Behavior Based Algorithmic Trading Strategy Identification

A Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

in partial fulfillment

of the requirements for the degree

Doctor of Philosophy

by

Steve Y. Yang

May

2012

APPROVAL SHEET

The dissertation
is submitted in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

AUTHOR

The dissertation has been read and approved by the examining committee:

Co-Advisor

Co-Advisor

George Overstreet

Accepted for the School of Engineering and Applied Science:

James H. Aylor

Dean, School of Engineering and Applied Science

May
2012

Behavior Based Algorithmic Trading Strategy Identification

by

Steve Y. Yang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(School of Engineering and Applied Science)
in The University of Virginia
2012

Doctoral Committee:

Assistant Professor Randy Cogill, Chair
Professor William Scherer, Co-advisor
Associate Professor Peter Beling, Co-advisor
Associate Professor Michael Smith
Dr. Andrei Kirilenko
Professor George A. Overstreet Jr.

”Beware the barrenness of a busy life.” - Socrates

When I applied for Ph.D. program, I applied only one school; When I applied faculty position, again I applied only one school; In both cases I got what I asked for. As much as my past two and half years have gone against the statistical odds, there is no excuse for me not to attribute what have accomplished to God first. I also want to dedicate my dissertation work to my family and many of my friends who have supported me through out the process. Their kind words and encouragement have kept me going until this day. A special feeling of gratitude to my parents Qian Yang and Peilan Chen whose words of encouragement and push for tenacity ring in my ears all through my life. I cannot say enough how much I have wished my mother have lived until this day to see my receiving my doctoral degree. I also want to dedicate this work to my lovely daughter Joyce. I wish that she will one day read my work, and know the days and nights that I missed her. I always wanted her to know that how much her dad loved her and wanted to be at her side.

ACKNOWLEDGEMENTS

I would like to thank my advisors, William Scherer, Peter Beling, Randy Cogill, and Michael Smith, for the considerable amount of time they have spent discussing this topic with me; I also want to give my extra gratitude to Dr. Andrei Kirilenko at the Commodity Futures Trading Commission who has provided me great research opportunities when I work as a research assistant at the Office of Chief Economist.

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF APPENDICES

Appendix

ABSTRACT

Behavior Based Algorithmic Trading Strategy Identification

by

Steve Y. Yang

Electronic markets have emerged as popular venues for the trading of a wide variety of financial assets, and computer based algorithmic trading has also asserted itself as a dominant force in financial markets across the world. Identifying and understanding the impact of algorithmic trading on financial markets has become a critical issue for market operators and regulators. We propose to characterize traders' behavior in terms of the reward functions most likely to have given rise to the observed trading actions. Our approach is to model trading decisions as a Markov Decision Process (MDP), and use observations of an optimal decision policy to find the reward function. This is known as Inverse Reinforcement Learning (IRL), and a variety of approaches for this problem are known. Our IRL-based approach to characterizing trader behavior strikes a balance between two desirable features in that it captures key empirical properties of order book dynamics and yet remains computationally tractable. Using an IRL algorithm based on linear programming, we are able to achieve more than 90% classification accuracy in distinguishing High Frequency Trading from other trading strategies in experiments on a simulated E-Mini S&P 500 futures market.

Furthermore we investigate and address incomplete observation and non-deterministic police issues related to real market observations. We develop models based on Gaussian Process Inverse Reinforcement Learning as well. The primary objective of this study is to model Algorithmic trading behavior using Bayesian inference under the framework of inverse reinforcement learning (IRL). We model trader's behavior as a Gaussian process in the reward space. With incomplete observations of different market participants, we aim to recover the optimal policies and the corresponding reward functions to explain their behaviors under different circumstances. We show that Algorithmic trading behavior can be accurately identified using Gaussian Process Inverse Reinforcement Learning (GPIRL) algorithm developed by Qiao and Beling (*Qiao and Beling* [2011]), and it is superior to the linear features maximization approach. Real market data experiments using GPIRL model give more than 95% trader identification accuracy consistently using support vector machines (SVM) based classification method. We also show that there is a clear connection between the existing summary statistic based trader classification (*Kirilenko et al.* [2011]) and our behavior based classification. In order to address potential change of trading behavior over time, we propose a score based classification approach to address variations of Algorithmic trading behavior under different market conditions. We further conjecture that because our behavior based identification is a better reflection of traders' choice of actions and value propositions under different market conditions than the summary statistic based method, it is therefore more informative and robust than the summary statistic based approach, and it is well suited for discovering new behavior patterns of market participants.

Overall, we prove the hypothesis that that Algorithmic Trading strategies can be accurately identified using behavior based modeling techniques under the Inverse Reinforcement Learning framework and these strategies can be profiled based on observations of individual trading actions for market surveillance and other economic researches regarding the impact of different Algorithmic Trading strategies to financial

market quality in general.

CHAPTER I

Introduction

Electronic markets have emerged as popular venues for the trading of a wide variety of financial assets, such as stocks, commodities, options and futures, etc. Many such electronic markets are organized as electronic limit order books. Through reduction in the frictions and costs of trading, electronic trading has the potential to enable more efficient risk sharing, facilitate hedging, improve liquidity, and make prices more efficient (*Hasbrouck* [2007], *Mike and Farmer* [2008], *Potters and Wyart* [2004], *Kockelkoren and Potters* [2006], *Lyons* [2006], *Farmer and Lillo* [2009], and *Hasbrouchk and Seppi* [2001a]). Ultimately, these benefits have the potential to lead to a reduction in the cost of capital for firms.

Many market participants now employ algorithmic trading, commonly defined as the use of computer algorithms to automatically make certain trading decisions, submit orders, and manage those orders after submission. By the time of the "Flash Crash (On May 6, 2010 during 25 minutes, stock index futures, options, and exchange-traded funds experienced a sudden price drop of more than 5 percent, followed by a rapid and near complete rebound.), algorithmic trading was thought to be responsible for more than 70% of trading volume in the U.S. *Brogaard* [2010]. Moreover, Kirilenko et al. (*Kirilenko et al.* [2011]) have shown that the key events in the Flash Crash have a clear interpretation in terms of algorithmic trading.

The rise of algorithmic trading has obvious broad and direct impacts on the financial markets. For example, the intense activity generated by algorithms threatens to overwhelm exchanges and market data providers, forcing significant upgrades to data management infrastructure. Researchers, regulators, and policymakers should be keenly interested in the broader implications of this sea change in trading. Important research and policy issues focus on the nature of the impact that algorithmic trading has on the markets and ways in which this impact can be shaped through regulation (*Brogaard* [2010], *Jones and Menkveld* [2011]). Important questions include: Should algorithmic trading be encouraged because it provides beneficial liquidity? Should regulation be used to limit the speed advantages that certain algorithmic trading firms enjoy?

Many machine learning techniques have been applied in financial market analysis and modeling to assist economists, policy makers and regulators to understand the behaviors of the market participants, market dynamics, and the price discovery process of the new electronic market phenomena algorithmic trading. We are particularly interested in modeling traders behavior as a Markov Decision Process (*Puterman* [1994]), and use the observations obtained in the past to infer traders trading strategies. More specifically, we aim to learn traders reward function in the context of multi-agent environments where agents/traders competing fast algorithmic trading strategies to explore and exploit the market microstructure within the preset market trading rules to maximize their profits.

Our proposed approach is based on the machine learning technique (*Schaeffer and Szafron* [1998], *Russell* [1998], *Sutton and Barto* [1998], *Barto and Williams* [1991], *Ng and Russel* [2000], *Abbeel and Ng* [2004], *Ramachandran and Amir* [2007], and *Bagnell and Dey* [2008]) known as Inverse Reinforcement Learning (IRL) (*Ng and Russel* [2000], *Abbeel and Ng* [2004]). In IRL, one aim is to estimate and infer the model that underlies solutions that have been chosen by decision makers. This model might then later be used to control an autonomous process that is governed by a model with the desirable characteristics that are implied by the observed solutions. For example, Pokerbots can improve performance against suboptimal human opponents by learning reward functions that account for the utility of money, preferences for certain hands or situations and other idiosyncrasies (*Schaeffer and Szafron* [1998]). Another objective in IRL is to use observations of the agents/traders actions to decide ones own behaviors. It is possible in this case to directly learn the policy from the past observations. The premise of this reward learning (IRL) is generally the most succinct, robust and transferable of the task, and completely determines the optimal policy (or a set of policies). It provides a better metric to measure agents/traders behaviors.

One of the important goals of learning traders trading strategies is to be able to identify and categorize the market participants, and be able to further understand their influences related to such important economic issues as multiple characterizations of price formation processes, market liquidity, and order flow, etc (*Hasbrouchk and Seppi* [2001a], *Gabaix and Gopikrishnan* [2004], *Gatheral* [2010], *Hasbrouck* [1991], *Jones et al.* [1994], *Karpoff* [2004]). We assert that enhanced understanding of the economic implication of these different algorithmic trading strategies will yield quantitative evidence of value to market policy makers and regulators seeking to maintain transparency, fairness and overall health in the financial markets.

The rest of the dissertation is as follows. In Chapter 2, we introduce and define the concept of behavior based Algorithmic trading stragety identification using Markov Decision Process and Inverse Reinforcement Learning. We postulate that Algorithmic trading behavior can be identified when we impose stationarity and rational expection constraints on non-experimenal observations under a stochastic control process framework. Chapter 3 studies order book events and their price impact using an order flow imbalance model through an empirical study of several futures markets.

We analyze a unique market dataset where an exchange accidentally injected artificial orders into several markets and caused numerous errant trades during a very short period of time . We use this as a natural experiment, and compare these concentrated trading activities with the normal markets in terms of volatility and price impact based on an order flow imbalance model (*Kukanov and Stoikov* [2011]). We first analyze volatility variation across different markets through this incident, and then we postulate an order flow model of price impact and show that order flow events of this incident significantly changed the market volatility and moved the market prices. This empirical evidence then corroborates an MDP model that we propose later for trader behavior modeling. In Chapter 4, we propose to characterize traders behavior in terms of the reward functions most likely to have given rise to the observed trading actions. Our approach is to model trading decisions as a Markov Decision Process (MDP), and use observations of an optimal decision policy to find the reward function. This is known as Inverse Reinforcement Learning (IRL). Our IRL-based approach to characterizing trader behavior strikes a balance between two desirable features in that it captures key empirical properties of order book dynamics and yet remains computationally tractable. Using an IRL algorithm based on linear programming, we are able to achieve more than 90% classification accuracy in distinguishing high frequency trading from other trading strategies in experiments on a simulated E-Mini S&P 500 futures market. The results of these empirical tests suggest that high frequency trading strategies can be accurately identified and profiled based on observations of individual trading actions. In Chapter 5, we propose a non-parametric Bayesian model using Gaussian process and preference graph theory. Using this model, we address incomplete observation and errant observation issues in the inverse reinforcement learning process. This approach only requires a finite number of observations that is much less stringent than the approaches based on feature expectations or value functions. It also presents itself as a robust framework to obtain computational efficiency for practical problems due to its convexity in the optimization process. We further prove that this Gaussian Process based IRL approach is a better way of modeling Algorithmic trading behaviors than the linear IRL approach we proposed in Chapter 4 due to its specific features in addressing market uncertainty and incomplete information. We apply the model on a month of actual E-Mini S&P 500 futures market data, and show that the identification rate has been greatly improved over the linear approach. Lastly, we summarize our general conclusions from this study in Chapter 6, and address some the critical issues we encounter in modeling Algorithmic trading strategies and point out some future research directions.

CHAPTER II

Problem Definitions

2.1 Introduction to Behavior Modeling

Markov Decision Processes (MDP) provide a broad framework for modeling sequencial decision making under uncertainty. MDP's have two sorts of variables: state variables s_t and control variables d_t, both of which are indexed by time $t = 0, 1, 2, 3...., T$, where the horizon T may be infinity. A decision-maker or agent can be represented by a set of primitives (r, p, β) where $r(s_t, d_t)$ is a reward function representing the agent's preferences at time t, $p(s_{t+1}|s_t, d)$ is a Markov transition probability representing the agent's subjective beliefs about uncertain future states, and $\beta \in (0, 1)$ is the rate at which the agent discounts reward in future periods. Agents are assumed to be rational: they behave according to an optimal decision rule $d_t = \delta(s_t)$ that solves $V_0^T(s) \equiv \max_\delta E_\delta \sum_{t=0}^{T} \beta^t r(s_t, d_t)|s_0 = s$ where E_δ denotes expectation with respect to the controlled stochastic process s_t, d_t induced by the decision rule δ.

MDPs have been extensively used in theoretical studies because the framework is rich enough to model most economic problems involving choices made over time and under uncertainty. Applications include the pioneering work on optimal inventory policy by Arrow et al. (1951), investment under uncertainty [Lucas and Prescott (1971)] optimal intertemporal consumption/savings and portfolio selection under uncertainty [Phelps (1962), Hakansson (1970), Levhari and Srinivasan (1969), Merton (1969) and Samuelson (1969)], optimal growth under uncertainty [-Brock and Mirman (1972), Leland (1974)], models of asset pricing [Lucas (1978), Brock (1982)], and models of equilibrium business cycles [Kydland and Prescott (1982), Long and Plosser (1983)]. By the early 1980's the use of MDP's had become widespread in both micro- and macroeconomic theory as well as in finance and operations research. The first to develop this type of discrete decision model was John Rust (*Rust* [1987]), and provided a more complete description of the DP approach in Rust (*Rust* [1995a], and *Rust* [1995b]). Some applied economists have used various similar methods in explaining various economic behaviors (*Miranda and Schnitkey* [1995], *Baerenklau and Provencher* [2005], and *Hendel and Aviv* [2006]).

In addition to providing a normative theory of how rational agents "should" behave, econometricians soon realized that MDP's might provide good empirical models of how real-world decision-makers actually behave. Most data sets take the form d_t^a, s_t^a where d_t^a is the decision and s_t^a is the state of an agent a at time t. Stochastic control theory can also be used to model "learning" behavior in which agents update beliefs about unobserved state variables and unknown parameters of the transition probabilities according to the Bayes rule. Reduced-form estimation methods can be viewed as uncovering agents' decision rules or, more generally, the stochastic process from which the realizations (d_t, s_t) were "drawn", but are generally independent of any particular behavioral theory. Our focus is on uncovering (estimating) the primitives (r, p, β) that generated it under the hypothesis that d_t^a, s_t^a is a realization of a controlled stochastic process.

The first question to be answered is whether this is even logically possible, i.e. whether (r, p, β) is identified. Rust [*Rust* [1997]] discussed the identification problem, and showed that the question of identification depends on what type of data we have access to (i.e. experimental vs. non-experimental), and what kinds of a priori restrictions we are willing to impose on (r, p, β). If we only have access to non-experimental data (i.e. uncontrolled observations of agents "in the wild"), and if we are unwilling to impose any prior restrictions on (r, p, β) beyond basic measurability and regularity conditions on r and p, then it is impossible to consistently estimate (r, p, β), i.e. the class of all MDP's is non-parametrically unidentified. On the other hand, if we are willing to restrict r and p to a finite-dimensional parametric family, say $r = r_\theta, p = p_{\theta,}|\theta \in \Theta \subseteq R^K$, then the primitives (r, p, β) are identified (generically). If we are willing to impose an even stronger prior restriction, stationarity and Rational Expectations (RE), then we only need parametric restrictions on r in order to identify (r, p, β) since stationarity and the RE hypothesis allow us to use non-parametric methods to consistently estimate agents' subjective beliefs from observations of their past states and decisions. Given that we are already imposing strong prior assumptions by modelling agents' behavior as an optimal decision rule to an MDP, it would be somewhat schizophrenic to be unwilling to impose any additional prior restrictions on (r, p, β). In the sequel, I assume that the econometrician is willing to bring to bear prior knowledge in the form of a parametric representation for (r, p, fl). This reduces the problem of structural estimation to the technical issue of estimating a parameter vector $\theta \in \Theta$ where Θ is a compact subset of R^K.

Furthermore, Rust (*Rust* [1997]) showed that, from an econometric standpoint, the expected-reward framework is sufficiently rich to model virtually any type of observed behavior. Our ability to discriminate between expected reward and the more subtle non-expected-reward theories of choice under uncertainty may require quasi-econometric methods such as controlled experiments. The justification for focusing on expected reward is that it remains the most tractable framework for modelling choice under uncertainty.

Ng and Russel [2000] formulate IRL problem as an optimization problem to maximize the sum of differences between the quality of the optimal action and the quality of the next-best action. Based on this linear approximation of reward function, other algorithms have been developed or integrated into apprenticeship learning. The principal idea of apprenticeship learning using IRL is to search mixed solutions in a space of learned policies with the goal that the accumulative feature expectation is near that of the expert *Abbeel and Ng* [2004], *Bowling and Schapire* [2008]. Recent theoretical works on IRL improve the learning performance through various other methods, such as the framework of linear-solvable MDP in *Dvijotham and Todorov* [2010], the bootstrap learning in *Boularias and Chaib-draa* [2010] and feature construction in *Popovic and Koltun* [2010]. IRL has also been successfully applied to many real-world problems, such as automatic control of helicopter flight *Coates and Ng* [2010] and motion control of an animation system in computer graphics *Lee and Zoran* [2010]. In *Baker et al.* [2009], IRL is viewed from the perspective of human decision making as a method for modeling human action understanding, and the results of psychophysical experiments using animated stimuli of agents moving in simple masses provide quantitative evidence that the inverse planning models can predict human goal function.

Based on the existing work on modeling economic behaviors and recent development in reward learning, we aim to develop a framework to quantify Algorithmic trading behavior by solving an inverse Markov decision process. We first try to understand the financial microstructure through an emperical study. Through literature review and our own market modeling effort, we establish empirical basis for describing financial market using MDP models. We then develop a model and cast the behavior learning problems as the reward learning through linear approximation, which underlies a number of IRL approaches. And then we explore the non-parametric modeling techniques and postulate a prior distribution over the variables we wish to predict and consider them in a random field. By applying Bayesian rule, we employ an Gaussian based approach (**?**) to predict traders' behavior with the assumption that we have only incomplete observations. We prove our IRL based behavior framework is able to consistently predict future behavior of incomplete recent observations.

2.2 Background and Related Work

The purpose of this section is to introduce the notations that will be used throughout this study and lay out the existing theories that we need to quantify trading behavior under the Stochastic modeling framework - Markov Decision Processes.

2.2.1 Inverse Markov Decision Process (IMDP)

The input data to IRL is collected from the observations of an expert whose decision process is modeled as Markov decision processes. We restrict our attention to countable MDP here for easy exposition, but our algorithms can be extended to the MDP problems in continuous domains. A discounted finite MDP is defined as a tuple $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, where

- $\mathcal{S} = \{s_n\}_{n=1}^{N}$ is a set of N states. Let $\mathcal{N} = \{1, 2, \cdots, N\}$.
- $\mathcal{A} = \{a_m\}_{m=1}^{M}$ is a set of M actions. Let $\mathcal{M} = \{1, 2, \cdots, M\}$.
- $\mathcal{P} = \{\mathbf{P}_{a_m}\}_{m=1}^{M}$ is a set of state transition probabilities (Here $\mathbf{P}_{a_m}$ is a $N \times N$ matrix. Each row, denoted as $\mathbf{P}_{a_m}(s_n, :)$, contains the transition probabilities upon taking action a_m in state s_n. The entry $\mathbf{P}_{a_m}(s_n, s_{n'})$ is the probability of moving to state $s_{n'}, n' \in \mathcal{N}$ in the next stage.).
- $\gamma \in [0, 1]$ is a discount factor.
- r denotes the reward function, mapping from $\mathcal{S} \times \mathcal{A}$ to $\Re$ with the property that

$$r(s_n, a_m) \triangleq \sum_{n' \in \mathcal{N}} \mathbf{P}_{a_m}(s_n, s_{n'}) r(s_n, a_m, s_{n'})$$

where $r(s_n, a_m, s)$ denotes the function giving the reward of moving to next state $s_{n'}$ after taking action a_m in current state s_n. The reward function $r(s_n, a_m)$ may be further reduced to $r(s_n)$, if we neglect the action's influence.

In MDP, an agent selects an action at each sequential stage. A rule describing the way the actions are selected is called a *policy* (*behavior*), some mapping between state and action. A behavior of an agent defines a random state-action sequence $(s^0, a^0, s^1, a^1, \cdots s^t, a^t, \cdots)$, [1] where s^{t+1} is connected to (s^t, a^t) by $\mathbf{P}_{a^t}(s^t, s^{t+1})$. The policy, which makes the agent reach the goal, is called *proper policy*.

The rational agents in MDP model behave according to the optimal decision rule that each action selected at any stage should maximize the value function. The *value function* for a policy π evaluated at any state s^0 is given as $V^\pi(s^0) = E[\sum_{t=0}^{\infty} \gamma^t r(s^t, a^t)|\pi]$. This expectation is over the distribution of the state sequence $\{s^0, s^1, ...\}$ given policy $\pi = \{\mu^0, \mu^1, \cdots\}$, where $a^t = \mu^t(s^t)$, $\mu^t(s^t) \in U(s^t)$ and $U(s^t) \subset \mathcal{A}$. The objective at state s is to choose a policy maximizing the value of $V^\pi(s)$. Similarly, there is another function called *Q-functions* (*Q-factors*) that judges how good an action is performed in a given state. Notation $Q^\pi(s, a)$ represents the expected return from state s, taking action a and thereafter following policy π.

Some essential facts that we will need from the theory of MDPs *R.* [1957] are listed as follows,

Theorem II.1 (Bellman Equations)**.** *Given a stationary policy π, $\forall n \in \mathcal{N}, m \in \mathcal{M}$, $V^\pi(s_n)$ and $Q^\pi(s_n, a_m)$ satisfy*

$$\begin{aligned} V^\pi(s_n) &= r(s_n, \pi(s_n)) + \gamma \sum_{n' \in \mathcal{N}} \mathbf{P}_{\pi(s_n)}(s_n, s_{n'}) V^\pi(s_{n'}), \\ Q^\pi(s_n, a_m) &= r(s_n, a_m) + \gamma \sum_{n' \in \mathcal{N}} \mathbf{P}_{a_m}(s_n, s_{n'}) V^\pi(s_{n'}). \end{aligned}$$

[1] Superscripts index time. E.g. s^t and a^t, with the upper-index $t \in \{1, 2, \cdots\}$, denote state and action at t-th horizon stage, while s_n (or a_m) represents the n-th state (or m-th action) in $\mathcal{S}$ (or $\mathcal{A}$).

Lemma II.2. *The optimal value functions and Q-functions are defined as,*

$$\begin{aligned} V^*(s) &= \sup_a Q^*(s,a) \\ Q^*(s,a) &= r(s,a) + \gamma \sum_{s'} P_{a,s}(s')V^*(s') \end{aligned}$$

Theorem II.3 (Bellman Optimality). *π is optimal if and only if, $\forall n \in \mathcal{N}$, $\pi(s_n) \in \arg\max_{a \in \mathcal{A}} Q^\pi(s,a)$.*

Based on the above definitions of MDP, let us define the *inverse Markov Decision Process (IMDP)*.

Definition II.4. An IMDP model, denoted as $M_I = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{O})$, contains MDP variables such as, state set $\mathcal{S}$, action set $\mathcal{A}$, state transition probability set $\mathcal{P}$ and the discount factor γ. The variable $\mathcal{O}$ is a set of observations sampled from the decision-making process.

Remark II.5. The set $\mathcal{O}$ can be viewed as a subset of the Cartesian product of $\hat{\mathcal{S}}$ and $\hat{\mathcal{A}}$, where $\hat{\mathcal{S}} \subset \mathcal{S}$ and $\hat{\mathcal{A}} \subset \mathcal{A}$. So $\forall s \in \hat{\mathcal{S}}$, there is at least one action $a \in \hat{\mathcal{A}}$ providing $(s,a) \in \mathcal{O}$. We treat every $(s,a) \in \mathcal{O}$ as optimal in the expert's decision making process. The goal of IRL is to learn the reward function of the MDP model that generates $\mathcal{O}$.

2.2.2 Inverse Reinforcement Learning (IRL)

The primary objective of IRL is to determine the reward function that an agent is optimizing to achieve its optimal decision rules. It is formulated via MDP model by Ng and Russel in *Ng and Russel* [2000], under the assumption that the reward $r = \omega^T \phi(s)$, where ω is a coefficient vector and $\phi(s) : s \rightarrow [0,1]^k$ is a function mapping state s to a k dimensional vector. Under this formulation IRL is casted as an optimization problem to maximize the sum of differences between the quality of the optimal action and the quality of the next-best action. Based on this linear approximation of reward function, other algorithms have been developed or integrated into apprenticeship learning. The principal idea of apprenticeship learning using IRL is to search mixed solutions in a space of learned policies with the goal that the accumulative feature expectation is near that of the expert *Abbeel and Ng* [2004], *Bowling and Schapire* [2008]. A game-theoretic approach to apprenticeship learning using IRL is developed in the context of a two-player zero-sum game in which the apprentice chooses a policy and the environment chooses a reward function *Schapire* [2008]. Another algorithm for IRL is policy matching in which the loss function penalizing deviations from expert's policy is minimized by tuning the parameters of reward functions *Neu and Szepesvari* [2007]. Maximum entropy IRL is proposed in the context of modeling real-world navigation and driving behaviors in **?**. The algorithms for apprenticeship learning using IRL do not actually aim to recover the reward function but only the optimal policy.

Recent theoretical works on IRL improve the learning performance through various methods, such as the framework of linear-solvable MDP in *Dvijotham and Todorov* [2010], the bootstrap learning in *Boularias and Chaib-draa* [2010] and feature construction in *Popovic and Koltun* [2010]. IRL has also been successfully applied to many real-world problems, such as automatic control of helicopter flight *Coates and Ng* [2010] and motion control of an animation system in computer graphics *Lee and Zoran* [2010]. In *Baker et al.* [2009], IRL is viewed from the perspective of human decision making as a method for modeling human action understanding, and the results of psychophysical experiments using animated stimuli of agents moving in simple masses provide quantitative evidence that the inverse planning models can predict human goal function.

The assumption that the reward function can be linearly approximated, which underlies a number of IRL approaches, may not be reasonable for many problems of practical interest. The failure to satisfy this assumption may result in divergent algorithms. At this point, the non-parametric modeling can be more powerful than the parametric method. In this study we explore two different approaches in estimating the reward function, and use reward functions to characterize agents' behaviors.

2.2.3 Financial Algorithmic Trading

Algorithmic trading (AT) is commonly defined as the use of computer algorithms to automatically make certain trading decisions, submit orders, and manage those orders after submission (*Jones and Menkveld* [2011]). There are many different algorithms, used by many different types of market participants. Some hedge funds and broker-dealers supply liquidity using algorithms, competing with designated market-makers and other liquidity suppliers (e.g., *Jovanovic and Menkveld* [2010]). For assets that trade on multiple venues, liquidity demanders often use smart order routers to determine where to send an order (e.g., *Foucault and Menkveld* [2008]). Statistical arbitrage funds use computers to quickly process large amounts of information contained in the order flow and price moves in various securities, trading at high frequency based on patterns in the data. Last but not least, algorithms are used by institutional investors to trade large quantities of stock gradually over time. High Frequency Trading (HFT) refers to a super-fast algorithmic trading strategy where a trader moves in and out of stocks with extremely short holding intervals in an attempt to capture small profits per trade. In 2010, the Securities and Exchange Commission (SEC) published a concept release paper regarding current financial market conditions in which the Commission describe the HFTs with the following key characteristics: (1) the use of extraordinarily high-speed and sophisticated computer programs for generating, routing, and executing orders; (2) use of co-location services and individual data feeds offered by exchanges and others to minimize network and other types of latencies; (3) very short time frames for establishing and liquidating positions; (4)

the submission of numerous orders that are cancelled shortly after submission; and (5) ending the trading day in as close to a flat position as possible. Estimate of HFT volume in the equity markets vary widely, though they typically are 50% or higher [2].

However there are only few academic papers address questions regarding HFT. The theoretical work relating to HFT is mostly devoted to address its economic implication to the financial market quality. However it is still too earlier to draw any decisive conclusions: some show that, depending on the model, HFTrs may improve market quality while still others claim HFTs degrade market characteristics. *Cvitanic and Kirilenko* [2010] build the first theoretical model to address how HFTrs impact market conditions. They find that the presence of HFTrs yield transaction prices that differ from the HFTr-free price; when a HFTr is present, the distribution of transaction prices will have thinner tails and more mass near the mean; and as humans increase their order submissions, liquidity proportionally increases. While *Cvitanic and Kirilenko* [2010] build a theoretical framework that directly addresses HFT, other work has been conducted to understand how market quality will be impacted when investors have different investment time horizons. *Scharfstein and Stein* [1992] find that short-term speculators may put too much emphasis on short term information and not enough on stock fundamental information. The result is a decrease in the informational quality of asset prices. *Vives* [1995] finds that the market impact of short term investors depends on how information arrives. The informativeness of asset prices is impacted differently based on the arrival of information: "with concentrated arrival of information, short horizons reduce final price informativeness; with discover arrival of information, short horizons enhance it" (*Vives* [1995]). The theoretical work on short horizon investors suggests that HFT may either benetit or harm the informational quality of asset prices. So far, no one has attempted to characterize behaviors of Algorithmic trading practices using structural decision models. A major

[2] Jonathan Spicer and Herbert Lash, Who's Afraid of High-Frequency Trading?, Reuters.com, December 2, 2009

contribution of this study is to model Algorithmic trading strategies from behavior perspective, and eventually these behaviors manifested in the corresponding trading strategies can be used to explain the market quality and price formation proceses.

2.3 Primary Objectives and Hypothsis

The primary objective of this research is to model Algorithmic trading behavior under the framework of inverse reinforcement learning. With the advent of the electronic financial markets, technologies have dramatically improved the speed, capacity and sophistication of the trading functions that are available to market participants. Advanced data feed and audit trail information from the market operators also provide the possibility that the market participants' behavior can be fully observed. With the observations of certain market participant, we aim to recover the optimal policies that the market participants employ and the corresponding reward functions to explain their behaviors under different circumstances.

However, the observations we acquire often present the behavior of our subjects with probabilistic nature. Therefore understanding and addressing the non-deterministic policies and establish the connections with deterministic policies bear important ramification to strategy identification. In order to answer this question, we need to first understand the relationship between a deterministic policy versus non-deterministic policy. We use notation MD for Markov deterministic policy, and MR for Markov non-deterministic policy. We can establish the relationship between the optimality of a deterministic policy versus a non-deterministic policy through the proposition 1 (A) in the following theorems:

Theorem II.6. *Suppose* $0 \leq \gamma \leq 1$*, S is finite or countable, and $r(s, a)$ is is bounded.*

(a) Then there exists a $v* \in V$ satisfying $Lv = v*$. Further, $v*$ is the only element of V with this property and equals v^*_γ.*

(b) For each $d \in D^{MR}$, there exists a unique $v \in V$ satisfying $L_d v = v$. Further, v is the unique solution and equals $v_\gamma^{d^\infty}$.

