Random Parameters Analysis of Competitive Agents in an Imperfect Information Environment

Alex Moore, BSc(Mathematics)

Department of Statistics, Macquarie University

This thesis is presented for the degree of Doctor of Philosophy

September 2011 (original), September 2012 (with corrections)

Contents

List of Figures

Summary

Many real life problems have to be solved by making rapid decisions in competitive environments with only partial information available. This thesis describes the development of an adaptive learning algorithm to maximise expectancy in such cases. The working example in this thesis is a suite of computer programs that form an adaptive learning system to play on-line Texas Hold'em poker. This provides a basis for constructing systems for various up-scaled problems such as financial markets trading.

In this thesis I have opted for a statistical approach to machine learning. The methods described in this thesis are not based on human cognition. Rather, they utilise the advantages of modern computing power in database recall and computational power and do not require the human skills of perception and pattern recognition that modern computers only possess in very small measure.

This approach is a example of real world applicable unsupervised learning [8]. It relies heavily on statistical and probabilistic solutions to certain problems rather than seeking hard and fast deterministic answers.

The focus in this thesis is on the mathematics underpinning the system rather than details such as how to decode on-line sites.

What now follows is a summary of the thesis in more detail:

Much of the research into poker has built on the concepts developed by Von Neumann [27] and Borel [4]. Game Theory concepts have been developed and used in conjunction with cutting edge techniques such as Neural Networks to develop playing algorithms in a similar way to which games such as Chess or Checkers have been solved. These techniques are not necessarily gambling related, even though poker is usually played as a gambling game. The problem with the Game Theory approach is that analysis of full scale poker rapidly becomes intractable. Prominent researchers in the Game Theory approach ([3], [18], [21],[20],[2], [1]) have recognised this and much of the subsequent research has centred around reducing the size of the problem to be solved. Derivation of a Game Tree for a reduced problem was also the approach adopted by von Neumann and Borel. This approach has been successful and can be summarised as finding an exact solution to an approximating problem. Methods seeking approximations to the full problem that retain the properties of the real problem have been developed and are ingenious.

In this thesis a different approach has been used. Instead of seeking approximating problems which may be solved exactly, approximate solutions to the full problem are sought. Regression techniques have been used to find approximate solutions to the full problem. Rather than try to tune a

strategy engine using Neural Networks to solve the problem in a human-like way, the population variance in preferences (as measured by coefficients) has been measured and past data for given contexts is used to estimate where on the population distribution of preferences a particular context resides. The resulting system still "learns", but in a different way to a Neural Network. The play has been reduced to a sequence of actions which are chosen to maximise profit expectation. This has much in common with financial market trading where a sequence of trades is likewise chosen to maximise profit.

The Game Theory approach also maximises profit expectancy, but because that approach necessitates analysis of a computationally intractable Game Tree, the expectation actually maximised is that of an approximating problem.

In a sense the approach adopted here is the opposite of the approach adopted by the authors mentioned above in that they seek exact solutions to an approximating problem whereas this research seeks approximate solutions to the exact problem. It is expected that the methods developed here would apply to more general problems than poker but, somewhat paradoxically, there are other variants of poker to which the regression approach developed here would not be suited. Tournament play and No-Limit poker are not examples of a possibly infinite sequence of small trades (bets), and a Game Theory approach would work better for them.

While the discussion in this thesis focuses on on-line poker, the methods used are applicable to a wide range of applications not normally regarded

as gambling. However, many problems can be simplified when it is recognised they can be formulated as gambling problems. In this thesis the term *gambling* will be used to denote any activity where finite resources have to be invested for returns that are uncertain *a priori.* It should be stated that the use of the term *gambling* does not imply that reliance on luck is the main factor in these activities. In fact, the poker robot discussed in this thesis wins with high regularity and this is due in no small part to recognising some events can only be predicted probabilistically and the appropriate course of action is the one that maximises expected advantage, that is, one that maximises

(Probability of occurrence $\times$ Expected return) $-$ Expected Investment.

There are many gambling strategies some common examples being

- Reliance on luck or hunches.
- Variance reduction or *hedging.* A few words should be said about this because it is quite common, especially in financial markets trading and sports betting. Consider a fair toss of an evenly balanced coin for which we are given bookmaker odds of 2:1 on heads and 4:6 on tails. It is shown on page 9 that in this situation both sides can be backed profitably. Suppose we invest \$10.00 and bet \$3.57 on heads and \$6.43 on tails. If heads wins we collect 2 x \$3.57 plus our investment of \$3.57 for a total of \$10.71. If tails wins we collect 4 / 6 x \$6.43 + \$6.43 back

or \$10.71. Either way we win \$0.71 per spin and never lose. Because profits can be guaranteed, this sort of system is sometimes thought to be the best.

- Maximising Expectancy. This is the central strategy discussed in this thesis. The expectancy to be maximised will be profit expectancy. Consider again the coin toss experiment above. If we only take heads, we win 50% on turnover which can be calculated as 3.00 decimal odds multiplied by probability 0.5 giving an advantage 1.5 (as return on investment) or 1.5 - 1 = 0.5 (as profit on investment), or an expectancy of 0.5 x \$10.00 = \$5.00 per spin instead of \$0.71 per spin. This of course will also entail some fluctuations, half the bets will be losing bets and we could be behind after some number of bets. But, after a sufficiently high number of trials, our win rate will exceed that of any other strategy.

To some extent, the "best" strategy to adopt can be a matter of preference. Various institutions entrusted with shareholders' money are often forced to adopt a variance reduction approach because of the attitude of investors to fluctuations in profit level.

Central Themes

- The central strategy advocated in this thesis will be to maximise profit expectancy.

- The modelling strategy employed will be to seek approximations to profit maximimising strategies to the full problem.

- The methods used will be to develop a basic model assuming a homogeneous population of opponents and game contexts.

- Assumptions of homogeneity of opponents and game conditions will be progressively relaxed.

- As part of the relaxation of these assumptions model properties of adaptation or basic "learning" of the model to adapt to a changing environment but still exhibit profit maximising behavior will be observed.

Statement by Candidate

All of the work in this thesis was completed while I was a candidate for the degree of Doctor of Philosophy at Macquarie University. None of the content of this thesis has been previously submitted for a higher degree neither at Macquarie University nor at any other University or Institution.

Alex Moore,

14 September, 2012

Acknowledgements

I would like to thank the following people. Peter Petocz was the long suffering supervisor for this thesis and bravely tried to keep the whole thing on track. David Bulger co-supervised and tried to keep the mathematics comprehensible, consistent and suggested better notation than what was originally submitted. David also pointed out some embarrassing errors in the mathematics which were fortunately able to be fixed. Geoff Smith contributed to the shape and substance of this thesis with his expert knowledge of LaTeX and scientific writing generally. Wendy Noble as an academic writing specialist provided much needed assistance with all aspects of writing and had to endure all the fits of depression that result when untalented writers attempt a thesis. To all these people a huge debt of gratitude is owed. This thesis would not have been possible without them.

A very special and very personal acknowledgment must go to Lachlan, my son. I hope that when you are a little older you will understand why I have not discussed much of this with you. I wanted to shelter you from discussions about gambling which I feel might be unhealthy for you at your

age. I have tried to show you how beautiful and powerful mathematics is and how it constantly surprises us with what it can do. When I feel you are ready for it, nothing will give me greater pleasure than to fully explain to you exactly what Dad does in his office.

Alex Moore

September, 2012.

Chapter 1

Trading in uncertainty

Yes, I am a very rich man but I'd trade it all for just a little bit more.—C. Montgomery Burns

This chapter contains an introduction to gambling and a discussion of its common concepts and terminology. In this thesis, the word *gambling* is used broadly to mean *any activity that requires the investment of finite resources for an uncertain return.* This is followed by a discussion of the poker variant investigated in this thesis, namely Texas Hold'em poker. Some background on poker terminology is also provided. The chapter concludes with a discussion of modelling concepts and a rationale for choosing poker as an application for the ideas developed in this thesis.

1.1 Background Material

1.1.1 Definitions

1. *Advantage* (A) is the expected value of return per unit investment:

$$A = E[\frac{\text{Return}}{\text{Investment}}]$$

Let $E[\pi]$ = expected profit. It can easily be seen that $E[\pi] = A - 1$.

2. *Imperial odds* are expressions of a dividend showing bettor's profit for a successful wager.

3. *Decimal odds* are expressions of a dividend showing bettor's return for a successful wager. This style is fast becoming the standard and will be the format adopted in this thesis unless otherwise stated. Almost always decimal odds have values $D > 1$. It would be unusual to make a wager where the return is less than the investment, thereby guaranteeing a loss.

4. A $\pi-$*game* is a set of events that can be expressed as a sequence of bets.

5. For a $\pi-$game, the (profit) expectancy, or *expectation* is

$$\pi_\Sigma = \sum_{n=1}^{N} (p_n D_n - 1) I_n$$

where p_n is the bet success, D_n is the decimal odds dividend and I_n is the amount invested on the *nth* bet of the π-game.

6. Strategies that maximise π_Σ are called *Expectation Maximisers.* The notation adopted here for these are $\sup(\pi_\Sigma)$. The suprema of $Max(\pi_\Sigma, 0)$ is what is sought.

7. An *advantage game* is a π-game where there exists some Expectation Maximiser such that $\pi_\Sigma > 0$.

8. The *Implied Probability* ($\widehat{p}$)of an event is the dividend required such that $A = 1$. For decimal odds, this is simply the reciprocal, for Imperial Odds

$$a : b,\ \widehat{p} = \frac{b}{a+b}$$

If Implied Probability = Actual Probability, then $A = 1$ and the bet is considered "fair".

9. *Market percentage* $M = \sum_{n=1}^{N} \widehat{p_n}$ is the sum of the Implied Probabilities for all of the possible outcomes of an event. $(M < 1) \iff$ arbitrage possible.

10. In a $\pi-$game, our *Turnover* is the amount we invest, that is the sum total of all bets.

11. A *respondent* is a member of some population under consideration. Alternatively a *population* can be thought of as the set of all possible

respondents.

12. A *Generic Opponents model* in the context of poker is a model that assumes opponents are homogeneous throughout the population.

13. A *Random Parameters Model* is a model that has parameters that may vary across a possibly heterogeneous population.

14. A *Specific Opponents model* treats opponents as possibly heterogeneous in that behavioral preferences differ.

15. A *Context Specific model* treats the context being considered as possibly heterogeneous. Thus we may not know much about each individual player, but may know that for example players play differently on large limit tables than on small ones. Larger groups of players may also behave differently than single opponents. This can be regarded as a generalisation of Specific Opponents models. The same players may play differently when, say on a larger limit table or when confronted by a larger number of opponents.

1.1.2 Gambling, Probability and Odds

A familiarity with basic notions of probability will be assumed. Dividends can be expressed in many ways:

- Common expressions are in the form of *odds*. A bookmaker may offer odds in the form of, say 3:1. This an Imperial Odds expression. A winning wager of 1.00 is paid 4.00 for a clear profit of 3.00. Odds of 5:4 means that for every four units wagered a winning bet will result in five units of profit. Odds expressed in this way are the reverse of odds as expressed for Binary Logistic regression log-odds transformations [12] where 1:3 means the same as 3:1 in this context.

- Totalisator agencies often offer odds in the form of dividends such as 5.00, where a winning wager of 1.00 returns 5.00 for a profit of 4.00 This is a Decimal Odds expression. Decimal odds express the bettor's return per unit of investment for a successful wager. By subtracting the investment unit (usually 1.00) from the dividend, one can express decimal odds as imperial odds. Thus a tote dividend of 6.00 is the same as a bookmaker dividend of 5:1 assuming a 1.00 betting unit. Increasingly, decimal odds are phasing out imperial odds as the standard used to express dividends. Unless specified otherwise decimal odds expressions will be used in this thesis. Normally decimal odds values are greater than one.

- Dividends can be thought of as probabilities: a fair toss of an evenly balanced coin should pay a decimal odds dividend of 2.00 to be "fair", because the Implied Probability would then be the same as the actual probability. If a bookmaker displays (Imperial) odds of 6:4, they are

paying as if the probability is 4/10. Of course, if the actual probability is, say 1/5 then the actual dividend should be 4:1 to be "fair". The bookmaker is building a profit into the dividends provided because the implied probability exceeds the actual probability. It can easily be seen that the implied probability for decimal odds is the reciprocal. Thus, a decimal odds dividend of 5.00 has an implied probability of 1/5 and is the same thing as imperial odds of 4:1.

- The sum of all possible Implied Probabilities is the Market Percentage. If the market percentage exceeds 1 as is usually the case, the house has an inbuilt edge and if wagers are made in proportion to each item's implied probability, the house will profit by the market percentage. This is why bookmakers often move their prices, so they can keep a market percentage of over 1, but induce the public to wager amounts such that the proportions are as close as possible to the implied probabilities so that the bookmaker is guaranteed an overall profit.

- If the market percentage is less than 1, it is possible to back all alternatives and win. This is called Arbitrage. For instance, if we have a possibly biased coin and a bookmaker gives 9:10 on heads and 4:5 on tails, then the market percentage is

$$\frac{10}{19} + \frac{5}{9} \approx 1.082.$$

The bookmaker has an average of a little over 8% built into this market

and the market percentage here is a little over 108% . It will be shown shortly that a profit cannot be guaranteed from betting in this case and no arbitrage opportunity exists. On the other hand, if the (Imperial) odds are 9:10 heads and 5:2 tails the market percentage is

$$\frac{10}{19} + \frac{2}{7} \approx 0.812$$

or an 81.2% market. Then for a 9.00 stake a bettor could bet 6.00 on heads at 9:10 and 3.00 on tails at 5:2. If heads wins the bettor's return is 11.40 and if tails wins the bettor's return is 10.50, so the bettor will win with certainty every time no matter what bias the coin may have.

Consider an event with an exhaustive list of implied probabilities; every possible outcome has a dividend with an associated implied probability. The return on a wager on the *nth* possible outcome is $R_n = D_n\, I_n$ where D_n is the (decimal) dividend and I_n is the investment.. This can also be expressed as

$$R_n = \frac{I_n}{\widehat{p_n}}$$

where $\widehat{p_n}$ is the *nth* implied probability. If a bettor is to make an overall profit for all possible bets that can be made on the event;

$$R_n > \sum_{i=1}^{N} I_i.$$

That is, return exceeds total investment. Consider the situation where a bettor wishes to back all alternatives and win. The dividends and therefore the implied probabilities are all known. Each dividend is assumed to be finite and greater than 1, so each results in some profit for that particular wager if the wager is successful. This meets the criterion for implied probabilities discussed above, putting each in the open interval (0,1). Next, consider what an implied probability $\widehat{p}_k$ can also mean. Writing

$$\widehat{p}_k = \frac{a_k}{b_k},$$

it can be seen from the definition of implied probability that a wager of a_k results in a return of b_k if the bet is successful. Dividends and therefore implied probabilities are all rational numbers, but for the purposes of this discussion we do not even require that. All we require is $0 < a_k < b_k$. For a collection of implied probabilities relating to an exhaustive list of alternatives where every possible outcome of a bet is covered, each of the implied probabilities can be expressed as a ratio with a common denominator,

$$\widehat{p}_k = \frac{\widehat{a}_k}{d}, \ \forall k.$$

This is still the same implied probability. A wager of $\widehat{a}_k$ results in a return of d. Suppose

$$\sum_{k=i}^{N} \frac{\widehat{a}_k}{d} < 1.$$

Then

$$\sum_{k=1}^{N} \widehat{a}_k < d.$$

Thus, if the market percentage is less than 1, the sum total of all investments is less than the return, which has been constructed to be the same in each case. It follows that if the market percentage is less than 1, a bettor can bet on each alternative and profit irrespective of the result. It can also readily be seen that if the return is greater than the sum total of all investments then again the market percentage is less than one. If the market percentage is greater than 1, then the sum of possible investments exceeds the sum of all possible returns, so a bettor cannot be guaranteed of a profit by backing every alternative. It therefore follows:

Lemma 1 *Arbitrage Opportunity Exists* $\Leftrightarrow$ $M = \sum_{n=1}^{N} \widehat{p}_n < 1$

As a very simple example, suppose a bettor has an exhaustive list of two possible alternatives with dividends $D_1 = 10.00$ and $D_2 = 1.50$. Implied probabilities are

$$\widehat{p}_1 = \frac{1}{10.00} = \frac{3}{30}$$

and

$$\widehat{p}_2 = \frac{1}{1.5} = \frac{20}{30}.$$

The market percentage is 23/30 (less than 1), so in this case an arbitrage opportunity exists. If the bettor invests $\widehat{a}_1 = 3.00$ and $\widehat{a}_2 = 20.00$, then the bettor has invested a total of 23.00 and will return 30.00 irrespective of

the outcome.

For decimal odds, a bet's advantage is $A = pD$ where p is the actual (not implied) probability of the event and D is the decimal odds dividend. If A is less than 1, then the bettor can expect to lose in the long term. Conversely, if A exceeds 1, in the long term the bettor will win. If $A = 1$ then returns $R \to I$ as the number of trials gets arbitrarily large. This formalises what is described as "fair" above. Expected profit per investment, $E[\frac{\pi}{I}]$ can easily be seen to be $A - 1$.

Anything that can be described as a sequence of bets will be called a $\pi-$game in the context of this thesis (so poker, blackjack, real estate, options trading would all be $\pi-$games). This terminology was adopted so as not to cause confusion with already existing terminology in the well defined area of Game Theory and also so that the betting of the outcomes of an event can be seen as distinct from the performance of the event.

1.1.3 The Kelly criterion

Consider how much should be wagered in a given situation. The seminal work in this area is Kelly [13], which was actually concerned with data communication. He specified the problem as one where a gambler has a private wire transmitting the results of a chance situation before they become common knowledge and the gambler may still bet at the original odds. The specified problem concerned a noisy transmission channel so there was doubt

as to whether the received signal is the same as the transmitted signal. Kelly was specifying the problem to investigate how much of their capital a gambler should bet on the result of such a noisy signal. It is unclear how useful this was to the field of data transmission. What is very clear is that his results are very useful to gambling situations. The essence of Kelly's paper is that the criterion to be adopted is to maximise the expected value of the logarithm of capital. This has nothing to do with any value function the gambler may have attached to their capital, but merely with the fact that it is the logarithm which is additive in repeated bets and to which the Law of Large Numbers applies.

The fraction f^* of the entire bankroll that should be wagered is given by

$$f^* = \frac{bp + p - 1}{b}$$

where p is the probability of winning and b is the odds-to-one received upon winning. Odds-to-one are the odds expressed against a unit investment. For example 5:1 would have odds-to-one of 5 and 5:2 would have odds-to-one of 2.5. It can immediately be seen that $(b + 1)$ is simply the decimal odds representation of the dividend and so

$$p(b + 1) - 1 = A - 1 = E[\pi]$$

where A is the advantage and $E[\pi]$ is the expected profit per investment. But if the dividend is odds-on (actual probability exceeds 0.5) , then the bet-to-

one is less than one. If additionally the expected profit is positive, then the above formula would seem to imply that more than the entire bankroll be wagered on every proposition that is more than 50% likely and expected profit is positive. This is obviously not what Kelly advocates in his paper and it is suggested here that this formula, that unfortunately often seems to be applied as circulated, contains a misprint. The circulation of the alleged misprint is common enough that the above formulation appears on Wikipedia as the correct formula. It is proposed here to modify the formula slightly to read

$$f^* = \frac{bp + p - 1}{b + 1} = \frac{E[\pi]}{\omega}$$

where ω is the dividend expressed in decimal odds. For $E[\pi] > 0$, $A > 1$ and so $\omega p > 1$. It can now be seen that

$$f^* = p - \frac{1}{\omega}$$

so

$$f^* \to p. \text{ as } \omega \to \infty$$

If, for example a bettor was offered a dividend of 1000.00 on a certain event, then under the revised Kelly formula the recommended bet size would be

$$f^* = \frac{p\omega - 1}{\omega} = \frac{999}{1000},$$

or 99.9% of available funds. The difference between this and all of the available funds is to facilitate betting in proportion to advantage , so theoretically larger wagers can be made at even more outrageous dividends.

Even adopting the Kelly criterion can result in more volatility than many are comfortable with. Various refinements have been proposed such as betting half Kelly. The view adopted in this thesis—admittedly based on experience only—is that so long as the (revised) Kelly recommendation is not exceeded, then the best level is a matter of personal preference and may depend on the specific context. Note that for disadvantage and break square bets ($E[\pi] = 0$) the optimal fraction is zero, so the player must have some positive profit expectancy to justify betting at all. Kelly's criterion was adopted and popularised by Edward O. Thorpe [25] who analysed casino games, especially blackjack.

1.2 Texas Hold'em Poker

The only poker variant that will be considered in this thesis is *Texas Hold'em*. Texas Hold'em has high and increasing popularity. It is offered on most, if not all, on-line poker sites. Interested readers may refer to [11] for an overview or [23] for a detailed and expert discussion.

