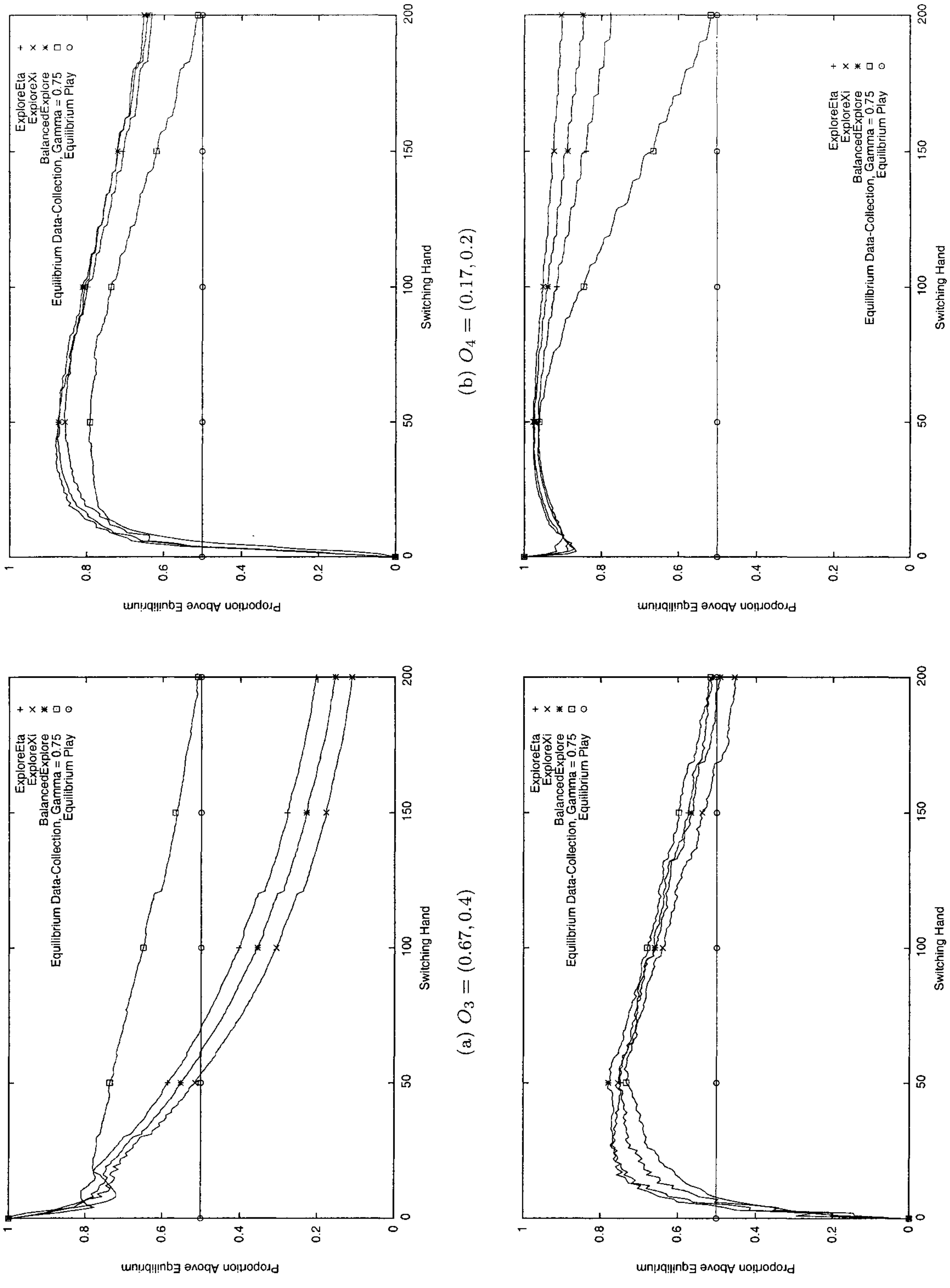
(a) Randomly Generated Opponents with Exploitability 0.055

(b) Randomly Generated Opponents with Exploitability 0

(c) $O_1 = (0.8, 0.29)$

(d) $O_2 = (0.75, 0.8)$

Figure 4.16: Exploratory Data-Collection Strategies Proportion-Above-Equilibrium Plots



(a) $O_3 = (0.67, 0.4)$

(b) $O_4 = (0.17, 0.2)$

(c) $O_5 = (0.25, 0.17)$

(d) $O_6 = (0.25, 0.67)$

Figure 4.17: Exploratory Data-Collection Strategies Proportion-Above-Equilibrium Plots

payoff-rates of 0.1 \$/hand and 0.0917 \$/hand respectively. All other pure strategies have payoff-rates less than -0.1 \$/hand against O_2. The counter-strategy to the initial estimates ($\eta = 0.5, \xi = 0.5$) is to play each of S_2 and S_3 half the time; this strategy has an expected payoff-rate of 0.0958 \$/hand against O_2.

Table 4.3 shows the number of trials for which the modeller suggests one of the good counter-strategies (S_2, S_3, or half S_2 and half S_3), as well as the number of trials that the modeller suggests some other counter-strategy, for the equilibrium data-collection method with $\gamma = 0.75$. At hand 0, all 30000 trials have an expected payoff-rate of 0.0958 \$/hand. As the modeller collects a few hands of data, many of the sequences of observations will lead to models that identify one of the good counter-strategies as the best strategy. However, some sequences of observations lead to models that do not identify either S_2 or S_3 as the best counter-strategy, which results in a huge decrease in the payoff-rate for the modeller in these trials. Table 4.3 shows that for a small number of hands, a significant portion of the trials can experience a sequence of observations leading away from the good initial estimates to a bad model. After 7 hands, nearly 1/4 of the trials have experienced such a sequence, and consequently the average payoff-rate decreases. As the number, t, of hands of data collected increases, sequences of observations of length t leading to bad models become less and less likely to occur. This is illustrated in the table by the increasing number of trials that find a good counter-strategy from hand 8 onwards. As more and more trials recommend a good counter-strategy, the payoff-rate increases.

In summary, at hand 0 the modeller in each trial has an identical model, that happens to be very good. Due to bad luck in the first few observations, some trials produce models with much lower payoff-rates. Because the marginal increase in payoff-rates made by the trials which improve does not balance the large decrease in payoff-rates for the trials which got worse, the overall average payoff-rate decreases, which explains the initial drop in the payoff-rate plot. As the number of hands of data increases, sequences leading to bad models become less likely, since these sequences consist of the modeller repeatedly observing events which are unlikely based on the opponent's parameter settings. As sequences leading to bad models become less likely, the number of trials which find a good counter-strategy increases, and this corresponds to the recovery of the data-collection methods in the payoff-rate plot.

The explanation of the peculiarity in the proportion-above-equilibrium plots is not quite as intuitive as the explanation of the dips in the payoff-rate graphs. The fundamental idea is that each trial's winnings at each t must be an integer, while the minimum winnings required to have projected total winnings above equilibrium is a real number that changes by small fractional steps as t increases. When this threshold value crosses an integer boundary the

t	S_2, S_3, or S_2/S_3	Other Counter-Strategy	Average Switching Payoff-Rate
0	30000	0	0.0958
1	28411	1589	0.0911
2	26467	3533	0.0831
3	24924	5076	0.0757
4	23936	6064	0.0698
5	23360	6640	0.0658
6	23036	6964	0.0630
7	22946	7054	0.0614
8	23014	6986	0.0604
9	23256	6744	0.0605
10	23597	6403	0.0614
11	23967	6033	0.0625
12	24309	5691	0.0638
13	24702	5298	0.0657
14	25070	4930	0.0676
15	25424	4576	0.0693
16	25794	4206	0.0713
17	26125	3875	0.0733
18	26345	3655	0.0746
19	26596	3404	0.0761
20	26794	3206	0.0774

Table 4.3: Frequencies of Different Suggested Counter-Strategies

status of all of the trials on the borderline suddenly changes in one timestep, leading to either a jump or a dropoff in the proportion-above-equilibrium graph.

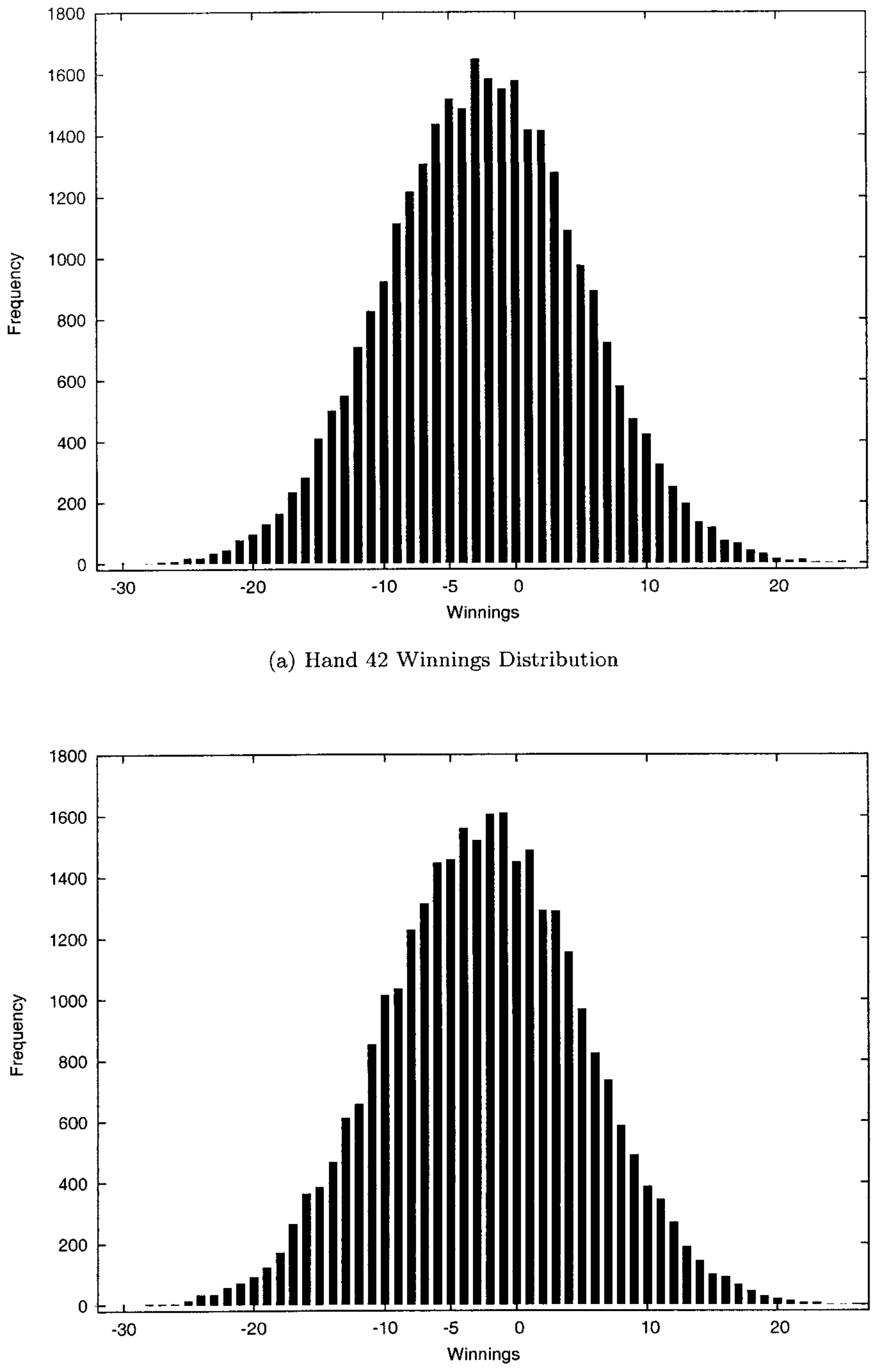
The example that is analyzed here is the $\gamma = 0$ data-collection method against O_4. Suppose every trial recommends the same counter-strategy S, for all t; let V_0 be the payoff-rate of this strategy versus the opponent. For trial i to have a projected total winnings above equilibrium after hand t, it must be the case that i's winnings over the first t hands, $w_i(t)$, satisfies

$$w_i(t) + (200 - t) * V_0 > 200 * (\text{Equilibrium Payoff-Rate})$$

This gives rise to the condition that all trials which have winnings greater than $w_{\min}(t)$ have projected winnings greater than equilibrium, where

$$\begin{aligned} w_{\min}(t) &= 200 * (\text{Equilibrium Payoff-Rate}) - (200 - t) * V_0 \\ &= V_0 t + \text{constant} \end{aligned}$$

Thus if V_0 is negative, then $w_{\min}(t)$ decreases over time, which is the case for O_4 and the counter-strategy S_1, which is the best counter-strategy that the $\gamma = 0$ equilibrium data-collection method can find (since this method never receives data about P2's η parameter, if the fictitious data suggests $\eta > 1/3$, then the only counter-strategies that it can identify



(a) Hand 42 Winnings Distribution

(b) Hand 43 Winnings Distribution

Figure 4.18: Distribution of Winnings at Hands 42 and 43

are S_1, S_2 and S_3). Figure 4.18(a) is a histogram which shows the number of trials for which $w_i(42) = w$, for each of the different levels of winnings, w. Figure 4.18(b) is a similar histogram for hand 43. These particular histograms are shown for three reasons. The first reason is that by hand 42, S_1 has been identified as the best counter-strategy in a majority of the trials. The second reason is that by this point of the match the histograms do not change much from one hand to the next; the equilibrium data-collection strategy has an expected winning rate of -0.0556 \$/hand, so there is a tiny leftward shift of mass in the histograms. The third reason is that there is a jump in the proportion-above-equilibrium graph between hand 42 and hand 43.

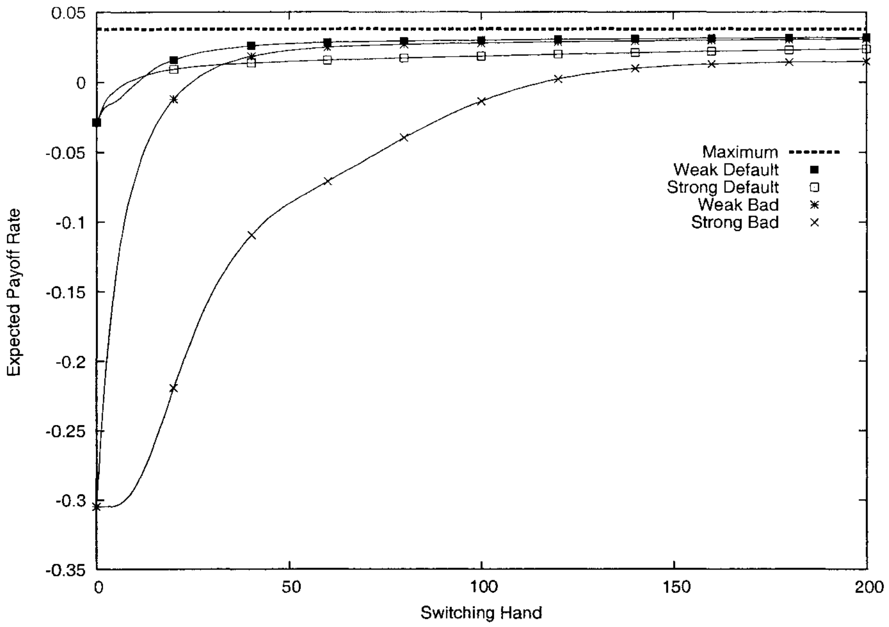
The missing piece of the puzzle is the behaviour of $w_{\min}(t)$ over this period; $w_{\min}(42) = -4.9667$, and $w_{\min}(43) = -5.0056$. What this means is that at hand 42, all trials with winnings of -4 or greater (and recommend the counter-strategy S_1) are projected to have total winnings above equilibrium. At hand 43, all trials with winnings of -5 or greater are projected to have total winnings above equilibrium, increasing the number of trials above equilibrium by roughly 1400.

Between jumps there is a slow decline in the proportion-above-equilibrium plot, as there is a leftward shift of mass in the winnings histograms (recall that the expected payoff-rate of the equilibrium data-collection strategy is -0.0556 \$/hand). While the threshold $w_{\min}(t)$ remains between a pair of integers x_1 and $x_1 - 1$ over such a period, there are a few less trials with winnings of x_1 or greater after each hand.

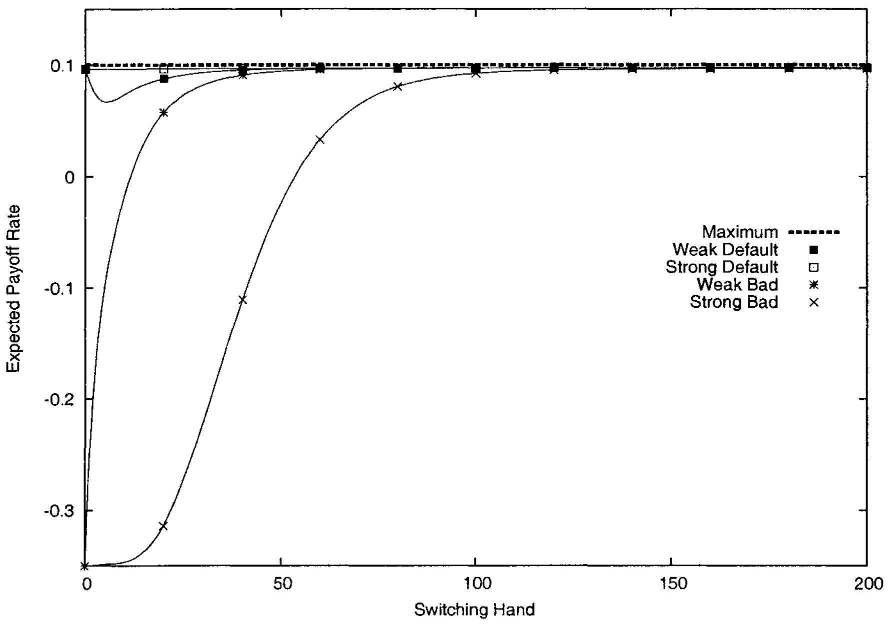
4.2.5 Changing the Initial Estimates

In the previous sections, the modeller begins with fictitous data that generates the initial estimates ($\eta = 0.5$, $\xi = 0.5$) and each of these estimates is generated from 2 fictitious points, for all of the experiments shown. These initial estimates can give the modelling methods very high initial payoff-rates against some opponents, particularly against O_2. This study varies the initial estimates and the weight placed on the estimates for the parameter estimation methods. The purpose is to see how the methods recover from bad initial models, as well as to measure how more persistent initial models affect the results.

Figure 4.19 shows payoff-rate results for $O_1 = (0.8, 0.29)$ and $O_2 = (0.75, 0.8)$ with fictitious data supporting bad initial models of (0.2, 0.75) and (0.15, 0.1) respectively. Results are shown for the BalancedExplore data-collection method with four series shown on each graph. The first series is the weak default estimates, which is the settings used for all previous experiments shown (initial estimates are (0.5, 0.5) generated from two fictitious points each). The second is the strong default belief, which is the initial model (0.5, 0.5)



(a) $O_1 = (0.8, 0.29)$



(b) $O_2 = (0.75, 0.8)$

Figure 4.19: BalancedExplore Payoff-Rate Plots for Different Initial Estimates

weighted by 20 fictitious datapoints each. The third and fourth series are the weak and strong bad initial estimates, similarly weighted with two points and 20 points of fictitious data respectively.

The plots show that when the modeller has weak bad initial estimates, they can easily be overcome, and this series has nearly as high a payoff-rate as the weak default estimates by hand 50. Strong initial estimates are harder to eliminate from the model, which is why the payoff-rate for the strong bad initial estimates takes much longer to converge to the maximum. For O_1, the series corresponding to strong bad initial estimates has not fully recovered by hand 200, due to the fact there is only one good counter-strategy for this opponent (Table 4.1 shows that the best counter-strategy has a payoff-rate of 0.0389 \$/hand while the second-best counter-strategy has a payoff-rate of 0.0143 \$/hand). The series corresponding to strong bad initial estimates for O_2 converges much more quickly, recovering by about hand 120, due to the fact that there are two very good counter-strategies against this opponent.

The results suggest that it is safest to give the initial estimates a low weight, so that they quickly become negligible when actual observations are made.

4.3 P2 Modelling P1 in Kuhn Poker

The case where P2 models P1 is quite different from the opposite case studied in the last section; one major difference is that if P2 plays his equilibrium strategy, he expects to win against P1 at a rate of 0.0556 \$/hand. This means that P2 may be less motivated to change strategies to take advantage of P1's mistakes, as P2 already has a winning strategy. Another difference is that P2 only has four possible pure counter-strategies, while P1 has six possible pure counter-strategies in the opposite case. This difference suggests that P2 may have more success in finding the best counter-strategy than P1 had in the last section. Another difference is that P2 has to estimate three parameters as opposed to P1 estimating two, which may make explicit modelling more difficult for P2. Another major difference is that P2 has a unique equibrium strategy, ($\eta = 1/3, \xi = 1/3$), while P1 has an infinite number of equilibrium strategies.

As in the last section, P2 has two options to gain information about P1. P2 can try to learn as much as possible about P1 while restricting himself to playing within the space of equilibrium strategies, or P2 can play exploitable strategies which do more exploration of P1's strategy in an attempt to learn faster. When playing an exploratory data-collection strategy, P2 wants to set η high so that he calls bets when holding the Queen more often and learns about P1's settings of α and γ. To learn more about β, P2 should bet often with the

Jack when P1 passes, which is achieved by setting ξ high. The strategy ($\eta = 1, \xi = 1$) will be denoted ExploreAggressively in the plots in this section. This exploration strategy is highly exploitable, as P1 could potentially win 0.167 \$/hand against it. The safer data-collection strategy ($\eta = 2/3, \xi = 2/3$), which has a maximum exploitability of 0.0556 \$/hand, is denoted ExploreModerately and is also shown on the plots in this section.

P2 also has the option of playing the dominated strategy of passing with the King in Round Two when facing a pass, in order to learn what card P1 held when he passed rather than be forced to guess what P1 held if he folds. This dominated strategy will help P2 learn about α, but will prevent P2 from winning extra money from the cases when P1 would call in Round Three with the Queen. P2 will also miss out on the chance to learn about β when he plays the dominated strategy. The strategy where P2 makes this dominated error 25% of the time he is put in that situation and otherwise plays the ExploreAggressive strategy will be denoted as ExploreDominated in the plots in this section.

4.3.1 Experimental Setup

There are four strategy settings that are used for P1 when being modelled by P2 in this chapter, allowing the modeller to face an opponent corresponding to each of the four different best-response strategies. The four P1 strategies, written in the form (α, β, γ), are $O'_1 = (0.2, 0.5, 0.9)$, $O'_2 = (0.12, 0.65, 0.6)$, $O'_3 = (0.25, 0.35, 0.3)$, and $O'_4 = (0.35, 0.65, 0.7)$. These opponents have different levels of exploitability and other significant differences, which means the results between testpoints are not directly comparable.

A single trial consists of a 900-hand match, where the modeller remains in P2 position and the opponent being modelled remains in P1 position. For each hand, the holdings of the two players is randomly selected from the six possibilities according to the uniform distribution. For each decision, the action selected is randomly chosen according to the distribution defined by the acting player's strategy. Results are shown for each of the four test-points, where the results for each test-point are averaged over 30000 trials. For all of the following experiments, P2's initial estimates are ($\alpha = 0.5, \beta = 0.5, \gamma = 0.5$), each generated from two fictitious datapoints.

Properties of the Test Points

The first test point, $O'_1 = (0.2, 0.5, 0.9)$, is in the region where $(\eta, \xi) = (0, 1)$ is the best-response strategy, which achieves a payoff rate of 0.1167 \$/hand for P2. The Explore-Aggressively strategy wins against this opponent at a slightly higher payoff rate than the equilibrium strategy, at a rate of 0.0667 \$/hand, which means there is a good chance that the

ExploreAggressively strategy will be the most successful data-collection strategy. However, since the initial estimate for the β parameter is actually correct, the ExploreDominated strategy may prove more successful as it focuses more learning on the α parameter while sacrificing exploration of the β parameter.

The second test point, $O'_2 = (0.12, 0.65, 0.6)$, is in the region where $(0, 0)$ is the best-response strategy, which achieves a payoff rate of 0.0883 \$/hand. This test point is unique in that the three P2 pure strategies which are not the best-response strategy achieve lower payoff rates than the equilibrium strategy. This means if P2 does not find the best counter-strategy he will not do as well as he would by just playing the equilibrium strategy for the match.

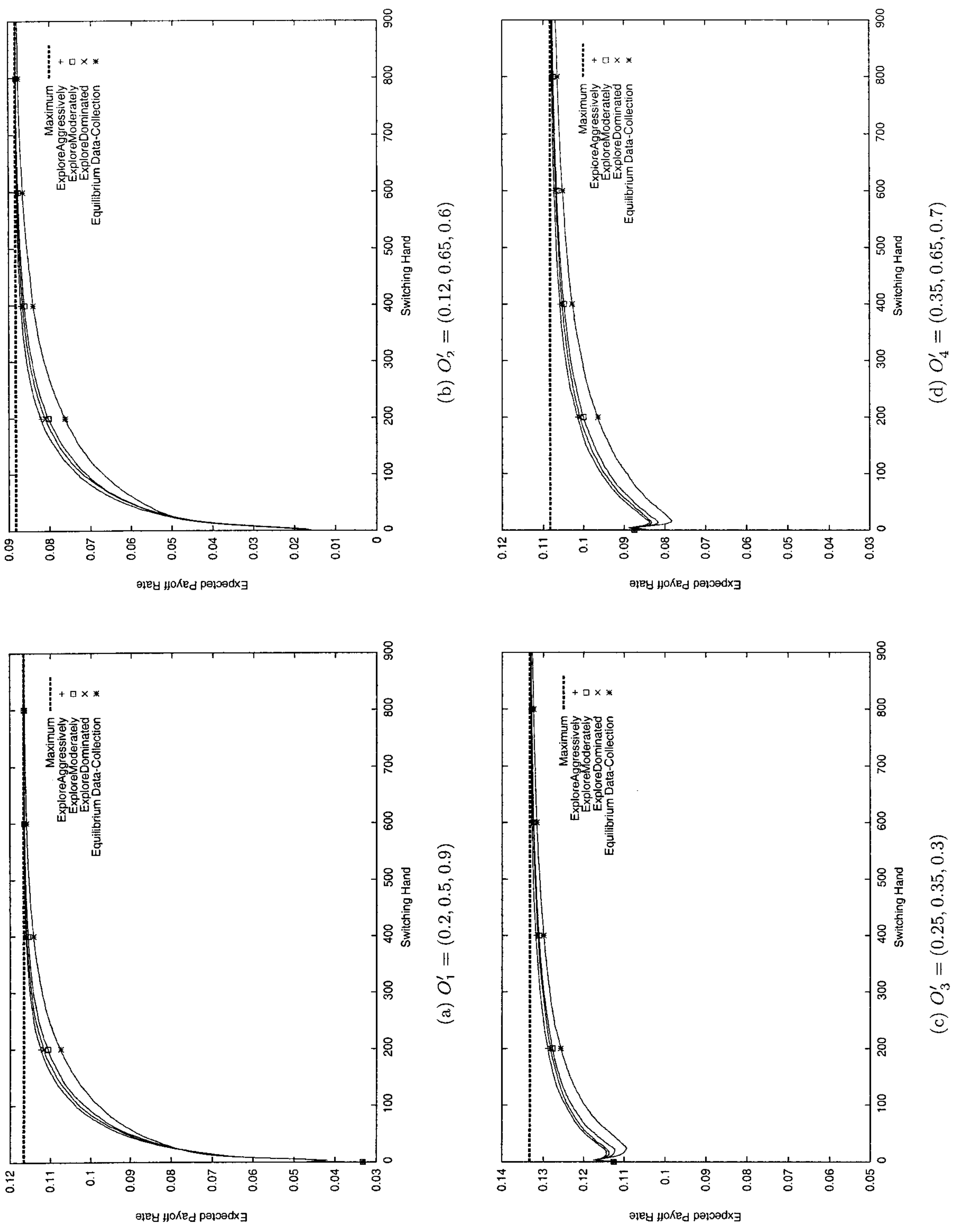
The third test point, $O'_3 = (0.25, 0.35, 0.3)$, is in the region where $(1, 1)$ is the best-response strategy, which achieves a payoff rate of 0.1333 \$/hand. This test point is the opposite of the previous test point in that three of P2's pure strategies achieve a higher payoff rate than the equilibrium rate. It is expected that the ExploreAggressively strategy will perform very well against this opponent, as the ExploreAggressively strategy is exactly the best-response strategy.

