

Inventory-based versus Prior-based Options Trading Agents*

Abraham Othman and Tuomas Sandholm
Computer Science Department
Carnegie Mellon University
{aothman, sandholm}@cs.cmu.edu

January 27, 2012

Abstract

Options are a basic, widely-traded form of financial derivative that offer payouts based on the future price of an underlying asset. The finance literature gives us option-trading algorithms that take into consideration information about how prices move over time but do not explicitly involve the trades the agent made in the past. In contrast, the prediction market literature gives us automated market-making agents (like the popular LMSR) that are event-independent and price trades based only on the inventories the agent holds. We simulate the performance of five trading agents inspired by these literatures on a large database of recent historical option prices. We find that a combination of the two approaches produced the best results in our experiments: a trading agent that keeps track of previously-made trades combined with a good prior distribution on how prices move over time. The experimental success of this synthesized trader has implications for agent design in both financial and prediction markets.

1 Introduction

Many prediction markets today rely on *automated market makers* to price contracts. These are electronic agents that stand ready to buy and sell con-

*To appear in *Algorithmic Finance*.

tracts with participants and adjust the prices they offer in response to those trades. Prediction markets use these agents because there may not be enough organic liquidity in a market to have narrow bid/ask spreads, or the event space the prediction market runs over is so large that buyers and sellers could have trouble matching their orders without the help of an intermediary [Pennock and Sami, 2007, Chen and Pennock, 2010]. These automated market making agents typically price contracts *solely* based on the trades the market-making agent made in the past (i.e., the agent’s inventory). Consequently, automated market makers are usually outcome-agnostic, which means that they can be applied to virtually any elicitation problem without particular domain knowledge. This unlocks the “long tail” of elicitation problems that are of specific, rather than general, interest. For instance, Inkling Markets, Crowdcast, and Consensus Point, which are start-up companies, offer clients the ability to use an internal prediction market mediated by an automated market maker to forecast the outcome of uncertain events that are of interest to only a small number of people (for instance, whether a project will be completed on time).

These market makers are almost always used in fake-money applications, not with actual money on the line. In a prediction market, the goal of a market maker is often to assist in information elicitation, and so the losses that inevitably result from these market makers are viewed as subsidies to encourage traders to participate and reveal their information [Hanson, 2003, Pennock and Sami, 2007, Chen and Pennock, 2010]. This reasoning provides a contrast between a market maker in a prediction market, which has a designated role to provide liquidity and can lose money acceptably, and a trading agent in a financial market which speculates for its own account. We specifically study the latter in this paper. A trading agent will make or lose money based on whether its distribution over futures states of the world is better or worse than that of its counterparties, and an outcome-agnostic agent, the norm for making prediction markets, generally has too much entropy in its prior over outcomes to profit.

The finance literature, and particularly the finance literature as it pertains to derivatives, has taken a different modeling approach than the prediction market literature. The modern derivatives literature started with the seminal work of Black-Scholes and Merton (BSM) [Black and Scholes, 1973, Merton, 1973]. By providing a practical way to price options contracts effectively, BSM led to the options markets that are the precursors of modern derivatives trading. Additionally, because the formula for calculating prices is entirely

self-contained, it provided the constructive groundwork for the notion of *autarky* within finance theory. The BSM formula takes only three inputs—the current price, volatility, and the risk-free rate of return—to produce an ensemble of options prices. Although it is problematic to generalize over the entire finance literature, speaking broadly, most models of asset pricing operate using autarky as a guiding principle. In an autarky model, prices are philosophically prior to the agents that trade on them, so these models have no reliance on an agent’s past actions in determining prices.

In this paper, we synthesize ideas from the prediction markets and finance literatures to create more successful trading agents than either literature on its own. The key insight in this paper is that these two notions: having good priors, and learning from inventories, are not oppositional. We combine them to create a trading agent that develops actionable prices based on both factors. Interestingly, the combination of the two ideas is not straightforward, and there are significant theoretical hurdles that serve to restrict the valid combinations of prior distributions and the utility models that determine how an agent reacts to its inventory.

The principal contribution of our work is the experimental simulation of five different trading strategies on a large body of recent options data:

1. A zero-intelligence agent, added as an experimental control, that trades randomly.
2. An orthodox BSM Log-normal distribution trader.
3. A Normal distribution trader, with mean and variance matched to the Log-normal trader.
4. The *Logarithmic Market Scoring Rule (LMSR)*, the most popular automated market maker in Internet prediction markets [Hanson, 2003, 2007, Pennock and Sami, 2007] which is equivalent to an exponential utility trading agent with constant uniform priors.
5. A hybrid agent that combines exponential utility with normal distribution priors.

We find that by many different measures, including expected return and worst-observed performance, the hybrid trader outperforms the other traders. Consequently, our results support the hypothesis that a trader’s current exposure can be a profitable influence on future actions, and that a trader can learn from their past actions to create a more accurate estimate of the future.

Even though the hybrid trader performs the best on several key metrics, it does not stochastically dominate the performance of the parametric traders from the finance literature. This means it is possible to construct a coherent utility function that would prefer the performance of the parametric traders. However, the relative performance of the hybrid trader relative to the parametric traders provides insights into the larger qualitative question of *how* the hybrid trader is able to perform well. We believe our results are best interpreted by the hybrid trader profitably insuring against the risk that its model of the future is inaccurate, a claim which we justify in detail.

It is important to clarify what we believe is significant about our results. We do not believe that our hybrid trader is the best options trading agent that could be devised. Certainly, BSM can be considered a theoretical model, rather than a practical trading agent, and there are likely other agents that could be constructed that would have better performance on our dataset. These agents could employ more sophisticated models about how prices move through time, detailed order book information, or outside information like press releases and macroeconomic forecasts. But what is significant about our work is this: by incorporating inventories into a Normal distribution trader, we increase performance *without incorporating any new or more sophisticated information into the trading process*. Instead, we achieve these performance gains simply by paying attention to information that autarky models discard.

As artificial intelligence researchers we are particularly keen on interpreting our results within the intellectual framework of our discipline. Much of modern, optimization-based AI is grounded in the idea of modeling (or actually constructing) a robot that can observe its environment, effect an action, and receive a reward that depends on its state and the actions it has chosen [see Russell and Norvig, 2003, Chapter 2]. This model fits naturally into trading options, where the environment is the market, the actions are the contracts to trade, and the rewards are literal monetary profits and losses. History-keeping is vital to what constitutes a *rational agent* in the AI literature. In a general environment an agent that does not retain its history *cannot be rational*. So in this context, our results are not surprising: agent performance is enhanced by both built-in knowledge about the future and the ability to learn from one's past actions.

2 Options

In this section we introduce options and their associated terminology, as well as our dataset.

2.1 What are options?

Options contracts involve the future opportunity to buy or sell some kind of underlying instrument (*the underlying*) at a set price (the *strike price*).

There are two types of option contracts. A *call* option gives its holder the right to buy an obligation at a specified price, and a *put* option gives its holder the right to sell an obligation at a specified price. These contracts have a hinged form of payouts.

Definition 1. Let the underlying expire at price π . A (European) *call option* with strike price s has value $\max(\pi - s, 0)$. A (European) *put option* with strike price s has value $\max(s - \pi, 0)$.

Options that strike close to the current price of the underlying are known as *at the money*. Options that are valuable at the current price of the underlying (high-strike puts or low-strike calls) are known as *in the money*. Options that are worthless at the current price of the underlying (low-strike puts or high-strike calls) are known as *out of the money*.

There are two principal ways that govern the exercise of options. European options expire in cash at a certain date, the *strike date*. In contrast, American options can be exercised for delivery of the underlying at any point before the strike date. The additional optionality of American options makes them necessarily at least as expensive as European options. However, this additional optionality is rarely exercised, making American options essentially European in practice. Varian [1987] provides a theoretical argument based on no-arbitrage principles for European and American options having exactly the same prices. We examine both European and American options in our dataset.

Options with the same underlying and strike date form what is called an *options chain*. Chains consist of an ensemble of contracts along with their associated prices. For instance, a chain might consist of calls and puts with strikes of 900, 1000, and 1100 for the $\tilde{\text{SPX}}$ underlying expiring on December 22, 2007. Each of these contracts has a price at which they can be bought or sold (the *ask* and *bid* prices, respectively), and theoretically all of these

prices are based on some underlying distribution over the expiration price. This distribution changes over time as the price of the underlying and the time until expiration changes. When we perform our experiments, we step through simulating trading agents on snapshots of each options chain as it evolves over time, from initiation until expiration.

2.2 Historical dataset

The dataset we use covers almost seven years of data on eleven underlyings, taken at 15-minute intervals. It is comprised of nearly 300 million {underlying, expiration date, strike price, datetime, best bid, best ask} tuples, and the corresponding {underlying, datetime, underlying best bid, underlying best ask} tuples for the underlying. The data spans from January 2004 through September 2010. To our knowledge this is the one of the more-detailed datasets used in an academic study on options—studies generally use data from daily closing prices, which is much less detailed (and one of the most widely-cited papers on empirical option pricing, Dumas et al. [1998], uses *weekly* data). Table 1 gives an overview of the dataset.

In order to provide a clean train/test separation, we divide the first two years (January 2004 through December 2005) to learn the relevant parameters for our simulation, and only test on the remaining data. A naïve split between training and testing data that randomly placed chains or days into different partitions would be tainted, because the training and test sets would overlap temporally. Consequently, only options chains that expire in 2006 and beyond are in our testing set. The number of complete chains in the testing set for each underlying is noted in the “Testing chains” column of Table 1.

3 Agents based on Log-Normal and Normal Distributions

In this section we introduce the first two traders we used in our experiments, the Log-normal and Normal distribution traders. These traders are derived from existing work in the finance literature.

Underlying	Description	Clearing	Number of records	Testing chains
^SPX	S&P 500 Index	European	53.6 million	6
^DJX	Dow-Jones Index	European	43.7 million	9
^NDX	NASDAQ Index	European	30.1 million	3
^IRX	Thirteen-week treasury bill	European	3.4 million	19
^FVX	Five-year treasury yield	European	6.2 million	19
^TNX	Ten-year treasury yield	European	6.9 million	16
^XAU	Gold-Dollar Index	European	10.1 million	16
X	US Steel	American	9.3 million	5
C	Citigroup	American	7.5 million	6
GE	General Electric	American	7.7 million	5
MSFT	Microsoft	American	8.5 million	5
XOM	Exxon-Mobil	American	7.7 million	5

Table 1: Our options dataset includes a diverse mix of underlyings, including indices, bonds, commodities, and equities of very liquid to moderately liquid contracts cleared both in American and European ways.

3.1 The BSM model

Option pricing was revolutionized and popularized by the work of Black and Scholes [1973] and Merton [1973] (BSM). Those authors described a parametric framework under which prices on the underlying change according to a log-normal distribution, which was the solution to a differential equation.

This framework was essentially unchallenged until “Black Monday” of 1987, where stock prices dropped precipitously in a single day. After this, options now show a persistent “volatility smile” or “volatility skew”, where out-of-the-money options are overpriced relative to BSM [MacKenzie, 2006]. An interpretation of this phenomenon is that the log-normal distribution of future prices as predicted by BSM is inaccurate, and the skew represents an effort to make the predicted distribution heavier-tailed. Another interpretation is that investors use options to provide insurance against the state of the world in which extremely low values of the underlying are realized.

3.2 Calculating contract values when the underlying is log-normally distributed

We use a constant (daily) volatility parameter σ for each underlying. These values are learned from our training data in order to assure a clean train/test separation. Table 2 shows the values we used in our experiments.

In addition to the current price and the volatility parameter σ , the BSM model takes an additional input, the so-called *risk-free rate of return*. This value reflects the time-cost of money. In our exploratory data analysis over our training data we did not see significant changes in performance for different realistic values of the risk-free rate (between zero and five percent annualized). For consistency, in our tests we set this value equal to zero for all the trading agents. In practice, banks, market makers, and large hedge funds will have small risk-free rates over short time horizons. Furthermore, setting this rate equal to zero means that any returns generated are exclusively from options trading, and not from interest on passive income.

