A Practical Liquidity-Sensitive Automated Market Maker

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

David M. Pennock, Daniel M. Reeves
Yahoo! Research New York
{dpennock,dreeves}@yahoo-inc.com

ABSTRACT

Current automated market makers over binary events suffer from two problems that make them impractical. First, they are unable to adapt to liquidity, so trades cause prices to move the same amount in both thick and thin markets. Second, under normal circumstances, the market maker runs at a deficit. In this paper, we construct a market maker that is both sensitive to liquidity and can run at a profit. Our market maker has bounded loss for any initial level of liquidity and, as the initial level of liquidity approaches zero, worst-case loss approaches zero. For any level of initial liquidity we can establish a boundary in market state space such that, if the market terminates within that boundary, the market maker books a profit regardless of the realized outcome. Furthermore, we provide guidance as to how our market maker can be implemented over very large event spaces through a novel cost-function-based sampling method.

Categories and Subject Descriptors

J.4 [**Social and Behavioral Sciences**]: Economics

General Terms

Economics

Keywords

Market Design, Automated Market Making, Prediction Markets, Pricing Rules, Market Scoring Rules, Liquidity

1. INTRODUCTION

In liquid markets like the New York Stock Exchange nearly every asset has open interest, providing two benefits: (1) price takers can buy or sell at any time, and (2) observers can continually monitor a precise value of every asset. A prediction market, or market explicitly designed to uncover the value of an asset, relies heavily on (2) holding true. If an asset has poor price support (i.e., no open interest or large bid-ask spread), then observers learn little or nothing about its value, disabling the very purpose of the market. For example, some popular contracts on intrade.com, one of the largest prediction markets, attract millions of dollars in trades. Yet still thousands of other Intrade contracts suffer from low liquidity and thus reveal little in the way of predictive information.

Prediction markets therefore benefit from automated market makers, or algorithmic traders that maintain constant open interest on every asset, providing liquidity that may be hard to support organically. Combinatorial prediction markets with vast numbers of outcomes to predict (e.g., a 64-team tournament with 2^{63} or 9.2 quintillion outcomes) almost seem nonsensical without some form of automated pricing. Companies like WeatherBill and Bet365 (sports) are beginning to use proprietary automated market makers to offer instantaneous price quotes across thousands or millions of highly customizable assets.

Hanson's logarithmic market scoring rule (LMSR) is an automated market maker with particularly nice properties and behavior (Hanson, 2003, 2007). 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 and Sandholm, 2010a). It is also the focus of a number of academic studies about market microstructure (Ostrovsky, 2009; Othman and Sandholm, 2010b). (Other companies like HSX.com and Crowdcast employ their own automated market makers.)

The amount of liquidity in LMSR is a parameter set *a priori* before the market maker knows what bets traders will place. Setting the liquidity is more art than science—a constant dilemma for almost everyone who has implemented LMSR. Too little liquidity makes prices fluctuate wildly after every trade; too much makes prices barely budge even following large bets. Exacerbating the problem, the amount prices move for a fixed bet in LMSR is a constant. The 1,000,001st dollar moves the price as much as the first, counter to intuitive notions of liquidity.

Higher liquidity is good for traders but comes at the cost of increasing the market maker's worst-case loss. In general, an LMSR operator can expect to lose money in proportion to the liquidity it provides (Pennock and Sami, 2007). The cost is rationalized as payment for traders' information. Yet subsidized markets are the exception rather than the rule. The vast majority of markets run at a profit. It's no coincidence that most examples of LMSR in practice are games based on virtual currency rather than real money.

EC'10, June 7–11, 2010, Cambridge, Massachusetts, USA.

We present a variant of LMSR that is better suited for practical use in two ways. First, our market maker automatically adjusts how easily prices change according to how much activity it sees: prices become less elastic as more dollars flow in. The market operator need not somehow try to anticipate traders' level of interest to set liquidity manually. Second, our market maker can ensure an arbitrarily small loss in the worst case and a positive profit over a wide range of final states. Unlike in LMSR, prices of disjoint assets can sum to greater than $1. However, we prove that dropping this arbitrage property is a theoretical requirement for any liquidity sensitive and path independent market maker. Moreover, relaxing the arbitrage property is precisely what allows our market maker to expect a profit, more closely resembling typical bookmaker policies.

In Section 2 we motivate the properties of pricing rules from first principles using vector calculus. We show that no market maker can satisfy three desirable properties: path independence, no-arbitrage, and liquidity sensitivity.

With this motivation, in Section 3 we proceed to weaken the least natural property, no-arbitrage, introducing a new market maker that can run at a profit. We illustrate the features of our market maker in detail, including obtaining tight bounds on the sum of prices.

Finally, in Section 4, we discuss how to implement our market maker over very large event spaces. We describe how current sampling techniques based around estimating prices do not work well with our market maker, and introduce a novel cost-function-based sampling method that makes very large event spaces tractable.

2. PRICING RULES

In this section we derive from first principles the properties of pricing rules from vector calculus. This study will allow us to explore the central tension behind automated market making: that no market maker can be liquidity sensitive, path-independent, and no-arbitrage. This axiomatic characterization is distinct from the work of both Chen and Pennock (2007), who explore utility-based market makers, and Agrawal et al. (2009), who use convex optimization to synthesize different strands of automated market making.

2.1 Vector calculus for pricing rules

We begin by partitioning the event space into n distinct exhaustive events, exactly one of which will occur. The state of the market is kept by the vector $\mathbf{q}$, whose i-th element determines the payout owed to traders if the i-th event occurs. Pricing rules translate between this quantity vector and prices.

Definition 1. A *pricing rule* is a differentiable function $\mathbf{p} : \mathbb{R}^n \mapsto (0,1)^n$ that maps a vector of quantities to a vector of prices.

For convenience we sometimes use the notation $\mathbf{p}(q_i, \mathbf{q_{-i}})$ where we isolate one element, q_i, of the quantity vector from the other elements, $\mathbf{q_{-i}}$.

2.1.1 Three required properties

Regardless of further descriptions, all pricing rules should have three properties: they should have a *convex pre-image*, they should satisfy a *monotonicity* property, and they should be appropriately *surjective*. We now proceed to discuss these three properties and to explain why they are natural requirements.