The fourth test point, $O'_4 = (0.35, 0.65, 0.7)$, is in the region where $(1, 0)$ is the best-response strategy, which achieves a payoff rate of 0.1083 \$/hand. The ExploreAggressively strategy has a payoff rate of 0.0667 \$/hand, which is slightly higher than the equilibrium rate. Once again, this suggests that the ExploreAggressively strategy may be the most successful data-collection strategy.

4.3.2 Experimental Results

The payoff-rate plots shown in Figure 4.20 once again show that the exploratory data-collection strategies learn faster than the equilibrium data-collection strategy. The ExploreAggressively strategy seems to consistently learn better models than the ExploreDominated strategy, as a side-effect of playing the dominated strategy to learn more about the α parameter is that there are fewer opportunities to learn about the β parameter. The plots suggest that this is a bad tradeoff. Between the exploratory strategies, ExploreAggressively learns slightly faster than ExploreDominated, which in turn learns slightly faster than ExploreModerately, for each of the test opponents. The payoff-rate plots for O'_3 and O'_4 exhibit the initial dips discussed in Section 4.2.4, again due to the high payoff-rates of the initial models, which leaves little room for improvement but lots of room for deterioration.

The total winnings plots shown in Figure 4.21 show the price paid by the ExploreDominated strategy for using the dominated strategy, as ExploreDominated has a noticeably



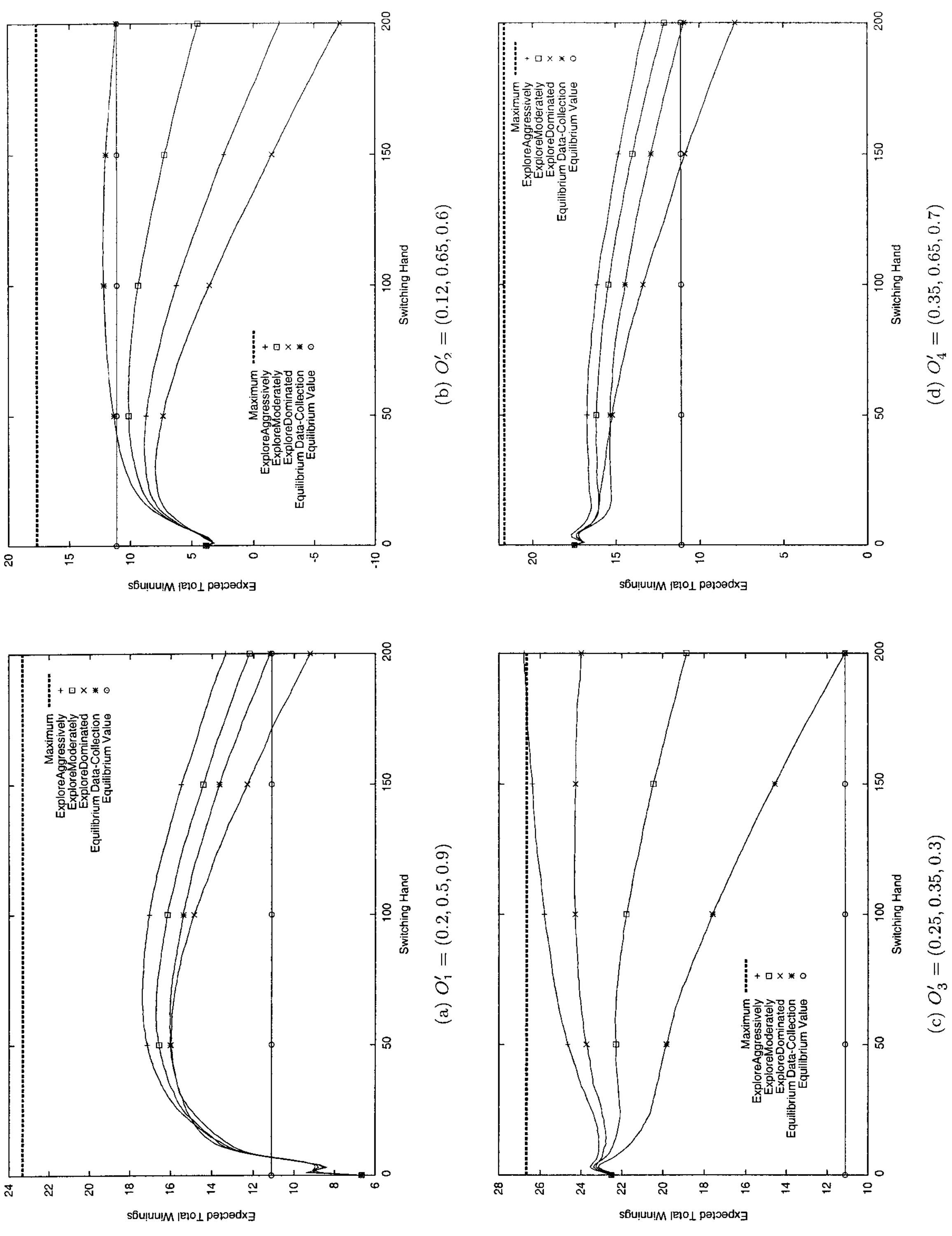
(a) $O_1' = (0.2, 0.5, 0.9)$

(b) $O_2' = (0.12, 0.65, 0.6)$

(c) $O_3' = (0.25, 0.35, 0.3)$

(d) $O_4' = (0.35, 0.65, 0.7)$

Figure 4.20: P2 Data-Collection Strategies Payoff-Rate Plots



(a) $O'_1 = (0.2, 0.5, 0.9)$

(b) $O'_2 = (0.12, 0.65, 0.6)$

(c) $O'_3 = (0.25, 0.35, 0.3)$

(d) $O'_4 = (0.35, 0.65, 0.7)$

Figure 4.21: P2 Data-Collection Strategies Total Winnings Plots

smaller total winnings than the ExploreAggressively strategy for every testpoint. A misleading property of these experiments is that the ExploreAggressively strategy has a payoff-rate higher than the equilibrium value against three of the four testpoints. This seems to suggest that it is a good strategy to use to collect data, as in addition to collecting more data than the equilibrium data-collection strategy, the ExploreAggressively strategy does not risk much by playing in an exploitable fashion. However, a different selection of testpoints (maintaining the property of having one test opponent for each counter-strategy) could have had the opposite property, with ExploreAgressively achieving lower than equilibrium rates against three of the four testpoints.

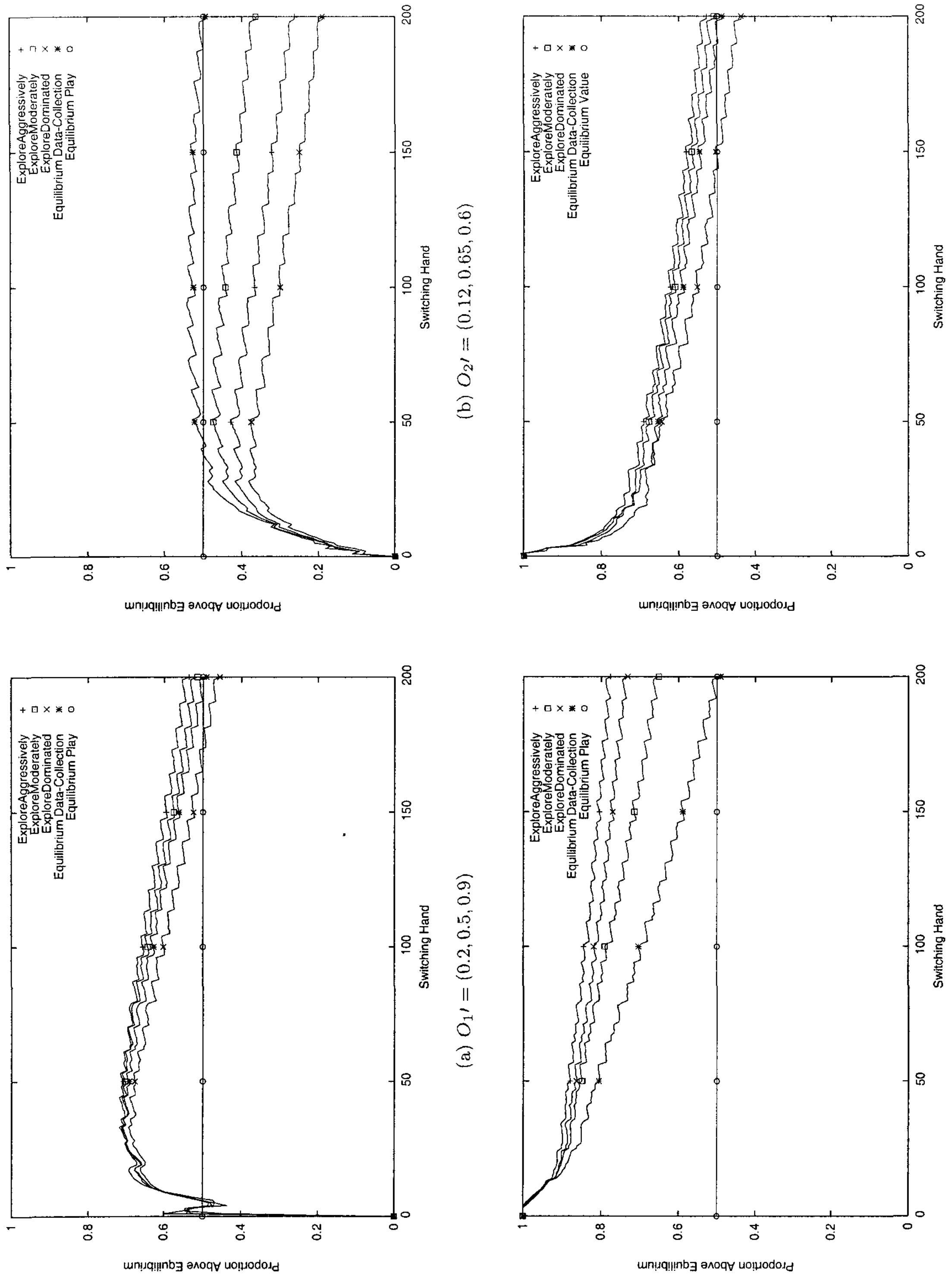
Figure 4.22 shows the proportion-above-equilibrium plots for the four testpoints. In the plots for O'_1 and O'_4, the different strategies are all grouped closely together, meaning none appear much more favourable than any other. In contrast, the plot for O'_2 has only the equilibrium data-collection strategy ever rising above the line representing equilibrium play, making it the clear favourite to use against this opponent. The plot for O'_3 exhibits the opposite result, as each of the exploratory strategies have winning rates higher than the equilibrium rate against this opponent. This results in the exploratory strategies having the best results against O'_3.

4.4 Conclusions

The results shown in this chapter have shown the effectiveness and some of the limitations of explicit modelling. One surprising result is that the modeller does not always converge to the best-response strategy, even after a large number of hands. While this would not be surprising in a large game where there are a huge number of situations to keep track of and data is sparse, this is surprising in the tiny game of Kuhn Poker.

There are several reasons why there is a lack of total convergence within 900 hands. One reason is that when a player uses a nondeterministic strategy, there is a certain amount of variance that is expected to be present. For example, if a player chooses action L in a given situation with probability 0.7, and the player faces that situation 10 times, he will not necessarily take action L 7 times (which would provide the correct estimate). However, as the player is put into that situation more and more often, the proportion of times he chooses L will converge to 0.7.

A second reason why there is slow convergence is that Kuhn Poker is a game of imperfect information and partial observability, meaning that for estimates to be made, assumptions about the modelled players hidden cards must be made. The assumptions described in this thesis depend on the different deals of the game occurring an equal number of times. It



(a) $O_1\prime = (0.2, 0.5, 0.9)$

(b) $O_2\prime = (0.12, 0.65, 0.6)$

(c) $O_3\prime = (0.25, 0.35, 0.3)$

(d) $O_4\prime = (0.35, 0.65, 0.7)$

Figure 4.22: P2 Data-Collection Strategies Proportion-Above-Equilibrium Plots

is unlikely that this will happen exactly, although the number of occurrences of each deal should be relatively close after a large number of hands.

Regardless of the reasons, it appears that it's more difficult to converge to the best-response in all cases than might be expected. The lack of convergence does not diminish the many interesting results which were demonstrated in this chapter.

One important result is that among a set of data-collection strategies which have a fixed exploitability, there can be a wide range of exploration values. Thus not all equilibrium strategies are equally valuable, even though each achieves the same payoff-rate against a competent opponent (one who does not use superfluous strategies). In the Kuhn Poker experiments shown, the P1 equilibrium strategies corresponding to $\gamma = 0.75$ and $\gamma = 1.0$ are efficient in collecting data for opponent models. On the other hand, the equilibrium strategy corresponding to $\gamma = 0$ can produce terrible models, as it gains no data on P2's η parameter, as it never puts P2 into a situation where the parameter applies. The goal of finding solutions to games is to find some strategy that is the least exploitable; when solving games in the future, one should not be satisfied with simply finding a solution, but should also try to find a solution with a high exploratory value.

One question that is difficult to answer is whether an exploitable strategy should be used to collect data, or whether the modeller should play it safer and stick with equilibrium strategies to collect data. Although the use of exploratory data-collection strategies does put the modeller at risk of losing more than he would by playing equilibrium strategies, the information gained be the use of exploratory strategies can often result in higher expected total winnings. Additionally, the exploratory strategy used may win more than the equilibrium rate against the opponent. On the other hand, there are certain opponents which will exploit a modeller using an exploratory strategy, and the winnings the modeller loses during the data-collection period may be impossible to recuperate. The guarantee of the equilibrium expected payoff-rate is a very nice property of the equilibrium data-collection methods.

In the experiments shown here, the use of dominated strategies to explore did not improve the learning rate. This is because the use of the dominated strategy sacrificed exploration of another opponent parameter. In larger games there exist dominated strategies which do not make this sacrifice, and it is expected that they would improve the learning rate. However, the use of dominated strategies is very likely to lose money, so the value of the improved model may be offset by the losses incurred during data-collection.

Another interesting result in this chapter is that the switching hand that maximizes the expected total winnings for the modeller doesn't seem to change much from one testpoint to the next. In particular, hand 50 is a reasonable switching hand for every test opponent when using a good equilibrium data-collection strategy; when using an exploratory data-collection strategy, the maximal switching hand often occurs earlier, as the losses of the data-collection strategy quickly outweigh the diminishing gains of improvements to the model.

One final result that may have gone unnoticed is that the use of explicit modelling with a good equilibrium data-collection method has shown at least a minor improvement against every test opponent over the alternative of strictly playing an equilibrium strategy. Thus the use of explicit modelling has not produced any results which suggest that these methods should not be used.

The next chapter explores the idea of implicitly modelling an opponent and examines the effectiveness of such methods in experiments.

Chapter 5

Implicit Modelling

5.1 Introduction

While the last two chapters have focused on explicitly modelling an opponent, this chapter is devoted to the task of implicitly modelling an opponent. Explicit modelling involves identifying the opponent's strategy in order to determine a counter-strategy that takes advantage of the opponent's mistakes. Implicit modelling simply involves identifying a good counter-strategy while being oblivious to the precise nature of the mistakes the opponent is making. The opponent modelling methods used in this chapter will try different strategies against the opponent, evaluate each strategy's performance, and recommend the counter-strategy with the highest score.

The methods discussed in this chapter are derived from the Hedge and Exp3 algorithms designed by Auer et al [2]. These algorithms were developed for multi-armed bandit problems[1], in which one player has to choose which of $\mathcal{K}$ different slot machines to play and receives some reward from the chosen machine; the opposing player gets to choose what reward is available for each machine prior to the first player's choice. This is analogous to a matrix game where prior to the game being played the column player chooses which column he is going to play; the row player's possible rewards are then set, and he receives the reward corresponding to the row he chooses to play.

Implicit modelling will consist of two phases: exploration, where various counter-strategies are used against the opponent and evaluated, and exploitation, where the highest-scoring strategy (or strategies, if two or more strategies are tied with the highest score) are played against the opponent. As in the explicit modelling case, a major concern of implicit modelling is when to switch from exploration to exploitation. If the modeller chooses to exploit

[1] Coin-operated gambling machines (called slot machines in this thesis) that are played by inserting a coin and pulling a handle (an "arm") on the side of the machine have been nicknamed "one-armed bandits", due to the large amounts of money taken in by the machines from players.

his model too soon, then he may have a counter-strategy which does not perform well against the opponent. However, if the modeller waits too long to begin exploiting his opponent, he may not have enough time to recover the losses incurred during the exploration phase. The Exp3 and Hedge algorithms allow the modeller to adjust his strategy during the data-collection phase to use a more exploitive data-collection strategy, and possibly incur fewer losses, as well as focus data-collection on promising counter-strategies. The two parameters in the Exp3 and Hedge algorithms, ψ and ρ play key roles in the exploration versus exploitation issue. The ψ parameter in the Exp3 algorithm controls the amount of uniform exploration among potential counter-strategies used by the modeller while exploring. The ρ parameter controls the amount of emphasis placed on playing the highest-rated strategies during the exploration phase.

Because the algorithms created by Auer et al. were designed for a slightly different problem than the one studied in this thesis, there are some modifications which can be made to improve the performance for this problem. These modifications are discussed in Section 5.3. This chapter will proceed by describing the algorithms presented by Auer et al., and show experimentally that the algorithms are inadequate for the problem being studied. The modified algorithms are then presented with justifications for the modifications, along with further experimental results which verify that the modifications improve the performance of the basic algorithms.

5.2 The Hedge and Exp3 Algorithms

The Hedge algorithm is designed for the full information multi-armed bandit problem. For this problem, each trial consists of the adversary first setting the rewards available to each slot machine, and then the player choosing one slot machine to play; the player then receives the corresponding reward. In the full information problem the player also learns what rewards were available at each of the other slot machines.

The Hedge and Exp3 algorithms are given below. Note that the parameter γ in Exp3 has been changed from the original publication to ψ to avoid confusion with the Kuhn Poker γ parameter.

```
Algorithm 1: Hedge
1. Parameter: a real number $\rho > 0$
2. Initialization: Set $W_i(0) = 0$ for $i = 1, \ldots, \mathcal{K}$ ($\mathcal{K}$ is the number of experts)
3. Repeat for $t = 1, 2, \ldots$ until match ends
    (a) Choose action $i_t$ according to the distribution $p(t)$, where
            $p_i(t) = \frac{(1+\rho)^{W_i(t-1)}}{Z_t}$
        where $Z_t$ is a normalizing factor that ensures that the probabilities sum to
            one:
            $Z_t = \sum_{j=1}^{\mathcal{K}} (1+\rho)^{W_j(t-1)}$
    (b) Receive a reward vector $\vec{x}(t)$, where $x_i(t) \in [0, \inf)$ for $i = 1, \ldots, \mathcal{K}$
    (c) Set $W_i(t) = W_i(t-1) + x_i(t)$ for $i = 1, \ldots, \mathcal{K}$
```

The basic idea of the Hedge algorithm is to focus play towards actions which have resulted in high rewards in the past. This is accomplished by keeping running totals of the rewards assigned to each action, and then assigning a probability to playing each action that is exponentially related to the total for it. The ρ parameter controls how much emphasis is put on playing the action with the highest cumulative reward. A large ρ (eg. $\rho = 100$), will cause the player to choose among only the actions with the highest cumulative reward on each trial with very high probability, while a small ρ (eg. $\rho = 0.01$) will cause the player to choose among all of the actions in each round with near-uniform probability.

The Exp3 algorithm is designed for the partial information bandit game, where the player only receives information about the reward for the slot machine he chooses to play. The basic idea of the algorithm is to simulate the full information game by supplying Hedge with a reward vector that contains the actual reward received for the chosen action (scaled by the probability of choosing the action) and rewards of value 0 for the actions not chosen. Since information is only gained about one action for each trial, the Exp3 algorithm contains the parameter ψ which ensures that the player will not just pick one action throughout the trials; otherwise if all rewards are nonnegative the machine chosen in the first trial could immediately start out with a nearly insurmountable lead, being repeatedly chosen since it has the only nontrivial probability. A setting of $\psi = 1$ corresponds to all actions being chosen from uniformly on every round, while the setting $\psi = 0$ corresponds to the most exploitive case where experts that have been used and well-rewarded in the past are highly favoured to be chosen repeatedly.

```
Algorithm 2: Exp3
1. Parameters: reals $\rho > 0$ and $\psi$, $0 \leq \psi \leq 1$
2. Initialization: Initialize Hedge($\rho$)
3. Repeat for $t = 1, 2, \ldots$ until match ends
    (a) Get distribution $p(t)$ from Hedge
    (b) Select action $i_t$ to be $j$ with probability $\hat{p}_j(t) = (1 - \psi)p_j(t) + \frac{\psi}{\mathcal{K}}$
    (c) Receive reward $x_i(t) \in [0, 1]$
    (d) Feed simulated reward vector $\hat{x}(t)$ into Hedge, where
        $\hat{x}_j(t) = \left\{ \begin{array}{ll} \frac{x_i(t)}{\hat{p}_i(t)} & \text{if } j = i_t \\ 0 & \text{otherwise} \end{array} \right\}$
```

The reward is scaled (divided by the probability of choosing the action taken) so that the expected total reward for each action equals the total reward for the full information game. The expected total reward for expert j after T rounds is:

$$\begin{aligned} EV[\sum_{t=1}^{T} \hat{x}_j(t)] &= \sum_{t=1}^{T} EV[\hat{x}_j(t)] \\ &= \sum_{t=1}^{T} \left(\hat{p}_j(t) \frac{x_j(t)}{\hat{p}_j(t)} + (1 - \hat{p}_j(t))0 \right) \\ &= \sum_{t=1}^{T} x_j(t) \end{aligned}$$

What this means is that the total of the scaled rewards for action j after T hands approximates the total of the unscaled rewards that would be achieved if action j was taken every hand. Note that the scaling of rewards prohibits the setting $\psi = 0$, since in this case it would often be the case that some actions would have a probability near zero of being selected. On the rare occasion that such an action is selected, that action would receive an extremely large reward (due to the division by a miniscule probability), leading to numerical instabilities in the algorithm.

One of the nice properties of the Exp3 algorithm, is that it has been shown to have an average external regret that converges to zero (when the set of actions is all possible actions). Recall that external regret is the difference between the rewards achieved by the algorithm and the rewards achieved by the best pure strategy played against the opponent. In the paper introducing Exp3 [2], the authors prove bounds that external regret for Exp3 at time T is $O(\sqrt{T\mathcal{K}\log\mathcal{K}})$.

In an abbreviated variant of the original paper, published in 2000, there are minor modifications made to the algorithms [3]. Hedge only accepts rewards in the interval $[0, 1]$ and Exp3 must scale the rewards fed into Hedge by $\psi/\mathcal{K}$ to meet this restriction. These do not seem to be necessary changes as all of the bounds were thoroughly proved in the original publication, and the changes are equivalent to setting $1 + \rho_{1995} = (1 + \rho_{2000})^{\mathcal{K}/\psi}$. Thus the algorithms are equivalent with different parameter settings. The changes appear to have been made to simplify the proof of regret bounds for the Hedge algorithm. The implementations presented here are based on the algorithms published in the original paper [2].

In the multi-armed bandit problem, the player takes only a single action in each game. This is nearly equivalent to choosing a single fixed strategy at the beginning of a larger game that consists of multiple moves for each player. Once each player's strategy is fixed, the average outcome of the game is determined, as it is just the weighted sum of the rewards available at each leaf of the game tree multiplied by the probability of reaching that leaf. Unfortunately the modeller does not get to observe the average outcome, but instead observes a single outcome based on chance (the deal) and from both players using their strategies to make decisions over the course of one hand. For the purpose of discussing these multiple-move games, Exp3 and Hedge's action sets will instead be referred to as the set of *expert strategies* or *experts*. The strategy that is selected in step 3.b (for hand t) will be denoted e_t.

5.2.1 Experimental Setup

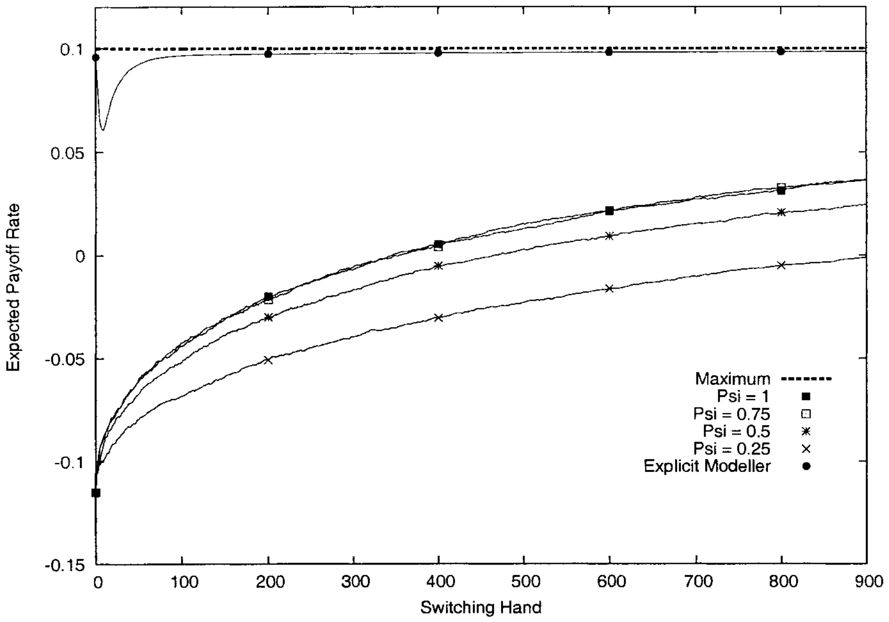
For the problem of P1 modelling P2 in Kuhn poker, P1 will have seven possible expert strategies for each trial in the experiments shown here. Six of these experts will be the possible best-response strategies which partition the P2 strategy space, as shown in Figure 2.5 in Chapter 2; the seventh expert is the P1 equilibrium strategy with $\gamma = 0.5$, $\alpha = 0.167$ and $\beta = 0.5$. This means that for every P2 instance that P1 faces, the best-response strategy is among P1's possible counter-strategies, as well as a safe equilibrium strategy if P1 has difficulty identifying a stronger pure counter-strategy. The test points listed in Section 4.2.1 are reused for these experiments.