Theorem II.7. *Let S be discrete, and suppose that the supremum is attained in 5.1 for all $v \in V$. Then*

(a) There exists a conserving policy $d* \in D^{MD}$;

(b) If $d*$ is conserving, the deterministic stationary policy $(d^*)^\infty$ is optimal; and

(c) $v_\gamma^* = \sup_{d \in D^{MD}}(v_\gamma^{d^\infty})$.

For policy $d* \in D^{MD}$, a policy $d*$ is conserving if

$$L_d v_\gamma^* \equiv r_{d*} + \gamma \mathbf{P}_{d^*} v_\gamma^* \tag{2.1}$$

or, alternatively, if

$$d^* \in \underset{d \in D^{MD}}{\operatorname{argmax}}\, r_d + \gamma \mathbf{P}_{d^*} v_\gamma^* \tag{2.2}$$

A policy prescribes a procedure for action selection in each state at a specified decision epoch. Policies range in general from deterministic Markovian to randomized history dependent, depending on how they incorporate past information and how they select actions. In the financial trading world, traders deploy different trading strategies where each strategy has a unique value proposition. We can theoretically use reward functions to represent the value system that encapsulated in the various different trading strategies. For example, a simple keep-or-cancel strategy for buying one unit, the trader has to decide when to place the order and when to cancel the order based on the market environment (may likely be characterized stochastic processes). However the value system under which the trader is looking for gaining profit is to buy one unit of the security at a lowest price possible. This could be realized in number of ways. It could be described as a function $R(s)$ meaning when the system is in state s the trader is always looking fixed reward. This notion of value proposition drives the trader to take corresponding actions according to the market conditions. This ultimately constitutes policies. Therefore a strategy under certain value proposition can be consistently programmed in algorithms to achieve its goal of buy-one-unit in an optimal way.

However for a learning agent to recover the optimal policy, it is a challenge to even just capture all the states in a discrete Markov Decision Process (MDP) model. In other words, the model we develop for the environment may likely be imperfect. Moreover, when the learning agent tries to infer the reward function of traders, often we encounter certain unknowns in the learning process:

(a) whether the trader is deploying a randomized policy or deterministic one;

(b) the observations of the traders behavior are noisy;

(c) unobservable features.

The problem is motivated to estimate the unknown reward function as accurately as possible. This reward function should reflect the traders optimal policy as closely as possible.

Let us say that we receive a series of observations of a particular traders behavior $(s_i, a_i) \in \mathcal{O}$ where the trader is in state s_i and takes action a_i at time step i. Furthermore we defined the traders reward function as R and make the following assumptions:

(a) Attempts to maximize the total accumulated reward according to R;

(b) Executes a stationary policy, i.e. it is invariant w.r.t time and does not change depending on the actions and observations made in previous time steps.

Based on the proposition 1 A, the optimal value attained by a randomized policy is the same as the one attained by a deterministic policy, and there exists an optimal value and it is unique in V by the Theorem 1. We also know from Theorem 1 that each policy in the non-deterministic set has a unique value v. Combining with the Theorem 2, we know that supremum value obtained from all policies can be used to recover the an equivalent optimal stationary deterministic policy.

Use a formalism, we assume the problem we undertake is in an ergodic MDP environment. Intuitively if an MDP is finite, stationary and ergodic, then it should be possible for an adaptive policy to eventually achieve an optimal level of expected reward per cycle. We introduce definition of ergoic MDP by Shane Legg, et al. (*Legg and Hutter* [2004]).

Definition II.8. A MDP is ergodic, if there exists an agent that under policy p every possible observation $o \in \mathcal{O}$ occurs infinitely often with probability 1.

Intuitively this means that the environment never becomes restricted to some subset of possibilities, instead everything that is possible in the environment initially always remains possible. It means that when we observe a traders behavior long enough (assume that the trader has a stationary policy), we observe a set of stationary policies. Furthermore, if we defined a set of policies by:

$$\pi = \sum_{i=1}^{n} \lambda_i \pi_i, \lambda_i \geq 0, \sum_{i} \lambda_i = 1, \tag{2.3}$$

where λ_i is a random variable.

Essentially we are looking for an optimal deterministic stationary policy which achieves the same optimal value as the non-deterministic policy. This will guarantee the learning agent to obtain a unique reward function that achieves the optimal value. The merit of this approach is that the reward function will be unique for a specific set of observations. We will not be concerned about whether the traders' real policy is deterministic or not. This is especially useful in the problem where we attempt to identify traders trading strategies based on a series of observations.

Our hypothesis is that under this reward learning framework, Algorithmic trading strategies can be identified with a targeted accuarcy based on past observations.

CHAPTER III

The Price Impact of Order Events

3.1 Introduction

Most of the existing market price studies have focused on modelling the price impact as a function of trade volume. Earlier empirical studies (*Mike and Sen* [2004], *Jones et al.* [1994], *Karpoff* [2004]) extensively investigated this relationship using various data and statistical tools. In the empirical literature, price impact has been characterized by numerous authors as transient, temporary, instantaneous, permanent, long-memory. In general, there are three very important findings in the field of studying market price impact thus far. First of all, it seems to confirm the intuitive notion that buy trades push prices up and sell trade push price down. This notion is deeply rooted in the market demand and supply model and information propagation during the market price formation process. Secondly, there is a seemingly unanimous consensus: it is that the price impact of trades is an increasing concave function of their sizes (*Farmer and Lillo* [2004]). Lastly, the sign of market orders is strongly autocorrelated in time (*Farmer and Lillo* [2009]). However, there is strong evidence that the limit orders and market activities play an important role in modelling price dynamics (*Potters and Wyart* [2004], *Plerou and Stanley* [2003]). Knez and Ready (*Gatheral* [2010]) have shown that outstanding limit orders (also known as market depth) significantly affect the impact of an individual trade and low depth is a necessary condition for large price changes (*Knez and Ready* [1996]). Bouchaud et al. *Potters and Wyart* [2004] model the dynamic interaction between market orders, limit orders and cancelations on the level of individual events. R. Cont et al. (**?**) also create market order imbalance variables to model their price impact functions. These studies generally expand the analysis of price impact of trades to different forms of limit order events.

The primary objective of this study is to understand empirically how market prices and volatility react to the impact of a series of intensive exogenous order events. We take advantage of the unique characteristics of a dataset collected from a number of Futures markets during a market incident. Since these orders were accidentally injected into the subject markets during an illiquid time period, the expected volatility and price impact should be significant, and furthermore we view this case as a natural experiment where no informational traders are present during the short injection period. We argue that the insight we gain from this study may shed lights on the market impact of High Frequency Trading (HFT) where information is not central among the HFT strategies and yet HFTs share several key characteristics of this natural experiment.

This chapter is structured as follows. Section 2 describes our dataset, and highlights the key characteristics that will be explored in the analysis later. Section 3 studies market volatility impact on both outright markets and spread markets. Section 4 expands the findings from the Section 3 and establishes characteristics of price impact of these high intensity trading activities to the ordinary illiquid markets. Section 5 summarizes the observations from the Section 3 and Section 4 and concludes the implications of the findings to studying high frequency trading strategies and provides precautions of events as such in the future.

3.2 Data and Market Characterization

In this study, we use an order book dataset on September 13, 2010. On that day, between 14:38pm CDT [1] and 14:44pm CDT, test data intended to be placed into the GLOBEX test environment as part of CME Groups normal testing regimen was inadvertently injected into the life system. During the 6 minutes injection period, a total number of 102,154 erroneous messages were injected, and as a result 27,325 errant orders were executed and turned into 42,056 errant contracts. These errant contracts affected 8 energy and metals markets, such as Brent Crude Oil Last Day Financial Futures, Heating Oil Futures, E-mini Natural Gas Futures, Silver Futures, etc.

During the six-minute injection period, a total of 79,325 contracts traded in all contract months in subject markets. Of this volume, 38,116 contracts (48%) involved enormous orders on at least one side of the transactions, and 41,209 contracts (52%) were executed where neither side was generated by an erroneous order. Additional 24,473 contracts traded in the subject markets between 14:44 and the close of the GLOBEX session. Of this volume, 3,940 contracts (16%) involved erroneous orders on at least one side of the transaction and 20533 (84%) contracts were executed that contained no erroneous orders. Those involved with erroneous orders were the orders resting in the market. The total volume associated with trades involving erroneous orders for the period was 42,056 contracts or 40.5%; non-erroneous volume was 61,742 contracts or 59.5%.

As part of the investigation, the Office of Chief Economist (OCE) at the Commodity Futures Trading Commission (CFTC) called audit trial data for Sep. 13 and three other regular trading days from the CME Group, and performed price impact analysis and economic impact assessment. In order to see the changes of trading activities, reference data include first trading day of the two weeks prior Sep. 13, 2010 (i.e. Aug.

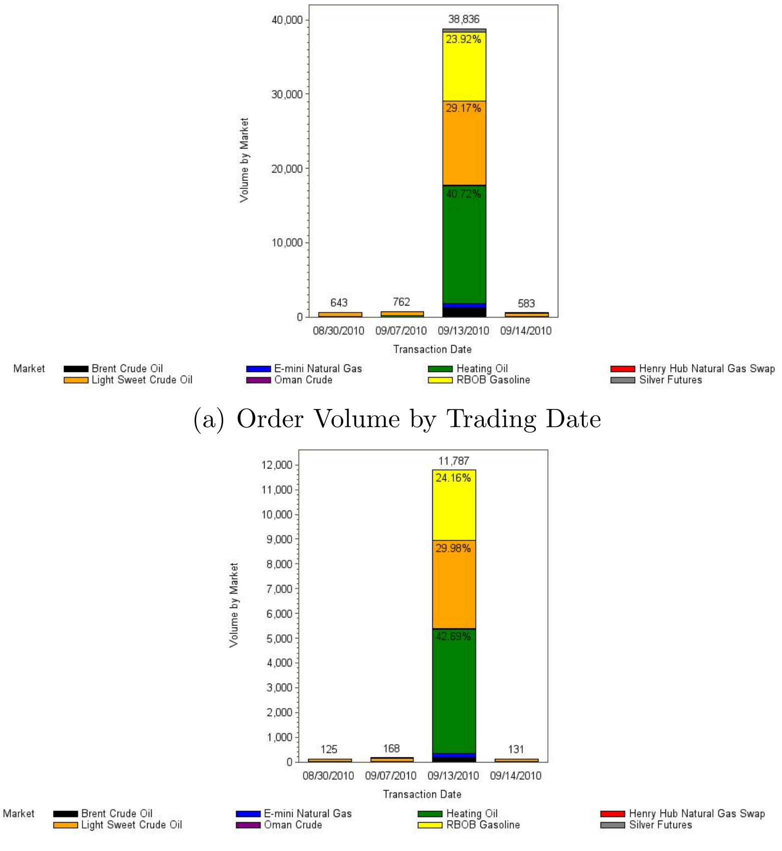
[1] All time cited in the body of this document is Central Daylight Time (CDT).

30, 2010 and Sep. 07, 2010) and the day immediately after the incident (i.e. Sep. 14, 2010). For the 8 affected markets, the total volume transacted during the 6 minutes on Sep. 13th is 65 times the size of the other regular trading days (Table 3.1 shows the trade volume of the eight outright markets, and Figure 3.1 show the proportion of the affected markets in comparison with that of the regular trading days). Even for the relatively liquid market (Light Sweet Crude Oil), it is more than 20 times of the regular size. Furthermore, if we look at the total trade volume throughout the 6 minutes injection period second-by-second in comparison with the three regular trading days, we see that the injection period present a relatively liquid market. The volume traded per second is 16 times higher than the normal trading days (see Table 3.2). It is even more evident, if we look at the order and trade activities in Figure 3.2 and Figure 3.3.

Table 3.1: Trade Volume by Market (6 minutes)

		Trading Date				
$Market$	$Symbol$	08/30/2010	09/07/2010	09/13/2010	09/14/2010	$GrandTotal$
Heating Oil	HO	59	124	15,814	52	16,049
Light Sweet Crude Oil	CL	515	550	11,328	453	12,846
RBOB Gasoline	RB	59	69	9,290	11	9,429
Brent Crude Oil	BZ	1		1,211	7	1,219
E-Mini Natural Gas	QG	2		568	1	571
Silver Futures	SI	8	20	434	59	521
Henry Hub Natural Gas	NN			160		160
Oman Crude	OQ			32		32
Grand Total		644	763	38,837	583	48,827

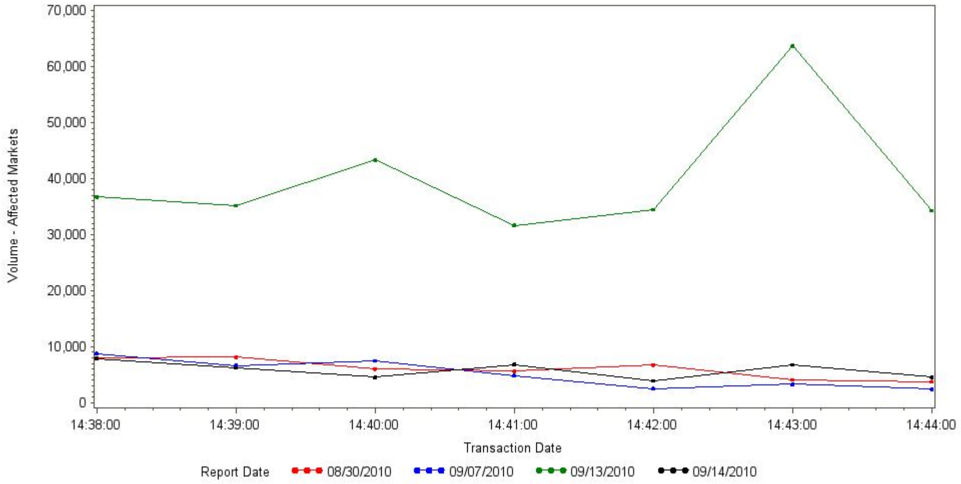
This incident provides a perfect experimental environment to understand the characteristics of market reaction to high frequency trading activities. There are two important aspects of this dataset that highlight the value of this empirical study: First of all, since no one had prior information about this incident, we can safely assume all the traders are uninformed during this time period about the incoming orders. Secondly, since market is normally illiquid during this period, and the injection of



(a) Order Volume by Trading Date

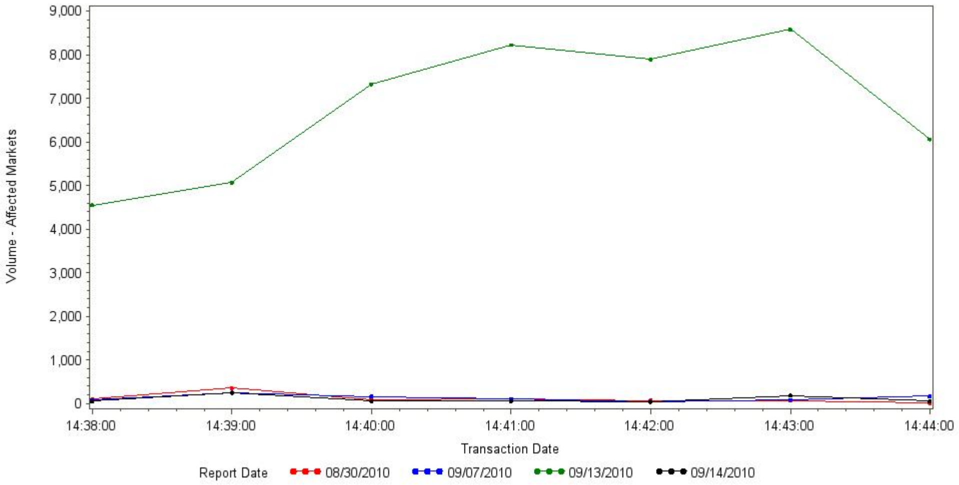
(b) Order Volume by Trading Date

Figure 3.1: **Order Volume** a). Trade Volume on Outright Markets b). Trade Volume on Spread Markets



(a) Order Volume by Trading Date

Figure 3.2: **Order Volume by Trading Date** This figure shows the total order volume throughout the six minutes period in comparison with other three regular trading days. The order volume per minute is 310 times larger than that of the regular trading day.



(a) Trade Volume by Trading Date

Figure 3.3: **Trade Volume by Trading Date** This figure shows the total volume traded throughout the six minutes period in comparison with other three regular trading days.

the erroneous orders bears the key characteristics of the high frequency trading to the regular relatively low frequency trading. This parallel rests on the fact that the trading volume ratio 70% [*Brogaard* [2010]] is strikingly similar to the ratio of erroneous trading to the regular trading (on average the errant trade consists of 71% of the total trade volume). Lastly, the intensity of the trades is 23 times that of the regular trading. Therefore the effect of the injection of these erroneous orders presents itself a perfect opportunity to understand the resulting market volatility and price movement when the high frequency trading is introduced into an ordinary financial market. The goal of this study is to characterize how inactive markets respond to intensive high frequency trading activities. We hope the empirical results from this study will provide a valuable evidence for better understanding of impact of high frequency trading strategies to the market, and help future price impact modelling in the high frequency trading paradigm.

Table 3.2: Trade Volume per Second (6 minutes)

Analysis Variable: Trade Volume per Second					
TradeDate	*NumofObservations*	*Mean*	*StdDev*	*Minimum*	*Maximum*
08/03/2010	121	7	24	1	253
09/07/2010	131	7	13	1	74
09/13/2010	418	114	95	2	653
09/14/2010	153	5	14	1	150

3.3 Volitility Impact Of The Concentrated Trading

3.3.1 Realized Volatility on Intraday Data

In this section, we treat the erroneous orders as exogenous shock to an illiquid market and to study their impact on market volatility. Financial market volatility is indispensable for asset and derivative pricing, asset allocation, and risk management. By far the most popular approach is to obtain volatility estimates using the statistical models that have been proposed in the ARCH and Stochastic Volatility literature (*Weber and Bernd* [2006]). But this approach of measuring volatility is only valid under the specific assumptions of the models used. Yet, the concept of volatility itself is somewhat elusive, as many ways exist to measure it and hence to model it (*Bollerslev and Diebold* [2002]). Moreover, in recent times, the availability of ultra-high frequency data and the work done on them has shed new light on the concept of volatility: as a matter of fact, data sampled at regular intra-daily intervals can be summarized into a measure called realized volatility which under some assumptions is a consistent estimator of the quadratic variation of the underlying diffusion process. Such a measure was widely adopted as a target of forecast accuracy, but the dependence of the measure upon the frequency of observation of the data makes it difficult to come to clear conclusions. Moreover, as shown by Oomen R.C.A. (*Oomen* [2005]) such a measure may be biased if returns used to compute it are serially correlated. In principle, the volatility measures derived from ultra-high frequency data should prove to be more accurate, hence allowing for forecast efficiency gains. We look volatility changes for 8 heavily treaded outright markets and related heavily traded spread markets to understand the volatility impact. The reason we choose these representative markets is partly because the other less frequently markets may fall short the assumption for intra-daily realized volatility. Therefore it may invalidate the applicability of this volatility measure. Nevertheless using the 8 indicative markets is sufficient to understand the general direction of the exogenous shock impact.

To set forth the notation, let $p_{n,t}$ denote the time $n \geq 0$ logarithmic price at time t. The discretely observed time series of continuously compounded returns with N observations per period is then defined as follows:

$$r_{n,t} = p_{n,t} - p_{n-1,t} \tag{3.1}$$

$$r_{n,t} = p^H_{n,t} - p^L_{n-1,t} \tag{3.2}$$

where $n = 1, ..., N$ and $t = 1, ..., T$. If $N = 1$, and $p^H_{n,t}$ is the highest logarithm price and $p^L_{n,t}$ is the lowest logarithmic price. For any series we ignore the first subscript n and thus r_t dontes the time series of periodic return.

We assume following:

$$E[r_{n,t}] = 0 \tag{3.3}$$

$$E[r_{n,t}, r_{m,s}] = 0, n, m, s, tbutnotn = mands = t \tag{3.4}$$

$$E[r^2_{n,t}, r^2_{m,s}] < \infty n, m, s, andt \tag{3.5}$$

Hence, returns are assumed to have mean zero and to be uncorrelated and it is assumed that the variance and covariance of squared returns exist and are finite.

The continuously compounded squared returns may be decomposed as:

$$s_t^2 = \sum_{n=1}^{N} r_{n,t}^2 \tag{3.6}$$

as:

$$E[s_t^2] = \sigma_t^2 \tag{3.7}$$

where σ_t^2 is population variance for period t. We need to take note that it has long been recognized that the spread between the highest price and the lowest price is a function of the volatility during a period, and it is proven a better estimate of volatility. We will use this method in our realized volatility calculation.

3.3.2 Empirical Data Analysis

In the data section, we mentioned that there are only eight markets were affected by the erroneous orders (Table 3.3). We further define an errant trade as those trades that either side of the transaction contains an erroneous order. For a spread market, if either leg involves an erroneous order, we then identify the spread as an errant trade. In order to satisfy the sufficient frequency of activities for computing volatility, we focus our attention to the top four heavily traded markets. Within each of these four markets, we then picked 2 markets which have the top two highest volumes to represent that market. We compute realized volatility through the 6 minutes injection period for all 8 markets.

Furthermore, this time period, i.e. between 14:38 and 14:44, is generally a period of low liquidity, because it occurs after the close of Regular Trading Hours, following settlement of the markets. Sep. 13, 2010 was no exception. We divide the 6 minutes into 360-second periods, and we use (3.2) and (3.6) to estimate the realized volatility. We present these results in Table 4. From this table, we see that volatility is mostly low for the 8 outright markets, and they are relatively high among the 8 spread markets. These realized volatility estimates are consistent with the observations we have on these spread markets.

From Table 3.4, we see that the spread market RBV0-RBZ0 (RBOB Gasoline Futures) experienced extremely high price volatility. We reviewed the RBV0-RBZ0 spread for the period from 14:38 through 14:45. This spread is barely traded during this period from data we have on Aug 30. 2010, Sep 07, 2010 and Sep 14, 2010. On September 07 the first trading day a week prior the incident, during the six minutes period, the RBV0-RBZ0 order book reflected bids and offers (20 ticks wide) with no volume trading. The order books for the component legs RBV0 and RBZ0 during the same period were very thin. This pattern is consistent with the data we have for the first trading day of two weeks prior the incident and the day after the incident. However on Sep. 13, 2010, during the injection period, the differential between bid and offer traded for the RBV0-RBZ0 spread fluctuated to up to 51 ticks wide. The market traded, with the spread moving from 21 (the first trade at 14:38:03) to -30 by 14:43:40. The spread widened because orders injected in outright and related spread market were executed against the RBV0-RBZ0 spread bids. As bids for the RBV0-RBZ0 spread were matched against outright and related spread legs, new orders at lower differentials replaced them. The price of the RBV0-RBZ0 bid moved down as the top of the order book was executed. The decline was not steady; prices fluctuated reflecting the effect of orders injected into outright and related spread order books. The market traded in a range of -10 to 30 through 14:44:19. By 14:44:43 most of the offers had been withdrawn from the market and very few trades occurred through the end of the period reviewed. In excess of from 75

Another spread market that experienced large volatility is RBV0-RBF1, but it is almost 35 times less than RBV0-RBZ0 that we mentioned earlier. However its magnitude is in line with the rest of the spread markets we studied. It experienced

Table 3.3: Market Volume Affected by the Erroneous Orders

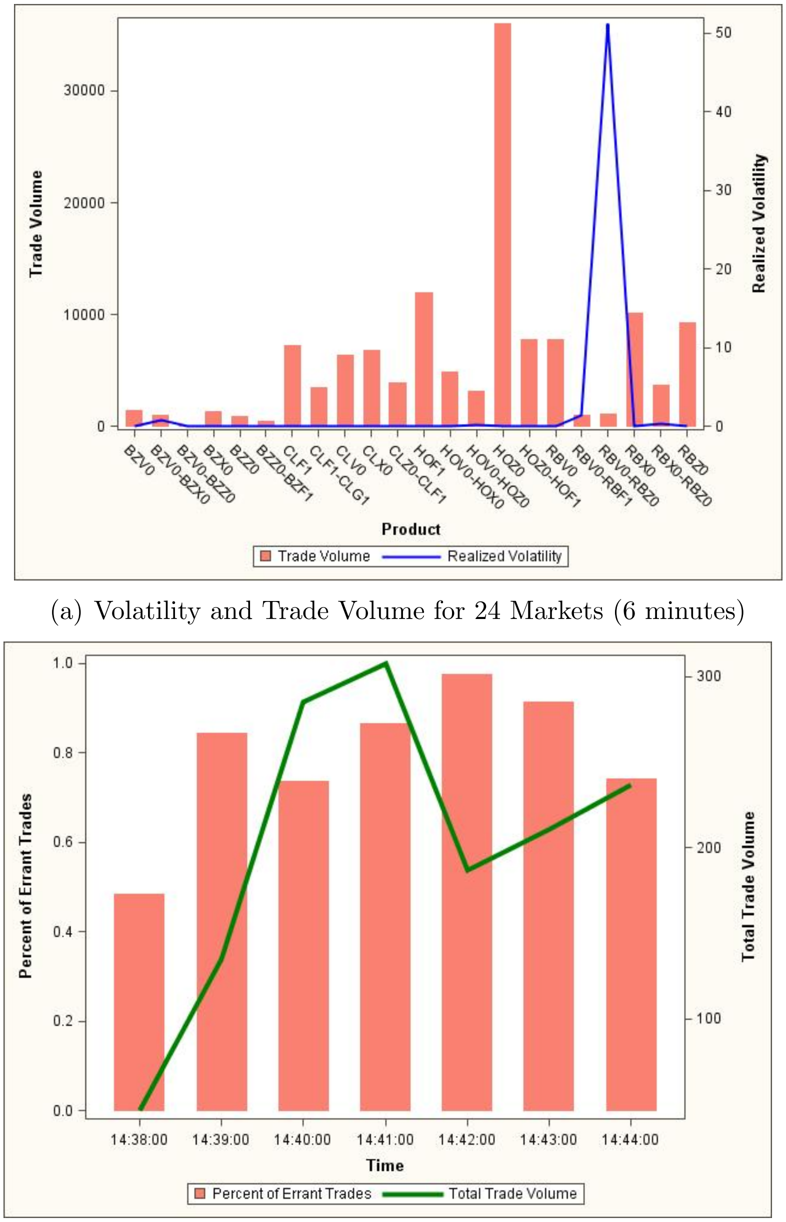
$MarketName$	$Volume$	$\%Total$	$Trades$	$\%Total$
Heating Oil	19,154	45.50	12,378	45.30
Light Sweet Crude Oil	11,031	26.20	5,679	20.80
RBOB Gasoline	7,041	16.70	6,384	23.40
Brent Crude Oil	2,617	6.20	1,659	6.10
E-Mini Natural Gas	1,232	2.90	483	1.80
Silver Futures	600	1.40	400	1.50
Henry Hub Natural Gas	286	0.70	286	1.00
Oman Crude	95	0.20	56	0.20
Grand Total	42,056	100.00	27,325	100.00

larger swing from -173 (14:38:03) to -283 (14:43:27). In other words, it fluctuated up to 110 ticks. After 14:42:32, the market traded in a range of -211 to -283, and the volume started to thin out toward the end of the injection. The dramatic volatility difference between these two markets can be explain by the fact that the component leg RBF1(Figure 3.5 a) has experienced much less volatility compared with the component leg RBZ0 (Figure 3.5 b).

3.3.3 Results

Because orders injected entered the markets during this relatively illiquid time period, for many spreads in the subject markets, the effect on spread pricing was pronounced. Two examples of how the presence of this relatively less liquid state was impacted by the flow of the erroneous orders are analysed above. Using realized volatility measure, we show that this intensive exogenous order flow has more significant impacted on market volatility of the spread markets than that of the outright markets (Table 3.4). Unfortunately, due to the illiquid nature of this trading period on regular trading days, we find extremely low trading volume for the subject markets, and we cannot measure volatility for regular trading days for comparison.

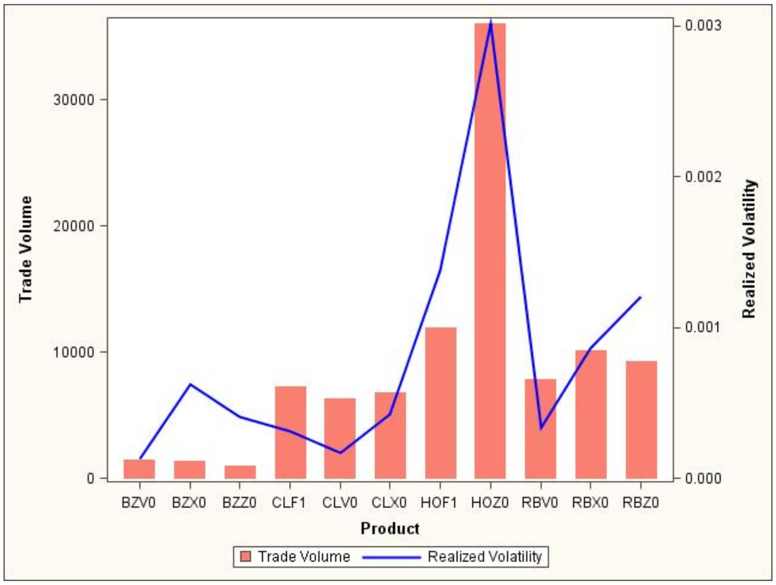
If we plot distribution of the realized volatility for both the selected outright and spread markets (see Figure 3.4 b), we observe that the volatility of the outright



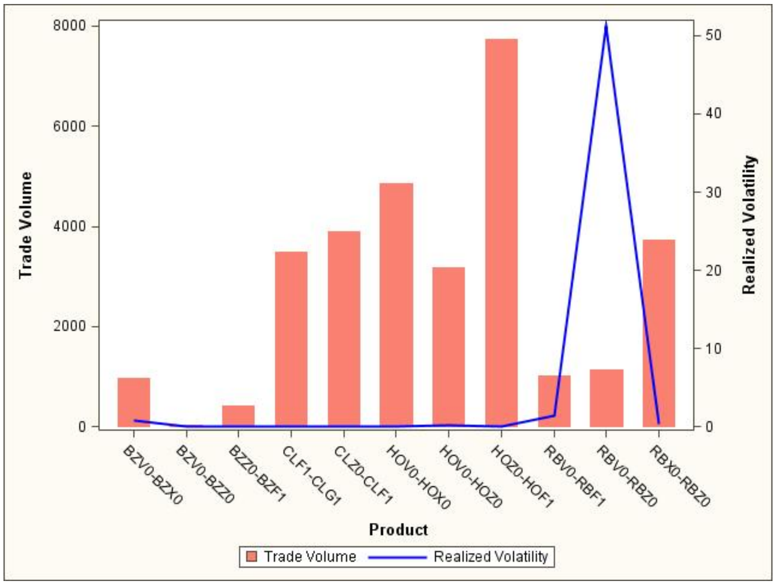
(a) Volatility and Trade Volume for 24 Markets (6 minutes)

(b) Fraction of Errant Trade Volume for RBV0-RBZ0(6 minutes)

Figure 3.4: **Volatility and Trade Volume**
a). The figure shows the total volume traded and the realized volatility during the injection period. Volatility of RBV0-RBZ0 market is extremely higher than the rest of the markets. But on average the volatility of spread markets is significantly higher than the outright markets. b). This spread market experienced the highest volatility among all the affected markets. We observe that 70% 80% of trades involved erroneous orders.