Because of the popularity of the BSM model, the formulas to calculate prices from a log-normal distribution are well-known. The so-called *partial expectation* of the log-normal distribution has an analytic expression. Let f denote the density function and F the distribution function of a log-normal distribution parametrized by μ and σ . Then the partial expectation formula is:

Underlying	σ
$\hat{\text{SPX}}$.006814
$\hat{\text{DJX}}$.006781
$\hat{\text{NDX}}$.010448
$\hat{\text{IRX}}$.014504
$\hat{\text{FVX}}$.016970
$\hat{\text{TNX}}$.012679
$\hat{\text{XAU}}$.018846
X	.027520
C	.008436
GE	.009329
MSFT	.010200
XOM	.012430

Table 2: The (daily) volatility parameters σ are learned by taking the MLE of the daily changes of each underlying in the training set.

$$\int_s^\infty x f(x) dx = e^{\mu+\sigma^2/2} \Phi\left(\frac{\mu + \sigma - \log s}{\sigma}\right)$$

where $\Phi(\cdot)$ is the distribution function of the standard normal distribution.

When this value is known, the price of a call option can be calculated, because the price of a call option with strike s is

$$\begin{aligned} \int_s^\infty (x - s) f(x) dx &= \int_s^\infty x f(x) dx - \int_s^\infty s f(x) dx \\ &= e^{\mu+\sigma^2/2} \Phi\left(\frac{\mu + \sigma - \log s}{\sigma}\right) - s(1 - F(s)). \end{aligned}$$

By the use of the well-known *put-call parity* we can calculate the price of a put option at the same strike. The parity says that the price of a call at a strike, plus that strike, equals the price of a put at that strike, plus the value of the underlying. (Throughout this work, when we buy or sell underlyings it is assumed that the position will be closed when the options expire.) Once we have calculated the value of the call, the only unknown value in the put-call parity equation is the value of the put.

3.3 Calculating contract values when the underlying is normally distributed

We also implemented a trader that models the underlying’s expiration price as a normal distribution, instead of a log-normal. This trader sets the mean and variance of the normal distribution to match that of the log-normal distribution.

Normal distributions are a feature of much of the literature on market making in both prediction markets and finance. As O’Hara [1995] discusses, theoretical frameworks often model underlying prices as normal distributions because the conjugate prior to a normal distribution (with known variance) is another normal distribution. Consequently, it is common for agents to have a normal prior distribution and then update that distribution to a posterior normal distribution as more (normally-distributed) information arrives. This allows agents in models to act rationally while still retaining closed-form expressions for analytical tractability. Examples of models using normal distributions to project the future price of an asset include the classic models of Glosten and Milgrom [1985] and Kyle [1985], as well as more recent models such as that of Das and Magdon-Ismail [2009].

In the context of options, however, normal distributions are conceptually much more problematic than log-normal distributions. This is because they have support over the whole real line, rather than just over positive values. Since negative values cannot exist as termination prices (shareholders are not personally liable for the debts of a company), this makes the normal distribution inherently unrealistic^a. Recognition of this problem dates back to some of the earliest work on asset pricing [Merton, 1971]. Reflecting this intuition, our results showed that the normal distribution trader generally performed worse than the log-normal distribution trader, even though the two traders matched the mean and variance of their distributions.

To solve for prices using a normal prior, we used *Gauss-Hermite quadrature* [Judd, 1998]. Let $\mathbb{E}_{\mu,\sigma}(f)$ denote the expectation of the function f under a normal distribution with mean μ and standard deviation σ . Gauss-Hermite

^aOne way to overcome this limitation is to instead deal with truncated normal distributions, where the probability mass that exists below zero is re-distributed over the positive values (e.g., Chang and Shanker [1986]). However, these concerns are more theoretical than practical, because in our simulations the total probability mass for negative realization values appeared to be small enough that truncating the distribution would not have changed agent actions.

quadrature provides a set of nodes x_i and weights w_i such that

$$\mathbb{E}_{\mu,\sigma}(f) \approx \sum_i w_i f(x_i)$$

by implicitly converting the function f to an approximation by orthonormal polynomials.

4 The LMSR as an option trading agent

In this section we discuss how we implemented the LMSR, the *de facto* automated market maker used in Internet prediction markets, as an options trading agent.

4.1 Background

Internet prediction markets typically do not have enough organic liquidity or interest to support trade. Therefore, prediction markets can benefit from *automated market makers*—algorithmic agents that provide liquidity. In a typical market of this type, traders place bets directly with the automated market maker, who then adjusts the prices of the bets offered to traders in response to the inventory the market maker now holds. The presence of an automated market maker gives even the least (organically) liquid markets a relatively tight bid/ask spread to facilitate price discovery and the ability for traders to liquidate their positions whenever they choose. Introductions to automated market making can be found in Pennock and Sami [2007] and Chen and Pennock [2010].

Unlike market makers in financial markets, the automated market makers of prediction markets are designed to facilitate trade first and make profits second. In fact, many prediction markets are designed so that the market maker will lose money in expectation to the set of traders; these losses are viewed as a subsidy paid from the market administrator to the traders to pay for the information they supply. Because of this subsidy, most Internet prediction markets use artificial currencies or raffle tickets, rather than real cash.

The most popular automated market maker used in Internet prediction markets is the LMSR, an automated market maker with particularly desirable

properties [Hanson, 2003, 2007]. The LMSR is used by a number of companies including Inkling Markets, Consensus Point, Yahoo!, Microsoft, and the large-scale non-commercial Gates Hillman Prediction Market at Carnegie Mellon [Othman et al., 2010]. The LMSR is also the focus of academic studies about market microstructure [Othman and Sandholm, 2010b] and laboratory studies of market maker performance [Das, 2008, Brahma et al., 2010, Chakraborty et al., 2011]. We proceed to give a mathematical formulation of the LMSR.

4.2 Representation

Let the space of possible futures be exhaustively partitioned into n events, $\{\omega_1, \dots, \omega_n\}$, so that exactly one ω_i will be realized. Let \mathbf{x} be a vector of payouts, so that x_i is the amount the market maker must pay out if ω_i is realized. The LMSR is a *cost function* that translates these payout vectors into a single scalar value. The LMSR is given by

$$C(\mathbf{x}) = b \log \left(\sum_i \exp(x_i/b) \right)$$

Where $b > 0$ is an exogenous constant known as the *liquidity parameter*. Larger values of b correspond to larger worst-case losses by the LMSR, which loses at most $b \log n$. On the other hand, larger values of b produce tighter bid/ask spreads. When $b = 1$, the LMSR is equivalent to the *entropic risk measure* from the finance literature [Föllmer and Schied, 2002], however, the techniques were developed independently. (Agrawal et al. [2009] discusses the similarities between automated market makers and risk measures.)

Automated market makers work by charging a trader that wishes to move the market maker's payout vector from \mathbf{x} to \mathbf{y} the amount $C(\mathbf{y}) - C(\mathbf{x})$. Consider, for instance, an automated market maker pricing bets on a baseball game between the Red Sox and Yankees, so that the events are $\omega_1 =$ Red Sox win and $\omega_2 =$ Yankees win. Assume that $b = 100$, and that the market maker's current state is $(300, 200)$, so that the market maker must pay out 300 dollars if the Red Sox win and 200 dollars if the Yankees win. Now, imagine a trader wishes to buy a bet that would pay out 50 dollars if the Red Sox win. Accepting that bet would change the market maker's payout vector from $(300, 200)$ to $(350, 200)$. Consequently, the market maker

quotes the trader a price of

$$C((350, 200)) - C((300, 200)) \approx 39$$

dollars for the bet.

One feature of the LMSR is that the gradient of its cost function produces a probability distribution over the states

$$\nabla_i C(\mathbf{x}) = \frac{\exp(x_i/b)}{\sum_j \exp(x_j/b)}.$$

4.3 Compressing the state space

For the log-normal and normal distributions, we took the view that the underlying would expire as a continuous process. In reality, the space of expirations is countably infinite, delimited by one-cent intervals.

It is natural, however, to reduce this infinite space to a finite range of possibilities. This is a lossy operation; there is perhaps the chance the final outcome will fall outside the range we specify. Therefore, we should take a wide range. Consider the $\hat{\text{SPX}}$ underlying, which tracks the S&P 500; it has a value of around 1000. It is reasonable to assume that for near-term options, its expiration price will be between 100 and 10,000. With one-cent discretization, this implies a space of about $n = 1$ million events.

Very large event spaces like this pose two problems for the LMSR; one practical, and the other theoretical. First, the LMSR is numerically unstable over large event spaces; this problem was noted in the Gates Hillman Prediction Market [Othman and Sandholm, 2010a], and was also a problem in Yahoo's Predictalot^b (which ran over a combinatorially large event space). This numerical instability makes implementing the LMSR over very large event spaces challenging, unwieldy, and potentially inaccurate. The second problem with large event spaces is that they correspond to larger worst-case losses; in order to maintain realistic worst-case losses the b parameter would need to be much smaller, leading to a large bid/ask spread that would not facilitate much trade.

In full form, the size of the event space we would need to consider makes applying the LMSR to options markets extremely challenging. Fortunately, we can achieve a significant lossless dimensionality reduction that makes

^b<http://www.predictalot.com>

automated market making for options feasible. The key to compressing the state space is to focus only on the strike prices, not the expiration prices. Let the set of ordered strike prices be given by $\mathbf{s} = \{s_1, s_2, \dots, s_n\}$ and the market maker have corresponding payout vector $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$. This idea, of compressing the state space only to relevant strike prices, is a feature of many options trading models (e.g., Varian [1987]).

Lemma 1. *Let the expiration price be $s_i < s < s_{i+1}$, such that $s = \alpha s_i + (1 - \alpha)s_{i+1}$. Then if there exist no contracts written for a strike price between s_i and s_{i+1} , the market maker must pay out $\alpha x_i + (1 - \alpha)x_{i+1}$.*

Proof. An options contract is piecewise linear with a joint at its strike price, underlyings are linear, and our portfolio consists only of these contracts. Since a combination of linear functions is also linear, if no contract has a strike price between s_i and s_{i+1} , our payoffs will move linearly with the realized price between s_i and s_{i+1} . \square

This linearity result allows us to, in effect, collapse the continuous state space of possible prices $[s_1, s_n]$ into the set of discrete prices $\{s_1, \dots, s_n\}$ by bounding our realized loss.

Proposition 1. *Let $\bar{x} = \max_i x_i$ represent the maximum value the market maker must pay out if $s \in \{s_1, \dots, s_n\}$ is realized. Then \bar{x} is also the maximum value the market maker must pay out if $s \in [s_1, s_n]$ is realized.*

If a value outside of the strike price range is realized, we might lose more than what would be suggested by the x_i . One way to solve this problem is to include two dummy strike prices: one at zero, and another at an arbitrarily large value. This essentially prevents the final state from falling outside of the span of the strike prices. While this is appropriate for more general settings (e.g., sparsely-traded underlyings) we did not implement these dummy strikes into our LMSR trading agent. We found that for our underlyings the extreme strike prices of our option chains generally gave reasonable bounds on the expiration price.

4.4 Implementation details

Recall that trades are priced in the LMSR based on the vector of payouts currently held by the market maker. In this section we describe how to go from an inventory of options contracts to a payout vector. Let $\{s_1, \dots, s_n\}$

be the ordered set of strikes we are considering, and $\mathbf{x} = \{x_1, \dots, x_n\}$ be the payout vector corresponding to the realization of each strike. Formally, selling a call at strike s corresponds to the payout vector $\mathbf{x} = \{x_i\}_{i=1}^n$ where

$$x_i = \begin{cases} s_i - s & \text{if } s_i \geq s \\ 0 & \text{if } s_i < s \end{cases}$$

Selling a put at strike s corresponds to the payout vector

$$x_i = \begin{cases} 0 & \text{if } s_i > s \\ s - s_i & \text{if } s_i \leq s \end{cases}$$

Selling the underlying corresponds to a payout vector of

$$x_i = s_i$$

The payout associated with an event (i.e., x_i) depends directly on the strike price associated with that event (i.e., s_i). For example, selling a contract of the underlying corresponds to a payout of 30 dollars if the underlying expires at 30, and a payout of 50 dollars if the underlying expires at 50. Selling a call at a strike of 20 corresponds to a payout of 0 if the underlying expires at 20 (or below), but a payout of 30 if the underlying expires at 50.

Buying any contract induces the negative payout vector of selling that contract. Observe that when we quote the price to buy a contract the value will be negative, suggesting that we need to compensate our counterparty (i.e., pay out money for) the contract in question.