The first required property of pricing rules is that they have a *convex pre-image*. (However, we do not require that the pre-image of the pricing rule encompasses the entire domain $\mathbb{R}^n$.) Convexity is a natural property. Imagine a trader holding a portfolio $\mathbf{q}$. Convexity ensures that the trader can sell any fraction of that portfolio back to the market maker and still have defined prices. We now define this notion formally.

Definition 2. A point in $\mathbb{R}^n$ is *valid* if it is in the pre-image of the pricing rule $\mathbf{p}$.

Definition 3. **(Convex pre-image)** Pricing rule $\mathbf{p}$ has a *convex pre-image* if all convex combinations of valid vectors are also valid.

The second required property of pricing rules is *monotonicity*. The first part of the monotonicity definition below is natural to require: the more demand there is, the higher the price should be. The second part reflects the fact that the events i and j are natural substitutes because the events form an exhaustive and disjoint partition of the space: exactly one event will occur.

Definition 4. **(Monotonicity)** A pricing rule is *monotonic* if for every i, the price, p_i, is monotonically increasing in q_i and monotonically decreasing in every $q_j, j \neq i$.

The third requirement of pricing rules is *surjection*: the rule always allows any final price vector to be reached in a straightforward manner from any initial price vector.

Definition 5. **(Surjection)** A pricing rule is *surjective* if for every i and $\mathbf{q_{-i}}$, there exists a q_i such that $p_i(q_i, \mathbf{q_{-i}}) = x$ for all $x \in (0,1)$.

Throughout the rest of the paper we will assume these three properties.

2.1.2 Three desired properties

We can also identify three additional properties we would like a pricing rule to have: that it be *path independent*, that it be *no-arbitrage*, and that it be *liquidity sensitive*.

Path independence means that any way the market moves from one state to another state yields the same payment or cost to the traders in aggregate (Hanson, 2003). We proceed to give a technical definition.

Definition 6. **(Path independence)** Pricing rule $\mathbf{p}$ is *path independent* if every piece-wise smooth closed curve of transactions is budget-balanced.

Compared to convexity, path independence fits less squarely with our traditional conception of how markets work. For instance, consider a traditional continuous double auction with a bid/ask spread. A trader who buys and then sells a quantity of shares will incur a loss (which shrinks as the bid/ask spread becomes smaller, i.e., for higher levels of liquidity). But path dependence is not necessarily driven only by a bid/ask spread. Consider a market maker that incorporates prior transactions into the prices that it presents to traders, as in Das (2008). Since different trades will affect these market makers' internal state, they may be path dependent.

Path independence offers three important benefits. First, it is a sufficient condition for ensuring that there does not exist a money pump in the market: a trader cannot place a series of trades and profit without assuming some risk. Second, it provides a minimum representation of state: we only need to know the quantity vector. Finally, because a trader gets the same odds from participating all at once as in a set of small trades, traders do not need to strategize how they make trades (e.g., making a series of small purchases instead of a single large trade).

Now, an important connection follows immediately from vector calculus:

PROPOSITION 1. *If a pricing rule* $\mathbf{p}$ *is path-independent and has a convex pre-image, then* $\mathbf{p}$ *is the gradient of a scalar potential field.*

Tying this to convention in the prediction market literature, we call this scalar field a *cost function* and denote it by $C(\cdot)$. The cost function maps vectors of quantities to a single scalar value, and prices are determined by the partial derivatives with respect to each coordinate of the vector.

The cost function represents the (path-independent) integral over instantaneous prices, so it is a measure of how much money has been paid into the system. To view this another way, imagine that a set of traders, collectively, has d dollars and the market is initially at state $C(\mathbf{q^0})$. After all the traders invest all their money, the combined holdings of the traders can be those vectors with non-negative components $\mathbf{q}$ such that

$$C(\mathbf{q^0} + \mathbf{q}) = C(\mathbf{q^0}) + d$$

The second desired property is *no-arbitrage* (Agrawal et al., 2009): that the cost of buying a guaranteed payout of x always costs x.

Definition 7. **(No-arbitrage)** A pricing rule is *no-arbitrage* if prices always sum to unity. Formally:

$$\sum_i p_i(\mathbf{q}) = 1$$

for all valid $\mathbf{q}$.

Most prediction markets in use do not preserve no-arbitrage, and with good reason: a no-arbitrage pricing rule ensures that the market maker will take a loss as long as the final market prices are better than the initial market prices, a condition that is essentially tautological (if it were false, there would be little need to run a market in the first place). The simplest way to see this is to characterize the way market makers function in standard, familiar markets. A market maker takes on a risk when setting prices: if the prices are not the actual expected final prices, the market maker has a negative expectation. Market makers counter this risk by charging different prices on both market sides so that they are both more expensive than unity. Then, the market maker profits from traders purchasing on both sides of the market, leaving a cut (aka the "spread" or "vig") for the market maker. A no-arbitrage rule shrinks the size of the spread to zero, leaving the market maker exposed to the negative downside risk of offering prices without any upside.

Conversely, the no-arbitrage condition guarantees that no trader can arbitrage (exploit without risk) the market maker by taking on a guaranteed payout for less than the payout.

The most direct benefit of a no-arbitrage pricing rule is that it preserves the direct translation between the *price* of an event and the *probability* of that event occurring. Both prices and probabilities will be non-negative and (when exhaustively partitioned) will sum to unity.

No-arbitrage rules also guarantee the "law of one price", so that if two bets offer the same payouts in all states, they will have the same price. Put another way, imagine the Yankees and Red Sox are playing a baseball game. The law of one price asserts that placing a bet of a certain amount towards the Yankees winning will be priced equivalently to placing a bet of the same amount on the Red Sox losing. While logically straightforward, this condition does not necessarily hold in practice in traditional continuous double auctions, as the administrators of the Iowa Electronic Markets have discussed (Berg et al., 2001; Oliven and Rietz, 2004). This condition can thus also be viewed as a necessary condition for efficient information aggregation in a market. If the law of one price is not satisfied, there are opportunities for unsophisticated traders to pay too much or get paid too little.

As the third desired property, we would like market makers to adjust the elasticity of their pricing response based on the volume of activity in the market. We call market makers that are unable to adjust in this way *liquidity insensitive.*

Definition 8. **(Liquidity sensitivity)** Define the n-dimensional vector $\mathbf{1} \equiv (1, 1, \ldots, 1)$. A pricing rule is *liquidity insensitive* if

$$p_i(\mathbf{q} + \alpha\mathbf{1}) = p_i(\mathbf{q})$$

for all valid $\mathbf{q}$ and all α.

