# An Analysis of Profit Expectation and Prediction Accuracy for the Automated Market Maker when modifying the LMSR Model

Rangel Milushev, Gopal K. Vashishtha, Tomislav Zabcic-Matic

December 2018

**Abstract**

The LMSR automated market maker is one of the most popular designs for prediction markets in the literature. However, it runs a deficit, rendering this model impractical to implement with real currency. In this paper we have analyzed the profit expectation and prediction accuracy of the LMSR market maker that occurs when we relax the translation invariance property. We use a modification first introduced by Othman et al. (2013) for our simulations. While relaxing translation invariance does allow the market maker to be considerably more profitable than the original version, we lose the strict correspondence between prices and probabilities, rendering this model less accurate. We set out to investigate how large that trade-off actually is and whether the "profitable" LMSR is an adequate alternative.

## 1 Introduction.

Prediction markets are widely used to provide a platform where traders can speculate on the outcomes of various events: from election results to weather forecasting. Usually, a contract is issued for every possible outcome of the event. The most common type of contract, and the one we will be working with throughout the paper is a *winner-takes-all* contract[1]. The design idea is that traders of these contracts will equilibriate around a certain price that reflects the aggregate belief of the population for the various outcomes associated with a given event (Parkes and Seuken, 2019). An automated market maker (AMM) is an algorithmic system provided by the market operator that is always willing to trade. Rather than allowing for short-selling, AMMs allow for trades on each possible outcome. The most common AMM used in Internet prediction markets is Hanson's logarithmic market scoring rule (LMSR), an AMM with particularly desirable properties (Hanson 2003, 2007).

The LMSR has two major shortcomings, which deem it impractical and impair its implementation in an environment that uses real money. First, it is not *liquidity sensitive* in the sense that trades cause prices to move the same amount in both heavily and lightly traded markets, and second, it runs at a deficit (Othman et al., 2013; Hanson, 2007). Othman et al. (2013) propose a practical liquidity sensitive alternative to the widely accepted LMSR that is better suited for practical use in the following two ways. First, their modified LMSR market maker adjusts automatically by design how easily prices change according to market activity.

[1] In a winner-take-all contract there is some event that will or will not occur. If it occurs, the contract makes a specified payment to the holder of the contract, if it does not occur the holder of the contract receives nothing.

Second, their market maker arguably ensures "an arbitrarily small loss in the worst case and a positive profit over a wide range of final states". This all, however, comes at a loss of prediction accuracy. In our simulations, we test their model and provide an extensive analysis of how their modifications impact revenue, prediction accuracy and trader behaviour of the AMM.

# 2 Automated Market Maker.

Let us denote $O = (o_0, \ldots, o_{m-1})$ as the set of outcomes, with $m$ outcomes in total. Let $x \in \mathbb{R}^m_{\geq 0}$ be the market state, where $x_k \geq 0$ is the total quantity of contract k sold by the market maker. If the final outcome is $k$, the market maker pays 1 USD to all the $x_k$ contracts and nothing to all $x_i$ contracts for outcome $i$, where $i \neq k$ and $i \in m$. The initial market state is $x = (0; \ldots; 0)$. Trades are represented by a *quantity vector* $Z \in \mathbb{R}^m$. The market state then updates to

$$x := x + Z$$

The *position* for each trader shows how many units of a contract they own.

## 2.1 Cost-based Market Maker

A cost-based market maker defines a cost function, $C : \mathbb{R}^m_{\geq 0} \mapsto \mathbb{R}$, which is differentiable, and strictly increasing in $x_k$ for every state $x$, and every contract $k$ (Parkes and Seuken, 2019).

If we are currently in the market state $x$, then $C(x) - C(0) \geq 0$ gives us the total payments made by traders to the market. Using this, we denote $\pi(x, Z) = C(x + Z) - C(x)$ as the payment by a trader to the market maker for trade $Z$ in state $x$. Let us also denote $\pi_k(x)$ as the *instantaneous price* of contract $k$ at state $x$. This is the price for any trader to purchase a negligibly small fraction of contract $k$. This instantaneous price is then given by the partial derivative of the cost function:

$$\pi_k(x) = \frac{\partial}{\partial x_k} C(x)$$

A vector of the form $\pi(x) = (p_0, \ldots, p_{m-1})$ represents the current market belief at state $x$. The market belief shows the aggregate belief of the population of traders about the outcome of a certain event. Given the above defined market belief, we can construct the payment for a trade $Z$ as:

$$\pi(x, Z) = \int_{z=x}^{x+Z} \pi(z) dz$$

## 2.2 Desirable Properties

We want our AMM to have the following desirable properties (Othman et al, 2013; Hanson, 2003):

**P1:** *Path independence*[2]: Pricing rule $\pi$ is path independent if the value of line integral (cost) between any

[2]Path independence offers three important benefits. First, it ensures that there does not exist a money pump in the market. Second, it provides a minimum representation of state, because we only need to know the quantity vector. Third, traders will

two quantity vectors depends only on those quantity vectors, and not on how you get from one state to the other (the path between them).