Texas Hold'em has many subvarieties. There is *No Limit* Texas Hold'em, where the bet on any hand can be any amount up to an opponent's holding. This is an exciting and cut-throat variant, well suited to television due to

the knockout nature of the tournament and the size of the bets placed.

In *Pot Limit* Texas Hold'em players may bet up to the size of the *Pot*, the total pooled bets placed so far in the game. This variant is available on most on-line sites.

This thesis concentrates on modelling *Limit* Texas Hold'em, where each table has predefined maximum and minimum bets. For the first half of the game all bets are for the minimum amount exactly and for the last half of the game all bets are for the maximum amount exactly. This is very popular in on-line sites because of the range of costs to play. So-called "micro-limit" games can be found where the minimum bet is 2 cents and the maximum bet is 4 cents, while tables with limits $1000 and $2000 or even larger are commonly available.

1.2.1 The play in Texas Hold'em

At the start of play, one seat is designated as the *dealer* and each player takes it in turn to be the dealer. In the online games the dealing of cards is automated, while in casino and tournament play a professional dealer is normally used to actually deal the cards. The seat to the left of the dealer is the *Small Blind* the next seat to the left is the *Big Blind*. The players in these positions post forced bets at the start of the game, usually half of the minimum bet for the table for the Small Blind and the table minimum bet for the Big Blind. Different tables have different rounding rules for the small blind but this rounding error represents such a tiny part of expectancy that

it may be neglected. When a player joins a table, that player posts a *buy in* of the same size as the Big Blind. For this reason, most players wait until it is their turn to post a Big Blind before sitting down and most online sites have a feature whereby one can elect to wait for the Big Blind before the site's software seats the player. So, usually a game starts with a Big Blind and a Small Blind in the *pot*. The pot is the pooled collection of all bets minus the *rakes*. A rake is a (usually small) percentage of each pot that the house keeps for itself.

The game starts once the blinds and buy ins have been posted. The first stage is the *Pre-Flop*. Each player is dealt two face-down *hole cards*. The next stage is the *Flop* where three *community cards* are dealt face up in the middle for all players to use. Then comes the *Turn* where an extra community card is dealt and finally the *River* where a fifth community card is dealt. The *best hand* is the best five card poker hand that can be constructed with the hole cards and the community cards. Each stage has up to three rounds of betting. For the Pre-Flop and the Flop all bets are for the table minimum bet, while for the Turn and the River all bets are for the table maximum bet. If there are no more bets in any round then the game goes to the next stage. If a player bets, then the other players can *call* by matching the bet, they can *raise* the bet by a further increase, or they can *fold*, in which case they no longer participate in that game and surrender any bets they may have made. Players that have not folded are said to be *active*. In the event of a *raise* all players before the raise must either call

by matching the raised bet or fold. If no bets have been made a player can elect to *check* by doing nothing. If the game is left with only one active participant at any stage, then that player is deemed the winner and they take all the money in the pot after any rakes have been deducted. In this situation the winning player is not obliged to reveal their cards but can elect to do this. If after the final round at the River more than one player is still active in the game, the game is decided by a showdown where all active players reveal their hands and the best five card poker hand constructed from the seven cards at any player's disposal wins. In the event of a tie, the pot is split equally between all tied players.

The rules of the game are not complicated, but the game is surprisingly subtle due to the imperfect information available to the players. The fact that a player can hold a weak hand and play as if holding a very strong hand and *vice versa* introduces many dimensions to the strategy. A well-known strategy is *bluffing* where a player bets very aggressively with a weak hand. This is done in the hope that other players will mistakenly think they are facing a strong hand and fold rather than putting extra money into the pot. Obviously if the other players do not fold, then the bluffing player will have invested bets with only a weak hand's chance of winning. The opposite of bluffing is *slow playing* where a player checks and calls and plays conservatively with a strong hand. They do this in the hope that other players will not fold and may possibly even raise into this very strong hand, thereby increasing the pot for the slow player. These are all pre-

defined strategies that have been developed by human players. These will not be discussed in detail because the modelling developed in this thesis seeks to arrive at strategies as emergent phenomena, behavior that was not specifically programmed but rather which has been independently "learned" by the process. The interested reader is referred to any of the books by David Sklansky [23], [15] a celebrated poker professional whose books provide an engaging account of human strategies which are detailed enough for the vast majority of readers.

1.3 Poker as an advantage game

There are many strategies that are used in poker. The strategy that will be dealt with here is that of maximising profit expectancy. A value for $\sup(\pi_\Sigma)$ (page 3, definition 6) is sought. In a $\pi-$game (page 2, definition 4) a variety of trading strategies could be employed. Strategies that rely on human intuition are often very successfully employed. However, this research concerns an automated style of play that can readily be implemented as computer code, so that the stereotypical view of poker playing that depends in large part on an intuitive understanding of the psychology of the opposition will not be considered. As the strategy is one of maximising profit expectancy, solutions where $\sup(\pi_\Sigma)$ have a positive profit expectancy (page 2, definition 1) are sought.

Consider possible states at the end of the game. At the final stage played

T, there will be a pot size Q_T. The pot will be divided between N_T players. If a single player has the best hand at the conclusion of the game either by all other players folding or by having the best hand resolved in a showdown, then $N_T = 1$. Otherwise $N_T > 1$. The probability of a tie is P_{tie}. The probability that we have the best hand is P_{best}. It is quite common in Texas Hold'em that a tie results from having a board hand that is very strong and on which no players can improve. In that case the pot is divided equally between the participants. The expected return $E[R]$ is therefore given by

$$E[R] = \left(\frac{P_{tie}\ Q_T}{N_T} \right) + (P_{best}\ Q_T)$$

The expected investment is estimated as the expected cost to play out the game. The expected cost to play and expected pot size will depend on current states, some regressors (which are described later) and the action taken (e.g. bet, raise, call or check). The advantage (page 2, definition 1) is calculated for each of the possible actions available and then a decision is made as to which action to take. The criterion used is the course of action that results in the highest profit expectancy. This can have some interesting implications. Even with a fairly weak hand, the pot size can reach a point where further investment should be made because the effective odds, the final pot size compared to required investment, is sufficiently large to warrant such investment. This is sometimes interpreted by other players as an attempt to bluff, but it is no such thing. All that was occurring was profit maximising

behavior.

Poker has traditionally been played face-to-face and many expert players in this form of the game have developed impressive psychological skills to exploit nervous or excited looks from opponents or to use their own appearance to create false impressions about their own state of mind. In addition to this, many techniques for approximating some of the very difficult calculations that need to be undertaken have been developed. In the online environment one cannot see the opponents and so psychological skill is less important, while on the other hand players can arm themselves with all sorts of reference material and computational aids that would never be allowed in a face-to-face game. The automated system discussed here is an extreme example of a computational aid that would never be allowed in a face-to-face game. Even if it were, the information required to make each strategy decision could not be obtained practically.

Several layers of modelling are introduced, which go beyond simply needing to estimate a probability of "winning". It is not usually known what the final pot size will be or how much will have to be invested to play out the game. When fitting a model to estimate the probability of winning, the response variable could be past data and an indicator of whether or not the hand was won. This can be problematic. What about the event where the pot is split between more than one winner? A slightly more subtle problem is the following. To decide whether to continue playing, it is desirable to know whether or not a given hand will be the best hand at the showdown.

Hands folded early in the game may have been the strongest hand at the end of the game. This may be because everyone at the table has a weak hand and is trying to bully other players out of a pot or possibly because a hand had potential to be a strong hand but this was not realised until later in the game. For instance, if a player holds two clubs and then one more club comes out at the Flop, the player may fold their hand. If subsequently two more clubs may come out at the Turn and the River the player could have had the strongest hand at the end of the game. This will be known if the hand goes to a showdown or if the eventual winner reveals their cards. To estimate the probability of having the best hand at the end of the game, the response variable should be whether the hand was the best and not just whether or not the hand won. This is feasible to do in the online scenario although it would be very difficult in a face-to-face game.

1.4 Concluding Comments

The work done in this thesis has applications beyond playing online poker. Interest has been expressed in the methodology by parties engaged in foreign exchange trading and indeed, when one views the system in action, it is easy to see how such applications might provide a natural extension to this research. This research could have applications in any activity where quick decisions need to be made about complex issues in a rapidly changing environment and especially in an online scenario which seems to be ever more

prevalent these days. The methods advocated here tend to require large training data sets.

There are several reasons for using poker as an application of the methods developed in this thesis.

1. It demonstrates real-world evidence that the methods can result in a practical system.

2. Although the analysis of poker is difficult, the required bankroll to actually use the resultant model and demonstrate effectiveness is very modest when compared to other applications such as financial markets.

Chapter 2

Literature Review

You must learn from the mistakes of others. You can't possibly live long enough to make them all yourself— Sam Levenson

Chapter 1 contains a broad introduction to the scope and nature of this thesis as well as some discussion of the approaches adopted. The current section discusses and critiques some of the methods that have been used by other research workers. Some works discussed in this review are from papers that have been released online, but not published in peer reviewed journals. This has been a trend in recent years for many articles, some of which have come to be highly regarded and referenced in future work. This is an inherent problem of Texas Hold'em and gambling generally and makes it necessary to review various online-only articles.

The bulk of previous work in the area of poker built on seminal works of Borel ([4]) and Von Neumann ([27]) to produce Game Theoretic solutions in

([18], [3], [9], [20], [2]). These works and their methods have been referenced, although in this thesis a completely different approach was adopted. In the Game Theoretic approaches, Neural Networks were employed and it was evident that these were "trained" using starting points as recommended by the views of a celebrated professional in ([23], [22]). This has necessitated inclusion of some texts from the popular market. No such initial training took place with the Adaptive Regression methods developed in this thesis as Expectancy Maximisation was always the criterion adopted. However, for the purposes of comparison, referencing of texts from the popular market became inevitable.

The new direction taken in this thesis of an Adaptive Regression process meant that only a few references were available. Regression techniques and Random Parameters estimation are of course not new and referenced works appear in the bibliography, but adaptively tuning the "best" coefficients as more data about the current context comes to light is original and few references exist. One of the original contributions of this thesis is the way in which previous well known principles were put together and for which relevant references are not abundant.

Openly treating a problem as a way to profit by gambling also put this thesis on a less travelled path. Previous poker research such as ([18], [3], [9], [18], [20], [2]) was conducted on a site developed by those researchers where no money was involved and competition was machine vs machine. This was different to the actual on-line experience where most competition was against

humans. It has been observed in areas such as financial markets trading, a reluctance exists to describe the activity as gambling and this leads to researchers describing their work in rather abstract ways. Given that the application was actual on-line play, it seemed absurd to describe the activity as anything other than gambling and this placed the thesis in a category for which very few references exist in mathematical articles. There are of course many articles about gambling, but not many that advocate methods to employ in its pursuit.

Even in the well researched area of Game Theory, some previous work was sourced from areas that are unusual for a doctoral thesis. Indeed, a ground breaking work was ([9]) which was a prize winning paper presented at a seminar but never published and ([2]) which was an internal publication only. The fact that neither of these were published in journals is certainly no criticism and indeed they are both very significant research achievements. Rather, it shows that this field is young and the availability of reference material is not as common as in other fields. This being the case, the key references have been used. Indeed, it could be argued that this thesis could have been built on just ([4], [27], [3]) as references.

These same comments obviously apply to the Literature Review undertaken in this chapter also. There are fewer works referenced than is usual for a doctoral thesis and for the same reasons as described above.

2.1 Emile Borel

The book, *Applications aux Jeux des Hazard* [4], published in 1938, appears to be the first attempt at a mathematical analysis of the problem of poker. The model *La Relance* is proposed in Chapter 5. Each player makes a compulsory bet and they then receive cards randomly and independently from the $[0, 1]$ interval. Player 1 may either make a bet of $B > 0$ or fold. If Player 1 bets, then Player 2 responds by either calling with a bet $B > 0$ or folding. If Player 2 calls then the hands are compared and the higher hand wins. The case of a draw occurs with probability 0. The difference between the Borel and the von Neumann versions ([27] to be discussed below) is that under the Borel rules, if Player 1 does not bet then Player 1 automatically loses the pot. Under the von Neumann scheme, if Player 1 does not bet then the hands are compared and the best hand wins. The von Neumann scheme better approximates "real" poker, but neither is the same as real poker.

Borel finds a unique optimal strategy for Player 2 and all of the (non-unique) optimal strategies for Player 1. The game favours Player 2. Note that under the von Neumann formulation, the advantage resides with Player 1. In a similar way to von Neumann, Borel finds that the optimal strategy is to "bluff" occasionally. For

$$c = \frac{B}{B\ +\ 2}\ ,$$

Player 1 should bluff with probability $c - c^2$. Again, this is different from

our formulation where we estimate the probability of a player folding when confronted by our betting. However, this was an initial exposition performed a long time ago and better solutions have become available [23].

Much of the game theory terminology was used here by Borel before von Neumann [27]. It looks like von Neumann would have acquainted himself with this work prior to embarking on his own analysis and making the connection with larger issues such as economic theory.

As a first analysis this is indeed significant work, but it does appear that this formulation does not lend itself to further useful development. From our point of view in developing a successful system, this work is only of interest in that it was probably the first serious research to be undertaken on this problem. However, it is not obvious why so many of the attempts to develop this field have concentrated on Borel's work rather than that of von Neumann. As an example, the work of Sakaguchi and Sakai [21] concerns itself with relaxing some of the assumptions of *La Relance* which they address by specifying a joint distribution of hands as a Farlie-Gumbel-Morgenstern distribution. They derive some slightly different optimal strategies which is unsurprising, given the slightly different distribution of hands. They use standard game theoretic methods and the results obtained are very specific to the particular game being analysed under the obscure distribution of hands. There is little that applies more generally and the analysis is still only for two player games and so does not extrapolate even to full scale poker. This is after von Neumann established a research path that gives analyses meaning

beyond the confines of these artificial games.

It is now known that the game theoretic approach becomes intractable so fast that it is unlikely ever to be useful for full scale poker, let alone bigger systems such as economies. It is possible that such larger systems will never succumb to having true optimal strategies derived for them. At present for some smaller problems (such as full scale poker) we can derive "near optimal" strategies and these show promise for being scalable to much larger problems. Such near optimal strategies still reveal something of the structure of the system and the interpretation of the resultant model provides qualitative assessments of the agents of the system rather than just quantitative measures. For example, our poker model can identify situations where it is unwise to bluff because at least one active player will call with high probability. Similarly, it can also identify contexts where more bluffing than usual should be engaged in because under some circumstances a different set of players are more likely to fold than the population average. We can therefore see that we are uncovering something about the particular agents involved rather than making population level statements such as an overall bluffing frequency.

2.2 von Neumann and Morgenstern

Although not the first discussion on this topic, the ground-breaking work *Theory of Games and Economic Behavior* [27] by von Neumann and Mor-

genstern established the need to investigate games as a model for human behavior especially in the context of the field of economics. This text paved the way for much of the subsequent development of the field which came to be known as Game Theory. The specific investigation of poker advanced only slightly between this work and the ground breaking work of Billings, Schaeffer, Davidson and Szafron [3], which will be discussed below.

The authors explain that much development in the more mature sciences such as physics occurred because simple problems were considered first and that directly attempting to provide a solution to a universal problem is not usually fruitful. In Section 1.3.2, the authors state their belief that in order to mathematize economics, it is necessary to know as much as possible about the behavior of the individual and about the simplest forms of exchange. They observe that economists frequently point to large "burning" questions and brush aside everything which prevents them from making statements about these, while the experience of more advanced sciences, such as physics, indicates that this impatience merely delays progress, including the treatment of the "burning" question.

This seems to us a very sensible way to initiate research. However, the research can stagnate if only a smaller and smaller class of problems is investigated. Most of the work in poker has been to introduce new hypothetical poker variants that are easier to analyse, and then restrict attention to them. Apart from going some way to investigate the wider class of problems for which this methodology was designed, for a long time (over 50 years),

researchers seemed to be getting progressively further away from being able to extend the class of problems that the method can handle.

Von Neumann and Morgenstern develop much of their theory around the search for optimal solutions. In 1944 they would not have had access to the computational power we have today. They would not have known that even now we cannot reasonably hope to obtain an optimal solution to this problem because of the issue of computational intractability. We concern ourselves with "reasonable" strategies rather than optimal strategies (I use the word "reasonable" instead of "good" because "good" has a specific meaning in game theory). We regard this as a more pragmatic approach. We refer to dominating strategies as those which have a positive expectation against the competitors at hand. If there exists a dominating strategy and we employ it, we will outperform these competitors given sufficient time. An optimal strategy is also a dominating strategy. If we have a dominant strategy that is not optimal and we employ it, then the only difference between it and an optimal strategy is the time taken to obtain any superior position over any competition. Given sufficient time and resources of our own, the dominant non-optimal strategy will still inherit all the resources of the competitors.

Because these strategic considerations need to be applied to larger problems (such as poker games extrapolated to economics) the issue of computational intractability may mean that a dominant strategy is the best one can hope to obtain in a practical setting.

Von Neumann and Morgenstern begin to address poker specifically in

chapter 19. They reduce the game to a simpler version as follows:

1. There are only two players.

2. There are two betsize possibilities a, b such that $0 < b < a$.

3. Each player is dealt a "hand". This is a draw from one of the S possibilities with each hand having the same probability. This seems to imply sampling with replacement, which obviously is not the same as real poker. Von Neumann's hand generation is not a uniform $[0, 1]$ draw because a true uniform draw generates a tie with probability 0, while Von Neumann explicitly caters for the possibility of a tie. Von Neumann talks about the continuous case which is uniform on $[0, 1]$, but this is only to enhance some explanations and avoid combinatorial algebra where it is not necessary. After he has made his point he reverts back to the discrete case.

4. Each player makes a secret bet, either a or b.

If both players bid "high" or both "low" the hands are compared and the stronger hand receives the profit of a or b respectively. If the hands are equal, then no payment is made. If one player bids "high" and the other "low" then the player with the low bid can "Pass" or "See". "Passing" means paying the opponent the amount of the "low" bid (irrespective of their hands). "Seeing" means that the "low" bid is increased to the "high" bid and the situation is treated as if both players had bid "high" in the first place. Thus, there is a

restriction on the number of bets that can be played as well as the obvious restriction to the number of players. This version of the game is similar to that proposed by Borel with the important difference that in the Borel version, player 1 can either bet or fold. In the von Neumann version, player 1 can elect to "See" rather than just fold. Von Neumann then calculates the optimal strategies. When reading his analysis one is immediately struck by his clarity and perception. While von Neumann had tamed many more difficult problems than this, we found ourselves in awe of how his analysis proceeded even though this was a completely new field. In 1944 von Neumann was working on the Manhattan project, so it is remarkable that he was able to find time for anything else, let alone something ground-breaking such as this work, which even today is still one of the best developments in game theory ever written. Obviously, this game differs from actual poker and even more so from the Texas Hold'em variety that we investigate. One must bear in mind though, that a simple example was exactly what was called for here. Von Neumann was more concerned with developing a new mathematical discipline than a winning poker model and a full version of the game would, by its complexity, have obscured the new methods of analysis that von Neumann was revealing.

An essential concept in our approach is that of an *advantage bet*. This really just means a bet that has a positive expectancy. Von Neumann (and all later game theory discussions) make no reference to this, because it is built into the method. By only betting when there is a positive pay-off, the

game theory method advocates folding in situations where the player is at a disadvantage.

In von Neumann's version of the game, Player 1 has a unique optimal strategy. Player 2 has multiple optimal strategies and von Neumann finds them all. The game favours Player 1. Von Neumann, from the properties of his optimal result identifies situations where Player 1 should behave counter-intuitively in the sense that he should sometimes bet when he has a poor hand. This of course is what poker players mean by "bluffing". In the Borel version, a player should bluff when he has a moderate hand. In the von Neumann version, players only bluff with their worst hands. This makes the situation more similar to actual poker, but our model advocates different behavior in the context of Texas Hold'em. We estimate group level parameters for the remaining players in the game for a regression on the probabilities that given players will fold subject to our bet based on their previous behavior. We then calculate our expectancy for our outlay given this information and if it is above some threshold, we bet. Otherwise we fold. It only rarely occurs that our model will advocate bluffing against numerous opponents as there is usually too much chance of at least one player "calling" us. This is an important difference between our approach and game theory. We are much more reliant on modelling the behavior of the competition. Against some players, our algorithm will bluff often, against other players, our algorithm will never bluff. There is also another interesting aspect to this that does not become apparent in the von Neumann or Borel games

because of their simplifications. In full scale Texas Hold'em, we can get a hand that has no current value, but has a probability of improvement that is out of proportion to the effective odds from the pot and the expected final pot size. This can mean that even though we have a weak hand now, our expectation is still positive. So, we bet. If our hand is not developed because of bad community cards, it can frequently happen that the rest of the game can be played out at no cost. When this happens, the remaining players often assume that we have bluffed with our original bet. This is not what happened. We actually took the odds of getting good cards but that event did not occur. In our view this is different to bluffing, although it is often perceived as such by some players.