Sensitivity to liquidity is desirable because it squares intuitively with the way we would want markets to function: small investments move prices less in thick (liquid) markets than in thin (illiquid) markets.

One can also think about sensitivity from a Bayesian perspective. The 1000th flip of a coin moves the posterior estimate of that coin's probability of coming up heads much less than the first flip. This is because, after 1000 flips, we already have a great deal of information about the probability of the coin coming up heads. Similarly, if we have a lot of information about the objective price of a contract (a deep market), small bets in the market should not impact prices much.

2.1.3 Tension among the desired properties

In this section we show that no market maker can satisfy all three of the desired properties.

Definition 9. Any market maker that satisfies no-arbitrage and path-independence is a *Hanson market maker.*

This name is inspired by Robin Hanson, who provided an approach to building such market makers from strictly proper scoring rules. All of the example market makers given by Hanson and subsequent authors (Agrawal et al., 2009) are liquidity insensitive. We now show why: liquidity sensitivity is in fact impossible to achieve in the Hanson context.

PROPOSITION 2. *No pricing rule is no-arbitrage, path-independent, and liquidity sensitive.*

PROOF. We prove this result by showing that a Hanson market maker, which is by definition no-arbitrage and path-independent, has constant prices along $\mathbf{1}$ and is therefore liquidity insensitive.

Because Hanson market makers are path independent, prices are given by the gradient of a scalar field, the cost function. Consider the Hessian of that cost function

$$\nabla^2 C(\cdot) = \begin{bmatrix} \frac{\partial p_1}{\partial q_1} & \cdots & \frac{\partial p_n}{\partial q_1} \\ \vdots & \ddots & \\ \frac{\partial p_1}{\partial q_n} & \cdots & \frac{\partial p_n}{\partial q_n} \end{bmatrix}$$

This matrix has strictly positive values along its diagonal, and strictly negative values everywhere else. Denoting a positive entry by a "+" and a negative entry by a "−", this matrix has the form

$$\begin{bmatrix} + & - & \cdots & - \\ - & + & \cdots & - \\ \vdots & & \ddots & \\ - & \cdots & - & + \end{bmatrix}$$

The sum of the entries of the i-th row of this matrix represents the change in sum of prices from adjusting q_i. Since prices always sum to 1, the rows of the matrix sum to 0. By the symmetry of second derivatives, the Hessian matrix is symmetric, so the entries in each column also sum to 0. The sum of the entries of the i-th column of this matrix represents the directional derivative along $\mathbf{1}$ of p_i, and thus prices are constant along $\mathbf{1}$. ■

2.2 The LMSR

Our pricing rule is derived from the *logarithmic market scoring rule (LMSR)* (Hanson, 2003). The LMSR uses the following cost function:

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

where $b > 0$ is some constant. This function's pre-image is the entire space $\mathbb{R}^n$. The function's gradient, the pricing rule, is

$$p_i(\mathbf{q}) = \frac{\exp(q_i/b)}{\sum_j \exp(q_j/b)}$$

This cost function has worst-case loss $b \log n$ for the market maker. (This loss is achieved by starting from identical prices on all events.)

The LMSR is the most well-studied rule and most widely used in practice for prediction markets. It is used by a number of startup companies including Inkling Markets and Consensus Point. It is used internally at Microsoft and has been used in online games run by Yahoo! and at Carnegie Mellon University. Before its demise, Tradesports, a large sports-betting site, was beginning to incorporate a discrete version of the logarithmic rule into some of its markets.

3. OUR MARKET MAKER

In this section we will derive our automated market maker. Since we cannot satisfy all three desiderata simultaneously, we should consider which of them to loosen. Of the three desiderata, the least natural constraint is no-arbitrage, because it does not match how we would expect a market to function in the real world. In particular, we would like our market maker to be able to derive a profit from transacting with traders. So, rather than enforcing the no-arbitrage condition

$$\sum_i p_i(\mathbf{q}) = 1$$

for all valid $\mathbf{q}$, we would actually prefer

$$\sum_i p_i(\mathbf{q}) \geq 1$$

That way, if traders cannot take on negative quantities, the prices they face always sum to at least one.

3.1 Imposing a transaction cost and subsidizing liquidity

One approach to make a Hanson rule more practical is to directly impose a transaction cost on each trade. That is, bets are calculated from the regular Hanson rule, but an additional charge (e.g., 3%) is added to every transaction presented to a potential bettor. For instance, if we present a trader with a bet that would normally cost 1 dollar according to the Hanson rule, it would instead cost 1.03. The market maker can then keep 0.03.

Imposing a transaction cost enables a market maker to potentially run at a profit, assuming a sufficient level of market activity. However, this scheme is still not liquidity sensitive—prices respond identically to bets at all different volumes.

A second and more complex idea is to break the transaction fee between increasing liquidity and pocketing a fee. For instance, a market maker can charge a 3% fee, but only keep 1%, putting the other 2% towards increasing the b parameter (i.e., increasing the parameter so that the worst-case loss is larger by the amount of this 2% subsidy). Such a market maker would be liquidity sensitive and can run at a profit, but it has two shortcomings. First, increasing b in this manner distorts prices towards the mean, bringing the price of each contract towards $1/n$. Second, it breaks path independence, because a series of smaller orders will result in more updates to b than a single large order.

The market maker we introduce in the next section can be thought of as a way of adapting this scheme *continuously* with order volume, so that prices are not distorted and so that path independence is maintained.

3.2 Our cost function

The conventional LMSR cost function can be written as

$$C(\mathbf{q}) = b(\mathbf{q}) \log \left(\sum_i \exp(q_i/b(\mathbf{q})) \right),$$

where $b(\mathbf{q}) = b$ is an exogenously set constant. Instead, our market maker uses the LMSR cost function, but with a variable $b(\mathbf{q})$ that increases with market volume as follows:

$$b(\mathbf{q}) = \alpha \sum_i q_i,$$

where $\alpha > 0$ is a constant. The valid region for our market maker is the set of n-dimensional vectors with all non-negative components, excepting the origin.

3.3 Theoretical properties

This simple change results in a host of intriguing properties.