The performance of the Exp3 algorithm depends heavily on the choice of the parameters ψ and ρ. In the experiments listed here, results will be shown for ψ having each of the values $\{0.25, 0.5, 0.75, 1\}$ for the setting $\rho = 1$. The effect of changing ρ will be studied in Section 5.7. Results shown are averaged over 30000 trials for each method discussed against each test opponent. A single trial consists of a 900-hand match, where the player being modelled uses a fixed stochastic strategy while the modeller's data-collection strategy changes in accordance with ψ, ρ, and the observed rewards. Results are primarily shown for the testpoints O_2 and O_6 as these points have highly contrasting results for the methods studied.

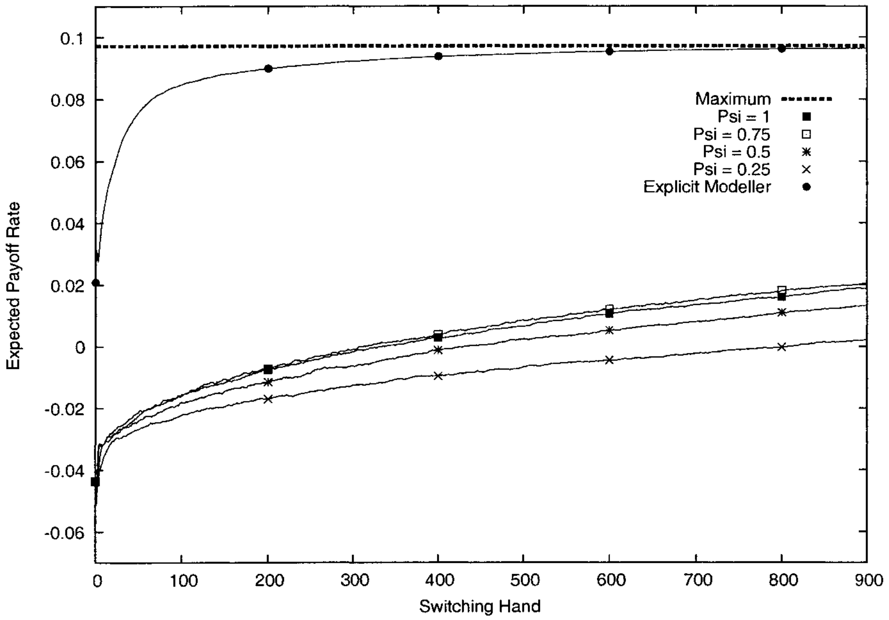
The charts shown will be the payoff-rate, total winnings, and proportion above equilibrium plots introduced in the previous chapter. Payoff-rate plots show the expected payoff-rate the modeller would recieve after hand t (referred to as the switching hand) if he switched at this point to only playing the strategy which has the highest score. In the event that ℓ strategies are tied with the highest score, the counter-strategy used is to play the mixed strategy with each of the tied strategies having the probability $1/\ell$ of being played. Total winnings plots show what total winnings the modeller would expect to achieve if he switched to the expert(s) with the highest score immediately after the switching hand in a fixed-length match, and is computed as the sum of the winnings achieved in exploring up until the switching hand plus the expected winnings over the remainder of the match. Proportion above equilibrium plots show the proportion of trials that would expect to win more than the equilibrium value in a 200-hand match if they switched to the highest scoring expert(s) at the switching hand.

The payoff-rate plots shown in Figure 5.1 show that for any setting of ψ, the Exp3 algorithm has a very slow convergence to the best-response payoff-rate. The explicit modelling method with equilibrium ($\gamma = 0.75$) data-collection has been plotted in all results plots in this chapter for comparison. When comparing the results of the Exp3 algorithm and the explicit modelling method, it is clear that the Exp3 algorithm is very far behind. The initial payoff-rate of the Exp3 algorithms is the average payoff-rate of all seven experts (each expert is tied with the highest score at hand 0); the initial payoff-rate of the explicit modelling method is the counter-strategy to the initial estimates ($\eta = 0.5, \xi = 0.5$). Although the initial payoff-rate for the explicit modelling method is much higher for the two opponents shown here, this initial head start is not responsible for the superior performance of the explicit modelling methods. Experiments are shown in Section 4.2.5 for which the explicit modelling methods have a bad initial payoff-rate but still quickly converge to the best-response rate.

The total winnings plots in Figure 5.2 show that using Exp3 would achieve close to the equilibrium payoff rate against these opponents. The total winnings plots are much flatter than the corresponding explicit modelling plots, because the Exp3 data-collection strategy is partially exploitive when $\psi < 1$ (the amount of exploitation depends on the setting of ρ and ψ). This suggests that a player using Exp3 would have a much larger interval over which he could switch from exploration to exploitation, without losing a great deal in total

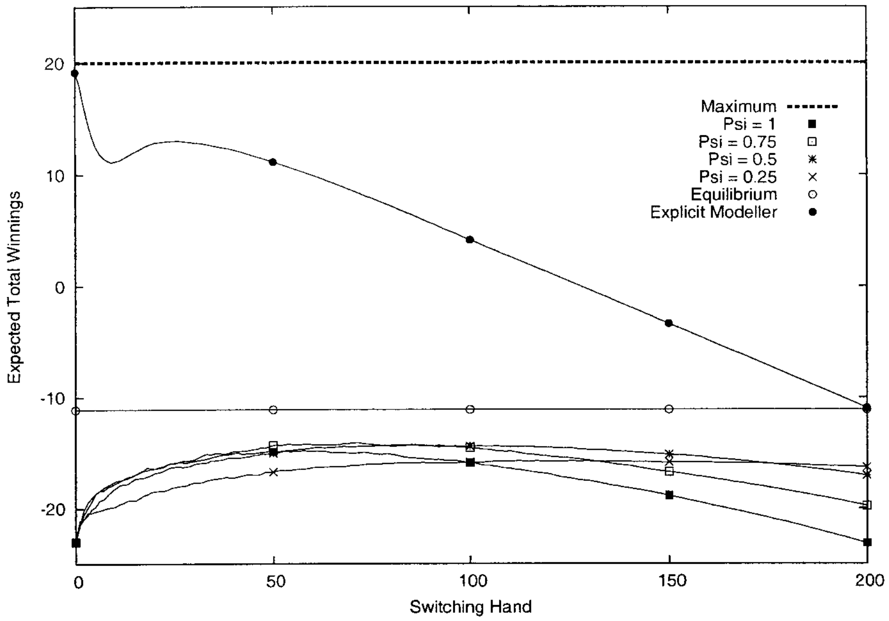


(a) $O_2 = (0.75, 0.8)$

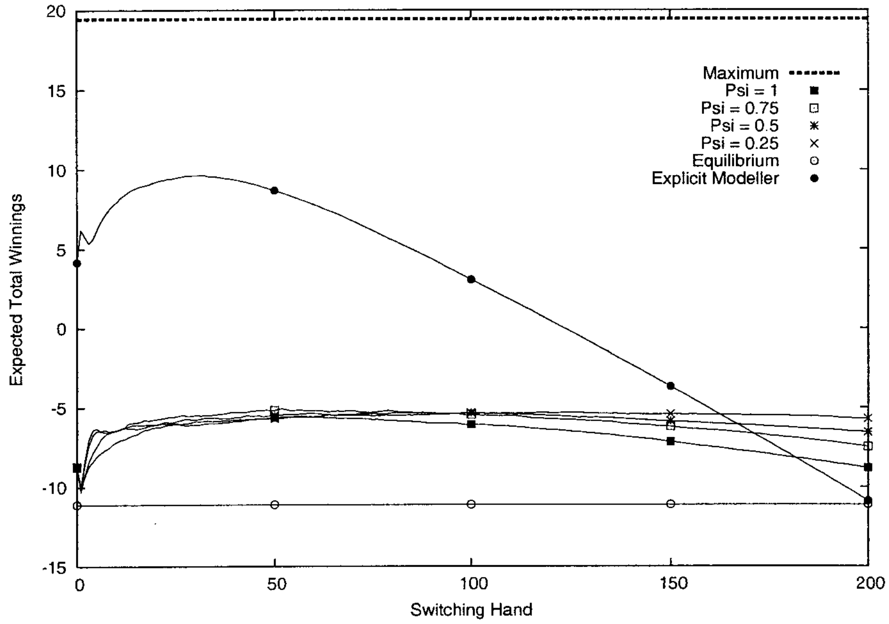


(b) $O_6 = (0.25, 0.67)$

Figure 5.1: Exp3 Method ($\rho = 1.0$) Payoff-Rate Plots

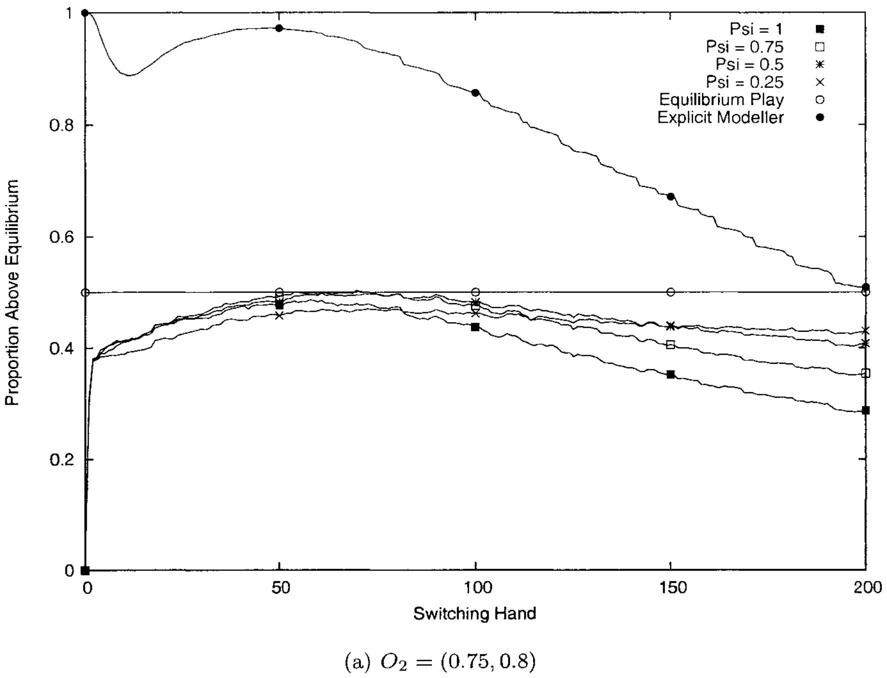


(a) $O_2 = (0.75, 0.8)$

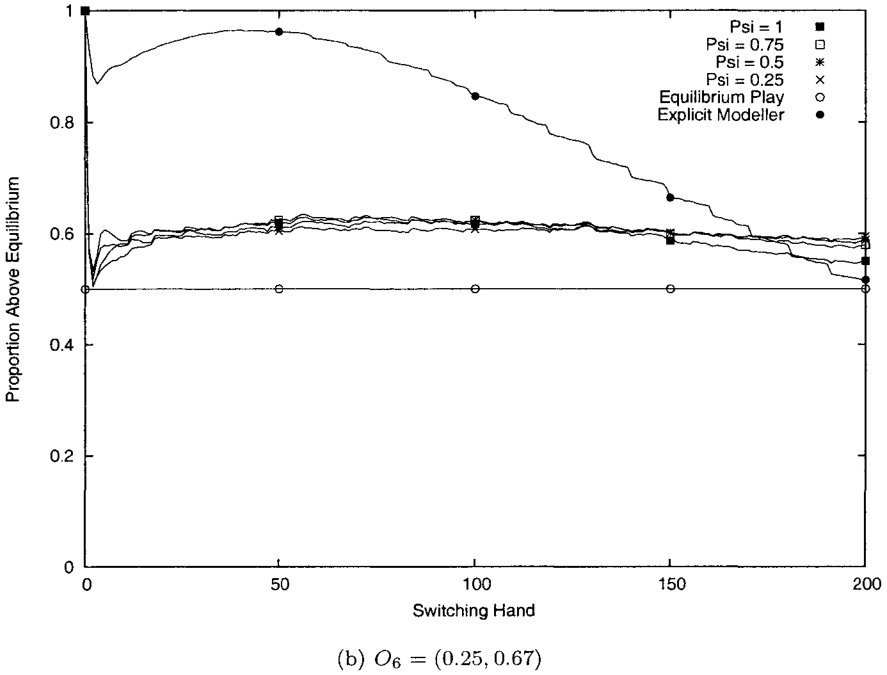


(b) $O_6 = (0.25, 0.67)$

Figure 5.2: Exp3 Method ($\rho = 1.0$) Total Winnings Plots



(a) $O_2 = (0.75, 0.8)$



(b) $O_6 = (0.25, 0.67)$

Figure 5.3: Exp3 Method ($\rho = 1.0$) Proportion Above Equilibrium Plots

winnings.

The proportion above equilibrium plots in Figure 5.3 further show that the Exp3 method achieves close to equilibrium against these opponents (recall that playing equilibrium strategies only achieves the equilibrium payoff-rate or higher in 50% of trials).

Similar results also hold for other test opponents and for the case of P2 modelling P1.

5.3 Improving the Algorithms

Although the Exp3 algorithm has been shown to have an average external regret that converges to zero, the results shown for the algorithm suggest that this convergence is slow for this application. Since this thesis is more concerned with short-term results, modifications must be made to improve the Exp3 algorithm for the method of implicit modelling to be a viable alternative to explicit modelling.

5.3.1 Sharing Rewards between Agreeing Strategies

One key observation that can be made about the game of Kuhn Poker is that for each individual hand, the modeller is only dealt one of the three potential holdings that he could have. Despite the fact that the expert strategies differ from one another over how to play the entire game of Kuhn Poker, several experts could agree on what to do for the decisions that were faced by the expert, e_t, chosen to play hand t. Since several expert strategies may have acted the same as e_t, it is logical to reward the agreeing strategies as well as the chosen expert. This idea results in a variation of Exp3 called SharingExp3 which has two distinct differences from the normal Exp3 algorithm. The first difference is that multiple strategies may have a nonzero reward in the simulated reward vector fed into Hedge. The second difference is that the reward is no longer scaled by the probability of choosing e_t, but is instead scaled by the probability of choosing any one of the agreeing experts. The SharingExp3 algorithm is defined fully as Algorithm 3:

```
Algorithm 3: SharingExp3
1. Parameters: reals $\rho > 0$ and $\psi$, $0 \le \psi \le 1$
2. Initialization: Initialize Hedge($\rho$)
3. Repeat for $t = 1, 2, \ldots$ until match ends
    (a) Get distribution $p(t)$ from Hedge
    (b) Query each expert strategy for which pure strategy they recommend on
        this round.
    (c) Select pure strategy $e_t$ to be the pure strategy recommended by expert
        $j$ with probability $\hat{p}_j(t) = (1 - \psi)p_j(t) + \frac{\psi}{K}$
    (d) Observe game sequence $D_t$ and receive reward $x_{e_t}(t) \in [0, 1]$
    (e) Let $\mathcal{E}_t$ be the set of experts which recommended a pure strategy that
        would have made the same decisions $e_t$ did to generate the game sequence
        $D_t$. Compute $q_t(D_t)$, the probability of generating $D_t$ given the
        adversary's decisions are fixed:
            $q_t(D_t) = \sum_{j \in \mathcal{E}_t} \hat{p}_j(t)$
    (f) Feed simulated reward vector $\hat{x}(t)$ into Hedge, where
            $\hat{x}_j(t) = \left\{ \begin{array}{ll} \frac{x_{e_t}(t)}{q_t(D_t)} & \text{if } j \in \mathcal{E}_t \\ 0 & \text{otherwise} \end{array} \right\}$
```

The "purification" of experts in step 3.b of Algorithm 3 is done to make the sharing of rewards simpler, as the scaling factor for the reward is easy to compute, and each of the agreeing experts gets an equal share of the reward. However, it is also possible to avoid this purification step by considering expert i to be in partial agreement with the pure strategy e_t used if i would choose a pure strategy agreeing with e_t with probability $a_i(e_t)$. In this case the probability of choosing an agreeing pure strategy is

$$q_t(D_t) = \sum_{j}^{\mathcal{K}} \hat{p}_j(t) a_j(e_t),$$

and the reward given to expert i is

$$\hat{x}_i(t) = a_i(e_t) \frac{x_{e_t}(t)}{q_t(D_t)}.$$

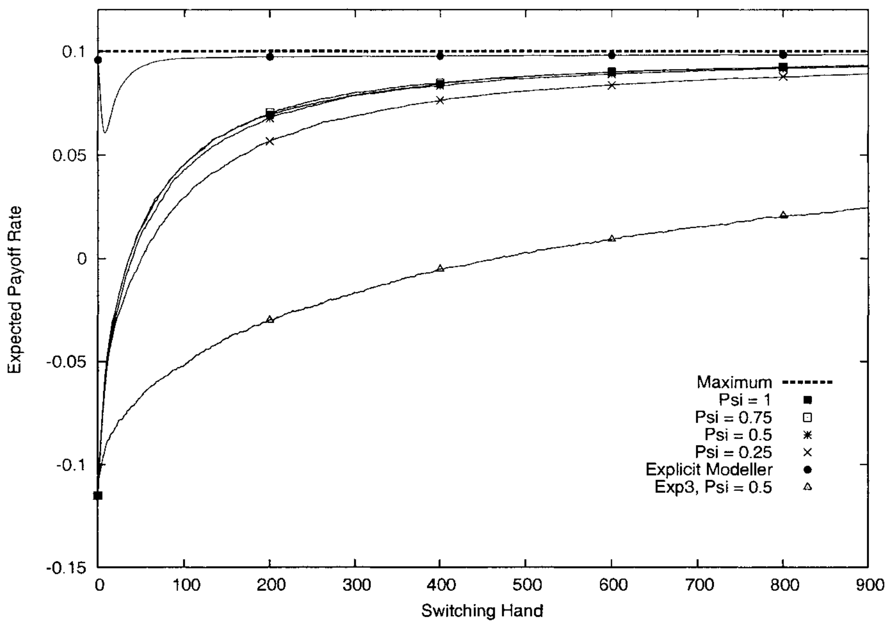
The effectiveness of this method of giving partially agreeing experts a partial reward is left for future studies. It is unlikely that this method would greatly affect the results in this thesis since the set of experts contains only one mixed strategy (the equilibrium strategy corresponding to $\gamma = 0.5$).

The sharing among partially agreeing experts done by SharingExp3 seems to be in the spirit of the Exp4 algorithm proposed by Auer et al [2]. The Exp4 algorithm considers the case where there are $\mathcal{N}$ experts, where each expert suggests playing a probability distribution over the $\mathcal{K}$ actions in the $\mathcal{K}$-armed bandit problem. The experts' distributions are added together to obtain a single distribution over the actions and then a single action is selected. All experts that suggest playing the selected action with nonzero probability are given a partial reward. The full Exp4 algorithm is given below for comparison, although it has not been implemented for the problem being studied here. Some of the indices and parameters used have been changed here from the original publication to avoid confusion with parameters used in this thesis.

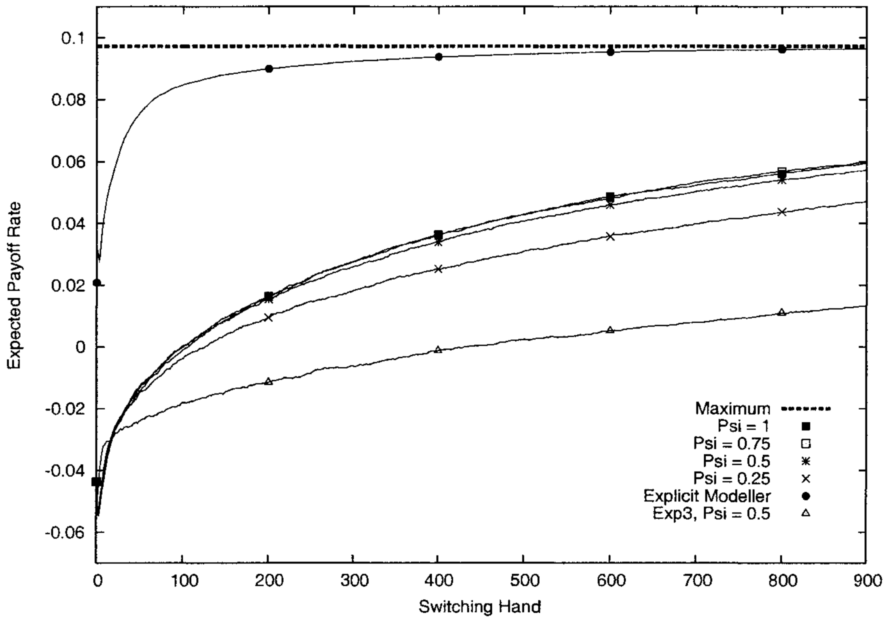
```
Algorithm 4: Exp4
1. Parameters: Reals $\rho > 0$ and $\psi \in [0, 1]$
2. Initialization: Initialize Hedge($\rho$) (with $\mathcal{K}$ replaced by $\mathcal{N}$)
3. Repeat for $t = 1, 2, \ldots$ until match ends
    (a) Get the distribution $\mathbf{q}(t) \in [0, 1]^{\mathcal{N}}$ from Hedge, where $q_n(t)$ is the probability of choosing expert $n$.
    (b) Get advice vectors $\mathbf{\Delta}^n(t) \in [0, 1]^{\mathcal{K}}$ representing expert $n$'s probability distribution over the $K$ actions. Let $\mathbf{p}(t) := \sum_{n=1}^{\mathcal{N}} q_n(t)\mathbf{\Delta}^n(t)$, so that $p_j(t)$ is the overall probability of choosing action $j$.
    (c) Select action $i_t$ to be action $j$ with probability $\hat{p}_j(t) = (1 - \psi)p_j(t) + \frac{\psi}{\mathcal{K}}$.
    (d) Receive reward $x_{i_t}(t) \in [0, 1]$.
    (e) Compute the simulated reward vector $\hat{\mathbf{x}}(t) \in [0, \mathcal{K}/\psi]^{\mathcal{K}}$ as
        $\hat{x}_j(t) = \left\{ \begin{array}{ll} \frac{x_{i_t}(t)}{p_{\hat{i}_t}(t)} & \text{if } j = i_t \\ 0 & \text{otherwise.} \end{array} \right\}$
    (f) Feed the vector $\hat{y}(t) \in [0, \mathcal{K}/\psi]^{\mathcal{N}}$ into Hedge, where $\hat{y}_n(t) \doteq \mathbf{\Delta^n(t)} \cdot \hat{\mathbf{x}}(\mathbf{t})$. This results in distributing the reward among the expert strategies which choose $i_t$ with nonzero probability.
```

There are a few key differences between SharingExp3 and Exp4. SharingExp3 is meant for a game where only a small part of the game tree is traversed on each hand. Thus several pure strategies for the entire game agree on the actual subgame played and these strategies share the reward. Exp4 is meant for a matrix game where experts are mixed strategies; the experts which contribute to the probability of choosing the actual row selected share the reward (in proportion to the amount contributed by the expert).

The payoff plots shown in Figure 5.4 show that the sharing of rewards greatly speeds up the convergence to the maximum payoff rate. The Exp3 algorithm with $\rho = 1.0$ and



(a) $O_2 = (0.75, 0.8)$



(b) $O_6 = (0.25, 0.67)$

Figure 5.4: SharingExp3 Method ($\rho = 1.0$) Payoff-Rate Plots

$\psi = 0.5$ has been plotted for comparison. For the opponent O_2, the SharingExp3 algorithm converges to being near the maximum payoff-rate of 0.1 \$/hand after 900 hands, while the Exp3 series with the highest payoff-rate results only achieves around 0.04 \$/hand after the same number of hands (note that the $\psi = 0.5$ Exp3 series plotted does not achieve this rate but a different ψ setting in Figure 5.1 does).

The total winnings plots in Figure 5.5 show a minor improvement of the SharingExp3 algorithm over Exp3 against O_6 and a huge improvement against O_2. Similar to the Exp3 total winnings plots, the SharingExp3 plots are much flatter in comparison to the total winnings plots for the explicit modelling methods in Chapter 4, meaning a player using SharingExp3 has a large interval within which to change from exploration to exploitation without sacrificing a great deal of total winnings.

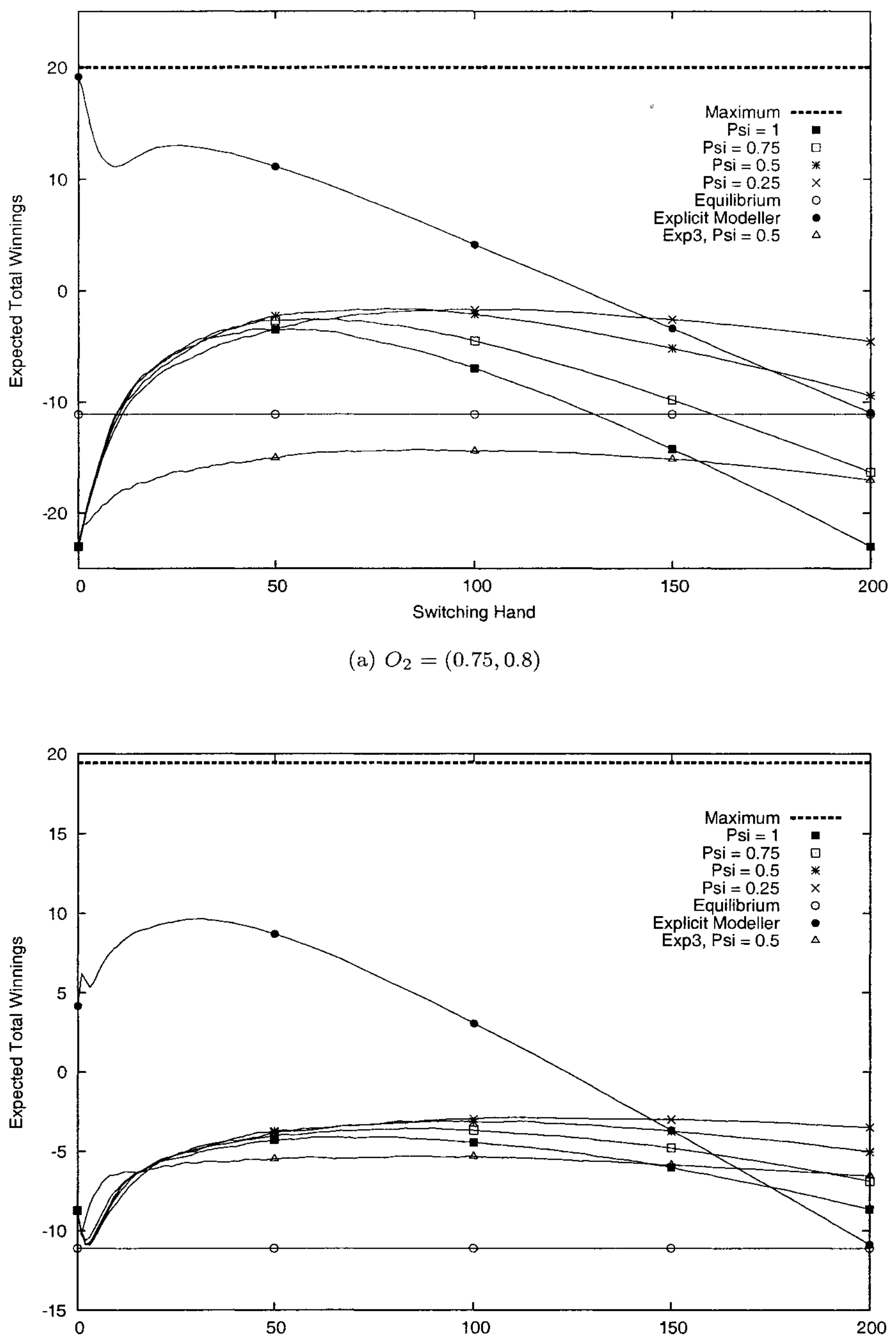
Figure 5.6 shows that for both of the opponents shown, the SharingExp3 algorithm outperforms the equilibrium payoff rate in over 60% of the trials at hand 75. These plots suggest that using SharingExp3 and switching to exploitation at hand 75 is often favourable to using an equilibrium strategy for the entire match, as an equilibrium playing strategy achieves the equilibrium rate or greater only 50% of the time.