While von Neumann only really devotes one chapter specifically to poker (chapter 19), it is a very important analysis because it provides an excellent example of how his new techniques, which came to be known as game theory, can be applied to an actual game. His work was not significantly improved until the work of Billings, Schaeffer, Davidson and Szafron well over 50 years later.

2.3 Dennis Papp

Dennis Papp's thesis *Dealing with imperfect information in poker* [18] describes the author's system he calls *Loki,* possibly named after the Norse god of luck. The full text is available for download through Professor Jonathon

Schaeffer's homepage at the University of Alberta[1]. The thesis, which utilises a Game Theory approach, provides a basis for extensive developments by other researchers from the same university and amounts to a careful automation of the texts by David Sklansky [23],[15] ; human behavior is modelled so that the style of play is a close approximation to that advocated by Sklansky.

The author is a part of the research community at the University of Alberta that has conducted extensive and ground breaking research into poker using Game Theory methods. The approach is quite different from the one adopted in this thesis.

Loki uses a *weight array* for opponent modelling. This is an array of numbers, one for each possible two card combination hand that represents the conditional probability that the player would have played in the observed manner if they had that hand. Generic opponent modelling basically amounts to having a fixed set of weights and specific opponent models result from updating the weight table for the opponents. *Loki* retains frequencies for a number of categories such as the number of active opponents, the number of raises in the round, bets to call and the game round (e.g. pre-flop, turn). This method of developing a Specific Opponents model is different from the method adopted in this thesis namely to progressively relax the assumptions of the Generic Opponents model.

[1] http://webdocs.cs.ualberta.ca/~games/poker/publications/papp.msc.pdf
Website last accessed 1 Sept 2011.

In section 6.7 Papp critiques his own work. He raises the point that calculating expected values (perhaps using simulation) may be preferable to the expert-dependant strategy he uses and he proposes introducing ways of check-raising and slow-playing that are very similar to very early versions of the work described in this thesis, but which have subsequently been greatly refined. In later work, including Billings *et al* ([3], to be discussed below), some of Papp's ideas have been implemented, but in a way dependent on expert strategy.

In chapter 7, Papp discusses in some detail how *Loki's* opponent modelling works. For a specific opponent model, the heart of the method for *Loki* is the reweighting mechanism. If no reweighting takes place, then the model does not change for specific opponents and is thus a generic opponent model.

In section 7.2.3 the three post-flop rounds are considered. *Loki* calculates an "out" count in a similar way that a human player would. This will result in calculated probabilities of winning the hand that get more similar to ours as the game progresses, but could be quite different for the earlier rounds. Calculation of an "out" count is as recommended by Sklansky. The method advocated in this thesis is to use simulation results as predictors in a regression function to estimate conditional probabilities. The use of "out" counts was developed by human experts as a technique for rough mental calculation of probabilities of interest. Usage of this system almost guarantees a human-like choice of strategy because the probability calculations are the same (albeit less error prone by computer), which is not necessarily a bad

thing if a high quality poker playing system is required, but could conceivably make it more difficult for the computer system to uncover emergent phenomena about the game. That is, phenomena arising from the strategy engine without being specifically programmed by the researcher.

2.4 Billings et al

The first contribution discussed here, entitled *The challenge of poker* [3] focuses on Texas Hold'em poker and resulted in *Poki*, a highly regarded poker robot. *Poki* was used on the virtual online site maintained by the authors (Internet Relay Chat). Games are not for money, but there is sufficient pride at stake to ensure high standards of play. The authors use a mixture of game theoretic results to base strategy decisions on enumerations performed in the current context of the game. The focus of their work is on artificial intelligence (AI) which is different to von Neumann's reasons for studying this problem.

In their article, Billings *et al* [3] contrast poker, a game of imperfect information, with games like checkers and chess and explain why techniques that have been successful in perfect information games do not fare so well with games like poker.

The authors discuss pure game theory approaches, simulation approaches and enumeration approaches. Game theory is infeasible for full scale poker, at least for multiplayer games. As assumptions are relaxed, the game tree

gets larger at an exponential rate. Simulation methods play out a hand many times and use some statistics from the generated sample. Enumeration approaches consider each possible opponent hand to exactly calculate probabilities. Typically the enumerations are only for a limited look ahead and in this application tend to be used to estimate probabilities that the system currently has the best hand and what some of the more likely hand improvements may be. This approach uses results from game theory and will be referred to as *Game Theory / Enumeration*. They do use some simulation results, but these are situations that often occur and are simulated outside the main model which just uses the results of those simulations. Billings *et al* mimic human players in that they seem very concerned with whether or not they are "in front" . The enumeration approach advocated by Billings et al contrasts the simulation approach advocated in this thesis where simulations are made to the end of the game and then regression techniques are used to translate the results of various simulations to the probability of an event and whether that probability represents a positive expectancy at the conclusion of the game. Using an enumeration technique for such an analysis rapidly become intractable.

Poki is an improvement on the *Loki* system already discussed in [18]. The main difference between *Poki* and *Loki* is *Poki's* improved specific opponent modelling. Papp in his thesis [18] suggests several improvements to his own work, most of which have not been adopted in *Poki*. *Poki* seeks to predict probabilities of opponents behaving in certain ways in the current context of

the game.

The paper *Approximating Game-Theoretic Strategies for Full-scale Poker* [2] outlines the techniques used by the research group at University of Alberta to address issues confronting the Game Theoretic analysis of poker and focuses primarily on reduction of the Game Tree. The authors provide the result that for 2-player Texas Hold'em, the Game Tree has size $O(10^{18})$ and they show how various simplifications to the game can result in a Game Tree of size $O(10^7)$ while retaining the fundamental structure of the game. After some introductions to Game Theory and poker, they begin discussing various methods of simplifying the game by reducing the number of game states. These techniques are collectively known as *abstractions.* The authors discuss betting round reduction whereby players are allocated a maximum of 3 rather than 4 betting rounds per stage. They then investigate elimination of betting rounds completely, so for instance a post-flop model can be employed that ignores the distinction with the pre-flop round. Possibly the most important abstraction technique is *bucketing,* where sets of possible hands are partitioned into equivalence classes. There are a number of ways this can be done ranging from quite crude buckets such as equivalent hand strengths to others that cater for the potential of hand strengths. It is not completely clear which of the bucketing abstractions are data driven and which depend on an external expert judgement as to which hands are equivalent especially when the potential for further development of a hand is considered. This is not a problem as expert judgements can be incorpo-

rated into an analysis. In this thesis however only data driven mechanisms are considered. Various software implementations are considered and tested against each other. The software implementations considered do seem to bear out the claim that abstractions can be considered that do substantially reduce the size of the Game Tree, while retaining essential characteristics of the game. The most successful implementation was tested against a very high quality human opponent, Guatam Rao (aka "The Count"). After 7,000 hands, the human player won overall in a 2 player game against the robot. The authors point this out and propose various mechanisms by which their algorithm can be improved. The variance in the results and especially the fact that the robot was winning after 4,000 hands can also be interpreted as a possibly inconclusive result in that further trials may result in a better outcome (for the robot). The program was a Generic Opponents model, so the strategy engine was not adaptive. Once some sort of specific opponents model is implemented, it is conceivable that the program could be developed to defeat even such world class opposition. The most important result though was not the play, but the development and outline of the various methods of abstraction. While the research is of only minor relevance to the work conducted in this thesis, which is not concerned with Game Theory, it is still a very important document in the field of poker research because most poker research is conducted from a Game Theory perspective.

2.5 Nicolas Abou Risk

This research, entitled *Using Counterfactual Regret Minimisation to create a Competitive Multiplayer Poker Agent* [20], from the Poker Research Group at the University of Alberta builds on past research by that group using a Game Theory approach. A central part of the Game Theory development is the search for Nash Equilibria. This is discussed in section 2.1.3. Issues whereby the strong results for Nash Equilibria for 2 player games not applying universally to multiplayer games are discussed as is the ever present (in Game Theory) problem of computational intractability. The proposal in the thesis is to adopt the weaker requirements of ϵ-Nash equilibria. The author then introduces the notion of Counterfactual Regret Minimization (CFR) and cites previous results that show that strategy profiles that satisfy this condition lead to ϵ-Nash equilibria. This is a very elegant result and enables usage of a calculus of Game Theory that is guaranteed to provide strategy profiles with the properties required. As with all the Game Theory approaches known, computational intractability is an issue and requires simplification by abstraction of the game in a manner similar to [18].

The author starts with an analysis of 2-player or "heads up" poker. The resultant Game Tree is manageable and the author notes that many multiplayer games reduce to heads-up games due to players folding out. The different approach used in this thesis makes use of some properties of the game from players folding, an obvious example being the number of players

who have folded. So, the modelling in this thesis would suggest that games where all but 2 players have folded out are not the same as games with only 2 players because the number of players who have folded out contains information that can be used. The author uses the 2-player game as a platform for further development because the much smaller Game Tree in the 2-player case requires less abstraction to achieve computational tractability.

Computational intractablity is not completely resolved by CFR and some simplification of the game is still required. The methods used which essentially group similar sorts of contexts together. These methods are outlined in section 2.2 and relate to such things as grouping hands irrespective of suit, reducing betting rounds and so-called bucketing whereby hands of similar strength are regarded as equivalent.

In chapter 3, the author extends the methodology to 3-player games and shows the resultant system outperformed *Poki* [3]. This is a significant achievement.

In chapter 4, the CFR motivated robot is compared with strategies recommended by various world-class experts. Tests of this nature can be a bit misleading because often human opponents will not be able to specify in advance exactly how they will play in any given situation. However, a test against a fixed strategy can be illuminating and of course if a simple fixed strategy cannot be comprehensively defeated it can be indicative of a problem with the model.

The main innovation with this author's work is the use of CFR. This is

a significant contribution to the Game Theoretical approach. The concept of ϵ-Nash equilibria represents an approximation of sorts in that as $\epsilon \to 0$, the Nash Equilibrium is approached, however there is no guarantee that the optimum behaviour is "close" to Nash Equilibrium behaviour for an even small value of $\epsilon > 0$. This of course is an inherent issue (possibly of no consequence) and is not a criticism of this author's work.

Chapter 3

Generic Opponents Model

I pushed and pushed. I just kept pushing. I made every mistake there was, but I just kept pushing—Rene Descartes.

The reader is reminded of the following definition:

Definition 1 *A Generic Opponents Model is a model that assumes homogeneity among the population of sets of opponents.*

Generalised Linear Models were employed for the purpose of implementing a Generic Opponents Model. Such models are very powerful and are often hard to surpass in terms of predictive power. They are easy to estimate and understand. There are a large number of texts on Generalised Linear Models; some excellent resources are [14] for an in-depth discussion, [7] for a more introductory treatment and [12] for a book length treatment of Binary Logistic Regression . The Generic Opponents Model was the initial attempt made

in this research to model opponents' play and estimate profit expectancy. The models developed in this chapter represent the rationale around which the system is built. The more sophisticated models introduced later use the same formulation, but with different methods of obtaining expected win rates subject to actions taken.

3.1 Models implemented

The details of the individual models estimated as well as descriptions of predictors used appear in the Appendices (page 127 for predictor descriptions and page 131 for regression output). For the sake of clarity, the models presented in this thesis are not the final versions used. They are earlier and smaller models. This is because the later models, having more predictors, were much more complicated and would have required lengthy explanations without adding significantly to the salient features of the process. The main features of the process are unchanged by using the smaller models for explanatory purposes.

The following terminology will be used throughout the thesis:

1. P_WIN: The probability of being the sole winner of the game, irrespective of hand strength. P_WIN is a measure of the probability of eventually winning the hand without folding out, either by being the sole survivor or having the best hand. In the training data set, being left as the only surviving player allows P_WIN to be recorded as a

success. Often the sole survivor will reveal their hand even though they are not required to, possibly to establish a psychological advantage over the opposition. Folding but having a superior hand to the winner, will also record P_WIN as a success. It is assumed that the other players' actions would have been unchanged by the decision to not fold and the hand would eventually have been won. This variable is estimated directly from a Binary Logistic Regression.

2. P_TIE: The probability of a tie if the robot plays the game out fully. P_TIE is a measure of the probability that more than one player will share the winning hand. This can occur when the board cards represent a strong hand that no player can improve on, but also those remaining in the game cannot lose. In this case, the pot is divided between the surviving players. The conservative assumption that no further players will fold in the eventuality of a tie and that therefore each player's return is the final pot divided by the number of players at the time of the decision is adopted. This can of course mean that some hands are folded when the actual dividend may be a lot higher because other players may fold, but the conservative assumption is adopted. The policy that it is better to miss some advantage bets than to take more disadvantage bets is adopted. This variable is estimated directly from a Binary Logistic Regression.

3. E_POT: The expected pot size at the end of the game. This is a

calculated quantity, detailed below.

4. E_COST: The expected cost to play out the rest of the game. This is a calculated quantity, detailed below.

Many formulations of on-line poker seek to mathematically implement human methods of play. They typically try to estimate which player is "in front" at a given stage and to determine how likely this situation is to remain the case. In contrast, the methods developed in this thesis attempt to estimate the probability of having the best hand at the end of the game, the normalised pot size at the end of the game, the cost to play out the remainder of the game and to seek advantage bets (page 3, definition 7).

In all the models, different coefficients and predictors were employed for each game stage. Attempts to amalgamate game stages resulted in inferior out-of-sample tests. Breaking the data down this way was a luxury afforded by having such large quantities of data with which to work.

Every time a strategy decision was required, some quantities are estimated via regression functions:

1. POT_SIZE_DELTA: The amount that the pot will grow in units of table minimum bets. This is estimated via an Ordinary Linear Regression.

2. NORM_COST_TO_PLAY. The cost to play out the hand to conclusion in units of table minimum bets. . This is estimated via an Ordinary

Linear Regression. Thus

$$\text{E\_POT} = \text{Current Pot Size} + \text{Action Cost} + (\text{POT\_SIZE\_DELTA} \times \text{Table minimum})$$

$$\text{E\_COST} = \text{Action Cost} + (\text{NORM\_COST\_TO\_PLAY} \times \text{Table minimum})$$

Action Cost is the cost of the strategy decision being contemplated, the cost to bet or cost to raise or cost to call depending on the action chosen. Other costs such as cost to check and fold are zero. The Action Cost is provided by the on-line site.

P_WIN and P_TIE are estimated directly from the regression functions. Subject to each possible strategy decision, a dividend is estimated. For action α, the dividend is

$$D_\alpha = \frac{\text{E\_POT}}{\text{E\_COST}},$$

where E_POT and E_COST are both calculated subject to the action α which may be a bet, raise etc. The convention adopted is that for a fold, the dividend is zero and obviously the Action Cost for a fold is zero also. The dividend is received in the event of a win with probability estimated by P_WIN. A share of the dividend, conservatively estimated as shared equally among all the active players (OPPONENT_COUNT + 1) is received with probability estimated by P_TIE. Otherwise, the return is zero. The estimate of advantage (page 2, definition 1) is therefore

$$A_\alpha = (\text{P\_WIN} + \frac{\text{P\_TIE}}{\text{OPPONENT\_COUNT} + 1})\, D_\alpha$$

Recall in the definitions the profit expectation on a π-game (page 2, definition 5) is

$$\pi_\Sigma = \sum_{n=1}^{N} (p_n D_n - 1) I_n$$

In this case for a proposed action α the Expectation Maximiser $\sup(\pi_\Sigma)$ (page 3, definition 6) is the action α that results in the highest value of $\pi_\Sigma|\alpha = (A_\alpha - 1) \times \text{E\_COST}$. There are only a small number of possible alternative decisions and so $\sup(\pi_\Sigma)$ can be found quite quickly and easily by just trying all the alternatives. In the event that the Expectation Maximiser is not unique, the lowest cost alternative was chosen. This was motivated by nothing more than a desire to choose the most conservative option. It happens only very rarely, so makes little difference overall.

3.2 Model Estimation

The output for the model is displayed in Appendix B, page 131, with a description of predictors shown in Appendix A, page 127. In order to estimate the more complex models presented later, custom model estimation software was written. A first step was to estimate the models presented in this section which are nothing more than generalised linear models (GLMs). Commer-

cial software could have been used for this, but then custom software would have been required anyway for the models Context Specific models needed later. All the results for the GLMs estimated in this section were checked against commercial software, usually Arc, LIMDEP, or SPSS. No significant differences were found. The BFGS (Broyden-Fletcher-Goldfarb-Shannon) algorithm was used exclusively exclusively. A detailed discussion of BFGS and similar routines can be found in [19]. BFGS is one of the so-called *variable metric minimisation routines* (also called *quasi-Newton methods)* which construct an approximation to the Hessian (called the *Arc Hessian)* iteratively rather than needing a specific formulation. BFGS was appealing because the second-derivative matrix for models discussed later can be very expensive to calculate and avoiding the need to calculate it at every iteration resulted in a more efficient estimator in terms of the computational effort required. Texts such as *Generalised Latent Variable Modelling* [24] point out that a problem with quasi-Newton methods is that the Arc-Hessian may not be a good approximation of the actual Hessian and therefore that even though the optimisation result is valid, the Arc-Hessian should not be used for making statements about, for example, the significance of predictors. BFGS can still be employed and at convergence the Hessian can be calculated.

3.3 Concluding remarks

When the model was implemented it was found that when started against unfamiliar players, it did better than against familiar players as can be seen from the graphical output in Appendix C, page 141. This is indicative that the style of play was predictable and that the opponents were exploiting this. Typically one or two opponents would bet and raise very aggressively early in the game knowing that my robot would fold unless it had a very strong hand. Consequently, at the Pre-Flop and Post-Flop, opponents would play very aggressively and the robot would usually fold as a result. If it still hadn't folded by the time the Turn card came out, all the remaining players would fold. Thus, the robot was always bullied out of pots when it did not have an initial strong hand and not played against when it did have a strong hand. Attempts were made to make the robot "braver" against repeated bets, but it still played very predictably and opponents simply exploited this new, but still predictable, play in different ways. Some human players have a remarkable ability to spot any element of predictability in very short periods of time.

To see the nature of this problem in a different context, consider the game "Paper, Rock, Scissors". Simple games like this yield to game theoretic analysis. It can be shown that the optimal strategy for playing this game is to randomly choose between the alternatives with equal probability. Getting a computer to display truly random behavior is extremely difficult.

If we naively generate equiprobable (often confused with random) choices by simply cycling through the alternatives, then our play would be predictable and after a few games most astute human players would have no trouble defeating us unless we changed our strategy. A more complicated version of this phenomenon is what happens when poker is played with a fixed strategy. Astute human players realise very quickly that the strategy is predictable and exploit this weakness. It is partly because of this that most attempts to play on-line poker with an automated system result in long term losses. It is very easy to convince oneself that a model fitted on past data would be profitable and then find that shortly after using this model, some of the opponents change the nature of their play to the detriment of the model that looked so promising.

Some success was observed by introducing a random component into the play of the suite of programs called THOR, from **T**exas **Ho**ldem **R**obot. With a strong hand, instead of always raising, raise with probability p that could either be pre-determined or could possibly depend on advantage and check or call with probability $(1 - p)$ to stay in the game at minimum cost. Simply adopting this can have the effect of advanced plays such as:

- "Check/raising" where one first checks with a strong hand to embolden other players to bet and then raises after they have committed an investment

- "Slow playing" where one plays conservatively with a strong hand.

This made play a lot less predictable, but good players were still able to sense the correlation between the robot betting and the robot having a good hand. The variations to play, being random, did not satisfy the $\sup(\pi_\Sigma)$ criterion (page 3, definition 6), that is, they did not maximise profit expectancy. Betting with a high advantage was not the same as betting with a very strong hand. Sometimes the robot would bet with a weaker hand if the implied dividend from the pot size and cost to raise or cost to call represented an advantage bet. With the specific opponents model introduced later, the process for determining whether to bet or check/call is so complicated, that it appears to other opponents to be completely random. However, the actual system is quite deterministic and indeed is calculated to maximise expectancy, so departures from set plays are not random but are actually $\sup(\pi_\Sigma)$ strategies.

It can be seen that the course being adopted here is very different from that advocated by Game Theorists such as [18], [3], [21], [20]. Game Theory is not being employed. No attempt is made to determine which player is "in front" at any stage. The decision process seeks to estimate an advantage for investing further into the game. Plays that may be interpreted by the opposition as, say, bluffing are not embarked on for the reasons assumed of a typical player. Sometimes calls and bets will be made when holding weaker hands, but this is because the estimated dividend for investment constitutes an advantage bet. At this stage, this philosophy reduces to always playing expected pot-odds [23] which is admittedly naive, but this behaviour will

be seen to be enriched by the more realistic models employed later. This highlights the difference in philosophy of trying to approximately solve the exact problem as advocated here , rather than trying to exactly solve an approximating problem as most previous research into this area has sought to do ([18], [3], [21], [20], [2] to name a few).