3.3.1 Prices

In a path-independent market maker, the price of state i is given by the partial derivative of the cost function along i. With constant b, this expression is simply

$$p_i(\mathbf{q}) = \frac{\exp(q_i/b)}{\sum_j \exp(q_j/b)}$$

When $b(\mathbf{q}) = \alpha \sum_i q_i$, however, the expression becomes much more complex:

$$p_i(\mathbf{q}) = \alpha \log\left(\sum_j \exp(q_j/b(\mathbf{q}))\right) + \frac{\sum_j q_j \exp(q_i/b(\mathbf{q})) - \sum_j q_j \exp(q_j/b(\mathbf{q}))}{\sum_j q_j \sum_j \exp(q_j/b(\mathbf{q}))}$$

Figure 1 illustrates the liquidity sensitivity of these prices in a 2-event market. As the number of shares of the complementary event increases, the market's price response for a fixed-size investment becomes less pronounced.

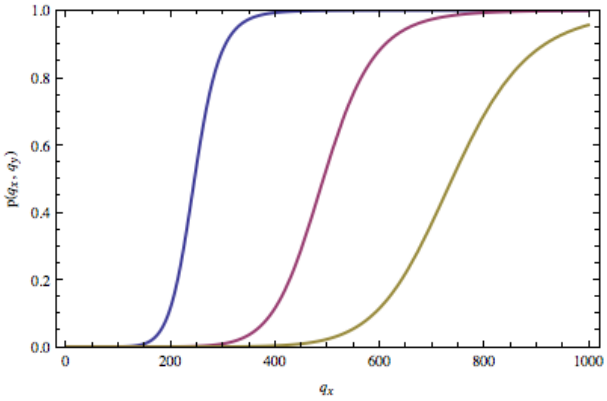

Figure 1: In a 2-event market with $\alpha = .05$, this plot illustrates the relationship between q_x and p_x for $q_y = 250$, 500, and 750, respectively. The liquidity sensitivity of our market maker is evident in the decreasing slope of the price response for increasing q_y.

3.3.2 Tight bounds on the sum of prices

In this section, we establish tight bounds on the sum of prices. In particular, we show that

$$1 \leq \sum_i p_i(\mathbf{q}) \leq 1 + \alpha n \log n,$$

where the prices achieve the upper bound only when $\mathbf{q} = k\mathbf{1}$ for $k > 0$, and prices achieve the lower bound as $q_i \to \infty$. (Recall that $\mathbf{1}$ is the vector where each element is a 1, so the product $k\mathbf{1}$ yields a vector where each element is a k.)

PROPOSITION 3. *Prices at $k\mathbf{1}$, for all $k > 0$, sum to $1 + \alpha n \log n$.*

PROOF. For $\mathbf{q} = k\mathbf{1}$, we have $q_i = q_j$ for all i and j, which allows us to simplify considerably.

$$\begin{aligned}\sum_i p_i(k\mathbf{1}) &= \sum_i \alpha \log\left(\sum_j \exp(q_j/b(\mathbf{q}))\right) \\ &= n\alpha \log\left(\sum_j \exp(q_j/b(\mathbf{q}))\right) \\ &= n\alpha \log\left(n \exp\left(\frac{1}{\alpha n}\right)\right) \\ &= n\alpha \log\left(\exp\left(\frac{1}{\alpha n}\right)\right) + n\alpha \log n \\ &= 1 + \alpha n \log n\end{aligned}$$

■

PROPOSITION 4. *The maximum of the sum of prices is obtained at every point of the form $k\mathbf{1}$, where $k > 0$. Furthermore, these are the only points that achieve the maximum.*

PROOF. Consider the set of all quantity vectors that sum to $\bar{b} > 0$. We will show that the quantity vector where each event has equal quantity (each one having $\bar{b}/n$) maximizes the sum of prices.

The sum of prices at quantity vector $\mathbf{q}$ is given by

$$\sum_i p_i(\mathbf{q})$$

Without loss of generality, take $\sum_i q_i = 1/\alpha$, so that the space of vectors we consider are those for which $b(\mathbf{q}) = 1$.

So without loss of generality we can rewrite the sum of prices as

$$1 + n\alpha \left[\log\left(\sum_j \exp(q_j)\right) - \frac{\sum_j q_j \exp(q_j)}{\sum_j \exp(q_j)}\right]$$

We will show that

$$\log\left(\sum_j \exp(q_j)\right) - \frac{\sum_j q_j \exp(q_j)}{\sum_j \exp(q_j)} \leq \log n,$$

with equality occurring only when $\mathbf{q} = k\mathbf{1}$. We can rewrite the above expression as

$$\sum_j q_j \exp(q_j) \geq \left(\sum_j \exp(q_j)\right) \log\left(\frac{\sum_j \exp(q_j)}{n}\right)$$

Take $p_j \equiv \exp(q_j)$. The expression then becomes

$$\sum_j p_j \log(p_j) \geq \sum_j p_j \log\left(\frac{\sum_j p_j}{n}\right)$$

Without loss of generality, we can scale the p_j to define a probability distribution, to get

$$\begin{aligned}\sum_j p_j \log(p_j) &\geq \log\left(\frac{\sum_j p_j}{n}\right) \\ &\geq -\log(n)\end{aligned}$$

This is a result from basic information theory, which establishes that the uniform distribution has maximum entropy over all possible probability distributions (Cover and Thomas, 1991). Therefore, equality holds only in the case of a uniform distribution, which corresponds to the quantity vector having equal components ($\mathbf{q} = k\mathbf{1}$). ■

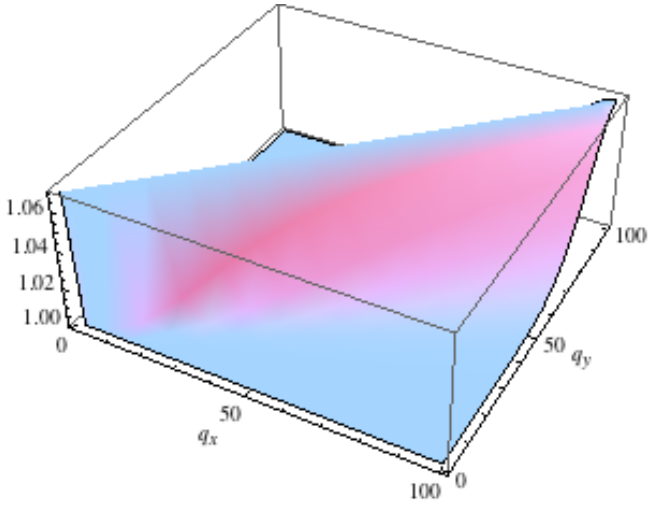

Figure 2: Sum of prices where $n = 2$ and $\alpha = 0.05$. The sum is bounded between 1 and $1+\alpha n \log n \approx 1.07$, achieving its maximum where $q_x = q_y$.