**P2:** *Translation Invariance or Normalization*: A cost function is translation invariant if prices always sum to unity. Formally:

$$\sum_{k=0}^{m-1} \pi_k(x) = 1$$

for every valid market state $x$, so that the instantaneous prices always sum to one.

**P3:** *Liquidity Sensitivity*[3] : Given the $m$-dimentional unit vector $v = (1; \ldots; 1)$. A pricing rule is *liquidity insensitive* if:

$$\pi_i(x + kv) = \pi_i(x)$$

for all valid $x$ and all $k$. We want our pricing rule to be liquidity sensitive.

# 3 The LMSR Market Model.

One popular AMM that satisfies *path independence and translation invariance*, but not *liquidity sensitivity* (Othman et al., 2013) is the *logarithmic market scoring rule* market model first introduced by Hanson (2003). The logarithmic market scoring rule is strictly proper (Hanson, 2003; Parkes and Seuken, 2019). The LMSR market maker is a cost-based automated market maker with a cost function:

$$C(x) = \beta \ln \left( \sum_{k=0}^{m-1} e^{x_k/\beta} \right)$$

where $\beta > 0$ is a constant adjusted for liquidity.

The instantaneous price on contract $k$ in the LMSR market maker (taking partial derivatives) is

$$\pi_k(x) = \frac{e^{x_k/\beta}}{\sum_{j=0}^{m-1} e^{x_j/\beta}}$$

Using an approximation we can easily see that for a sufficiently large $\beta$ the price remains close to $1/m$ ($m$ being the number of possible outcomes) for all market states, whereas for negligibly small values it converges to 1 (Othman et al., 2013). For the LMSR the loss is bounded and the maximum expected payment is ($\beta$ times) the entropy (Hanson, 2003).

not need to strategize about placing bets: making a series of small trades gives the same odds as a single large trade (Othman et al., 2013).

[3] Unlike the desired adjustable liquidity property defined by Parkes and Seuken(2019) that requires a constant $\beta$ that adjusts the amount of liquidity in the market, the *liquidity sensitivity* property is about adapting the liquidity in the market dynamically from one state to the next, such that when the market is deep, price move less from a fixed-size bet than when the market is shallow

# 4 Modifying LMSR.

As proven by Othman et al. (2013) it is not possible to satisfy all three desirable properties within the same model. Therefore, in order to account for the shortcomings of the automated market model presented in the previous section, we allow for the relaxation of the *translation invariance* property, which leads to satisfying *liquidity sensitivity*.

## 4.1 Relaxing Translation Invariance

Othman et al. (2013) propose a modification of the cost function of the above-defined LMSR model that results in adjusting the sum of the price of contracts to be greater than or equal to 1. he normalization is modified to the bounded inequality:

$$1 \leq \sum_{k} \pi_k(x) \leq 1 + \alpha m \ln m$$

in which $m$ is the number of possible outcomes, equivalent to the number of different contracts and $\alpha > 0$ is a constant. This shows that due to the bound, prices would not rise above a specific maximum.

## 4.2 Cost function

The main weakness of the LMSR cost function (*Definition 4*) is the low liquidity adaptability: prices change by the same amount irrespective of the already purchased contracts amount or trader behaviour. To solve this problem we use a method proposed by Othman et al. (2013), in which the constant $\beta$ becomes a function of the market state $x$:

$$\beta(x) = \alpha \sum_{k=0}^{m-1} x_k$$

where $\alpha$ is a suitably chosen constant. A key property of this modification is that $\beta$ is now a function that increases with market volume. The cost function is modified to:

$$C(x) = \beta(x) \ln \left( \sum_{k=0}^{m-1} e^{x_k/\beta(x)} \right)$$

It is important to note here that this cost function still holds desirable properties like bounded loss and reduction of the price of contracts when too many are outstanding (Othman et al., 2013).

## 4.3 Instantaneous Prices and Market Belief

With the above defined $\beta(x)$, the instantaneous pricing function becomes more complex, but can be expressed by (Othman et al, 2013)[4]:

$$\pi_k(x) = \alpha \log \left( \sum_{i=0}^{m-1} e^{x_i/\beta(x)} \right) + \frac{\sum_{i=0}^{m-1} x_i e^{x_k/\beta(x)} - \sum_{i=0}^{m-1} x_i e^{x_i/\beta(x)}}{\sum_{i=0}^{m-1} x_i \sum_{i=0}^{m-1} e^{x_i/\beta(x)}}$$

[4]Othman et al.(2013) also prove that since the entropy operator is bounded below by zero, the sum of prices is at least 1.