5.3.2 Inference

The improved performance of SharingExp3 over Exp3 inspires the search for further modifications that produce large benefits. Since updating experts more often improved SharingExp3, it seems logical to continue this process. In SharingExp3, the experts that agree with the chosen expert are the ones that are updated in every round. Sometimes it is possible to infer what would have happened if an alternative action had been taken, which means it's possible to update experts outside the agreeing set on some rounds.

An example of inference occurs when P1 passes when holding the Jack in Round One and observes a P2 bet in Round Two. Because P2 bet in Round Two, P1 can deduce that P2 held the King and can then infer that if P1 had taken the alternative action of betting in Round One he would have lost \$2 (P2 would have called with the King). Likewise, if P1 bets in Round One and loses \$2 when P2 calls with the King, P1 can infer that he would have lost \$1 by passing. An example where inference cannot be performed occurs when P1 passes in Round One with the Jack and P2 passes in Round Two with the Queen. In this case P1 cannot infer what would have happened if he were to have bet with the Jack in Round One.

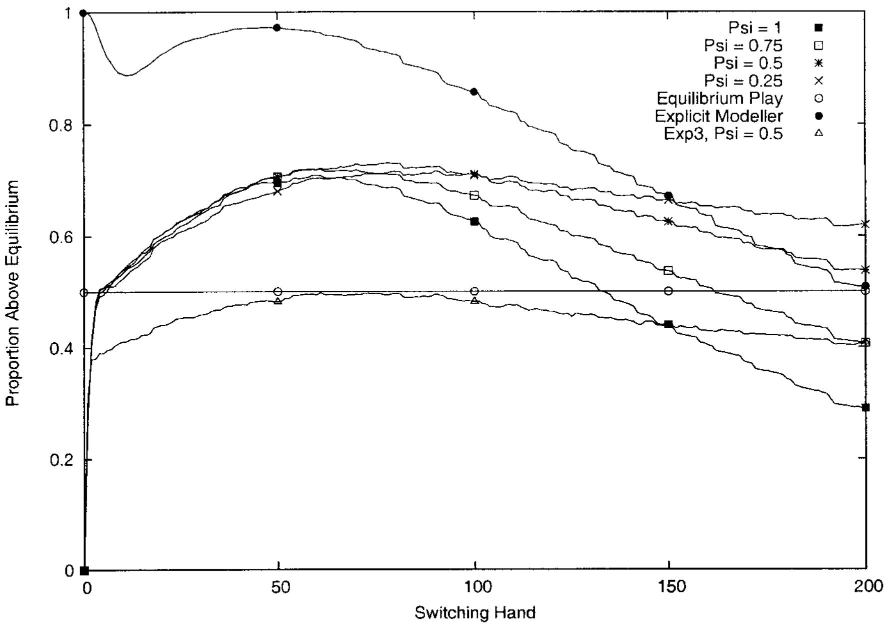
Unfortunately, the inference method suffers from a problem of imbalanced updates, which means some experts are updated for holdings in proportions which are greatly out of balance



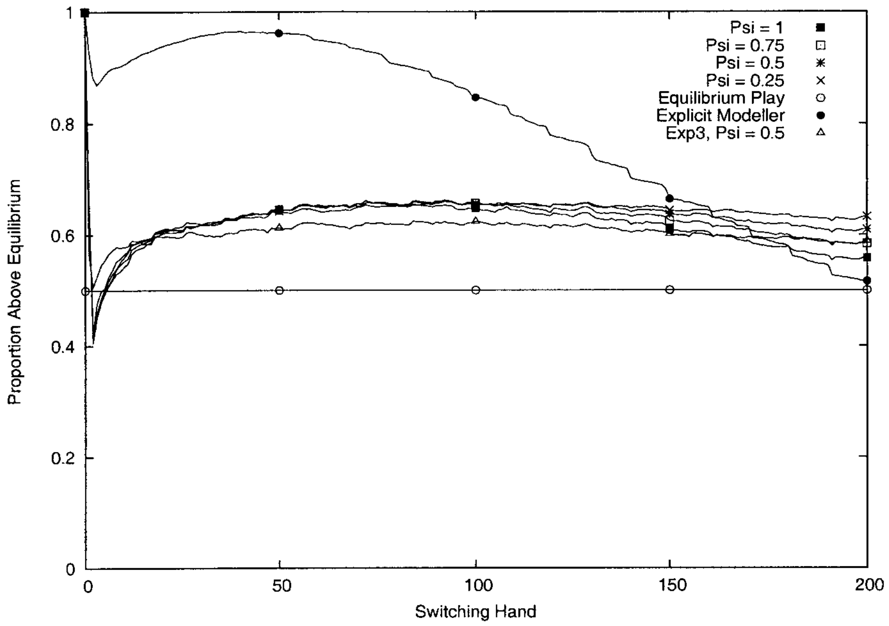
(a) $O_2 = (0.75, 0.8)$

(b) $O_6 = (0.25, 0.67)$

Figure 5.5: SharingExp3 Method ($\rho = 1.0$) Total Winnings Plots



(a) $O_2 = (0.75, 0.8)$



(b) $O_6 = (0.25, 0.67)$

Figure 5.6: SharingExp3 Method ($\rho = 1.0$) Proportion Above Equilibrium Plots

relative to the probability of the holdings. For the following example, assume P1 performs inference when the J|K deal is identified, but no inference is done for any other deals. When the *pbp* sequence occurs, P1 infers a loss of \$2 for the action of betting instead of passing in Round One. When the *bb* sequence occurs and P2 shows that he holds the King in the showdown, P1 infers a loss of \$1 for the action of passing in Round One. Furthermore, suppose P1 bets or passes with the Jack in Round One with equal probability (thus all rewards are scaled by the same probability, and the scaling can be ignored for now). Figure 5.7 shows hypothetical observations that P1 has made after 20 hands of holding the Jack; the numbers in the leaf nodes represent the number of times P1 has observed those leaf nodes, while the payoffs for each node are below the nodes. There have been ten hands where the J|K deal occurred, five of which P1 bet in Round One and lost \$2 and five of which P1 passed and lost \$1. There have also been ten hands where the J|Q deal occurred, five of which P1 bet (and saw one P2 call and four passes), and five which P1 passed and went to a showdown. The observations of P2 folding four times with the Queen and calling once suggest that P2's η may be lower than 1/3 and the best counter-strategy to such a parameter setting is to always bet with the Jack in Round One.

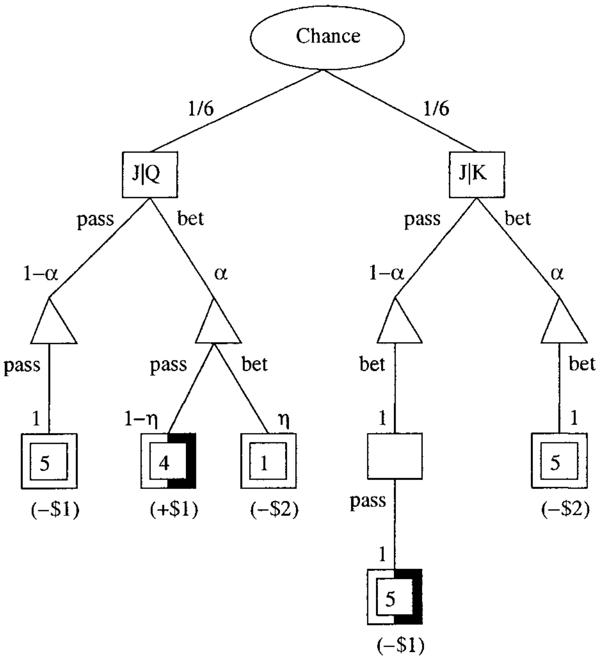


Figure 5.7: Observations made by P1 without inference

Without inference, betting has five hands of J|K data (-\$10 total) and five hands of J|Q data (+\$1 for each pass and -\$2 for each call, +\$2 total), giving a cumulative reward of -\$8.

Passing has five hands of J|K data (-$5 total) and five hands of J|Q data (-$5 total), giving a cumulative reward of -$10. Without inference, betting with the Jack in Round One is correctly identified as the best counter-strategy.

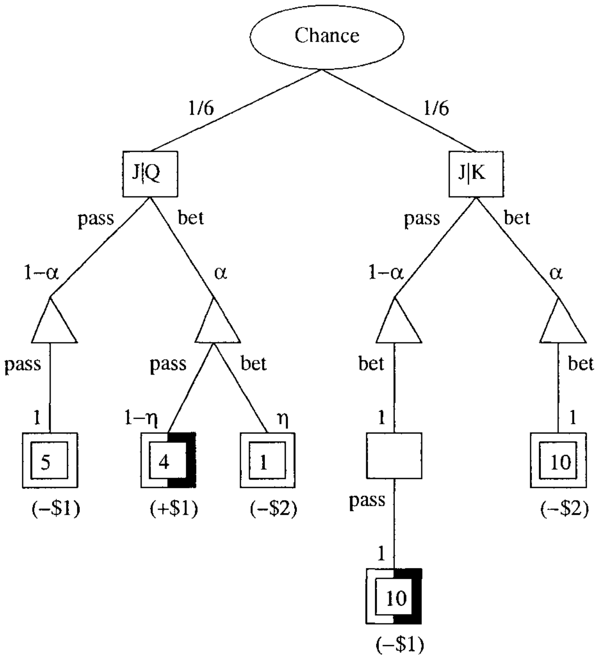


Figure 5.8: Observations made by P1 (actual and inferred observations)

Figure 5.8 shows the observation counts for P1 when he uses inference. With inference, betting has ten hands of J|K data (-$20 total) and five hands of J|Q data (+$2 total), which gives a cumulative reward of (-$20 + $2) = -$18. With inference, passing has ten hands of J|K data (-$10 total) and five hands of J|Q data (-$5 total), which gives a cumulative reward of -$15. Thus with inference, passing is seen as the best option. The reason for this is that the experts have an imbalance in data; twice as much data is available for the J|K case than the J|Q case, suggesting to the experts that the J|K deal is twice as likely as the J|Q deal.

One idea to address this imbalance problem is to divide any rewards (observed and inferred) given to an expert by the probability of observing the situation directly plus the probability of inferring the situation. Unfortunately, inference by strategy j about strategy i often depends on the opponent taking a particular sequence of actions. This means the probability of inference depends on the probability of the opponent's actions, which is usually unknown (unless all of the opponent's alternative actions are dominated). This solution fixes the imbalance example introduced above, as each J|K datapoint in Figure 5.8 would be divided by 100% (inferences are made about the action not taken every time this deal occurs), while each J|Q datapoint would be divided by 50% (inferences were not made in the example when this deal occurred).

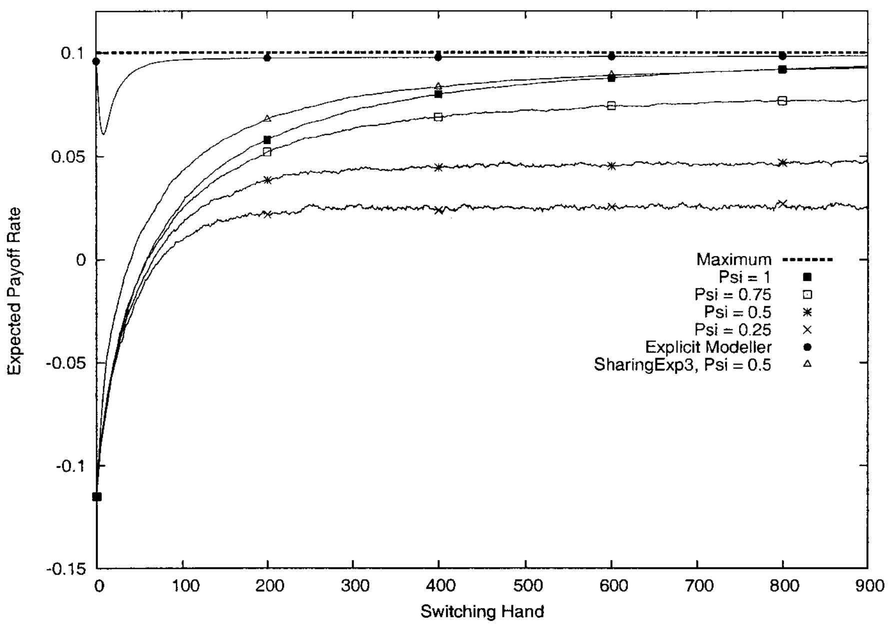
However, this solution also excludes other situations where inference would intuitively seem possible. For example, when the K|Q deal and the *bb* sequence occurs, it would seem possible to make use of the knowledge that if P1 had passed with the King in Round One, P2 would also have passed (it would be a dominated action for P2 to bet). However, this inference depends on P2 calling with the Queen after P1's initial bet (which is what actually occurred), and the probability of this event is unknown.

5.3.3 Average Rewards versus Cumulative Scaled Rewards

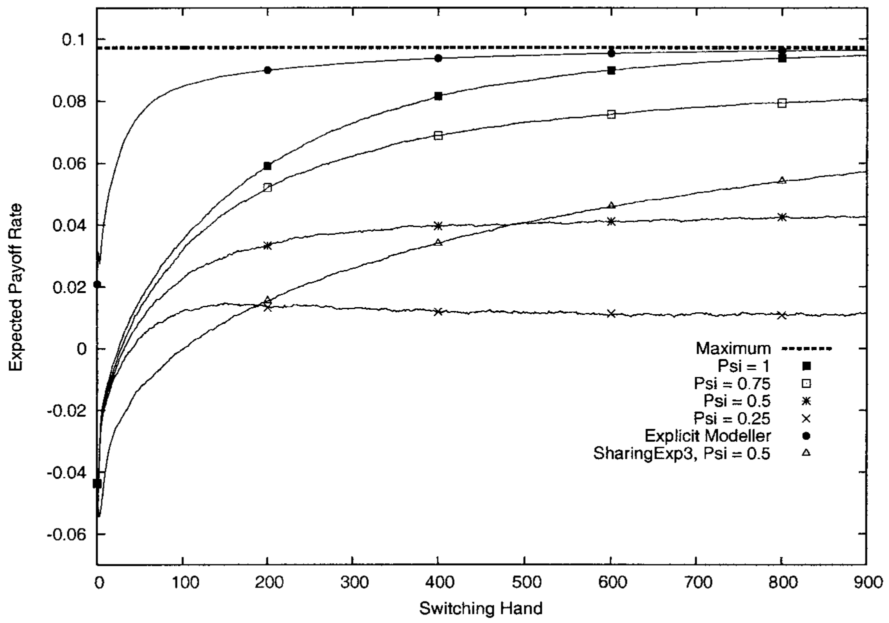
The motivation for scaling the rewards in both Exp3 and SharingExp3 is to compensate for the fact that certain experts may be chosen more often than others and thus have their cumulative winnings increased more often. The goal of scaling the rewards is for the cumulative winnings of expert j after hand t to be approximately the average reward (if j was rewarded on every round) for expert j multiplied by t. However, this depends on experts being chosen and updated as often as the probabilities suggest that they should be, which may not be the case in the short term. For example, suppose expert j_1 has a probability of being updated 20% of the time (typically this probability changes from one game to the next, but assume it is constant for this example), but over a 100-game period expert j_1 is only updated 15 times, then j_1's cumulative score will not be as high as it would be if it were updated with unscaled rewards 100 times. Conversely, if expert j_2 also has a probability of being updated 20% of the time and is updated 25 times in a 100-game period, then j_2 could have a higher cumulative score than j_1 while having a lower reward on average.

A modification to Exp3 to address this problem is to simply keep track of the number of times each expert has been updated and compute each expert's observed average unscaled reward. This algorithm is AverageExp3, which generates the payoff-rate plots in Figure 5.9. The results in these payoff rate plots are clearly worse than the SharingExp3 algorithm, except when $\psi = 1$ and experts are chosen from uniformly in each round. The reason for these poor results is that the AverageExp3 algorithm can also suffer from an imbalance problem.

The SharingExp3 method will also be applied for the following example, to show how it avoids the imbalance problem suffered by AverageExp3. In order to apply SharingExp3, the rewards have been converted to the interval $[0, 1]$ using the conversion formula $x_{[0,1]} = (x_{[-2,2]} + 2)/4$.



(a) $O_2 = (0.75, 0.8)$



(b) $O_6 = (0.25, 0.67)$

Figure 5.9: AverageExp3 Method ($\rho = 1.0$) Payoff-Rate Plots

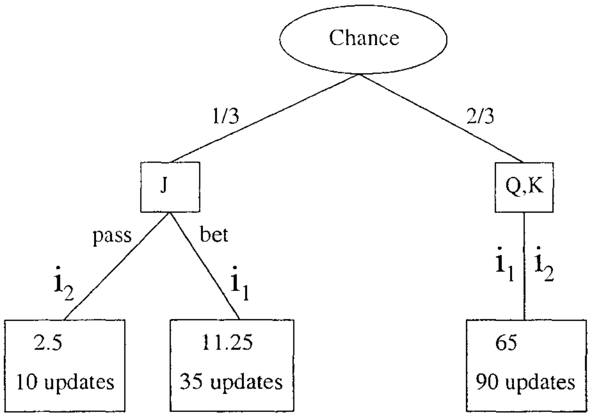


Figure 5.10: Observations made by P1

Suppose there are only two experts, i_1 and i_2, and they both advise the same action for all situations except when holding the Jack in Round One, in which case i_1 advises betting and i_2 advises passing. Figure 5.10 summarizes the observations made by P1. For the situations where the experts agree (ie. when the Queen or King is held), the two experts are updated 90 times for a total reward of 65. For the situation where they disagree, i_1 is updated 35 times for a total reward of 11.25, and i_2 is updated ten times for a total reward of 2.5. Note that i_1 has a higher average reward than i_2 when considering only actions taken with the Jack (i_1 has an average reward of 0.321 while i_2 has an average reward of 0.25), which means that i_1 should be the recommended expert.

The average reward for i_1 is (65 + 11.25))/(125 updates) = 0.61 units/update, while the average reward for i_2 is (65 + 2.5))/(100 updates) = 0.675 units/update. AverageExp3 would recommend i_2 as the best counter-strategy. The interesting thing is that i_1 has a higher average reward than i_2 when only the actions with the Jack are taken into consideration, and they both have the same average reward for the remaining situations, but i_2 has the higher overall average[2]. The reason for this is that 90% of i_2's average comes from holding strong cards (the King and the Queen), while only 72% of i_1's average comes from holding the strong cards. Thus AverageExp3 also suffers from a problem of imbalancing, as a bad expert can appear stronger than a good expert because the good expert was updated more often when the weak card was held. Typically this imbalance does not occur simply due to bad luck, which would be the case if both the good and bad experts were equally likely to be updated with the weak card, but the good strategy just happened to catch all the bad hands. The imbalance typically occurs because the good expert has a higher proba-

[2] This is an instance of Simpson's paradox: "It is not necessarily true that averaging the averages of different populations gives the average of the combined population." [42]

bility of being updated when the weak card is dealt. This occurs because the set of experts is partitioned into different agreeing sets for each card, meaning an expert can have different probabilities of being updated for each card (the probability that expert i is updated given card C is the sum of the probabilities of the experts in i's agreeing set for C).

This imbalance problem does not occur in SharingExp3 because of the reward scaling. When i_2 has a lower probability of being updated when the Jack is dealt, his reward when updated in this situation is scaled higher to compensate for the lower frequency of updates. Suppose for the hypothetical data in Figure 5.10 that i_1 has an 80% chance of being updated and i_2 has a 20% chance of being updated when the Jack is held. Both have a 100% chance of being updated when the Queen or King is held. Scaling the rewards by the probability of the observations gives the cumulative reward for i_1 of $((65/1.0)+(11.25/0.8)) = 79.0625$; the cumulative scaled reward for i_2 is $((65/1.0) + 2.5/0.2)) = 77.5$. SharingExp3 recommends i_1 as the best expert, although the margin of victory is pretty narrow due to the fact that i_2 has been updated slightly more often than the probabilities dictate.

In Kuhn Poker it is known that the probability of receiving each card is 1/3; this means that about 1/3 of each expert's score should come from holding each card. The poor results of the AverageExp3 algorithm are due to the fact that some experts can be much luckier than others, in terms of being updated more often with strong cards than with weak cards. This luck is then reflected in higher average rewards received by the lucky strategies than unlucky strategies. One way of reducing this luck factor is to compute separate averages for each expert for the cases of holding each card, and combine these averages to form the overall average. This method is used in the ComponentAverageHedge and ComponentAverageExp3 algorithms, Algorithms 5 and 6.

Algorithm 5: ComponentAverageHedge

```
1. Parameter: a real number $\rho > 0$
2. Initialization: Set $W_{i,\mathrm{J}}(0) = W_{i,\mathrm{Q}}(0) = W_{i,\mathrm{K}}(0) = 0$ and $C_{i,\mathrm{J}}(0) = C_{i,\mathrm{Q}}(0) = C_{i,\mathrm{K}}(0) = 0$ for $i = 1, \dots, \mathcal{K}$
3. Repeat for $t = 1, 2, \dots$ until match ends
    (a) Compute average winnings, $A_{i,\mathrm{H}}(t-1)$ for $i = 1, \dots, \mathcal{K}$ and H = J, Q, K:
        $A_{i,\mathrm{H}}(t-1) = \left\{ \begin{array}{ll} \frac{W_{i,\mathrm{H}}(t-1)}{C_{i,\mathrm{H}}(t-1))} & \text{if } C_{i,\mathrm{H}}(t-1) > 0 \\ 0 & \text{otherwise} \end{array} \right\}$
    (b) Compute overall average, $\hat{A}_i(t-1)$ for $i = 1, \dots, \mathcal{K}$:
        $\hat{A}_i(t-1) = \frac{A_{i,\mathrm{J}}(t-1) + A_{i,\mathrm{Q}}(t-1) + A_{i,\mathrm{K}}(t-1)}{3}$
    (c) Choose action $i_t$ according to the distribution $p(t)$, where
        $p_i(t) = \frac{(1+\rho)^{t * A_i(t-1)}}{Z_t}$
    (d) Receive hand H, reward vector $\vec{x}(t)$ and update vector $\vec{u}(t)$
    (e) Set $W_{i,\mathrm{H}}(t) = W_{i,\mathrm{H}}(t-1) + x_i(t)$ and $C_{i,\mathrm{H}}(t) = C_{i,\mathrm{H}}(t-1) + u_i(t)$ for $i = 1, \dots, \mathcal{K}$
```

Algorithm 6: ComponentAverageExp3

```
1. Parameters: reals $\rho > 0$ and $\psi$, $0 \le \psi \le 1$
2. Initialization: Initialize ComponentAverageHedge($\rho$)
3. Repeat for $t = 1, 2, \dots$ until match ends
    (a) Get distribution $p(t)$ from ComponentAverageHedge
    (b) Select strategy $e_t$ to be expert strategy $j$ with probability $\hat{p}_j(t) = (1-\psi)p_j(t) + \frac{\psi}{\mathcal{K}}$
    (c) Observe game sequence $S_t$ and receive reward $x_{e_t}(t) \in [0,1]$
    (d) Let $E_t$ be the set of experts which would have taken the actions required to generate the game sequence $S_t$.
    (e) Feed hand H along with simulated reward vector $\hat{x}(t)$ and update vector $\hat{u}(t)$ into ComponentAverageHedge, where
        $\hat{x}_j(t) = \left\{ \begin{array}{ll} x_{e_t}(t) & \text{if } j \in E_t \\ 0 & \text{otherwise} \end{array} \right\}$
        $\hat{u}_j(t) = \left\{ \begin{array}{ll} 1 & \text{if } j \in E_t \\ 0 & \text{otherwise} \end{array} \right\}$
```

While these algorithms have been designed specifically for Kuhn Poker, they can easily be adapted for larger games and games where the holdings are not equally likely. In such games, separate averages can be kept for each possible holding, and these averages can then be combined according to the probability of each holding. For example, if holding H_1 occurs with 50% likelihood, H_2 occurs with 30% likelihood, and H_3 occurs with 20% likelihood, then the overall average score for expert i would be $\hat{A}_i = 0.5 * A_{i,1} + 0.3 * A_{i,2} + 0.2 * A_{i,3}$.

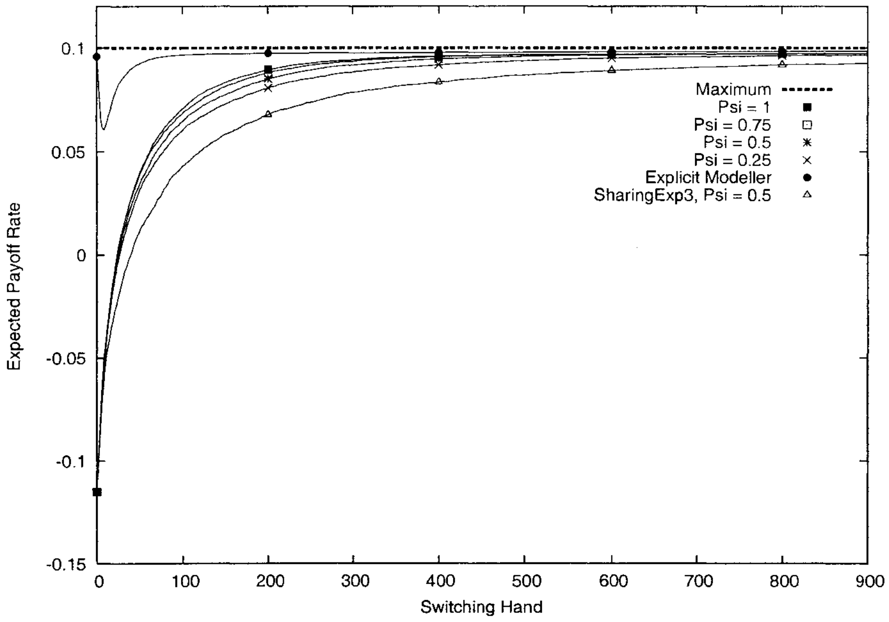
The payoff-rate plots in Figure 5.11 show that the ComponentAverageExp3 method converges to best-response more quickly than any of the previous Exp3 algorithms. However, the explicit modelling method plotted still outperforms the ComponentAverageExp3 algorithm, as it converges even more quickly to the best-response strategy. Reasons why explicit modelling methods perform better in Kuhn Poker are discussed in Section 5.8.