PROPOSITION 5. *At any valid* $\mathbf{q}$, $\sum_i p_i(\mathbf{q}) \geq 1$.

PROOF. Once again, without loss of generality we can rewrite the sum of prices as

$$1 + n\alpha \left[\log \left(\sum_j \exp(q_j) \right) - \frac{\sum_j q_j \exp(q_j)}{\sum_j \exp(q_j)} \right]$$

We will show that

$$\log \left(\sum_j \exp(q_j) \right) - \frac{\sum_j q_j \exp(q_j)}{\sum_j \exp(q_j)} \geq 0$$

Rewriting with $p_j \equiv \log q_j$ this expression becomes

$$\log \left(\sum_j p_j \right) \sum_j p_j \geq \sum_j p_j \log p_j$$

Without loss of generality we can normalize so that the p_j form a probability distribution. The expression then simplifies to

$$\sum_j p_j \log p_j \leq 0,$$

which holds because the entropy of a distribution is never positive. Here, equality holds only in the limit as the probability distribution becomes a point mass on a single point. In our original $\mathbf{q}$ space, this is equivalent to $\mathbf{q}$ having a component approaching infinite quantity. ■

Figure 2 is a plot of the sum of prices in a simple two-quantity market. Prices achieve their highest sum when $q_x = q_y$ and are bounded below by 1.

3.3.3 Selecting α

A possible complaint about our scheme is that we have replaced one *a priori* fixed value, b, of the LMSR with another *a priori* fixed value, our α. In this section, we discuss how the α parameter has a natural interpretation that makes its selection relatively straightforward.

The α parameter can be thought of as the commission taken by the market maker. Higher values of α correspond to larger commissions, which leads to more revenue. At the same time, setting α too large discourages trade.

As we have shown, the sum of prices with our market maker is bounded by $1 + \alpha n \log n$, and this value is achieved only when all quantities are equal. This insight provides a guide to help set α.

How large should administrators set α within our market maker? We can look to existing market makers (and bookies) for an answer. Market makers generally operate with a commission of somewhere between 5 and 20 percent. To emulate a commission that does not exceed v in our market maker, one can simply set

$$\alpha = \frac{v}{n \log n}$$

So, the larger the event space (larger n), the smaller α should be set to maintain a given percentage commission.

3.3.4 Bounded loss

One concern with any pricing rule is whether the market maker's loss is bounded. Of course, every non-trivial[1] pricing rule has some monetary loss in the worst case, but it is highly undesirable for a pricing rule to lose an infinite amount in some cases.

The condition that the pricing rule not lose an infinite amount is stricter than surjection and monotonicity together (which imply that the price of an event approaches 1 as the quantity of that event gets large). For instance, if prices approach 1 as $1 - \frac{1}{q_i}$, then loss, $\log q_i$, is unbounded as $q_i \to \infty$. We now formally define loss and boundedness of the loss.

Definition 10. The *loss* of a market maker that starts in state $\mathbf{q^0}$ and ends in state $\mathbf{q}$, with the realization of event i, is

$$C(\mathbf{q^0}) - C(\mathbf{q}) + q_i$$

Recall that here $C(\cdot)$ is the cost function and q_i is the amount the market maker has to pay out in the end upon event i occurring.

Definition 11. A pricing rule has *bounded loss* if for all initial states $\mathbf{q^0}$ and all states $\mathbf{q}$,

$$C(\mathbf{q^0}) - C(\mathbf{q}) + q_i < \infty$$

PROPOSITION 6. *Our pricing rule has bounded loss. Specifically, it loses at most* $C(\mathbf{q^0})$.

PROOF. We will show that the term $q_i - C(\mathbf{q})$ is non-positive and is zero in the limit as one of the q_i grow large. By inspection,

$$q_i = b(\mathbf{q}) \log \left(e^{q_i / b(\mathbf{q})} \right) \leq b(\mathbf{q}) \log \left(\sum_j e^{q_j / b(\mathbf{q})} \right) = C(\mathbf{q})$$

Therefore our market maker has bounded loss, because the term $C(\mathbf{q^0})$ is finite. Now consider the limit as one of the q_i gets large. Without loss of generality, consider the market starting from $\mathbf{1}$. Then let

$$b(q_i) \equiv b(q_i, \mathbf{1_{-q_i}}) = \alpha(q_i + n - 1)$$

[1]Agrawal et al. (2009) present a market maker that never incurs a loss, but it is not practical since it charges for every bet at least the amount the trader would win in any state, and therefore traders never have incentive to participate.

Then

$$\begin{aligned}
&\lim_{q_i \to \infty} q_i - C(q_i, \mathbf{1}_{-q_i}) \\
&\quad = \lim_{q_i \to \infty} q_i - b(q_i) \log \left(e^{\frac{q_i}{b(q_i)}} + \sum_{i \neq q_i} e^{\frac{1}{b(q_i)}} \right) \\
&\quad = \lim_{q_i \to \infty} q_i - \alpha(q_i + n - 1) \log \left(e^{\frac{q_i}{\alpha(q_i + n - 1)}} + \sum_{i \neq q_i} e^{\frac{1}{b(q_i)}} \right) \\
&\quad = q_i - \alpha q_i \log(\exp(1/\alpha)) \\
&\quad = 0
\end{aligned}$$

Since this limit is zero, the market maker suffers zero worst-case loss from the $q_i - C(\mathbf{q})$ term in the loss expression. Therefore, our market maker has worst-case loss equal to the amount of the initial subsidy, $C(\mathbf{q^0})$. ■

Since

$$\lim_{\mathbf{q} \to \mathbf{0}} C(\mathbf{q}) = 0,$$

setting the initial market quantities close to $\mathbf{0}$, worst-case loss becomes arbitrarily small. Reducing the initial vector too much comes at a cost, however, because

$$\lim_{\mathbf{q} \to \mathbf{0}} b(\mathbf{q}) = 0$$

so the market becomes arbitrarily sensitive to small bets in its initial stage.