(a) Outright Market Volatility (6 minutes)



(b) Spread Market Volatility (6 minutes)

Figure 3.5: **Market Volatility** a). This graph shows that overall volatility of the outright markets are low with mean of 0.000738 and median of 0.000521. The largest volatility coincides with the market with highest trading volume (most liquid market). b). This graph shows that volatility of the spread markets is relatively high with mean of 4.921221 and median of 0.072359. The volatility of RBV0-RBZ0 market is substantially higher than the other spread markets.

Table 3.4: Market Volatility

Market	*Outright*	*Trade*	*Realized*
Name	*Indicator*	*Volume*	*Volatility*
RBV0-RBZ0	0	1133	51.22053
RBV0-RBF1	0	1019	1.453483
BZV0-BZX0	0	973	0.771852
RBX0-RBZ0	0	3731	0.385033
HOV0-HOZ0	0	3165	0.150942
HOV0-HOX0	0	4849	0.072359
CLF1-CLG1	0	3481	0.030538
HOZ0-HOF1	0	7745	0.026175
CLZ0-CLF1	0	1951	0.009393
CLV0-CLX0	0	1557	0.008441
BZZ0-BZF1	0	423	0.003741
HOZ0	1	18002	0.001508
HOF1	1	11916	0.001382
RBZ0	1	9254	0.001204
RBX0	1	10136	0.000859
HOX0	1	11472	0.000815
BZX0	1	1352	0.000620
CLX0	1	6786	0.000422
BZZ0	1	924	0.000406
RBV0	1	7808	0.000338
CLF1	1	7272	0.000309
BZV0	1	1436	0.000127

markets follows roughly a Gaussian distribution with mean of 0.000738 and median of 0.000521. However the distribution of that of the spread markets is highly skewed with mean of 4.921221 and median of 0.072359. It is clear that the volatility of the spread markets is significantly higher than the outright markets.

There is an extreme outlier which is RBV0-RBZ0 market. As we analyzed above, this extreme volatility can be attributed to the extreme movement of its component legs. On average, the number of erroneous orders is 47.80% of the total order volume, and the number of erroneous trades is 71.32% of the total trade volume during this incident. Among all the relatively liquid markets, the two leg components of this spread have relatively low proportion of erroneous trades (40.28% for RBV0 and 30.70% for RBZ0). It means that large portion of the trades came from regular orders.

It indicates that the erroneous orders have triggered more regular order inflow to the related outright markets. Since there is no clear information to incorporate into the price, the market price moved in a much more disorientated fashion which caused extra volatility.

3.4 Order Events Based Impact Model

3.4.1 Price Impact Model based on Limit Order Events

The relation between order flow and price changes has attracted considerable attention in the recent years (*Hasbrouck* [2007], *Mike and Farmer* [2008], *Potters and Wyart* [2004], *Kockelkoren and Potters* [2006], *Lyons* [2006] and *Farmer and Lillo* [2009]). The aim of this section is to provide an estimate of the impact of all order book events: market orders, limit orders and cancellations. We study the correlation between all event types and signs. Assuming a second order model of impact, we map out from empirical data the average impact of these orders.

Stephens et al. (*Waelbroeck and Mendoza* [2009]) showed that the concurrent limit order activity can make a difference in terms of trades' impact. They argued that the shape of the price impact function essentially depends on the contemporaneous limit order activity. R. Cont et al. (*Kukanov and Stoikov* [2010]) took a different approach suggested that order book events have a linear impact on prices which can be related to the model proposed by Bouchaud et al. (*Bouchaud and Kockelkoren*). The major difference between these two models lies in the aggregation across time and events. As argued by Bouchaud et al., order book events have complicated auto- and cross-correlation structures on the timescale of individual events, which typically vanish after 10 seconds. This allows us to measure more accurately the average impact of all types of orders, and to assess precisely the importance of impact fluctuations due to changes in the gaps behind the best quotes. The order flow imbalance (OFI) represents the net order flow at the bid and ask, and it tracks changes in the size of the bid and ask queues. They found that this aggregate variable explains mid-price changes over short time scales in a linear fashion. Taking a similar approach suggested by R. Cont et al. (*Stoikov and Talreja* [2010b]), we assume that the price impact coefficient is a constant over a given interval $[t_(k-1), t_k]$ and estimate the model by ordinary least squares regression:

$$\nabla p_t = p_{n,t} - p_{n-1,t} \tag{3.8}$$

where, $n = 1, ..., N$ period, and $p_(n, t)$ is the logarithm price at the end of period n at day t.

$$\nabla p_t = \beta_{0,t} + \beta_{1,t} OFI_t + \beta_{2,t} OFI_t * OFI_t + \epsilon_t \tag{3.9}$$

$$\nabla OFI_t = \sum_{n=N(t_{k-1}+1}^{N(t_k} [q_{n,t}^{bid-order} - q_{n,t}^{ask-order} + q_{n,t}^{ask-cancellation} - q_{n,t}^{bid-cancellation}] \tag{3.10}$$

where $N(t_{k-1}) + 1$ and $N(t_k)$ are the index of the first and the index of the last event in the interval $[t_{k-1}, t_k]$.The order flow imbalance is a measure of supply/demand imbalance, which encompasses limit orders and cancelations. We define here q_n^{event} as the quantity of each order book event at n in the interval $[t_{k-1}, t_k]$. We consider four types of events ask order, bid order, ask cancellation, bid cancellation. This single variable allows us to keep track of the net order flow at the bid and ask, and changes in the size of the bid and ask queues.

3.4.2 Empirical Data Analysis

We take 5 levels order book data, and assume that the price impact coefficient β is constant over the period of our interest. We also consider higher order/nonlinear dependence with a quadratic term in equation 3.9. Every observation of the bid and the ask consists of the bid price P^B, the size q^B of the bid queue (in number of shares), the ask price P^A and the size q^A of the ask queue (in number of shares). As it is argued by Bouchaud et al. (*Farmer and Lillo* [2009]), order book events have complicated auto-correlation and cross-correlation structures on the timescale of individual events, which typically vanish after 10 seconds. We therefore choose this time scale for our intraday data. We calculate OFI for every 10 seconds and build a least square regression models for each of the 24 markets we choose. We also run the regression on the data from the 3 regular trading days. We list the R^2 in 3.5. The average R^2 value is 3.21%, which indicates the percentage influence of order flow imbalance to the corresponding market prices. However on Sep. 13, 2010, the average R^2 value is 14.85% which is 4.6 times larger than the normal market. If we consider the outright and spread markets separately, we find the average R^2 value for the outright markets is 13.52% and the average R^2 value for spread markets is 17.23%. It can be translated to that the price impact of the order flow imbalance for spread markets is 27% higher than that of the outright markets.

Furthermore, we use stepwise model selection method on the OFI and OFI*OFI dependent variables with entrance significant level of .15 to check whether the impact of order flow imbalance to market prices is statistically significant. For stepwise model selection method, variables are added one by one to the model, and the statistic for a variable to be added must be significant at a preset entry level. The stepwise process ends when none of the variables outside the model has an statistic significant at the entry level and every variable in the model is significant at the stay level, or when the variable to be added to the model is the one just deleted from it. We show the results in 3.6. The blank cell means that the impact model does not pass statistical test (z-test at 10% significant level). Since there is only 6 minutes, we require at least 30 observations for regression. If there are not enough observations, we assume the model is not statistically significant. For the models on the incident data, we not only use the total imbalance during the period, but we also considered the erroneous order imbalance as another dependent variable [2].

Table 3.5: Explanation Power of Price Change over Order Flow Imbalance

	R^2			
	8/30/2010	9/7/2010	9/13/2010	9/14/2010
BZV0	1.34%	1.25%	0.96%	10.95%
BZX0	0.00%	3.47%	4.31%	5.90%
BZZ0	0.00%	0.00%	1.44%	0.00%
RBV0	22.77%	50.00%	6.14%	1.33%
RBX0	0.00%	1.32%	17.67%	0.90%
RBZ0	0.00%	0.00%	24.78%	0.00%
HOX0	0.00%	0.00%	10.36%	0.00%
HOZ0	0.00%	0.00%	11.64%	3.34%
HOF1	0.00%	0.00%	6.46%	0.00%
CLX0	2.80%	3.50%	16.46%	1.60%
CLF1	0.83%	0.36%	29.09%	2.66%
CLF1	2.85%	7.56%	20.36%	8.22%
BZV0-BZX0	0.00%	0.00%	13.58%	0.00%
BZV0-BZZ0	0.00%	0.00%	9.35%	0.00%
BZZ0-BZF1	0.00%	0.00%	9.34%	0.00%
RBV0-RBZ0	7.68%	1.42%	10.70%	1.74%
RBV0-RBF1	2.51%	24.49%	45.15%	6.02%
RBX0-RBZ0	7.68%	1.42%	24.90%	1.74%
HOZ0-HOF1	0.00%	6.06%	15.40%	1.53%
HOV0-HOZ0	12.05%	0.79%	5.01%	2.30%
HOV0-HOX0	2.38%	1.69%	12.13%	0.42%
CLV0-CLX0	0.00%	0.00%	28.03%	0.00%
CLF1-CLG1	6.16%	0.66%	8.73%	0.57%
CLZ0-CLF1	0.50%	6.67%	24.41%	1.92%
Average	2.90%	4.61%	14.85%	2.13%

From this result, we see that the percentage of the markets that experienced price

[2] We note that OFI_t^2 enters the model at 90% level of z-test for most of the cases (80%), where the price impact is statistically significant. This suggests a strong non-linear price impact function for order flow imbalance.

impact on Sep. 13, 2010 is 5.5 times higher than the average of that of the regular trading days before and after the incident. If we consider the outright and spread markets separately, the number of impacted spread markets is 21 times more than that of the normal trading days. The number of impacted outright markets is 2.4 times more than that of the normal trading days. More interestingly, the erroneous order imbalance variable alone has significant price impact in most of the impacted markets. This may not be a surprise, because the most of the order events involve erroneous orders.

Table 3.6: Price Impact Statistical Significance Test

	8/30/2010		9/7/2010		9/13/2010		9/14/2010	
	$FValue$	$Pr > F$	$FValue$	$Pr > F$	$FValue$	$Pr > F$	$FValue$	$Pr > F$
BZV0							2.97	0.09360
BZX0								
BZZ0			12.55	0.001800				
RBV0	6.89	0.014600	33.04	0.000100				
RBX0								
RBZ0					6.17	0.0048		
HOX0					3.84	0.0574		
HOZ0					3.61	0.0650		
HOF1								
CLX0								
CLF1			4.12	0.023900	14.49	0.0005		
CLF1								
BZV0-BZX0					4.09	0.0524		
BZV0-BZZ0								
BZZ0-BZF1					4.20	0.0516		
RBV0-RBZ0								
RBV0-RBF1			5.51	0.008400	24.53	0.0001		
RBX0-RBZ0					2.99	0.0315		
HOZ0-HOF1					4.85	0.0336		
HOV0-HOZ0								
HOV0-HOX0								
CLV0-CLX0					6.36	0.0042		
CLF1-CLG1								
CLZ0-CLF1					5.05	0.0303		
Impacted		1		4		11		1
% Impacted		4%		17%		46%		4%

3.4.3 Results

From our price impact analysis of the 24 representative markets using the order flow imbalance variable, we conclude that the order flow events on Sep. 13, 2010 exerted higher price impact on both the number of markets and the proportional influence on price change compared with the regular trading days. Furthermore, the effect is consistent with the volatility impact in that the impact to spread markets was more pronounced than the outright markets. Out of the 24 markets studied, 58% of spread markets had significant price impact, while 33% of outright markets had significant price impact. Compared with the regular trading days, the number of impacted markets is 5.5 times more on Sep. 13, 2010.

3.5 Conclusions

In this study, we analyzed the market impact of some intensive exogenous order events to relatively illiquid markets as a natural experiment. The unique contribution of this study is to characterize the market responds to large intensive orders in a relatively short period of time without true supply and demand information on the horizon. In this case, the exogenous order volumes are 3-10 times larger than the normal market orders (Figure 3.2). We show that intensive orders as such generate significant market impact in terms of both volatility and prices. Using realized volatility and order flow imbalance variable, we examined the impact of market volatility and market prices, and we conclude following:

(a) Intensive exogenous order events generate more market volatility in spread markets than that in outright markets.

(b) On average, intensive exogenous order events change market prices in the magnitude of 3-5 times than that of the normal market condition, and

(c) Price impact is more pronounced in spread markets than outright markets.

Although it is intuitive that the market would respond to large quantity of exogenous order events in the similar way it would under the regular market condition, market impact under the normal market condition is much more complex due to many reasons, such as supply and demand, the interplay of informed and uninformed traders, etc. This study provides an empirical evidence to such events where there is almost no information inflow in the process, and yet the volatility and price impact dynamics are evident. In the HFT context, most of the high frequency traders are not informed, and they trade close to 70% of the market volume. The implication of this kind of events may help draw parallel to the HFT to the traditional low frequency market.

Furthermore, these results would also provide cautionary evidence for market participants and regulators to understand the impact of the unexpected events as such would do to the market. Therefore it may help them to design mitigation plans to reduced their risk exposures accordingly.

CHAPTER IV

Behavior Based Learning in Identifying High Frequency Trading Strategies

4.1 Introduction

Many financial market participants now employ algorithmic trading, commonly defined as the use of computer algorithms to automatically make certain trading decisions, submit orders, and manage those orders after submission. By the time of the "Flash Crash" (On May 6, 2010 during 25 minutes, stock index futures, options, and exchange-traded funds experienced a sudden price drop of more than 5 percent, followed by a rapid and near complete rebound), algorithmic trading was thought to be responsible for more than 70% of trading volume in the U.S. ([*Hendershott and Riordan*, 2008], [*Brogaard*, 2010], [*Jones and Menkveld*, 2011], and [*Kirilenko et al.*, 2011]). Moreover, Kirilenko et al. [*Kirilenko et al.*, 2011] have shown that the key events in the Flash Crash have a clear interpretation in terms of algorithmic trading.

A variety of machine learning techniques have been applied in financial market analysis and modeling to assist market operators, regulators, and policy makers to understand the behaviors of the market participants, market dynamics, and the price discovery process of the new electronic market phenomena of algorithmic trading(*Hasbrouchk and Seppi* [2001a], *Hendershott and Riordan* [2008], *Gatheral* [2010],

?, *Jones et al.* [1994], and *Karpoff* [2004]). We propose modeling traders' behavior as a Markov Decision Process (MDP), using observations of individual trading actions to characterize or infer trading strategies. More specifically, we aim to learn traders' reward functions in the context of multi-agent environments where traders compete using fast algorithmic trading strategies to explore and exploit market microstructure.

Our proposed approach is based on a machine learning technique (*Sutton and Barto* [1998], *Barto and Williams* [1991], and *Bagnell and Dey* [2008]) known as Inverse Reinforcement Learning (IRL) (*Ng and Russel* [2000], *Abbeel and Ng* [2004], *Syed and Schapire* [2007], *Russell* [1998], and *Ramachandran and Amir* [2007]). In IRL, one aims to infer the model that underlies solutions that have been chosen by decision makers. In this case the reward function is of interest by itself in characterizing agent's behavior irregardless of its circumstances. For example, Pokerbots can improve performance against suboptimal human opponents by learning reward functions that account for the utility of money, preferences for certain hands or situations, and other idiosyncrasies (*Schaeffer and Szafron* [1998]) Another objective in IRL is to use observations of the traders' actions to decide ones' own behaviors. It is possible in this case to directly learn the reward functions from the past observations and be able derive new policies based on the reward functions learned in a new environment to govern a new autonomous process (apprenticeship learning). In this paper, we focus our attention on the former problem to identify trader's behavior using reward functions.

The rest of the paper is structured as follows: In Section 2, we define notation and formulate the IRL model. In Section 3, we first propose a concise MDP model of the limit order book to obtain reward functions of different trading strategies, and then solve the IRL problem using a linear programming approach based on an assumption of rational decision making. In Section 4, we present our agent-based simulation model for E-Mini S&P 500 futures market and provide validation results that suggest this model replicates with high fidelity the real E-Mini S&P 500 futures market. Using this simulation model we generate simulated market data and perform two experiments. In the first experiment, we show that we can reliably identify High Frequency Trading (HFT) strategies from other algorithmic trading strategies using IRL. In the second experiment, we apply IRL on HFTs and show that we can accurately identify a manipulative HFT strategy (Spoofing) from the other HFT strategies. Section 5 discusses the conclusion of this study and the future work.

4.2 Problem Formulation - Inverse Reinforcement Learning Model

The primary objective of our study is to find the reward function that, in some sense, best explains the observed behavior of a decision agent. In the field of reinforcement learning, it is a principle that the reward function is the most succinct, robust and transferable representation of a decision task, and completely determines the optimal policy (or set of policies) (*Ramachandran and Amir* [2007]). In addition, knowledge of the reward function allows a learning agent to generalize better, since such knowledge is necessary to compute new policies in response to changes in environment. These points motive our hypothesis that IRL is a suitable method for characterizing trading strategies.

4.2.1 General Problem Definition

Lets define a (infinite horizon, discounted) MDP model first. Let $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{R}\}$, where:

$s \in \mathcal{S}$ where $\mathcal{S} = \{s_1, s_2, ..., s_N\}$ is a set of N states;

$\mathcal{A} = \{a_1, a_2, ..., a_k\}$ is a set of k possible actions;

$\mathcal{P} = \{P_{a_j}\}_{j=1}^{k}$, where P_{a_j} is a transition matrix such that $P_{sa_j}(s')$ is the probability of transitioning to state s' given action a_j taken in state s;

$\gamma \in (0, 1)$ is a discount factor;

$\mathcal{R}$ is a reward function such that R or $R(s, a)$ is the reward received given action a is taken when in state s.

Within the MDP construct, a trader or an algorithmic trading strategy can be represented by a set of primitives $(\mathcal{P}, \gamma, \mathcal{R})$ where $\mathcal{R}$ is a reward function representing the trader's preferences, $\mathcal{P}$ is a Markov transition probability representing the trader's subjective beliefs about uncertain future states, and γ is the rate at which the agent discounts reward in future periods. In using IRL to identify trading strategies, the first question that needs to be answered is whether $(\mathcal{P}, \gamma, \mathcal{R})$ is identified. Rust (*Rust* [1997]) discussed this identification problem in his earlier work in economic decision modeling. He concluded that if we are willing to impose an even stronger prior restriction, stationarity and rational expectations, then we can use non-parametric methods to consistently estimate decision makers' subjective beliefs from observations of their past states and decisions. Hence in formulating the IRL problem in identifying trading strategies, we will have to make two basic assumptions: first, we assume the policies we model are stationary; second, the trading strategies are rational expected-reward maximizers.

Here we define the *value function* at state s with respect to policy π and discount γ to be $V_{\gamma}^{\pi}(s) = E[\sum_{t=0}^{\infty} \gamma R(s^t, \pi(s^t))|\pi]$, where the expectation is over the distribution of the state sequence $\{s^0, s^1, ..., s^t\}$ given policy π (superscripts index time). We also define the $Q_{\gamma}^{\pi}(s, a)$ for state s and action a under policy π and discount γ to be the expected return from state s, taking action a and thereafter following policy π. And then we have the following two classical results for MDPs (see, e.g., *Sutton and Barto* [1998], *Bertsekas* [2007]):

Theorem 1: (Bellman Equations) Let an MDP $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{R}\}$, and a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ be given. Then, for all $s \in \mathcal{S}, a \in \mathcal{A}, V_\gamma^\pi$ and Q_γ^π satisfy:

$$V_\gamma^\pi(s) = R_\pi(s, \pi(s)) + \gamma \sum_{j \in S} P_{s\pi(s)}(j) V_\gamma^\pi(j), \forall s \in \mathcal{S} \tag{4.1}$$

$$Q_\gamma^\pi(s, a) = R_\pi(s, \pi(s)) + \gamma \sum_{j \in S} P_{sa}(j) V_\gamma^\pi(j), \forall s \in \mathcal{S} \tag{4.2}$$

Theorem 2: (Bellman Optimality) Let an MDP $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{R}\}$, and a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ be given. Then, π is an optimal policy for M if and only if, for all $s \in \mathcal{S}$:

$$\begin{multline} V_\gamma^{\pi^*}(s) = \max_{a \in \mathcal{A}} [R_\pi(s, \pi(s)) + \gamma \sum_{j \in \mathcal{S}} P_{s\pi(s)}(j) V_\gamma^\pi(j)], \\ \forall s \in \mathcal{S} \tag{4.3} \end{multline}$$

The Bellman Optimality condition can be written in matrix format as follows:

Theorem 3: Let a finite state space $\mathcal{S}$, a set of $a \in \mathcal{A}$, transition probability matrix P_a and a discount factor $\gamma \in (0, 1)$ be given. The a policy given by π is an optimal policy for M if and only if, for all $a \in \mathcal{A} \setminus \pi$, the reward R satisfies:

$$(P_\pi - P_a)(I - \gamma P_\pi)R \succeq 0 \tag{4.4}$$

4.2.2 Linear Programming Approach to IRL

The IRL problem is, in general, highly underspecified, which has led researchers to consider various models for restricting the set of reward vectors under consideration. The only reward vectors consistent with an optimal policy π are those that satisfy the set of inequalities in Theorem 3. Note that the degenerate solution $R = 0$ satisfies these constraints, which highlights the underspecified nature of the problem and the need for reward selection mechanisms. Ng and Russel (*Ng and Russel* [2000]) advance the idea choosing the reward function to maximize the difference between the optimal and suboptimal policies, which can be done using a linear programming formulation. We adopt this approach, maximizing:

$$\sum_{s\in\mathcal{S}}[Q^{\pi}_{\gamma}(s,a') - \gamma \max_{a\in\mathcal{A}\backslash a'} Q^{\pi}_{\gamma}], \forall a \in \mathcal{A} \tag{4.5}$$

Putting theorem 4.4 and 4.5 together, we have an optimization problem to solve to obtain a reward function under an optimal policy:

$$\begin{aligned} \max_{R}[\sum_{s\in\mathcal{S}}\beta(s) - \lambda\sum_{s\in\mathcal{S}}\alpha(s)] \\ s.t. \\ \alpha(s) \succeq \beta(s), \forall s \in \mathcal{S} \\ (P_{\pi} - P_{a})(I - \gamma P_{\pi})R \succeq \beta(s), \forall a \in \mathcal{A}, \forall s \in \mathcal{S} \\ (P_{\pi} - P_{a})(I - \gamma P_{\pi})R \succeq 0 \end{aligned} \tag{4.6}$$

In summary, we assume an ergodic MDP process. In particular, we assume the policy defined in the system has a proper stationary distribution. And we further assume that trader's trading strategies are rational expected reward maximizers. There are specific issues regarding the non-deterministic nature of trader's trading strategies when dealing with empirical observations, and we will address them later in the next section.

4.2.3 Key Modeling Issues

One of the key issues that arise in applications of IRL or apprenticeship learning to algorithmic trading is that the trader under observation may not appear to follow a deterministic policy. In particular, a trader observed in the same state on two different occasions may take two different actions, either because the trader is following a randomized policy or because the state space used in the model lacks the fidelity to capture all the factors that influence the trader's decision. To address the issue of non-deterministic policies, we need to first understand the relationship between a deterministic policy versus non-deterministic policy under the assumption we made earlier. We use notation MD for Markov deterministic policy, and MR for Markov non-deterministic policy. We can establish the relationship between the optimality of a deterministic policy versus a non-deterministic policy through the following proposition (*Puterman* [1994]):

Proposition: For all $v \in V$ and $0 \leq \gamma \leq 1$:

$$\sup_{d \in D^{MD}} \{R_d + \gamma P_d v\} = \sup_{d \in D^{MR}} \{R_d + \gamma P_d v\}, \forall d \in \mathcal{A} \tag{4.7}$$

Policies range in general from deterministic Markovian to randomized history dependent, depending on how they incorporate past information and how they select actions. In the financial trading world, traders deploy different trading strategies where each strategy has a unique value proposition. We can theoretically use cumulative reward to represent the value system encapsulated in the various different trading strategies. For example in a simple keep-or-cancel strategy for buying one unit, the trader has to decide when to place an order and when to cancel the order based on the market environment (can be characterized as Stochastic processes) to maximize its cumulative reward under the constraint of the traders' risk utility and capital limit. This can be realized in a number of ways. It can be described as a function R(s) meaning when the system is in state s the trader is always looking for a fixed reward. This notion of value proposition drives the traders to take corresponding optimal actions according to the market conditions. However due to the uncertainty of the environment and the random error of the measurement in the observations, a deterministic policy could very likely be perceived to have a non-deterministic nature.

Based on the proposition or equation 4.7, the optimal value attained by a randomized policy is the same as the one attained by a deterministic policy, and there exists an optimal value and it is unique in V. Therefore, we know that the supremum value obtained from all policies can be used to recover an equivalent optimal stationary deterministic policy. Essentially we are looking for an optimal deterministic stationary policy which achieves the same optimal value as the non-deterministic policy. This guarantees the learning agent to obtain a unique reward function that achieves the optimal value. The merit of this approach is that the reward function will be unique for a specific set of observations. We will not be concerned about whether the trader's real policy is deterministic or not. This is especially useful in the problem where we attempt to identify traders' trading strategies based on a series of observations.

4.3 A MDP Model for Limit Order Book

Cont et al. (*Stoikov and Talreja* [2010b], and *Kukanov and Stoikov* [2011]) make the claim that order flow imbalance and order volume imbalance have the strongest link with the price changes. It seems that these two variables can best capture the limit order book dynamics. It has been proven effective in modeling buy-one-unit and make-the-spread strategies by Hunt, et al. *Hult and Kiessling* [2010] where three price levels have shown significantly good resemblance to the real market characteristics. Other financial market microstructure studies also provide strong evidence of using order book imbalance to represent the market supply and demand dynamics or information asymmetry (*Hasbrouchk and Seppi* [2001b], *Karpoff* [2004], *Jones et al.* [1994], *Bouchaud and Kockelkoren* and *Obizhaeva and Wang* [2005]). Based on this evidence, we choose two bid/ask volume imbalance variables to capture the market environment, and we choose position/inventory level as a private variable of the trader. In summary, we use three sensory variables to characterize the environment in which the traders operate. Now we can define state $s = [TIM, NIM, POS]^T$, and each variable takes following discrete values:

TIM - volume imbalance at the best bid/ask: {-1, 0, 1};

NIM - volume imbalance at the 3rd best bid/ask: {-1, 0, 1};

POS - position status: {-1, 0, 1}.

When the variable takes value 0 (in neutral state), it means that the variable takes mean (μ) value within $\mu \pm 1.96\sigma$; when the value is above $\mu + 1.96\sigma$, we define it as high; and when the value is below $\mu - 1.96\sigma$, we define it as low. Essentially we have two external variables: TIM and NIM. Variables TIM and NIM inform the traders whether volume imbalance is moving toward sell side (low), neutral, or toward buy side (high), as well as the momentum of the market price movement. The private variable POS informs traders whether his or her inventory is low, neutral or high. All three variables are very essential for algorithmic traders to make their trade decisions. We also define a set of actions that correspond to traders' trading choices at each state a = {PBL, PBH, PSL, PSH, CBL, CBH, CSL, CSH, TBL, TBH, TSL, TSH}, and each value is defined in TABLE 4.1.

Table 4.1: Action definition.

Action Code	**Action Description**
1	PBH - place buy order higher than the 3rd best bid price
2	PBL - place buy order lower than the 3rd best bid price
3	PSH - place sell order higher than the 3rd best ask price
4	PSL - place sell order lower than the 3rd best ask price
5	CBH - cancel buy order higher than the 3rd best bid price
6	CBL - cancel buy order lower than the 3rd best bid price
7	CSH - cancel sell order higher than the 3rd best ask price
8	CSL - cancel sell order lower than the 3rd best ask price
9	TBH - Trade buy order higher than the 3rd best bid price
10	TBL - Trade buy order lower than the 3rd best bid price
11	TSH - Trade sell order higher than the 3rd best ask price
12	TSL - Trade sell order lower than the 3rd best ask price

We assume a highly liquid market where market orders will always be filled, and we apply the model to a simulated order book where both limit orders and market orders are equally present.

4.4 Experiments

In this section, we conduct two experiments using the MDP model defined earlier to identify algorithmic trading strategies. We use the six trader classes defined by Kirilenko et. al. (*Kirilenko et al.* [2011]), namely High Frequency Traders, Market Makers, Opportunistic Traders, Fundamental Buyers, Fundamental Sellers and Small Traders. In general, HFTs have a set of distinctive characteristics, such as, very high activity volume throughout a trading day, frequent modification of orders, maintenance of very low inventory levels, and an agnostic orientation toward long or short positions. Market Makers are short horizon investors who follow a strategy of buying and selling a large number of contracts to stay around a relatively low target level of inventory. Opportunistic Traders sometimes behave as Market Makers buying and selling around a target position, and sometimes they act as Fundamental Traders accumulating long or short positions. Fundamental Buyers and Sellers are net buyers and sellers who accumulate positions in one single direction in general. Small Traders are the ones who have significant less activities during a typical trading day.

In the first experiment, we are interested in separating HFT strategies from Market Making and Opportunistic Trading strategies in the simulated E-Mini S&P 500 futures market. From Figure (b) in Fig. 4.1, we see that the behaviors of the Fundamental Buyers/Sellers are distinctively different from the other algorithmic traders. It is clear that classification between the HFTs and these classes of trading strategies are relatively trivial. We therefore devote our attention to separate HFT strategies from the Market Marking and the Opportunistic Trading strategies. We will start this section with a description of the design of our agent-based simulation for E-Mini S&P 500 futures market (*Paddrik et al.* [2011]). We will then use the data generated from this simulation as observations to recover the reward functions of different kinds of trading strategies, and we apply various classification methods on these trading strategies in the reward space to see whether we can accurately identify the different trading strategy classes. In the second experiment, we will focus on a specific HFT strategy called Spoofing, and try to separate this trading strategy from the other HFT strategies. In general, we test the hypothesis that reward functions can be used to effectively identify HFT strategies in both within-group and across-group situations.

4.4.1 Simulated E-Mini S&P 500 Futures Market

When simulating a system it is convenient to decompose the system into its basic parts. A financial market can be understood as a set of market participants, a trading mechanism, and a security. Agent-based models have a similar structure and include a set of agents, a topology and an environment. Through this framework it is possible to describe market participants as a set of agents with a set of actions and constraints, the market mechanism as the topology, and the exogenous flow of information relevant to market as the environment (*Macal and North* [1999]).

Using this framework, the simulation is tuned to replicate the same market conditions and variables as that of the nearest month E-Mini S&P 500 futures contract market. The agents in the model reflect closely the classes of participants observed in the actual S&P 500 E-mini futures market and the market mechanism is implemented as an electronic limit order book (see Fig. 4.1). Each class of participants is then characterized by their trade speed, position limit, order size distribution, and order price distribution. All these characterizations are based on the order book data from the E-Mini S&P 500 futures contracts provided by the Commodity Futures Trading Commission (CFTC). (see TABLE 4.2).