Chapter 4

Random Parameters Model

A short jump is certainly easier than a long one, but no one wanting to get across a wide ditch would begin by jumping half way—Carl von Clausewitz

It was evident the Generic Opponents model had shortcomings. Part of the reason for this appears to emanate from variation in player styles. Because a Generic Opponents model always plays the same way, an experienced human player can probe for weaknesses caused by this predictability. Clearly what is required is a Specific Opponents Model (page 4, definition 14) at a minimum. Within the framework of this research the first step is to obtain a Random Parameters Model. Recall the following definition:

Definition 1 *A Random Parameters Model is a model that has parameters that may vary across a possibly heterogeneous population.*

This type of model is described in [24]. This modelling technique is very popular in the Econometrics community and an excellent discussion can be found in **[26]** Chapter 6**.** While the setting is a little different, the concepts employed are almost the same.

4.1 Background

People respond differently to a given set of circumstances. A regression estimating some behavioral characteristic performed on a large group of individuals often estimates the item of interest only for a typical member of the group. Most regression-type analyses fall into this category; estimated model parameters are implicitly assumed to apply uniformly across the entire population.

In estimating the probability of a shopper purchasing some item using possibly a Binary Logistic Regression model [12], one might expect that price might plays a significant role for most members of the population and that price would have a negative regression coefficient, so as price increases the probability of purchase would go down. However, some members of the population might have a less negative or possibly zero coefficient, meaning that price would not be an overriding concern for them and that other factors may play a larger role in their case. For other members of the population, price might actually have a positive coefficient. For example higher price may be interpreted as higher quality. While it may be true that for most members

of the population, price would have a negative coefficient, a more realistic specification is that the coefficient (or preference) for price is distributed among the population with a mean that is negative but with enough variance to account for discrepancies in behavior. Similar considerations apply to poker. Take for example the number of bet/raises made at the current stage. It is reasonable to expect that the number of bet/raises regressor would have a negative coefficient for the P_WIN model (page 46), in that our confidence of having a winning hand may decrease as the number of opponent bets and raises increases. The logic here is that more opponent bet/raises would indicate stronger opponent hands and therefore our hand may not be as strong as thought, so P_WIN decreases. However, some opponents may bluff with uncommonly high frequency and also slow play strong hands with high frequency. It can be costly if we fail to identify opponents who may have a zero or possibly even a positive coefficient (or behavioral preference) for active stage bet/raises. If instead a model is specified where the coefficients vary according to some probability distribution and then the parameters of that distribution are estimated, more accurate estimates of behavioral characteristics can be obtained.

The basic model formulation described in Chapter 3 will be extended to account for variation in the style of play between opponents. This different type of regression will form the basis for the very powerful model to be introduced in Chapter 5. An essential ingredient of long term survival in the online poker environment is the ability to adapt to different styles of

play. This will eventually be developed to a specific *context* model rather than a specific *opponents* model because changes in playing style are required for reasons other than opponents' attitudes. The model sought is one that adapts to various contexts, not just the playing style of the opponents, although opponent style forms a significant part of what is referred to as the game context. Indicators of opponents' styles to categorise them as "aggressive" or "loose" or any other pre-defined category will not be sought as they are in, for example [3]. This is a difference between this thesis and systems such as Poki [3], [9]. Soft boundary classifications that are defined in some data-driven way by the effect that a particular context has on estimates of various quantities such as win probability and pot-size will be estimated. Then, an adaptive system could be one that identifies where on the probability distribution of playing styles a given game context is. This will form a very loosely defined group and that group will be characterised by a best fitting set of coefficients. Thus, the robot will play differently in different circumstances without trying to apply any sort of hard boundary group. Facilitation will be sought to identify contexts where different coefficients satisfy $\sup(\pi_\Sigma)$ (page 3, definition 6). The resultant categories will vary continuously rather that putting players in discrete groups. This continuous spectrum is a *fuzzy* group. Using this concept, abstract groups based on the data can be assigned for which no real *heuristic* explanation exists. The context categorisation is data driven and not defined beforehand. Many classifications of opponents will be significant but not easy to

interpret.

4.2 Population level model

Within the population, the model parameters will be assumed to have some probability distribution f, the *mixing distribution*. The parameters of the mixing distribution are θ and the modelling task becomes estimation of θ. We start by assuming a multivariate normal mix, that is that the coefficients are multivariate normal distributed, and we seek to estimate the population level mean vector and covariance matrix of this distribution. The formulation of the Generic Opponents P_WIN model is Binary Logistic and thus

$$P(Y_n = 1|x_n) = \frac{\exp(x_n^T\beta)}{1 \ + \ \exp(x_n^T\beta)}.$$

If a mixing distribution f is assumed, estimates of parameters of that distribution θ are sought. So the probability becomes

$$P(Y_n = 1|x_n) = \int_{\beta} \frac{\exp(x_n^T\beta)}{1 \ + \ \exp(x_n^T\beta)} \ f\ (\beta|\theta)\ \partial\beta \tag{4.2.1}$$

There will usually be no closed form solution for this expression, but simulation methods can be used to approximate to any desired level of accuracy. Choose a sufficiently large R and generate R standard normal vectors n_r for $r = 1, \ldots, R$, that is, each is a k dimensional vector with each component being a random draw from a standard normal distribution having a mean of

zero and a variance of 1. Many books on Numerical Analysis show numerous ways to achieve this, we used "Numerical Recipes in C" [19]. The estimated value of θ provides us with the mean vector μ and the covariance matrix W. We operate with the Cholesky decomposition L of W, with $LL^T = W$. Again many numerical analysis texts provide details on how to calculate L. We used the definition of the Cholesky decomposition to calculate this explicitly, but good algorithms can be found in [19]. We generate R draws from a Multivariate Normal distribution with mean μ and covariance W, this is $\beta_r = \mu + Ln_r$ where $W = LL^T$.

Converting to a simulation approximation as described in equation (4.2.1) above, we obtain

$$P(Y_n = 1|x_n) \approx \frac{1}{R}\sum_{r=1}^{R}\frac{\exp(x_n^T\beta_r)}{1 + \exp(x_n^T\beta_r)}. \tag{4.2.2}$$

Expanding this equation to reveal the explicit form of the simulation gives

$$P(Y_n = 1|x_n) \approx \frac{1}{R}\sum_{r=1}^{R}\frac{\exp(x_n^T\,\mu + x_n^T\,Ln_r)}{1 + \exp(x_n^T\,\mu + x_n^T\,Ln_r)}. \tag{4.2.3}$$

This is the simulation estimate of the required probability for the nth case. To obtain the likelihood, multiply out all N cases, or as standard practice to avoid floating point underflow dictates, sum the logarithm of each of the N

cases to provide the log-likelihood given by

$$\begin{aligned} G(\mu, L) &= -n\log(R) + \sum_{n=1}^{N} \log\left[\sum_{r=1}^{R} \frac{\exp(x_n^T \beta_r)}{1 + \exp(x_n^T \beta_r)}\right] \\ &= -n\log(R) + \sum_{n=1}^{N} \log\left[\sum_{r=1}^{R} \frac{\exp(x_n^T \mu + x_n^T Ln_r)}{1 + \exp(x_n^T \mu + x_n^T Ln_r)}\right]. \end{aligned}$$

Derivatives are required with respect to each component of μ and also each component of the lower diagonal of L. All components of L above the main diagonal are zero. Firstly for the components of μ, consider a typical representative $[\mu]_a$, which is the ath component of the vector μ. For a typical component a of μ

$$\left[\frac{\partial G}{\partial \mu}\right]_a = \left[\frac{\partial}{\partial \mu}\right]_a \sum_{n=1}^{N} \left[\log\left[\sum_{r=1}^{R} \frac{\exp(x_n^T \mu + x_n^T Ln_r)}{1 + \exp(x_n^T \mu + x_n^T Ln_r)}\right]\right],$$

where

$$\left[\frac{\partial}{\partial \mu}\right]_a f(\mu)$$

is the derivative of f with respect to the ath component of the vector μ. The N situations relate to every decision made, so if we are called upon to make, say two decisions at the Pre-Flop, one decision at the Post Flop, two decisions at the Turn and one at the River, then in that game we have made six decisions which will each be included as a data row. Consequently, the N decisions considerably outnumber the number of games played. The k

components of the gradient will then be the sum over N data rows for each component, here a typical one, a, is considered. Proceeding with the algebra, we get

$$\left[\frac{\partial G}{\partial \mu}\right]_a = \sum_{n=1}^{N} \left[\frac{\sum_{r=1}^{R} \frac{[x_n]_a}{1 + \exp(x_n^T \beta_r)} \cdot \frac{\exp(x_n^T \beta_r)}{1 + \exp(x_n^T \beta_r)}}{\sum_{r=1}^{R} \frac{\exp(x_n^T \beta_r)}{1 + \exp(x_n^T \beta_r)}} \right],$$

where $[x_n]_a$ is the ath component of the nth data row. This can be expressed as

$$\left[\frac{\partial G}{\partial \mu}\right]_a = \sum_{n=1}^{N} [x_n]_a \left[\frac{\sum_{r=1}^{R} \frac{Q_{r,n}}{T_{r,n}}}{\sum_{r=1}^{R} Q_{r,n}} \right],$$

where $Z_{r,n} = \exp(x_n^T \beta_r)$, $T_{r,n} = 1 + Z_{r,n}$ and $Q_{r,n} = Z_{r,n}/T_{r,n}$ This notation, aside from looking neater, shows how the expression was implemented in computer code and also introduces some terms that will be used next when the derivative with respect to the components of L are obtained.

The derivatives with respect to each component of the lower triangular matrix L are also required. All components of this matrix above the main diagonal are zero and so do not affect any of the calculations. For a k dimensional mean vector there will be $k(k-1)/2$ components below the main diagonal and k components on the main diagonal giving a total of $k(k+1)/2$ possibly distinct components of the Cholesky decomposition matrix L. Consider a typical representative of this matrix, the b, c element where $b \leq c$,

being part of the lower diagonal of the matrix L.

A straightforward but lengthy calculation leads to

$$\left[\frac{\partial G}{\partial L}\right]_{b,c} = \sum_{n=1}^{N} [x_n]_b \frac{\sum_{r=1}^{R} \frac{[n_r]\exp(x_n^T\mu + x_n^T L n_r)}{(1 + \exp(x_n^T\mu + x_n^T L n_r))^2}}{\sum_{r=1}^{R} \frac{\exp(x_n^T\mu + x_n^T L n_r)}{1 + \exp(x_n^T\mu + x_n^T L n_r)}}.$$

Expressing all the parameters of the likelihood function as a packed array of terms, the first N being for the mean vector and after that for the elements of the covariance matrix, an expression for the gradient has now been obtained. This was used as the gradient function required by the BFGS optimisation routine. After optimisation is complete, the packed array of terms can then be unpacked into the components of μ and W.

4.2.1 Implementation

All parameters were estimated simultaneously. A packed array of terms was constructed. If an intercept term is fitted, then the first element of each data tensor x_n is padded with 1 as the first element and then

$$k = \text{ number of regressors fitted} + 1,$$

otherwise

$$k = \text{number of regressors fitted.}$$

By default the process was started from the Generic Opponents Model estimate. So, in the case of the P_WIN model, the Logistic Regression ([12]) coefficients were used as the initial value for the mean μ. Some experimentation to find a good starting value for L was conducted. Among the candidates tried for starting values for L, were the Cholesky decomposition of the Hessian matrix for the Generic Opponents model and various, sometimes complicated, functions of the first derivative and the data. However, simply using an identity matrix consistently provided a good starting value for L. Convergence and converged values always seemed stable and as computationally fast as any other starting values used.

Experimentation with other various ways to produce the R vectors n_r was conducted. So called *low discrepancy sequences* are described by Neiderreiter [16], [17] where they are referred to as *(t,m,s) nets*. This does indeed increase computational efficiency. Very good convergence properties with $R = 2000$ uniform draws and about the same with 200 low discrepancy draws. The extra overhead in generating the low discrepancy draws is not significant as it occurs only once at the start. Uniform draws were still used in spite of this, because it was discovered that a few models among the thousands estimated converged to nonsensical values. It is unclear where the problem was, but it must be said it is unlikely that the problem was with the *(t,m,s) net* methodology. A decision was made to persevere with uniform draws as no such issues were evident and the difference in estimation time was not huge. The same problematic model runs using uniform draws had no convergence

issues. Rather than embarking on a possibly very long debugging run, the decision was made to use the much simpler uniform draw methodology. The fact that uniform draws can be easier to work with in terms of software implementation increases their attractiveness.

4.2.2 Remarks on simulation methods

The issue of simulation bias caused by the logarithmic transform depends on the relationship between the number of simulation draws R and the sample size N. If R is fixed then the bias problem prevents convergence to the correct parameters. because the logarithm function "stretches" bias on the untransformed scale to arbitrarily large biases on the log scale the closer the likelihood function gets to zero. If however R rises faster than $\sqrt{N}$ then the optimised estimator from simulation is asymptotically equivalent to the Maximum Likelihood estimator. See Section 6.2, page 94 for a more in depth discussion. An outstanding presentation of this can also be found in [26], Chapter 10. Resolving this problem has been seen as a compelling reason to adopt Bayesian estimation methods. Bayesian estimates can be constructed which converge to exact likelihood estimates. Bayesian methods of estimation via Markov Chain Monte Carlo (MCMC) methods and Gibbs samplers were initially employed, they were found to be difficult to program and debug because it cannot be seen iteration by iteration that a diagnostic statistic is indicating convergence. The simulators can drift around for long periods of time and it can be difficult to determine whether this is correct

behavior or whether a mistake has been made. With maximum likelihood it can be seen that the log likelihood decreases with every iteration. Also, if using mixing distributions other than a Normal distribution, it can be hard to specify a conjugate prior distribution. Of course, a non-conjugate prior distribution could be employed, but then there would be no guarantee that the convergent distribution is the one we require [10]. A significant improvement to the population level model based on using different distributions for each regressor and still specifying a correlation matrix will be presented in 6. This would cause considerable problems for a Bayesian estimator because no conjugate prior would be available and a non-conjugate prior would not guarantee that the converged estimate would be the same as the data. The ease with which improvements can be quickly incorporated into the maximum likelihood approach can more than make up for the admittedly unsatisfying ways to address the issue of bias incurred by the logarithmic transformation of simulated probabilities. This application required computational efficiency and the ability to easily specify non-standard mixing distributions which favours a frequentist approach. All that would be accomplished by adopting Bayesian methods would be trading one set of problems for another.

The bias in simulating the log-likelihood function can be expressed by taking a second degree Taylor expansion of $\log(\widehat{P})$, the log simulated probability around the actual probability P:

$$\log(\widehat{P}) \approx \log(P) \; + \; \frac{\widehat{P} \; - P}{P} \; - \; \frac{(\widehat{P} - P)^2}{2P^2},$$

and taking the expected value of this relationship implies

$$SML \; - \; MLE \; \approx \; -\frac{Var(\widehat{P})}{2P^2} \; < \; 0,$$

where SML is the simulated maximum likelihood and MLE is the maximum likelihood estimate based on the true probability P.

4.2.3 Other mixing distributions

So far, the correlation matrix can still be maintained in the same manner as for any multivariate normal distribution, with the proviso that coefficients of known sign have a ± 1 multiple of their logarithm being expressed as a correlation. Other mixing distributions can also be specified by simulating β_r appropriately. For instance, to specify heavier tailed distributions, say, a Multivariate Students t distribution could be employed. Instead of simulating β_r from a normal distribution as above, draw from a Students t distribution by borrowing from some insights developed by Bayesian analysis. In [10] section 3.1 the Students t distribution is shown to be a mixture mixed over a χ^2 distribution and thus one can make random draws from a Multivariate Students t distribution with location vector μ, scaling matrix Σ and degrees of freedom ν by drawing a vector η from Normal(0, I) and scalar $x_r \tilde{} \chi^2_\nu$ (draw from a Chi-Squared distribution with ν degrees of freedom) and then compute

$$\beta_r \; = \; \mu \; + \; \sqrt{\frac{\nu}{x_r}} \; L \; n_r,$$

where $LL^T = \Sigma$. One can then specify ν as, say, 4 to then generate much heavier tailed coefficient distributions.

However, simply being able to specify different mixing distributions is not enough. Different distributions for each regressor can be employed that can also correctly specify correlations between regressors having different mixing distributions. This is a *hybrid* mixing distribution. Such a probability distribution is very abstract and difficult to visualise, the resulting correlation matrix is very difficult to interpret. However, it makes sense that there is no compelling reason why a population distribution of coefficients for one regressor should be the same as a different regressor. The superior fit that such a model generates is strong evidence of the validity of this approach. As these regressors will eventually be used in a learning system, this shows what a complex process learning is and how intricate and subtle poker can be.

Recall from Section 4.2 that when β_r is simulated from a multivariate normal distribution $\beta_r = \mu + Ln_r$. So

$$E[\beta_r] = E[\mu + Ln_r] = \mu + L.E[n_r] = \mu$$

because n_r is a vector composed of terms each taken from a 1 dimensional standard normal distribution each term having a mean of zero and a variance of 1, so component-wise the expected value is a vector of zeroes. The variance

of

$$\begin{aligned} Ln_r &= E[(Ln_r)(Ln_r)^T] \\ &= E[L.n_r n_r^T.L^T] \\ &= L.E[n_r n_r^T].L^T \\ &= L.I.L^T \\ &= LL^T = W \end{aligned}$$

and as the variance does not change when a constant is added to each term in a sequence, this variance is unchanged by adding the constant vector μ. So β_r comes from a normal distribution having mean vector μ and variance W, which is of course the required distribution. The random component in the draw comes from the specification of n_r. From the specification above, a draw for a multivariate students t distribution can be expressed as above or by

$$\beta_r = \mu + L\widehat{n}_r,$$

where

$$\widehat{n}_r = \sqrt{\frac{\nu}{x_r}}.n_r$$

and with an additional draw $x_r \tilde{} \chi^2_v$. The vector $\widehat{n}_r$ is a vector that transforms the draw to achieve the required distribution. Now, if $[\beta]_1$ is to be Normally distributed and $[\beta]_2$ Students t(ν) distributed, $\widehat{[n_r]}_1$ can be specified to be from a 1 dimensional standard normal distribution and $\widehat{[n_r]}_2$ is $\sqrt{\frac{\nu}{x}}$ multiplied

by a draw from a standard normal distribution. Similarly, a component is to come from a uniform distribution, then that component can be defined to be from a uniform distribution having a mean of zero and a variance of 1. The correlation matrix is maintained in exactly the same way as for a multivariate normal distribution, but has the stochastic component updating the mean vector with what is now a hybrid distribution. All that needs to be calculated beforehand is how to draw from the required 1 dimensional distribution with the required mean of 0 and variance of 1. In the case of distributions such as a gamma variate, that always take a given sign, draw from the log-gamma distribution and exponentiate on use similarly to the specification of a log-normal distribution. So, in the estimation phase all distributions are transformed in such a way that the transformed values can take any real value. A hybrid distribution can then be specified as above and then the inverse transformation is used for those coefficients that have been transformed on use or when calculating log-likelihood values. The following approach, motivated by experience with many model fits became standard adoption. First use a Multivariate Normal mix for all regressors. A uniform mix for indicator variables can also be tried. The estimated location and spread of the uniform distribution implies values for the upper and lower boundaries of the uniform interval. Refit the model at this stage, accepting the new hybrid model if the log likelihood goes up significantly. Now try using log normal mix for regressors that necessarily take a given sign and refit and accept if the log likelihood improves. Next then try using a Students t

(4) model to each of the remaining regressors to test whether the model fit is improved with the resultant heavier tails. Each Students t regressor used in this way incurs an extra degree of freedom, tests of "goodness of model" criteria will need to be satisfied

Information Criteria Measures Commonly used measures, discussed in [6] are:

$$AIC = -2\ \log(L)\ + k$$

$$BIC = -2\ \log(L)\ + (\log(N)\ k)$$

$$CAIC = -2\ \log(L)\ + (1 + \log(N))\ k$$

Bayesian Difficulties with Hybrid Mixing Distributions Specifying a hybrid distribution would be extremely difficult using a Bayesian estimator. Even just specifying a Students t distribution is difficult because it is not clear what a conjugate prior should be. Of course a non-conjugate prior can always be specified, but that is a dangerous practice because there is then no guarantee of the posterior distribution being in the required family of distributions. Attempting to find a conjugate prior that meets a requirement for a hybrid distribution across different dimensions is a difficult problem and provides further justification for persevering with a frequentist approach.

Chapter 5

Adaptive Model

When we see men of contrary character we should turn inward and examine ourselves—Confucius

5.1 Adaptive modelling

Billings *et al* in [9] present a strong case for the need for adaptive play in an automated poker system. The approach adopted here is different in that no use is made of Neural Networks. The systems *Loki* [18] and later *Poki* [3] use a weight array which is a device whereby a computer can mimic the style of play advocated by David Sklansky [23] . In this sense, a system like *Poki* which uses a Neural Network, is playing in a very similar manner to an expert human player and the system is a very good example of well implemented Artificial Intelligence. Pre-defined player classifications such as

"tight" or "aggressive" are used to classify players into a best-fitting group as their choices reveal their attributes.