The total winnings plots in Figure 5.12 show that the ComponentAverageExp3 algorithm is a significant improvement over the SharingExp3 algorithm. It is now possible to have positive total winnings against both opponents, which was previously only achieved by explicit modelling methods. The ComponentAverageExp3 series are again much flatter than the explicit modelling series, allowing for a larger interval to switch in without losing much in total winnings.

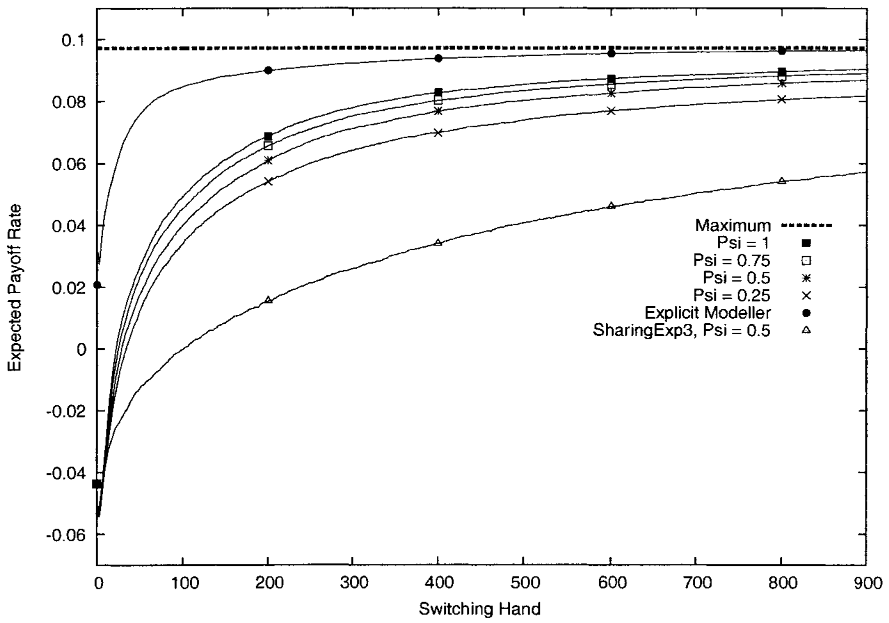
The proportion above equilibrium plots in Figure 5.13 show that nearly 80% of the trials have expected total winnings greater than equilibrium at hand 50 for the ComponentAverageExp3 method. For settings of $\psi <= 0.5$, more than 50% of the trials finish with total winnings above equilibrium, which is better than the explicit modelling methods.

5.4 Decomposition into Subgames

In the previous section, the ComponentAverageExp3 method was derived to ensure that 1/3 of each experts' score came from holding each card in Kuhn Poker, as in theory each card will be held 1/3 of the time. Another way to approach this problem is to treat each holding as a separate game altogether, use the AverageExp3 method in each of these subgames, and combine the perceived best strategies in the subgames to form a counter-strategy for the complete game (henceforth referred to as the *supergame*). Kuhn Poker decomposes into three subgames (the Jack subgame, the Queen subgame, and the King subgame), each of which has a single parameter in the case where P1 is the modeller and dominated strategies are removed. Each Kuhn P1 subgame then has two pure expert strategies, which results in up to $2^3 = 8$ pure counter-strategies generated for the supergame. The imbalance problem for AverageExp3 which was discussed earlier is that experts may be more likely to be updated with certain cards than they are with others. Thus the experts scores often do not have

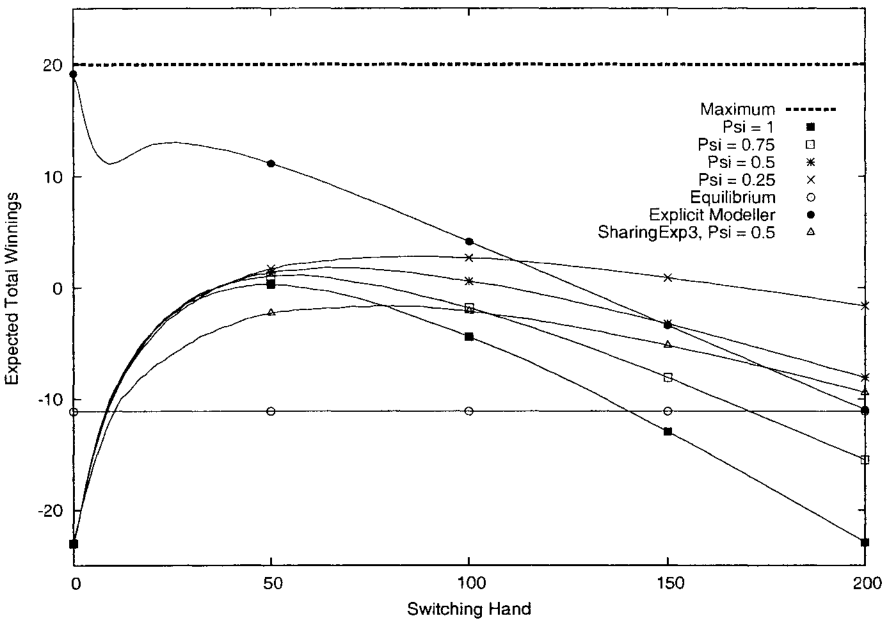


(a) $O_2 = (0.75, 0.8)$

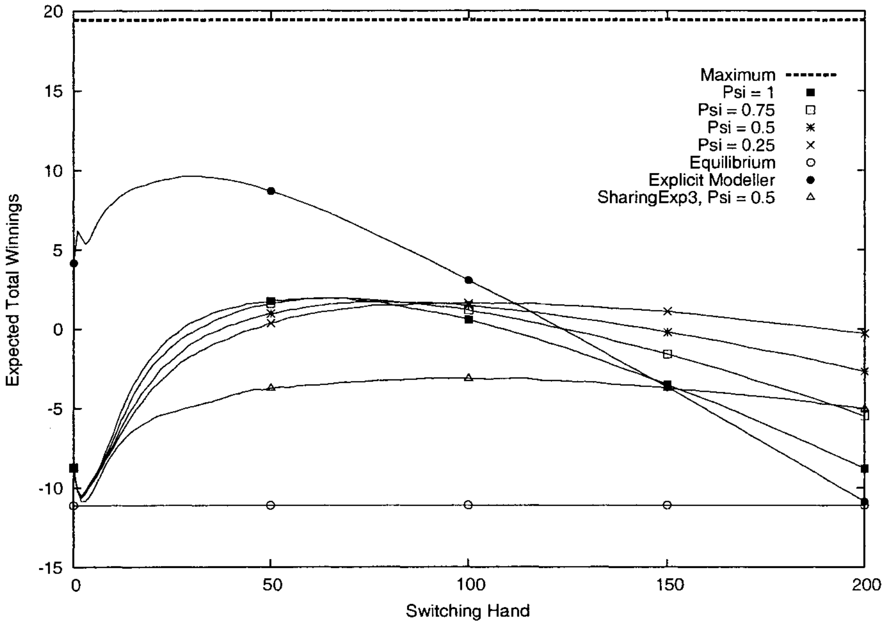


(b) $O_6 = (0.25, 0.67)$

Figure 5.11: ComponentAverageExp3 Method ($\rho = 1.0$) Payoff-Rate Plots

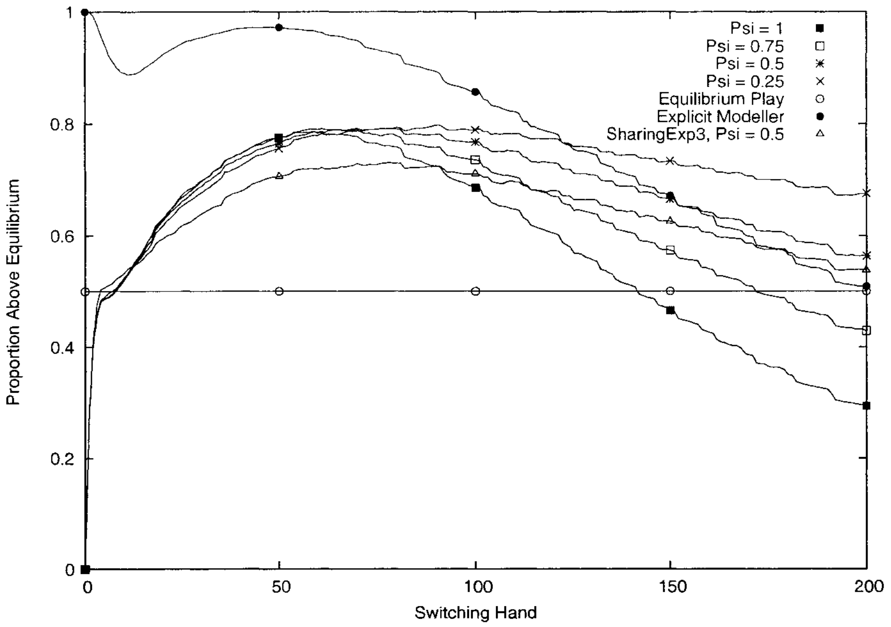


(a) $O_2 = (0.75, 0.8)$

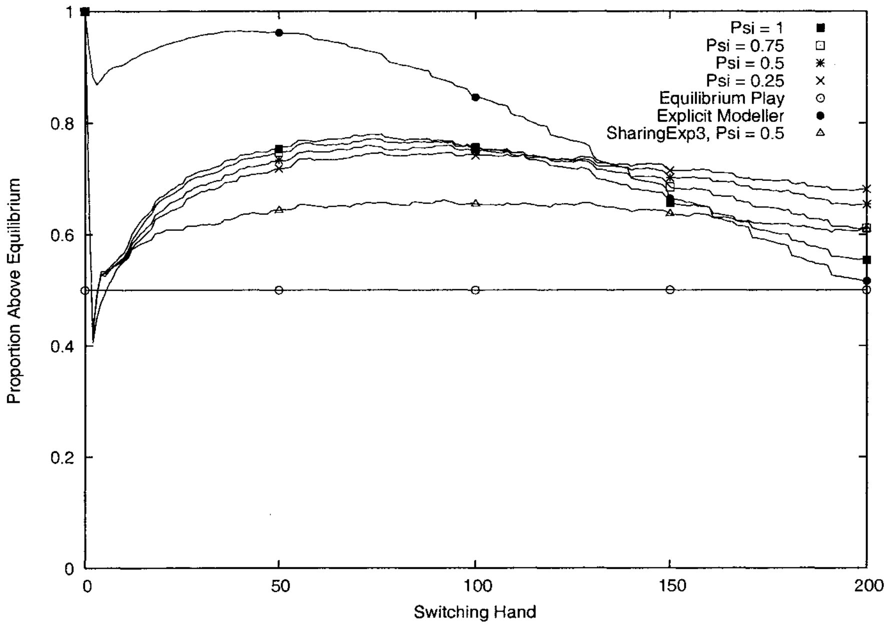


(b) $O_6 = (0.25, 0.67)$

Figure 5.12: ComponentAverageExp3 Method ($\rho = 1.0$) Total Winnings Plots



(a) $O_2 = (0.75, 0.8)$



(b) $O_6 = (0.25, 0.67)$

Figure 5.13: ComponentAverageExp3 Method ($\rho = 1.0$) Proportion Above Equilibrium Plots

balanced contributions from each of the possible holdings. AverageExp3 can be applied here because each subgame is defined by the player having a particular card, which makes it impossible for the holdings to be out of balance within an expert's score as there is only one holding.

```
Algorithm 7: DecompositionExp3
1. Parameters: reals $\rho > 0$ and $\psi$, $0 \leq \psi \leq 1$
2. Initialization: (i) Initialize AverageHedge($\rho$) for J subgame (2 experts)
       (ii) Initialize AverageHedge($\rho$) for Q subgame (2 experts)
       (iii) Initialize AverageHedge($\rho$) for K subgame (2 experts)
3. Repeat for $t = 1, 2, \ldots$ until match ends
    (a) Receive holding H.
    (b) Get distribution $p(t)$ from AverageHedge corresponding to H subgame
    (c) Select strategy $i_t$ to be expert strategy $j$ with probability $\hat{p}_j(t) = (1 - \psi)p_j(t) + \frac{\psi}{2}$
    (d) Observe game sequence $S_t$ and receive reward $x_i(t) \in [0, 1]$
    (e) Let $E_t$ be the set of experts which would have taken the actions required
        to generate the game sequence $S_t$.
    (f) Feed simulated reward vector $\hat{x}(t)$ and update vector $\hat{u}(t)$ into Average-
        Hedge corresponding to the holding H, where
            $\hat{x}_j(t) = \left\{ \begin{array}{ll} x_i(t) & \text{if } j \in E_t \\ 0 & \text{otherwise} \end{array} \right\}$
            $\hat{u}_j(t) = \left\{ \begin{array}{ll} 1 & \text{if } j \in E_t \\ 0 & \text{otherwise} \end{array} \right\}$
```

The DecompositionExp3 algorithm is very similar to the ComponentAverageExp3 algorithm. The decomposition method determines the best expert for each subgame corresponding to the possible holdings and combines these experts to form a strategy for the supergame, while the component average method computes scores for each supergame strategy which weight the subgame scores in proportion to the probability of each subgame. The two methods are identical if the set of experts for the supergame in the ComponentAverageExp3 algorithm consists of all the pure non-dominated strategies, but this is not the case in the studies shown here.

In the experiments in this chapter, the experts for the Exp3 algorithms are the six pure strategies which are possible best-response strategies to P2's possible strategies, as well as the equilibrium strategy corresponding to $\gamma = 0.5$. This means the results for the two algorithms, ComponentAverageExp3 and DecompositionExp3, can be slightly different. The

DecompositionExp3 algorithm may recommend one of the strategies ($\alpha = 0, \beta = 0, \gamma = 0$) or $(1, 1, 1)$ as the best counter-strategy which could not be identified by the ComponentAverageExp3 algorithm. Similarly, the ComponentAverageExp3 algorithm could recommend $(0.17, 0.5, 0.5)$ as the best counter-strategy, but this strategy could not be identified by the DecompositionExp3 algorithm.

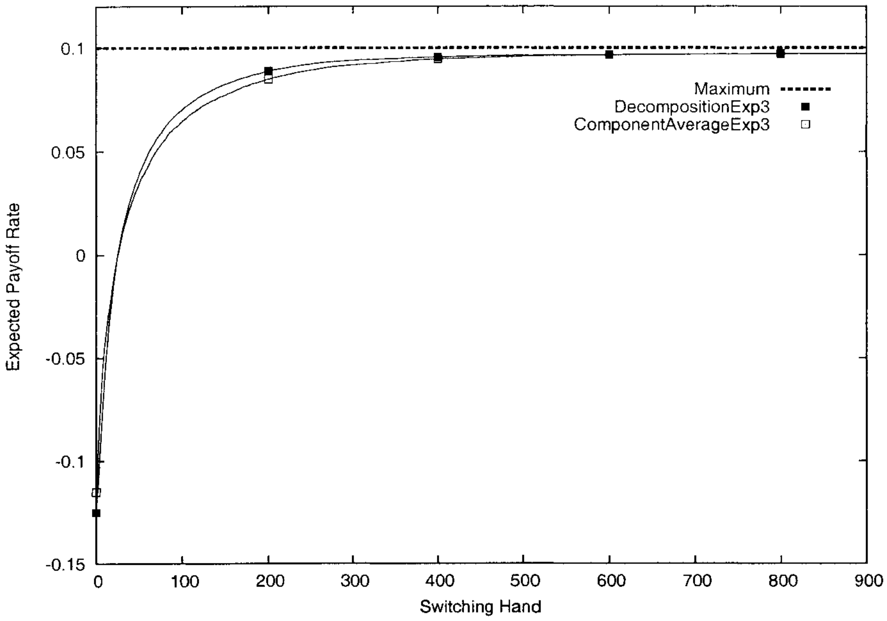
For extreme settings of the ρ and ψ parameters, the data-collection strategies used by the two algorithms may also be different. If ρ is set very high and ψ is very low for the DecompositionExp3 algorithm, then the pure strategies initially chosen for each subgame will likely be chosen over and over again, as the alternatives are stuck at the cumulative score of 0.

For these settings in the ComponentAverageExp3 algorithm, a similar effect occurs, as well as the fact that the initial choice in one subgame affects the choice of strategies in the subsequent subgames. For example, if P1 was initially dealt the Jack and took the action corresponding to $\alpha = 1$, then the experts (1,0,0), (1,0,1), and (1,1,0) would all receive the reward, which is assumed to be greater than zero for this discussion (also assume that the strategy (0.17, 0.5, 0.5) recommended the action corresponding to $\alpha = 0$ and thus does not receive the reward). Given the extreme parameter settings, these experts would now be the only experts considered for hands following the first one, and if P1 was dealt the Queen in the second hand, the setting $\beta = 0$ would be twice as likely to be chosen as $\beta = 1$. However, with reasonable settings of the parameters ρ and ψ, the results of the ComponentAverageExp3 and DecompositionExp3 algorithms in Kuhn Poker are nearly identical, as demonstrated by the payoff-rate plots shown in Figure 5.14. In these plots, the DecompositionExp3 ($\rho = 1, \psi = 0.5$) method has a slightly higher payoff-rate than the ComponentAverageExp3 ($\rho = 1, \psi = 0.5$) method.

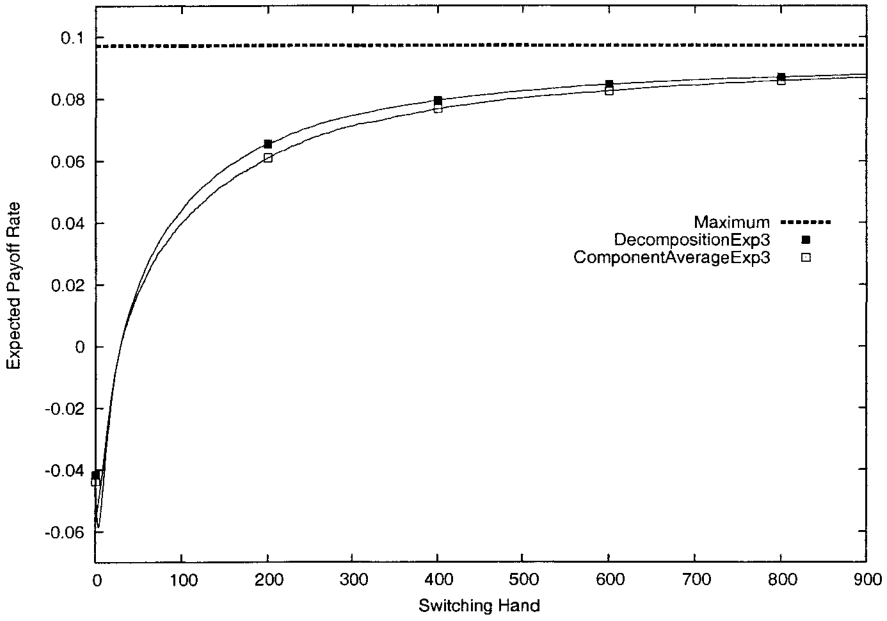
5.5 Initial Weights for Expert Strategies

One of the most notable properties of the payoff-rate plots shown in this chapter is that the initial payoff-rate is often very different for the implicit modelling techniques than it is for the explicit modellers. For the implicit modellers, each of the expert strategies is initially tied with a score of 0, which means the counter-strategy initially recommended is to play each expert one-seventh of the time. Meanwhile, the explicit modellers begin with the initial estimates ($\eta = 0.5, \xi = 0.5$) which leads them to recommend the strategy of playing S_2 half the time and S_3 half the time.

One of the concerns arising from these payoff-rate plots is that the explicit modelling methods might be outperforming the implicit modelling methods because of a better starting



(a) $O_2 = (0.75, 0.8)$



(b) $O_6 = (0.25, 0.67)$

Figure 5.14: DecompositionExp3 Method ($\rho = 1.0$) Payoff-Rate Plots

point. This study adjusts the initial payoff-rate of the implicit modellers to match that of the explicit modellers; this adjustment is done by adding a small amount of weight to the scores of the expert strategies S_2 and S_3. This results in both the implicit and explicit modellers recommending the same initial counter-strategy.

Explicitly, the score for expert i after hand t is computed as

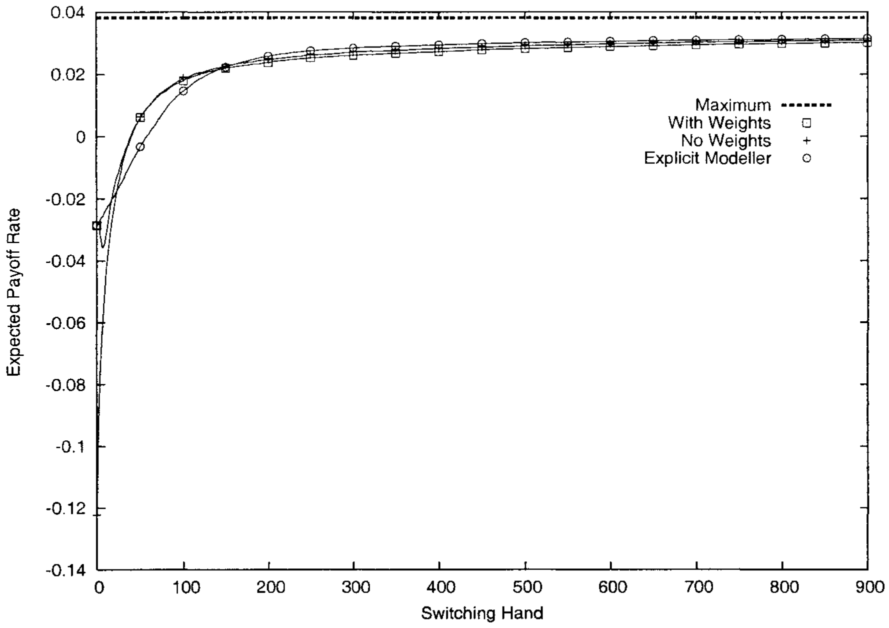
$$\hat{W}_i(t) = \begin{cases} t * \hat{A}_i(t) + 1 & \text{if } i = 2 \text{ or } i = 3 \\ t * \hat{A}_i(t) & \text{otherwise} \end{cases}$$

where $\hat{A}_i(t)$ is computed as in Algorithm 5. The modeller uses the ComponentAverageExp3 algorithm with $\rho = 1.0$ and $\psi = 0.5$.

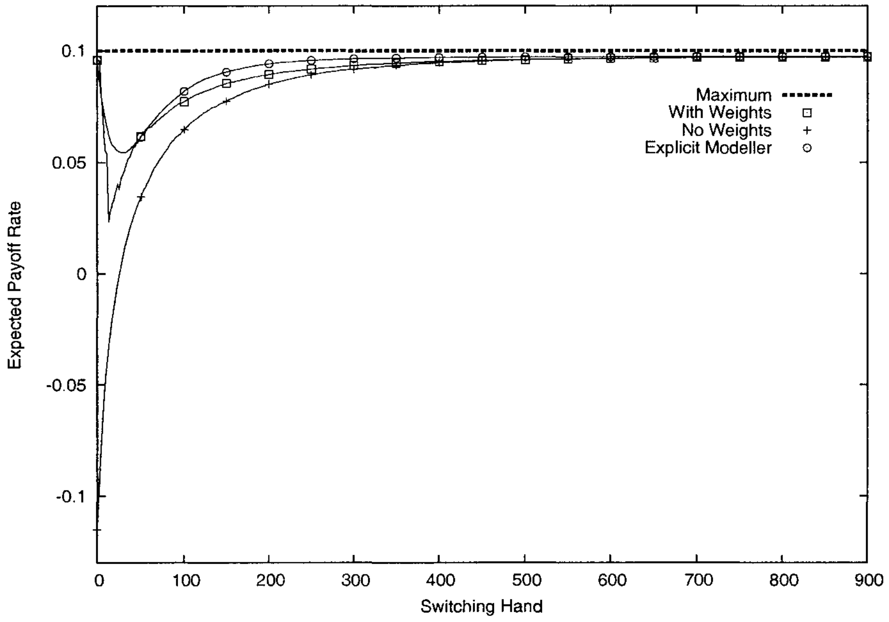
Figure 5.15 shows that having experts S_2 and S_3 weighted provides a better initial payoff-rate for testpoints O_1 and O_2. However, the implicit modeller with no weights quickly catches up in the payoff-rate plots, catching up within 50 hands for O_1, while taking about 300 hands to catch up for O_2. Neither implicit modeller is able to match the convergence speed of the explicit modeller plotted for comparison, who uses the $\gamma = 0.75$ equilibrium data-collection strategy.

The total winnings plots in Figure 5.16 show that the modeller with weighted experts has a higher initial total expected winnings for these testpoints, but the modeller with no initial weights quickly closes the gap. This gap may not close completely (as is the case for O_2) for two reasons. The first reason is that the modeller with no weights added to the experts may not have a model with as high a payoff-rate as the modeller with weighted experts before the match is over, which means as long as there are hands to be played, the modeller with weighted experts has a higher expected winnings for the remainder of the match. O_2 is an opponent for which the model with the weighted experts achieves a higher payoff-rate until about hand 300, so the modeller with no weights will have lower expected total winnings for an entire 200-hand match. The second reason is that the data-collection strategy is partially exploitive, which means the implicit modeller with the better model will win more during the data-collection phase.

In contrast, Figure 5.17 shows that having experts S_2 and S_3 weighted provides a lower initial payoff-rate for testpoint O_4, and in this case, the weighted modeller never closes the gap in the total winnings plot. This shows that weighting experts S_2 and S_3 does not improve the results for all testpoints; any weighting scheme is likely to improve results for some testpoints and be detrimental to the results of others.