After the model is simulated there are two stages of validation. The first stage consists of a validation of the basic statistics for each set of agents, such as arrival rates, cancellations rates, and trade volume (see TABLE 4.3). The values observed

Table 4.2: Trader group characterization

Trader Type	**Number of Traders**	**Speed of Order**	**Position Limits**	**Market Volume**
Small	6880	2 hours	$-30 \sim 30$	1%
Fundamental Buyers	1268	1 minute	$- \propto \sim \propto$	9%
Fundamental Sellers	1276	1 minute	$- \propto \sim \propto$	9%
Market Makers	176	20 seconds	$-120 \sim 120$	10%
Opportunistic	5808	2 minutes	$-120 \sim 120$	33%
HFTs	16	0.35 seconds	$-3000 \sim 3000$	38%

in the simulation are compared to data of participants in the actual market. The second stage of validation consists of verifying that the price time-series produced by the simulation exhibits "stylized facts" (Kullmann,1999 *Kanto and Kaski* [1999]) that characterize financial data. These include heavy tailed distribution of returns[1] (Appendix A Fig. 4.7), absence of autocorrelation of returns[2] (Appendix A Fig. 4.8), volatility clustering[3] (Appendix A Fig. 4.9), and aggregational normality[4] (Appendix A Fig. 4.10).

[1] The empirical distributions of financial returns and log-returns are fat-tailed. It has been widely observed starting from Mandelbrot *Mandelbrot* [1963] and Gopikrishnan et al. *Meyer and Stanley* [1999]. Even though it is the most widely acknowledged and the most elementary one, this stylized fact is not easily met by all financial modelling.

[2] There is no evidence of correlation between successive returns. As it is pointed out by Pagan *Pagan* [1996] and Cont et al. *Potters and Bouchaud* [1997], the autocorrelation function decays very rapidly to zero, even for a few lags of 1 minute.

[3] Absolute returns or squared returns exhibit a long-range slowly decaying autocorrelation function. It is first formulated by Mandelbrot *Mandelbrot* [1963] as "large changes tend to be followed by large changes of either sign, and small changes tend to be followed by small changes".

[4] As the time scale increases, the fat-tail property diminishes and the return distribution approaches Gaussian distribution. This cross-over phenomenon is documented by Kullmann et al. *Kanto and Kaski* [1999].

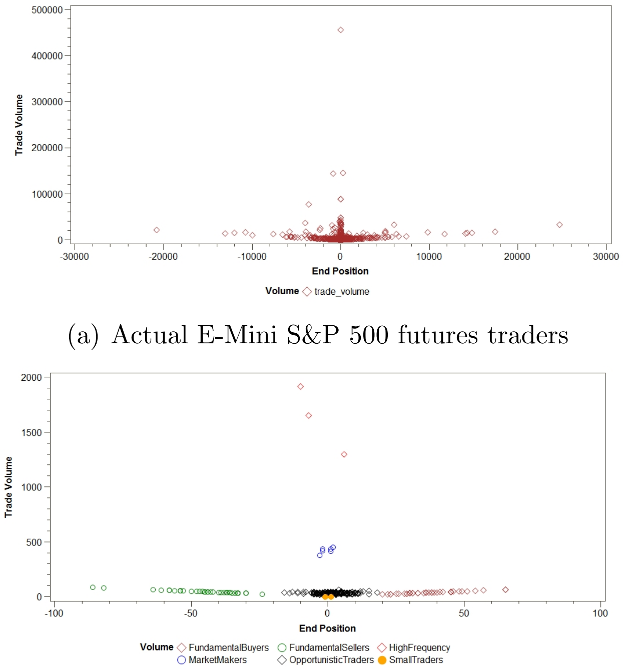
Table 4.3: Trader group validation

Trader	**Simulated**	**Actual**	**Rate-Simulated**	**Rate-Actual**
Type	**Volume**	**Volume**	**Cancellation**	**Cancellation**
Small	1%	1%	40%	20 – 40%
Fundamental Buyers	10%	9%	44%	20 – 40%
Fundamental Sellers	10%	9%	44%	20 – 40%
Market Makers	10%	10%	35%	20 – 40%
Opportunistic	31%	33%	50%	40 – 60%
HFTs	38%	38%	77%	70 – 80%

4.4.2 Identify HFTs from Market Making and Opportunistic Trading Strategies

Using the IRL model that we formulated above, we learn the corresponding reward functions from 18 simulation runs where each run consists of approximately 300,000 activities including orders, cancellations, and trades. We then use the different classification methods on the rewards to see how well we can separate the HFTs from the other two different trading strategies.

From Fig. 4.2, we see that reward space has a very succinct structure, which tends to confirm the observations made in (*Ramachandran and Amir* [2007], and *Qiao and Beling* [2011]) that policies are generally noisier than reward functions. We also observe that the reward function converges faster than the policy as observation time increases. In addition to the lack of robustness in policy space, the lack of portability of learned policies is another important drawback in the use of policies to characterize trading strategies. Furthermore, the fact that actions are notional makes it unclear how one could use policies to measure differences among trading strategies. Hence, our study focuses attention on reward space. Using Principal Component



(a) Actual E-Mini S&P 500 futures traders

(b) Simulated E-Mini S&P 500 futures traders

Figure 4.1: **E-Mini S&P 500 Actual vs. Simulated** E-Mini S&P 500 futures traders' end-of-day position vs. trading volume.

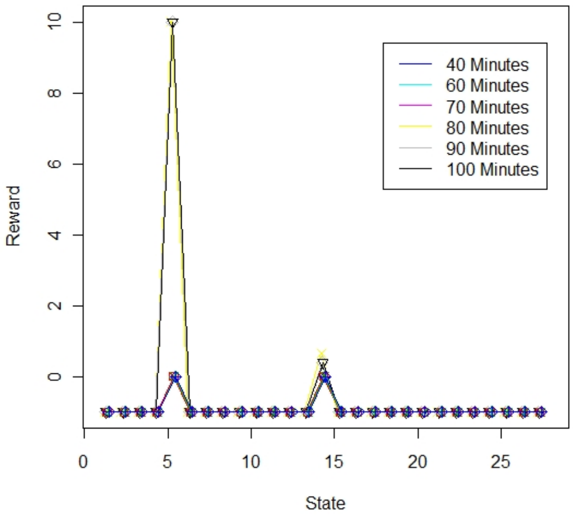


Figure 4.2: **Reward Space Convergence** For a series of observations of a particular trader, as time interval increases, the reward at state 5 converges from 10 to 0, and the reward at state 14 converges from 0.66 to 0. At all the other states, the reward remains at -1.

dimension shrinkage method, we are able to compare the two trading strategies in a three dimensional space visually. Fig. 3 and Fig. 4 show a clear separation of the HFT strategies from the other two classes of trading strategies.

Three different classification methods are then applied on the learned reward functions. From the comparison (Table 4.4) of the results of the three different classification methods, i.e. Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), and Multi-Gaussian Discriminant Analysis (MDA). The two nonlinear methods perform better than the linear one. It can be seen from the visualization reward distributions. The highest classification accuracy achieved by all three methods is 100%. In general, all of them achieved relatively high accuracy in the range between 95% and 100%. The sensitivity (i.e. true positive) is in the range between 89% and 94%. The specificity (i.e. true negative) is in general better, and it is 100% across all three classification methods.

Table 4.4: Trading Strategy Classification Results

High Frequency Traders vs. Opportunistic Traders			
	LDA	QDA	MDA
Accuracy	97%	100%	97%
Sensitivity	94%	100%	94%
Specificity	100%	100%	100%
High Frequency Traders vs. Market Makers			
	LDA	QDA	MDA
Accuracy	95%	97%	95%
Sensitivity	88%	94%	88%
Specificity	100%	100%	100%
Opportunistic Traders vs. Market Makers			
	LDA	QDA	MDA
Accuracy	70%	75%	83%
Sensitivity	39%	100%	72%
Specificity	100%	100%	94%

The results using this model for separating Opportunistic Traders vs. Market Makers are not as good compared with those between HFT vs. Market Making and HFT vs. Opportunistic strategies (TABLE 4.4). From the classification results,

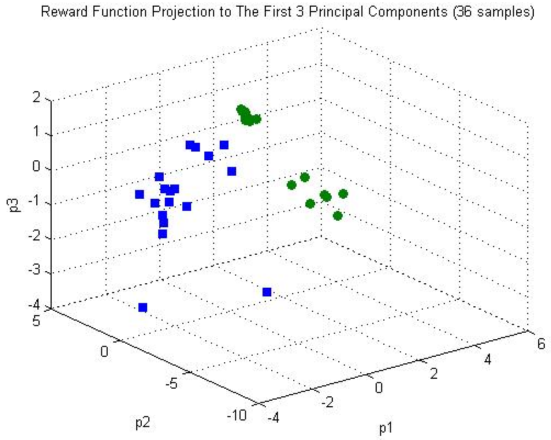


Figure 4.3: **Reward Space Clustering** Reward space clustering between HFT strategies vs. Opportunistic Trading strategies

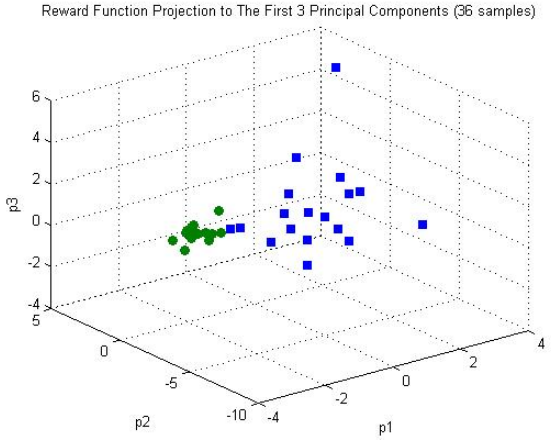


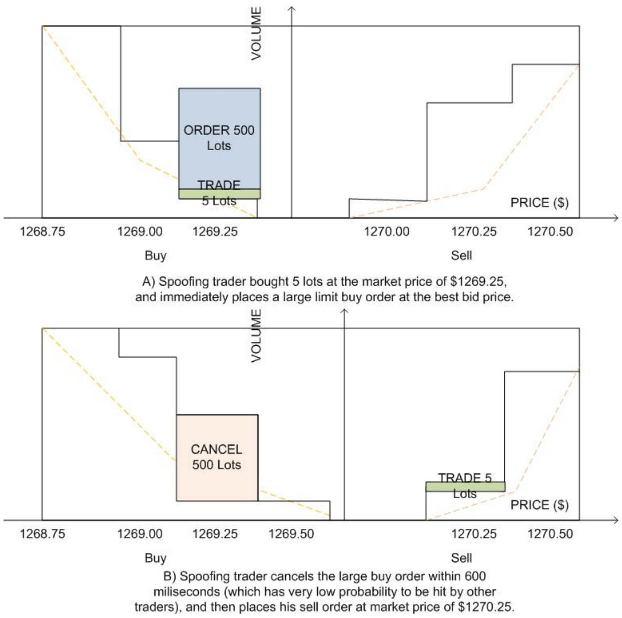
Figure 4.4: **Reward Space Clustering** Reward space clustering between HFT strategies vs. Market Marking Trading strategies

we can see that MDA classification performed the best and achieved 83% accuracy, 72% sensitivity, and 94% specificity. However, this result is expected in that the current order book model is specifically targeted at characterizing HFT strategies. In order to achieve better results between Opportunistic and Market Making strategies, we will have to consider other factors that can best characterize the Opportunistic Trader's behaviors. Further study of the these two classes' behaviors will be critical in improving the classification performance between these two classes' of trading strategies.

4.4.3 Identify A Spoofing Strategy from Other HFTs

In this section, we are interested in one particular manipulative strategy in the High Frequency Trading paradigm: Spoofing, which sometimes is referred to as "Hype and Dump" manipulation (*Aggarwal and Wu* [2003], and *Eren and Ozsoylev* [2006]). Both empirical and theoretical evidence show that the manipulators can profit from this manipulative trading practice. In this scheme, the manipulator artificially inflates the asset price through promotion in order to sell at the inflated price, or deflates the asset price through false hype in order to buy at the deflated price. One concrete example of this trading strategy is illustrated in Fig. 4.5 A. Suppose a trader intends to sell 5 shares of an asset, he first submits a large limit-buy order with a bid at or below the current market price making the buy side of the order book seem large. Based on the market information infusion process or supply-demand theory, the market price will tend to move higher. And the spoofing trader will then submit a market-sell order and consequently cancels the original buy order as it is illustrated in Fig. 4.5 B.

This manipulative practice is illegal under the U.S. securities law, yet it has been frequently discovered in both equity and futures markets. Our simulated spoofing trading strategy is based on our observations on a futures market where a trader



A) Spoofing trader bought 5 lots at the market price of $1269.25, and immediately places a large limit buy order at the best bid price.

B) Spoofing trader cancels the large buy order within 600 miliseconds (which has very low probability to be hit by other traders), and then places his sell order at market price of $1270.25.

Figure 4.5: **Spoofing Example** Market microstructure-based manipulation example: buy spoofing

repeatedly exercised the spoofing pattern over a month period. Due to the nature of the CFTC investigation, we will not be able to disclose the specifics for publications, but we are able to capture the deterministic nature of their strategy in the simulation. Specifically, in our discrete time agent-based simulation model, we design a spoofing agent as one of the HFTs except that it deploys additional trading plots: first they engage in a signaling game and then a trading game. In the signaling stage, the spoofing agent places a large buy order at the best bid price. After 600 milliseconds (it is designed with relative to the speed of HFT's cancellation rate), it transitions into the trading stage where they cancel the original limit order and places a market order. Since the trader is a HFT, in order to maintain the constraint of his inventory, the trader will have to spoof and trade on the other side of the book at certain point.

As we have done for the general simulation, we run 18 times of the simulation to generate 18 market instances. And then we randomly select 18 samples for all the general HFT trading strategies, and select 18 samples for the Spoofing trading strategy for IRL. We then obtain 36 reward functions with labels and apply three classification methods on these samples, and obtain results in TABLE 4.5. From these results, we see that we can identify the Spoofing strategy from the other HFT strategies with at least 92% accuracy. We also observe again that the non-linear classification rule works better in general.

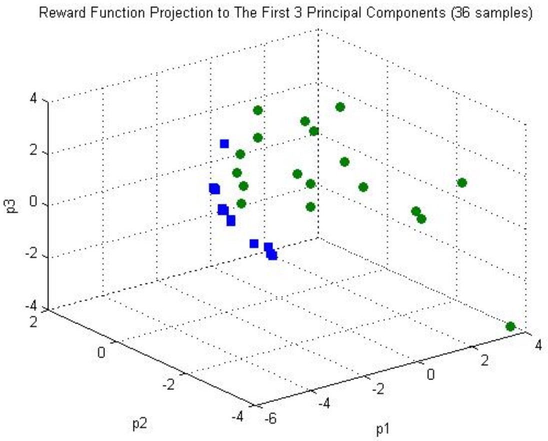


Figure 4.6: **Reward Clustering** Reward space clustering between HFT strategies vs. the Spoofing strategy

Table 4.5: Spoofing Trading Strategy vs. Other HFT Classification Results
Market Makers vs. Opportunistic Traders

	LDA	QDA	MDA
Accuracy	92%	97%	97%
Sensitivity	100%	100%	100%
Specificity	83%	94%	94%

4.5 Conclusions

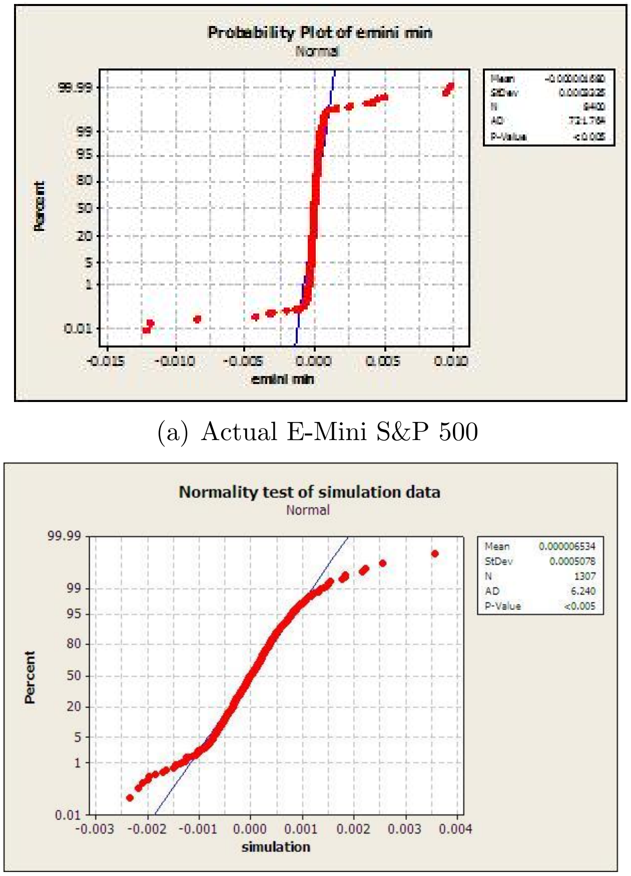
The primary focus of this paper is to use Inverse Reinforcement Learning method to capture the key characteristics of the HFT strategies. From the results using a linear programming method for solving IRL with simulated E-Mini S&P 500 futures market data, we attain a high identification accuracy ranging between 95% and 100% for the targeted trading strategy class, namely High Frequency Trading from Market

Making and Opportunistic strategies. We also show that the algorithm can accurately (between 92% and 95%) identify a particular type of HFT spoofing strategy from other HFT strategies. And we also argue that the reward space is better suited for identification of trading strategies than the policy space.

We investigate and address the issues of modeling algorithmic trading strategies using IRL models such as, addressing non-deterministic nature of the observed policies in learning, constructing efficient MDP models to capture order book dynamics, achieving better identification accuracy in reward space, etc. With a reliably validated agent based market simulation, we capture the essential characteristics of the algorithmic trading strategies. The practical implication of this research is that we demonstrate that the market operators and regulators can use this behavior based learning approach to perform trader behavior based profiling, and consequently monitor the emergence of new HFTs and study their impact to the market.

Here is a list of future research to be done:

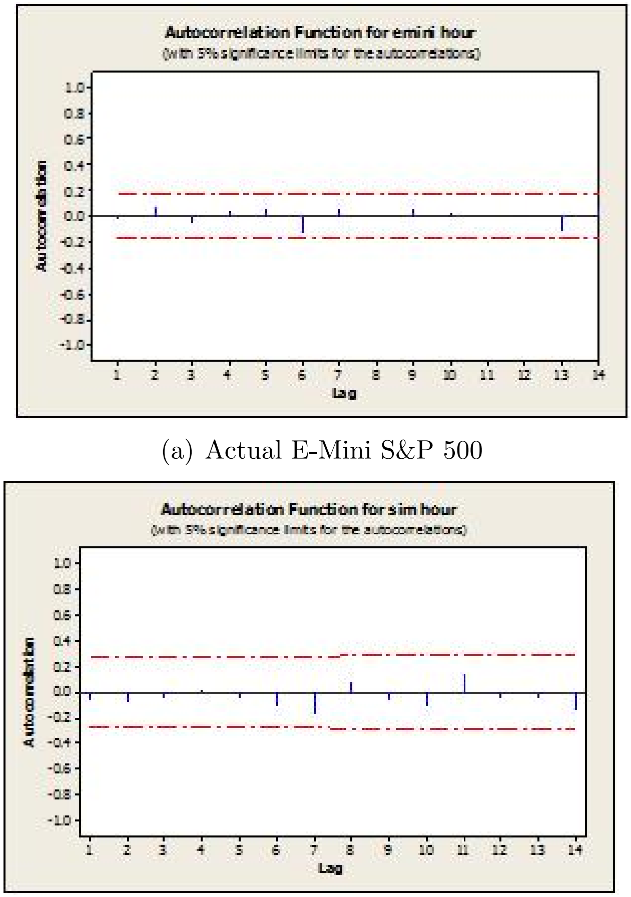
- Apply both the linear programming approach and maximum likelihood approaches to the simulated trading strategies and the Spoofing data collected from the actual market, and compare the results of these two approaches in terms of identification accuracy.
- Create simulation agent based on reward functions learned from the actual market observations, and study the new trading strategy's impact to the market quality.



(a) Actual E-Mini S&P 500

(b) Simulated E-Mini S&P 500

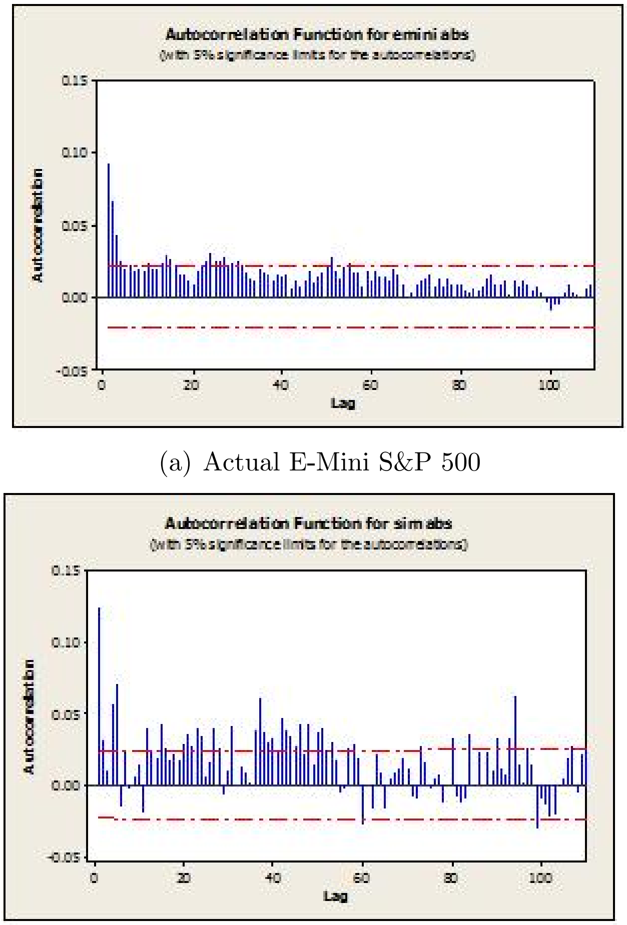
Figure 4.7: **E-Mini S&P 500 Heavy Tailed Distribution of Returns** From panel (a) and (b), we see normality tests of returns for both actual and simulated E-Mini S&P 500 show deviation from Gaussian distribution toward both tails.



(a) Actual E-Mini S&P 500

(b) Simulated E-Mini S&P 500

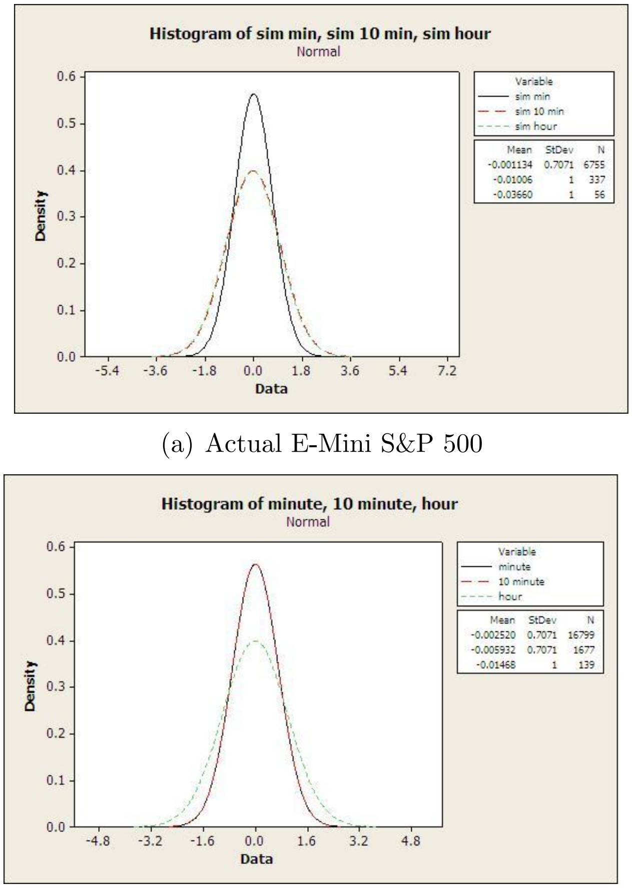
Figure 4.8: **E-Mini S&P 500 Absence of Autocorrelation of Returns** From panel (a) and (b), we see autocorrelation of returns for both actual and simulated E-Mini S&P 500 are all close to zero within 95% confidence level.



(a) Actual E-Mini S&P 500

(b) Simulated E-Mini S&P 500

Figure 4.9: **E-Mini S&P 500 Autocorrelation Clustering** From panel (a) and (b), we see returns decay slowly for both actual and simulated market. Even though there are few lags outside the 95% confidence lines, the simulation decaying pattern closely resembles that of the actual market as lag increases.



(a) Actual E-Mini S&P 500

(b) Simulated E-Mini S&P 500

Figure 4.10: **E-Mini S&P 500 Aggregational Normality** As shown in panel (a) and (b), returns approaches to Gaussian distribution as the time scale increase for both actual and the simulated market.

CHAPTER V

Gaussian Process Based Trading Strategy Identification

5.1 Introduction

Financial market has changed dramatically in recent years. These changes reflect the culmination of a decade-long trend from a market structure with primarily manual floor trading to a market structure with primarily computer automated trading. A primary driving force of this accelerated transformation has been the increased evolution of technologies for generating, routing, and executing orders. These technologies have dramatically improved the speed, capacity, and sophistication of the trading functions that are available to market participants.

High quality trading markets promote capital raising and capital allocation by establishing prices for securities and by enabling investors to enter and exit their positions in securities whenever they wish to do so. The one important feature of all kinds of Algorithmic trading strategies is discovering the underlying persistent tradable phenomena and generating trading opportunities. These trading opportunities range from microsecond price movement allowing a trader to benefit from market-making trades, to several minute-long strategies that trade on momentum forecast-ed by micro-structure theories, to several hour-long market moves surrounding recurring events and deviations from statistical relationship (*Aldridge* [2010]). Algorithmic traders then design their trading algorithms and systems aiming to generate signals that result in consistent positive outcomes under different market circumstances. These market circumstances can be described in high frequency terms. Different strategies may target different frequencies, and the profitability of a trading strategy is often measured by certain return metric. The most commonly used measure is Sharpe ratio, a risk-adjusted return metric first proposed by Sharpe (*Sharpe* [1966]).

In this study, we model trading behavior of different market participants from the solution to inverse Markov decision process (MDP). We try to describe how the traders are able to take actions in a highly uncertain environment to reach its return goals at different horizons. This task can be solved by using dynamic programming (DP) and reinforcement learning (RL) based on MDP. The model accounts for trader's preferences and expectations of uncertain state variables. In a general MDP modeling setting we describe these variables in two spaces: state space and action space. From trading decision perspective, we can parameterize learning agents using the reward functions that depend on state and action. We consider the market dynamics in view of the learning agents' subjective beliefs. The agents perform DP/RL through a sense, trial and learn cycle. First, the agents gain state information from sensory input. Based on the current state, knowledge and goals, agents find and choose a best action. Upon the new feedback, the agents learn to update the knowledge with a goal of maximizing their accumulative expected reward. In discrete-valued state and action problem space, DP and RL methods take similar techniques involving policy iteration and value iteration algorithms (*Bertsekas* [2007], and *Sutton and Barto* [1998]) to solve the MDP problems. Formalisms for solving forward problems of RL are often divided into model based and model free approaches (*D. et al.* [2005], and *Sutton and Barto* [1998]).

As it is framed by Abbeel et al. (*Abbeel and Ng* [2004]) under the Inverse Reinforcement Learning (IRL) framework, the entire field of reinforcement learning is founded on the presupposition that the reward function, rather than the policy is the most succinct, robust, and transferable definition of the task. However, the reward function is often difficult to know in advance for some real-world tasks. Such difficulties may arise in the following situations: 1) We have no experience to tackle the problem; 2) We have experience but can not interpret the reward function explicitly; 3) The problem we solve may be interacting with the adversarial decision makers who make all their effort to keep the reward function secrete. In comparison with the difficulty of accessing the true reward function, it is easier to observe some other agent's (or we call teacher/expert) behavior showing how to solve the problems. Hence, we have the motivation to learn from observations. Technical approaches to learning from observations generally fall into two broad categories *Dey and Srinivasa* [2009]. The first one, called imitation learning, attempts to use supervised learning to predict actions directly from observations of features of the environment, which is unstable and vulnerable to highly uncertain environment. The other is concerned with how to learn the reward function that characterizes the agent's objectives and preferences in MDP, which is called IRL (*Ng and Russel* [2000]).

IRL was first introduced in machine learning research by Ng and Russel (*Ng and Russel* [2000]), under the assumption that the reward $r = \omega^T\phi(s)$, where ω is a coefficient vector and $\phi(s) : s \to [0,1]^k$ is a function mapping state s to a k dimensional vector. Then they formulate IRL problem as an optimization problem to maximize the sum of differences between the quality of the optimal action and the quality of the next-best action. Based on this linear approximation of reward function, other algorithms have been developed or integrated into apprenticeship learning. The principal idea of apprenticeship learning using IRL is to search mixed solutions in a space of learned policies with the goal that the accumulative feature expectation is near that of the expert (*Abbeel and Ng* [2004] and *Bowling and Schapire* [2008]).

Other algorithms have also developed under the IRL framework. A game-theoretic approach to apprenticeship learning using IRL is developed in the context of a two-player zero-sum game in which the apprentice chooses a policy and the environment chooses a reward function (*Schapire* [2008]). Another algorithm for IRL is policy matching in which the loss function penalizing deviations from expert's policy that is minimized by tuning the parameters of reward functions (*Neu and Szepesvari* [2007]). Maximum entropy IRL is proposed in the context of modeling real-world navigation and driving behaviors (*Bagnell and Dey* [2008]). The algorithms for apprenticeship learning using IRL do not actually aim to recover the reward function but only the optimal policy. Ramachandran and Amir consider IRL from a Bayesian perspective without assuming the linear approximation of the reward function (*Deepak and Eyal* [2007]). Their model interprets the observations from the expert as the evidence that is used to obtain a posterior distribution over reward using Markov Chain Monte Carlo simulation. Recent theoretical works on IRL such as the framework of linear-solvable MDP (*Dvijotham and Todorov* [2010]), the bootstrap learning (*Boularias and Chaib-draa* [2010]) and feature construction (*Popovic and Koltun* [2010]), have also published to improve the learning performance. IRL has also been successfully applied to many real-world problems, such as automatic control of helicopter flight (*Coates and Ng* [2010]) and motion control of an animation system in computer graphics (*Lee and Zoran* [2010]).

We apply an Gaussian process based IRL (GPIRL) model proposed by Qiao et al. (*Qiao and Beling* [2011]) to learning trading behavior of a particular financial market. In this GPIRL, a Gaussian prior is assigned on the reward function and the reward function is treated as a Gaussian process. This approach is similar in perspective to that Ramachandran and Eyal (*Deepak and Eyal* [2007]), who view the state-action samples from the expert as the evidence that will be used to update a prior on the reward function, under a Bayesian framework. The solution (*Deepak* *and Eyal* [2007]) depends on non-convex optimization using Markov Chain Monte Carlo simulation. Moreover, the ill-posed nature of the inverse learning problem also presents difficulties. Multiple reward functions may yield the same optimal policy, and there may be multiple observations at a state given the true reward function. GPIRL model aims to deal with the ill-posed nature by applying Bayesian inference and preference graphs. One of the main novelties of this approach is that it not only bears a probabilistically coherent view but also is computationally tractable.