The set of strike prices we model for the LMSR trader is all the currently-offered strike prices, plus any strike prices corresponding to contracts we traded in the past. Given an inventory \mathfrak{J} of bought and sold contracts, we can calculate the payout p_i at any strike s_i by doing an element-wise sum for the payout vector \mathbf{x} of each accumulated contract:

$$p_i = \sum_{\mathbf{x} \in \mathfrak{J}} x_i$$

This cumulative payout vector \mathbf{p} is then used to price the available options in the chain; if a prospective contract induces a payout vector \mathbf{y} the LMSR trader prices the contract at

$$C(\mathbf{p} + \mathbf{y}) - C(\mathbf{p})$$

The final issue is how to set the liquidity parameter b . For our experiments we set b equal to 2500 times the initial underlying price, a value that yielded good performance over the training data. Values much smaller than this resulted in sharply diminished trade because the bid/ask spread was too large. Values much larger than this resulted in marginal prices which stayed close to a uniform distribution for the entire trading period.

4.5 The LMSR as a constant-utility cost function

In this section we introduce an alternate formulation of the LMSR as a constant-utility cost function [Chen and Pennock, 2007]. The cost functions define a utility function and then charge interacting traders precisely as much as required to maintain constant expected utility. In the finance literature, these schemes are known as *indifference pricing* [Carmona, 2009].

Definition 2. Let $u : \mathbb{R} \mapsto \mathbb{R}$ be an increasing concave function, π be a probability mass function over the ω_i , and $x^0 \in \text{dom } u$. A *constant-utility cost function* implicitly solves

$$\sum_i \pi_i u(C(\mathbf{x}) - x_i) = u(x^0)$$

The marginal prices of a constant-utility cost function are given by

$$\nabla_i C(\mathbf{x}) = \frac{\pi_i u'(C(\mathbf{x}) - x_i)}{\sum_j \pi_j u'(C(\mathbf{x}) - x_j)}$$

(see Jackwerth [2000], Chen and Pennock [2007] for details.)

By recalling that the marginal prices in the LMSR are given by

$$\nabla_i C(\mathbf{x}) = \frac{\exp(x_i/b)}{\sum_j \exp(x_j/b)}$$

it is easy to see that the LMSR is equivalent to a constant-utility cost function with $\pi_i = 1/n$ and $u(x) = -\exp(-x/b)$. In words, the LMSR is an exponential utility agent with a uniform prior over the events. In the next section, we explore how to change the prior distribution from uniform to something more accurate.

5 Trading agents based on constant exponential utility

We have presented two different ways of thinking about how to price options. The first is the traditional approach from finance, derived from projecting a distribution over the future and pricing obligations based on this projection. The second approach is derived from automated market making in prediction markets. It involves pricing obligations based only on trades previously made. In this section, we explore how to achieve a synthesis between these two ideas, creating a market maker with a good prior that also responds to past trades.

This effort is immediately complicated by the fact that models from finance typically involve generating continuous distributions over the final strike price, while the LMSR, as we discussed in the previous section, is for discrete distributions. In order to synthesize these two models one needs to either discretize a continuous distribution, or develop a continuous analogue to the LMSR. Since the existing literature in financial options is based around continuous distributions, we choose to do the latter in order to better align our work with that literature. (We study a version of the LMSR with a good discrete prior in Appendix C. We found that it performs better than the maximum-entropy LMSR, but much worse than the trader developed in this section.)

We synthesize the two methodologies by developing a version of the LMSR over continuous spaces by viewing it as a constant-utility cost function. We can generalize the constant-utility cost function framework we described in Section 4.5 to continuous event spaces. In the continuous setting, our payout vectors are functions $\mathbf{x} : \mathbb{R} \mapsto \mathbb{R}$, and cost functions become functionals that map these functions to scalars, $C : (\mathbb{R} \mapsto \mathbb{R}) \mapsto \mathbb{R}$.

Definition 3. Let μ be a probability distribution over possible expiration prices, $u : \mathbb{R} \mapsto \mathbb{R}$ be an increasing concave function, and $x^0 \in \text{dom } u$. A *continuous constant-utility cost function* $C(\mathbf{x})$ is given implicitly by the solution to

$$\int_0^\infty \mu(t) u(C(\mathbf{x}(t)) - \mathbf{x}(t)) dt = u(x^0)$$

Given this framework, it seems like it would be straightforward to combine the LMSR with the orthodox BSM forecasting model: simply set μ to be equal to the appropriate log-normal and set u equal to exponential utility.

Surprisingly, this approach does not work as planned and instead produces undefined prices for simple actions.

5.1 The undefined prices phenomenon

In a nutshell, what produces undefined prices is the tradeoff between how quickly the tails of the agent's prior distribution fall off, and how severely the trading agent's risk aversion reacts to extreme losses. If the aversion to large losses is strong enough, it can outweigh the very small probabilities associated with those large losses. The resulting trading agent would not offer to trade a contract that could produce those losses at any price.

The specific failure of exponential utility and log-normal priors to produce always-defined prices is known in the finance literature [Henderson, 2002], but working through a realistic example will shed light on how and why this pairing fails. We will then generalize the intuition gained from the example to multiple trades, distributions, and utilities, with a particular focus on what happens with exponential utility.

5.1.1 Example

For this example, we will assume a log-normal prior with $\mu = 5$ and $\sigma = 0.5$. Now consider the calculation involved in pricing the sale of the underlying. The sale of the underlying is given by the payout vector (function) $\mathbf{x}(t) = t$. With exponential utility, recall that the cost function solves, for some $v < 0$:

$$\int_0^\infty -e^{-\left(\frac{C(\mathbf{x})-\mathbf{x}(t)}{b}\right)} \mu(t) dt = v$$

Because of the form of the utility function, we can uncouple the cost, which does not feature the dummy integrating variable t , from the vector of payouts

$$\int_0^\infty -e^{\mathbf{x}(t)/b} \mu(t) dt = ve^{C(\mathbf{x})/b}$$

which shows that the cost function is defined if and only if

$$\int_0^\infty -e^{\mathbf{x}(t)/b} \mu(t) dt$$

converges. For our specific example, with the sale of the underlying and the log-normal distribution, this integral is

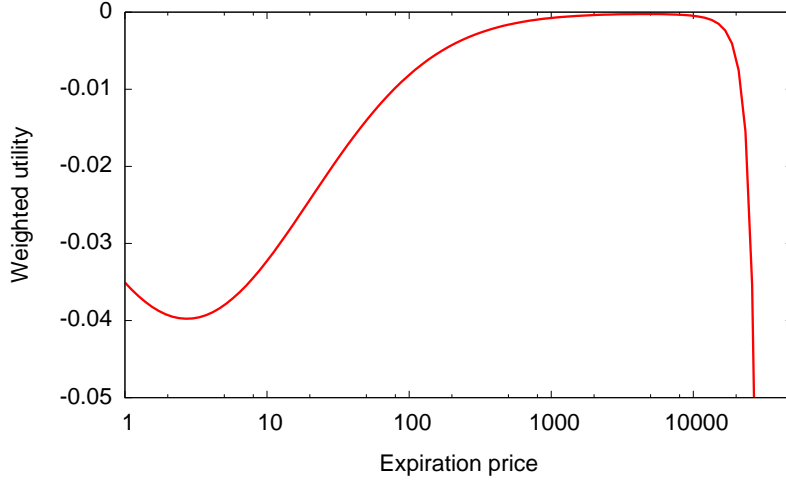


Figure 1: Because exponential utility increases faster than log-normal probability falls off, our sensitivity to large losses grows unboundedly.

$$\int_0^{\infty} -e^{t/b} \left(\frac{e^{(\log t - \mu)^2 / 2\sigma^2}}{t\sqrt{2\pi\sigma^2}} \right) dt$$

Figure 1 shows a plot of the integrand. The x -axis is log-scaled. The integrand tends towards zero for large expiration values (in the thousands, the median of the prior distribution is $e^5 \approx 148$). However, for unrealistically extreme expiration values (more than 100 times the fictional current price) the integrand explodes negatively, and the integral diverges.

Why does this occur? It is because for extremely large realization values our sensitivity to the prospect of extreme loss (since we are selling the underlying we lose when it expires high) outweighs the extremely small probabilities that the log-normal distribution produces for those values. We can show that this behavior holds regardless of the particular b, μ, σ parameterization chosen. Again, consider the pricing integral corresponding to the sale of the underlying. With a log-normal distribution and exponential utility, we have

$$\mu(t) \in \Theta(e^{-\log^2 t}) \text{ and } u(\mathbf{x}(t)) \in \Theta(e^t),$$

so

$$u(\mathbf{x}(t))\mu(t) \in \Theta(e^{t-\log^2 t}).$$

Since

$$\lim_{t \rightarrow \infty} e^{t - \log^2 t} \neq 0,$$

the pricing integral diverges. It is easy to see that this asymptotic analysis also applies to selling a call, since when we sell a call with strike price s ,

$$u(\mathbf{x}(t)) \in \Theta(e^{t-s}) \in \Theta(e^t).$$

Consequently, *there exists no amount of money an exponential utility cost function with a log-normal prior would sell an underlying or a call for.*

5.1.2 The theoretical basis of undefined prices

For any probability distribution, the density at extremely large realizations is very small, but our utility function could react to these realizations in a pronounced way. Hence, we have a tension between the density of the distribution at its tails and the response of the utility function. In this section, we provide theoretical bounds for this phenomenon, motivated by demonstrating why using exponential utility with normal priors succeeds, while with log-normal (and many other) priors it fails. Geweke [2001] also explores diverging (undefined) prices, given certain priors and utilities, but only for a specific family of utility functions.

The following result rules out using utility functions like \log and $-1/x$ with infinite-domain probability distributions.

Proposition 2. *In a continuous constant-utility cost function, if the prior probability distribution is defined over $(0, \infty)$ but the utility function is not defined over all of \mathbb{R} , then there exists a transaction with undefined price.*

Proof. We will show that the utility function will be evaluated at an undefined value. Let $\mathbf{x}(t) = t$ denote the function corresponding to selling an underlying. By assumption, the utility function is not defined at $v \in \mathbb{R}$. Now, consider $C(\mathbf{0}) > v$. Because the utility function is increasing, the cost function is increasing, and therefore $C(\mathbf{x}) > C(\mathbf{0})$. But because $C(\mathbf{x})$ is finite and the prior distribution is defined at every $t \in (0, \infty)$, there exists some t for which $C(\mathbf{x}) - \mathbf{x}(t) = C(\mathbf{x}) - t = v$, so the utility function will be evaluated at v , producing an undefined value. This argument holds by symmetry in the case where $C(\mathbf{0}) < v$, in which case we would buy the underlying instead to force the undefined evaluation. \square

The following result shows that one way to achieve well-defined costs is to only consider finite probability distributions.

Proposition 3. *In a continuous constant-utility cost function, if the utility function is defined and finite over all of \mathbb{R} and the prior distribution is bounded, so that there exists a T such that for all $t > T$, $\mu(t) = 0$, then the cost function is well-defined.*

Proof. The limits of integration are finite and the utility function is defined and finite for any argument. It follows that the integral to determine the cost function is always well-defined. \square

One application of this result to the option trading problem is to consider a trading agent with a log-normal prior distribution that is truncated above some upper boundary T . (The remaining probability mass above the upper bound could be re-distributed below the bound, or, because it is likely to be so small in magnitude, safely ignored.) Proposition 3 suggests then that the cost function would always be defined when using such a distribution.

However, this truncation scheme is numerically hazardous because it provides no guidance for *which* upper bound of T to select. Consider again Figure 1, which shows the weighted utility of a contract. Since the utility of the contract calculated without an upper bound on the prior diverges, where the upper bound T is set will have a great effect on the calculated utility of taking on the contract. To be concrete, it is clear from inspection that setting the upper bound at 1,000, 10,000, or 100,000 will produce vastly different calculations for the utility of the trader, and therefore for the fair price of the contract.

Now specifically focusing on exponential utility, it turns out we must have very light tails for the cost function to be well-defined.

Proposition 4. *A continuous exponential-utility cost function is defined for every set of options transactions if and only if*

$$\int_0^{\infty} e^{c \cdot t} \mu(t) dt$$

is bounded for every $c \geq 0$.