In contrast, to get near-zero loss in the LMSR, one would have to set b near zero, which would cause arbitrary sensitivity to small bets throughout the duration of the market. Since other Hanson market makers are not liquidity sensitive either, they suffer from the same problem. In our market maker, by setting the initial quantities close to zero, we achieve near-zero loss while containing the high sensitivity to the initial stage only.

3.3.5 Worst-case revenue

In addition to always having bounded loss (and near-zero loss if desired), under broad conditions on the final state of the market, we can guarantee that our market maker actually makes a profit (regardless of which event gets realized). The worst-case revenue is

$$\mathfrak{R}(\mathbf{q}) \equiv C(\mathbf{q}) - \max_i q_i - C(\mathbf{q^0})$$

If $\mathfrak{R}(\mathbf{q}) > 0$ when the market closes, the market maker will book a profit regardless of the outcome that is realized. Figures 3 and 4 show the set of market states for which $\mathfrak{R}(\mathbf{q}) > 0$ for various values of α and initial quantity vectors $\mathbf{q^0}$. Figure 3 shows varying values of α. Figure 4 shows varying initial quantity vectors. It might appear that large portions of the state space will result in our market maker losing money. However, prices and quantities have a highly non-linear relationship: prices quickly approach 1 as quantities become imbalanced. The straight black rays on the plane represent a price of .95 for one of the two events. Therefore, the plots indicate that as long as markets are terminated while events have reasonable levels of uncertainty, the market maker can easily book a profit even in the worst case.

3.3.6 Homogeneity

In this section we examine and explore the implications of our cost function being homogeneous.

PROPOSITION 7. *Our cost function is positive homogeneous of degree one.*

PROOF. Let $\gamma > 0$ be a scalar and $\mathbf{q}$ be some valid quantity vector. Without loss of generality, we can assume $\sum_i q_i = 1$. Then

$$\begin{aligned}
C(\gamma \mathbf{q}) &= b(\gamma \mathbf{q}) \log \left(\sum_i \exp(\gamma q_i / b(\gamma \mathbf{q})) \right) \\
&= \gamma \alpha \log \left(\sum_i \exp \left(\frac{\gamma q_i}{\gamma \alpha} \right) \right) \\
&= \gamma C(\mathbf{q})
\end{aligned}$$

■

It is crucial that the cost function be homogeneous of degree one because that allows the price response to scale appropriately in response to increased quantities. In fact, only homogeneous cost functions provide this price response.

Definition 12. Prices *scale proportionately* if

$$p_i(\mathbf{q}) = p_i(\gamma \mathbf{q})$$

for all i, $\mathbf{q}$ and scalar $\gamma > 0$.

PROPOSITION 8. *Prices scale proportionately if and only if the cost function is positive homogeneous of degree one.*

PROOF. Proportional scaling is equivalent to the price functions being homogeneous of degree zero. Recalling that the the k-th derivative of a suitably smooth homogeneous function of degree d is itself a homogeneous function of degree $d - k$ and since $\gamma > 0$, if and only if the cost function is positive homogeneous of degree one will prices scale proportionately. ■

What would happen if the cost function were not homogeneous of degree one?

- If the cost function increased at a slower pace, so that

  $$C(\gamma \mathbf{q}) < \gamma C(\mathbf{q}),$$

  then the worst-case loss is unbounded. It is easy to see that in this case, the quantity

  $$\lim_{\gamma \to \infty} \max_i \gamma q_i - C(\gamma \mathbf{q})$$

  diverges, implying we have unbounded worst-case loss as the quantity vector grows large.

- If the cost function increased at a more rapid pace, so that

  $$C(\gamma \mathbf{q}) > \gamma C(\mathbf{q}),$$

  then prices would change with larger proportion at larger quantity vectors. So the price response at suitably large $\mathbf{q}$ would become very brittle, changing rapidly in proportion to input quantities. For large enough $\mathbf{q}$, prices would resemble an indicator function, so that the price of all but one index would be close to zero.

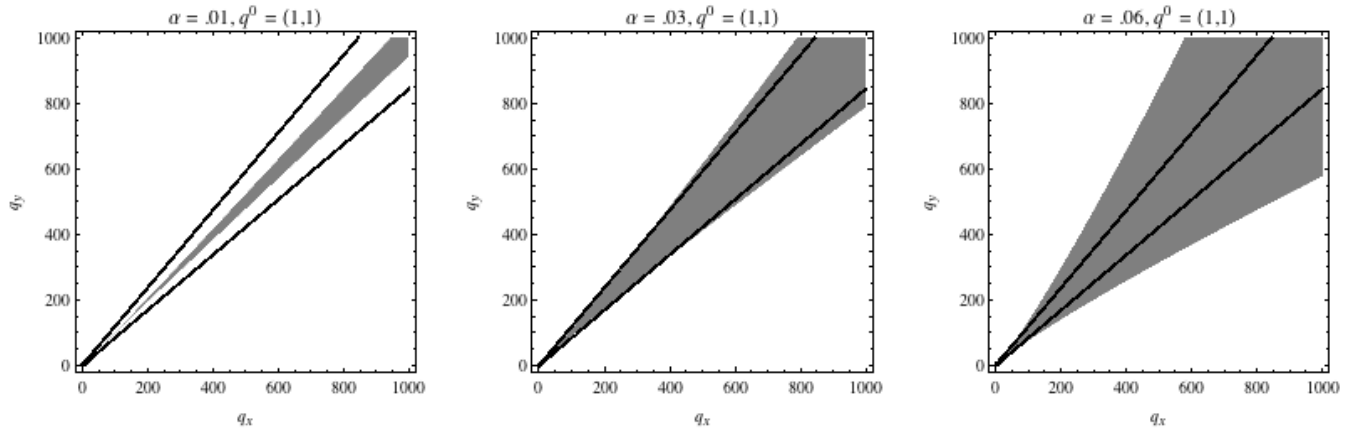

Figure 3: The shaded regions show where the market maker's worst-case revenue is greater than zero in a two-outcome market with initial quantity vector $(1, 1)$ and various values of α. The top black ray represents $p_y = .95$ and the bottom black ray represents $p_x = .95$.