In estimating a Random Parameters model (chapter 4) the distribution of the coefficients was described. Now, that information will be used to estimate, on the basis of data being revealed, where on the distribution of coefficients a given context may be and therefore which coefficients should be used for that particular context. No pre-defined classes are invoked and no attempt is made to interpret player motives. The whole process is simply geared to using the extra information to form better realisations of the expectation maximisers $\sup(\pi_\Sigma)$ (page 3, definition 6). The more sensitive and less predictable nature of the process being employed makes play a lot less predictable, so seeking to maximise expectancy (page 2, definition 1)will no longer amount to just playing pot-odds.

The system learns, but in a different way to the systems of Neural Networks employed in [18] and [3]. The learning system is unsupervised in that no prior input is required after the population model has been estimated. In a sense, the attributes of given contexts follow from the assumptions adopted about the mixing distribution (chapter 4).

5.2 Mathematical analysis

Let the population coefficient probability distribution be f and the parameters of this distribution be θ. In the case of the model developed in Chapter 4,

f will be the Multivariate Normal distribution and θ will consist of the mean vector μ and the covariance matrix W. In the case of the P_WIN model (page 46), consider a given context for which previous data X , the *contextual data,* and previous responses Y, the *contextual responses* are available. Best fitting coefficients β are required subject to X, Y and θ.

Consider the joint density of Y and β subject to X, θ.

$$\begin{aligned} \aleph(Y\ ,\ \beta \mid X\ ,\ \theta) &= q(Y \mid X,\ \theta)\ .\ h(\beta \mid X,\ Y,\ \theta) \\ &= f\ (\beta \mid X,\ \theta)\ .\ p(Y \mid \beta,\ X,\ \theta). \end{aligned}$$

Hence

$$h(\beta \mid X,\ Y,\ \theta) = \frac{f\ (\beta \mid X,\ \theta)\ .\ p(Y \mid \beta,\ X,\ \theta)}{q(Y \mid X,\ \theta)}.$$

Conditioning has been conducted both ways. The functions p and q are probabilities because they describe the variable Y, which is defined probabilistically. The functions h and f are probability density functions. The joint density $\aleph$ is then an expression for either the probability of Y occurring in the population weighted by the distribution of contextual parameters or the probability of Y occurring in our context, weighted against the density of contextual coefficients occurring in the population. The function q, which is the probability of Y occurring in the population, is the population parameter space estimate of Y which is just the population level random parameters

estimate or,

$$q(Y \mid X,\ \theta) \ = \ \int_{\beta} \text{Prob}(Y \mid X,\ \beta)\ f\ (\beta \mid \theta)\ d\beta.$$

Note that the underlying density of coefficients within the population does not depend on X, and so

$$f\ (\beta \mid X,\ \theta) \ = \ f\ (\beta \mid \theta).$$

The distribution f will be recognised as the mixing distribution defined in the population level model. Also, once β has been specified, the values within θ no longer matter because θ is just a description of the possible values of β, so

$$p(Y \mid \beta,\ X,\ \theta) \ = \ p(Y \mid \beta, \theta) \ = \ \text{Prob}(Y \mid \beta, X)$$

It is also clear that the distribution h is what is being sought This defines the "category" for the context. Upon rearrangement, we have

$$h(\beta \mid X,\ Y,\ \theta) \ = \ \frac{\text{Prob}(Y \mid \beta, X)\ \ f\ (\beta \mid X,\ \theta)}{\int_{\beta} \text{Prob}(Y \mid \beta, X)\ f\ (\beta \mid \theta)\ d\beta}$$

This expression lends itself to a pleasing interpretation. The conditional dis-

tribution of the contextual coefficients is the probability of the response data given those coefficients weighted against the distribution of the coefficients within the population. The numerator is a normalising term ensuring that the distribution function h integrates to unity.

Next, the expected value of β under h is calculated. When this expected value differs from μ, the robot will have changed its style of play for the context under consideration. Let

$$\begin{aligned} B &= E\ [\beta \text{ in } h] \\ &= \int_{\beta} \beta \ \left[\frac{\text{Prob}(Y \mid \beta, X)\ \ f\ (\beta \mid X,\ \theta)}{(\int_{\beta}\ \text{Prob}(Y \mid X,\ \beta)\ f\ (\beta \mid \theta)\ d\beta)} \right]\ \ d\beta \\ &= \frac{\int_{\beta} \beta\ \ \text{Prob}(Y \mid \beta, X)\ \ f\ (\beta \mid X,\ \theta)\ d\beta}{\int_{\beta}\ \text{Prob}(Y \mid X,\ \beta)\ f\ (\beta \mid \theta)\ d\beta}\ . \end{aligned} \tag{5.2.1}$$

There will rarely be a closed form for this expression. However, simulation methods can be employed to achieve an arbitrarily accurate approximation. For sufficiently large R, we can simulate both the numerator and the denominator by

$$E[B]\ \approx\ \frac{\sum_{r=1}^{R}\ \beta_r\ \text{Prob}(Y \mid \beta_r,\ X)}{\sum_{r=1}^{R}\ \text{Prob}(Y \mid \beta_r,\ X)}, \quad \text{where } \beta_r\ \sim\ f\ (\beta \mid \theta). \tag{5.2.2}$$

This is the required approximation to the expression (5.2.1). It is easy to program and debug.

Of course, h represents a mixing distribution by itself and so a variance estimate can be obtained by calculating

$$E[BB^T] \approx \frac{\sum_{r=1}^{R} \beta_r \, \beta_r^T \, \text{Prob}(Y \mid \beta_r, \; X)}{\sum_{r=1}^{R} \text{Prob}(Y \mid \beta_r, \; X)},$$

from which

$$\begin{aligned} Var[B] &= E[BB^T] \; - \; E[B] \; E[B]^T \\ &= E[BB^T] \; - \; E[B] \; E[B^T] \end{aligned}$$

With the last line being a slightly different but equivalent notation that is sometimes preferred. Whether the transpose of the expected value of the coefficients is preferred or the expected value of the transpose of the coefficients is preferred is a matter for personal preference as the two expressions are equivalent.

Dependencies between the elements of B are captured by the simulation process because the draws of β_r values are made subject to W, the covariance matrix.

5.3 Identification issues

In most real world problems there are always limits to how much can be known about things which cannot be directly observed. An opponent's preferences cannot be realistically observed, and therefore their coefficients (preferences) must be inferred from the data. Unfortunately, if all parameters are allowed to vary randomly within the model, then this inference can be non-unique. Such parameters are referred to as *unidentified* and the situation as an *identification problem*.

Identification problems can occur when all parameters are treated as random. Specification of a fixed intercept with the regressor coefficients being considered as randomly varying within the population was found to fix the problem. So long as at least one predictor is considered fixed, identification problems were not evident Of course an underlying distribution will most likely have variance across all parameters. Solving an identification problem by specifying some parameters as fixed does not imply that they can't change, merely that a convergent model can't be otherwise estimated.

Chapter 6

Tree Based Contextual Coefficients

We all agree that your theory is crazy, but is it crazy enough?—Niels Bohr

Context specific coefficients (which can be interpreted as preferences) can now be assigned using the model developed in Chapter 5. When insufficient data are available to assign a context, the model reverts to the Generic Opponents model developed in Chapter 4. However inhomogeneity of the population of possible contexts may also be identified in less specific situations. For example, typical player behaviour on small limit tables may be "different" to large limit games. "Difference" here means that player behavioral preferences may result in different coefficients being appropriate and that therefore the data can be split into more homogeneous groups with separate models applying to each. This would enable a different contextual

model to be employed with the more specific models described in Chapter 5 still being available as data pertaining to specific contexts is revealed.

6.1 CART models

The idea of Classification and Regression Tree Models, hereafter known as CART models was developed by Breiman, Freidman, Olshen & Stone [5] Usage of the techniques published in [5] and via the Salford Systems computer package, also called CART, revealed that additional features would be required for this research which justified the development of custom software. As an example, consider the problem of whether different coefficients should be fitted for certain ranges of numbers of opponents in the P_WIN model (Section 3.1, page 46). Because many of the predictors used (Appendix A, page 127) are based on simulation results and those simulations are for the number of opponents encountered, the number of opponents (OPPONENT_COUNT) has already (indirectly) entered the model and so its significance as a predictor by itself is reduced. However, even though OPPONENT_COUNT is not a significant regressor by itself, it may happen that other regressors (independent variables) may have best fitting coefficients for different ranges of OPPONENT_COUNT. When a predictor is used to divide the data set into groups that display some sort of homogeneity, those predictors are called *auxiliary variables* in the CART literature and the same convention will be adopted here. A regressor (a predictor that enters the

regression function) can also be an auxiliary variable. Popular splitting rules for CART models include the *twoing* and *GINI* criteria ([5], [8]). Gambling problems are somewhat dissimilar to many other regression analyses in that the main focus is not whether or not a candidate regressor is significant. Rather, it is necessary to require the most accurate probability and expected value statements from the resultant model. The model output is as important as interpretation. More accurate probabilities here are not just "brag numbers". They could well mean the difference between winning and losing in the long term.

CART splitting rules are not in general guaranteed to maximise log-likelihood. New splitting rules were constructed that have this property. Consider a classification rule such as (OPPONENT_COUNT $<$ 4) or ($4 \leq$ OPPONENT_COUNT $<$ 7) where the split results in a higher log-likelihood by an amount sufficient to justify the extra model degrees of freedom. The regression coefficients are different in the split regions as a result of empirical heterogeneity. The construction really amounts to a CART tree with regression model end-points. This is different from the Regression Trees developed in [5]. This is in fact a classification tree of regression models.

6.1.1 Auxiliary variables and split data sets

Consider the auxiliary variable OPPONENT_COUNT (Appendix A, page 127). One hypothesis may be that games with different ranges of OPPONENT_COUNT may be inhomogeneous. This can be evidenced by different

coefficients (or preferences) for different regions of the data space corresponding to ranges of OPPONENT_COUNT values. For a candidate data space region, the data are split and a model is fitted on each of the child segments. The log likelihood for each of the segments is added to arrive at a total log likelihood value. The best candidate split is that which results in the highest total log likelihood value. In this way, the log likelihood is guaranteed to at least not decrease with every split. Of course, every split we select in this way will not decrease the log likelihood value and whether any increase in log likelihood is significant should be assessed. If the unsplit model has k degrees of freedom and one split has been conducted, then the new model has k degrees of freedom in each split section plus one for the critical value of the auxiliary variable. Thus the model degrees of freedom are now $2k+1$. Some goodness-of-model criteria need to be satisfied to balance the increase in log-likelihood against the extra model parameters. AIC and similar measures (Section 4.2.3, page 73) were employed for this. The process is repeated until a *stopping rule* is satisfied ([5]) . The stopping rule adopted here was that if further splits cannot result in enough increase in log-likelihood to satisfy the goodness-of-model criteria chosen, the process is halted. Even AIC, being the most liberal goodness-of-model test of those outlined in Section 4.2.3, page 73, typically only allows a few splits. Parsimonious models tend to be chosen.

6.1.2 Trees and split data sets

A tree consists of a *root node,* which is the data set prior to any splitting and *child modes* if splits have been conducted. On the terminal branches of the tree are *terminal nodes*, which are nodes that have not been split. When a split occurs, a terminal node becomes a *splitting node* and results in 2 *child nodes*. Each terminal node except for the root node prior to any splits has a *parent node*, which is the splitting node that results in the split. Each splitting node has a *splitting rule*, which is the condition on which the split occurs. Consistent with the example above a splitting rule may be OPPONENT_COUNT $\leq$ 5, which would result in 2 child nodes, namely an affirmative child node relating to all data meeting the condition and a negative child node relating to all data not meeting the condition. Splitting occurs until some *stopping rule* is satisfied.

Figure 6.1.1 shows the most basic type of split for a data set. The splitting node is the oval shape and the terminal nodes are the rectangles. The splitting rule is the condition displayed in the splitting node oval. In this case, a variable x_k is tested to see whether it is less than or equal to some critical value c_1. The choice of x_k and c_1 is made by the estimating process. The convention is that the affirmative data points in the split go to the left hand side of the diagram and negative values go to the right hand side.

In the case of regression trees, the data are part of a continuous range rather than a discrete number of mutually exclusive groups. Typically there

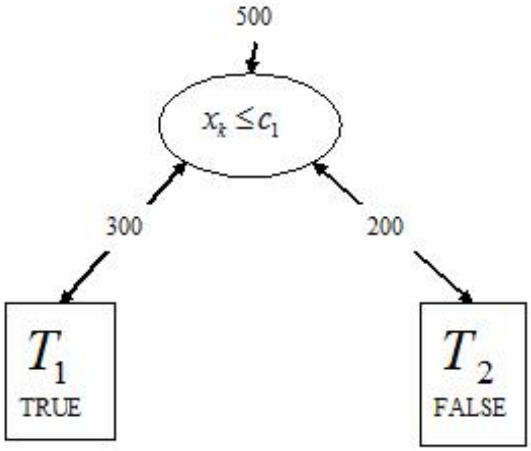


Figure 6.1.1: **Basic split for a dataset**

must be some additional criteria used to interpret the terminal nodes of a regression tree. This could be some convention (used in [5]) such as using the mean of the data points in each terminal node to be the regression estimate for data in that node. The process used here classifies data as belonging to some group that uses a certain set of coefficients, so the allocation of elements is discrete in that the data space is divided into a finite number mutually exclusive regions each of which is associated with an endpoint model. The classification groups are decided a posteriori as part of the estimation process and are not known a priori.

In Figure 6.1.2 a new split has occurred for another variable x_m tested against another constant c_2 on what was terminal node T_2 in Figure 6.1.1.

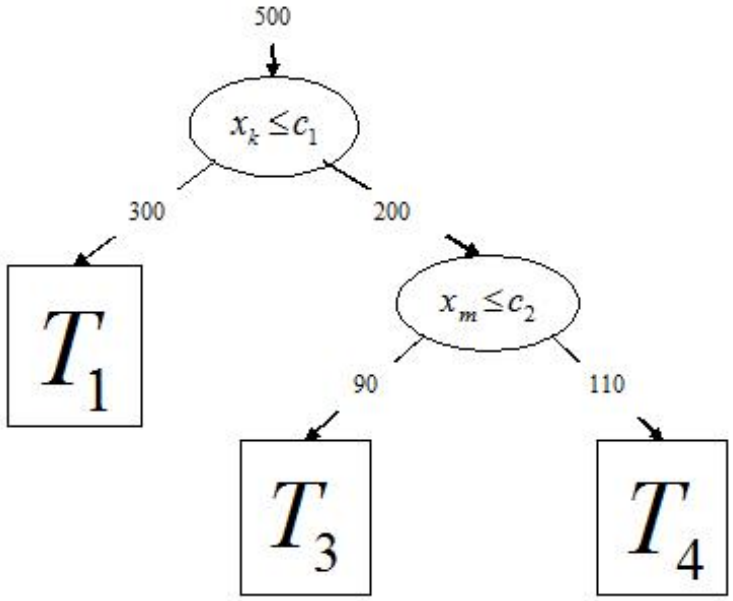


Figure 6.1.2: **Resplitting a node**

Node 2 changes in character from a terminal node to a splitting node. It is quite acceptable to re-split on the same variable. Node numbers are retained even though they may change in character from terminal to splitting nodes, with the root node always labelled zero. Resplits can occur *ad infinitum* until stopping criteria are satisfied.

Whenever a split occurs, a terminal node must become a splitting node and 2 more terminal nodes result. Thus, for each split, the number of nodes goes up by 2 and the number of terminal nodes increases by 1. For the tree object T, the number of terminal nodes is $|T|$. Thus $|T|$ is the number of

segments into which the data set has been split and by induction the total number of splitting nodes must be $|T| - 1$. Hence the total number of nodes is $2|T| - 1$.

Tree models have advantages and disadvantages. An attractive feature of the tree model is ease of interpretation. For example, a model classifying patients into risk groups for heart complications in an emergency ward could be used by nursing staff via a mechanism as simple as a wall chart. The various regressors could be traced down the tree which may go something like "is patient over 65", "is patient male", "does patient smoke" etc. This may lead to ease of interpretation by staff even if they have little mathematical background. However, some such models can have over-fitting issues. In Chapter 8 of reference [5] regression trees are discussed and a typical tree is grown. The implementation is such that data are dropped into the tree and found to reside in some terminal node and the predicted value is the mean of the in-sample values for that terminal node. Out-of-sample data may well have a much higher variance in a terminal node than the sample data and such an approach "stratifies" the data in the sense that the tree model's predictions will all be exactly the same for data until it varies by more than a critical amount, whereupon predicted values will then jump to a new level. This is a little unsatisfying.

6.2 Implementation

As has been stated, the population is not homogeneous in the sense that tastes, preferences and behavior patterns (as characterised by coefficients) vary. Conditioning on previous actions reveals in which segment of the population a new respondent set might reside. The segmentation of the population data achieved by this mechanism did not result in easily defined groups and they were referred to as "fuzzy groups" in Chapter 5. There are however many ways that a data set could be divided more simply . It could conceivably be the case that players on high-limit tables play differently to players on small-limit tables and that these differences in behavior patterns and preferences would be characterised by different coefficients. In such a case it might be expected that some segmentation of the data into groups defined by table limit size might be appropriate. A sub-model fitted to a small table limit group might have significantly different coefficients to a large table limit group and the resultant total fit might be significantly better.

Searching for optimal segmentations of the data set is a task to which a tree model is very well suited. Most tree models described in [5] provide a single number for all data points that come to rest in each terminal node. This number is a class number in the case of a classification tree or a real number in the case of a regression tree. The tree developed here yielded a different model for each terminal node.

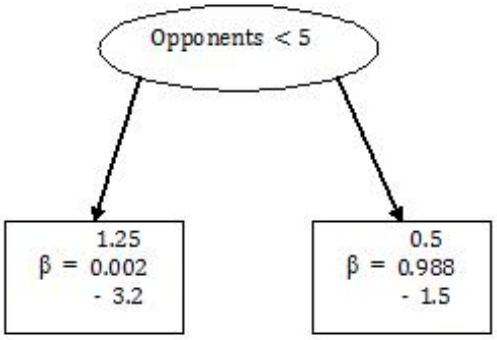


Figure 6.2.1: **Test Model**

In Figure 6.2.1 actual data and a very small test model with three regressors and one auxiliary variable is shown. For a table with a small number of opponents (in this case 4 or fewer), different best fitting coefficients exist than for a table with a large number of opponents. This does appear to be evidence of non-homogeneity in the population with regard to opponent behavior patterns. Data that resides in the left terminal node is simply data from tables where OPPONENT_COUNT is in the range {1,2,3,4} and data residing in the second terminal node is all other data. Data from the first terminal node does not always imply the same probability of having the best hand because the non-auxiliary data can vary. In this case the tree models parameter count after one split was 3 regressors in each of 2 terminal nodes plus one split, so the number of parameters is 7. The unsplit

base model has 3 parameters, an extra 4 parameters are being used. AIC incurs a unit penalty for each additional parameter and so requires an improvement in log-likelihood of 4 to justify the extra model parameters. The actual observed improvement was 38, providing evidence of heterogeneity in the manner postulated.

For a GLM (Generalised Linear Model [14]) with k regressors, there will be k model parameters in each terminal node plus one degree of freedom for each splitting node. So the tree model degrees of freedom is $k\,|T| + |T| - 1 = (k+1)\,|T| + 1$. Using AIC, each time a split is made the required improvement in log likelihood goes up by $(k+1)$. This very quickly gets difficult to achieve.

In practice a large amount of data is usually required to fit a model with a large number of regressors, so if k is large and only very significant regressors are fitted, then usually the number of cases N being modelled is also large and therefore there can be potential for large improvements in log-likelihood if there is some heterogeneity in the data that can be explained by an auxiliary variable. If on the other hand, a large number of regressors are fitted and the assumed distribution of errors is in very close agreement with the actual distribution of data in the data set, then no significant splits will be found because no splits will result in statistically significant differences in log-likelihood. The fact that significant splits are found can be considered evidence that the assumptions made about the distribution of errors in the chosen model form have been violated to some degree. No further significant

splits being found in each terminal nodes data can likewise be considered evidence that (at least locally) the assumed distribution of errors is more reasonable.

A further improvement is to also require some minimum number of data points in a terminal node before a split is accepted. Obviously if there are fewer than k items in a terminal node then the local model cannot even be estimated, so this could be considered as some absolute minimum. However, in order to fit any model significantly more data should be used. The task when estimating any GLM is to find parameters $\theta \in \mathbb{R}^q$, the q dimensional vector to be estimated. Random variables are sought that converge in probability to a value θ^* or, $\widehat{\theta} \xrightarrow{p} \theta^*$. This can also be expressed as the probability limit of $\widehat{\theta}$ equals θ^* or

$$p\lim \widehat{\theta} \;=\; \theta^*$$

The probability limit θ^* the is called the *pseudo-true* value. If, in the model $\theta = \widetilde{\theta}$ and the pseudo-true value is such that $\widetilde{\theta} = \widehat{\theta}$, then $\widehat{\theta}$ is said to be *consistent* for $\widetilde{\theta}$. GLM estimators $\widehat{\theta}$ are usually root-n consistent for θ^* and asymptotically normally distributed. For n cases in the data set, the random variable $\sqrt{n}(\widehat{\theta} \;-\; \theta^*)$ converges in distribution to the Multivariate Normal distribution with mean vector $\mathbf{0}$ and covariance matrix $\mathbf{C}$. Or

$$\sqrt{n}(\widehat{\theta} \;-\; \theta^*) \xrightarrow{d} \; N[\mathbf{0}, \mathbf{C}]$$

So, in order to satisfy the dual aims of achieving consistency and also numerical stability, the required data should have a $\sqrt{n}$ value that rises faster than k.