Proof. Recall we have defined prices if and only if

$$\int_0^{\infty} -e^{\frac{x(t)}{b}} \mu(t) dt$$

is well-defined. By removing constant factors, we see that

$$\int_0^\infty e^{\mathbf{x}(t)} \mu(t) dt$$

must be well-defined. Options contracts are continuous, piecewise-linear functions, so therefore $\mathbf{x}(t)$ is a continuous, piecewise-linear function. Therefore there exists some c such that $\mathbf{x}(t) \leq c \cdot t$, where the bound is tight by selling c underlyings. Now since e^x is an increasing function, this implies

$$\int_0^\infty e^{\mathbf{x}(t)} \mu(t) dt \leq \int_0^\infty e^{c \cdot t} \mu(t) dt$$

so if the right-hand equation is finite, the left-hand equation is finite also. \square

Recall that the expression

$$\int_0^\infty e^{c \cdot t} \mu(t) dt$$

is also known as the *moment generating function* of the distribution μ . However, moment generating functions are generally used only for the values of their derivatives at the argument $c = 0$, whereas for our result here the function must be defined for any positive c .

This result has the effect of significantly limiting what distributions we can use with exponential utility. We have already discussed how we cannot use the log-normal distribution, but other distributions, like the chi-square, exponential, and Weibull are also ruled out. (This begs the question of whether there is a theoretical or empirical reason to use these distributions.)

Now, consider that for the normal distribution,

$$\mu(t) \in \Theta(e^{-t^2})$$

This means that for arbitrary c the integrand is

$$\Theta(e^{c \cdot t - t^2})$$

which results in a well-defined integral for any c . This means we can combine a normal distribution prior with exponential utility and still get defined prices.

5.2 Our Exponential utility trader

For our Exponential utility trading agent, we use the same normal distribution prior as the Normal trader and the same b parameter as the LMSR trader. This is to better facilitate comparisons between the traders.

6 Random agent as a control

In order to provide a performance benchmark, we introduce a random trading agent as a control in our experiments. *Random* is a trading agent that performs a uniform random action over the set of possible actions at each time step. That is, for all the bids and asks on the relevant calls, puts, and the affiliated underlying, Random selects a bid or ask to trade uniformly at random. Random can be thought of as a *zero-intelligence agent* [Gode and Sunder, 1993] that does not learn or optimize.

We would expect Random to consistently produce slightly negative returns. Consider that the bid and ask prices are spread, so we should expect that an agent that takes either side of the market at random should consistently “eat” this spread. That is, we can roughly think of the Random trader as buying an option at price $p + \epsilon$ and then selling it at $p - \epsilon$, resulting in a guaranteed loss of 2ϵ .

7 Experimental setup

As we discussed in Section 2, we simulated the performance of the trading agents on each options chain in our testing set. In our simulations, we step through 15-minute increments on each chain from initiation to expiration. Figure 2 shows a flowchart of the simulation steps over each testing chain (recall the number of testing chains for each underlying is listed in Table 1). Appendix A features a worked example of the simulation process with each trading agent on one snapshot of a single option chain.

Any simulation on historical data is fraught with the risk of overfitting, producing an unreasonably rosy picture of real-world performance. We took several steps to combat overfitting. These limits are intended to give a more accurate picture of live performance than a naïve optimization over the dataset.

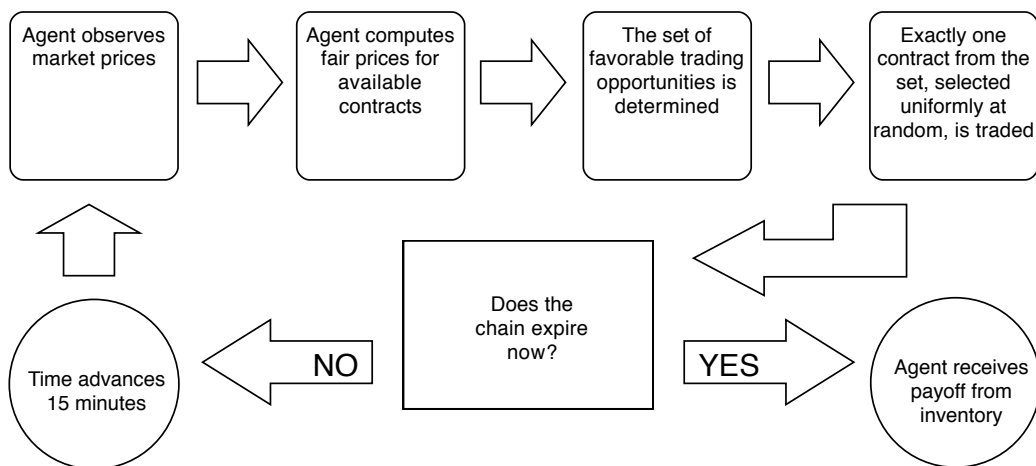


Figure 2: The steps taken by an agent over each option chain. Agents differ in the second step, the way they determine the fair prices for the offered contracts.

It is, of course, impossible in hindsight to accurately produce a counterfactual answer to the question of how well a trading bot would have performed. The bids and asks we fill could cause changes in the behavior of other traders, which is an effect we cannot judge from past data. However, Even-Dar et al. [2006] suggest that, as long as agents in the market trade on *absolute* values (fixed prices) rather than *relative* values (based on the current state of the order books), overall price series will be resistant to the small changes produced by simulation. Options markets are driven mainly by BSM-style models and often feature relatively large (compared to the underlying) bid/ask spreads. Therefore, we believe that these markets are populated mostly by traders of the absolute, rather than the relative type, making simulation by an agent trading a small amount of total volume appropriate.

Another concern related to the difference between real trading and simulation involves the effect of adding orders. In the real world, we would be able to add our own limit orders to the order book so that we would both provide as well as consume liquidity. However, for robustness we do not model this property within our data; we only take prices and do not simulate the effect of adding limit orders to the dataset. Presumably, adding the ability to place limit orders at auspicious prices would only help the performance of these trading agents in the real world.

We take two more steps to handicap the performance of our trading agents to avoid overfitting to the historical data:

- We limit the frequency of trading to only a single contract every 15 minutes. We are concerned about *slippage*—the tendency of prices to move against a trader’s actions as the trader absorbs liquidity. Trading a single contract is a conservative measure of performance that avoids slippage, because when a counterparty sets a price, the counterparty must sell at least one contract at that price.
- We choose *which* contract to trade uniformly at random from the set of desirable contracts. Say that our algorithm has identified purchasing an underlying, selling a call at 20, and buying a put at 30 to be beneficial trades given the current market prices. Then we will do only one of these, each with probability one third. This is the case even if, say, our trading algorithm thinks that buying the underlying would be better than selling the call at 20. This restriction is designed so as to not overfit on our static snapshots of prices. In a real setting, trading opportunities may arise and disappear quickly and we may not be able to trade the best opportunity that exists at a given moment.

We contrast the results of trading a random contract with trading only the contract an agent thinks is best in detail in Appendix B. Our results show the former model disadvantages the inventory-based traders more than the finance literature traders, so this restriction is conservative, as desired, for the conclusions we will draw in Section 8.

Furthermore, when we examined the trading behavior that arose from only trading the best contracts, we saw that it was biased towards trading the same contracts again and again. This is unrealistic behavior, because presumably the liquidity associated with those contracts will be exhausted if they are traded so frequently, and the prices would slip. Since a trading agent generally has several contracts it is interested in trading at a given time step, trading uniformly at random produced more realistic behavior in the simulation by spreading trading activity among several contracts.

8 Results

We begin by providing the numerical results from our experiments and then discussing those results qualitatively.

8.1 Quantitative results

In terms of real-world trading performance, it would be most appropriate to quantify performance of trading agents in terms of net annualized return (e.g., “10% a year”). Of course, net return is a function of both value generated as well as value risked. Because of the form of the options contracts, determining the value risked is not straightforward. Single contracts, like selling a call, could lose an unbounded amount of money in the worst case. Combinations of options could amplify or hedge these losses. In practice, traders need to put up a certain fraction of their positions with the exchange (the margin) in order to maintain those positions. The precise margin amount depends on the rules of the particular exchange the options are traded on and is generally based around historical models of how prices move over time.

While percent return is difficult to calculate and depends on a host of practical matters, net performance (gain or loss) is simple to calculate. Therefore, we use net performance as a measure of trading agent performance. To normalize net performance, we divide a trading agent’s gain or loss for a chain by the initial underlying price and by the number of days the chain is active. The resulting figure gives a meaningful way to compare chains where the underlying is in the thousands (like $\hat{\text{SPX}}$) or the ones (like C), over differing numbers of days. We refer to this normalized value as *net underlyings per day (NUPD)*.

Table 3 provides summary statistics for the performance of each trading agent over the 114 testing chains in terms of NUPD. The Exponential utility agent had the most positive instances, highest mean, and best worst-observed performance. The Log-normal agent had the highest median performance. The Random trading agent had the worst performance along each of these dimensions.

To assess the significance of the values in Table 3, we performed a bootstrap analysis that simulated running our experiments 10,000 times. Bootstrap analysis was a natural choice for this setting because of the complexity of estimating e.g., worst-observed loss using standard techniques [?]. The head-to-head results of this analysis is given in Table 4. For instance, the

Agent	Frac. positive	Mean	Median	Worst
Log-normal	.54	.15	.89	-6.6
Normal	.54	.12	.85	-6.7
LMSR	.46	-1.57	-.76	-37.8
Exponential utility	.55	.32	.70	-3.6
Random	.08	-1.58	-.82	-38.5

Table 3: A summary of comparing each agent along a number of dimensions. Mean, median, and worst-observed trials are measured in terms of NUPD. Higher values are better.

first value in the table indicates that in 68% of our bootstrapped experiments the Exponential utility trader had a higher fraction of positive runs than the Log-normal trader.

Trader	Frac. positive	Mean	Median	Worst	Opponent
Exp. utility	68	93	26	99	Log-normal
Exp. utility	60	95	25	100	Normal
Exp. utility	100	100	100	100	LMSR
Exp. utility	100	100	100	100	Random
Log-normal	48	57	56	98	Normal
Log-normal	100	100	100	100	LMSR
Log-normal	100	100	100	100	Random
Normal	100	100	99	100	LMSR
Normal	100	100	100	100	Random
LMSR	100	57	42	98	Random

Table 4: Percent of the bootstrapped experiments in which the trader on the left had a higher number of positive instances, mean, median, and worst-observed loss relative to the trader on the right.

Table 5 shows the relative performance of each trading agent against the others. The values are the number of testing chains in which the agent in the row beat the agent in the column. Our results establish the strict transitive ordering Exponential utility > Log-normal > Normal > LMSR > Random, where “ $a > b$ ” means that trading agent a beat trading agent b in a majority of our testing chains.

	Log-normal	Normal	LMSR	Exp. utility	Random
Log-normal	X	66	69	52	75
Normal	48	X	70	49	77
LMSR	45	44	X	41	62
Exp. utility	62	65	73	X	82
Random	39	37	52	32	X

Table 5: The number of times the agent in the row beat the agent in the column in our 114 testing chains. Majority winners are denoted in bold.

Table 6 provides statistical significance context to the results observed in Table 5. Given the produced data, the hypothesis “Trader A has a higher NUPD than Trader B on this chain” was tested for all traders and chains. If the probability that a trader outperforms the other was greater than 0.99, it was recorded as a win for that trader and a loss for the other. All the chains in which the probability lied between 1% and 99% are recorded as ties. Table 6 shows the resulting counts. All of the conclusions from Table 5 still hold; in particular, the Exponential utility trader is still the Condorcet winner.

Trader	Wins	Ties	Losses	Opponent
Exp. utility	41	40	33	Log-normal
Exp. utility	36	44	34	Normal
Exp. utility	71	5	38	LMSR
Exp. utility	75	12	27	Random
Log-normal	20	80	14	Normal
Log-normal	68	7	39	LMSR
Log-normal	68	13	33	Random
Normal	66	5	43	LMSR
Normal	71	8	35	Random
LMSR	57	9	48	Random

Table 6: Counts of head-to-head performance of traders taking into account statistical significance. If a trader had $> 99\%$ chance of out-performing its opponent on a chain, it is recorded as a “Win”. If it had $< 1\%$ chance, it is recorded as a “Loss”. All other significance levels are recorded as “Ties”.

Table 7 compares the performance of the Exponential utility trader against its “parents”, the Normal distribution trader and the LMSR trader. The Exponential utility agent beats the performance of both of these traders in 42 of the testing chains (37%) while losing to both in only 18 of the chains (16%). It outperformed a blend composed of equal parts of Normal and LMSR traders in 76 testing chains (67%).

Relative Exponential utility performance	Frequency
Beats both	.37
Beats one, loses to one	.47
Loses to both	.16
Beats blended	.67

Table 7: Distribution of the performance of the Exponential utility trader against the Normal and LMSR traders. “Blended” refers to an equal parts mix of the Normal and LMSR traders.