As the number of contracts for the complementary event increases, the markets price response for a fixed-size investment becomes less pronounced as more liquidity enters the market. The price of a fixed-size contract is shown to shrink as the level of outstanding quantities increase (Othman et al, 2013). It is important to note here that since we are breaking *translation invariance* the instantaneous price is no longer representative of the market belief.

In the case of binary outcomes, as in our simulation, the state vector is $(x_0, x_1)$. The probability of event $o_0$ under normal LMSR can be represented as $\pi_0(x)$, or equivalently $1 - \pi_1(x)$. In the new market maker, we instead derive a range of possible market probabilities: $[\min(\pi_0(x), 1 - \pi_1(x)), \max(\pi_0(x), 1 - \pi_1(x))]$

### 4.4 Choice of constant $\alpha$

In the case of the modified LMSR, $\alpha$ can be thought of the commission the market maker takes (Othman et al, 2013). The cost function is non-decreasing in $\alpha$. Other market makers tend to operate between 2% and 20% of commission. While we will be testing for various values of $\alpha$ in our simulations, one can think of setting alpha to emulate a certain commission that does not exceed a value $v$ as:

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

Again, $m$ being the number of possible outcomes. We will be using binary outcomes, so this becomes:

$$\alpha = \frac{v}{2 \log 2}, \quad v \in [0.02, 0.2], \quad \alpha \in [0.01, 0.14]$$

## 5 Simulation.

We have built a simulator to study how pronounced the difference between revenue and prediction accuracy is between the two different market models. To do so we have implemented three distinct agent designs. For the full code we were using, please see https://github.com/gvashishtha/cs136_project

### 5.1 Simulation Design

We assume a *winner-takes-all* prediction market on two outcomes, $o_0$ and $o_1$ such that at the conclusion of the market one of either $o_0$ or $o_1$ must occur, and only one of them occurs. The state of the market is represented as a vector $x$, such that $x_0$ represents shares sold on outcome $o_0$ and $x_1$ represents shares sold on $o_1$. At the conclusion of the market, each share of the outcome that occurs pays out \$1. We are implementing a *no selling* scheme[5] described by Othman et al. (2013).

A single trial of our simulated market runs as follows:

1. The user specifies an initial state of the market, $x$. It is important that $\sum_i x_i \neq 0$, because in the modified LMSR, we calculate $\beta$ as $\alpha \sum_{i=0}^{m-1} x_i$.

[5] Agents always buy shares on outcomes from the market maker. With this scheme when a trader imposes an obligation and then sells it back to the market maker, the trader ends up with a net loss so our agents only purchase but do not sell contracts.

2. The user specifies an underlying beta distribution representing the state of the world. Beta distributions can be fully specified by two parameters, $\alpha$ and $\beta$, and their outcome lies on the range $[0, 1]$

3. From this true underlying beta distribution, draw a value, $p_{true}$, representing the true probability the event will occur. As discussed in (Othman et al, 2013), "a market with capital-constrained traders where the true state of the world fluctuates frequently" may not be the best place to use the Othman Market Maker, so we opted to keep the true probability of occurrence constant throughout the simulation.

4. Bidding proceeds in rounds. Each round, each agent gets to bid once, and each round, we permute the order in which agents can bid. At each bidding step, the world emits a $\{0, 1\}$ signal, $s$ according to a Bernoulli distribution with parameter $p_{true}$.

5. Given $s$, all agents get a chance to update their prior distribution over possible outcomes. With a certain user-specified noise probability, $n \in [0, 1]$, drawn independently for each agent, agents will receive the opposite signal from the true signal. Through this noise parameter, we can model the sophistication of the population of betting agents.

6. After updating its prior, the agent whose turn it is to bid then submits a vector, $t$, representing a trade they would like to make with the market maker. If they have sufficient capital, they pay $C(t+x)-C(x)$ to the market maker where $x$ is the current state of the market and $C$ is the cost function. As discussed in section 3.3 of (Othman et al, 2013), it is important to not allow traders to sell to the market maker, so traders are only allowed to either buy shares of $o_0$ or buy shares of $o_1$.

7. At the end of the user-specified number of rounds, the outcome $\in \{0, 1\}$ is drawn from a Bernoulli distribution with parameter $p_{true}$. Traders are compensated accordingly and the profit to the market maker is calculated.