Due to the dynamic nature of the financial markets, it is possible to postulate a priori a relationship between the market variables we observe and those we wish to predict. The main contributions of this study can be summarized as follows,

1. We model the reward function using Gaussian process, which offers the advantage that is relatively insensitive to the number of the observations and it performs better than other algorithms when we only have partial market observations on the trading strategies we try to recover.

2. We apply preference graphs to address non-deterministic nature of the observed trading behaviors, reducing the uncertainty and computation burden caused by the ill-posed nature of the inverse learning problem. We build new likelihood functions for preference graphs and prove the effectiveness of these formulations in experiments.

3. We also perform clustering of behavior representation of the trading strategies, and we make connections between the existing summary statistic based trader type classification approach (*Kirilenko et al.* [2011]) with our behavior based classification approach. We propose a quantitative behavior approach to categorizing Algorithmic trading strategies using weighted scores over time.

The rest of this paper is organized as follows: First we go through related work and preliminaries in Section 5.2. In Section 5.3, we discuss IRL formulations and provide a Bayesian probabilistic model to infer the reward function using Gaussian processes. We apply the GPIRL algorithm to the most active financial market E-Mini S&P 500 Futures market as experiments in Section 5.4. We show that the GPIRL algorithm can accurately capture algorithmic trading behaviors based on observations taken from the high frequency data. We also compare our behavior based classification results with the results from Kirilenko et al. (*Kirilenko et al.* [2011]), and show a consistency and improvement of our behavior approach. Finally we offer concluding remarks in Section 5.5 about GPIRL and its application.

5.2 Background and Related Work

Our solution to the inference from observations is based on the general Inverse Reinforcement Learning framework. In this section, we first introduce notations that will be used throughout this paper, and we then discuss some essential facts that we need from the theory of Markov Decision Processes.

5.2.1 Inverse MDP Problem

The primary aim of our trading behavior based learning approach is to uncover decision maker's policies and reward functions through observations of an expert whose decision process is modeled as Markov Decision Processes. In this paper, we restrict our attention to finite countable MDP for easy exposition, but our approach can be extended to continuous problems if so desires. A discounted finite MDP is defined as a tuple $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, where

- $\mathcal{S} = \{s_n\}_{n=1}^{N}$ is a set of N states. Let $\mathcal{N} = \{1, 2, \cdots, N\}$.
- $\mathcal{A} = \{a_m\}_{m=1}^{M}$ is a set of M actions. Let $\mathcal{M} = \{1, 2, \cdots, M\}$.
- $\mathcal{P} = \{\mathbf{P}_{a_m}\}_{m=1}^{M}$ is a set of state transition probabilities (Here $\mathbf{P}_{a_m}$ is a $N \times N$ matrix. Each row, denoted as $\mathbf{P}_{a_m}(s_n, :)$, contains the transition probabilities

upon taking action a_m in state s_n. The entry $\mathbf{P}_{a_m}(s_n, s_{n'})$ is the probability of moving to state $s_{n'}, n' \in \mathcal{N}$ in the next stage.).

- $\gamma \in [0, 1]$ is a discount factor.
- r denotes the reward function, mapping from $\mathcal{S} \times \mathcal{A}$ to $\Re$ with the property that

$$r(s_n, a_m) \triangleq \sum_{n' \in \mathcal{N}} \mathbf{P}_{a_m}(s_n, s_{n'}) r(s_n, a_m, s_{n'})$$

where $r(s_n, a_m, s)$ denotes the function giving the reward of moving to next state $s_{n'}$ after taking action a_m in current state s_n. The reward function $r(s_n, a_m)$ may be further reduced to $r(s_n)$, if we neglect the action's influence.

In MDP, an agent selects an action at each sequential stage, and we define a *policy* (*behavior*) as the way the actions are selected by a decision maker/agent. Hence it can be described as a mapping between state and action i.e. a random state-action sequence $(s^0, a^0, s^1, a^1, \cdots s^t, a^t, \cdots)$, [1] where s^{t+1} is connected to (s^t, a^t) by $\mathbf{P}_{a^t}(s^t, s^{t+1})$. The policy, which makes the agent reach its goal, is called *proper policy*.

We also define rational agents as those that behave according to the optimal decision rule where each action selected at any stage maximizes the value function. The *value function* for a policy π evaluated at any state s^0 is given as $V^\pi(s^0) = E[\sum_{t=0}^{\infty} \gamma^t r(s^t, a^t)|\pi]$. This expectation is over the distribution of the state sequence $\{s^0, s^1, ...\}$ given policy $\pi = \{\mu^0, \mu^1, \cdots\}$, where $a^t = \mu^t(s^t)$, $\mu^t(s^t) \in U(s^t)$ and $U(s^t) \subset \mathcal{A}$. The objective at state s is to choose a policy maximizing the value of $V^\pi(s)$. Similarly, there is another function called *Q-functions* (*Q-factors*) that judges how good an action is performed in a given state. Notation $Q^\pi(s, a)$ represents the

[1] Superscripts index time. E.g. s^t and a^t, with the upper-index $t \in \{1, 2, \cdots\}$, denote state and action at t-th horizon stage, while s_n (or a_m) represents the n-th state (or m-th action) in $\mathcal{S}$ (or $\mathcal{A}$).

expected return from state s, taking action a and thereafter following policy π.

In the infinite-horizon case, the stationarity Markovian structure of the problem implies that the only variable that affects the agent's decision rule and corresponding value function should be time invariant. We then have the essential theory of MDPs *R.* [1957] as follows,

Theorem V.1 (Bellman Equations)**.** *Given a stationary policy π, $\forall n \in \mathcal{N}, m \in \mathcal{M}$, $V^{\pi}(s_n)$ and $Q^{\pi}(s_n, a_m)$ satisfy*

$$\begin{aligned} V^{\pi}(s_n) &= r(s_n, \pi(s_n)) + \gamma \sum_{n' \in \mathcal{N}} \mathbf{P}_{\pi(s_n)}(s_n, s_{n'}) V^{\pi}(s_{n'}), \\ Q^{\pi}(s_n, a_m) &= r(s_n, a_m) + \gamma \sum_{n' \in \mathcal{N}} \mathbf{P}_{a_m}(s_n, s_{n'}) V^{\pi}(s_{n'}). \end{aligned}$$

Theorem V.2 (Bellman Optimality)**.** *π is optimal if and only if, $\forall n \in \mathcal{N}$, $\pi(s_n) \in \arg\max_{a \in \mathcal{A}} Q^{\pi}(s, a)$.*

Based on the above definitions of MDP, we further introduce the *inverse Markov Decision Process (IMDP).*

Definition V.3. An IMDP model, denoted as $M_I = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, \mathcal{O})$, contains MDP variables such as, state set $\mathcal{S}$, action set $\mathcal{A}$, state transition probability set $\mathcal{P}$ and the discount factor γ. The variable $\mathcal{O}$ is a set of observations sampled from the decision-making process.

The set $\mathcal{O}$ can be viewed as a subset of the Cartesian product of $\hat{\mathcal{S}}$ and $\hat{\mathcal{A}}$, where $\hat{\mathcal{S}} \subset \mathcal{S}$ and $\hat{\mathcal{A}} \subset \mathcal{A}$. So $\forall s \in \hat{\mathcal{S}}$, there is at least one action $a \in \hat{\mathcal{A}}$ providing $(s, a) \in \mathcal{O}$. We treat every $(s, a) \in \mathcal{O}$ as optimal in the expert's decision making process. The goal of IRL is to learn the reward function of the MDP model that generates $\mathcal{O}$.

5.3 Gaussian Process for Generalized IRL Problem

We now turn to an IRL problem that deals with observations from a decision making process in which the reward function has been contaminated by Gaussian noise. In particular, we assume that the reward vector can be modeled as $r+\mathcal{N}(0,\sigma^2)$, where $\mathcal{N}(0,\sigma^2)$ is Gaussian noise. In the financial trading problem setting, we may observe certain trading behavior over a period of time, but we may not observe the complete polices behind a particular trading strategy. As we have discussed earlier, different trading strategies tend to look at different time horizons. Therefore the observation period becomes critical in the learning process. Furthermore, there are potentially two kinds of errors may be introduced into our observations: The first source of errors may be introduced during our modeling process. Resolution of these discrete models will introduce errors into our observations. The second source of errors potentially comes from the strategy execution process. Due to the uncertainty of market movements, execution errors will occur and eventually be transfered into our observations in determining the true policy. Overall there are two kinds of challenges in this learning problem: One is the uncertainty about reward functions given the observation of decision behavior and the other is the ambiguity of observing multiple actions at a state.

Qiao and Belling (*Qiao and Beling* [2011]) argue two different modeling techniques in learning reward functions. To lessen the ambiguity of observing multiple actions at a state, they argue that Bayesian inference should be the basis for understanding the agent's preferences over the action space. This argument is reasonable because the goal of IRL is to learn the reward subjectively perceived by the decision maker from whom we have collected the observation data. The intuition is that the decision makers will select some actions at a given state because they prefer these actions to others. These preferences among countable actions can be used to represent the multiple observations at one state. In the following, we introduce the preference theory for IMDP model first. Then, we formalize the ideas here.

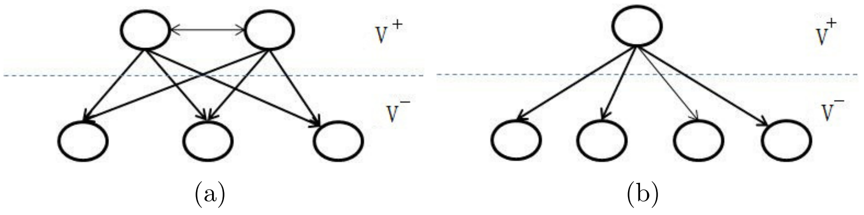


Figure 5.1: **Examples of Preference Graph** (a) An example of observing two actions at a state. (b) An example of unique observation at a state.

5.3.1 Action Preference Learning

In this section, we first define the action preference relationship and action preference graph. At state s_n, $\forall \hat{a}, \check{a} \in \mathcal{A}$, we define the *action preference relation* as:

1. Action $\hat{a}$ is weakly preferred to $\check{a}$, denoted as $\hat{a} \succeq_{s_n} \check{a}$, if $Q(s_n, \hat{a}) \geq Q(s_n, \check{a})$;
2. Action $\hat{a}$ is strictly preferred to $\check{a}$, denoted as $\hat{a} \succ_{s_n} \check{a}$, if $Q(s_n, \hat{a}) > Q(s_n, \check{a})$;
3. Action $\hat{a}$ is equivalent to $\check{a}$, denoted as $\hat{a} \sim_{s_n} \check{a}$, if and only if $\hat{a} \succeq_{s_n} \check{a}$ and $\check{a} \succeq_{s_n} \hat{a}$.

An *action preference graph* is a simple directed graph showing preference relations among the countable actions at a given state. At state s_n, its action preference graph $G_n = (\mathcal{V}_n, \mathcal{E}_n)$ comprising a set $\mathcal{V}_n$ of nodes together with a set $\mathcal{E}_n$ of edges. About node and edge in graph G_n, let us define

1. Each node represents an action in $\mathcal{A}$. Define a one-to-one mapping $\varphi : \mathcal{V}_n \to \mathcal{A}$.
2. Each edge indicates a preference relation.

Furthermore, we give Lemma (V.4) as a rule to build the preference graph, and then we show how to draw a preference graph at state s_n.

Lemma V.4. *At state s_n, if action $\hat{a}$ is observed, we have these preference relations: $\hat{a} \succ_{s_n} \check{a}, \forall \check{a} \in \mathcal{A} \setminus \{\hat{a}\}$.*

It is therefore straightforward to show the following according to Bellman optimality. $\hat{a}$ is observed if and only if $\hat{a} \in \arg\max_{a \in \mathcal{A}} Q(s_n, a)$. Therefore, we have

$$Q(s_n, \hat{a}) > Q(s_n, \check{a}), \ \forall \check{a} \in \mathcal{A} \setminus \{\hat{a}\}$$

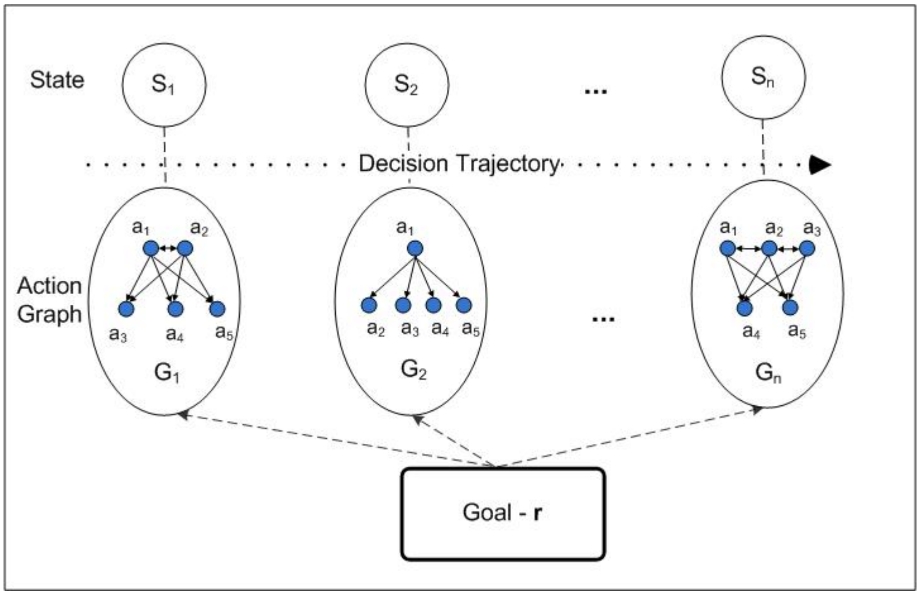
According to the definition on preference relations, it follows that if $Q(s_n, \hat{a}) > Q(s_n, \check{a})$, we have $\hat{a} \succ_{s_n} \check{a}$. Hence, we can show that preference relationship has the following properties:

1. If $\hat{a}, \check{a} \in \mathcal{A}$, then at state s_n either $\hat{a} \succeq_{s_n} \check{a}$ or $\check{a} \succeq_{s_n} \hat{a}$.

2. If $\hat{a} \succeq_{s_n} \check{a}$ or $\check{a} \succeq_{s_n} \tilde{a}$, then $\hat{a} \succeq_{s_n} \tilde{a}$.

At this point, we have a simple representation of the action preference graph that is constructed by a two-layer directed graph. We may have either Figure (5.1) (a) multiple actions at s_n and Figure (5.1) (b) unique action at s_n. In this two-layer directed graph, the top layer $\mathcal{V}_n^+$ is a set of nodes representing the observed actions and the bottom layer $\mathcal{V}_n^-$ has the nodes denoting other actions. The edge in the edge set $\mathcal{E}_n$ can be represented by a formulation of its beginning node u and ending node v. We write the k-th edge as $(u \to v)_k$ if $u \in \mathcal{V}_n^+, v \in \mathcal{V}_n^-$, or the l-th edge $(u \leftrightarrow v)_l$ if $u \in \mathcal{V}_n^-, v \in \mathcal{V}_n^-$. Recall the mapping between $\mathcal{V}_n$ and $\mathcal{A}$, the representation $u \to v$ indicates that action $\varphi(u)$ is preferred to $\varphi(v)$. Similarly, $u \leftrightarrow v$ means that action $\varphi(u)$ is equivalent to $\varphi(v)$.

In the context of financial trading decision process, we may observe multiple actions from one particular trader under certain market conditions. That is to say that the observation data $\mathcal{O}$ may be multiple decision trajectories generated by non-deterministic policies. To address IRL problems in those cases, Qiao and Belling (*Qiao and Beling* [2011]) propose to process $\mathcal{O}$ into the form of pairs of state and preference graph, e.g. the representation shown in Figure (5.2), and then we apply Bayesian inference using the new formulation.



(a)

Figure 5.2: Proposed observation structure for MDP.

In order to apply Bayesian inference on reward function, we need to show equivalence of evidence for inference of reward function between the use of decision trajectories and the independent pairs of state and preference graph (See Proposition V.5). We need the following proposition (see a proof from Appendix Gaussian Processes).

Proposition V.5. *The observation dataset $\mathcal{O}_1$ is given as a set of decision trajectories. Assume independence among the observed decision trajectories. Observation of policy at a state can be specified by an action preference graph. Let $\mathcal{O}_2$ be the set of independent pairs of state and action preference graph, which is written as $\mathcal{O}_2 = \{(s_n, G_n)\}_{n=1}^{N}$. The inference of reward function drawn from $\mathcal{O}_1$ and $\mathcal{O}_2$ is identical. There is a constant factor C that makes likelihood function $p(\mathcal{O}_1|r) = Cp(\mathcal{O}_2|r)$.*

Based on Proposition V.5, we can represent $\mathcal{O}$ as shown in Figure (5.2). At state s_n, its action preference graph is constructed by a two-layer directed graph: a set of nodes $\mathcal{V}_n^+$ in the top layer and a set of nodes $\mathcal{V}_n^-$ in the bottom layer. Under the non-deterministic policy assumption, we adopt a reward structure depending on both state and action.

5.3.2 Bayesian Inference of Gaussian Reward Process

We adopt a variation of likelihood function proposed by Chu and Ghahramani in *Wei and Zoubin* [2005] to capture the strict and equivalent preference relations. The likelihood function for the reward without noise is as follows,

$$p_{\text{ideal}}(\hat{a} \succ_{s_n} \check{a} | \mathbf{r}_{\hat{a}}(s_n), \mathbf{r}_{\check{a}}(s_n)) = \begin{cases} 1 & \text{if } Q(s_n, \hat{a}, \mathbf{r}) > Q(s_n, \check{a}, \mathbf{r}) \\ \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

$$p((\hat{a} \sim_{s_n} \hat{a}')_l | \mathbf{r}) \propto e^{-\frac{1}{2}(Q(s_n,\hat{a}) - Q(s_n,\hat{a}'))^2} = e^{-\sigma^2 f^2(\mathbf{r}, s_n, l)} \tag{5.2}$$

As we stated earlier, if we model the reward functions as being contaminated with Gaussian noise that has zero mean and unknown variance σ^2, we can then define the likelihood function for both k-th strict preference relation and l-th equivalent preference relation. Finally we can formulate the following proposition:

Proposition V.6. *The likelihood function, giving the evidence of the observation data $\mathcal{O}$ in the form of pairs of state and action preference graph, is calculated by*

$$p(\mathcal{G}|\mathcal{S}, \boldsymbol{r}) = \prod_{n=1}^{N} p(G_n | s_n, \boldsymbol{r}) = \prod_{n=1}^{N} \prod_{k=1}^{n_n} \Phi(f(\boldsymbol{r}, s_n, k)) e^{\sum_{n=1}^{N} \sum_{l=1}^{m_n} -\sigma^2 f^2(\boldsymbol{r}, s_n, l)} \tag{5.3}$$

In conclusion, the probabilistic IRL model is controlled by kernel parameters κ_{a_m} and σ_{a_m} for computing the covariance matrix of reward realizations, and σ to tune the noise level in the likelihood function. We put these parameters into the hyper-parameter vector $\boldsymbol{\theta} = (\kappa_{a_m}, \sigma_{a_m}, \sigma)$. More often than not, we do not know beforehand the knowledge about the hyper-parameters. And then we can apply maximum a posterior estimate to evaluate the hyper-parameters.

Essentially we now have a hierarchical model. At the lowest level we have reward function values encoded as a parameter vector $\mathbf{r}$. At the top level we have hyperparameters in $\boldsymbol{\theta}$ controlling the distribution of the parameters. Inference takes place one level at a time. At the bottom level, the posterior over function values is given by Bayes' rule:

$$p(\mathbf{r}|\mathcal{S},\mathcal{G},\boldsymbol{\theta}) = \frac{p(\mathcal{G}|\mathcal{S},\boldsymbol{\theta},\mathbf{r})p(\mathbf{r}|\mathcal{S},\boldsymbol{\theta})}{p(\mathcal{G}|\mathcal{S},\boldsymbol{\theta})}. \tag{5.4}$$

The posterior combines the information from the prior and the data, which reflects the updated belief about $\mathbf{r}$ after observing the decision behavior. We can calculate the denominator in Eq.5.4 by integrating $p(\mathcal{G}|\mathcal{S},\boldsymbol{\theta},\mathbf{r})$ over the function space with respect to $\mathbf{r}$, which requires high computation capacity. Fortunately, we are able to maximize the unnormalized posterior density of $\mathbf{r}$ without calculating the normalizing denominator, since the denominator $p(\mathcal{G}|\mathcal{S},\boldsymbol{\theta})$ is independent of the values of $\mathbf{r}$. In practice, we obtain the maximum posterior by minimizing the negative log posterior, which is written as

$$\begin{aligned} U(\mathbf{r}) &\triangleq \frac{1}{2}\sum_{m=1}^{M}\mathbf{r}_{a_m}^T\mathbf{K}_{a_m}^{-1}\mathbf{r}_{a_m} - \sum_{n=1}^{N}\sum_{k=1}^{n_n}\ln\Phi(\sum_{m=1}^{M}\rho_{a_m}^{nk}\mathbf{r}_{a_m}) \\ &+\sum_{n=1}^{N}\sum_{l=1}^{m_n}\frac{1}{2}(\sum_{m=1}^{M}\rho_{a_m}^{nl}\mathbf{r}_{a_m})^2 \end{aligned} \tag{5.5}$$

where given $(\hat{a} \sim_{s_n} \hat{a}')_l$, let $\boldsymbol{\Delta}_l \triangleq \gamma(\mathbf{P}_{\hat{a}}(s_n,:) - \mathbf{P}_{\hat{a}'}(s_n,:))(\mathbf{I}_N - \gamma\mathbf{P}_{\pi(s_n)}(s_n,:))^{-1}$, then we have

$$\rho_{a_m}^{nl} = \mathbf{e}_n[\mathbf{1}(a_m = \hat{a}) - \mathbf{1}(a_m = \hat{a}')] + \boldsymbol{\Delta}_l\hat{\mathbf{I}}_{a_m}$$

where $\mathbf{e}_n$ is a $1 \times N$ vector whose entry $\mathbf{e}_n(n) = 1$, and $\mathbf{e}_n(j) = 0, \forall j \in \mathcal{N} \setminus \{n\}$. The notation $\mathbf{1}(.)$ is an indicator function. Similarly, $\rho_{a_m}^{nk}$ denotes the coefficient vector for the k-th strict preference relation $\hat{a} \succ_{s_n} \check{a}$.

Qiao and Beling (*Qiao and Beling* [2011]) give a proof that Proposition (5.5) is a convex optimization problem (see Appendix Convex Optimization Problem). At the minimum of $U(\mathbf{r})$ we have

$$\frac{\partial U}{\partial \mathbf{r}_{a_m}} = 0 \Rightarrow \hat{\mathbf{r}}_{a_m} = K_{a_m}(\nabla \log P(\mathcal{G}|\mathcal{S}, \hat{\mathbf{r}}, \boldsymbol{\theta})) \tag{5.6}$$

where $\hat{\mathbf{r}} = (\hat{\mathbf{r}}_1, \cdots, \hat{\mathbf{r}}_{a_m}, \cdots, \hat{\mathbf{r}}_m)$. In Eq.5.6, we can use Newton's method to find the maximum of U with the iteration,

$$\mathbf{r}_{a_m}^{\text{new}} = \mathbf{r}_{a_m} - (\frac{\partial^2 U}{\partial \mathbf{r}_{a_m} \partial \mathbf{r}_{a_m}})^{-1} \frac{\partial U}{\partial \mathbf{r}_{a_m}}$$

5.4 Experiment with E-Mini S&P 500 Equity Index Futures Market

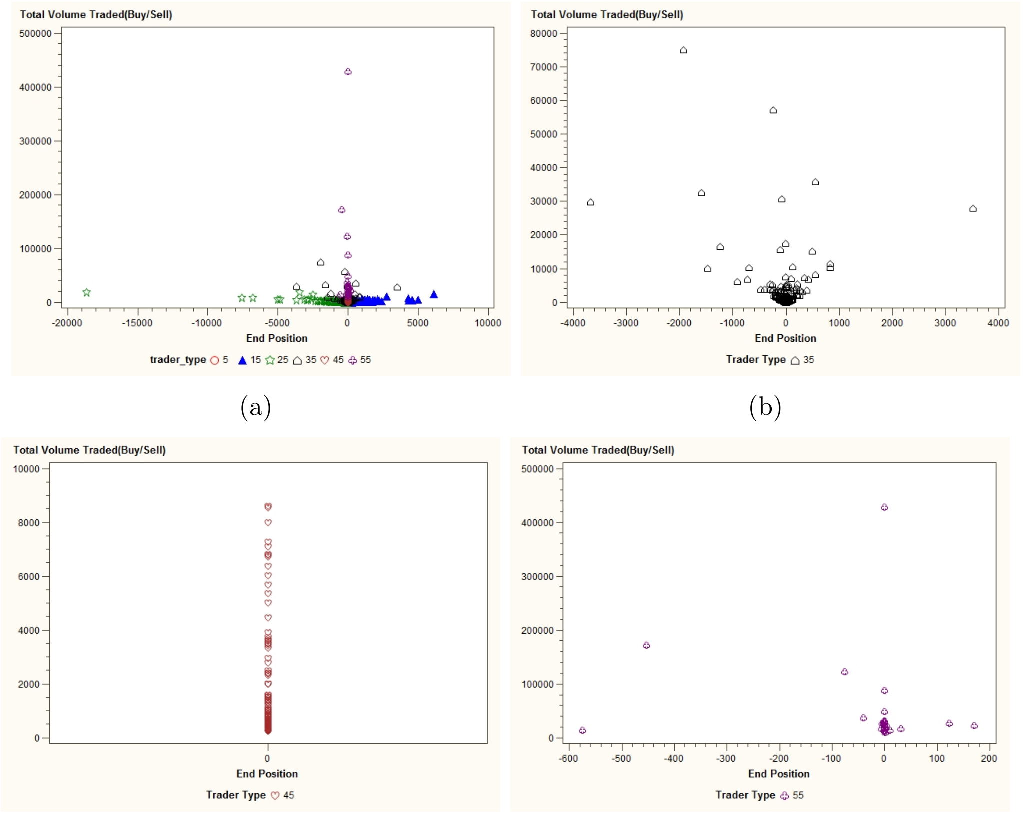
5.4.1 Market Data Description

E-Mini S&P 500 is a stock market index futures contract traded on the Chicago Mercantile Exchange's (CME) Globex electronic trading platform. The notional value of one contract is \$50 times the value of the S&P 500 stock index. The tick size for the E-Mini S&P 500 is 0.25 index points or \$12.50. If, for example, the S&P 500 Index futures contract is trading at \$1,400.00, the value of one contract is \$70,000. The advantages to trading E-mini S&P 500 contracts include liquidity, greater affordability for individual investors and around-the-clock trading.

Trading takes place 24 hours a day with the exception of short technical maintenance shutdown from 4:30 p.m. to 5:00 p.m. The E-Mini S&P 500 expiration months are March, June, September, and December. On any given day, the contract with the nearest expiration date is called the front-month contract. The E-Mini S&P 500 is cash-settled against the value of the underlying index and the last trading day is the third Friday of the contract expiration month. Initial margin for speculators and hedgers are $5,625 and $4,500 respectively. Maintenance margins for both speculators and hedgers are $4,500. There is no limit on how many contracts can be outstanding at any given time.

The CME Globex matching algorithm for the E-Mini S&P 500 offers strict price and time priority. Specifically, limit orders that offer more favorable terms of trade (sell at lower prices and buy at higher prices) are executed prior to pre-existing orders. Orders that arrived earlier are matched against the orders from the other side of book before other orders at the same price. This market operates under complete price transparency. This straight forward matching algorithm allows us to reconstruct the order book using audit trail messages archived by the exchanges. Hence we can replay the market dynamics at any given moment.

Under the classification rule documented by Kirilenko et al. (*Kirilenko et al.* [2011]), we can designate individual trading accounts into six categories based on their trading activities. These categories include: High Frequency Traders (high volume and low inventory), Intermediaries (low inventory), Fundamental Buyers (consistent intraday net buyers), Fundamental Sellers (consistent intraday net sellers), Opportunistic Traders (all other traders not classified) and Small Traders (low volume). For Fundamental Traders, we apply calculation of their end of day net position. And if it is more than 15% of their total trading volume of that day, we categorize them either as Fundamental Buyers or Fundamental Sellers depending on their trading directions. We can also easily identify Small Traders as those accounts with 9 or less trading volume. We also apply the criteria (*Kirilenko et al.* [2011]) for Intermediaries, Opportunistic Traders and High Frequency Traders, and we found that we can get pretty consistent results for Intermediaries for the one-month data. There are two steps involved. First, we make sure the account's net holdings fluctuate within 1.5% of its end of day level, and second, we make sure the account's end of day net position is no more than 5% of its daily trading volume. Then if we define HFTs as a subset of Intermediaries (top 7% in daily trading volume), we found there is a significant amount of overlap between HFTs and Opportunistic Traders. The problem is that the first criteria is not well defined. The definition of fluctuation of net holdings is very vague. It could be measure in different ways. After talking to the authors, we decided to use standard deviation of an account's net position measured in event clock as a measure of an account's holding fluctuation. With this definition, it turns out that 1.5% fluctuation is too stringent for HFTs. It is manifested in the fact that a lot of accounts with high trading volume are getting classified as Opportunistic Traders. But in reality their end of day positions are still very low compared with other Opportunistic traders. Therefore we decided to relax the first criteria as the standard deviation of the account's net holdings throughout the day that is less than its end of day holding level. From Figure 5.3 (a), we see that with the newly adjusted criteria most of high volume trading accounts are classified as HFTs (without this adjustment almost all the top trading accounts are classified as Opportunistic Traders). After applying the new classification rule, we summarized the statistics in Table 5.1. From this result, we find that we have more HFTs identified using the modified classification criteria. On average there are 38 HTF accounts, and 118 Intermediary accounts, 2,658 Opportunistic accounts, 906 Fundamental Buyer accounts, 775 Fundamental Seller accounts, and 5,127 Small Trader accounts. We then look over the 4 weeks period and found that amount the 120 accounts that consistently traded over this period only 36% of them are consistently classified as the same type of traders. If we rank these accounts by their daily trading volume, we find that only 40% of the top 10 accounts are consistently classified as the same trader types. The variation is among the three types i.e. HFTs, Intermediaries, and Opportunistic Traders. Next we will show how our IRL based behavior identification approach is much superior to the summary statistics based approach. We will use the top 10 trading accounts as examples to demonstrate further improvement of behavior based trading strategy identification using Gaussian Preference IRL model.



(a) (b)

(c) (d)

Figure 5.3: **E-mini S&P 500 Market Participant Classification:** (a) All market participants by the total volume traded and the end of day position. (b) Opportunistic traders by the total volume traded and the end of day position. (c) Intermediaries by the total volume traded and the end of day position. (d) High Frequency traders by the total volume traded and the end of day position.