The NUPD of each trading agent over our testing chains can be viewed as noisy realizations of a continuous random variable. We can recover this variable by smoothing the realizations with a kernel. Figure 3 shows the kernel-smoothed CDFs of these random variables for a likelihood-maximizing Gaussian kernel. The figure plots the fraction of instances that had net performance no better than the given value. Both the LMSR and Random trading agents had chains on which they performed worse than -10 NUPD, and so the CDFs for those traders do not start 0 on the plot. The Normal and Log-normal CDFs are indistinguishably close together for much of the plot.

Our 114 chains include a mix of bonds, indices, equities, and commodities. We did not observe a substantial difference between the relative performance of our trading agents in any of the underlyings. This may be due to the fact that the volatility parameter σ is fit differently for each underlying, allowing for appropriate responses to both volatile and stable underlyings.

One notable feature of the financial markets captured in our dataset was the financial collapse of late 2008. Chains that expired in late 2008 and early 2009 showed the worst performance for our parametric traders. In particular, the Log-normal and Normal traders delivered their worst performances over the \hat{FVX} chain that expired December 20th, 2008. The performance of the underlying from the expiration of the prior chain on September 22nd to

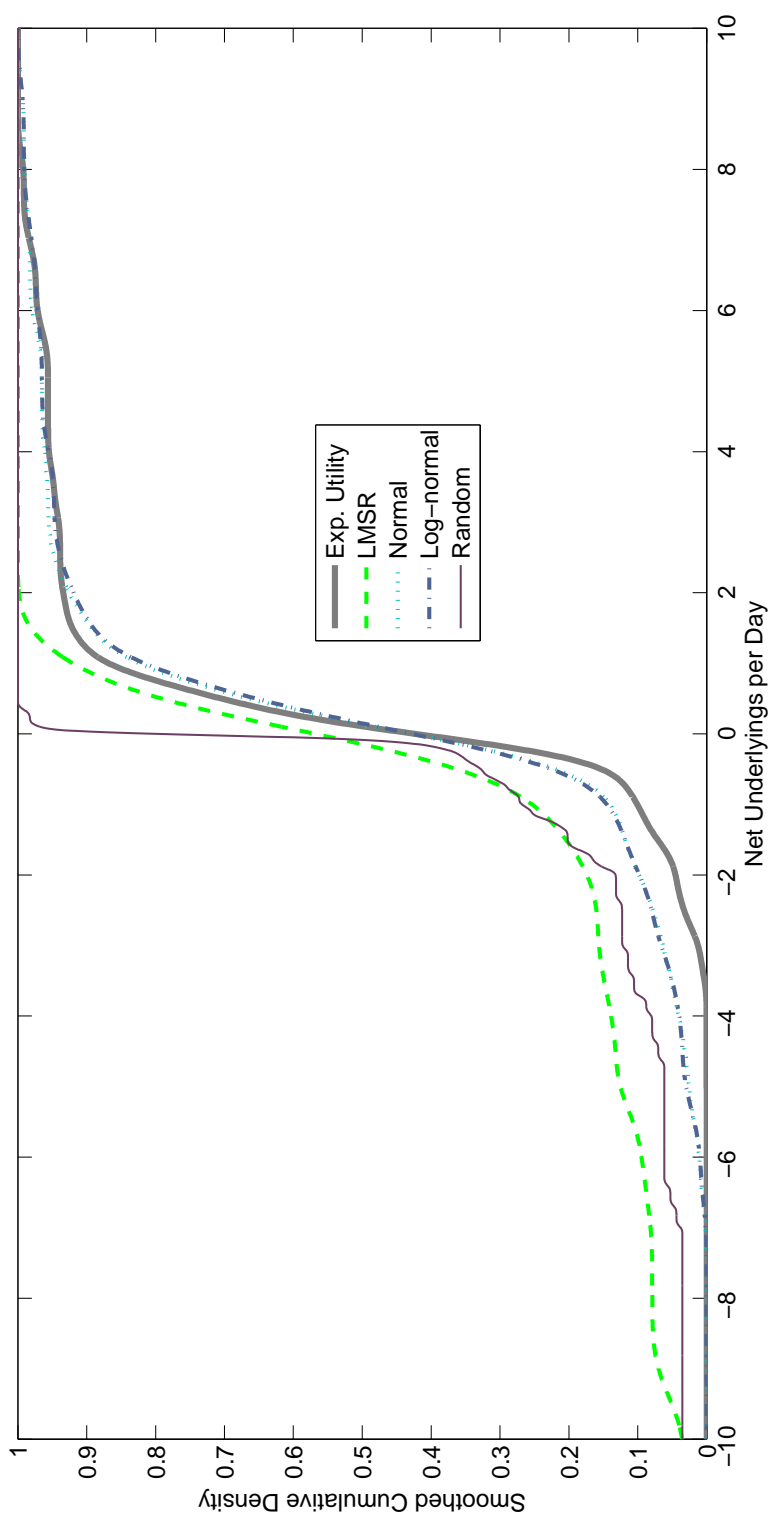


Figure 3: The kernel-smoothed CDFs of the NUPD for each of our traders. The Log-normal and Normal CDFs coincide for much of the plot. Lower function values are better.

expiry is plotted in Figure 4. There are several days in which the underlying moved down or up more than 10%. The collapse was not an “anomaly” in our dataset. It was a real event that our trading agents would have been involved in and must be considered when evaluating trading agents on real data.

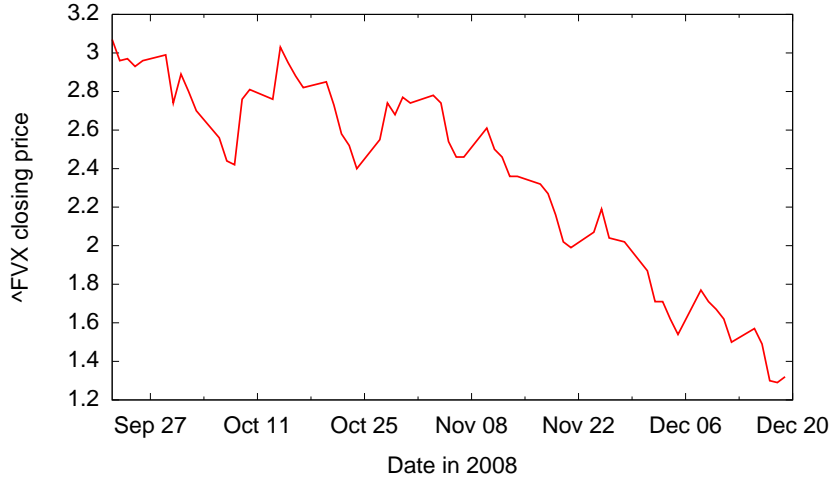


Figure 4: The option chain expiring December 20th, 2008 was particularly volatile, leading to poor performance by the parametric traders.

Finally, we did not see any significant difference in the total volume traded by each agent. We attribute this to the diversity of contracts offered to the trading agents at each time step, all of which generally have tight bid/ask spreads. In our simulation, an agent needs to find only one of the dozens of possible actions desirable at each time step in order to trade. This result could be seen as a consequence of selecting a b parameter for the inventory-based traders large enough to result in small bid/ask spreads (e.g., the snapshot example in Appendix A); substantially smaller b values would have resulted in larger bid/ask spreads in the agents’ prices and consequently less trading activity.

8.2 Qualitative results

In this section we attempt to distill our quantitative findings into qualitative facts about the performance of our automated traders.

8.2.1 The Random and LMSR traders had the worst performance

Both the Random and LMSR traders were characterized by low mean and median performance and terrifically bad worst-case losses. As Figure 3 shows, the LMSR was much more volatile than Random. Random had its performance on the vast majority of the testing chains (about 80%) fall between -2 and 0 NUPD, while the LMSR had many testing chains do better or worse.

As we have discussed, the LMSR is equivalent to an agent with exponential utility and a uniform prior over the strikes. One interpretation of this uniform prior is that the LMSR neither has nor relies on any domain knowledge. Our quantitative results with the LMSR are in line with the recent findings of Brahma et al. [2010] that suggest the LMSR struggles in comparison to trading agents with domain knowledge, and of Chakraborty et al. [2011], who compare the LMSR to a Bayesian market maker that relies on both priors and inventory. Their lab experiments showed that the Bayesian market maker was generally much more profitable than the LMSR. Furthermore, the LMSR’s results are not unexpected; it is traditionally used to provide liquidity and subsidize a set of traders for their information in Internet prediction markets. With its losses here, the LMSR did the same thing in our experiments.

One dimension along which the LMSR was able to out-perform Random significantly was the fraction of positive instances over the testing chains. Random recorded positive NUPD in fewer than eight percent of our trials, while the LMSR was positive on about 47% of the trials. Table 4 shows that in 100% of our bootstrapped experiments the LMSR trader had a higher fraction of positive instances than the Random trader. This confirms our intuition that the Random trader would “eat the spread” and lock in small losses. Interestingly, the performance of Random was the best relative to the other traders on the highly volatile chains at the end of 2008. For instance, the Random trader had the best performance of all the traders on the \hat{FVX} chain that expired December 20th, 2008, losing about 1.2 NUPD (better than its mean performance over the testing set as a whole). We credit this to the fact that the Random trader is highly non-parametric, and so its performance is not affected by the relative volatility of the underlying.

8.2.2 The LMSR learned plausible distributions

While the LMSR lagged in quantitative performance, that does not mean the concepts behind it are unsound. Figure 5 shows an in-progress run of the LMSR (on an $\hat{\text{IRX}}$ chain). The implicit probability distribution over strike prices in the LMSR closely matches the fit produced by the Normal distribution trader. This is significant because the Normal trader knows the historical volatility and the current underlying price, while the LMSR trader only knows the trades it has made. This is made more remarkable by the fact that the trading agent has only six crudely-shaped tools (buying or selling calls, puts, or the underlying) to create this distribution.

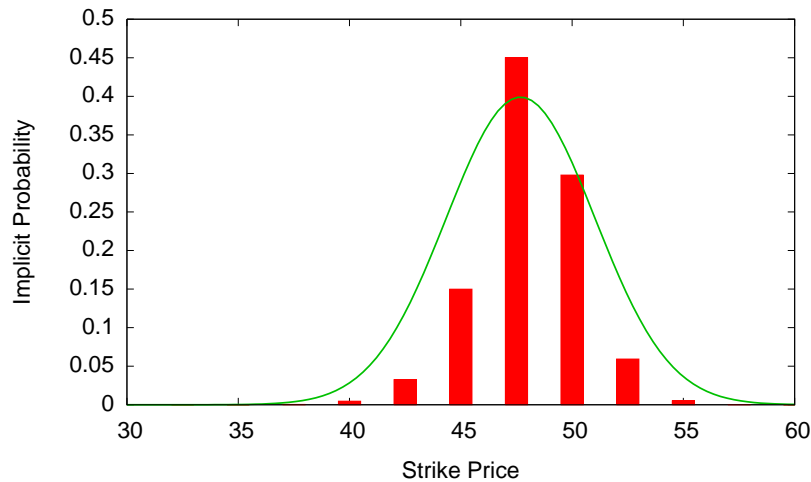


Figure 5: In this capture of an in-progress run, the LMSR trader’s implied probability distribution (red bars) closely matches the projection of the Normal distribution trader (green curve).

One perspective on what is happening is that the LMSR learns the correct distribution of prices because it is equivalent to a no-regret learning algorithm [Chen et al., 2008, Chen and Vaughan, 2010]. Essentially, with each time step through the options chain the LMSR trader makes a small correction to get its implicit probability distribution closer to the market’s distribution. This also implies the similarity between the LMSR’s probabilities and the Normal distribution trader in Figure 5 is partly fallacious, because the LMSR has not been learning from that specific snapshot of the

chain but rather making a series of small adjustments in probabilities over time.

A deeper perspective on this learning process is visible in Figure 6. This figure shows the $\hat{\text{IRX}}$ chain used above and three other (arbitrarily chosen) chains with nearby expiration dates, and measures the K-L divergence of the LMSR trader's marginal prices and the Log-normal trader's probability distribution.