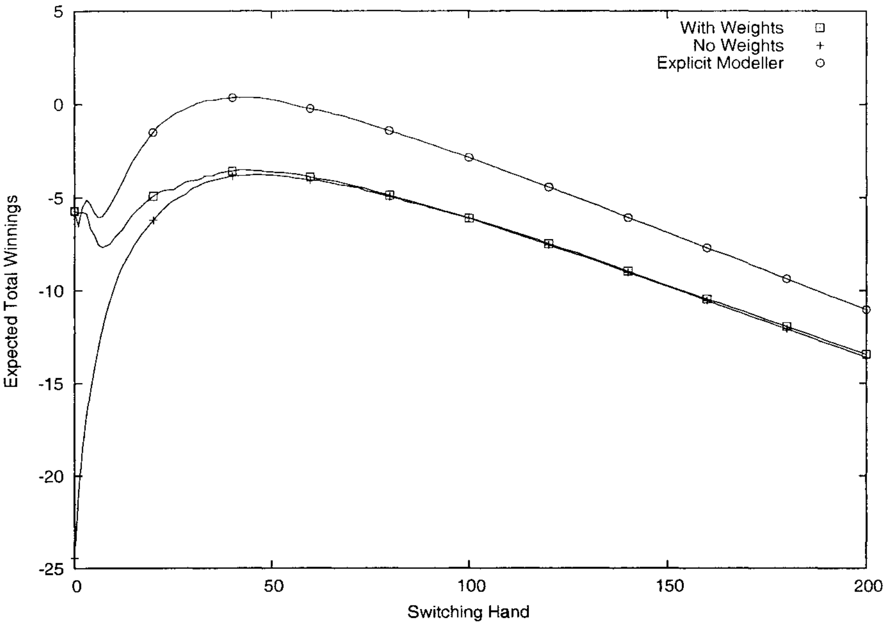


(a) $O_1 = (0.8, 0.29)$

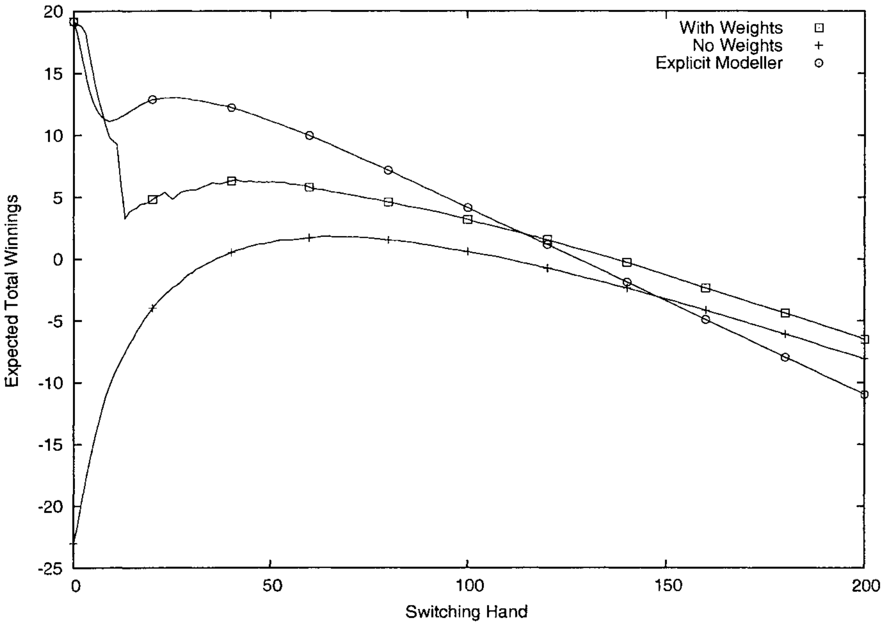


(b) $O_2 = (0.75, 0.8)$

Figure 5.15: Weighted Experts Experiment (CompAverageExp3 ($\rho = 1.0, \psi = 0.5$)) Payoff-Rate Plots

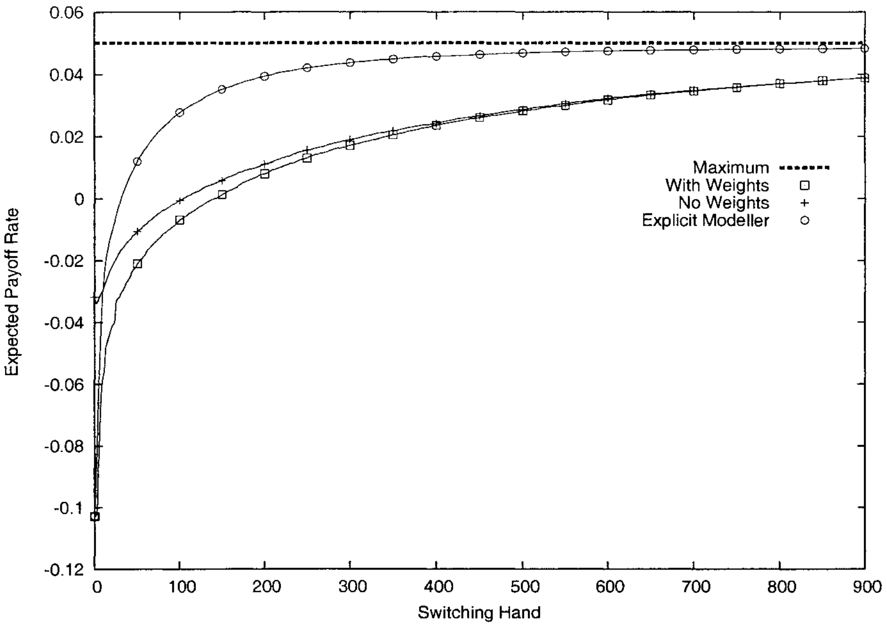


(a) $O_1 = (0.8, 0.29)$

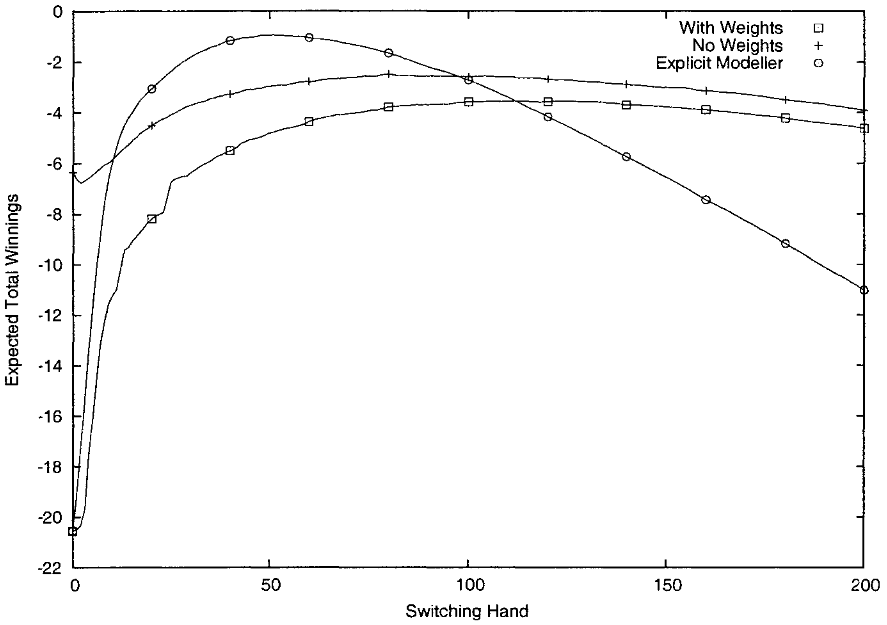


(b) $O_2 = (0.75, 0.8)$

Figure 5.16: Weighted Experts Experiment (CompAverageExp3 ($\rho = 1.0, \psi = 0.5$)) Total Winnings Plots



(a) Payoff-Rate Plot



(b) Total Winnings Plot

Figure 5.17: Effect of Weighted Experts on $O_4 = (0.17, 0.2)$

5.6 The Effect of Match Length on Expected Winnings

All of the previous total winnings plots that have been shown have match lengths of 200 hands. The purpose of this study is to see how changing the match length affects the results of the opponent modelling methods. It is also interesting to see how the switching hand that maximizes the expected total winnings changes for different match lengths.

Figure 5.18 shows two plots, where each plot shows total winnings for a single modelling method for different match lengths. These results are averaged against random opponents that have exploitability 0.0556 \$/hand. The explicit modeller, using the equilibrium data-collection method, is not able to achieve positive winnings when matches are 100 hands or less, as there is not enough time to learn a good model and have time to use it. The explicit modeller is able to achieve positive winnings for longer matches. The implicit modeller, using ComponentAverageExp3 with $\rho = 1.0$ and $\psi = 0.5$, is not able to expect positive winnings against this set of opponents when the match lasts 400 hands or less.

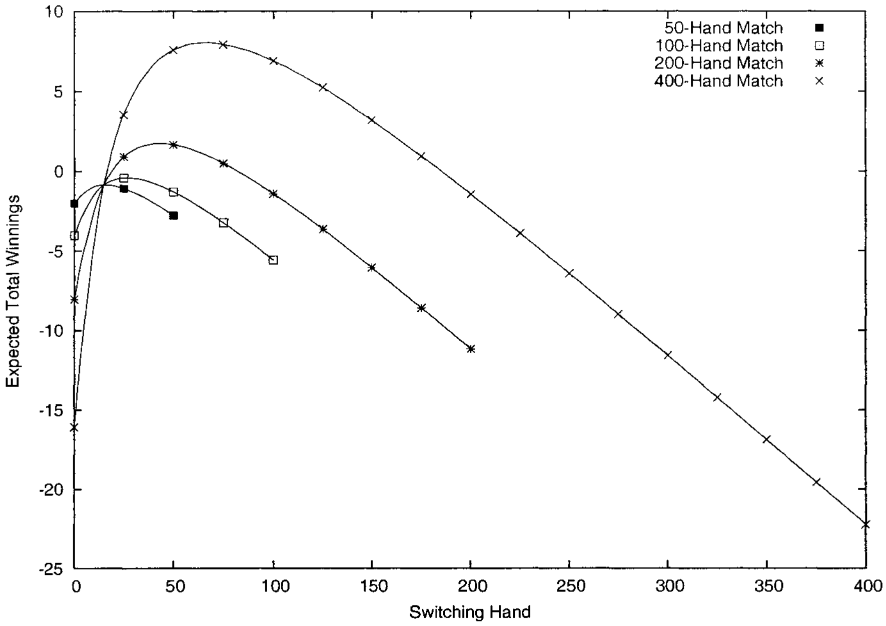
Figure 5.19 shows the switching hand that achieves the highest expected total winnings for each match length. This maximal switching hand seems to grow linearly with the match length for both the explicit and implicit methods. The line for the implicit modeller suggests that the modeller should switch from exploration to exploitation after about one-quarter of the match is complete. The explicit modeller should switch after collecting data for about one-eighth of the match.

If the precise length of the match is not known in advance, the total winnings plots shown in Figure 5.18 suggest that for any match longer than 50 hands, the modeller can achieve expected total winnings close to the maximal amount if he switches near hand 25 for the explicit modeller shown or near hand 50 for the implicit modeller, for any of the match lengths shown in the plot.

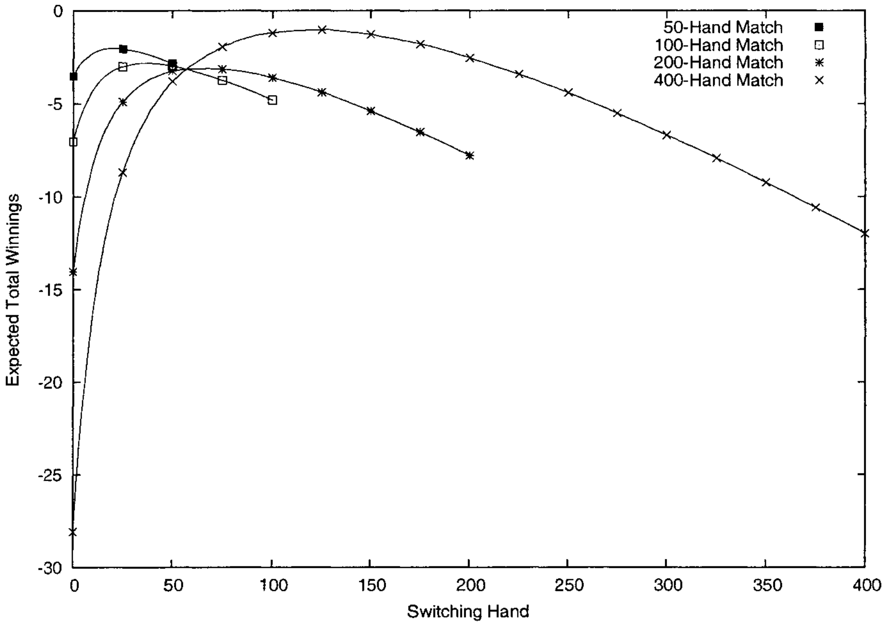
5.7 The Effect of Varying ρ

The results for each of the different Exp3 methods shown thus far have been for $\rho = 1.0$ and varying values of ψ. This allowed for the impact of the ψ parameter to be seen, but not the impact of different values of ρ. This study rectifies this situation by varying the value of ρ for fixed values of ψ using the ComponentAverageExp3 algorithm.

Figures 5.20 and 5.21 show the payoff-rate and total winnings plots for the two opponents O_1 and O_2, for various values of ρ with $\psi = 0.5$. For these two opponents and this setting of ψ, changing ρ has no effect on the payoff-rate plots. It does have an effect on the total winnings plots as larger values of ρ (such as 3 or 9) result in higher total winnings, as the



(a) Equilibrium ($\gamma = 0.75$) Data-Collection Explicit Modeller



(b) ComponentAverageExp3 ($\rho = 1.0, \psi = 0.5$)

Figure 5.18: Total Winnings Plots for Different Match Lengths

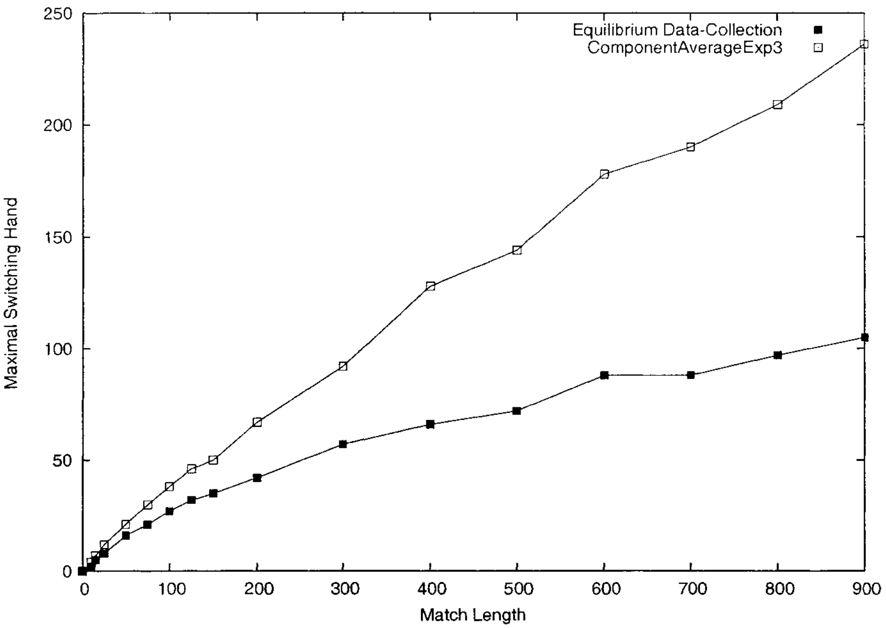


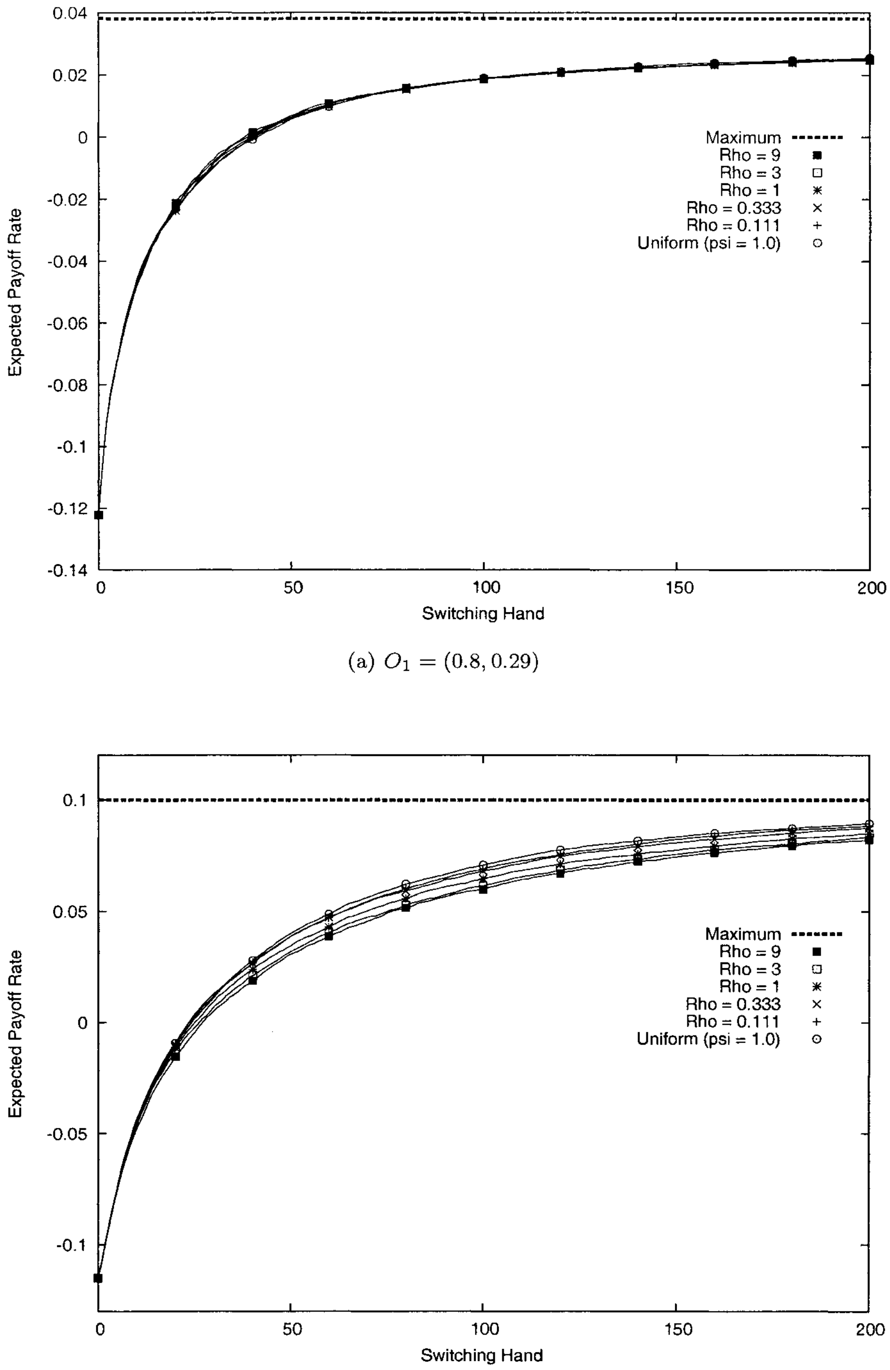
Figure 5.19: Maximal Switching Hand for Varying Match Lengths

data-collection strategy is more exploitive. Lower values of ρ (such as 0.111) result in a data-collection strategy that is close to the uniform exploration strategy, which is why the corresponding total winnings series are very close (the $\rho = 0.111$ series has slightly higher winnings than the uniform exploration strategy at the end of 200 hands).

Lower values of ψ result in less uniform exploration, and a stronger impact of the ρ parameter. Figures 5.22 and 5.23 show the payoff-rate and total winnings plots for various ρ values with $\psi = 0.25$. Here there is a little bit of separation between the series in the payoff-rate plot for O_2, as lower values of ρ generate better opponent models; however the total winnings plot shows that these better models come at a higher cost, as the more exploitive values of ρ achieve higher total winnings.

Due to numerical issues arising, the setting $\psi = 0$ cannot be used for the Exp3 algorithm, as rewards are divided by the probability of choosing the selected expert e_t, and this probability is sometimes very near zero. In addition, with large settings of ρ, the first expert that receives a nonzero reward is very likely to be repeatedly selected, as this expert is the only expert with a nontrivial probability of being selected. The ComponentAverage-Exp3 algorithm does not have the numerical issues, as rewards are not scaled. Furthermore, the sharing of rewards between experts ensures that multiple experts are rewarded on each hand, and no single expert jumps out to an insurmountable lead.

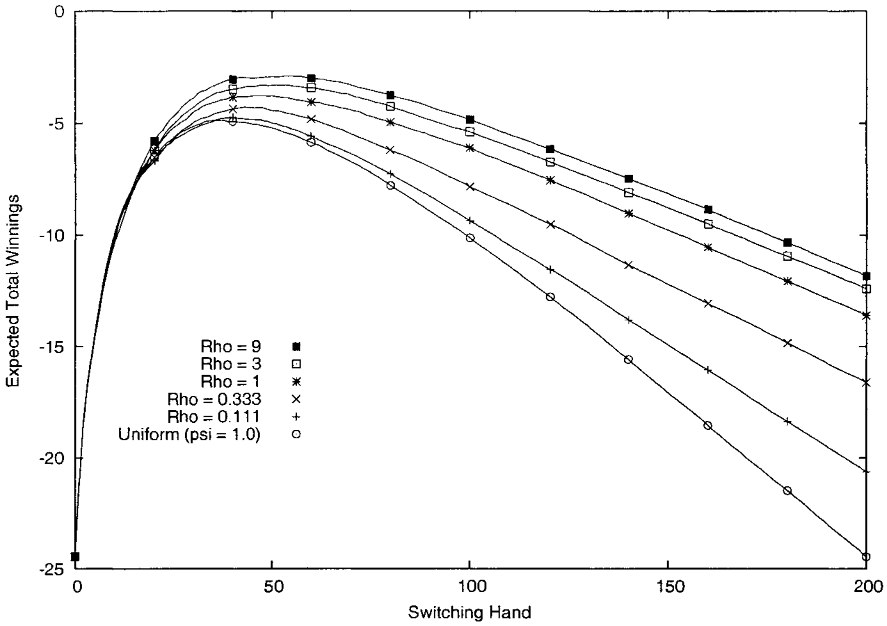
Figures 5.24 and 5.25 show the payoff-rate and total winnings plots for various ρ values