5.4.2 A MDP Model for Market Dynamics

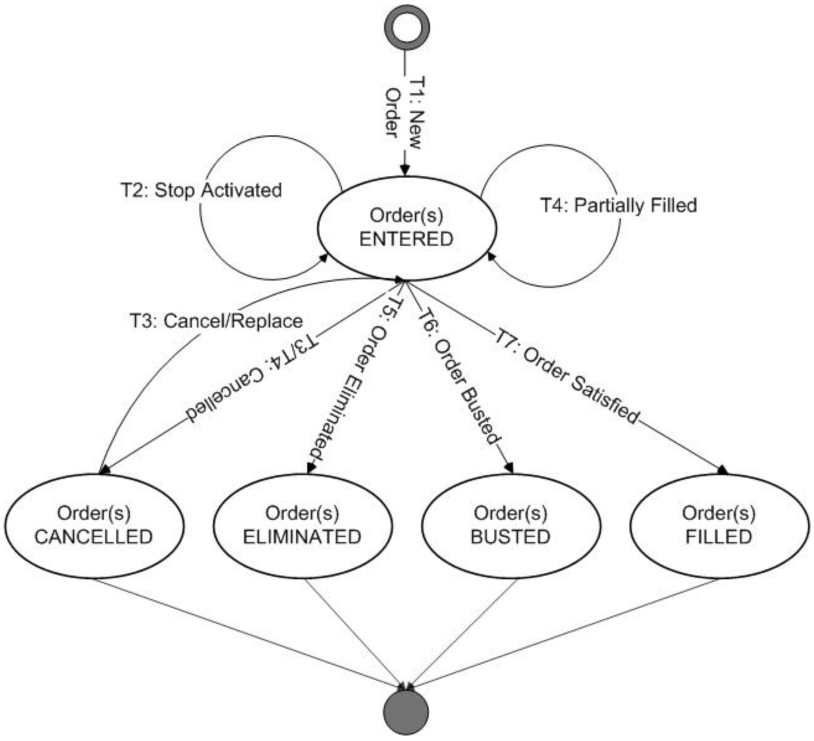
In this study we use a month of order book audit trial data from the E-Mini S&P 500 Futures contract market. The audit trail data includes all the order book events timestamped at millisecond time resolution. We use the following data fields:

Table 5.1: E-Mini S&P 500 Futures Market Data Summary

		Market	***Opportunistic***	***Fundamental***	***Fundamental***	***Total Number***	***Total Trading***
Date	**HFTs**	**Makers**	**Traders**	**Buyers**	**Sellers**	**of Accounts**	**Volume**
10/04/2012	39	193	2,833	940	818	10,425	3,261,852
10/05/2012	38	162	2,598	1191	1055	11,495	3,875,232
10/06/2012	38	167	2,401	895	712	9,065	2,852,244
10/07/2012	39	196	2,726	919	747	9,841	3,424,768
10/08/2012	32	162	2,511	847	812	9,210	3,096,800
10/11/2012	21	118	1,428	636	573	6,230	1,765,254
10/12/2012	38	186	2,687	896	745	9,771	3,236,904
10/13/2012	38	187	2,582	1020	840	10,297	3,699,108
10/14/2012	30	198	3,001	1070	795	10,591	4,057,824
10/15/2012	46	210	3,109	890	773	9,918	4,437,826
10/18/2012	37	173	2,126	869	724	8,735	2,458,510
10/19/2012	52	216	3,651	1030	974	11,600	5,272,672
10/20/2012	39	176	2,949	951	877	10,745	3,956,790
10/21/2012	43	240	3,370	952	771	10,980	4,230,194
10/22/2012	32	143	1,837	676	629	7,370	2,026,234
10/25/2012	38	181	2,533	888	684	9,228	3,074,558
10/26/2012	37	175	2,726	816	709	9,568	3,000,628
10/27/2012	45	186	2,973	919	820	10,472	3,850,556
10/28/2012	39	185	2,873	914	705	9,777	3,485,910
10/29/2012	37	160	2,247	794	744	8,369	3,012,860

date, time (time when order is submitted to the exchange from clients), conf_time (time when the order is confirmed by the matching engine), customer account, tag_50 (trader identification number), buy or sell flag, price, quantity, order ID, order type (market or limit), and func_code (message type, i.e. order, modification, cancellation, trade, etc.).

Figure 5.4 shows the entire life-cycle of any given order initiated by a client. The order book audit trial data contains these messages, and the entire order history (i.e. order creation, order modifications, fills, cancellation, etc.) can be retrieved and analyzed. The first step of our study is to reconstruct the limit order book using the



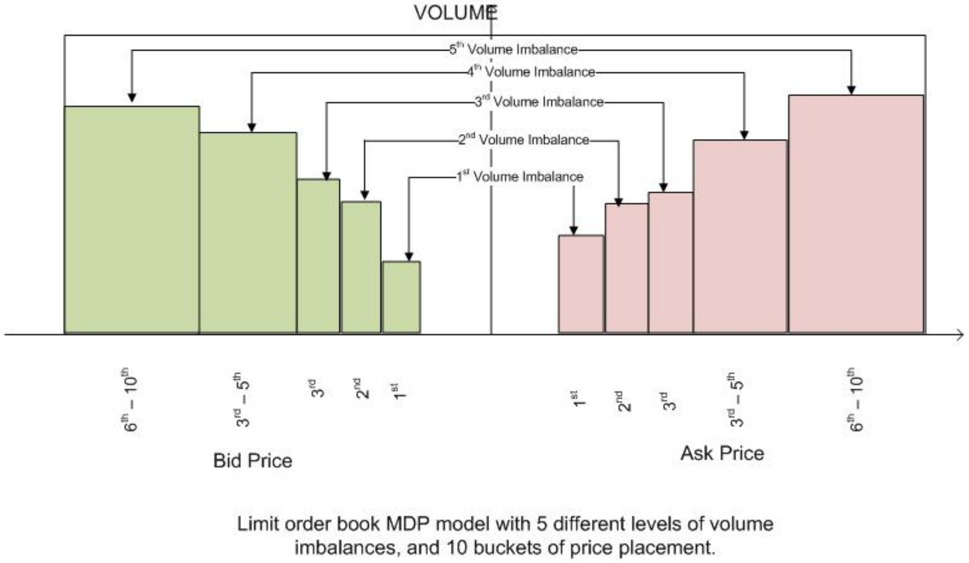
(a) Order Lifecycle on CME Globex

Figure 5.4: **CME Globex Order Lifecycle.** T1: Trader submits a new order; T2: The state of an order is changed, if a stop is activated; T3: A trader may choose to cancel an order, and the state of an order can be modified multiple times; T4: When an order is partially filled, the quantity remaining decreases; T5: Order elimination is similar to order cancellation except it is initiated by the trading engine; T7: An order may be filled completely; T6: Trades can be busted after the fact by the exchanges.

audit trail messages. The order book will then give us bid/ask prices, market depth, liquidity, etc. During this process, we process billions of messages for each trading date, and we build price queues using the price and time priority rule.

Once we have the order book at any given event tick, we take market depth at five different levels as our base variables and then discretize these variables and generate our MDP model state space. In this study we extend the MDP model documented by Yang et al. (*Yang et al.* [2012]), and we end up with five variables, i.e. order volume imbalance between the best bid and the best ask prices, order volume imbalance between the 2nd best bid and the 2nd best ask prices, order volume imbalance between the 3rd best bid and the 3rd best ask prices, the order book imbalance at the 5th best bid and the 5th ask prices, and inventory level/holding position (see Figure 5.5 (b)). And then we discretize the values of the five variables into three levels defined as high (above $\mu + 1.96\sigma$), neutral ($\mu \pm 1.96$), and low ($\mu - 1.96\sigma$). As it is argued by Yang et al. (*Yang et al.* [2012]), these volume related variables reflect the market dynamics on which the traders/algorithms depend to place their orders at different prices. With the volume imbalance at the best bid/ask prices being the most sensitive indicator of trading behavior of HFTs, Intermediaries and some of the Opportunistic traders, we also hypothesize that volume imbalance at other prices close to the book prices will enhance the information we need to infer trader's behavior. As it is demonstrated in the previous work (*Yang et al.* [2012]), the private variable, trader's inventory levels, provides very critical information about trader's behavior. It has been documented (*Kirilenko et al.* [2011], *Easley et al.* [2010] and *Brogaard* [2010]) that traders in a high frequency environment strive to control their inventory level as a critical measure of controlling their position risk. It is also reported that the HFTs and Market Makers tend to turn over their inventory levels 5 or more times a day, and target to hold very small or even zero inventory positions at the end of the trading session. All these provide us strong evidence to introduce a position variable to characterize trader's behavior in our model. Therefore, together with the volume imbalance variables, we propose $3^5(243)$ states in our computational model.

And then the next we need to define the action space. In general we have three types of actions: place new order, cancel an existing order, or place a market order. We divide the limit order book into 10 buckets at any given point of time by the following price markers: the best bid price, the 2nd best bid price, the 3rd best bid price, between the 4th and the 5th bid price, below 5th bid price, the best ask price, the 2nd best ask price, the 3rd best ask price, between the 4th and the 5th ask price, and above the 5th ask price. And then at any given point of time, a trader can take



(a)

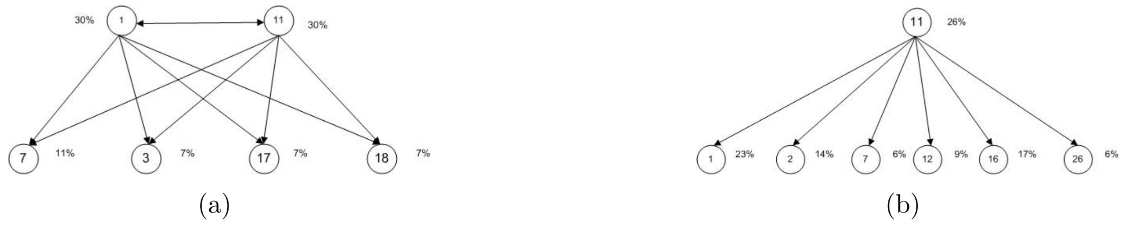
Figure 5.5: **Order Book MDP Model:** This graph shows the state variables using in the MDP model.

30 actions. The price markers used to define the price ranges are illustrated in Figure (5.5).

Table 5.2: Action Preference Graph Examples

State	Action	Frequency Observed	State	Action	Frequency Observed
14	1	0.23	158	1	0.30
14	2	0.14	158	3	0.07
14	7	0.06	158	7	0.11
14	11	0.26	158	11	0.30
14	12	0.09	158	17	0.07
14	16	0.17	158	18	0.07
14	26	0.06	158	20	0.07

Once we have the state and action space defined, we can create action preference graph based on the statistics of actions under different states. Here we use two examples to demonstrate how the action preference graphs have been constructed based on the MDP model and observed actions. Table 5.2 shows two example states where we have multiple actions observed. We then sort the frequency in descending order and construct a two-layer graph: top layer has the most frequently observed actions and the bottom layer are all the other actions. Based on this preference observation, we can construct two preference graphs as shown in Figure (5.6). The state transition matrix can be constructed for the entire market for the observation period. In our MDP model, we have a 243x243 matrix for every single action.



(a) (b)

Figure 5.6: **Action Preference Graph Examples:** (a). This graph shows an example action preference graph at state 158; (b). This graph shows an example action preference graph at state 14.

5.4.3 Trader Behavior Identification

Yang et al. (*Yang et al.* [2012]) examine different trading behaviors using a linear IRL (LNIRL) algorithm with simulated E-Mini S&P 500 market data. In that MDP model, there are three variables selected: volume imbalance at the bid/ask prices, volume imbalance at the 3rd best bid/ask prices, and position level. Even though this MDP model is a relatively simple one, it is evident from the experiment results that IRL reward space is effective in identifying trading strategies with a relatively high accuracy rate.

In this paper, we are trying to address two important issues during the modeling process in order to solve realistic market strategy learning problem using the real market data. The first issue is that in reality we often may not have complete observations of trader's policies. Since the market presents itself as a random process in terms of both prices and volume, it is unlikely that during our observation window that we will be able to capture all the possible states. While in the study performed by Yang et al. (*Yang et al.* [2012]), they assume a complete observation of trader's decision policies for the simulated trading strategies. In other words, the simulated policies by distribution can be completely captured when the simulation is run long enough. The study of the convergence of these simulated policies and the testing results are consistent with their assumptions. However, when we use real market data for strategy learning, we will have to address the incomplete observation problem. The second issue is in regard to deterministic policy vs. non-deterministic policy assumption. In the earlier work, Yang et al. (*Yang et al.* [2012]) make deterministic policy assumption. Under the linear feature optimization framework, non-deterministic policies can be represented by a single maximum deterministic policy (see proof in Appendix Deterministic Policy vs. Randomized Policy). In this study, we relax the deterministic policy assumption, and allow non-deterministic ones under a Gaussian process framework. As we argue earlier Gaussian process learning allows us to infer policies even when we have very limited observations. At the same time, we incorporate Gaussian preference learning into our inference process. It helps us to incorporate less frequently observed policies into our reward learning process. Together the proposed GPIRL approach induces a model which makes less requirement on observations, and fewer assumptions on the polices we are to learn.

5.4.4 Multi-class SVM Trader Classifier using GPIRL vs. LNIRL

In this section, we use support vector machine (SVM) classification method to identify traders based on reward functions that we recover from the observations of the trader's behaviors. We select a group of traders whose behaviors are consistently observed during the period we study. The primary reason for choosing SVM classification method is its flexibility that we can explore feature separation in different high dimensional spaces using kernel functions. We aim to compare performance of the two behavior learning algorithms i.e. LNIRL and GPIRL, and show that GPIRL has superior performance in addressing real world trading strategy identification.

We first review SVM formulation. The recent results in pattern recognition have shown that SVM classifiers often have superior recognition rates in comparison to other classification methods. The support vector machine is originally a binary classification method developed by Vapnik and colleagues at Bell laboratories (*Vapnik* [1999], *Burges* [1998], and *Joachims* [1998]). For a binary problem, we have training data points: $\mathbf{x}_i, y_i, i = 1, ..., l, y_i \in -1, 1, \mathbf{x}_i \in R^d$. Suppose we have some hyperplane which separates the positive from the negative examples - ("separating hyperplane"). $|b|/||w||$ is the perpendicular distance from the hyperplane to the origin, and $||w||$ is the Euclidean norm of w. For the linearly separable case, the support vector algorithm simply looks for the separating hyperplane with largest margin. This can be formulated as following inequalities:

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \forall i, \tag{5.7}$$

Thus we find the pair of hyperplanes which gives the maximum margin by minimizing $||\mathbf{w}||$, subject to constraint (5.7). We then introduce nonnegative Lagrange multipliers $\alpha_i, i = 1, ..., l$, for each of the inequality constraint (5.7). For equality constraints, the Lagrangian multipliers are unconstrained. This then gives a primal and dual problem:

$$w \equiv \sum_i \alpha_i y i \mathbf{x}_i \tag{5.8}$$

$$\sum_i \alpha_i y i = 0 \tag{5.9}$$

$$L_P \equiv \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i \tag{5.10}$$

$$L_D \equiv \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{5.11}$$

Support vector training therefore amounts to maximizing L_D with respect to the α_i, subject to constraints 5.9. In the solution, those points for which $\alpha_i > 0$ are called ”support vectors”, and lie on one of the hyperplanes defined in 5.7. If all other training points were removed (or moved around, but so as not to cross the hyperplanes), and training was repeated, the same separating hyperplane would be found.

Notice that the only way in which data appears in the training problem is in the form of dot product, $\mathbf{x}_i \cdot \mathbf{x}_j$. Now we can map the data to some other Euclidean space H, using a mapping called Φ, and then the training algorithm would only depend on the data through dot products in H, i.e. on functions of the form $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. This is called ”kernel function” $K(\mathbf{x}_i, \mathbf{x}_j = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j))$. We would only need K in the training algorithm, and would never need to explicitly know what Φ is. Now we have a separation plane in a different space. In test phase ans SVM is used by computing dot products of a given test point $\mathbf{x}$ with $\mathbf{w}$, or more specifically by computing the sign of (5.12):

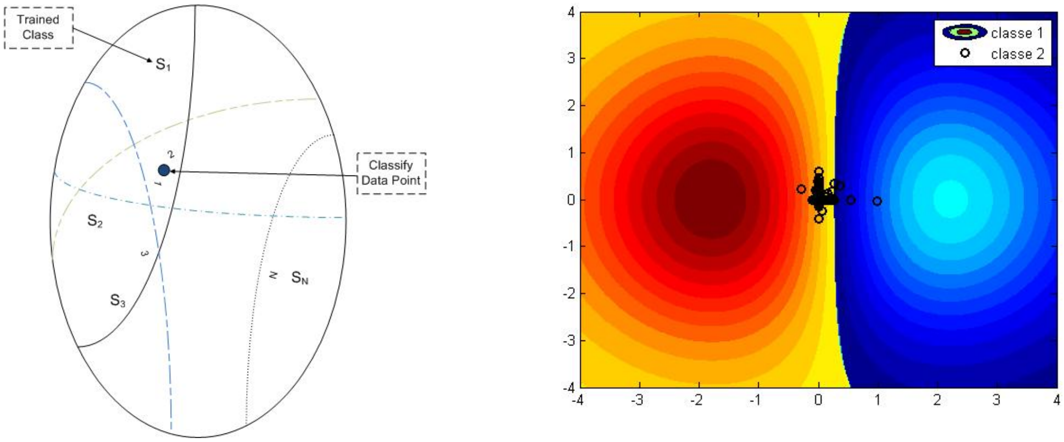
$$\{(\mathbf{x}) \equiv \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i \tag{5.12}$$

Here we constructed 80 sample trajectories/observations for each of the top 10 trading accounts. There are 121 trading accounts consistently traded over the 4 weeks period. In this study we will only focus our attention on the top 10 trading accounts. We apply both the LNIRL (*Ng and Russel* [2000] and *Yang et al.* [2012]), and GPIRL (*Qiao and Beling* [2011]) to these 800 samples. And then we apply SVM algorithm on the 10 traders using pair-wise classification. For each pair, we first training a SVM classifier (with Gaussian kernel) with randomly selected 60 samples, and test the classification on the remaining 20 samples. We repeat the sampling 100 times and then take the average classification accuracy. We list both LNIRL classification results in Table 5.9, and GPIRL results in Table 5.10. On average, LNIRL gives 0.6039 classification accuracy, while GPIRL gives 0.9650 classification accuracy. This result confirms our earlier assumption that GPIRL performs better when we have incomplete observations, and also incorporate nondeterministic policies through Gaussian preference learning.

However, pair-wise classification only provides us a basis for comparing two different behavior learning algorithms. Next we propose a binary tree based SVM algorithms for trading strategy identification.

As we know that the SVM was originally developed for binary decision problems, and its extension to multi-class problems is not straight-forward. How to effectively extend it for solving multi-class classification problem is still an on-going research issue. The popular methods for applying SVMs to multi-class classification problems usually decompose the multi-class problems into several two-class problems that can be addressed directly using several SVMs. A variety of techniques for decomposition of the multi-class problem into several binary problems using SVM as binary classification have been proposed, such as One-Against-All, One-Against-One, Directed Acyclic Graph SVM, and Binary Tree of SVM, and SVM using Binary Decision Tree, etc.

Here we chose One-Against-All method. In this method, the N-class problems ($N > 2$), N two-class SVM classifiers are constructed (*Vapnik* [1998]). The i^th SVM is trained while labeling the samples in the i^th class as positive examples and all the rest as negative examples (see Figure 5.7 (a)). In the recognition phase, a test example is presented to all N SVMs and is labeled according to the maximum output among the N classifiers. The disadvantage of this method is its training complexity, as the number of training samples is large. Each of the N classifiers is trained using all available samples. We use Gaussian kernel function, and Figure 5.7 (b) shows the separation boundary and support vectors in a projected two dimensional space of one class versus other classes separation.



(a)

(b)

Figure 5.7: **SVM Multi-class Classification:** (a). This graph shows the One-Against-All binary SVM method; (b). This graph shows a SVM separation and the support vectors at each step of the series of binary decisions.

When we compare the two IRL methods, we again see only slightly higher accuracy using GPIRL approach. On average with the binary tree search algorithm, the LNIRL gives 0.9413 classification accuracy while GPIRL yields 0.9973 accuracy rate. The difference is only about 6%. Intuitively we may attribute this improvement to SVM algorithm in which the One-Against-All method is able to take advantage of the increased sample and class population. Furthermore, we are dealing with the top 10 trading accounts, and the number of activities observed from them also greatly increase the reliability of the policies we can observe. Moreover, these accounts are most likely HFTs with very short trading horizons. We may not see a big advantage of using GPIRL over this population. But overall, we can conclude that GPIRL has superior classification in identifying a specific trading strategy based on incomplete observations.

5.4.5 Trading Strategy Clustering and Comparison with Summary Statistic Based Approach

In the previous section, we discovered that using reward functions we can reliably identify a particular trading strategy over a period of time with a relatively high accuracy. In this section, we want to study similarity among the different trading strategies based on their reward characterization. This problem can be characterized as unstructured learning problem - clustering. We have the characterization of rewards over the state space and action space, and we aim to group trading strategies based on their similarity over the Cartesian product of the state and action space. We also attempt to establish connections between these trading strategy classification definitions established by Kirilenko et al. (*Kirilenko et al.* [2011]) and our behavior based trading strategy clustering.

The first problem we have to address is dimensionality of the feature space. We essentially have a reward structure over a large set of feature set. This feature set is a product of the state space and action space in our computational model. Fortunately, under the LNIRL algorithm, we reduce the feature space to only the state space, because in this linear feature expectation optimization problem we only consider reward at a particular state. Under the deterministic policy assumption, we assume value function converge at a particular state. In another word, reward function is not a function of actions. In this case we have 243 features to be considered during the clustering. However, under the GPIRL framework we do not assume deterministic policy, and we have reward as a function of both states and actions. Therefore we have 243x30 features for the later approach. We also observe that the reward matrix is relatively sparse where there are zero values at many states. To consider computational tractability and efficiency, we examine the data structure through Principal Component Analysis first.

In the LNIRL case, the first two Principal Components (PCs) explain 79.78% of the data variation, and from the upper left plot in Figure (5.8) (a) we see that the first 200 PCs give us nearly 100% explanation power on the observations. While we look at the GPIRL case, the first two PCs only explain 38.98% of the data variation. Looking at the upper left plot in Figure (5.8) (b), we see that we have to select more PCs to have better representation of the data. To balance the accuracy and computing efficiency, we choose the first 200 PCs for the LNIRL and the first 400 PCs for the GPIRL case. With this reduction choice, we gain significant computing efficiency and we loose only less than 2% data variation (lower left figure in both Figure (5.8) (a) and (b)). From the upper right plots in both the IRL and the GPIRL space, we see that the first two PCs give us good representation along the first PC, and in the LNIRL case the feature vector representation is evenly distributed between the first two PCs. From data observation perspective, the LNIRL space has pretty distinct separation of the observations. On the other hand in the GPIRL space we see concentrations of these observations, but boundaries are not very clear. In both case, we would expect relatively good representation of the data variation using the PC dimension reduction technique.

Now we apply unsupervised learning method to group the trading behavior observed on a selected group of trading accounts over the observation period. We select 10 trading accounts with the highest average daily trading volume over a period of 4 weeks (20 days) in our first experiment. We define an observation instance as a continuous period covering two hours where we take all the activities from a particular trader including placing new orders, modifying and canceling existing orders, and placing market orders. For each trader, we take four observation instances on each trading date: two observation instances in the morning trading and two observation instances in the afternoon trading. The two observation periods in the morning and in the afternoon have an hour overlapping time, but the observations in the morning and the afternoon do not overlap. We do so based on the general theory of intraday U-shaped patterns in volume - namely, the heavy trading in the beginning and the end of the trading day and the relatively light trading in the middle of of the day. This has been documented in a numbers of studies (*Ekman* [1992], *Admati and Pfleiderer* [1988], *Lee et al.* [2001], and *Chordia et al.* [2001]). We also examined the traders' actions through out the entire trading day. We found that two-hour observation time is a good cut-off, and with the overlapping instances both in the morning and the afternoon we expect to capture U-shaped pattern for the market.

We then perform hierarchical clustering and generate a heat map and dendrogram of the observations in both the LNIRL reward space and the GPIRL reward space. The simplest form of matrix clustering clusters the rows and columns of a dataset using Euclidean distance metric and average linkage. For both Figure (5.9) (a) and (b), the left dendrogram shows the clustering of the observations (rows), and the top dendrogram shows the clustering of the PCs (columns). It is evident that there is a clear division of the observations (rows) in both cases. And then we take a closer look at the left dendrogram, and we see that we have two clusters: the top cluster and the bottom cluster. And there is a black divide strip lie in the middle of the the second small cluster. We then zoom into the small cluster and check the sources of these observations[2]. While in the LNIRL reward space, we find the small cluster consists observations mostly from trader 1 (observations numbered from 1 to 80) and trader 2 (observations numbered from 81 to 160). And observations from

[2] Note: In both Figure (5.8) (a) and (b), we group observations from the same trader together in our data matrix. We have 10 traders and each has 80 observations. From the lower left graph in both (a) and (b), observations are ordered by trader IDs sequentially. For example, observation 1 through observation 80 come from trader 1, and observation 81 through observation 160 come from trader 2. We do this toward right along the X-axis, and at the end we have observation 721 through observation 800 come from trader 10.

trader 9 (observations numbered from 641 to 720) form the black divide between these two groups. In the GPIRL reward space, we find the small cluster consists of three traders: trader 1 (observations numbered from 1 to 80), trader 2 (observations numbered from 81 to 160), and trader 5 (observations numbered from 321 to 400). And the observations from the rest of the traders lie on the other side of the divide. Moreover we find the observations from trader 9 (observations numbered from 641 to 720) form the black divide between these two groups. From these observations we see that majority of the top 10 traders form one group and there are 2 or 3 traders behave a little different from others. Furthermore, we observe that the clustering has less than perfect purity. In other words, there are individual observations from the top cluster lie in the small cluster at the bottom occasionally. It means change of behavior overtime. The interpretation of this observation is that there are times that the HFTs may behave like Opportunistic traders during a perhaps a short period of time. We also occasionally observe that Opportunistic traders behave like HFTs. That is when we have observations cross the divide and get into the top cluster.

Next we propose a continuous measure of clustering using hierarchical clustering method. We use summary statistic based trader classification method proposed by Kirilenko et al. (*Kirilenko et al.* [2011]) to create reference labels. For this market data, we do not have true labels on those trading accounts. We aim to improve the labeling methods documented by Kirilenko et al. The motivation for creating a continuous measure of clustering is to address the potential change of trading behavior over time. As we mentioned earlier, we applied the summary statistic based classification rule on the 200 observations over the 4 weeks period, and we only find 40% time that we can consistently label the traders as the same type of traders. Now we define a weighted scoring system to evaluate both the rule based classifier and the behavior based classifier. Within the 6 types of traders that we defined in the data section, we only concerned about labeling HFTs, Intermediaries, and Opportunistic

Traders. Other three types of traders, e.g. Fundamental Buyers, Fundamental Sellers, and Small Traders can be reliably identify using their daily volume and their end of day positions. Here we assign score 2, if a trader is classified as a HFT; we assign score 1, if it is classified as an Opportunistic Trader; and we assign 0, if it is classified as an Intermediary. Labels for clustering are assigned using majority voting rule based on the summary statistic classification rule. We then combine the scores using a weight which is defined as the frequency that a particular score being assigned to a particular trader. Here we want to compare the summary statistic based trader type classification with the behavior based trader type classification. We aim to find connections between these two methods.

From visual representations in Figure (5.10) (a), we see that trader 1 and trader 5 have a wide range of end of day positions, but their daily trading volume remains relatively at the same level. It is likely that we may sometime classify them as HFTs and sometime as Opportunistic traders. While for trader 2, even though the end day position range is smaller than trader 1 and trader 5. It shows a general pattern very similar to trader 1 and 5, and we should classify it as an Opportunistic trader. Based on this manual examination we should have trader 1, trader 2 and trader 5 as Opportunistic Traders and the rest as HFTs. Now we look at the comparison between the result from the summary statistic based classification rule an that from our behavior based classification results. From Figure (5.11) (a), we see in the LNIRL reward space we identify two groups traders. Eight out of ten are identified as HFTs and only trader 1 and trader 2 are classified as Opportunistic traders. This result is consistent with what we observe from the dendrogram in Figure (5.9) (a). When we compare this result from the GPIRL reward space and we can pick up all three traders, e.g. trader 1, trader 2 and trader 5, that we identified through the manual process. This result is also consistent with our observation from the heat map in Figure (5.9) (b). While the rule classification method misclassified trader 2. It is because in the rule based approach, the cut-off is based on a simple ratio between the trading volume and end position. We can see that trader 2 has a relatively small spread in terms of end position. However, the behavior based approach can catch this pattern and is able to cluster it with other traders with the similar pattern.

We run another experiment using randomly selected 10 traders from the top 30 traders based on their trading volume. We know this selection will only result in three types of traders, e.g. HFTs, Intermediaries and Opportunistic Traders. We feed these 800 observations to both LNIRL and GPIRL algorithms to get reward representations of their trading behavior. Based on visual examination (see Figure (5.10) (b)), we see that trader 1, trader 2 and trader 7 are Opportunistic Traders and the rest are HFTs. We apply the same technique as before and we use the same cut-off score (1.85 in LNIRL reward space, and 1.75 in GPIRL reward space). As a result we can accurately identified the two classes of traders using the same cut-off score we used for the top 10 case (see Figure (5.12)). And in this experiment the classification in LNIRL reward space gives the same result as that in GPIRL reward space. While the rule classification method misclassified trader 3 as an Opportunistic trader. But if we look at its daily end position, its daily total trading volume and its inventory variance, it should be classified as a HFT. And again this misclassification is due to the aggregate cut-off ratio. However, the behavior based approach can catch this pattern and is able to cluster it with other traders with the similar behavior pattern. Overall, we conjecture that GPIRL reward space score based classification rule is better than the summary statistic based approach in that it is based on the similarity in behavior and it has clear interpretation. Because it a better reflection of traders' choice of actions under different market conditions than the summary statistics, it is well suited for discover new behavior patterns of market participants. We also conclude that the GPIRL reward space is more informative than and is a superior measure of trading behavior to the LNIRL reward space.

5.5 Conclusion

We assume incomplete observation of Algorithmic Trading strategies, and we model trader's reward function as a Gaussian process. We also incorporate trader's action preference under different market conditions through preference graphs learning. The aim of this study is to quantify trader's behavior based on the IRL reward learning under a Bayesian inference framework. We apply both linear (a linear combination of known features) approach (*Abbeel and Ng* [2004]) and GPIRL (*Qiao and Beling* [2011]) on a real market dataset (E-Mini S&P 500 Futures), and we conclude that GPIRL is superior to the LNIRL methods with 6% increase in identification accuracy rate. Furthermore, we establish a connection between the summary statistic based classification (*Kirilenko et al.* [2011]) and our behavior based classification. We propose a score based method to classify trader types, and because of the transferable property of the reward structure the cut-off score for classifying a group of traders can be applied to different market conditions.