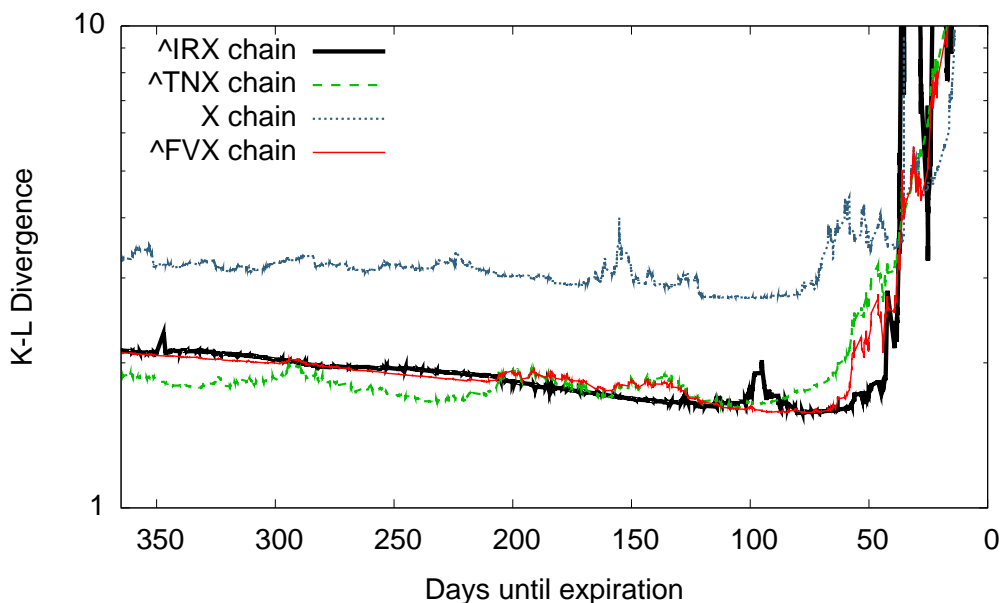


Figure 6: The K-L divergence of the LMSR from the log-normal distribution taken over four chains.

Recall that K-L divergence is a measure of how dissimilar two distributions are. (Decreasing K-L divergence means the distributions are more similar.) Formally, let LN^t denote the log-normal probability density function at time t , and let π_i^t be the marginal probability of the LMSR trader at strike price s_i . Then the K-L divergence at time t , KL^t , on the above plot is calculated as

$$KL^t \equiv \sum_i \pi_i^t \log_2 \left(\frac{\pi_i^t}{LN^t(i)} \right)$$

One interpretation of the K-L divergence is the number of extra bits required

for the log-normal distribution to encode the LMSR marginal distribution.

Although the trend is not consistent, the K-L divergence between the two distributions seems to decrease slowly until roughly 75 days before expiry, when it begins to increase significantly. We have truncated the plot at an upper boundary of 10, but in the final days before expiry the K-L divergence increases without bound. It appears to be accurate, then, to divide the LMSR’s behavior into two regimes: the last 75 days, in which the K-L divergence becomes arbitrarily large, and the time previous, in which the K-L divergence is stable or slightly falls. This earlier period represents the LMSR’s “learning” process, as the marginal prices begin to approach the log-normal distribution.

The reason the K-L divergence increases so dramatically immediately before the market expires is that the log-normal distribution becomes tighter and tighter around a single value (the current price), eventually converging to a single unit mass at the expiration price in the final time step. These progressively tighter distributions require huge numbers of additional bits to encode the more spread out LMSR distribution, because the extreme strike prices have such a low likelihood when the log-normal distribution is tight. Consequently, the K-L divergence between a tight, late-stage log-normal distribution and a more diffuse LMSR distribution is large, and as the log-normal distribution approaches a unit mass, it goes to infinity.

8.2.3 Log-normal slightly outperformed Normal

A log-normal distribution is intuitively a better and more-realistic fit for stock prices than a normal distribution, because the former reflects that the stock price cannot go below zero. Reflecting this intuition, the Log-normal trader performed better than the Normal trader in our experiments. The Log-normal trader had a slightly higher mean and median performance than the Normal trader and won the majority of head-to-head comparisons between the traders. However, the performance characteristics on the whole were close, as can be seen visually by the overlapping density lines in Figure 3. Table 4 shows that in our bootstrap analysis, neither trader had a higher fraction of positive instances, mean NUPD, or median NUPD in more than 57% of the experiments. Finally, Table 6 also shows that for the bulk of option chains (80 out of 114, or 70%), neither the Log-normal or Normal trader produced higher values with 99% confidence. This similarity could be considered as a likely consequence of the design of the two traders, because

the Normal trader matches the mean and standard deviation of the Log-normal trader at each timestep.

8.2.4 Exponential utility won but did not stochastically dominate

The Exponential utility trader was the winner in our trials by most of the measures we used. It had the highest mean and fraction of positive testing instances, a much better worst-case loss, and only a slightly lower median than the Log-normal and Normal traders. The bootstrap analysis in Table 4 suggests that the Exponential utility trader consistently outperformed the Log-normal and Normal traders in terms of mean performance and worst-case loss, but that neither the Exponential utility trader's higher fraction of positive trials nor the Normal and Log-normal traders' higher median performance were observed in more than 75% of the bootstrapped experiments. (However, the Exponential utility trader had higher mean performance in only 93 percent of the bootstrapped experiments, and so we should qualify our discussion of its relative performance by noting that more research is necessary to derive a result that is traditionally regarded as statistically significant for this specific case.)

The Exponential utility trader was also the Condorcet winner in head-to-head comparisons against the other traders, beating each of them over a majority of the testing chains. Furthermore, the Exponential utility trader outperformed a mix of the Normal and LMSR traders in two-thirds of our testing chains, indicating that it is more sophisticated than a mere combination of the two techniques. However, the Exponential utility agent did not stochastically dominate the finance literature agents, and so it is possible for some utility functions to prefer the returns of the standard BSM model instead. These utility functions would weight average- and better-case performance and discount worst-case losses.

8.2.5 Exponential utility had more accurate actionable beliefs

One perspective on how the Exponential utility agent performed so well can be found by considering the areas of Figure 3 where its CDF diverges from the CDF of the Normal and Log-normal traders. There is a large gap between the lines for negative NUPD, where the Exponential utility trader outperforms the Log-normal and Normal traders, and a smaller gap between 1 and 2 NUPD, where the Log-normal and Normal outperform Exponential utility.

What this implies is that the Exponential utility trader is practicing a form of insurance against bad outcomes. It transfers wealth between states of the world in which good things happen (the positive net return realizations) into states of the world in which bad things happen (negative net return realizations). As a result, the good cases become slightly worse (the gap between Normal/Log-normal and Exponential utility on the positive side) but the bad cases are severely reduced (the gap on the negative side).

This interpretation makes sense when we consider that the Exponential utility trader is a risk-averse analogue of the Normal trader. As a risk-averse trader the Exponential utility agent is more willing to hedge future risk, trading off future profits to avoid large losses, and will not increase its exposure to existing risks unless at the offered prices doing so seems exceptionally profitable.

One of the ways the Exponential utility trader accomplished this insurance is by having effectively heavier tails (more probability mass) on extreme cases than its corresponding normal prior. These tails make actions like buying a low-strike put or buying a high-strike call more desirable. These contracts are out of the money, and so they would require significant price movement to not expire worthless; consequently they are also priced cheaply. When a trading agent purchases one of these contracts, a small amount of wealth is transferred from states where extreme events are not realized to become a larger amount of wealth in states in which those extreme events are realized. Another way of thinking about this is that the Exponential utility trader endogenously produced the market's volatility "smirk" or "smile" by over-pricing out-of-the-money options relative to the Log-normal trader. Of course, the Exponential utility trader only produced this view as a result of the market's prodding; if the market prices in the dataset had corresponded to a volatility "frown", instead of a smile, then the Exponential utility trader would have mirrored that prediction.

This insurance allowed the Exponential utility trader to outperform the Normal and Log-normal traders, because those traders' models were not correct but those agents traded as if they were. Consider that, if a trader's beliefs are indeed the correct model of the world, then a risk-neutral agent trading on those beliefs will have a higher expected return than a risk-averse agent trading on those beliefs. (This is because a risk-neutral agent maximizes his expected return by definition.) Put another way, taking insurance should not increase a risk-neutral agent's payout if that agent was acting on correct beliefs. Since, in our experiments, the risk-averse Exponential utility

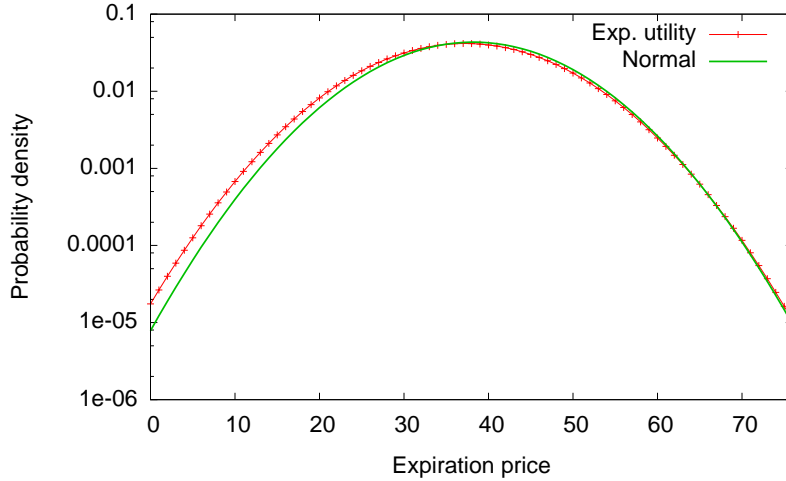


Figure 7: In this capture of an in-progress run, the heavier tails of the Exponential utility trader relative to the Normal trader are evident. The y -axis is log-scaled.

agent had higher expected returns than the risk-neutral parametric traders from the finance literature, the latter traders' models of the world were incorrect. Specifically, the heavier tails of the Exponential utility agent could be a more accurate distribution over the expiration price if there is a chance of large downward shocks to the price (e.g., in the financial crisis).

Figure 7 is an in-progress shot of heavy tails in a representative run (in this case, for a GE chain). Here, the underlying price is about 38. The implied probability distribution of the Exponential utility trader has about twice the density at low underlying realizations (the plot is log-scaled to make this more clear). For a continuous constant-utility cost function, we can calculate the implied probability density at t for vector \mathbf{x} by

$$\nabla C(\mathbf{x})(t) = \frac{\mu(t)u'(C(\mathbf{x}) - x(t))}{\int_0^\infty \mu(k)u'(C(\mathbf{x}) - x(k)) dk}$$

9 Discussion

In terms of practical impact, we acknowledge that there are considerable gaps between our experiments and the actual implementation of a trading

strategy. These include features like margin rules, which dictate how much of our currently-held position we are required to stake, and trading costs, which determine how expensive it would be to take on new positions. Of course, once these practical hurdles are known they could be incorporated into the decision logic of a trading agent, but regardless, they are likely to affect performance. However, we do believe that our experiments are robust in terms of implementation, particularly because we handicapped the trading volumes and frequencies of our agents. This was done with the intention of producing experimental results that would be more accurate reflections of the actual performance of those strategies in real markets. In summary, we are guardedly optimistic about the efficacy of this line of research in practice, although implementing these strategies in a real market poses obstacles that could deleteriously impact returns.

The framework we used to construct the Exponential utility trader, combining a utility function with a probability distribution, is very general. We believe there are significant opportunities for expanding and broadening the model in terms of each component.

Probability distribution In the time since the BSM model was first promulgated, other ways of analyzing how prices move through time have also been proposed that better fit actual performance. These include GARCH [Engle and Ng, 1993] and jump models [Kou, 2002], and they imply probability distributions over future realizations that are not log-normal. Another intriguing possibility for a prior distribution is the technique of Ait-Sahalia and Lo [1998] that uses non-parametric techniques to fit a probability distribution to the prices in an option chain.

Utility function On the other side, it would be interesting to explore other utility functions besides exponential utility. We are particularly concerned that exponential utility's extreme weighting at large values has a tendency to produce undefined prices for many distributions. Perhaps a utility function that had polynomial risk aversion, rather than exponential, would provide a more intuitive price response and would allow for the use of a broader range of prior distributions. Regardless of the utility function-prior distribution pairing, they must still obey our results in Section 5.1.2. In particular, power-law utilities (including log utility, which could be considered a standard choice) do not correspond to traders that produce meaningful prices.