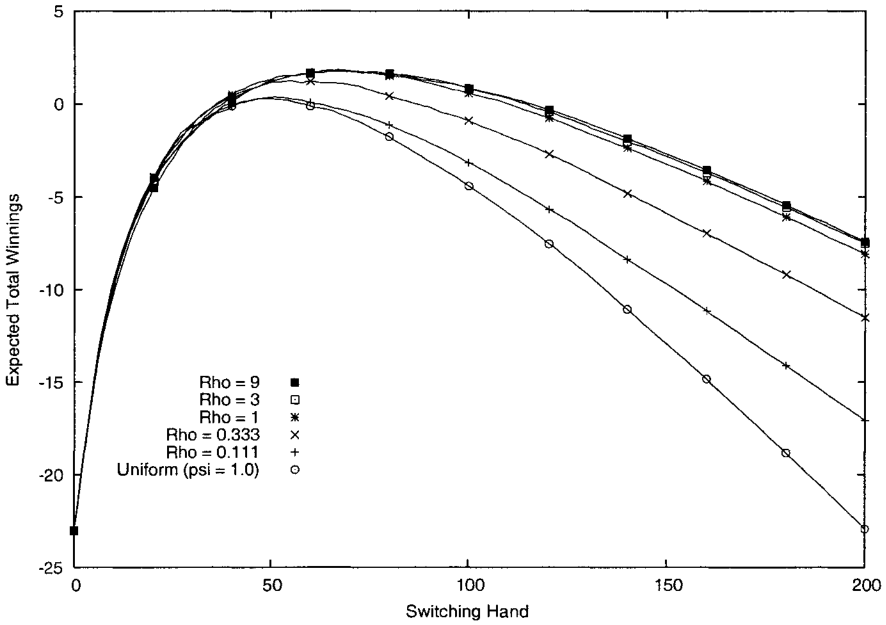
(a) $O_1 = (0.8, 0.29)$

(b) $O_2 = (0.75, 0.8)$

Figure 5.20: Varying rho (CompAverageExp3 ($\psi = 0.5$)) Payoff-Rate Plots

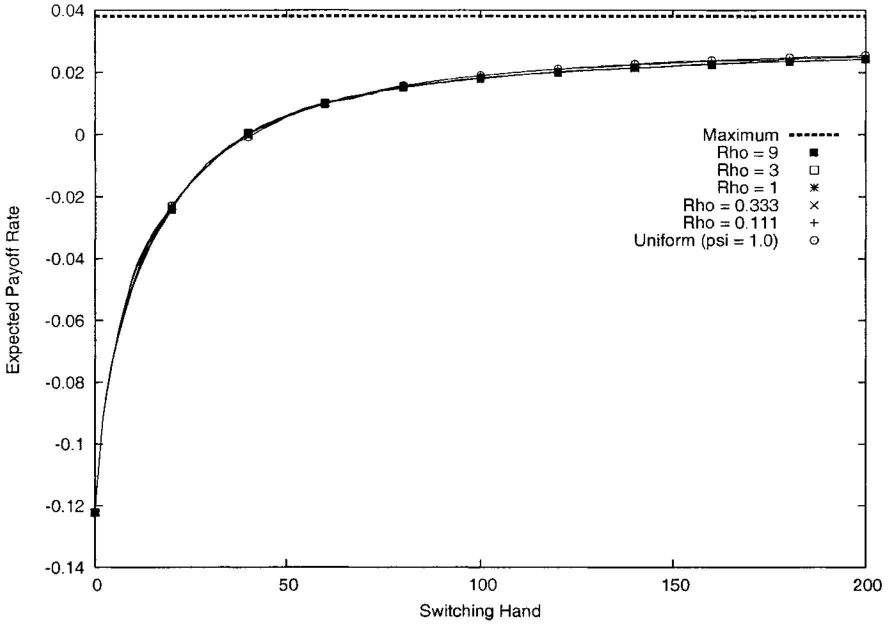


(a) $O_1 = (0.8, 0.29)$

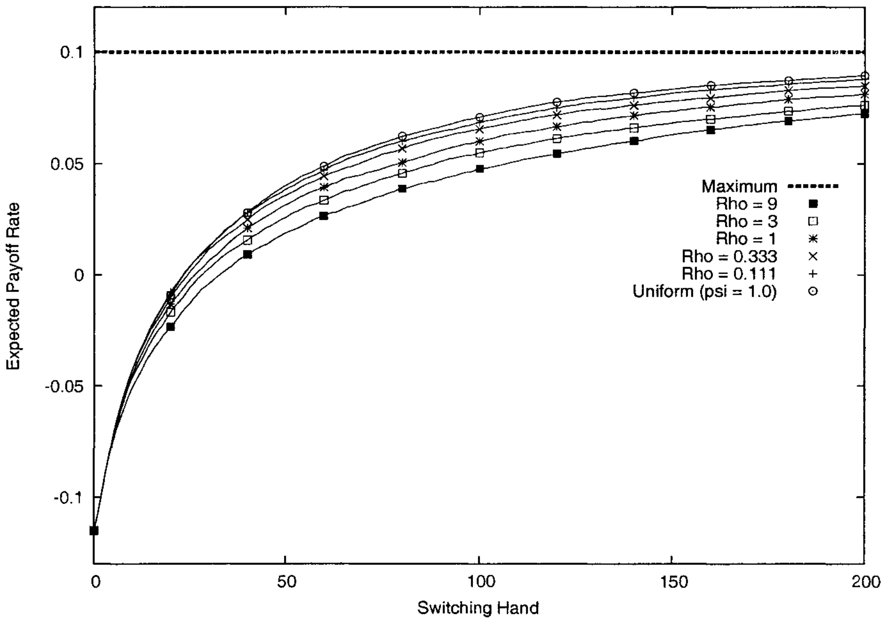


(b) $O_2 = (0.75, 0.8)$

Figure 5.21: Varying rho (CompAverageExp3 ($\psi = 0.5$)) Total Winnings Plots

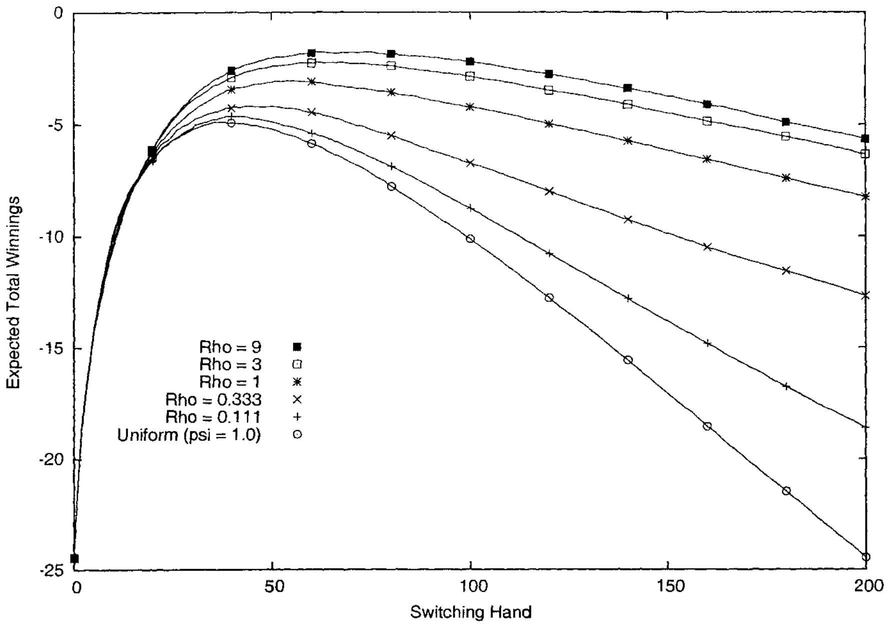


(a) $O_1 = (0.8, 0.29)$

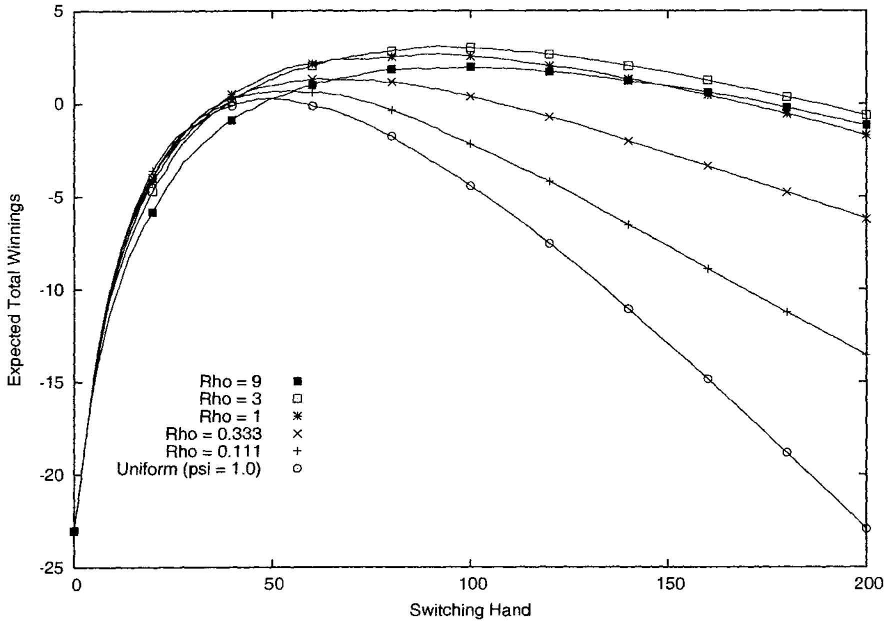


(b) $O_2 = (0.75, 0.8)$

Figure 5.22: Varying rho (CompAverageExp3 ($\psi = 0.25$)) Payoff-Rate Plots

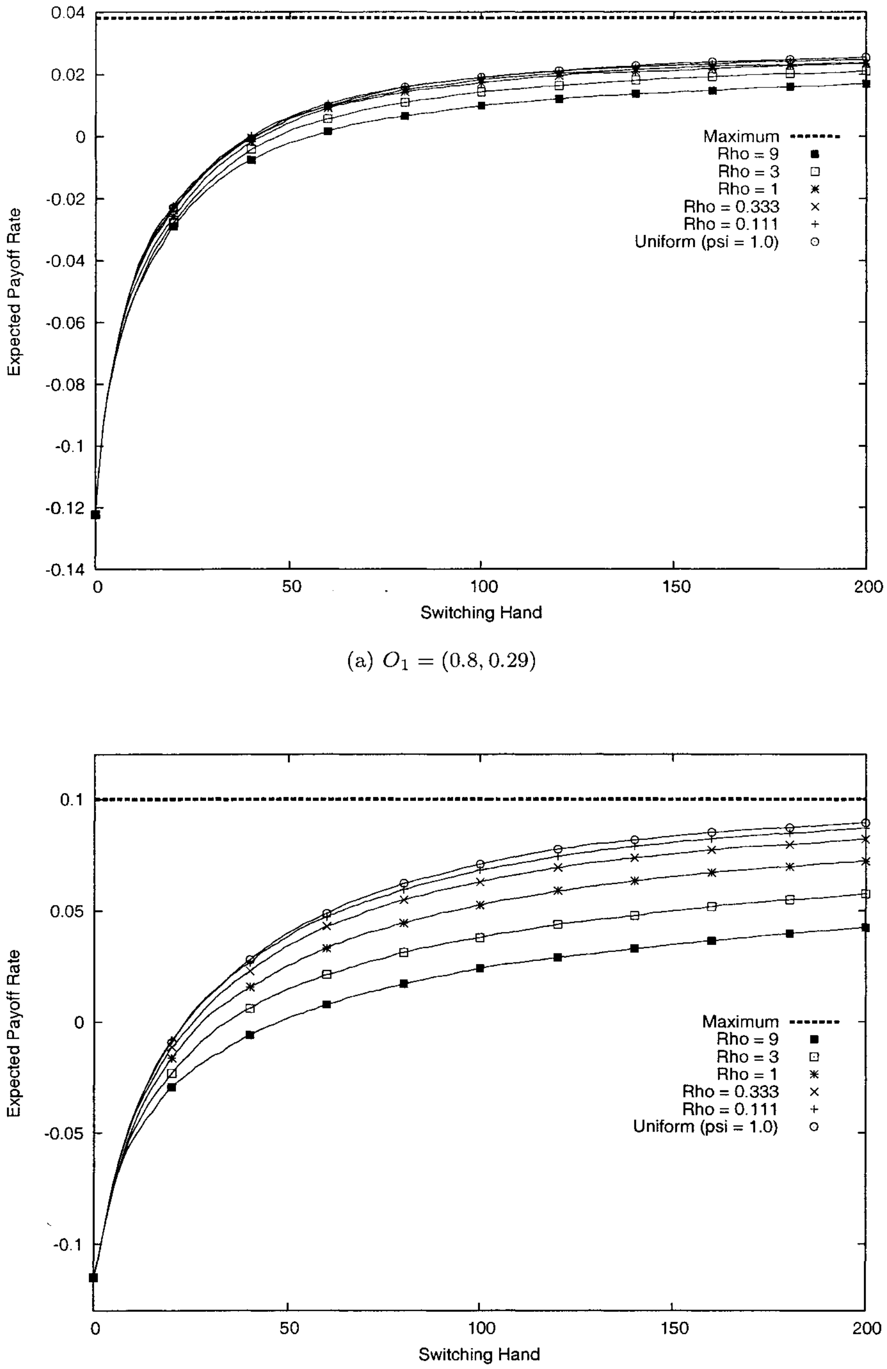


(a) $O_1 = (0.8, 0.29)$



(b) $O_2 = (0.75, 0.8)$

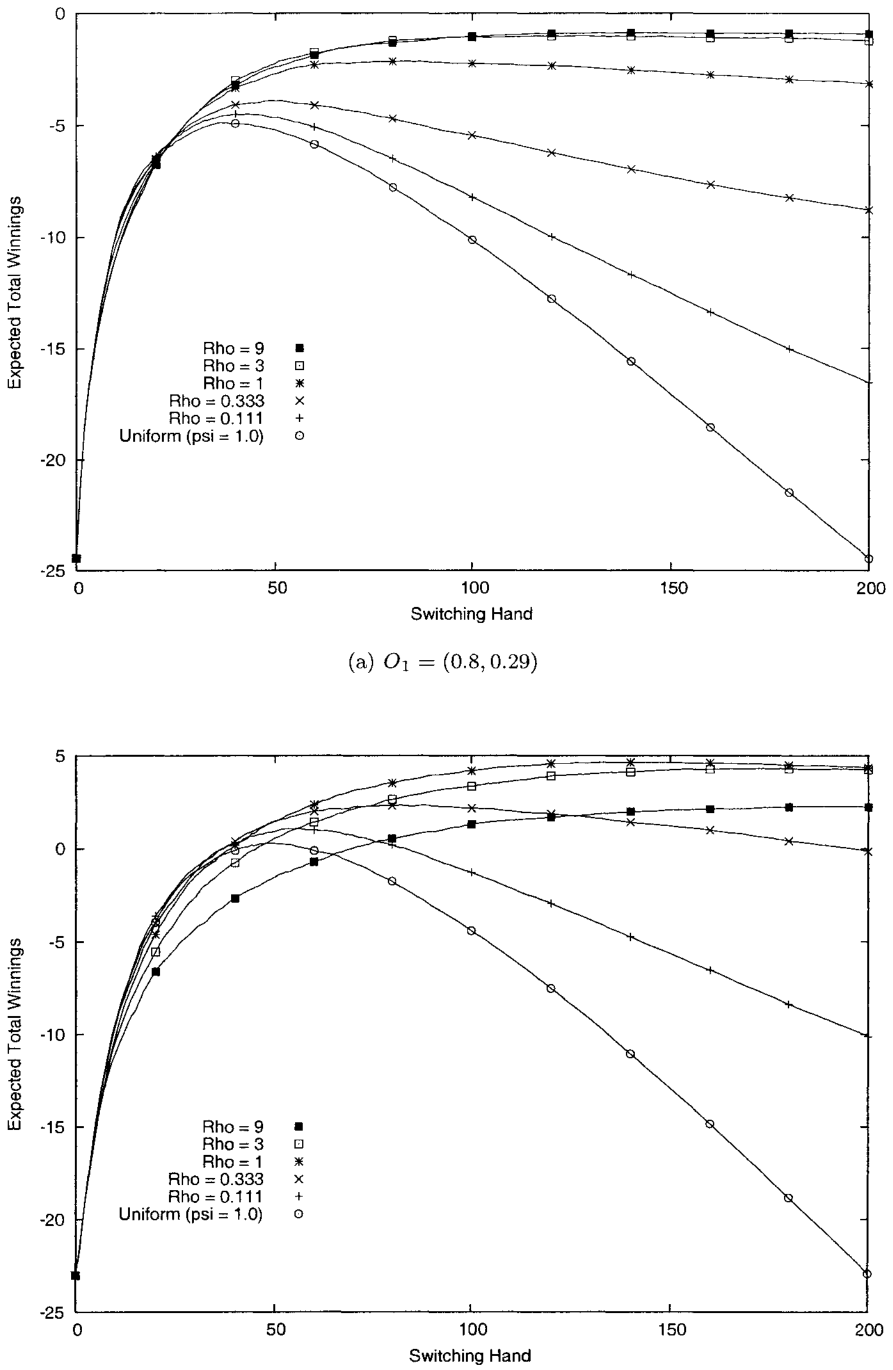
Figure 5.23: Varying rho (CompAverageExp3 ($\psi = 0.25$)) Total Winnings Plots



(a) $O_1 = (0.8, 0.29)$

(b) $O_2 = (0.75, 0.8)$

Figure 5.24: Varying rho (CompAverageExp3 ($\psi = 0$)) Payoff-Rate Plots



(a) $O_1 = (0.8, 0.29)$

(b) $O_2 = (0.75, 0.8)$

Figure 5.25: Varying rho (CompAverageExp3 ($\psi = 0$)) Total Winnings Plots

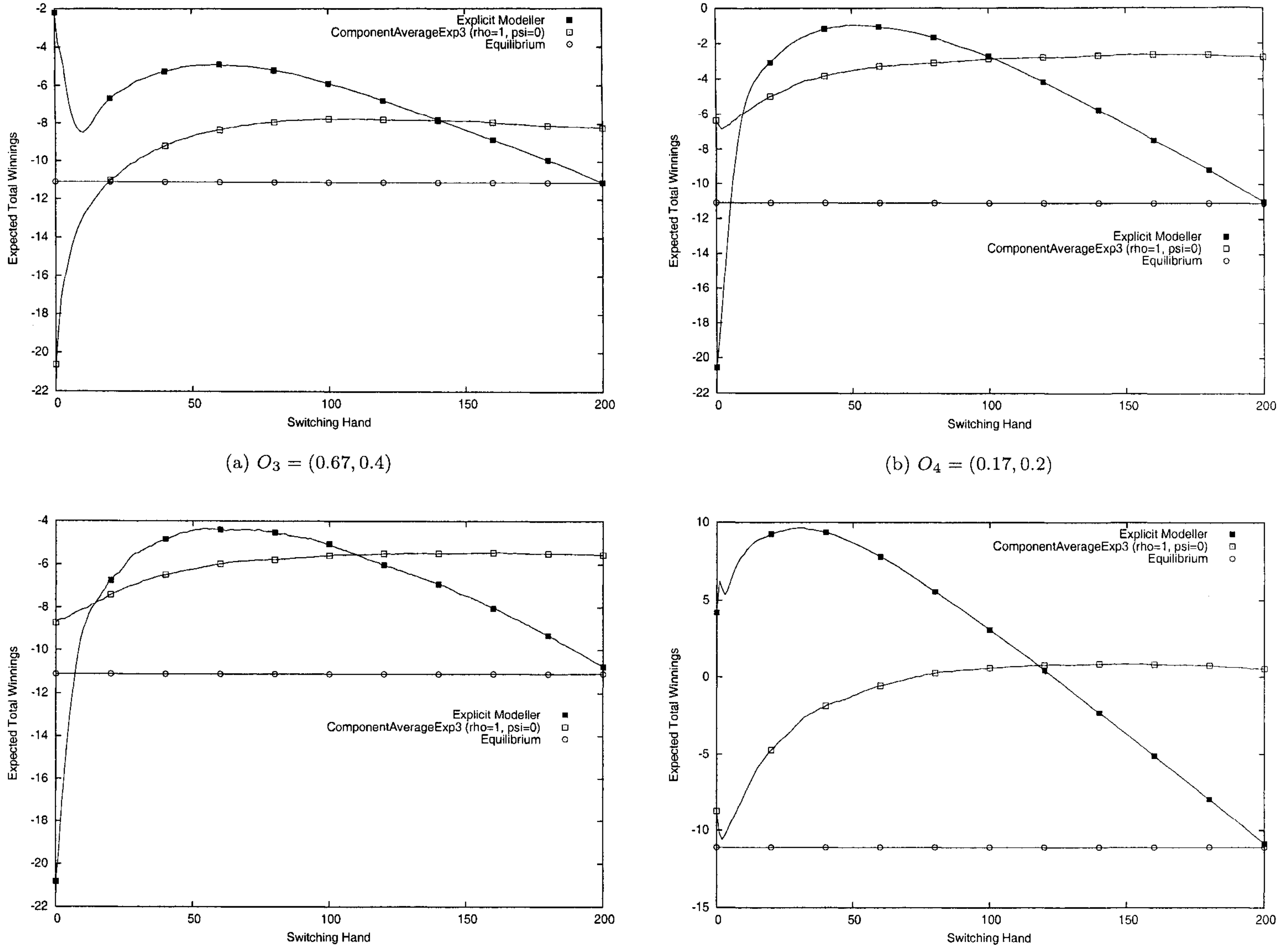
with $\psi = 0$. Since no uniform exploration is performed, the methods with larger settings of ρ do not always quickly build good models, as the first experts to show promise are used repeatedly during the data-collection phase; if the actual best expert, e_b, does not initially show promise, then the algorithm is unlikely to update e_b very often and it takes longer for e_b to be identified as the best expert. Thus for large values of ρ (ie. $\rho \geq 3$), the algorithm may not pay as much during the exploration phase when $\psi = 0$ as it does when $\psi \geq 0.25$, but this is offset by the decrease in modelling effectiveness which results in decreased winnings over the exploitation phase. With small settings of ρ (ie. $\rho \leq 1/3$), the methods still perform a great deal of exploration among all experts even with $\psi = 0$, leading to better models being developed than for large settings of ρ. However, the frequent use of bad experts during data-collection results in lower winnings during the exploration phase, offsetting the higher winnings of the exploitation phase. The best total winnings results for O_1 (and for many of the other testpoints that are not shown here) occur for the setting $\rho = 1$, when a good model is usually found, as the value of ρ is not so large that data-collection is solely focused on the initially promising experts, and the cost of exploration is not too high, as promising experts are played more often than bad experts during data-collection.

Overall, the parameter settings which achieve the highest expected total winnings experimentally are $\rho = 1$ and $\psi = 0$. However, even with these settings, the ComponentAverageExp3 method does not achieve as high a total winnings as the explicit modeller in the plots shown in Figure 5.26.

5.8 Conclusions

The first conclusion that seems clear from the results presented in this chapter is that in the setting of Kuhn Poker, explicit modelling is a much stronger modelling method than implicit modelling. The explicit modelling methods consistently generate stronger models and have higher expected total winnings. Although it appeared that this might be happening because the initial estimates held by the explicit modelling methods lead to better initial counter-strategies and the Exp3 algorithms could not catch up, this theory is disproved in Section 5.5.

In Kuhn Poker, explicit modelling should achieve better results than implicit modelling for a number of reasons. First, the game is small enough that the parameter model for P2 only has two parameters for P1 to estimate. Second, many games end with a showdown, and even when hands do not end in a showdown there are cases for which the card unseen by the modeller can be conclusively inferred. Explicit modelling techniques depend on this data, while implicit modellers are oblivious to whether or not showdowns occur. Third and



(a) $O_3 = (0.67, 0.4)$

(b) $O_4 = (0.17, 0.2)$

(c) $O_5 = (0.25, 0.17)$

(d) $O_6 = (0.25, 0.67)$

Figure 5.26: ComponentAverageExp3 ($\rho = 1, \psi = 0$) Total Winnings Plots

probably most importantly is that the explicit modelling method does a huge amount of information sharing, as it combines data from different instances of the opponent's parameters in the game tree, such as the data discovered about η when P1 holds the Jack and the data discovered about η from when he holds the King. Emulating this information sharing in the implicit modelling methods described in this chapter would undoubtedly raise data-balancing problems.

However, implicit modelling research is not a lost cause, as there are some valuable properties held by the methods. Due to the fact that data-collection is directed towards promising strategies, the winning rate of the implicit modelling method during the exploration phase is often higher than that of explicit modelling methods. This also results in flatter total winnings curves for the implicit modelling methods, which means they have a larger interval within to switch without losing a great deal of winnings in comparison to the best switching point. Exploratory data-collection strategies for explicit modellers can sometimes be highly exploited by the opponent, and be unable to recoup the losses even after an effective model has been generated. In this case the implicit modelling methods are preferrable as the data-collection strategy is adjusted to avoid losing as much during the exploration phase.

The implicit modelling methods are achieving better results than the explicit modelling method using the $\gamma = 0$ equilibrium data-collection strategy. In larger games where only a single equilibrium strategy is known, or if multiple equilibrium strategies are known but their exploration values are unknown, implicit modelling could be a viable alternative.

One major advantage of implicit modelling is that all that is required is a set of strategies to use; no information is required about the opponent's strategy, which may be incredibly complex for large games. This suggests that it may be easier to implement the implicit modelling techniques for larger games, as it does not need to keep track of a huge model.

It also seems possible that some hybrid of the two modelling techniques could be created; the implicit modelling method could be used to collect data, while the data collected is used to create an explicit model of the opponent. The set of counter-strategies could then be assigned scores based on the explicit model. This hybrid technique would combine the best of both worlds, taking advantage of the powerful data-utilization of the explicit modeller while risking less winnings during the exploration phase due to the exploitive nature of the implicit modeller's data-collection strategy.

Chapter 6

Related Work

Poker has been of interest to game theorists ever since the fundamental work of von Neumann and Morgenstern [40], which included an analysis of a small poker game. As the field of game theory developed, many game theorists developed their own small poker variants (including Kuhn Poker) in order to demonstrate game theory dynamics [21, 26, 28]. More recently, attempts have been made to apply game theory principles to larger, more popular poker variants, including Five-Card-Draw [1, 43], Stud Poker [43], and Texas Hold'em [8]. With the recent breakthrough of practical methods for solving games [23] and increasing computational power, more poker games are becoming solvable by game theory techniques.

Other early studies of poker include the simulation of human cognitive processes [17], as well as the application of machine learning techniques [35, 41].

Poker has recently experienced an explosion in popularity, and this has been mirrored in academia as well. The University of Alberta Computer Poker Research Group (CPRG) has been one of the leaders in poker research for the game of Limit Texas Hold'em, taking several different approaches to the challenge of creating a strong player. Recent approaches have resulted in the programs Poki, PsOpti, and Vexbot.

Poki [14, 15] is a program which does opponent modelling in Texas Hold'em, for play in a ring-game with up to ten players. Poki is a rule-based system which performs simulations of the rest of the game to decide what action is best. Explicit opponent models are used in the simulations to predict the holdings of the opponents as well as their future decisions. This program is slow to adapt, indicating that either the model is not being effectively developed, or it is not being effectively used. One problem is that the ten-player ring-game has an exponentially larger game tree than the two-player game, resulting in a need for an exponentially larger model.

PsOpti [8] was created using game-theory to solve a simplified version of two-player Texas Hold'em. While the program cannot be exploited in the simplified game, the mapping of the solution of the simplified game onto a strategy for the full game leaves "holes" in the strategy which can be exploited by strong players. Additionally, since the program plays a fixed strategy, opponents that discover weaknesses in the program's strategy can continually exploit these weaknesses.

Vexbot [7] is also designed for two-player Texas Hold'em, and uses explicit modelling in its approach. Vexbot searches a game tree to compute the expected value of each action, and uses opponent modelling to improve its evaluation function, using observations from past hands to estimate the probability of reaching each leaf in the game tree, as well as estimating the opponent's hand strength at each leaf. Vexbot is quickly able to take advantage (within 200-400 hands) of most weak rule-based computer programs, but takes longer (typically several thousand hands) to successfully model PsOpti, which is a much more complex than other rule-based systems that have been developed. Vexbot has had limited success against strong humans, who effectively change their style more quickly than Vexbot can adapt.

The CPRG is also currently investigating the use of Bayesian probabilistic models [36], to maintain a distribution over different explicit opponent models. After every hand, a posterior probability distribution over opponent models, Pr(Opponent Model | Observations), is updated; this requires that a prior distribution, Pr(Opponent Model), is initially specified. The fully Bayesian approach to using this distribution to exploit the opponent would involve computing a Bayesian Best Response (BBR), maximizing the expected value over all possible hands and opponent strategies, given the observations. However, computing the BBR can be quite expensive, particularly in large games. One alternative suggested is to find the maximum a posteriori (MAP) strategy of the opponent (the opponent strategy that is the most likely, given the observations) and compute a best-response strategy to this MAP strategy. Another alternative is to sample opponent strategies based on the posterior distribution, and play a best-response strategy to the strategy that is chosen on each hand (Thompson Response). The MAP response method is the method most similar to the explicit modelling performed in this thesis, as here the modeller computes a single strategy based on the observations and assumes the opponent is playing this strategy. The effect of using MAP estimates in Kuhn Poker is briefly explored in Section 6.1.

Other interesting approaches to creating strong poker programs include the use of evolutionary algorithms. One such approach [5, 22] evolves a parametrized strategy while playing at a table with opponents that use fixed strategies. This is an implicit modelling approach, as a counter-strategy is developed without explicitly modelling the opponents.

Another evolutionary approach [29, 30], uses Pareto coevolution, an enhancement of standard genetic algorithms which maintains a population of strategies, by playing the strategies against each other, removing weak strategies and adding new strategies that are combinations of strong strategies from the population, and repeating this process for thousands of generations. Another approach uses Bayesian networks in the implementation of an adaptive Stud Poker player [24].

One thing that is lacking in all of these poker studies is a good metric to measure program strength. While results against test suites of simple opponents can be insightful, stronger opposition is required for an adequate measure of strength. Unfortunately, matches against strong humans only provide anecdotal information, as it is very unlikely that a statistically significant number of hands is played. Hopefully as stronger programs are developed by independent sources, a set of reference players can be created which provide a challenging and informative test for new programs.

Opponent modelling has been used with varying degrees of success in many applications besides poker. The small domains of the iterated Prisoner's Dilemma [4] and Roshambo [6] have provided some very interesting and contrasting results. In the iterated Prisoner's Dilemma, a strategy that wins nearly every tournament is the very simple "tit-for-tat" strategy which does no complex opponent modelling, but simply repeats the opponent's last move. In contrast, Roshambo programs which win require the use of opponent modelling to take advantage of exploitable opponents, while also ensuring that the program itself is difficult to model.

A recent study of universal learning in repeated matrix games [32] used similar methods (explicit and implicit modellers) to those studied in this thesis. Extensions to handle systematic opponents by way of extending information sets to include knowledge of the actions taken in the previous round(s) were also introduced. However, these extensions also increase the complexity of the model, slowing down the learning in many cases.

One interesting opponent modelling study uses past games to train a decision tree to identify particular opponents or categories of opponents by their playing style [33]. Once an opponent is identified (or assigned a style category), his moves can be simulated in a search of the game tree. A nice consequence of this research is that the decision trees can be read and interpreted in a meaningful way by humans.

An algorithm, M^*, for using an explicit opponent model when searching a game tree has resulted in improved performance of a checkers program [10, 11]. When a complete opponent model is too complex to create from only a few observations, another approach is to develop a classifier which can categorize certain moves as being weak moves; recognizing areas where an opponent is weak can lead to quick exploitation as the modeller repeatedly directs his opponent into these situations [27]. This method of learning a weakness model of the opponent has been successfully applied to simple programs for games such as Connect Four and checkers[1].

Improved artificial intelligence has recently become a high priority in commercial games, leading to efforts to create adaptive players [38, 37]. Efficiency is a key issue, as these adaptive methods must work in real-time. An interesting property of this research is that while the motivation for doing opponent modelling is usually to be able to exploit the opponent as much as possible, this is not necessarily true for commercial games. The goal of commercial games is to entertain the player, not provide an unbeatable opponent that frustrates the player, causing the player to quit playing the game. Adaptive play is also introduced via dynamic scripting, in which characters are defined by sets of rules, and the probability that rules are present in future characters depends on the success of previously created characters. This seems similar to the implicit modelling methods studied here, where counter-strategies are sampled and evaluated, and counter-strategies which produce good results are used more in the future.