6.3 Future Directions For Tree Based Models

So far the model improvements considered in this chapter, being tree models, conduct splits based on critical values which necessarily split the data into regions parallel to an axis corresponding to an auxiliary variable. This worked reasonably well for the purposes required, but a more general approach would involve splits that were not required to be parallel to an axis. Linear combinations of auxiliary variables have much more chance of revealing heterogeneity characteristics in the data. Standard tree models are not well suited to this. Application of these ideas to Support Vector Machines ([8]) could result in better fitting splits being identified because the requirement for splitting parallel to an axis can be relaxed.

Chapter 7

Deployment

The fight is won or lost far away from witnesses - behind the lines, in the gym and out there on the road, long before I dance under those lights.

- Muhammad Ali

7.1 Background to *THOR*

The suite of programs that formulates strategy decisions such as Fold, Call, Check, Raise, Sit-Out is referred to as THOR (page 53). Each program in the suite of programs that comprises THOR will be referred to as a *layer* of THOR. The first layer is a modelling program which implements in code all of the concepts and particularly the mathematics contained in this thesis. This layer estimates the Generic Opponents (Chapter 3), Adaptive (Chapter 5) and Tree-Based (Chapter 6) models. It requires data which is stored in a

database back end. The architecture of the database is quite standard and will only be superficially discussed. All of the software was written with Delphi 7 Enterprise.

Many layers of THOR are implemented using a parallel computing cluster, so even though multiple computer programs may be running, these all use the resources of a cluster of computers. Each program can utilise the resources of multiple CPUs on self contained segments of data that can be processed in parallel with no need for synchronisation. The system evolved to be a central computer with necessary programs residing on it with many processor intensive tasks being divided up and delegated to other computers thus saving processing time by having all computers working together when intensive tasks are required. This architecture is not new, THOR lends itself to this sort of parallel computing cluster design.

Figure 7.1.1 gives a pictorial representation of the implementation of THOR. Note that THOR is not really a single software entity. Rather it comprises all the software technology pictured below the internet level.

Figure 7.1.1 represents THOR playing on M sites using N robots. The only required relationships between M , N and K are $K \geq 1$, $M \geq 1$, $N \geq M$. If there are more sites than robots then not all the sites can be engaged by an active robot, hence the condition $N \geq M$. Increasing the value of K does not directly facilitate playing more sites but increases the speed at which strategy decisions can be made by using more parallel

computers. This can be required as the number of active robots increases, but the requirement is indirect.

7.2 Collusion

A question that sometimes arises is "Do you use multiple clients on the same table?". That is, is collusion used? The implication here is that there could be more than one client playing on the same table and sharing information. The short answer is that collusion was not used. On a practical level the situation can occur whereby the only remaining clients on the table are the multiple robots sharing information. In order that this is not highlighted all hands should still be played to conclusion. One of the tell-tale signs of collusion would be a player with a strong hand folding early in a game to a hand slightly stronger. Thus, with collusion the multiple clients can often be playing against each other and so not generating any profit expectancy. The same number of players could have been playing on other tables generating profit expectancy. It was usually the case that overall expectancy was higher when extra clients were engaged to play other tables. Rather than colluding the extra players are better used at other tables.

7.3 Decoding sites

Each client program "knows" about the site it is playing. The different processes by which the output from a particular site is converted to data that can be used is encapsulated in each client program. The client programs are what most people think of as the "poker robot". Although this part of the implementation is very important, it is a very small section of the required system. The most important part of the system was the strategy engine. Software to decode poker sites can be purchased, but most of these do not contain logic to make strategy decisions and the ones that do tend to be rule-based, very predictable and weak. All of the sites played use their own custom software to display to the user the information from the site such as the user's hole cards, the pot size. These are usually accompanied by some graphics and animations designed to amuse the user and make the site software window look like a real world poker game. The information from the site is displayed in site software that the user runs on their own computer. The site software also administers security so that other users cannot see our hole cards etc. Decoding the site is the conversion of this interface to data THOR can use. This varies from site to site. Some sites use Windows, Apple or Linux controls which can be accessed via the appropriate API. Some sites use a stream of bitmap images. Some are ActiveX or Java controls. None of the sites is easy to decode and viable decoding procedures are specific to each site. The client software is a piece of software that converts the site

interface into data THOR can use. This is a two way process. After a strategy decision is made, that decision needs to be recoded into commands that can be inserted into the site software, so that the site software can receive commands and thus be appropriately controlled. This is known as *reverse engineering* the site interface in the jargon of software engineering. After appropriate decoding, the state of the game is known, what stage the game is at (Pre-Flop, Flop, Turn, River) what betting round the game is in (typically there are 3 betting rounds per stage, but this number can vary according to the site), the hole cards, any community cards, what the pot size is and what other player's actions have been in the game so far. The interface needs to relay data from the site to the model. That process results in a strategy decision which is then dispatched back to the site as if a human user had made a decision and clicked the appropriate control.

7.4 Hand Rating

Classes of hand types were allocated to allow sufficient range to include all possible rankings of hands within that type. The classes are (all expressed in hexadecimal):

$$
\begin{aligned}
\text{High Card} &= 0_{16} \\
\text{One Pair} &= 10000_{16} \\
\text{Two Pairs} &= 20000_{16} \\
\text{Three of a kind} &= 40000_{16} \\
\text{Straight} &= 80000_{16} \\
\text{Flush} &= 100000_{16} \\
\text{Full House} &= 200000_{16} \\
\text{Four of a kind} &= 400000_{16} \\
\text{Straight Flush} &= 800000_{16} \\
\text{Royal Flush} &= 1000000_{16}
\end{aligned}
$$

Hexadecimal values were used so that the presence of bits when converted to base 2 act as switches. This is standard computing practice and is discussed in detail in [19]. A quantity called the *card hand value* is introduced. The card hand value is just 2^v where v is the face value of the card with the

special cases of

$$
\begin{aligned}
v &= 11 \text{ for Jack} \\
v &= 12 \text{ for Queen} \\
v &= 13 \text{ for King} \\
v &= 14 \text{ for Ace High} \\
v &= 1 \text{ for Ace Low}
\end{aligned}
$$

The distinction between Ace Low and Ace High is that for a Straight an Ace can be valued at 1 or 14 as required to make the Straight. By sorting in order of value, then truncating to use only the top 5 cards and then looking at consecutive cards after sorting, pairs and three of kind and so on can easily be identified. By searching for consecutive cards all increasing in value (not card hand value) by one, Straights can also be identified. A special search where an ace is temporarily valued at 1 can identify ace low flushes. A pair with a "three of a kind" can be identified as a Full House. Then, the cards are sorted in order of suit. This identifies Flushes. If a Straight has already been identified then a Straight Flush has been found and if the top card is an ace, then a Royal Flush has been identified. Within each class the card hand values are combined with a Boolean AND operation ($\wedge$) and added to the class value. This way, the hands are guaranteed to be ordered in the sense that if one hand has a higher hand rating than another, then it is the better hand and if 2 hands have the same hand rating they tie. Due to

powers of 2 being used to describe card hand values, the top pair dominates all smaller pairs so high pairs in say 2 pair hands are always correctly ranked. Obviously the gaps between hands using this system are inconsistent. That is not a problem, but it should come as no surprise that where hand rating enters a regression function as a predictor, then the logarithm usually results in a better fit.

7.5 The Simulator

One way of looking at THOR is that it provides a method of weighting simulation results against other predictors and contexts of interest. In some regression functions (Appendix B, page 131), raw simulation results are used. In others, interest is focused on conditional results such as simulations subject to all remaining opponents having at least a high board pair, so if the highest card on the board is a King, then the results in question assume that all remaining opponents have at least a pair of Kings. In this case players in the simulation having less than a pair of Kings would be assumed to fold. Interesting results are obtained where conditions are imposed that players having less than some sort of "average" hand are assumed to fold. The measure of "average" that seems to result in the best fitting predictors relates to Geometric Mean. So all players in a simulation having a hand rating less than the Geometric Mean are assumed to fold. There is a very large number of candidate predictors than can be constructed in this manner. Just about

any assumption about player behavior can be considered by an appropriate simulation result and then become a candidate predictor.

All simulations are carried through to the conclusion of the simulated game and all start from the known condition of the hand. The number of wins, number of losses, number of ties and total simulated number of hands are the output for any of the simulations. Statistics of interest can be calculated from those values. Using a fixed set of simulation assumptions would be unlikely to result in an effective system, but having a system that can adaptively weight the various simulation assumptions against other predictors and the context at hand can capture sufficient variation in behaviour to estimate quantities of interest and therefore determine whether the cost associated with any given action (such as raise or call) when compared to the expected return constitutes an advantage bet.

7.6 Strategy

In order to make strategy decisions, the population and its heterogeneity requires modelling. This is a base level of modelling and is performed infrequently.

1. Estimate Generic Opponents Model (Chapter 3).

2. Optional but often useful step. Use Tree-Based Endpoint model to determine categories and heterogeneity boundaries. This process divides data into categories $(1..T)$, Chapter 6.

3. For each category estimate Random Parameters Model (Chapter 4). If no Tree-Based categories are determined, there will just be one category.

Now, a set of population parameters has been determined. On presentation of new data for each strategy decision required, subject to current category and past data as it occurs for the current context, the population preferences (coefficients) are conditioned to contextual coefficients as per formula 5.2.2, page 79. A set of context specific coefficients is at this point available. If the current context differs from the population characteristics sufficiently (if the coefficients are much different from the population mean), the context specific coefficients will change quickly as more data presents, ie the system will learn. The coefficients will relate to things such as win probabilities for the P_WIN model, tie probabilities for the P_TIE model, expected pot size and expected cost to play. Given these estimates, the expected cost to play and expected pot size can be calculated for any given action (Bet, Raise, Call etc.) by using the context specific coefficients in the appropriate regression function. Each action can then be associated with an $E[\pi]$ (definition 1, page 2) estimate, the action associated with the highest value of $E[\pi]$ is the $\sup(\pi_\Sigma)$ (definition 6, page 3) action and is the action undertaken. If cost to play is zero and no advantage bets can be made with the current hand, then the robot will check. If cost to play is greater than zero and no advantage bets can be found, then robot will fold. Cost to play is notified by the site when an action is required. The decision as

to whether available funds allow play on a given table is determined by the Kelly criterion (page 10).

Even though the decision process is completely deterministic, it is sufficiently complex and varies sufficiently with given hands in different contexts that the play is unpredictable to opponents at the table.

7.7 Other Details

The implementation of THOR is quite computer intensive, but with the low cost of computing power this does not present a large obstacle. Due to the very intensive requirements for some of the modelling layers required, particularly the Random Parameters model, a cluster was employed. Windows XP Professional operating system and Borland Delphi 7 Enterprise development environment were used. Both of these technologies would now be considered to be dated but they were more than adequate for all requirements.

In order for the cluster to do its job, there needs to be some way for the computers in the cluster to communicate. Delphi 7 provides many such mechanisms such as DDE, DCOM, SOAP. TCP/IP Socket components provided as standard with Delphi 7 Enterprise were used purely on the basis that they worked straight away and seemed easy to master.

MySQL 5 database management system was chosen as the database back end. This was downloaded free from MySQL, after reaching a predetermined profit figure the free Community edition was updated to the

Cluster edition.

The modelling software was written using Delphi 7. Some of the early models are merely standard Generalised Linear Models and thus are not very computationally intensive and any cluster architecture is excessive for such a task. However, estimating the Random Parameters models is computationally very intensive and would require weeks of processing on a single PC. The algorithms for the models are a Pascal translation of the mathematical algorithms presented in this thesis. The strategy of dividing the modelling dataset into segments for calculations such as log-likelihoods and gradients was employed. Each segment is processed in parallel on its own processor and then the results for each section are collated and summed centrally. Only tasks that could be divided into obvious segments that could easily be processed in parallel and in isolation were clustered in this way. Fortunately this represented well over 90% of the processing load and so the cluster architecture represented a huge gain in processing power.

When the client "sees" a snapshot of a game, this snapshot must be converted into information THOR can use. There are many different ways of displaying site information. Those sites that use operating system calls are the easiest to decode because one can write a hook into them and extract information. Those that display games as a stream of bitmap images are quite difficult to decode and probably these sites go to some effort to make their user interfaces even more difficult to decode. Player names and pot sizes are often displayed in fonts that are very inconsistent and seem designed

to be difficult for a computer process to decipher.

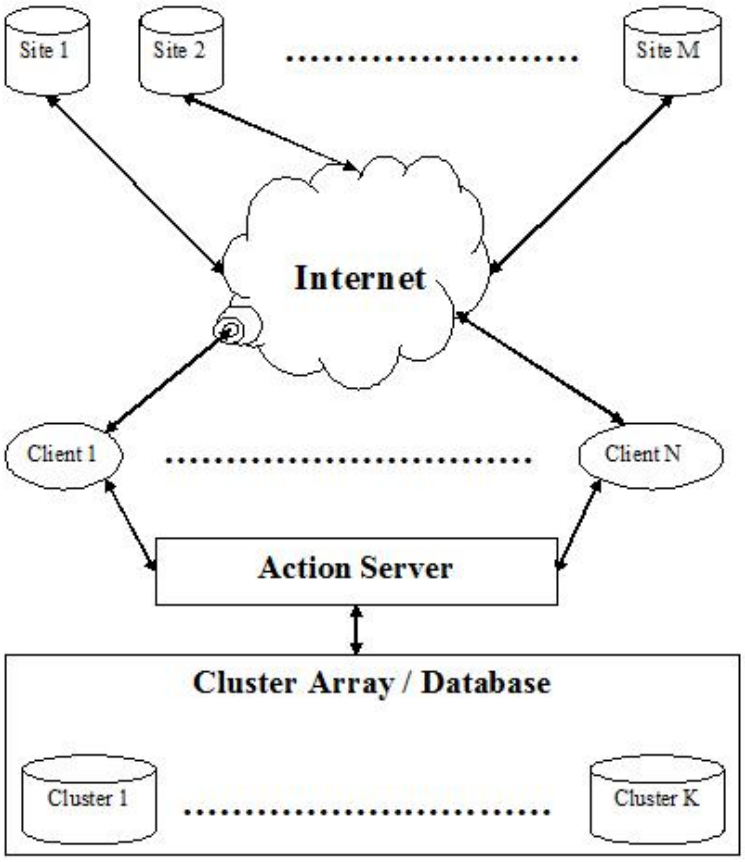


Figure 7.1.1: **THOR implementation**

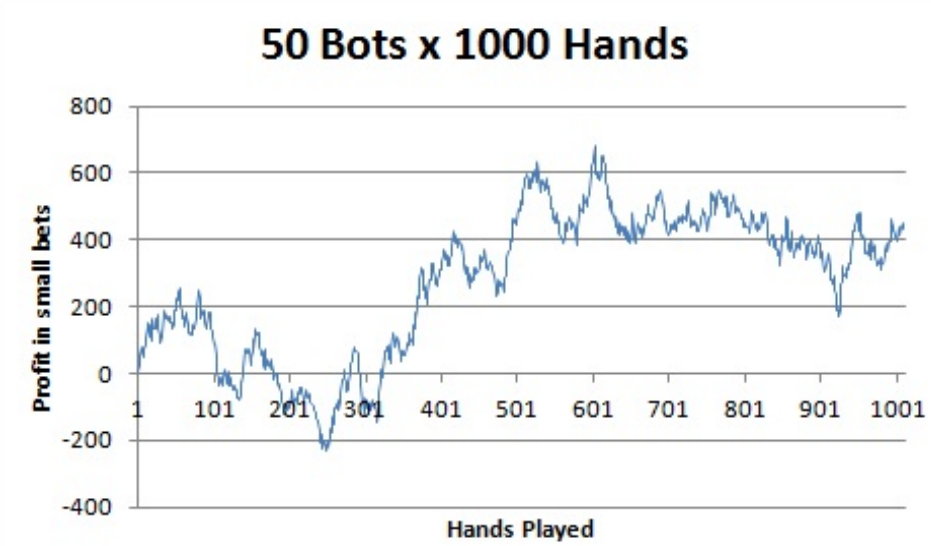
Chapter 8

Results

It doesn't matter how beautiful your theory is, it doesn't matter how smart you are. If it doesn't agree with experiment, it's wrong—Richard Feynman

8.1 Discussion

All hands played cannot be displayed because there were over 2 million. Some detailed output for a sequence of 1000 hands across 50 robots can be found in the appendices. The table limits vary. The individual trajectories are displayed graphically in Appendix C, page 141.



Results all bots

The overall graph in Figure 8.1 (page 112) is typical of tables with strong players present. To start with, where no context-specific data are available, all the robots are effectively playing with a Generic Opponents strategy. There tends to be an initial period of profitability, then some players adapt to the fixed strategy. The human skills of pattern recognition appear to take effect very quickly, and profitability typically decreases, in this case turning into early losses. As more data are gathered for specific contexts, the Adaptive Model detects the variance in the best-fitting coefficients and changes strategy accordingly. The model becomes more competitive and gradually improves. Then, what typically happens is that the profit trajectory oscillates as the opponents and then the Adaptive Model change strategy and

adapt to each other. The hope is that the Adaptive Model, being able to account for more predictors and being mathematically consistent can outperform the human ability to perceive patterns quickly and make intuitive changes. Overall, the 50 robots were 412 small bets ahead after 1000 hands. This translates to an average win rate of 0.00824 small bets per hand played. Previous runs conducted over similar numbers of hands resulted in higher profit rates than this, so this is actually a conservative sample in terms of profitability. Later models provided a much better win rate, but their extra complexity due to larger numbers of predictors did not serve to illustrate the methods employed as well as the simpler models discussed in this thesis. The later models employed the same methodology, the only difference was the larger number of predictors employed.

Obviously, averaging across 50 instances reduces the variance of the total by $O(\sqrt{n})$ so one should expect to see much less volatility in the total result than between individual tables.

The *Loki* robot (described by Papp [18]) was tested against various assumptions about player behavior and also against other robots. A win rate of 0.08 small bets per hour was reported. This is significantly better than the result reported here, but the test was against different versions of *Loki* and other robots, not against human opposition in a real-world setting. The win rate reported with this early model also falls short of the expected returns claimed by David Sklansky *et al* in [15]. This is unsurprising, David Sklansky *et al* are world-class expert poker players with decades of direct

experience.

The objective was to develop an original technique of adaptive machine learning rather than simply to develop a winning poker robot. Poker was chosen as an application for reasons already discussed in Chapter 1. Future applications were of more concern than iteratively refining poker robots. Many players take a dim view of automated play and numerous sites have taken quite aggressive action to prevent this and other automated systems playing. For these and other practical reasons, poker is not a long-term focus. Time is better spent investigating areas where automated trading is not frowned upon.

An article in *New Scientist* [1] described the work of a team including Billings, Bowling, Schaeffer et al who used a new development of the systems developed at University of Alberta called *Polaris* to play against some of the very best players in the world in No-Limit Texas Hold'em which *Polaris* won. However, the no limit form of the game is quite different from the limit form investigated in this thesis. Texas Hold'em No Limit is different in that the game has much more emphasis on how one performs relative to the opponents. Long term mathematical expectancy is not so much of as issue when a player can go "all in" and win the game sequence in one hand. Essentially the difference is assessing whether the current hand is "in-front" rather than looking at the expectation from a long sequence of small trials as in the limit game. The system developed in this thesis is not—in its current form—appropriate for the no limit game.

A possible weakness with THOR is a lack of a recency weighting system. Many neural network models generally (and not just with poker) tend to put too much stock in short term fluctuations. The result of trying to avoid this was to go too far the other way. Instead of conditioning on all of the past contextual data as outlined in Chapter 5 and specifically using Equation (5.2.2), decaying the influence of data as it gets older could also be investigated. How quickly (or even if) this decay should take place is a subject for future research. It may be that the current system is adequate but it may also be that recency weighting data may improve the decision making. That way, if an opponent plays differently now than, say 500 hands ago, an appropriate adjustment can be made. At the moment adjustments are made as more data come to light, but the contextual coefficients may be too slow to react to the dynamics of the situation.

Chapter 9

Conclusions

It is dangerous to be right in matters on which
the established authorities are wrong—Voltaire

9.1 The application

This thesis has described the development of adaptive models that respond to the behavior of competitive agents, using as its basis the application to on-line Limit Texas Hold'em Poker. Although there are other potential applications of these methods, many attempts to model real world behavior for applications with uncertain outcomes, such as financial markets trading, would need the support of external bodies such as banks, investment houses and so on, which has the potential to influence the direction of the research. For this research large quantities of publicly available data were required with

a small startup cost, no reliance on external funding and complete freedom of action. Online poker satisfied these requirements admirably and it was for this reason that the application was chosen.