Most well-established option trading strategies^c in the literature operate by trading multiple contracts in a single chain. We modeled our experimental methodology after these approaches, and in our experiments we treated each testing chain independently. But in practice, there is a correlation between chains with different expiration dates for the same underlying. For instance, if an underlying expires at a low realization, it is more likely that the chain expiring three months later will also expire at a low realization. Another extension to our model is to explore how to add a time dimension to incorporate information between the chains of different expiration dates for the same underlying. In this view, the prior distribution would be over *paths* of prices over time, and when contracts are traded at expirations it affects the probability distribution over those paths. Furthermore, different expiration dates are linked through the underlying; a trading agent may make plans to purchase an underlying now to sell at a specific date in the future, or to sell conditional on how prices move over time.

Finally, we have applied our techniques to options markets but the experimental success of the Exponential utility trader suggests traders that take into account both a good prior distribution as well as their previous trades could be successful in other real applications, too. There are many settings where we have good priors over the future, and they often involve considerable financial risk and reward; examples include a casino handling sports betting or a proprietary trading desk at a bank. We are interested in adapting the synthesized model to other trading activities where we have reasonable priors over the future and the ability to trade through time.

Acknowledgements

We thank the anonymous reviewers for their insights and feedback. We thank Jason Hitchings at Livevol for providing an academic discount for their data set. This work was supported by NSF grants CCF-1101668, IIS-0964579, and IIS-0905390, and by the Google PhD Fellowship in Market Algorithms.

^cThe Chicago Board Options Exchange (CBOE) operates <http://www.cboe.com/Strategies>, which features popular options trading strategies.

References

- S. Agrawal, E. Delage, M. Peters, Z. Wang, and Y. Ye. A unified framework for dynamic pari-mutuel information market design. In *ACM Conference on Electronic Commerce (EC)*, pages 255–264, 2009.
- Y. Ait-Sahalia and A. Lo. Nonparametric estimation of state-price densities implicit in financial asset prices. *The Journal of Finance*, 53(2):499–547, 1998.
- F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of political economy*, 81(3), 1973.
- A. Brahma, S. Das, and M. Magdon-Ismail. Comparing Prediction Market Structures, With an Application to Market Making. Technical report, Dept. of Computer Science, Rensselaer Polytechnic Institute, 2010.
- R. Carmona. *Indifference Pricing: Theory and Applications*. Princeton University Press, 2009.
- M. Chakraborty, S. Das, A. Lavoie, M. Magdon-Ismail, and Y. Naamad. Instructor rating markets. In *Conference on Auctions, Market Mechanisms and Their Applications (AMMA)*, 2011.
- J. Chang and L. Shanker. Hedging effectiveness of currency options and currency futures. *Journal of Futures Markets*, 6(2):289–305, 1986. ISSN 1096-9934.
- Y. Chen and D. M. Pennock. A utility framework for bounded-loss market makers. In *Proceedings of the 23rd Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 49–56, 2007.
- Y. Chen and D. M. Pennock. Designing markets for prediction. *AI Magazine*, 31(4), 2010.
- Y. Chen and J. W. Vaughan. A new understanding of prediction markets via no-regret learning. In *ACM Conference on Electronic Commerce (EC)*, pages 189–198, 2010.
- Y. Chen, L. Fortnow, N. Lambert, D. M. Pennock, and J. Wortman. Complexity of combinatorial market makers. In *ACM Conference on Electronic Commerce (EC)*, pages 190–199, 2008.
- S. Das. The effects of market-making on price dynamics. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 887–894, 2008.
- S. Das and M. Magdon-Ismail. Adapting to a market shock: Optimal sequential market-making. In *Advances in Neural Information Processing Systems 21 (NIPS)*, pages 361–368, 2009.

- B. Dumas, J. Fleming, and R. Whaley. Implied volatility functions: Empirical tests. *The Journal of Finance*, 53(6):2059–2106, 1998.
- R. Engle and V. Ng. Measuring and testing the impact of news on volatility. *Journal of finance*, pages 1749–1778, 1993.
- E. Even-Dar, S. M. Kakade, M. Kearns, and Y. Mansour. (In)Stability properties of limit order dynamics. In *ACM Conference on Electronic Commerce (EC)*, pages 120–129, 2006.
- H. Föllmer and A. Schied. *Stochastic Finance (Studies in Mathematics 27)*. De Gruyter, 2002.
- J. Geweke. A note on some limitations of CRRA utility. *Economics Letters*, 71(3):341–345, 2001.
- L. R. Glosten and P. R. Milgrom. Bid, ask and transaction prices in a specialist market with heterogeneously informed traders. *Journal of financial economics*, 14(1):71–100, 1985.
- D. Gode and S. Sunder. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of Political Economy*, pages 119–137, 1993.
- R. Hanson. Combinatorial information market design. *Information Systems Frontiers*, 5(1):107–119, 2003.
- R. Hanson. Logarithmic market scoring rules for modular combinatorial information aggregation. *Journal of Prediction Markets*, 1(1):1–15, 2007.
- V. Henderson. Valuation of claims on nontraded assets using utility maximization. *Mathematical Finance*, 12(4):351–373, 2002.
- J. Jackwerth. Recovering risk aversion from option prices and realized returns. *Review of Financial Studies*, 13(2):433, 2000.
- K. Judd. *Numerical methods in economics*. The MIT Press, 1998.
- S. Kou. A Jump-Diffusion Model for Option Pricing. *Management Science*, 48(8):1086–1101, 2002.
- A. S. Kyle. Continuous Auctions and Insider Trading. *Econometrica*, 53(6):1315–1335, 1985.
- N. S. Lambert, D. M. Pennock, and Y. Shoham. Eliciting properties of probability distributions. In *Proceedings of the 9th ACM conference on Electronic Commerce (EC)*, 2008.
- D. MacKenzie. *An engine, not a camera: How financial models shape markets*. The MIT Press, 2006.
- R. Merton. Theory of rational option pricing. *The Bell Journal of Economics and Management Science*, 4(1):141–183, 1973.
- R. C. Merton. Optimum consumption and portfolio rules in a continuous-

- time model. *Journal of Economic Theory*, 3(4):373–413, 1971.
- M. O’Hara. *Market Microstructure Theory*. Blackwell Publishing, 1995.
- M. Ostrovsky. Information aggregation in dynamic markets with strategic traders. In *ACM Conference on Electronic Commerce (EC)*, pages 253–254, 2009.
- A. Othman and T. Sandholm. Automated market-making in the large: the Gates Hillman prediction market. In *ACM Conference on Electronic Commerce (EC)*, pages 367–376, 2010a.
- A. Othman and T. Sandholm. When Do Markets with Simple Agents Fail? In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 865–872, Toronto, Canada, 2010b.
- A. Othman, D. M. Pennock, D. M. Reeves, and T. Sandholm. A practical liquidity-sensitive automated market maker. In *ACM Conference on Electronic Commerce (EC)*, pages 377–386, 2010.
- D. Pennock and R. Sami. Computational Aspects of Prediction Markets. In *Algorithmic Game Theory*, chapter 26, pages 651–674. Cambridge University Press, 2007.
- S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2nd edition, 2003.
- H. R. Varian. The arbitrage principle in financial economics. *J. Economic Perspectives*, 1(2):55–72, 1987.

A Snapshot of a single option chain

In this appendix we work through the pricing behavior of each agent on a single snapshot of an option chain, with the goal of giving the reader a better sense of what our experiments entailed and how our trading agents worked. We will consider the snapshot of the ^TNX option chain expiring December 19th, 2009, taken at 9:45 AM on October 22nd, 2009. This particular snapshot was chosen arbitrarily, but with attention to having a relatively small number of relevant strike prices. Table 8 shows the option chain.

Strike	Call Bid	Call Ask	Put Bid	Put Ask
17.5	15.6	20.4	X	1.5
20.0	13.1	17.9	X	1.5
22.5	10.6	15.4	X	1.5
25.0	9.0	12.0	X	1.5
27.5	6.6	9.6	X	1.5
30.0	4.6	7.0	X	1.5
32.5	2.6	5.0	X	1.5
Underlying: Current Price 34.34				
35.0	1.45	2.95	0.9	2.4
37.5	0.35	1.85	2.15	3.9
40.0	X	1.5	3.7	6.1
42.5	X	1.5	5.6	8.6
45.0	X	1.5	8.0	11.0
47.5	X	1.5	10.5	13.5
50.0	X	1.5	12.1	16.9
52.5	X	1.5	14.6	19.4
55.0	X	1.5	17.1	21.9
57.5	X	1.5	19.6	24.4
60.0	X	1.5	21.9	26.9

Table 8: The example option chain at our snapshot. Contracts that do not have open interest are designated by an “X”.

The actions of the Random trader are the simplest to describe. At this snapshot the Random trader picks one of the possible actions to perform uniformly at random. Observe that not all actions associated with the options chain are available. For instance, the Random trader cannot sell the call at strike 60, because there is nobody in the market who is offering to buy that

contract. The randomization is only over the set of contracts with open interest.

Now consider the Log-normal and Normal trading agents. Based on the historical volatility learned from the training set, the time until expiration, and the current price, the Log-normal trader sets its $\mu = 3.54$ and $\sigma = .0959$. Matching the mean and variance of this distribution, the Normal trader sets its $\mu = 34.5$ and $\sigma = 3.32$. From these values, by using the techniques we described in Section 3, we generate the prices in Table 9.

Strike	Log-normal Put	Log-normal Call	Normal Put	Normal Call
17.5	0.00	17.00	0.00	17.00
20.0	0.00	14.50	0.00	14.50
22.5	0.00	12.00	0.00	12.00
25.0	0.00	9.50	0.00	9.50
27.5	0.01	7.01	0.02	7.02
30.0	0.10	4.60	0.14	4.64
32.5	0.52	2.52	0.57	2.57
Underlying: Both value at 34.50				
35.0	1.60	1.09	1.60	1.10
37.5	3.37	0.37	3.33	0.33
40.0	5.60	0.09	5.57	0.07
42.5	8.02	0.02	8.01	0.01
45.0	10.50	0.00	10.50	0.00
47.5	13.00	0.00	13.00	0.00
50.0	15.50	0.00	15.50	0.00
52.5	18.00	0.00	18.00	0.00
55.0	20.50	0.00	20.50	0.00
57.5	23.00	0.00	23.00	0.00
60.0	25.50	0.00	25.50	0.00

Table 9: The prices generated by the Log-normal and Normal traders for our example.

By comparing the two tables, we see that the set of positive-expectation actions for both traders is buying the underlying and selling the call at 32.5. As per our rules, only one of these actions is chosen at random to be performed in this time step.

The LMSR and Exponential utility traders both depend on the set of

trades we have made in the past. For a tractable exposition that still exhibits realistic quantities, imagine that we currently hold the following portfolio:^d

1. Long 200 underlyings
2. Short 200 puts at 50
3. Short 200 puts at 45
4. Long 200 calls at 25
5. Short 200 calls at 35

This portfolio corresponds to the payout vector given in Table 10.

Strike	Payout
17.5	8500.0
20.0	7000.0
22.5	5500.0
25.0	4000.0
27.5	2000.0
30.0	0.0
32.5	-2000.0
35.0	-4000.0
37.5	-5500.0
40.0	-7000.0
42.5	-8500.0
45.0	-10000.0
47.5	-11000.0
50.0	-12000.0
52.5	-12500.0
55.0	-13000.0
57.5	-13500.0
60.0	-14000.0

Table 10: The payout vector used in the LMSR for our example.

^dIf we ran our trading agent on the chain over time, it is highly unlikely we would make 200 of the same trade, but we simulate prices with such a portfolio here to balance the competing desires of having a small number of distinct contracts with having a realistic-size portfolio.

The b parameter used in the LMSR and in the Exponential utility agents is 52,875, which is 2500 times the initial underlying price at the first instance of the chain, 21.15. Starting from our set of holdings, we can calculate the fair prices for the LMSR trader by incorporating a prospective contract into our holdings and calculating the difference in cost. Unlike in the Log-normal and Normal distribution traders, the LMSR prices the bid and ask of each contract separately, with a spread. (Due to decimal truncation, the spread is not always visible in the displayed prices.)