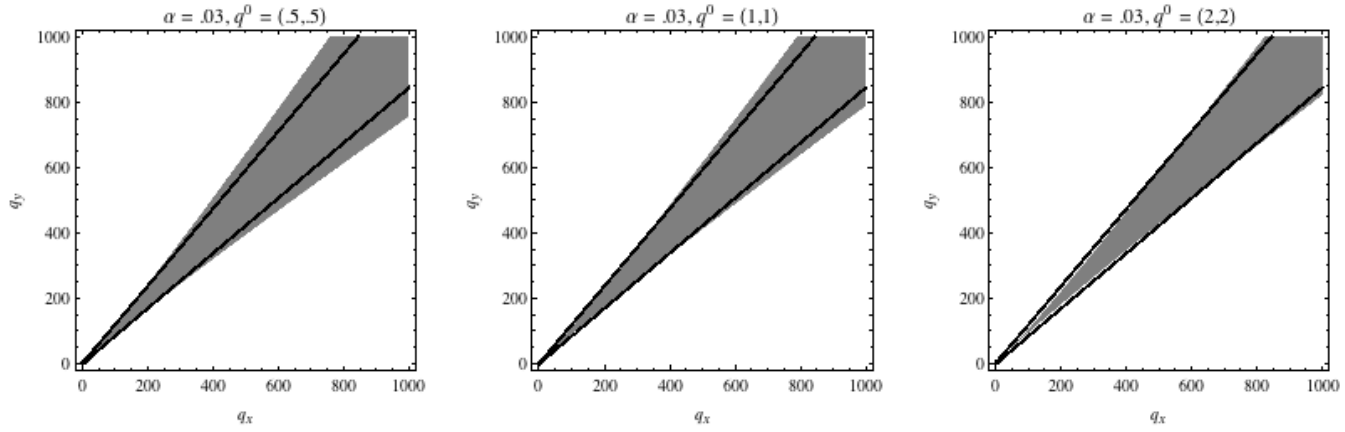

Figure 4: The shaded regions show where the market maker's worst-case revenue is greater than zero in a two-outcome market with $\alpha = .03$ and various initial quantity vectors. The top black ray represents $p_y = .95$ and the bottom black ray represents $p_x = .95$.

4. MAKING LARGE MARKETS

We have explored a new market maker that scales prices proportionally and has desirable revenue properties. However, inherent in our exposition is the idea that we can operate on a vector of size n. In some settings, however, this is not feasible. Consider, for instance, the annual NCAA men's basketball tournament held each March. This tournament features 63 games, so a complete state space accounting would have $n = 2^{63}$ (even if we only care, for each game, about the win versus loss rather than the score). This is a number far larger than can be reasonably operated upon.

In this section, we discuss how to deal with large liquidity-sensitive market makers. We show that the fast Gibbs-style sampling used by Chen et al. (2008) is not suitable for this setting, but we give a direct-sampling price-function-based method that can be used. We argue, however, that the best way to deal with large liquidity-sensitive markets is a cost-function-based method which we will introduce.

4.1 Price-based sampling

In price-based sampling, we use the price function to calculate the odds we offer traders. Since prices are given by the gradient of the cost function, if we can calculate price estimates accurately, then we can calculate the cost function, allowing us to calculate bets. Ideally, this would be done in closed form (Chen et al., 2008), but numerical quadrature techniques can also be used (Judd, 1998).

4.1.1 Gibbs-style sampling

The most compelling description of a very large-scale market maker is featured in the appendix of Chen et al. (2008). That approach uses sampling the Gibbs measure, a technique with roots in statistical mechanics, to estimate prices. We will briefly summarize that approach here. It uses the LMSR rule; recall that their b is a fixed constant.

Their sampling approach works as follows. Select a set of outcomes S from a generative prior $g : \Omega \mapsto \mathbb{R}^+$. For state s_i, initialize log-likelihood $ll_i = \log g(s_i)$. The market maker

then sets $q_i = b \cdot ll_i$. Weights are then calculated by

$$w_i = \exp(q_i/b - ll_i)$$

and we can use these weights to calculate instantaneous prices. Let a bet a come in which applies to the set of states $A \subset S$. We price the bet as

$$p_a = \frac{\sum_{j \in A} w_j}{\sum_i w_i}$$

When we change the quantities q_i, we update the set of weights.

We call this technique *price-based Gibbs-style sampling*.

Definition 13. A *price-based Gibbs-style sampling method* solves for prices based on indicator sampling over weights w on the set of states. Given a bet a that applies to sampled states $A \subseteq S$, we price the bet as

$$\mathbb{P}(a) = \frac{\sum_{j \in A} w_j}{\sum_i w_i}$$

One might hope that a price-based Gibbs-style sampling method would be effective in our setting as well. Unfortunately, it is straightforward to see that no Gibbs-style sampling method can return a price of greater than unity for any bet. Since our liquidity-sensitive market maker should have prices that sum to more than 1 (for instance, on a bet that encompasses the entire space of outcomes), price-based Gibbs-style methods are incapable of producing an engine to handle bets in our new liquidity-sensitive framework.

4.1.2 Direct price sampling

Instead, a market maker could use what we call *direct sampling*: directly calculating prices from the price function using sampled states. Recall that prices are given by

$$\sum_{s_i \in A} p_i(\mathbf{q}) = \sum_{s_i \in A} \alpha \log\left(\sum_j \exp(q_j/b(\mathbf{q}))\right) + \frac{\sum_j q_j \exp(q_i/b(\mathbf{q})) - \sum_j q_j \exp(q_j/b(\mathbf{q}))}{\sum_j q_j \sum_j \exp(q_j/b(\mathbf{q}))} \quad (1)$$

This rule is complex enough that it requires two phases to sample. Assume that we have drawn the states ω_i. Then we can estimate b as

$$\hat{b} = \alpha \sum_i q_i$$

With that value we can then estimate the price of the bet by evaluating Equation 1 with $b(\mathbf{q}) = \hat{b}$:

$$\sum_{s_i \in A} p_i(\mathbf{q}) = \sum_{s_i \in A} \alpha \log\left(\sum_j \exp(q_j/\hat{b})\right) + \frac{\sum_j q_j \exp(q_i/\hat{b}) - \sum_j q_j \exp(q_j/\hat{b})}{\sum_j q_j \sum_j \exp(q_j/\hat{b})}$$

Chen et al. (2008) were able to use price estimates in closed form by using a clever observation to estimate costs. We see no similar technique for this situation. Of course, if we can estimate instantaneous prices, we can also solve for costs using numerical techniques, but such techniques amplify the flaws of sampling at an individual state. That is, if estimating a single price is slow and inaccurate, then numerical integration techniques will also be slow and inaccurate. And since direct price sampling as described here is an unwieldy two-stage process, it is unlikely to be able to serve as the core of a robust market maker.