The algorithm presented in this thesis is not a *supervised learning* method. In supervised learning we have a label and predictor variables for each item and seek to associate predictor sets with a label. An example of this might be deciding which cars are "sports cars", based on observable characteristics. Another example might be to categorise poker players as "aggressive" or "tight" ([3]). This method entails having *a priori* labels which introduces subjectivity into the model, which is something that I have sought to avoid. This is not to denounce supervised learning methods. Some, such as Support Vector Machines, are incredibly powerful but these methods are not easily applied when the data are unlabelled. At times the data used in this thesis has been regarded as only probabilistically labelled and at other times as labelled by some latent process, so that the labels cannot be directly observed. Either way, the data does not have any firm label, so it is always necessary to consider a context (which may be a single opponent) in light of how interaction with that context is likely to affect what is being modelled (for example P_WIN), rather than applying some hard and fast classification such as "aggressive".

The methodology has some properties of an *on-line* process in the sense that it tunes itself as more data presents, but it also has some properties of a *batch process* because it relies on a historical training data.

9.2 Hard vs "Fuzzy" groups

Previous research into modelling poker sought to develop so-called Specific Opponents models ([3], [20]). The approach here is not to attempt to classify specific opponents, but rather to investigate how various contexts impact the models and what the coefficients (interpreted as strategic preferences) should be to use those impacts to best advantage. As has been discussed in Chapter 5 as part of the Adaptive Model, contexts are classified by how changes in various attributes (which are the regressors) affect the quantities being estimated (such as P_BEST, E_POT etc.) as the context changes. Thus we are looking for classifications that have a significant effect on whether our strategy is likely to have the envisaged outcome. Hard groups are—by their nature—finite and discrete groups. Fuzzy groups vary continuously within the population. The approach has been to estimate how the population of games contexts varies and this has been done by interpreting coefficients of the models as measures of preferences for various behavior patterns. The population model then seeks to estimate the probability distribution of these coefficients within the population and thereby estimate how behavioral preferences vary. Assessing an individual context then amounts to using past data on that context to see where on the population distribution it resides and therefore what coefficients are best to use. This gives a measure of the behavioral preferences for that particular set of opponents in a particular situation.

A submodel developed in this treatise is the Tree Based Contextual Coefficients model discussed in Chapter 6. This was sufficiently developed to handle single auxiliary variables, but the handling of multiple auxiliary variables was shown to be less than straightforward and potential problems with many existing tree based classification routines have been highlighted. The main problem was that splitting on each auxiliary variable parallel to an axis (and therefore without reference to the other auxiliary variables) can result in a chronically overfitted model which will fail out-of-sample testing and that this problem can be avoided by introducing splits that are dependant on the values of other auxiliary variables. This sort of tree based model has only been superficially investigated in this thesis. The development was only taken to the point required. Further research into this field could produce results far more wide ranging than those considered so far in this thesis. Methods of specifying dependent splits with soft boundaries and nonlinear dependence, whereby context specific coefficients can be estimated, would have very far reaching ramifications. Such models would be viable competitors for many random parameters and latent variable models because the model degrees of freedom are much less than the typical latent variable specification and a best fitting set of submodels could be efficiently obtained without the need for assumptions about the distribution of the submodels within the population.

9.3 Expert opponent vs population tests

A possible test of the validity of the methods presented here would be to engage high profile poker players to test the system. However, this approach has some shortcomings. In a game such as chess, nothing is lost by treating an opponent as an expert, but this is not the case in poker. An essential element of successful play in poker is that weak players need to be exploited and a system that cannot do this may have disappointing results. Also, in any sort of high profile man vs machine tests, the human player often seems to play in a different or unusual way in an attempt to "fool the machine". Thus even though a test against world class experts is significant, it only tells part of the story with poker and in some other applications it would be almost meaningless. For instance, if a financial market model can consistently profit from suboptimal decisions by some segment of the market, then that segment can be targeted and segments where suboptimal behavior is not evident can simply be avoided.

9.4 Further Research

The work in this thesis provides an alternative to Game Theory for certain types of problems. An issue with Game Theory is computational intractability. As the problem reaches even modest levels of complexity, the Game Tree grows exponentially. This is one of the properties of Game Theory that was specifically addressed in ([2] and [9]) in the context of poker. A common

strategy is to remove aspects of the problem that are not seen as vitally important and to solve an approximating problem. This can be thought of as finding an exact solution to an approximating problem. The introduction of ϵ-Nash equilibria as advocated in ([20]) also introduces some notion of an approximate solution, but not really comparable to what is accepted more generally in numerical analysis. The approximations provided by ϵ-Nash equilibria do not converge smoothly to an exact solution in the way that is expected in numerical analysis. But, sometimes a ϵ-Nash equilibria can be seen to have the same properties as an exact solution. The approach advocated in this thesis falls into a widely accepted class of models whereby an approximate solution is sought for the exact problem. If boundary conditions could be determined whereby the Adaptive Regression approach advocated in this thesis converges asymptotically to the full Game Theoretic solution to classes of problems, then the full Game Theoretic solution could be implemented in a computationally tractable way by using the regression approach with a sufficiently large data set. If such boundary conditions can be defined and for certain problems shown that the Game Theory and Adaptive Regression approaches are asymptotically the same, then the Game Theory solution to some problems can be determined by using the computationally less demanding Adpative Regression approach. Conversely, for smaller problems where the two solutions are asymptotically the same, the Game Theory solution can be invoked as the exact solution. Extra data is often easier to organise than exponentially increasing computational power. This sort

of extension work has the potential to open up many problems where Game Theory solutions are sought, but computational issues prevent implementation. Investigation of the classes of problems for which the two approaches are equivalent also invites us to ask for which classes are they not equivalent. As is often found in mathematics, such questions can illuminate further aspects to the problem that may otherwise be difficult to see. Even just having the Adaptive Regression approach as a check on Game Theory may be useful. If for certain problems these two completely different methodologies yield similar solutions, then researchers can be more confident in those solutions than if only one methodology had been utilised.

Interest has already been expressed in the methods developed here for trading in financial markets or indeed any sort of trading conducted in an exchange environment where competitive agents can exploit predictable behaviour. The analogies to poker are obvious with actions such as "Raise" and "Check" etc being replaced by actions such as "Buy", "Sell" and "Sit Out". The methods advocated here are quite "data hungry". This was at first not seen as a huge issue because interest could be centred on complex systems with large amounts of data involved which, due to computational intractability presents significant problems for the Game Theory approach. Rather than specifying a fully correlated covariance matrix at the population level model, a specification of a latent class model ([24]) could be employed. This would result in fewer model parameters and therefore lower data requirements. Indeed, this is one area that is being investigated in terms of some

exotic markets trading for certain financial markets. The reduction in the computational requirements would make the Adaptive Regression approach more attractive for large problems where intractability poses significant issues for Game Theoretic methods.

A fascinating possible future application came to light during an informal discussion with a law enforcement official. Evidently, some criminal groups employ sophisticated methods to minimise the probability of activities being detected by using techniques such as regression analyses of factors of interest to estimate such things as probabilities of illegal consignments being detected subject to certain conditions. Past behaviour and whether or not illegal consignments were detected are available currently as crime statistics and details of investigations (even those that did not result in successful prosecutions). If some sort of Generic Opponents model (definition 12, page 4) such as simply choosing conditions consistent with the lowest estimate of detection from a regression model is employed by criminal groups, then the predictable behaviour can very easily be exploited by an Adaptive Regression algorithm to stay "one step ahead" of any such process and to indicate to law enforcement officials how to foil such tactics. Given that in this case the opponents strategy would be a simple regression model, the Adaptive model would be pitted against a "baby" version of itself which would be a very attractive feature. The adaptive nature of the algorithm would ensure that any change on the part of the criminal group would be very quickly countered and that further if the law enforcement officials could hypothesise which criminal groups were

involved, the various illegal groups' own histories of behaviour (as investigations would reveal) could very quickly establish contexts from which their previous actions and the interaction of histories of different criminal groups could betray their tactics in an ongoing and adaptive fashion. Of course such a scheme would not replace investigative policing. It would merely detect patterns from past behaviour and thus provide possible "leads" in a way that would be difficult for human investigators to uncover. This could be seen as analagous to a very high stakes game of poker, but one played against Generic Opponents models and therefore opponents that an adaptive strategy should have some confidence in defeating.

Appendix A

Description of predictors

A prefix of "LN" indicates a natural logarithm. A prefix of "SQRT" indicates a positive square root.

N_POT_SIZE: The normalised pot size, that is pot size divided by the table minimum bet. This could also have been expressed as table maximum bets, minimums were chosen to allow for the possibility of No Limit games at some later stage.

OPPONENT_COUNT: The number of opponents active in the game at the time a strategy decision (e.g. Fold, Raise, Check) is made.

ACTIVE_STAGE_BETRAISE_COUNT: The count of the number of bets and raises made at the current stage (e.g. Turn, River) by player still active in the game. So, if a player bet early in a stage and then folded before the current decision, that bet would NOT be counted in this predictor.

ACTIVE_STAGE_CALL_COUNT:

Similar to ACTIVE_STAGE_BETRAISE_COUNT, but number of Calls are counted.

ACTIVE_PAST_STAGES_BETRAISE_COUNT: The count of bets and raises made in past stages (not including the current stage) for all players still active in the game.

FOLD_COUNT: The number of hands folded at the current point in the game.

HAND_RATING: All the hands are rated on an integer scale such that if 2 hand ratings are the same, then there is a tie between those hands and if one is greater than the other, then the greater one is the superior hand.

FLOOR_GEOMEAN_RIVER_LWIN: A predictor from simulation results. Each of the player hands is simulated a large number of times. The geometric mean of the best hand rating at the start of the River is calculated. Only those opponents having a hand rating at least as high as the geometric mean of the best one are considered. The empirical probability of the robot winning against this subset of hands is calculated from the simulation results. The log-odds of this probability enters as the predictor value. In the event that this probability is zero, a default value is provided. Default log odds of -50 was used.

FLOOR_GEOMEAN_RIVER_LTIE:

Similar to FLOOR_GEOMEAN_RIVER_LWIN, except the tie probability

log-odds is extracted.

FLOOR_GEOMEAN_ALL_OPP_WIN: A simulation result. The hand is played out to conclusion via simulation. The geometric mean of the best hand is calculated. Only those hands from the simulated hands for the robot having a hand rating at least as high as this geometric mean are considered. The empirical probability is the predictor value.

LOGIT_FLOOR_HIGH_BOARD_PAIR_WIN: The highest card from the board cards is taken. The simulation results where opponents have a hand at least as good as a pair against the highest board card are considered. If the board cards contain a hand that is better than the high board pair, then all simulations are considered. From the considered simulated hands, calculate the empirical probability of the robot winning against only those hands. The log odds of this probability is the predictor value.

IS_FIRST_UP: An indicator variable of one if the robot is the first to make a decision when the decision is required and zero otherwise.

LOGIT_SIM_WIN: The log-odds of the empirical simulated unconditional win probability.

Appendix B

Regression output

In order to save space, much of the regression output has been suppressed. The output displayed is sufficient for our discussions. Output such as p-values, while important does not add to what is being discussed here. The number of cases in each regression varied, however they were all in the millions of hands for each game stage. The later output has been suppressed, with the number of predictors fitted it makes almost no difference given that the number of cases is always in the millions of hands.

Pre Flop

Cost To Play

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = NORM_COST_TO_PLAY

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
CONSTANT	0.656493	0.004844	135.53618
N_POT_SIZE	0.097776	0.003153	31.009689
OPPONENT_COUNT	0.262832	0.018574	14.150316
ACTIVE_STAGE_BETRAISE_COUNT	0.641382	0.055425	11.572084

Pot Size Delta

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = NORM_POT_DELTA

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
N_POT_SIZE	0.465022	0.037056	12.548932
OPPONENT_COUNT	1.779646	0.237575	7.490852
ACTIVE_STAGE_BETRAISE_COUNT	2.925465	0.603145	4.850351
ACTIVE_STAGE_CALL_COUNT	0.452106	0.077653	5.822117

Tie Probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_TIE

Model Type = Binary Logistic

PREDICTOR	COEFF	S.E.	|COEFF/SE|
FLOOR_GEOMEAN_RIVER_LWIN	0.628099	0.037056	16.171013
FLOOR_GEOMEAN_RIVER_LTIE	0.715007	0.052094	13.725251

Win Probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_WIN

Model Type = Binary Logistic

PREDICTOR	COEFF	S.E.	|COEFF/SE|
ACTIVE_STAGE_BETRAISE_COUNT	-0.169074	0.008519	19.846970
ACTIVE_STAGE_CALL_COUNT	-0.130693	0.01034	12.638947
FLOOR_GEOMEAN_ALL_OPP_LWIN	0.759282	0.071133	10.674110
OPPONENT_COUNT	0.037877	0.00482	7.857564

Post Flop

Cost To Play

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = NORM_COST_TO_PLAY

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
CONSTANT	0.265197	0.001932	137.274916
N_POT_SIZE	0.048521	0.000883	54.940809
OPPONENT_COUNT	0.505495	0.048803	10.357807
ACTIVE_STAGE_BETRAISE_COUNT	0.348189	0.016653	20.907925
ACTIVE_STAGE_CHECK_COUNT	-0.274952	0.018875	14.566914
ACTIVE_PAST_BETRAISE_COUNT	0.372685	0.021307	17.490790

Pot Size Delta

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = NORM_POT_DELTA

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
N_POT_SIZE	0.314000	0.033273	9.437181
OPPONENT_COUNT	1.608300	0.041939	38.348476
ACTIVE_STAGE_BETRAISE_COUNT	1.080782	0.191566	5.641825
ACTIVE_STAGE_CHECK_COUNT	0.405580	0.085717	4.731596
ACTIVE_PAST_BETRAISE_COUNT	-0.775767	0.087325	8.883692

Tie Probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_TIE

Model Type = Binary Logistic

PREDICTOR	COEFF	S.E.	|COEFF/SE|
FLOOR_GEOMEAN_ALL_OPP_LOGIT_TIE	0.945817	0.109952	8.602096
ACTIVE_STAGE_CALL_COUNT	-0.389042	0.058248	6.679096

Win Probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_WIN

Model Type = Binary Logistic

PREDICTOR	COEFF	S.E.	|COEFF/SE|
IS_FIRST_UP	-0.497617	0.064197	7.751465
ACTIVE_PAST_BETRAISE_COUNT	-0.184747	0.021562	8.568024
ACTIVE_STAGE_CALL_COUNT	0.113489	0.018293	6.203868
SQRT_ACTIVE_STAGE_BETRAISE_COUNT	-0.393892	0.041234	9.552489
FLOOR_GEOMEAN_ALL_OPP_LWIN	0.122973	0.013293	9.251062
LOGIT_FLOOR_HIGH_BOARD_PAIR_WIN	0.635761	0.07018	9.058961

Turn

Cost To Play

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = NORM_COST_TO_PLAY

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
CONSTANT	-4.079826	0.158946	25.668041
N_POT_SIZE	0.026347	0.002476	10.640802
OPPONENT_COUNT	0.387943	0.034575	11.220294
ACTIVE_STAGE_CHECK_COUNT	-0.31745	0.045417	6.989680
ACTIVE_PAST_STAGES_BETRAISE_COUNT	0.213455	0.030059	7.101228
LN[HAND_RATING]	0.301350	0.045063	6.687316

Pot Size Delta

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = NORM_POT_DELTA

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
N_POT_SIZE	0.16092	0.020675	7.783343
OPPONENT_COUNT	1.039897	0.07237	14.369183
ACTIVE_STAGE_BETRAISE_COUNT	1.016252	0.087119	11.665118
ACTIVE_STAGE_CALL_COUNT	-0.867980	0.079689	10.892057
ACTIVE_PAST_STAGES_CALL_COUNT	0.430135	0.062299	6.904339
ACTIVE_STAGE_CHECK_COUNT	-0.665767	0.09481	7.022147
ACTIVE_PAST_STAGES_CHECK_COUNT	-0.581202	0.102594	5.665076
LN[HAND_RATING]	0.069903	0.015033	4.649917

Tie Probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_TIE

Model Type = Binary Logistic

PREDICTOR	COEFF	S.E.	|COEFF/SE|
FLOOR_GEOMEAN_RIVER_LTIE	0.828859	0.049086	16.885924
FLOOR_GEOMEAN_RIVER_LWIN	0.117492	0.006843	17.168966

Win Probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_WIN

Model Type = Binary Logistic

PREDICTOR	COEFF	S.E.	|COEFF/SE|
IS_FIRST_UP	-0.781897	0.135079	5.788443
ACTIVE_PAST_BETRAISE_COUNT	-0.151809	0.014201	10.689828
ACTIVE_STAGE_BETRAISE_COUNT	-0.716681	0.0692	10.356619
LOGIT_FLOOR_HIGH_BOARD_PAIR_WIN	0.635761	0.01937	32.821722
FLOOR_GEOMEAN_RIVER_LWIN	0.117492	0.002139	54.928291
FOLD_COUNT_TOTAL	-0.041658	0.008523	4.887880

River

Cost To Play

RESPONSE = NORM_COST_TO_PLAY

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
CONSTANT	-1.632724	0.011897	137.235829
OPPONENT_COUNT	0.158856	0.02694	5.896608
ACTIVE_PAST_STAGES_BETRAISE_COUNT	0.101966	0.02155	4.731553
LN[HAND_RATING]	0.112281	0.026555	4.228318

Pot Size Delta

RESPONSE = NORM_POT_DELTA

Model Type = Linear

PREDICTOR	COEFF	S.E.	|COEFF/SE|
N_POT_SIZE	0.039404	0.000725	54.381300
OPPONENT_COUNT	0.234011	0.017876	13.090716

Tie probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_TIE

Model Type = Binary Logistic

PREDICTOR	COEFF	S.E.	|COEFF/SE|
CONSTANT	-9.9074068	0.182766	54.208240
FLOOR_GEOMEAN_RIVER_LOGIT_TIE	0.258580	0.020436	12.652955
FLOOR_GEOMEAN_RIVER_LOGIT_WIN	-0.213782	0.025087	8.521717
FLOOR_GEOMEAN_BEST_OPP_LOGIT_TIE	0.129525	0.006291	20.590200
LN[HAND_RATING]	0.589533	0.069863	8.438359

Win probability

Normal distribution used for p-values. Max t DOF = 1000

Number of cases = 4,987,322

RESPONSE = P_WIN

Model Type = Binary Logistic

PREDICTOR	**COEFF**	**S.E.**	**|COEFF/SE|**
CONSTANT	-0.626687	0.007151	87.638316
ACTIVE_STAGE_BETRAISE_COUNT	-0.747666	0.086408	8.652782
ACTIVE_STAGE_CHECK_COUNT	0.615702	0.145243	4.239102
IS_FIRST_UP	-0.513715	0.128792	3.988720
ACTIVE_PAST_STAGES_BETRAISE_COUNT	-0.206485	0.04322	4.777521
LOGIT_SIM_WIN	0.805291	0.06107	13.186343
FOLD_COUNT	-0.067794	0.006792	9.980794

Appendix C

Graphical presentations

See immediately below. Results trajectory for all 50 bots playing over 1000 hands. In all graphs the vertical axis is the number of small bets profit (negatives indicate losses) and the horizontal axis is the number of hands played.

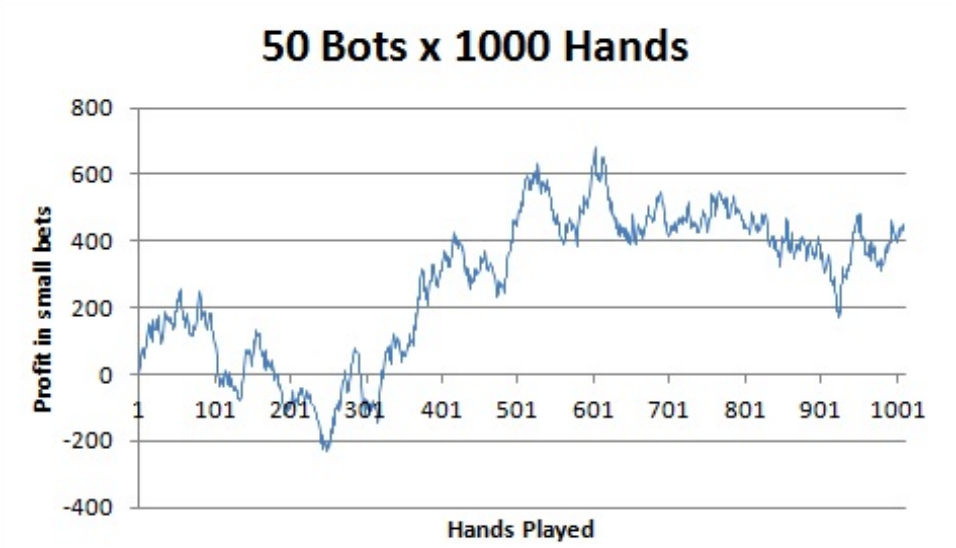


Fig C.0 Results all bots

Groups of 5 bots per graph to aid viewing.