Strike	Call Bid	Call Ask	Put Bid	Put Ask
17.5	19.42	19.42	0.0	0.0
20.0	17.1	17.1	0.18	0.18
22.5	14.95	14.95	0.53	0.53
25.0	12.96	12.97	1.04	1.04
27.5	11.14	11.14	1.72	1.72
30.0	9.47	9.48	2.55	2.55
32.5	7.96	7.96	3.54	3.54
35.0	6.59	6.59	4.67	4.67
Underlying: Bid 36.92 Ask 36.92				
37.5	5.36	5.36	5.94	5.94
40.0	4.26	4.26	7.34	7.34
42.5	3.3	3.3	8.88	8.88
45.0	2.46	2.47	10.54	10.54
47.5	1.75	1.75	12.33	12.33
50.0	1.17	1.17	14.24	14.24
52.5	0.7	0.7	16.27	16.28
55.0	0.35	0.35	18.42	18.43
57.5	0.12	0.12	20.69	20.69
60.0	0.0	0.0	23.08	23.08

Table 11: The option prices for the LMSR trader.

By comparing Tables 8 and 11, we find that the set of positive-expectation actions for the LMSR trader is to buy the call at between 25 and 47.5 inclusive, buy the underlying, and buy the puts between 27.5 and 42.5, inclusive.

For the Exponential utility trader, we use the same b parameter as the LMSR and the same μ, σ tuple as the Normal distribution. Just like in the LMSR, but unlike in the Normal distribution, we have separate bid and ask

prices for contracts (though in this example the prices are close enough that they are equal when truncated). Table 12 displays the option chain prices for the Exponential utility trader.

Strike	Call Bid	Call Ask	Put Bid	Put Ask
17.5	16.85	16.85	0.0	0.0
20.0	14.35	14.35	0.0	0.0
22.5	11.85	11.85	0.0	0.0
25.0	9.35	9.35	0.0	0.0
27.5	6.88	6.88	0.03	0.03
30.0	4.51	4.51	0.16	0.16
32.5	2.47	2.47	0.62	0.62
Underlying: Bid 34.35 Ask 34.35				
35.0	1.04	1.04	1.69	1.69
37.5	0.31	0.31	3.46	3.46
40.0	0.06	0.06	5.71	5.71
42.5	0.01	0.01	8.16	8.16
45.0	0.0	0.0	10.65	10.65
47.5	0.0	0.0	13.15	13.15
50.0	0.0	0.0	15.65	15.65
52.5	0.0	0.0	18.15	18.15
55.0	0.0	0.0	20.65	20.65
57.5	0.0	0.0	23.15	23.15
60.0	0.0	0.0	25.65	25.65

Table 12: The option prices for the Exponential utility trader.

The set of positive-expectation actions for the Exponential utility trader is to sell the call at between 30 and 37.5 inclusive, and to buy the underlying.

B On the random uniform trading restriction

Recall that our second constraint on trading simulations was to select a contract to trade uniformly at random from the set of contracts identified as favorable. This was done to avoid overfitting on the fact that our data comes in the form of static snapshots over the option chains. In a real trading environment, favorable trades will come and go, possibly quickly enough to preclude our trading on them. Therefore, selecting uniformly at

random provides a more conservative measure of how each trading agent would perform in real settings.

With this scheme, the performance of each trading agent on each testing chain becomes a random variable. When the trading agent keeps state (as in the LMSR and Exponential utility agents), then this random variable becomes quite complex as future actions depend on the actions selected randomly in the past.

This randomization gives rise to two concerns: First, that the variance between different runs (realizations of this random variable) is large enough to mitigate the significance of the differences between agent performance. Second, that uniformly trading, rather than trading the best contract from the set of actions, unfairly penalizes some agents over others. In this section, we study each of these concerns in turn.

B.1 The difference between runs was generally small

We completed four runs over each agent over each testing chain. To measure volatility, we took the maximum difference in NUPD between the four runs. This can be considered an adversarial measure that is particularly sensitive to the outliers from each run. The resulting differences appear in Table 13.

Agent	Percent with max difference smaller than					
	0.0625	0.125	0.25	0.5	1.0	2.0
Log-normal	64	83	96	99	99	99
Normal	66	83	92	97	97	98
LMSR	67	83	92	96	97	98
Exponential utility	68	82	92	97	97	98
Random	46	65	83	92	96	96

Table 13: We performed four runs over each trading agent in the dataset. The percent of chains with difference between maximum and minimum NUPD is noted for each agent.

These results indicate that the runs were fairly robust over different realizations of trading strategies. For all but the Random agent, about two-thirds of all runs had all four experimental runs fall in a range less than 1/16 NUPD wide, and those agents had all of their runs fall within a 0.5 NUPD band in more than 95% of the chains.

As could be expected, the Random agent showed particularly large volatility between different runs. This is because the Random agent had the largest set of possible actions in each time step (i.e., all the available contracts). The Random agent over the $\tilde{\text{XAU}}$ chain expiring on 2009-09-30 had the largest difference in trials overall, with NUPD of -0.10, -0.11, -10.3, and -10.4. This appears to be due to a possible anomaly within the dataset: the best ask on a put option at 105 on September 28th, 2009 is listed in our data set at 200,000 dollars. When the Random agent buys this contract, it induces a massive loss for the chain as a whole. It is unclear whether this contract was actually listed at this price on the exchange, or if it is an error in the data set. Regardless, because this price was so uncompetitive the other, intelligent, trading agents were able to avoid it.

B.2 Trading only the best contracts produces no qualitative changes

To see whether our results were significantly affected by trading uniformly at random from the set of contracts they deemed favorable, we also simulated our trading agents trading only the best (highest expected profit) contract from the set of contracts they deemed favorable. Observe that this produces a deterministic trading agent. (Consequently, the Random trading agent is no longer relevant in this setting and is omitted.) We are interested in testing whether our qualitative results are merely an artifact of the restriction to trading uniformly or not. We want to examine how the agents from the finance literature perform relative to the Exponential utility agent when only the best contract is traded.

Table 14 shows the mean and median NUPD of the deterministic agents relative to trading uniformly. They are phrased in terms of *surplus NUPD*, subtracting the deterministic NUPD for each chain from the average NUPD of that chain when trading occurs uniformly at random.

As might be expected, trading only the best contracts produces slightly higher median NUPD for all the deterministic agents. The LMSR agent is notable for having significantly higher mean surplus NUPD than the other agents. This appears to be due to a combination of two factors: trading only the best contracts allows the LMSR agent to avoid making some of the worst trades, and furthermore, the LMSR agent's performance was already poor enough to allow for a large boost. These results suggest that trading only the

Agent	Surplus NUPD	
	Mean	Median
Log-normal	0.00	0.04
Normal	0.04	0.09
LMSR	1.13	0.10
Exp. utility	0.26	0.13

Table 14: The difference in NUPD between deterministic agents that only trade the contracts they think will deliver the best return versus trading uniformly at random from the set of agreeable trades, over the 114 testing chains.

best contracts does not change the qualitative performance ordering of the agents: the Exponential utility trader outperforms the Log-normal trader, which does about as well as the Normal trader, which in turn outperforms the LMSR.

One feature of simulating only the best contract that we observed, especially for the parametric finance literature traders, was the tendency to buy (or sell) exactly the same contract over and over again. It is easy to see why this would be the case if there is not a tremendous amount of movement in the market in-between 15 minute intervals; in this case, the most agreeable contract at the current time step is likely to be the most agreeable contract 15 minutes later, as well. Particularly for contracts at extreme strike prices, this kind of behavior is inherently unrealistic in a simulation because sustained trade in a single contract is likely to move market prices. The Exponential utility and LMSR agents are able to mitigate this phenomenon because as they trade a contract, further expansion of that position becomes less desirable. However, our principal way of simulating in the paper—taking a uniform sample from the set of favorable trades—is a more conservative simulation. As we have shown here though focusing on only the best contracts does not change our qualitative conclusions.

C The LMSR with a better discrete prior

Recall that there were two differences between the Exponential utility trader and the LMSR trader: a better prior, and a continuous relaxation of the

event space. To investigate which of these changes was responsible for the improvement in performance, in this section we examine incorporating a good discrete prior into the traditional LMSR.

C.1 The discrete prior

There are two equivalent ways to incorporate a prior into the LMSR. In general, let the trader’s prior belief over the set of events be π_1, \dots, π_n .

The first way is to go back to the definition of marginal prices within constant-utility cost functions:

$$\nabla_i C(\mathbf{x}) = \frac{\pi_i u'(C(\mathbf{x}) - x_i)}{\sum_j \pi_j u'(C(\mathbf{x}) - x_j)}$$

This equation gives an easy way to incorporate the prior into prices. Since in the LMSR, $u(x) = -\exp(-x/b)$, these prices correspond to the following cost function

$$C(\mathbf{x}) = b \log \left(\sum_j \pi_j \exp(x_j/b) \right)$$

The second way is based on the relation of the LMSR to the logarithmic proper scoring rule. The connection between automated market makers and scoring rules is deep, and is the focus of much prior literature [Hanson, 2007, Pennock and Sami, 2007, Lambert et al., 2008, Chen and Pennock, 2010].

Here, we initialize a “starting vector” \mathbf{q}^0 as the payouts implied by the logarithmic scoring rule:

$$q_i^0 = b \log(\pi_i)$$

Then the cost function proceeds as usual, but with the addition of the \mathbf{q}^0 vector of payouts, so that

$$C(\mathbf{x}) = b \log \left(\sum_j \exp((x_j + q_j^0)/b) \right)$$

These two methodologies are in fact equivalent because

$$\pi_i \exp(x_i/b) = \exp(\log(\pi_i)) \exp(x_i/b) = \exp((x_i + b \log(\pi_i))/b) = \exp((x_i + q_i^0)/b)$$

We form the *Discrete-prior LMSR trader* by setting

$$\pi_i \propto LN(x_i)$$

where LN is the density function of the Log-normal trader for the same option chain at the same snapshot in time. (As the prior of the Log-normal trader changes over time, the discrete prior for the LMSR trader changes, too.) To facilitate comparison, we keep the b parameter the same as in the LMSR and Exponential utility trading agents, and we use the same compression of the state space to strike prices that we developed and motivated in Section 4.3.

C.2 Results and discussion

We replicated our experiments from Section 7 with the Discrete-prior LMSR trader. Table 15 compares the Discrete-prior LMSR trader against the LMSR, Log-normal, and Exponential utility traders. The Discrete-prior LMSR trader outperforms the LMSR trader over 63% of the option chains, but loses a similar frequency of head-to-head matchups against the Log-normal and Exponential utility trader.

Comparison Trader	Fraction
LMSR	.37
Log-normal	.61
Exponential utility	.64

Table 15: Fraction of options chains on which the LMSR, Log-normal, and Exponential utility traders out-performed the Discrete-prior LMSR.

Table 16 compares the performance of the Discrete-prior LMSR with the Log-normal, LMSR, and Exponential utility traders. The Discrete-prior LMSR had performance that closely matches the original LMSR, although with slightly better mean, median, and worst-case loss.

Agent	Frac. positive	Mean	Median	Worst
Discrete-prior LMSR	.46	-1.54	-.73	-37.7
Log-normal	.54	.15	.89	-6.6
LMSR	.46	-1.57	-.76	-37.8
Exponential utility	.55	.32	.70	-3.6

Table 16: Comparison of the Discrete-prior LMSR with other trading agents.

While the Discrete-prior LMSR trader outperformed the LMSR trader, the difference was slight. The common factor between the two agents was the compression of the state space to just the relevant strike prices, and our results suggest this compression was the dominant factor in determining the performance of the traders. Since both the Log-normal and Exponential utility traders have much better performance than the Discrete-prior LMSR, this suggests that as the number of events (future expirations) considered increases, the performance of the discrete-prior LMSR would improve. This is because as the number of events increases, the discrete market maker becomes a closer approximation to these continuous market makers. However, as we have discussed, an increasing number of events makes the computation of values much more challenging, and compressing the state space to only the traded strike prices made a huge state space tractable without opening the trader up to unbounded loss. Still, it seems that compressing the state space to strike prices, while not affecting worst-case loss, does come with the hidden cost of a much less expressive prior.

One issue that should not be lost in this discussion is that using a discrete market maker with a fine-grained prior does not provide a way to circumvent the impossibility results of Section 5. Consider using the LMSR with a large number of expiration prices (events) with a prior that matches the distribution of the Log-normal trader. The prior distribution that is actually being generated is a log-normal distribution truncated at the upper bound of the event space (the largest expiration price considered). This truncation scheme is covered by Proposition 3, which shows that exponential utility will produce meaningful prices if the prior distribution is identically zero above some upper bound. However, as we discussed in Section 5, this truncation is numerically hazardous because its effect on prices is unpredictable and highly sensitive to the upper bound.