4.2 Cost-function-based sampling

To circumvent these problems, instead of using price-based sampling, we can use *cost-function-based sampling* to price bets within our market maker.

4.2.1 Initial setup

First, develop a generative prior $g : \Omega \mapsto \mathbb{R}^+$. This prior maps states to their prior probability of occurring. The better the prior is, the better the price predictions will be.

Generate a set of states $S \subset \Omega$ according to the prior g, where $|S| << n$ is as large as possible but still small enough to work with (e.g., $2^{20} \approx 1$ million states). For each state $s \in S$, fix its initial quantity, q_s, to

$$q_s = g(s)$$

4.2.2 Pricing bets

Let the current quantity vector over the states in S be $\mathbf{q}$, and let bet a arrive, which wagers c_A dollars and applies to states $A \subseteq S$. Denote by $\mathbb{1}^A$ the pointwise indicator vector for state A, i.e.,

$$\mathbb{1}_i^A = \begin{cases} 1 & \text{if } S_i \in A, \\ 0 & \text{otherwise.} \end{cases}$$

Then, solve for the reward r that satisfies

$$C\left(\mathbf{q} + r\mathbb{1}^A\right) = C(\mathbf{q}) + c_A$$

Offer the bet to the trader to win reward r for risking c_A.

Though the vectors in question are large, r is just a scalar and the problem is therefore analogous to finding the root of a well-behaved single-variable function. Consequently, it can be solved quickly in practice; test examples with $|S| = 1$ million solved in less than a second in MATLAB on a standard commodity PC.

4.3 Sampling concerns

The primary concern with sampling is that, because we are only covering a tiny fraction of the actual state space, it might be possible for a trader to arbitrage the market. Say the trader placed a bet that does not apply to any of the states in our sample. Then, if we were using a cost-function-based scheme, we would tell the trader they could purchase an infinite payout for nothing!

Two ideas protect us from this possibility:

- We modify the scheme so that we never offer a trader odds, on any bet, greater than some value (e.g., 10,000:1).
- Only offer bets that cover relatively large chunks of the state space. This is in order to improve the accuracy of bets by making it more likely that multiple sampled states will be relevant to the bet in question. For example, consider bets on the outcome of the NCAA Men's Basketball tournament. A bet on a particular outcome of the tournament (all 63 games at once) will either have no coverage by samples or relatively too much coverage (in case that a sample happens to

match that event exactly). In contrast, a bet on the result of a single game will likely have accurate coverage by samples.

5. CONCLUSIONS

Two of the main practical hurdles to more widespread use of Hanson's LMSR market maker are (1) the liquidity level b is set manually and never changes, and (2) the operator can expect to lose money in proportion to b. We present a new automated market maker design that overcomes both while retaining path independence, thus ensuring the market maker cannot be exploited and greatly simplifying the implementation. We prove that if we want sensitivity to liquidity and path independence, then we must relax the arbitrage condition that constrains prices of disjoint and exhaustive assets to sum to exactly one dollar. In our case, prices can sum to more than one, but this turns out to be a practical benefit, enabling the market maker to extract a profit if the entropy of final prices is sufficiently high. With LMSR, the market operator must ante a larger subsidy to obtain reasonable liquidity. With our liquidity-sensitive market maker, the subsidy can be set arbitrarily low without harming liquidity (except in the initial stage). We also show that for a broad range of terminal market states, our market maker actually makes a profit regardless of the event that gets realized. We discuss sampling-based methods to approximate prices in combinatorial settings with outcome spaces too large for exact computation, including a new cost-function-based method.

We plan to experiment with our liquidity-sensitive market maker with real traders to see how they react and how the market performs. Though the LMSR converts nicely to a liquidity-sensitive form, we find that the quadratic scoring rule (Pennock and Sami, 2007) does not; we'd like to say something more general about the class of market scoring rules. Some open questions include: how an epsilon-subsidy market behaves at its open before liquidity builds, how to incorporate prior information or learning, how to handle persistent limit orders, and how to mix sequential and batch order processing.

Acknowledgments

We thank Vincent Conitzer, Ivan Corwin, Sharad Goel, and Alex Grubb for helpful discussions and suggestions. Othman and Sandholm are supported by NSF grant IIS-0905390.

References

S. Agrawal, E. Delage, M. Peters, Z. Wang, and Y. Ye. A unified framework for dynamic pari-mutuel information market design. In *Proceedings of the 10th ACM Conference on Electronic Commerce (EC)*, pages 255–264, 2009.

J. Berg, R. Forsythe, F. Nelson, and T. Rietz. Results from a Dozen Years of Election Futures Markets Research. *Handbook of Experimental Economics Results*, 2001.

Y. Chen and D. Pennock. A utility framework for bounded-loss market makers. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 49–56, 2007.

Y. Chen, S. Goel, and D. M. Pennock. Pricing combinatorial markets for tournaments. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 305–314, 2008.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

S. Das. The effects of market-making on price dynamics. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 887–894, 2008.

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.

K. Judd. *Numerical Methods in Economics*. The MIT Press, 1998.

K. Oliven and T. Rietz. Suckers are born but markets are made: Individual rationality, arbitrage, and market efficiency on an electronic futures market. *Management Science*, 50(3):336–351, 2004.

M. Ostrovsky. Information aggregation in dynamic markets with strategic traders. In *Proceedings of the 10th 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 *Proceedings of the 11th ACM Conference on Electronic Commerce (EC)*, 2010a.

A. Othman and T. Sandholm. When do markets with simple agents fail? In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2010b.

D. Pennock and R. Sami. Computational aspects of prediction markets. In *Algorithmic Game Theory*, chapter 26, pages 651–674. Cambridge University Press, 2007.