Since the publication of Exp3 and Exp4 [2], research has been done on improving regret algorithms and forming more complex notions of regret. The notion of response regret has been introduced [44], which examines the short-term consequences of actions rather than just the immediate consequences. In nonzero-sum games such as the iterated Prisoner's Dilemma algorithms minimizing response regret investigate how other agents respond to being treated nicely, leading to cooperative players rather than defecting players which are created by typical regret algorithms.

One concept that has not been greatly explored in this thesis is the *value of information* [13, 16]. Each action that the modeller can take has an associated information value (corresponding to the improvement made to the model based on the information gained) and an associated cost. Using an information theory framework, the problem of optimal data-collection can be formulated as a planning problem, but such problems are infeasible to solve exactly. Regardless, the idea of the value of information can be used to create more complex data-collection strategies than those used in the thesis, if focus is directed towards actions with a higher value of information; this in turn focuses data-collection away from parameters which have been sufficiently estimated while more information is required about other parameters.

[1] The checkers programs referred to here are not nearly as strong as the strongest computer player, Chinook [34], which does no opponent modelling but instead uses large endgame databases and deep game-tree searches to defeat all challengers.

6.1 The MAP Approach to Explicit Modelling in Kuhn Poker

A recent Bayesian study of poker [36] has suggested the possibility of computing an opponent's maximum a posteriori (MAP) strategy and playing the best-response to this strategy. While this MAP strategy can be difficult to compute in large games, it can be computed for the small game of Kuhn Poker[2]. The MAP parameter estimates derived differ slightly from the estimates discussed in Chapter 3.

Consider the portion of the game tree for which the modeller learns about P2's setting of η, shown in Figure 6.1:

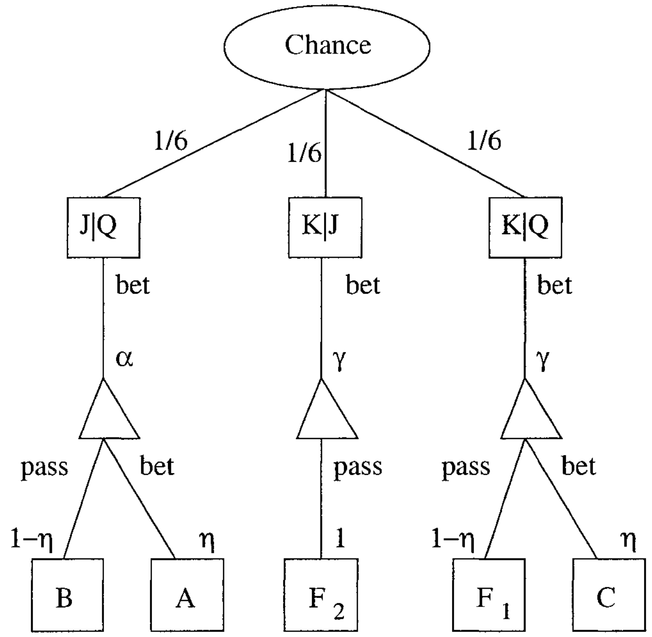


Figure 6.1: Portion of Kuhn Poker Game Tree Relevant to Estimating η

Here the numbers of times that the leaves have been reached are denoted A, B, C, F_1, and F_2. While A, B, and C are observable to P1, F_1 and F_2 events are indistinguishable, and P1 can only observe the sum of the numbers of these two events $F = F_1 + F_2$. The goal is to find the model with the highest probability of being the correct model, given the observations. It is difficult to directly compute the probability of a model given game observations, but it is much easier to compute the probability of the observations given the

[2] While Kuhn Poker is not discussed in [36], Finnegan Southey has shown derivations of the MAP estimates for Kuhn Poker in personal communications with the author.

model, which is why Bayes' Rule is important:

$$\Pr(\text{Model}|\text{Observations}) = \frac{\Pr(\text{Observations}|\text{Model})\Pr(\text{Model})}{\Pr(\text{Observations})}.$$

If all models are considered equally likely a priori, then

$$\Pr(\text{Model}|\text{Observations}) \sim \Pr(\text{Observations}|\text{Model})$$

and the task is to find a model which maximizes the right-hand quantity. Focusing just on η and the observed quantities A, B, C, and F, the goal is to find an η which maximizes

$$\Pr(A, B, C, F|\eta) =$$

$$\underbrace{\binom{A+B}{A}\eta^A(1-\eta)^B}_{(i)} \sum_{F_2=0}^{F} \underbrace{\binom{C+F}{F_2}\left(\frac{1}{2}\right)^{C+F_1}\left(\frac{1}{2}\right)^{F_2}}_{(ii)} \underbrace{\binom{C+F_1}{F_1}\eta^C(1-\eta)^{F_1}(1)^{F_2}}_{(iii)}.$$

The term (i) preceding the summation is the probability of the J|Q observations given η, while the probability of the K|? observations are being summed over all possible splittings of F into the unknown quantities F_1 and F_2. The term (ii) is the probability that P2 holds the Jack F_2 times and the Queen $C + F_1$ times ($F_1 = F - F_2$) when P1 holds the King and bets in Round One. This is multiplied by (iii), the probability that P2 would call C times with the Queen, fold F_1 times with the Queen, and fold F_2 times with the Jack.

The key difference between the estimates introduced in Chapter 3 and the estimate derived from continuing this MAP approach, is that previously a single splitting of F was considered (based on probabilities of the deal and the likelihood of the preceding sequence of events) which led to a simple formula for the estimate. The MAP approach considers all possible splittings of F and the probability of each splitting.

Continuing the MAP approach, the probability can be significantly simplified. Since this is a maximization problem, constants will be dropped as they are moved outside the summation, in order to simplify these formulas:

$$\begin{aligned}
\Pr(A, B, C, F|\eta) &\sim \eta^{A+C}(1-\eta)^B \sum_{F_2=0}^{F} \frac{(C+F)!}{F_2!(C+F_1)!}\frac{(C+F_1)!}{F_1!C!}(1-\eta)^{F_1}(1)^{F_2} \\
&\sim \eta^{A+C}(1-\eta)^B \sum_{F_2=0}^{F} \frac{F!}{F_1!F_2!}(1-\eta)^{F_1}(1)^{F_2} \\
&= \eta^{A+C}(1-\eta)^B \sum_{F_2=0}^{F} \binom{F}{F_2}(1-\eta)^{F-F_2}(1)^{F_2}
\end{aligned}$$

Recognizing that the summation is the binomial expansion of $(1 + (1-\eta))^F$, the goal is to find an η which maximizes

$$f(\eta) = \eta^{A+C}(1-\eta)^B(2-\eta)^F$$

Setting $\frac{\partial f}{\partial \eta} = 0$, one determines that η must satisfy the equation

$$(A + B + C + F)\eta^2 + (-3A - 3C - 2B - F)\eta + (2A + 2C) = 0$$

and the quadratic formula can be used to solve for η (note that only the root which lies in the interval $[0, 1]$ is considered),

$$\hat{\eta} = \frac{-b_\eta - \sqrt{b_\eta^2 - 4a_\eta c_\eta}}{2a_\eta}$$

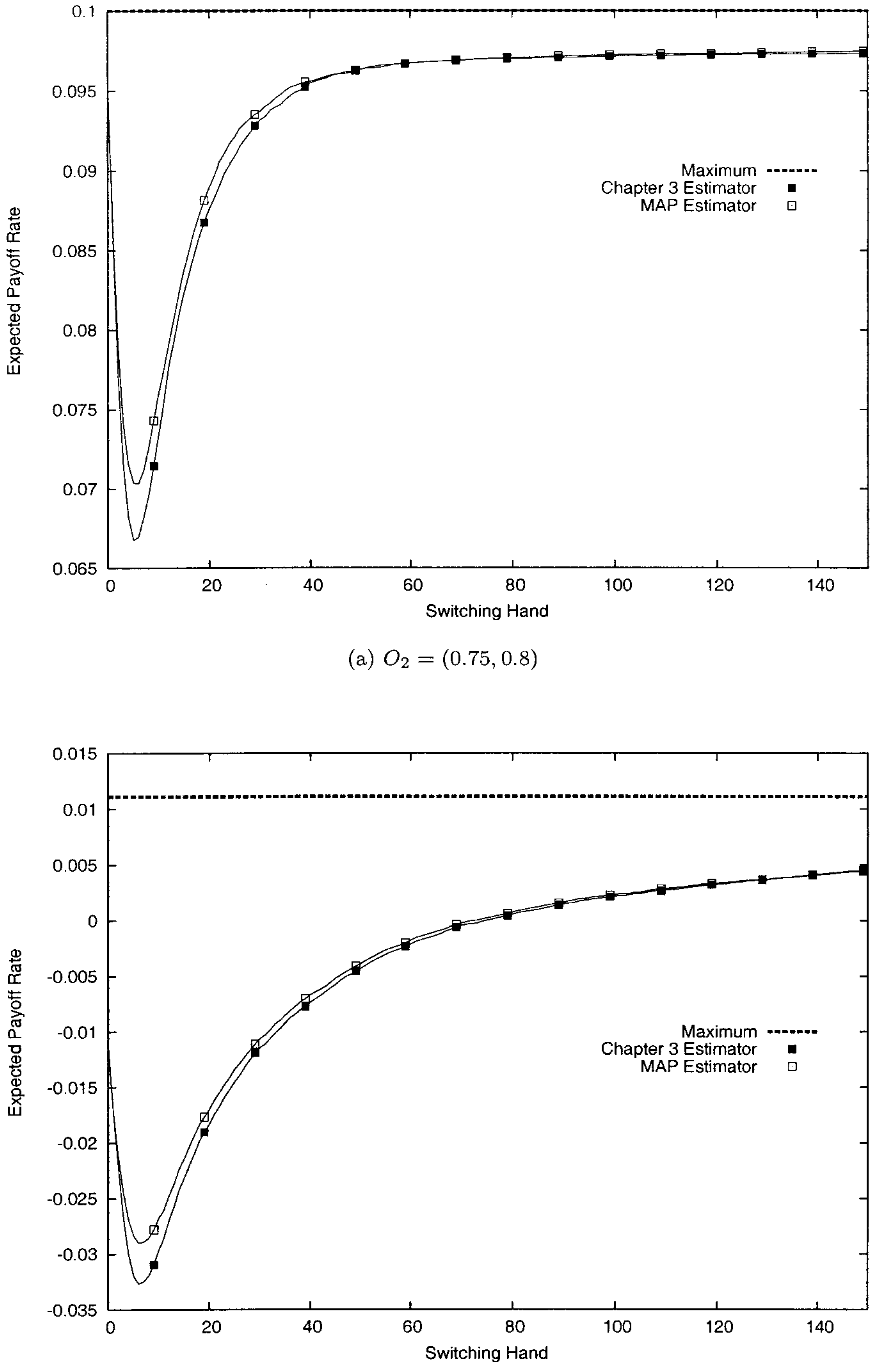
where

$$a_\eta = A + B + C + F$$
$$b_\eta = -3A - 3C - 2B - F$$
$$c_\eta = 2A + 2C$$

MAP estimates for ξ, α, β and γ can be derived similarly. The following study compares the results of the Chapter 3 estimators to the MAP estimators to see if either is more effective in practice.

Figure 6.2 shows payoff-rate plots for the explicit modeller using the BalancedExplore data-collection method with both the Chapter 3 estimators and the MAP estimators. The two estimators produce nearly identical results for all of the testpoints, except for the two testpoints shown. These testpoints, O_2 and O_3, are a little different because the initial estimates (0.5, 0.5) provides a very good initial payoff-rate to the modeller. Unlucky sequences of initial observations cause bad models to be formed in a small proportion of the trials, but the penalty for these bad models is large, while the rewards for the trials which improve their models is negligible. This causes the average payoff-rates of the models to initially decrease, before rebounding and climbing back towards the best-response rate.

In the plots shown in Figure 6.2, the payoff-rate for the modeller using the MAP estimators does not quite decrease as much as that of the modeller using the Chapter 3 estimators. This suggests that the MAP estimators are a little more likely to devalue the unlucky sequences of initial observations which cause bad models. However, the gap is quite small and following hand 40 for O_2 and hand 80 for O_3, the payoff-rates are virtually identical. Thus it appears that the MAP estimators create marginally better models for a few opponents, while producing identical results for most.



(a) $O_2 = (0.75, 0.8)$

(b) $O_3 = (0.67, 0.4)$

Figure 6.2: Payoff-Rate Comparison of MAP Estimates vs Chapter 3 Estimates

Chapter 7

Conclusions

7.1 Summary

This thesis has studied the effectiveness of two general types of opponent modelling, explicit and implicit, in a game of imperfect information. The game used was the small game of Kuhn Poker, whose small size allowed for indepth analysis of the positive and negative results achieved by the modelling methods. The test opponents used fixed strategies, as learning to model stationary opponents is a first step towards modelling nonstationary ones. Overall, this is an ideal setting for opponent modelling in an imperfect information game, as it avoids the problems of sparse data, high variance, and opponents that change strategies to remain a moving target for the modelling methods. The goal of this research was to be able to learn an exploitive model of an opponent quickly in this ideal setting, and identify any problems which make this goal difficult to achieve.

The explicit modelling techniques assume the opponent is playing a stochastic strategy, for which each undominated action has a unique parameter associated with it that specifies the probability of taking that action. The explicit modeller generates point estimates of each of the opponent's parameters, and then uses this model to identify opponent weaknesses and develop a counter-strategy which takes advantage of these weaknesses. Overcoming the partial observability of the game is a major issue, and techniques dealing with this issue were described in Chapter 3, which detailed how to create parameter estimates in situations of varying levels of information.

Implicit modelling is an approach which is basically oblivious to what precise errors are being made by the opponent. Instead, attempts are made to evaluate counter-strategies against the opponent, by playing these counter-strategies and observing the game outcomes. The implicit modelling approach was implemented in Chapter 5 by adapting the Exp3 algorithm [2] to the problem, and modifications were made to the algorithm to greatly improve the short-term performance. These modifications include doing information sharing between experts (so that multiple experts are rewarded on each hand), as well as calculating the average reward of each expert instead of the cumulative reward, and ensuring that equally likely holdings contribute equally within each expert's score.

The modelling techniques were evaluated with two key metrics, each of which assumes that the modeller learns a model during hands 1 to t (the exploration phase), and then stops all learning and plays the top-rated counter-strategy from hand $t + 1$ onwards (the exploitation phase). The first metric, the expected payoff-rate of the counter-strategy played from hand $t + 1$ onwards, indicates how good the model is at time t. The second metric, the expected total winnings (assuming a fixed-length match), factors the cost of learning the model into the evaluation. A third metric, the proportion of trials with expected total winnings above equilibrium, is essentially a supporting measure for the total winnings metric. The total winnings metric measures the average performance of the modelling methods, while the proportion above equilibrium metric indicates whether a good result on the total winnings graph is the average of a few very lucky trials and many mediocre ones, or instead an average of many good trials.

Explicit modellers using different methods of collecting data were compared in Chapter 4, where several interesting results were demonstrated. First, strategies which are equally exploitable often do not have equivalent data-collection value; there are some equilibrium strategies for the modeller that allow effective learning, while there are other equilibrium strategies that prevent learning from occuring. Second, the use of exploratory data-collection strategies (which play in an exploitable fashion in order to gain higher-quality information) allows for faster learning. However, this higher-quality data-collection often comes at an extra cost over safer data-collection strategies, and this forces the modeller to switch from exploration to exploitation early in a match if he expects to win. Third, it is not necessary to have good initial estimates to achieve good modelling results, as the impact of these estimates (positive or negative) is quickly eliminated when the estimates have low weights.

Implicit modellers, which were evaluated in experiments shown in Chapter 5, are not as successful in creating useful models in Kuhn Poker as the explicit modellers. For most test opponents, the payoff-rate graphs for the implicit modellers converge much more slowly to the best-response rate. Explicit modellers achieve better results because they attempt to understand the strategy played by the opponent, which is possible to do in this small setting. Implicit modellers ignore the strategy played by the opponent, focusing only on the scores given to the counter-strategies played. One positive aspect of the implicit modelling methods is that since they are based on the Exp3 framework, the data-collection strategy of the modeller is partially adaptive to the opponent, which often results in greater winnings for the implicit modellers during the exploration phase than the winnings achieved by the explicit modellers.

The key result of this research is that even in an ideal setting, opponent modelling can be difficult. Despite the elimination of the challenges of dealing with a very high variance game, sparse data, and a nonstationary opponent, the opponent modelling methods are not always able to find the best-response strategy within 200 hands. The fact that an opponent's decisions are often only partially observable, and that there is still variance in this small game result in incorrect models being generated. However, although the best-response strategy is not always found, in most cases the methods do discover a good counter-strategy to use against the opponent, and are able to achieve expected winnings beyond equilibrium.

7.2 Limitations

The fact that this research has been performed in an ideal setting means that it is difficult to draw general conclusions about the techniques used. The small size of the game allows the modelling methods to make many observations about each of the opponent's possible decisions, a situation that is not the case in most real-world games. One way of dealing with the lack of data available in a larger game is to use abstractions to create an approximation of the game which is smaller; for example, situations which are similar, such as when the opponent holds a very strong hand, can be treated as identical. Partial observability also becomes a much bigger problem in large games, as there are more possibilities for opponent holdings, making it very difficult to make conclusions about the opponent's decision making. One reason why implicit modelling techniques should be easier to implement in large games is that these techniques don't require knowledge of the opponent's hand.

Another major assumption of this research is that the opponent's strategy is fixed. A strong player in a real-world game such as poker is likely to vary his strategy over the course of a match. Many players are even known to present a specific "table image" (the model that the opponents develop of the player based on his actions) early in a match, before radically changing strategies to take advantage of their opponents' models. Thus being able to model nonstationary opponents is a key to success against strong players. One method of dealing with nonstationary opponents is to put more emphasis in the model on recent observations; the decaying of earlier observations will help to keep up with an opponent changing strategies, but may prove to be worse against stationary opponents, as useful older data is forgotten.

The assumption that the modeller plays a learning strategy from hands 1 to t and then freezes the model and plays only the recommended counter-strategy from that point on is a little too simplistic. First of all, it is likely that the modeller would continue to learn and refine his model after switching to an exploitive strategy, as he continues to make observations about the opponent; unfortunately, cases do arise in real games where the modeller's counter-strategy will prevent him from learning valuable information which might make him change his model. In a real game it is also unlikely that the modeller would switch to playing only the recommended counter-strategy for a couple of reasons; one is that the counter-strategy might not allow the modeller to refine and correct his model if it is incorrect (or becomes incorrect when the opponent changes strategies). The other reason is that the modeller does not want to become too predictable, which is likely to occur if he constantly repeats the same strategy, and could result in the opponent changing strategies to counter-attack. A change to the explicit model that might help in maintaining a variety of play (which hopefully aids continued learning) is to fit each opponent parameter to a distribution of possibilities, rather than a single point estimate; the counter-strategy recommended by the model could then be a mixture of the counter-strategies to each of the opponent's possible parameter settings.

The explicit modellers presented in this thesis use very simple strategies for collecting data, as they repeatedly use a single fixed strategy. Data-collection could be improved by changing strategies when some parameters have been precisely estimated and others require greater attention. Another alternative is that an explicit modeller could make use of an Exp3-like framework, playing promising counter-strategies more often during the exploration phase, in order to better discriminate between the promising strategies and increase winnings during the exploration phase as well.

The implicit modelling techniques presented here are very limited due to the lack of information sharing performed in comparison to that of the explicit modellers. For implicit modelling to achieve similar results, increasing the amount of information sharing is probably the only answer; the key is to ensure expert updates are balanced. Implicit modelling techniques do have many nice properties, including having zero average external regret in the long-term and being easy to implement, so if the information-sharing problem can be solved, these methods might become the preferred opponent modelling methods.

7.3 Final Word

Finding game-theoretic solutions to large games of imperfect information, such as Texas Hold'em poker, is beyond the limits of today's computational technology. In order to develop strong computer players for these games, opponent modelling techniques must be used to adapt to different opponents. This thesis has compared two general types of opponent modelling in an ideal setting, and has showed that modelling in this ideal setting is a non-trivial problem. In the process, analysis has shown why the difficulties arise, and which modelling methods best deal with these difficulties. Although modelling in this ideal setting has not produced perfect models of the opponents, there have been many positive results, including showing that opponent modelling is often better than equilibrium solutions. Finally, while the techniques of explicit modelling are superior in the setting studied here, both explicit and implicit methods hold promise for larger games.

Bibliography

[1] N. C. Ankeny. *Poker Strategy: Winning with Game Theory.* Basic Books, Inc., 1981.

[2] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: the adversarial multi-armed bandit problem. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 322–331. IEEE Computer Society Press, Los Alamitos, CA, 1995.

[3] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: the adversarial multi-armed bandit problem. In *Electronic Colloquium on Computational Complexity*, volume 7, 2000.

[4] R. AxelRod. More effective choice in the prisoner's dilemma. *Journal of Conflict Resolution*, 24(3):379–403, 1980.

[5] L. Barone and L. While. Adaptive learning for poker. In *Proceedings of the Genetic and Evolutionary Computation Conference 2000 (GECCO 2000)*, pages 560–573, 2000.

[6] D. Billings. The first international roshambo programming competition. *International Computer Games Association Journal*, 23(1):3–8, 42–50, 2000.

[7] D. Billings, M. Bowling, N. Burch, A. Davidson, R. Holte, J. Schaeffer, T. Schauenberg, and D. Szafron. Game tree search with adaptation in stochastic imperfect information games. In N. Netanyahu Y. Bjornsson and J. van den Herik, editors, *Computers and Games'04*. Springer-Verlag, 2004.

[8] D. Billings, N. Burch, A. Davidson, R. Holte, J. Schaeffer, T. Schauenberg, and D. Szafron. Approximating game-theoretic optimal strategies for full-scale poker. In *Eighteenth International Joint Conference on Artificial Intelligence (IJCAI'2003)*, 2003.

[9] D. Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6:1–8, 1956.

[10] D. Carmel and S. Markovitch. Learning models of opponent's strategies in game playing. In *Proceedings AAAI Fall Symposion on Games: Planning and Learning*, pages 140–147, 1993.

[11] D. Carmel and S. Markovitch. Incorporating opponent models into adversary search. In *AAAI National Conference*, pages 120–125, 1995.

[12] V. Chvátal. *Linear Programming.* W. H. Freeman and Company, 1983.

[13] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley, 1991.

[14] A. Davidson. Opponent modelling in poker: Learning and acting in a hostile and uncertain environment. Master's thesis, University of Alberta, 2002.

[15] A. Davidson, D. Billings, J. Schaeffer, and D. Szafron. Improved opponent modeling in poker. In *International Conference on Artificial Intelligence (IC-AI'2000)*, pages 1467–1473, 2000.

[16] D. Draper, S. Hanks, and D. Weld. Probabilistic planning with information gathering and contingent execution. In K. Hammond, editor, *Proceedings of the Second International Conference on AI Planning Systems*, pages 31–36, Menlo Park, California, 1994. American Association for Artificial Intelligence.

[17] N. Findler. Studies in machine cognition using the game of poker. *Communications of the ACM*, 20(4):230–245, 1977.

[18] D. Foster and R. Vohra. Regret in the on-line decision problem. *Games and Economic Behaviour*, 21:40–55, 1997.

[19] D. Gale and S. Sherman. Solutions of finite two-person games. *Contributions to the Theory of Games*, 1:37–49, 1950.

[20] A. Greenwald and A. Jafari. A general class of no-regret learning algorithms and game-theoretic equilibria. In *Proceedings of the 2003 Computational Learning Theory Conference*, pages 1–12, 2003.

[21] S. Karlin and R. Restrepo. Multistage poker models. *Contributions to the Theory of Games*, 3:337–364, 1957.

[22] G. Kendall and M. Willdig. An investigation of an adaptive poker player. In *Australian Joint Conference on Artificial Intelligence*, pages 189–200, 2001.

[23] D. Koller and A. Pfeffer. Representations and solutions for game-theoretic problems. *Artificial Intelligence*, 94(1):167–215, 1997.

[24] K. Korb, A. Nicholson, and N. Jitnah. Bayesian poker. In *Uncertainty in Artificial Intelligence*, pages 343–350, 1999.

[25] H. W. Kuhn. Extensive games and the problem of information. *Contributions to the Theory of Games*, 2:193–216, 1950.

[26] H. W. Kuhn. A simplified two-person poker. *Contributions to the Theory of Games*, 1:97–103, 1950.

[27] S. Markovitch and R. Reger. Learning and exploiting relative weaknesses of opponent agents. *Autonomous Agents and Multi-Agent Systems*, 10:103–130, 2005.

[28] J. F. Nash and L. S. Shapley. A simple three-person poker game. *Contributions to the Theory of Games*, 1:105–116, 1950.

[29] J. Noble. Finding robust texas hold'em poker strategies using pareto coevolution and deterministic crowding.

[30] J. Noble and R. A. Watson. Pareto coevolution: Using performance against coevolved opponents in a game as dimensions for pareto selection. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H. M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 493–500, San Francisco, California, USA, 7-11 2001. Morgan Kaufmann.

[31] M. Osborne and A. Rubinstein. *A Course in Game Theory.* MIT Press, 1994.

[32] J. Poland and M. Hutter. Universal learning of repeated matrix games. Technical Report IDSIA-18-05, 8 2005.

[33] J. Ramon, N. Jacobs, and H. Blockeel. Opponent modeling by analysing play. In M. Bowling, G. Kaminka, and R. Vincent, editors, *Proceedings of Workshop on agents in computer games*, 2002.

[34] J. Schaeffer. *One Jump Ahead: Challenging Human Supremacy in Checkers.* Springer Verlag, 1997.

[35] S. Smith. Flexible learning of problem solving heuristics through adaptive search. In *International Joint Conference on Artificial Intelligence*, pages 422–425, 1983.

[36] F. Southey, M. Bowling, B. Larson, C. Piccione, N. Burch, and D. Billings. Bayes' bluff: Opponent modelling in poker. In *21st Conference on Uncertainty in Artificial Intelligence (UAI-2005)*, 2005 (to appear).

[37] P. Spronck, I. Sprinkhuizen-Kuyper, and E. Postma. Online adaptation of game opponent ai in simulation and in practice. In Q. Mehdi and N. Gough, editors, *Proceedings of the 4th International Conference on Intelligent Games and Simulation (GAME-ON 2003)*, pages 93–100, 2003.

[38] H.J. van den Herik, H.H.L.M. Donkers, and P.H.M. Spronck. Opponent modelling and commercial games. In G. Kendall and S. Lucas, editors, *Proceedings of IEEE 2005 Symposium on Computational Intelligence and Games CIG05*, pages 15–25, 2005.

[39] J. von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928.

[40] J. von Neumann and O. Morgenstern. *The Theory of Games and Economic Behavior.* Princeton University Press, 2nd edition, 1947.

[41] D. Waterman. A generalization learning technique for automating the learning of heuristics. *Artificial Intelligence*, 1:121–170, 1970.

[42] E. W. Weisstein. Simpson's paradox. http://mathworld.wolfram.com/SimpsonsParadox.html, 1999.

[43] N. Zadeh. *Winning Poker Systems.* Prentice Hall, 1974.

[44] M. Zinkevich. *Theoretical Guarantees for Algorithms in Multi-Agent Settings.* PhD thesis, Carnegie Mellon University, 2004.