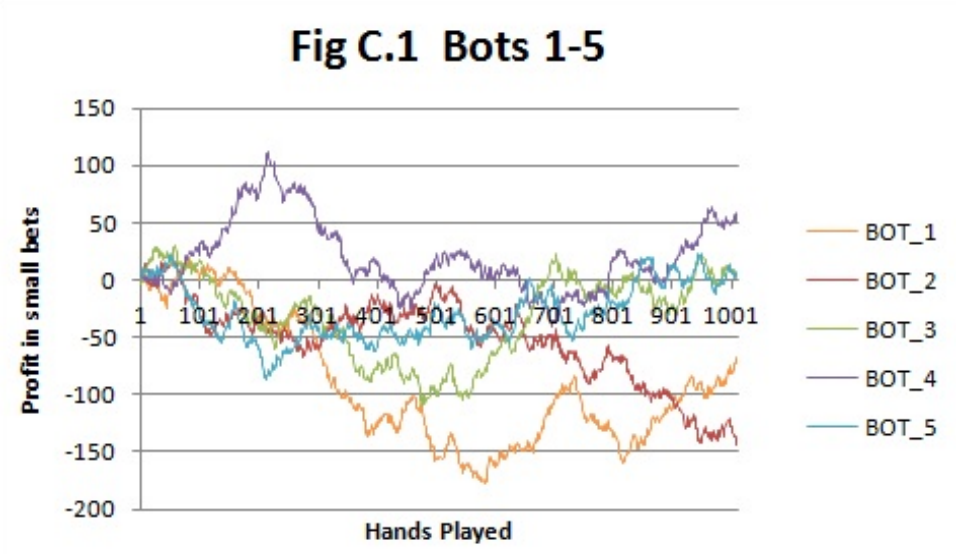


Fig C.2 Bots 6-10

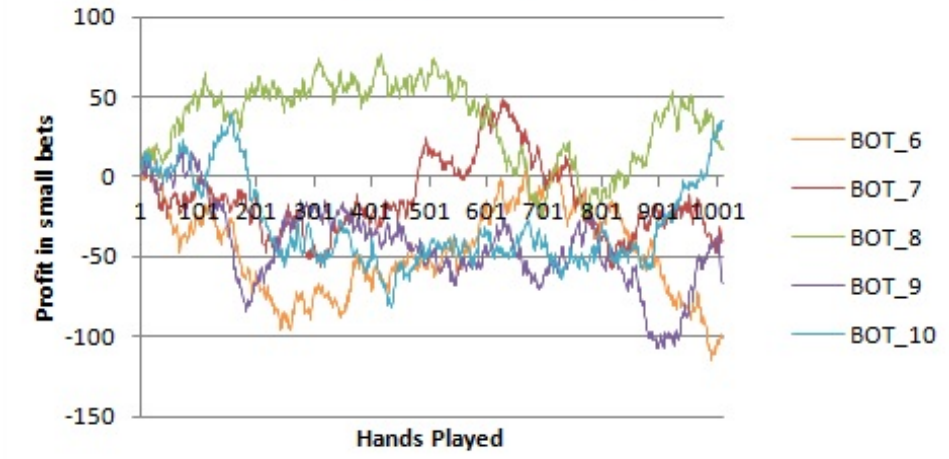
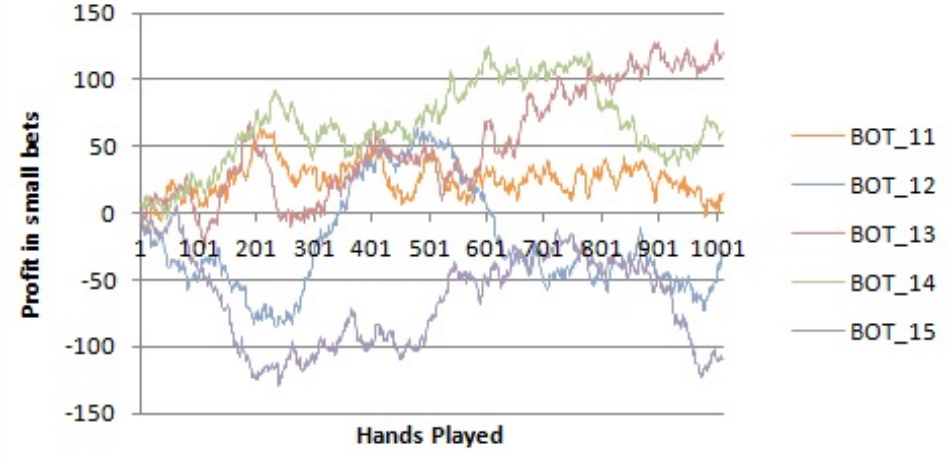


Fig C.3 Bots 11-15



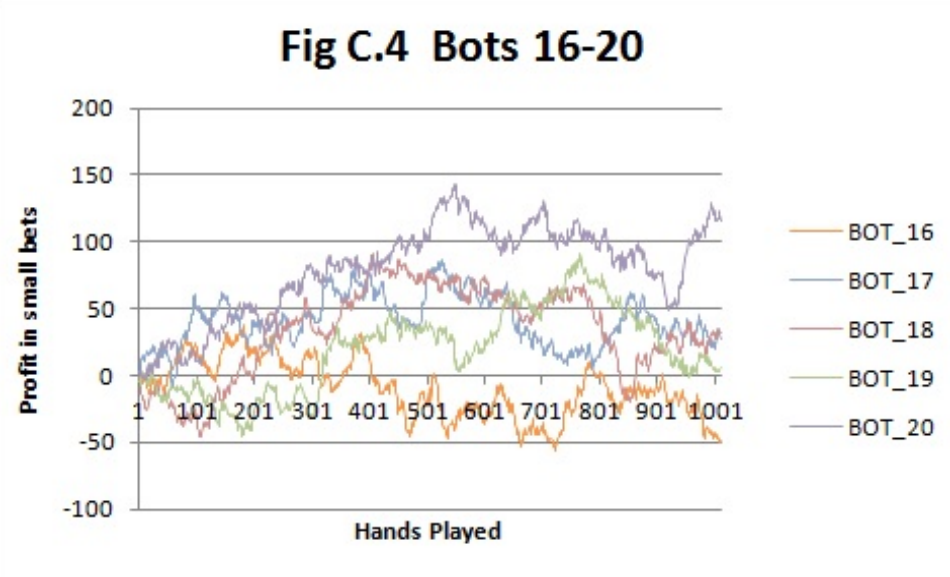


Fig C.4 Bots 16-20
Profit in small bets
200
150
100
50
0
-50
-100
1
101
201
301
401
501
601
701
801
901
1001
Hands Played
BOT_16
BOT_17
BOT_18
BOT_19
BOT_20

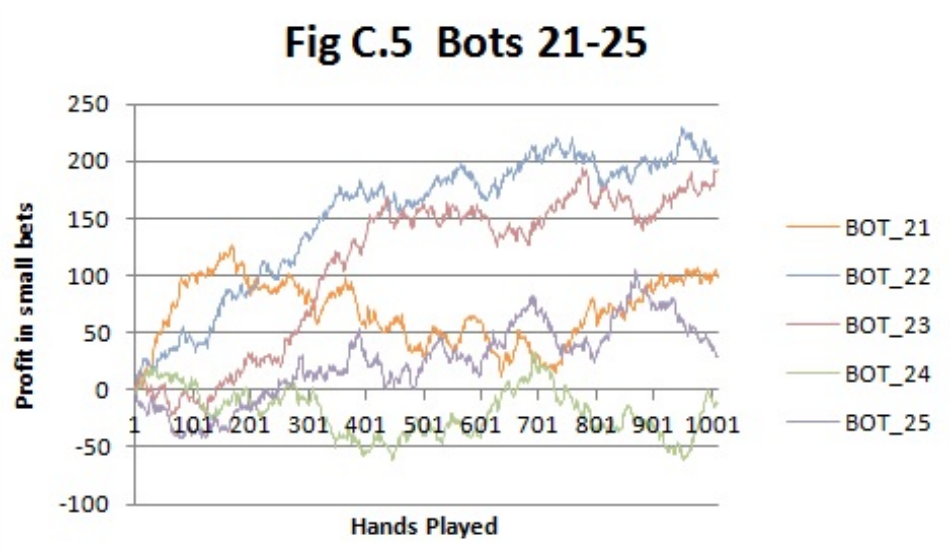


Fig C.5 Bots 21-25
Profit in small bets
250
200
150
100
50
0
-50
-100
1
101
201
301
401
501
601
701
801
901
1001
Hands Played
BOT_21
BOT_22
BOT_23
BOT_24
BOT_25

Fig C.6 Bots 26-30

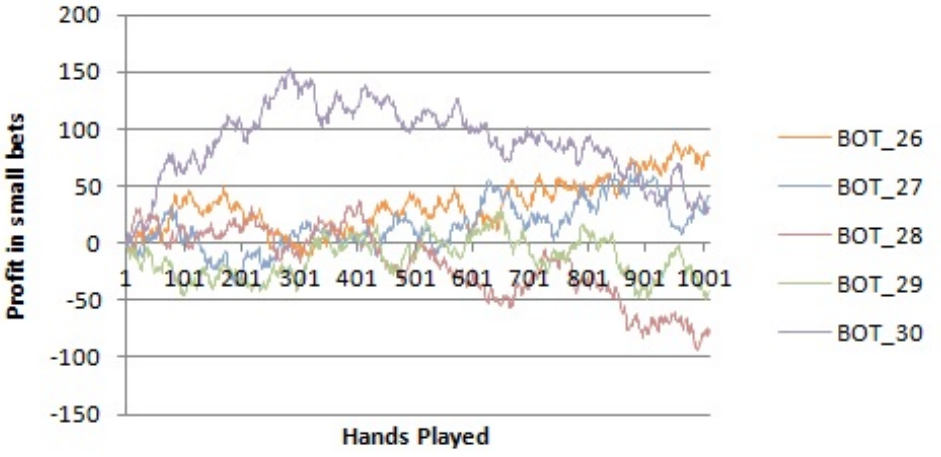
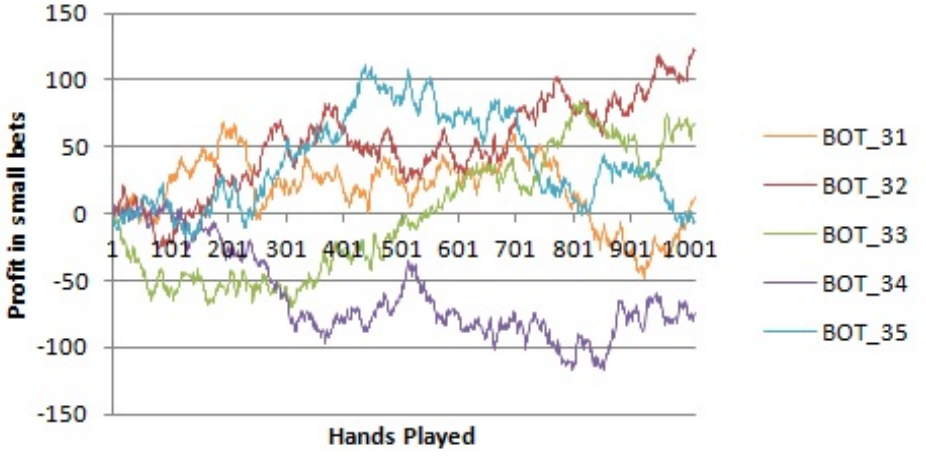


Fig C.7 Bots 31-35



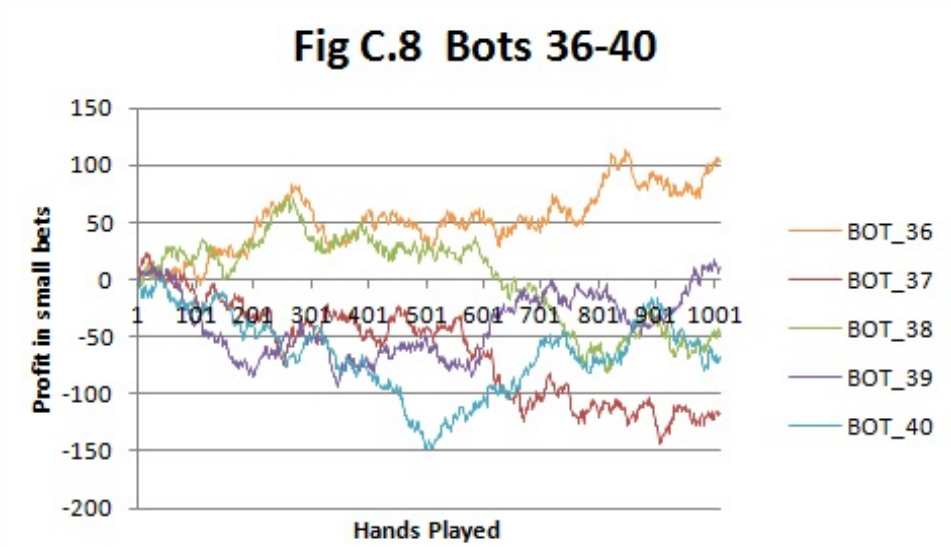


Fig C.8 Bots 36-40
150
100
50
0
-50
-100
-150
-200
1
101
201
301
401
501
601
701
801
901
1001
Profit in small bets
Hands Played
BOT_36
BOT_37
BOT_38
BOT_39
BOT_40

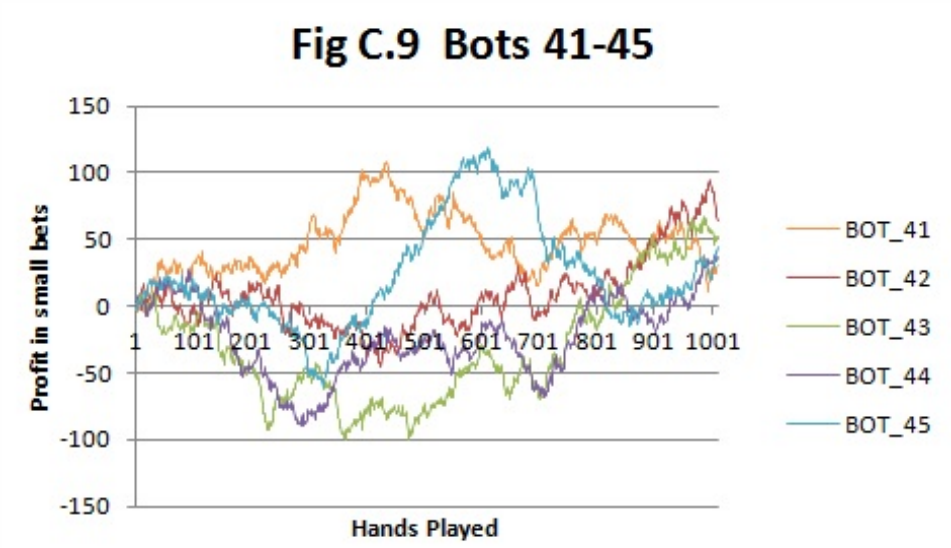


Fig C.9 Bots 41-45
150
100
50
0
-50
-100
-150
1
101
201
301
401
501
601
701
801
901
1001
Profit in small bets
Hands Played
BOT_41
BOT_42
BOT_43
BOT_44
BOT_45

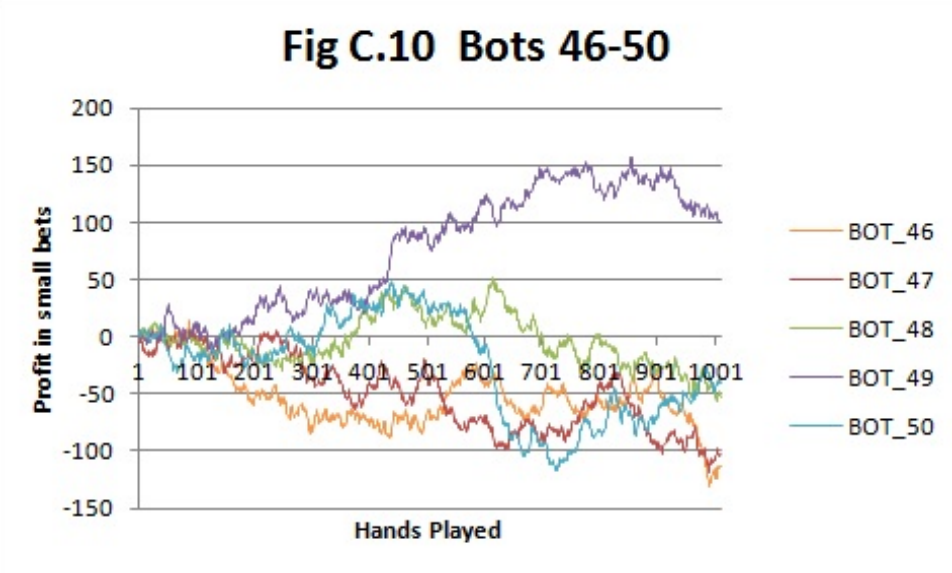


Fig C.10 Bots 46-50
Profit in small bets
200
150
100
50
0
-50
-100
-150
1
101
201
301
401
501
601
701
801
901
1001
Hands Played
BOT_46
BOT_47
BOT_48
BOT_49
BOT_50

Bibliography

[1] Humans set to fold as poker bots raise the stakes. *New Scientist 15 Nov* (2008), 28–29.

[2] Billings, D., Burch, N., Davidson, A., Holte, R., Schaeffer, J., Schauenberg, T., and Szafron, D. Approximating game-theoretic optimal strategies for full-scale poker. *Internal publication, University of Alberta* (2002).

[3] Billings, D., Davidson, A., Schaeffer, J., and Szafron, D. The challenge of poker. *Artificial Intelligence 134* (2002), 201–240.

[4] Borel, E. *Applications aux Jeux des Hazard*. unknown, 1938.

[5] Breiman, Friedman, Olshen, and Stone. *Classification and Regression Trees*. Chapman and Hall, 1984.

[6] Cameron, A. C., and Trivedi, P. K. *Regression Analysis of Count Data*, 1 ed. Cambridge University Press, 1998.

[7] Cook, R. D., and Weisberg, S. *Applied Regression Including Computing and Graphics*, 1 ed. John Wiley and Sons, 1999.

[8] Cristianini, N., and Shawe-Taylor, J. *An Introduction to Support Vector Machines and other kernel learning methods.* Cambridge University Press, 2003.

[9] Davidson, A., Billings, D., Schaeffer, J., and Szafron, D. Improved opponent modelling in poker. In *Proc. International Conference on Artificial Intelligence* (Las Vegas, NV, 2000), pp. 1467–1473.

[10] Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. *Bayesian Data Analysis*, 2 ed. Chapman Hall, 2004.

[11] Harlan, M. T. R., and Derossi, C. *Winning at Internet Poker For Dummies.* Wiley Publishing, 2005.

[12] Hosmer, D. W., and Lemeshow, S. *Applied Logistic Regression*, 2 ed. John Wiley and Sons, 2000.

[13] Kelly, J. A new interpretation of information rate. *Bell System Technical Journal 35* (1956), 917–926.

[14] McCullagh, P., and Nelder, J. A. *Generalized Linear Models*, 2 ed. Chapman Hall, 1989.

[15] Miller, E., Sklansky, D., and Malmuth, M. *Small Stakes Hold'em.* Creel Printers, 2005.

[16] Neiderreiter, H. Quasi monte carlo methods and pseudo-random numbers. *Bulletin of the American Mathematical Society 84* (1978), 957–1041.

[17] Neiderreiter, H. Low-discrepancy and low dispersion sequences. *Journal of Number Theory 30* (1988), 51–70.

[18] Papp, D. *Dealing with imperfect information in poker*. PhD thesis, University of Alberta, Edmonton, 1998.

[19] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. *Numerical Recipes in C*, 2 ed. Cambridge University Press, 1992.

[20] Risk, N. A. *Using Counterfactual Regret Minimisation To Create A Competitive Multiplayer Poker Agent*. PhD thesis, University of Alberta, Edmonton, 2009.

[21] Sakaguchi, and Sakai. Partial information in simplified two person poker. *Math. Japonica 26* (1981), 695–705.

[22] Sklansky, D. *Small Stakes Hold'em*. unkown, 1999.

[23] Sklansky, D. *Theory of Poker*, 4 ed. Two Plus Two Publishing LLC, 2007.

[24] Skrondal, A., and Rabe-Hesketh, S. *Generalised Latent Variable Modelling*. Chapman and Hall CRC, 2004.

[25] THORPE, E. O. *Beat The Dealer*. Vintage Books, 1966.

[26] TRAIN, K. E. *Discrete Choice Methods with Simulation*, 1 ed. Cambridge University Press, 2003.

[27] VON NEUMANN, AND MOGENSTERN. *Theory of Games and Economic Behavior*. Princeton UP, 1944.