The implication of this research is that the reward/utility based trading behavior identification can be applied to real market data to accurately identify specific trading strategies. As it is documented by Abbeel et al. (*Abbeel and Ng* [2004]) and confirmed by many other researchers, reward function is the most succinct, robust, and transferable definition of a control task. Therefore, the behavior learned under the reward space has much broader application than policies observed. Furthermore, these learned reward functions will allow us to replicate a particular trading behavior in a different environment to understand their impact to the market price movement and market quality in general.

We also want to point out some future research suggestions in the area of both improvement of identification accuracy and application of the behavior characterization:

- During our preference learning inference phase, we only considered a simple two layer preference graph. However, trader's preferences can be further distinguished with multi-layer graphs or other preference learning techniques;

- Our study focused on the top 10 Algorithmic traders on a market. Future study can extend this results to a large scale experimentation to include market participants (specifically Opportunistic traders), and study their behavior similarity through clustering. We can then associate the group behavior with market quality measures;

- Under the GPIRL framework, we are able to recover a detailed reward structure. These reward functions can be used to generate new policies under a simulated market condition to understand the complete behavior of certain trading strategies. It will be particularly interesting to the market regulator to see how the various trading strategies will interact during a stressed market condition;

Table 5.3: Pair-wise Trader Classification Accuracy using SVM Binary Classifier using LNIRL

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	0.0000	0.5437	0.5187	0.4812	0.6375	0.4812	0.5312	0.5750	0.7750	0.5937
[2,]	0.5437	0.0000	0.5250	0.5125	0.7437	0.5562	0.4937	0.4250	0.7625	0.6812
[3,]	0.5187	0.5250	0.0000	0.4687	0.6875	0.5250	0.5187	0.5250	0.7312	0.6250
[4,]	0.4812	0.5125	0.4687	0.0000	0.6937	0.5000	0.4937	0.5062	0.6562	0.6625
[5,]	0.6375	0.7437	0.6875	0.6937	0.0000	0.6625	0.7375	0.6875	0.7750	0.5437
[6,]	0.4812	0.5562	0.5250	0.5000	0.6625	0.0000	0.5500	0.5500	0.6500	0.6375
[7,]	0.5312	0.4937	0.5187	0.4937	0.7375	0.5500	0.0000	0.4937	0.8000	0.6125
[8,]	0.5750	0.4250	0.5250	0.5062	0.6875	0.5500	0.4937	0.0000	0.6437	0.6562
[9,]	0.7750	0.7625	0.7312	0.6562	0.7750	0.6500	0.8000	0.6437	0.0000	0.7437
[10,]	0.5937	0.6812	0.6250	0.6625	0.5437	0.6375	0.6125	0.6562	0.7437	0.0000
Notes: Columns and rows of this table represent all the traders by their anonymous IDs.										

Table 5.4: Pair-wise Trader Classification Sensitivity using SVM Binary Classifier in LNIRL Reward Space

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[,1]	0	0.4625	0.6625	0.6125	0.8125	0.5750	0.7125	0.7125	0.6875	0.8125
[,2]	0.4625	0	0.9250	0.8375	0.8375	0.8250	0.8625	0.7875	0.7750	0.8500
[,3]	0.6625	0.9250	0	0.6000	0.7875	0.6375	0.4625	0.4500	0.6375	0.6875
[,4]	0.6125	0.8375	0.6000	0	0.7000	0.4125	0.4625	0.3750	0.6500	0.6500
[,5]	0.8125	0.8375	0.7875	0.7000	0	0.4875	0.6000	0.5500	0.6875	0.3500
[,6]	0.5750	0.8250	0.6375	0.4125	0.4875	0	0.5500	0.4875	0.5625	0.6125
[,7]	0.7125	0.8625	0.4625	0.4625	0.6000	0.5500	0	0.3625	0.6750	0.6250
[,8]	0.7125	0.7875	0.4500	0.3750	0.5500	0.4875	0.3625	0	0.6375	0.6125
[,9]	0.6875	0.7750	0.6375	0.6500	0.6875	0.5625	0.6750	0.6375	0	0.8500
[,10]	0.8125	0.8500	0.6875	0.6500	0.3500	0.6125	0.6250	0.6125	0.8500	0
Notes: Columns and rows of this table represent all the traders by their anonymous IDs.										

Table 5.5: Pair-wise Trader Classification Specificity using SVM Binary Classifier in LNIRL Reward Space

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[,1]	0	0.6125	0.7000	0.7750	0.5875	0.7375	0.7750	0.7250	0.8125	0.5125
[,2]	0.6125	0	0.6625	0.6625	0.7250	0.6875	0.6875	0.6750	0.7625	0.6875
[,3]	0.7000	0.6625	0	0.5250	0.6875	0.5000	0.5125	0.6500	0.7625	0.5375
[,4]	0.7750	0.6625	0.5250	0	0.5500	0.4375	0.4500	0.4750	0.7875	0.4250
[,5]	0.5875	0.7250	0.6875	0.5500	0	0.7125	0.7000	0.6750	0.8250	0.7500
[,6]	0.7375	0.6875	0.5000	0.4375	0.7125	0	0.5250	0.4750	0.7750	0.4000
[,7]	0.7750	0.6875	0.5125	0.4500	0.7000	0.5250	0	0.5250	0.8625	0.5500
[,8]	0.7250	0.6750	0.6500	0.4750	0.6750	0.4750	0.5250	0	0.6375	0.4875
[,9]	0.8125	0.7625	0.7625	0.7875	0.8250	0.7750	0.8625	0.6375	0	0.6375
[,10]	0.5125	0.6875	0.5375	0.4250	0.7500	0.4000	0.5500	0.4875	0.6375	0
Notes: Columns and rows of this table represent all the traders by their anonymous IDs.										

Table 5.6: Pair-wise Trader Classification Accuracy using SVM Binary Classifier using GPIRL

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	0.0000	1.0000	0.9875	0.9750	0.9500	0.9750	0.9625	1.0000	0.9750	1.0000
[2,]	1.0000	0.0000	0.9750	0.9375	0.9875	0.9750	0.9625	0.9625	0.9875	1.0000
[3,]	0.9875	0.9750	0.0000	0.9750	0.9625	0.9875	1.0000	0.9750	0.9750	0.9875
[4,]	0.9750	0.9375	0.9750	0.0000	0.9750	0.9500	0.9375	0.9875	0.9875	0.9750
[5,]	0.9500	0.9875	0.9625	0.9750	0.0000	1.0000	1.0000	0.9625	0.9875	1.0000
[6,]	0.9750	0.9750	0.9875	0.9500	1.0000	0.0000	0.9625	0.8750	0.9125	0.9750
[7,]	0.9625	0.9625	1.0000	0.9375	1.0000	0.9625	0.0000	0.8625	0.9625	0.9875
[8,]	1.0000	0.9625	0.9750	0.9875	0.9625	0.8750	0.8625	0.0000	0.8000	1.0000
[9,]	0.9750	0.9875	0.9750	0.9875	0.9875	0.9125	0.9625	0.8000	0.0000	0.9625
[10,]	1.0000	1.0000	0.9875	0.9750	1.0000	0.9750	0.9875	1.0000	0.9625	0.0000
Notes: Columns and rows of this table represent all the traders by their anonymous IDs.										

Table 5.7: Pair-wise Trader Classification Sensitivity using SVM Binary Classifier in GPIRL Reward Space

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[,1]	0	1.0000	1.0000	1.0000	0.9750	1.0000	1.0000	1.0000	1.0000	1.0000
[,2]	1.0000	0	1.0000	1.0000	0.9875	1.0000	1.0000	1.0000	1.0000	1.0000
[,3]	1.0000	1.0000	0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
[,4]	1.0000	1.0000	1.0000	0	0.9875	1.0000	0.9625	0.9875	1.0000	0.9875
[,5]	0.9750	0.9875	1.0000	0.9875	0	1.0000	1.0000	0.9875	1.0000	1.0000
[,6]	1.0000	1.0000	1.0000	1.0000	1.0000	0	0.9500	0.7875	0.8875	0.9500
[,7]	1.0000	1.0000	1.0000	0.9625	1.0000	0.9500	0	0.8750	0.9125	1.0000
[,8]	1.0000	1.0000	1.0000	0.9875	0.9875	0.7875	0.8750	0	0.7875	0.9875
[,9]	1.0000	1.0000	1.0000	1.0000	1.0000	0.8875	0.9125	0.7875	0	1.0000
[,10]	1.0000	1.0000	1.0000	0.9875	1.0000	0.9500	1.0000	0.9875	1.0000	0
Notes: Columns and rows of this table represent all the traders by their anonymous IDs.										

Table 5.8: Pair-wise Trader Classification Specificity using SVM Binary Classifier in GPIRL Reward Space

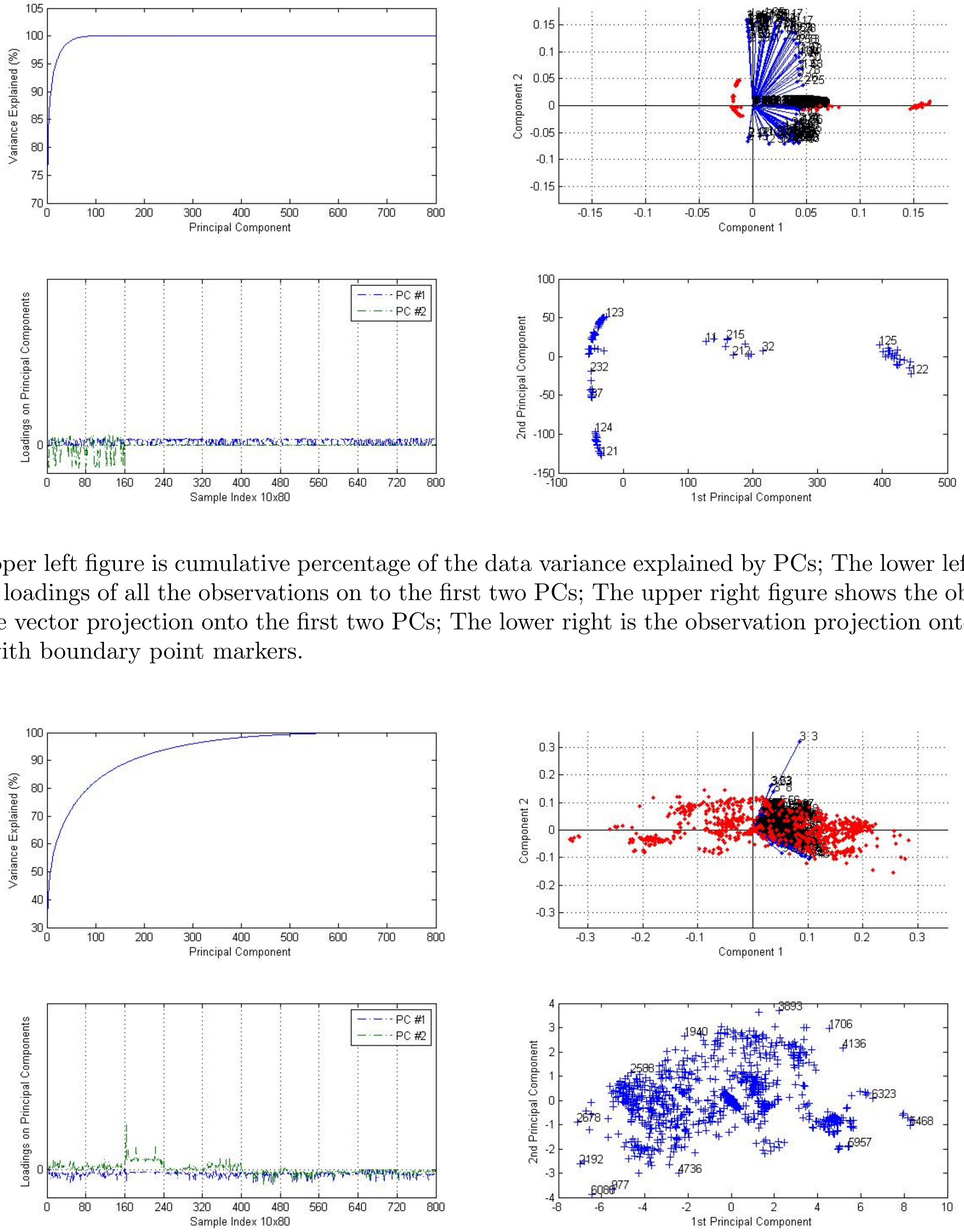
	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[,1]	0	0.9625	1.0000	1.0000	1.0000	1.0000	0.9875	1.0000	1.0000	0.9875
[,2]	0.9625	0	1.0000	0.9875	1.0000	0.9875	0.9875	0.9875	0.9875	1.0000
[,3]	1.0000	1.0000	0	0.9750	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
[,4]	1.0000	0.9875	0.9750	0	1.0000	1.0000	0.9875	1.0000	0.9875	1.0000
[,5]	1.0000	1.0000	1.0000	1.0000	0	1.0000	0.9875	0.9750	0.9875	0.9875
[,6]	1.0000	0.9875	1.0000	1.0000	1.0000	0	0.9750	0.8750	0.9125	0.9875
[,7]	0.9875	0.9875	1.0000	0.9875	0.9875	0.9750	0	0.8375	0.9125	1.0000
[,8]	1.0000	0.9875	1.0000	1.0000	0.9750	0.8750	0.8375	0	0.8250	0.9625
[,9]	1.0000	0.9875	1.0000	0.9875	0.9875	0.9125	0.9125	0.8250	0	0.9625
[,10]	0.9875	1.0000	1.0000	1.0000	0.9875	0.9875	1.0000	0.9625	0.9625	0
Notes: Columns and rows of this table represent all the traders by their anonymous IDs.										

Table 5.9: One-Against-All Trader Classification using SVM Binary Classifier using LNIRL

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	0.9275	0.9475	0.9275	0.9350	0.9300	0.9425	0.9450	0.9450	0.9525	0.9550
[2,]	0.9300	0.9350	0.9450	0.9325	0.9400	0.9475	0.9375	0.9175	0.9350	0.9300
[3,]	0.9425	0.9500	0.9375	0.9350	0.9225	0.9475	0.9375	0.9450	0.9275	0.9350
[4,]	0.9375	0.9350	0.9250	0.9425	0.9450	0.9350	0.9375	0.9400	0.9375	0.9350
[5,]	0.9350	0.9375	0.9325	0.9450	0.9475	0.9475	0.9400	0.9550	0.9325	0.9350
[6,]	0.9425	0.9500	0.9375	0.9350	0.9225	0.9475	0.9375	0.9450	0.9275	0.9350
[7,]	0.9400	0.9350	0.9450	0.9475	0.9525	0.9400	0.9525	0.9450	0.9400	0.9500
[8,]	0.9300	0.9350	0.9450	0.9325	0.9400	0.9475	0.9375	0.9175	0.9350	0.9300
[9,]	0.9375	0.9350	0.9250	0.9425	0.9450	0.9350	0.9375	0.9400	0.9375	0.9350
[10,]	0.9350	0.9375	0.9325	0.9450	0.9475	0.9475	0.9400	0.9550	0.9325	0.9350
Notes: Columns of the table represent all the traders by their anonymous IDs, and the rows represent 10 fold cross-validation results.										

Table 5.10: One-Against-All Trader Classification using SVM Binary Classifier using GPIRL

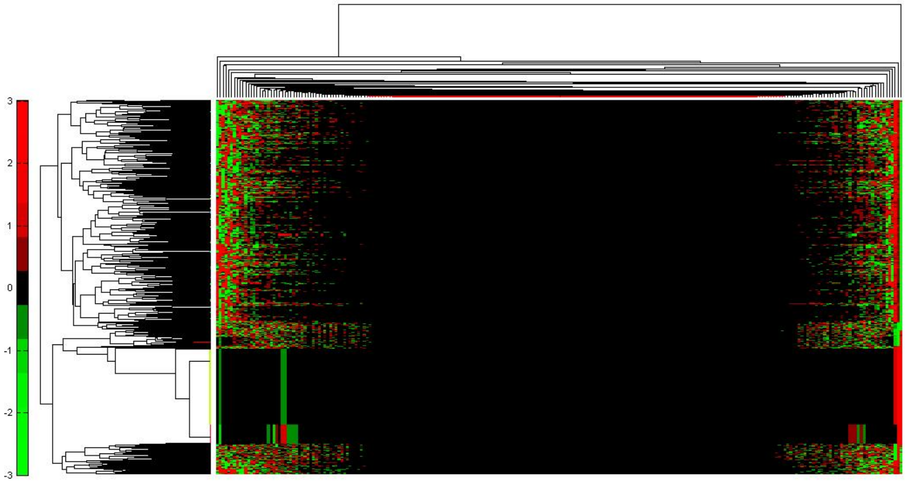
	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]
[1,]	0.9975	0.9975	0.9975	0.9975	0.9975	1.0000	0.9950	1.0000	1.0000	0.9975
[2,]	0.9950	0.9975	0.9750	0.9375	0.9875	0.9750	0.9625	0.9625	0.9875	1.0000
[3,]	0.9875	0.9750	1.0000	0.9750	0.9625	0.9875	1.0000	0.9750	0.9750	0.9875
[4,]	0.9750	0.9375	0.9750	1.0000	0.9750	0.9500	0.9375	0.9875	0.9875	0.9750
[5,]	0.9500	0.9875	0.9625	0.9750	1.0000	1.0000	1.0000	0.9625	0.9875	1.0000
[6,]	0.9750	0.9750	0.9875	0.9500	1.0000	1.0000	0.9625	0.8750	0.9125	0.9750
[7,]	0.9625	0.9625	1.0000	0.9375	1.0000	0.9625	1.0000	0.8625	0.9625	0.9875
[8,]	1.0000	0.9625	0.9750	0.9875	0.9625	0.8750	0.8625	1.0000	0.8000	1.0000
[9,]	0.9750	0.9875	0.9750	0.9875	0.9875	0.9125	0.9625	1.0000	1.0000	0.9625
[10,]	1.0000	1.0000	0.9875	0.9750	1.0000	0.9750	0.9875	1.0000	0.9625	1.0000
Notes: Columns of the table represent all the traders by their anonymous IDs, and the rows represent 10 fold cross-validation results.										



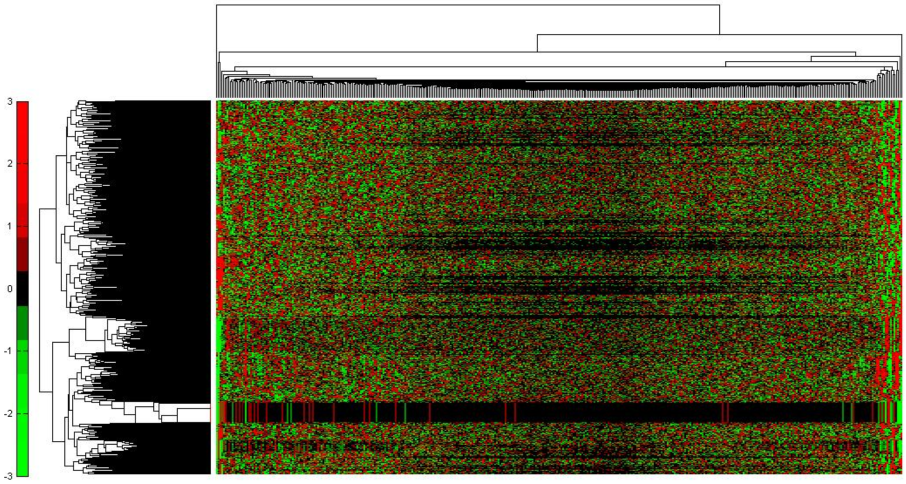
(a) The upper left figure is cumulative percentage of the data variance explained by PCs; The lower left figure is the plot of loadings of all the observations on to the first two PCs; The upper right figure shows the observation and feature vector projection onto the first two PCs; The lower right is the observation projection onto the first two PCs with boundary point markers.

(b) The upper left figure is cumulative percentage of the data variance explained by PCs; The lower left figure is the plot of loadings of all the observations on to the first two PCs; The upper right figure shows the observation and feature vector projection onto the first two PCs; The lower right is the observation projection onto the first two PCs with boundary point markers.

Figure 5.8: **Principal Component Representation of the Reward Data:** (a). Data representation under the first two Principal Components in the LNIRL Reward Space; (b). Data representation under the first two Principal Components in the GPIRL Reward Space.

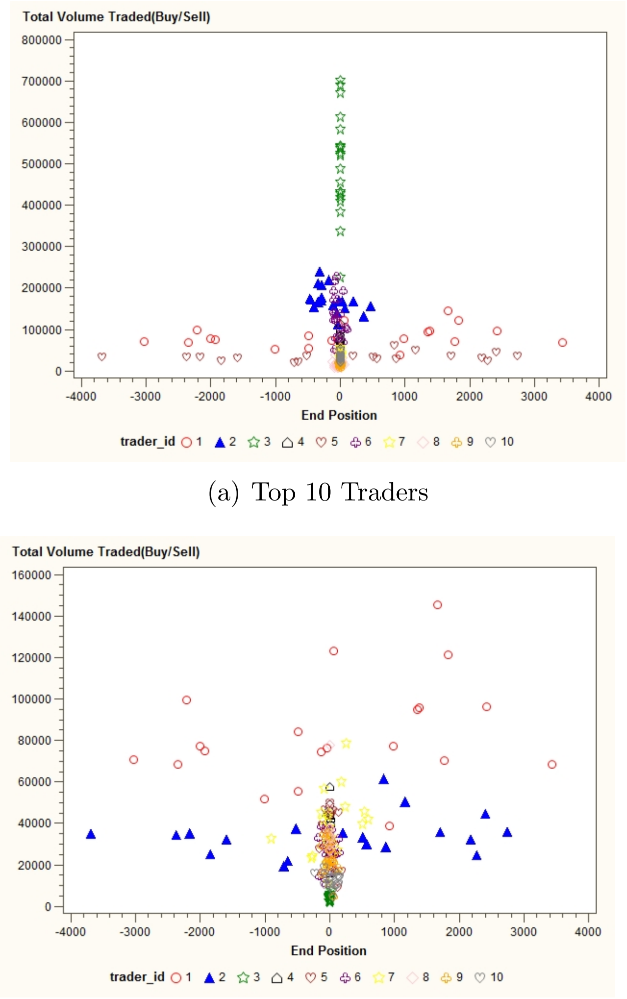


(a)



(b)

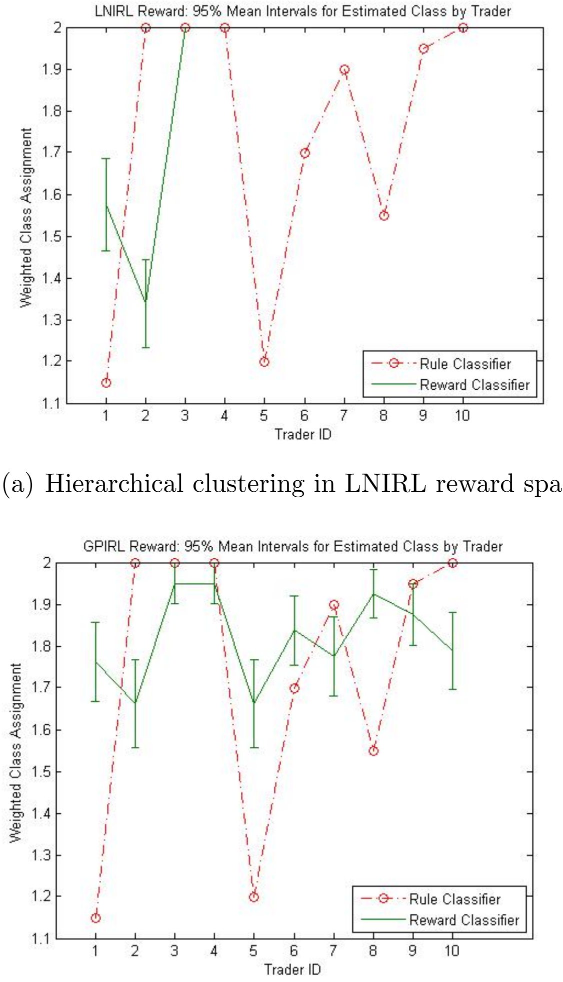
Figure 5.9: **Hierarchical Clustering of Data Matrix:** (a). Heat map of 800 observations of the Linear Rewards in the first 200 PCs; (b). Heat map of 800 observations in GPIRL Rewards in the first 400 PCs.



(a) Top 10 Traders

(b) Randomly Selected 10 Traders

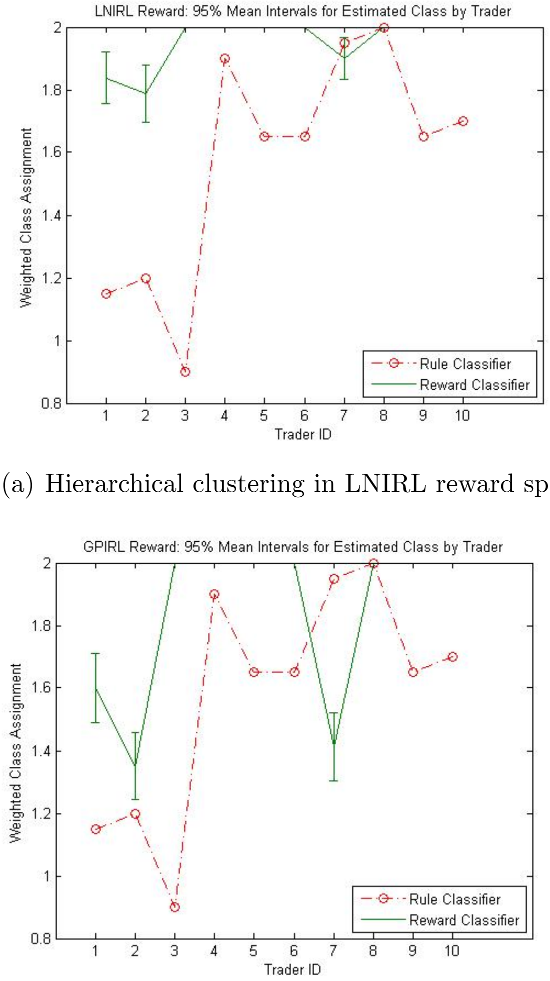
Figure 5.10: **Trader's Daily Trading Volume vs. Daily End Position during 20 Day's Period.** (a) Trader 1, 2 and 5 have varying end positions. (b) Trader 1, 2, and 7 have varying end positions.



(a) Hierarchical clustering in LNIRL reward space

(b) Hierarchical clustering in GPIRL reward space

Figure 5.11: **Trader Type Classification Compared with the Summary Statistic Based Rule Classification for the Top 10 Traders.**



(a) Hierarchical clustering in LNIRL reward space

(b) Hierarchical clustering in GPIRL reward space

Figure 5.12: Trader Type Classification Compared with the Summary Statistic Based Rule Classification for the Random 10 Traders.

CHAPTER VI

Conclusion and Future Work

6.1 Conclusions

We hypothesize that the Algorithmic Trading behavior can be accurately characterized in the reward space using IRL algorithms. This hypothysis is based on the research advancement in Reinforcement learning under the Markov Decision Process framework. We first establish the connection between market participants' activities including placing new orders, cancelling existing orders and placing market orders, etc. to market prices movement and price volatilities. Through an experical study of serveral Futures markets, we identified the volume imbalance variables as strong indicator of market movement and risk associated with it. We then construct a simple discrete MDP model with two volume imbalance variables and a position level variable. Using this MDP model and a simulated E-Mini S&P 500 Futures market, we formulate a linear algorithm (*Ng and Russel* [2000]) to learn traders's behavior in the reward space under the presuposition that the reward function, rather than the policy or the value function, is the most succinct, robust, and transferable definition of the task. The results are pretty satisfactory in that we can clearly separate High Frequency Trading strategies from other trading strategies with 90% accuracy rate, while the separation accuracy rate between Market Makers and Opportunistic Traders is 83%. When we apply the this model to real market data, we find the overall identification accuracy drop 10%. We attribute the accuracy limit to twofold factors. 1). For the real market data, we do not have full observation for some of the trading strategies, especially Opportunistic trading strategies. They generally tend to have longer policy horizon. Therefore the it is evident that the separation rate between HFTs and Opportunistic Traders, and between Market Makers and Opportunistic Traders will be relatively low. 2). In this model we assume stationary deterministic policies. This assumption may be two strong. In reality, some of the Algorithmic trading strategies may be designed to be non-deterministic in nature.

We further investigated a Baysian inference based model to identify trading strategies. We specifically targeted to address the incomplete observation issue and try to relax our assumption on the determinstric nature of the original trading strategies. We also incorporate traders' action preference under different market conditions through Preference graph learning technique. We model the traders' behavior as a Gaussian process in the reward space, and aim to recover the optimal policies and the corresponding reward functions to explain their behaviors. We prove that Algorithmic trading strategies can be accurately identified using GPIRL, and it is superior to the linear approach we studied earlier. Using the real market data (E-Mini S&P 500 Futures market), GPIRL consistently identify individual trading strategies with more than 95% accuracy. Compared with the linear IRL algorithm, it is a 60% better than the linear approach. However, using binary one-against-all SVM algorithm the gap is signifcantly arrowed to only 6%. This is purely because that SVM binary search algorithm is able to incorporate more information and be able to achive better classification accuracy using a poor classifier. But there are cost associated with this higher level search algorithm. From purely feature characterization perspective, GPIRL is better than the linear IRL approach by 60%.

The major contribution of this dissertation lies in the following areas:

1. We survey the currently established research results in the area of order flow

event impact to market prices and volatility. We also analyze the market impact of some intensive exogenous order events to relatively illiquid markets as a natural experiment, and we show that intensive order flow imbalance generate significant market impact in terms of both volatility and prices.

2. We investigate and address the issues of modeling algorithmic trading strategies using IRL models such as, addressing non-deterministic nature of the observed policies in learning, constructing efficient MDP models using order flow imbalances at different prices levels along with traders inventory levels to capture order book dynamics, achieving better identification accuracy in reward space, etc. With a reliably validated agent based market simulation, we capture the essential characteristics of the algorithmic trading strategies.

3. We model the reward function using Gaussian process, which offers the advantage that IRL using Gaussian process is relatively insensitive to the number of the observations and it performs better than other algorithms when we only have partial observations on the policies we try to recover.

4. We apply preference graphs to address non-deterministic nature of the observed trading behaviors, reducing the uncertainty and computation burden caused by the ill-posed nature of the inverse learning problem. We build new likelihood functions for preference graphs and prove the effectiveness of these formulations in experiments.

5. We also perform clustering of behavior representation of the trading strategies, and we make connections between the existing summary statistic based trader classification approach (*Kirilenko et al.* [2011]) with our behavior based classification approach. We further propose a score based classification approach to address variations of Algorithmic trading behavior under different market conditions. This approach provides a more quantitative method to categorizing

Algorithmic trading strategies, and we also show that this approach is more robust than the summary statistic based approach.

Overall, we conclude that behavor based behavior modeling based on IRL can dentify Algorithmic trading strategies with relativley hgih accuracy. With Gaussian process based Baysian inference, we can have algorithm that can work with real market trading strategies.

6.2 Future Work

This research concludes that the reward/utility based trading behavior identification can be applied to real market data to accurately identify specific trading strategies. Furthermore, the behaviors learned under the reward space has much broader application than policies observed. These learned reward functions will not only allow us accurately identify and classify market participants' behavior, but also help us to replicate a particular trading behavior in a different environment to understand their impact the market price movement and market quality in general.

We want to point out some future research suggestions in the area of both improvement of identification accuracy and application of the behavior characterization:

- During our preference learning inference phase, we only considered a simple two layer preference graph. However, trader's preferences can be further distinguished with multi-layer graphs or other preference learning techniques;

- Our study focused on the top 10 Algorithmic traders on a market. Future study can extend this results to a large scale experimentation to include market participants (specifically Opportunistic traders), and study their behavior similarity through clustering. We can then associate the group behavior with market quality measures;

- Under the GPIRL framework, we are able to recover a detailed reward structure. These reward functions can be used to generate new policies under a simulated market condition to understand the complete behavior of certain trading strategies. It will be particularly interesting to the market regulator to see how the various trading strategies will interact during a stressed market condition;

- Since reward can uniquely determine a control task, we explore construction of reward functions to generate better trading strategies.

- We may further understand the reward function and its relation to market risks. The idea here is that under current IRL framework, we only optimize the expected value. But in reality, algorithm designers also target to reduce trading system risks they may bear. Even though volatility may be implicitly considered in the order book dynamics (imbalances at different price levels), explicit modeling of risk may yield better results.

APPENDICES

APPENDIX A

Deterministic Policy versus Randomized Policy

Let $\pi = (d_1, d_2, ...) \in D^{MR}$. The expected total discounted reward of this policy is defined by

$$\begin{aligned} v_\gamma^\pi &= \sum_{t=1}^{\infty} \gamma^{t-1} P_\pi^{t-1} r_{d_t} \\ &= r_{d_1} + \gamma P_{d_1} r_{d_2} + \gamma^2 P_{d_1} P_{d_2} r_{d_3} + \\ &= r_{d_1} + \gamma P_{d_1}(r_{d_2} + \gamma P_{d_2} r_{d_2} + \gamma^2 P_{d_2} P_{d_3} r_{d_4} + ...) \\ v_\gamma^\pi &= r_{d_1} + v_\gamma^{\pi'}, where, \pi' = (d_2, d_3, ...) \in \Pi^{MR} \end{aligned} \tag{A.1}$$

or it can be expressed in component notation as:

$$v_\gamma^\pi(s) = r_{d_1}(s) + \sum_{j \in S} \gamma P_{d_1}(j|s) v_\gamma^{\pi'}(j) \tag{A.2}$$

The interpretation of this equation is that the discounted reward corresponding to policy p equals the discounted reward in a one-period problem in which the decision maker uses decision rule d_1 in the first period and receives the expected total discounted reward of policy π as a terminal reward. However, when p is stationary so that $\pi = \pi$, they simplify further. Let $d^{\infty} \equiv (d, d,)$ denote the stationary policy which uses policy $d^{\infty} \equiv (d, d,) \in D^{MR}$ at each decision epoch. For this policy equation A.2 becomes

$$v_{\gamma}^{d^{\infty}}(s) = r_{d_1}(s) + \sum_{j \in S} \gamma P_{d_1}(j|s) v_{\gamma}^{d^{\infty}}(j) \tag{A.3}$$

and equation A.3 becomes

$$v_{\gamma}^{d^{\infty}} = r_{d_1} + \gamma p_d v_{\gamma}^{d^{\infty}}, where, d^{\infty} = (d_2, d_3, ...) \in \Pi^{MR} \tag{A.4}$$

Thus, $v_{\gamma}^{d^{\infty}}$ satisfies the system equations

$$v = r_d + \gamma P_d v, or v = (I - \gamma \mathbf{P_d})^{-\mathbf{1}} \mathbf{r_d} \tag{A.5}$$

The matrix $v = (I - \gamma \mathbf{P_d})^{-\mathbf{1}} \mathbf{r_d}$ plays a crucial role in the theory of discounted Markov decision problems. The optimality equation or **Bellman** equations can be written as:

$$v^*(s) = \sup_{a \in A_s} r_d + \gamma \mathbf{P_d v} \tag{A.6}$$

Note that when the supremum on the right-hand side of A.7 is attained for all $v \in V$, we define L an operator on V by:

$$Lv \equiv \sup_{d \in D_{MD}} r_d + \gamma \mathbf{P_d v} \tag{A.7}$$

Lemma A.1. *Let w be a real-valued function on an arbitrary discrete set W and let $q(.)$ be a probability distribution on W. Then*

$$\sup_{u \in W} w(u) \geq \sum_{u \in W} q(u) w(u) \tag{A.8}$$

Proof. Let $w* = \sup_{u \in W} w(u)$. Then

$$w* = \sup_{u \in W} q(u) w* \geq \sum_{u \in W} q(u) w(u) \tag{A.9}$$

Note that the lemma remain valid with W a Borel subset of a measurable space, $w(u)$ an integrable function on W, and the summation replaced by Integration.

□

Proposition A.2. *For all $v \in V$ and $0 \leq \gamma \leq 1$,*

$$\sup_{d \in D_{MD}} r_d + \gamma \mathbf{P_d} \mathbf{v} = \sup_{d \in D_{MR}} r_d + \gamma \mathbf{P_d} \mathbf{v} \tag{A.10}$$

Proof. Since $D^{MR} \supset D^{MD}$, the right-hand side of A.10 must be at least as great as the left-hand side. To establish the reverse inequality, choose $v \in V, \delta \in D^{MR}$ and apply A.1 at each $s \in S$ with $W = A_s, q(\centerdot) = q_\delta(\centerdot)$, and

$$w(\centerdot) = r(s, \centerdot) + \sum_{j \in S} \gamma \mathbf{P}_d(j|s, \centerdot) v(j)$$

to show that

$$\sup_{d \in A_s} r(s, \mathbf{.}) + \sum_{j \in S} \gamma \mathbf{P}_d(j|s, \mathbf{.}) v(j) \geq \sum_{a \in A_s} q_\delta(a) [\sum_{j \in S} \gamma \mathbf{P}_d(j|s, \mathbf{.}) v(j)]$$

Therefore, for any $\delta \in D^{MR}$,

$$\sup_{d \in D_{MD}} r_d + \gamma \mathbf{P_d v} \geq r_d + \gamma \mathbf{P_d v}$$

□

From which it follows that the left-hand side of A.10 is at least as great as the right-hand side. Hence we show that we obtain the same value of the supremum in a deterministic policy as if we were to allow randomized policy.

APPENDIX B

Gaussian Processes

B.1 Gaussian Processes

Gaussian processes, as mathematical models of random phenomena that are Gaussian distribution, have attracted more and more attention in machine learning field, and they are very useful in applications. Since our work use Gaussian processes, we will give a brief introduction based on the work in *Rasmussen and K.I.Williams* [2006], *Wei and Zoubin* [2005], *Seeger* [2004], and *Hida and Hitsuda* [1993].

B.1.1 Gaussian Processes Regression and Classification

Given a data set $\{\mathbf{X}, \mathbf{y}\}$, where $\mathbf{X}$ is a matrix that is composed of N input example vectors $\mathbf{x}_c, c \in \mathcal{N}$ and $\mathbf{y}$ is a vector of corresponding targets value y_c (real value for regression or categorical value for classification). Gaussian process model treats the latent functions as random processes with Gaussian prior, which is different from the parametric form in classical statistical models. Denote the latent function by $u(\mathbf{x}_c)$, which is assumed to be a random process. Then the first and second order statistics of $u(\mathbf{x}_c)$ are its mean function $m(\mathbf{x}_c)$ and covariance function $k(\mathbf{x}_c, \mathbf{x}_d), \forall c, d \in \mathcal{N}$.

Both estimation of mean function and variance function depend on the finite dimensional distribution. Since Gaussian process is a process whose finite dimensional distributions are Gaussian, a Gaussian process is determined by its mean and variance functions. For every set of realizations of random variables of a Gaussian process, the symmetric matrix $\mathbf{K}$, whose entries are calculated by the covariance function $k(\mathbf{x}_c, \mathbf{x}_d)$, is positive semi-definite[1]. It is sufficient that for $\mathbf{K}$ is positive semi-definite, there exists a Gaussian random field with this covariance matrix and zero-mean function $m(\mathbf{x}_c) = 0$ (Kolmogorov's theorem *Kolmogorov* [1956]).We denote such random field as $u(\mathbf{x}_c) \sim N(0, \mathbf{K})$.

The simplest Gaussian process model is $y_c = u(\mathbf{x}_c) + \epsilon$, where ϵ is an independent Gaussian noise, written as $\epsilon \sim N(0, \sigma_\epsilon^2)$. Within a Bayesian framework, the inference of $u(\mathbf{x})$ at the test location $\mathbf{x}$ is described by maximization of the posterior probability

$$p(u(\mathbf{x})|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{X}, u(\mathbf{x}))p(u(\mathbf{x}))}{p(\mathbf{y}|\mathbf{X})}.$$

The joint distribution of the observed target values and the function values follows a Gaussian distribution. Therefor the posterior conditional distribution is written as

$$u(\mathbf{x})|\mathbf{X}, \mathbf{y} \sim N(\mathbf{K}(\sigma_\epsilon^2\mathbf{I} + \mathbf{K})^{-1}\mathbf{y}, \sigma_\epsilon^2(\sigma_\epsilon^2\mathbf{I} + \mathbf{K})^{-1}\mathbf{K}).$$

The generative model based on Gaussian processes paves an efficient road in which we are able to perform inference with finite observations in the large space. And it is also worth mentioning that it keeps tractable computation while offering a guarantee of performance.

[1] Positive semi-definite implies that $\mathbf{x}^T\mathbf{K}\mathbf{x} \geq 0$ for all $\mathbf{x}$.

B.1.2 Gaussian Process Preference Learning

In machine learning field, a learning scenario, called learning label preference, studies how to find the latent function that predicts preference relations among a finite set of labels, for an instance from the instance space. This scenario is a generalization of some standard settings, such as classification and label ranking *Furnkranz and Hullermeier* [2005]. Considering the latent function values as Gaussian process, Chu and Ghahramani observed that Bayesian framework is an efficient and competitive method for learning label preferences *Wei and Zoubin* [2005]. They proposed a novel likelihood function to capture the preference relations using *preference graph*, a directed graph encoding the label preferences of each sample *Aiolli and Sperduti* [2004], *Manning and Singer* [2004].

Let $u(\mathbf{x}, y)$ denote the latent function depending on both label y and instance $\mathbf{x}$, and $\mathcal{G}$ denote the observed preference graphs. The Bayesian inference is written as

$$\hat{u}(\mathbf{x}, y) \triangleq \arg\max_{u(\mathbf{x},y)} p(u(\mathbf{x}, y)|\mathcal{G}) \propto \arg\max_{u(\mathbf{x},y)} p(\mathcal{G}|u(\mathbf{x}, y))p(u(\mathbf{x}, y)) \tag{B.1}$$

where $p(\mathcal{G}|u(\mathbf{x}, y))$ is the likelihood function derived from preference relations. Given a new instance $\mathbf{x}^*$, the labels y^* can be predicted by ranking the values of latent function, $y^* = \arg\max_y \hat{u}(\mathbf{x}^*, y), y \in \mathcal{Y}$, where $\mathcal{Y}$ is a finite set of labels.

We also use Bayesian inference and build off several of the ideas in [10] and related work, but our method differs from label preference learning for classification and label ranking. Our input data depends on states and actions in the context of an MDP. Moreover, we are learning the reward that indirectly determines how actions are chosen during the sequential evolution of an MDP, while preference learning studies the latent functions preserving preferences. On the grounds of Bellman optimality for MDPs, the decision maker chooses optimal actions with the maximum value of Q-function at a given state. The preference relation will be determined by Q-functions, the expected long-term reward, while the random variable we concern is the immediate reward function, the intrinsic function determining the decision maker's behavior. Next, we give the details of our method.

APPENDIX C

Convex Optimization Problem

Proof. The second order derivative with respect to $\mathbf{r}_{a_m}$ is

$$\begin{aligned}\frac{\partial^2 U}{\partial \mathbf{r}_{a_m} \partial \mathbf{r}_{a_m}^T} &= \mathbf{K}_{a_m}^{-1} + \frac{\partial^2 \sum_{i=1}^{n} \sum_{k=1}^{n_i} - \ln \Phi(\sum_{j=1}^{m} \rho_{a_m}^{ik} \hat{\mathbf{r}}_{a_m})}{\partial \mathbf{r}_{a_m} \partial \mathbf{r}_{a_m}^T} \\ &\quad + \sum_{i=1}^{n} \sum_{l=1}^{m_i} (\rho_{a_m}^{il})^T \rho_{a_m}^{il} \end{aligned} \tag{C.1}$$

It is obvious that $\mathbf{K}_{a_m}^{-1}$ and $(\rho_{a_m}^{il})^T \rho_{a_m}^{il}$ are positive definite matrix. If the second part in Eq.C.1 is also positive definite, the minimization of Eq.5.5 is a convex problem. Let $\mathbf{W}$ denote the $n \times n$ matrix for the second part and W_{cd} be the entry at c-th row and d-th column, which is calculated in the following,

$$W_{cd} = \frac{\partial^2 \sum_{i=1}^{n} \sum_{k=1}^{n_i} - \ln \Phi(\sum_{j=1}^{m} \rho_{a_m}^{ik} \hat{\mathbf{r}}_{a_m})}{\partial \mathbf{r}_{a_m}(s_c) \partial \mathbf{r}_{a_m}(s_d)} \tag{C.2}$$

and let $z_i^k \triangleq \sum_{j=1}^m \rho_{a_m}^{ik} \hat{\mathbf{r}}_{a_m}$. We have

$$\begin{aligned}
&\frac{-\partial ln\Phi(z_i^k)}{\partial \mathbf{r}_{a_m}(s_c)} = -\frac{\rho_{a_m}^{ik}(x_c)N(z_i^k|0,1)}{\sqrt{2}\sigma\Phi(z_i^k)} \\
&\frac{-\partial^2 \ln \Phi(z_i^k)}{\partial \mathbf{r}_{a_m}(s_c)\partial \mathbf{r}_{a_m}(s_d)} = \\
&\frac{\rho_{a_m}^{ik}(s_c)\rho_{a_m}^{ij}(s_d)N(\sum_{j=1}^m \rho_{a_m}^{ik}\hat{\mathbf{r}}_{a_m}|0,1)}{2\sigma^2\Phi(z_i^k)}\left[z_i^k + \frac{N(z_i^k|0,1)}{\Phi(z_i^k)}\right]
\end{aligned}$$

where let $\omega_{ik} = \frac{N(z_i^k|0,1)}{2\sigma^2\Phi(z_i^k)}\left[z_i^k + \frac{N(z_i^k|0,1)}{\Phi(z_i^k)}\right]$, we have

$$W_{cd} = \begin{cases} \sum_{i=1}^n \sum_{k=1}^{n_i} \left[\rho_{a_m}^{ik}(s_c)\right]^2 \omega_{ik} \geq 0 \text{ if c=d} \\ \\ \sum_{i=1}^n \sum_{k=1}^{n_i} \rho_{a_m}^{ik}(s_c)\rho_{a_m}^{ik}(s_d) = W_{dc} \text{ otherwise} \end{cases} \tag{C.3}$$

Let $y = [y_1, y_2, \cdots, y_n]$ denotes a $n \times 1$ vector. Then

$$\begin{aligned}
&y^T W y = \sum_{i=1}^n \sum_{k=1}^{n_i} y_1 \sum_{b=1}^n \rho_{a_m}^{ik}(s_b)\rho_{a_m}^{ik}(s_1)y_b \\
&+\sum_{i=1}^n \sum_{k=1}^{n_i} y_2 \sum_{b=1}^n \rho_{a_m}^{ik}(s_b)\rho_{a_m}^{ik}(s_2)y_b \\
&+\cdots+\sum_{i=1}^n \sum_{k=1}^{n_i} y_n \sum_{b=1}^n \rho_{a_m}^{ik}(s_b)\rho_{a_m}^{ik}(s_n)y_b \\
&=\sum_{i=1}^n \sum_{k=1}^{n_i} [\sum_{b=1}^n y_b^2[\rho_{a_m}^{ik}(s_b)]^2 \\
&+2\sum_{b=1}^n \sum_{b'\neq b} y_b y_{b'} \rho_{a_m}^{ik}(s_b)\rho_{a_m}^{ik}(s_{b'})] \\
&=\sum_{i=1}^n \sum_{k=1}^{n_i} \left(\sum_{b=1}^n y_b \rho_{a_m}^{ik}(s_b)\right)^2 \geq 0
\end{aligned} \tag{C.4}$$

we prove the matrix W is semi-positive definite. So the Hessian matrix of Eq.5.5 is positive semi-definite on the interior of a convex set. Hence, minimizing Eq.5.5 is a convex programming problem. □

BIBLIOGRAPHY

BIBLIOGRAPHY

BIBLIOGRAPHY

Abbeel, P., and A. Y. Ng (2004), Apprenticeship learning via inverse reinforcement learning, pp. 1–.

Admati, A., and P. Pfleiderer (1988), A theory of intraday patterns: Volume and price variability, *The Review of Financial Studies*, *1*, 3–40.

Aggarwal, R. K., and G. Wu (2003), Stock market manipulation-theory and evidence, *AFA 2004 San Diego Meetings.*

Aiolli, F., and A. Sperduti (2004), Learning preferences for multiclass problems, in *Advances in Neural Information Processing Systems 17*, pp. 17–24, MIT Press.

Aldridge, I. (2010), *A Practical Guide to Algorithmic Strategies and Trading Systems - High Frequency Trading*, 339 pp., John Wiley & Sons, Inc.

Baerenklau, K., and B. Provencher (2005), Static modeling of dynamic recreation behavior: implications for prediction and welfare estimation, *Journal of Environmental Economics and Management*, *50(3)*, 617–636.

Bagnell, B. D. Z. A. M. J. A., and A. K. Dey (2008), Maximum entropy inverse reinforcement learning, *In Proceedings of the Twenty-Third AAAI on Artifical Intelligence*, p. 14331438.

Baker, C. L., R. Saxe, and J. B. Tenenbaum (2009), Action understanding as inverse planning, *Cognition.*

Barto, R. S. S. A. G., and R. J. Williams (1991), Reinforcement learning is direct adaptive optimal control, *American Control Conference*, pp. 26–28.

Bertsekas, D. (2007), *Neuro-Dynamic Programming*, Athena Scientific.

Bollerslev, T. A. T., and F. Diebold (2002), Parametric and nonparametric volatility measurement, *Handbook of Financial Econometrics. North-Holland, Amsterdam.*

Bouchaud, Z. E. J., and J. Kockelkoren (), The price impact of order book events: market orders, limit orders and cancellations, *Quantitative Finance*, *0*(0), 1–25.

Boularias, A., and B. Chaib-draa (2010), Bootstrapping apprenticeship learning, in *Advances in Neural Information Processing Systems 24*, MIT press.

Bowling, U. S. M., and R. E. Schapire (2008), Apprenticeship learning using linear programming, in *Proc. 25th international Conf. on Machine learning*, pp. 1032–1039, ACM.

Brogaard, J. (2010), High frequency trading and its impact on market quality, *Ph.D. Dissertation, Kellogg School of Management, Northwestern University.*

Burges, C. (1998), A tutorial on support vector machine for pattern recognition, *Data Mining Knowledge Discovery*, *2*, 121.

Chordia, T., R. Roll, and A. Subrahmanyam (2001), Market liquidity and trading activity, *Journal of Finance*, *56*, 501–530.

Coates, P. A. A., and A. Y. Ng (2010), Autonomous helicopter aerobatics through apprenticeship learning, *The International Journal of Robotics Research*, (1-31).

Cvitanic, J., and A. Kirilenko (2010), High frequency traders and asset prices, *Cal Tech Working Paper, California.*

D., D. N., N. Yael, and D. Peter (2005), Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control, *Nature Neuroscience*, *8*, 1704–1711.

Deepak, R., and A. Eyal (2007), Bayesian inverse reinforcement learning, in *Proc. 20th International Joint Conf. on Artificial Intelligence.*

Dey, N. R. B. Z. K. P. J. A. B. M. H. A. K., and S. Srinivasa (2009), Inverse optimal heuristic control for imitation learning, in *Proc. AISTATS*, pp. 424–431.

Dvijotham, K., and E. Todorov (2010), Inverse optimal control with linearly-solvable mdps, in *Proc. 27th International Conf. on Machine learning*, ACM.

Easley, D., M. L. de Prado, and M. O'Hara (2010), The microstructure of the "flash crash", *Cornell University Working Paper, 2010.*

Ekman, P. (1992), Intraday patterns in the s&p 500 index futures market, *The Journal of Futures Markets*, *12*, 365–381.

Eren, N., and H. N. Ozsoylev (2006), Hype and dump manipulation, *AFA 2008 New Orleans Meetings Paper.*

Farmer, J. B. J., and F. Lillo (2009), *How markets slowly digest changes in supply and demand, in Handbook of Financial Markets: Dynamics and Evolution*, 584 pp.

Farmer, J. D., and F. Lillo (2004), On the origin of power-law tails in price fluctuations, *Quantitative Finance*, *4*(1), 7–11.

Foucault, T., and A. Menkveld (2008), Competition for order flow and smart order routing systems, *Journal of Finance*, *63*, 119–158.

Furnkranz, J., and E. Hullermeier (2005), Preference learning, in *Kunstliche Intelligenz.*

Gabaix, V. P. H. E. S. X., and P. Gopikrishnan (2004), On the origin of power-law fluctuations in stock prices, *Quantitative Finance*, *4*(1), 11–15.

Gatheral, J. (2010), No-dynamic-arbitrage and market impact, *Quantitative Finance*, *10*(7), 749–759.

Hasbrouchk, J., and D. J. Seppi (2001a), Common factors in prices, order flows and liquidity, *Journal of Financial Economics*, *59*, 383–411.

Hasbrouchk, J., and D. J. Seppi (2001b), Common factors in prices, order flows, and liquidity, *Journal of Financial Economics*, *59*, 383–411.

Hasbrouck, J. (1991), Measuring the information content of stock trades, *The Journal of Finance*, *46*, 179–207.

Hasbrouck, J. (2007), *Empirical Market Microstructure: The Institutions, Economics, and Econometrics of Securities Trading*, 196 pp., Oxford University Press, 198 Madison Avenue, New York, New York 10016.

Hendel, I., and N. Aviv (2006), Measuring the implications of sales and consumer inventory behavior, *Econometrica*, *74(6)*, 1637–1673.

Hendershott, T., and R. Riordan (2008), Algorithmic trading and information, *NET Institute Working Paper*, pp. 09–08.

Hida, T., and M. Hitsuda (1993), *Gaussian Processes*, American Mathematical Society.

Hult, H., and J. Kiessling (2010), Algorithmic trading with markov chains, *Doctoral thesis, Stockholm University, Sweden 2010.*

Joachims, T. (1998), *Making large scale SVM learning practical*, MIT Press, Cambridge, MA.

Jones, C., G. Kaul, and M. Lipson (1994), Transactions, volume, and volatility, *7*(4), 631–651.

Jones, T. H. C. M., and A. J. Menkveld (2011), Does algorithmic trading improve liquidity?, *Journal of Finance*, *66*, 1–33.

Jovanovic, B., and A. Menkveld (2010), Middlemen in limit-order markets, *NYU Working Paper, New York.*

Kanto, L. K. J. T. A., and K. Kaski (1999), Characteristic times in stock market indices, *Physica A: Statistical Mechanics and its Applications*, *269*, 98–110.

Karpoff, J. M. (2004), The relation between price changes and trading volume: A survey, *Journal of Financial and Quantitative Analysis*, *22*, 109–126.

Kirilenko, A., A. S. Kyle, M. Samadi, and T. Tuzun (2011), The flash crash: The impact of high frequency trading on an electronic market, (1686004).

Knez, P., and M. Ready (1996), Estimating the profits from trading strategies, *Review of Financial Studies*, *9*, 1121.

Kockelkoren, J. B. J., and M. Potters (2006), Random walks, liquidity molasses and critical response in financial markets, *Quantitative Finance*, *6*, 115–123.

Kolmogorov, A. (1956), *Foundations of the Theory of Probability*, 2nd ed., AMS Chelsea.

Kukanov, R. C. A., and S. Stoikov (2010), Order book dynamics and price impact, *Columbia University, Working Paper.*

Kukanov, R. C. A., and S. Stoikov (2011), The price impact of order book events, *SSRN eLibrary.*

Lee, S., and P. Zoran (2010), Learning behavior styles with inverse reinforcement learning, in *SIGGRAPH '10: ACM SIGGRAPH 2010 papers*, pp. 1–7, ACM, New York, NY, USA.

Lee, Y., R. Fox, and Y. Liu (2001), Explaining intraday pattern of trading volume from the order flow data, *Journal of Business Finance and Accounting*, *28*, 306–686.

Legg, S., and M. Hutter (2004), Ergodic mdps admit self-optimising policies, *Working Paper.*

Lyons, R. K. (2006), *The Microstructure Approach to Exchange Rates*, MIT Press, Cambridge, MA US.

Macal, C. M., and M. J. North (1999), Tutorial on agent-based modelling and simulation, *Journal of Simulation*, *4*, 151162.

Mandelbrot, B. (1963), The variation of certain speculative prices, *Journal of Business*, *4*, 3082–3139.

Manning, O. D. C. D., and Y. Singer (2004), Log-linear models for label ranking, in *21st International Conference on Machine Learning.*

Meyer, P. G. V. P. L. A. M., and H. Stanley (1999), Scaling of the distribution of fluctuations of financial market indices, *Phys. Rev. E*, *60*, 53055316.

Mike, J. D. F. L. G. F. L. S., and A. Sen (2004), What really causes large price changes?, *Quantitative Finance*, *4*, 383–397.

Mike, S., and J. D. Farmer (2008), An empirical behavioral model of liquidity and volatility, *Journal of Economic Dynamics and Control*, *32*, 200234.

Miranda, M., and G. Schnitkey (1995), Estimation of dynamic agricultural decision models: The case of dairy cow replacement, *Journal of Applied Econometrics*, *10*, 41–56.

Neu, G., and C. Szepesvari (2007), Apprenticeship learning using inverse reinforcement learning and gradient methods, in *Proc. Uncertainty in Artificial Intelligence*.

Ng, A. Y., and S. Russel (2000), Algorithms for inverse reinforcement learning, *In Proc. ICML*, pp. 663–670.

Obizhaeva, A. A., and J. Wang (2005), Optimal trading strategy and supply/demand dynamics, *SSRN eLibrary*.

Oomen, R. (2005), Properties of bias-corrected realized variance under alternative sampling schemes, *Journal of Econometics*, *3*, 555–577.

Paddrik, M. E., R. H. Jr., A. Todd, S. Yang, W. Scherer, and P. Beling (2011), An agent based model of the e-mini s&p 500 and the flash crash, *SSRN Working Paper*.

Pagan, A. (1996), The econometrics of financial markets, *Journal of Empirical Finance*, *3*, 15–102.

Plerou, X. G. P. G. V., and H. E. Stanley (2003), A theory of power-law distributions in financial market fluctuations, *Nature*, *423*, 267–270.

Popovic, S. L. Z., and V. Koltun (2010), Feature construction for inverse reinforcement learning, in *Advances in Neural Information Processing Systems 24*, MIT press.

Potters, J. B. Y. G. M., and M. Wyart (2004), Fluctuations and response in financial markets: the subtle nature of random price changes, *Quantitative Finance*, *4*, 176–190.

Potters, R. C. M., and J. Bouchaud (1997), Scale invariance and beyond, *Les Houches Workshop 1997*.

Puterman, M. L. (1994), *Markov Decision Process: Discrete Stochastic Programming*, John Wiley and Sons, Inc, New York.

Qiao, Q., and P. Beling (2011), Inverse reinforcement learning with gaussian process, *Proceedings of 2011 American Control Conference*.

R., B. (1957), *Dynamic programming*, Princeton University Press.

Ramachandran, D., and E. Amir (2007), Bayesian inverse reinforcement learning, *In Proc. IJCAI*, p. 25862591.

Rasmussen, C. E., and C. K.I.Williams (2006), *Gaussian Processes for Machine Learning*, MIT Press.

Russell, S. (1998), Learning agents for uncertain environments, *Proceedings of the Eleventh Annual Conference on Computational Learning Theory.*

Rust, J. (1987), Optimal replacement of gmc bus engines: An empirical model of harold zurcher, *Econometirca*, *55*, 999–1033.

Rust, J. (1995a), Structural estimation of markov decision processes, *Handbook of Econometrics*, *IV*, 3082–3143.

Rust, J. (1995b), Estimation of dynamic structural models, problems and prospects: discrete decision processes, *Proceedings of the 6th World Congress of the Econometic Society*, *4*, 119–170.

Rust, J. (1997), Structural estimation of markov decision processes, *Review of Financial Studies*, *4*, 3082–3139.

Schaeffer, D. B. D. P. J., and D. Szafron (1998), Opponent modeling in poker, *AAAI*, p. 493498.

Schapire, U. S. R. E. (2008), A game-theoretic approach to apprenticeship learning, in *In Advances in Neural Information Processing Systems*, pp. 1449–1456, MIT Press.

Scharfstein, K. F. D., and J. Stein (1992), Herd on the street: Informational inefficiencies in a market with short-term speculation, *Journal of Finance*, *47*, 1461–1484.

Seeger, M. (2004), Gaussian processes for machine learning, *International Journal of Neural Systems*, *14*, 2004.

Sharpe, W. (1966), Mutual fund performance, *Journal of Business*, *39*, 119–138.

Stoikov, R. C. S., and R. Talreja (2010b), A stochastic model for order book dynamics, *SSRN Working Paper.*

Sutton, R. S., and A. G. Barto (1998), *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, Massachusetts.

Syed, U., and R. E. Schapire (2007), A game-theoretic approach to apprenticeship learning, *NIP*, pp. 663–670.

Vapnik, V. (1998), *Statistical Learning Theory*, Wiley, New York.

Vapnik, V. (1999), *The Nature of Statistical Learning Theory*, 2nd Edition, Spinger, New York, 1999.

Vives, X. (1995), Short-term investment and the informational efficiency of the market, *Review of Financial Studies*, *8*, 125–160.

Waelbroeck, C. S. H., and A. Mendoza (2009), Relating market impact to aggregate order flow: the role of supply and demand in explaining concavity and order flow dynamics, *Working Paper Series*.

Weber, P., and R. Bernd (2006), Large stock price changes: volume or liquidity?, *Quantitative Finance*, *6*(1), 7–14.

Wei, C., and G. Zoubin (2005), Preference learning with gaussian processes, in *Proc. 22th Iinternational Conf. on Machine learning*, pp. 137–144, ACM.

Yang, S., M. Paddrik, R. Hayes, T. Andrew, A. Kirilenko, P. Beling, and W. Scherer (2012), Behavior based learning in identifying high frequency trading strategies, *Proceedings of IEEE Computational Intelligence in Financial Engineering and Economics, 2012*.