## 5.2 Agent Design

We created three kinds of agents to participate in our market. All agents we designed are myopic, budget-constrained, and totally risk-averse, meaning that if they have a private belief that outcome $o_0$ will occur with probability $b$, and $p_0(x) < b$ they are willing to bet their entire endowment on buying $x_0$ until $p_0(x) \geq b$.

The strategy of updating agent priors based on a beta distribution has previously been proposed in (Othman, 2008). More details on our agents are below:

| Agent Name | Prior Distribution | Bidding Strategy |
|---|---|---|
| Zero Intelligence (Othman, 2008) | Constant $p_{private}$, drawn from true underlying $\beta$ distribution | Buy either $x_0$ or $x_1$ with equal probability, buy if instantaneous price is less than belief in that outcome |
| BuyOne | Maintain a private beta distribution, update it every time we receive a signal from the market. Parameters for initial prior are chosen randomly to simulate different initial beliefs | Buy one share on one outcome, as long as the share is priced below the agent's belief in that outcome's likelihood of occurrence. |
| Truthful | Maintain a private beta distribution as with BuyOne | Buy shares of outcome $o_i$ where $p_i(x) < b_i$ ($b_i$ is the agent's estimate of probability of $o_i$) until the instantaneous price of that outcome matches our belief in that outcome.[6] |

# 6 Simulation Analysis.

We perform our tests mainly on the modified LMSR market maker, which is path independent, adaptive to increased liquidity, and can arrive at situations in which it makes money regardless of realized outcome.

## 6.1 Profitability of LMSR and modified LMSR

In 1 below, we can see the difference in profitability between the regular LMSR and the modified version introduced by Othman et al. (2013). Running the simulations with 50 rounds and 50 trials for averaging purposes, we attain the following graph of LMSR vs modified LMSR profits for various agent populations.

[6]Because instantaneous price does not equate to market belief in the Othman Market Maker, we only use Truthful agents with the Hanson Market Maker. Given the agent is trying to purchase some quantity $q_i$, in which it has belief $b$, it buys a quantity equal to $\beta * (\ln(b) + x_{1-i}) - \ln(1-b) - q_i$

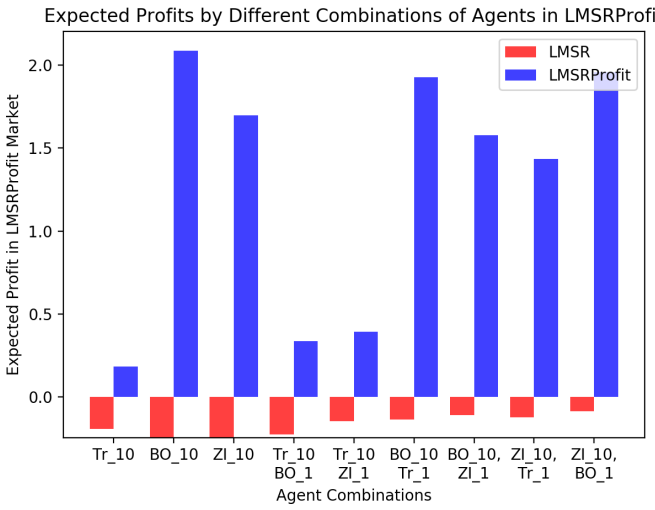


Figure 1: blue: (a) modified LMSR profit, red: (b) regular LMSR profit

As we can observe in 1, the modified version of the LMSR is consistently more profitable than the standard. In the sections below, we inspect how this difference impacts prediction accuracy, and what the trade-offs are if we are considering this model for a real-world implementation.

## 6.2 Tightness of Prediction Approximation

With a mixed population of 6 budget-constrained traders (3 Zero Intelligence and 3 BuyOne agents), the market probability converges to within a 0.04 level of accuracy to the true probability within 5 rounds. We can see this in 2(a) below:

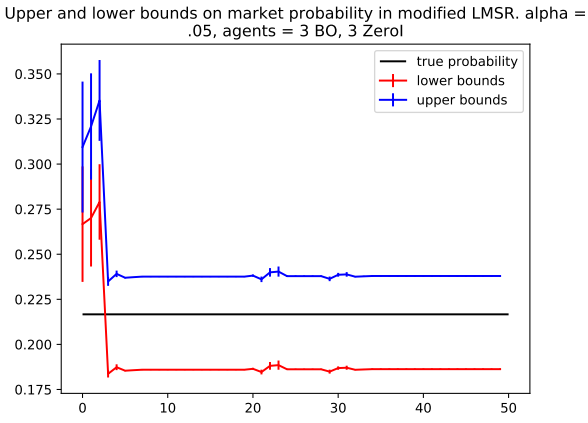


Figure 2: **Prediction accuracy with alpha 0.05**(a) left

In 3, we notice that as $\alpha$ increases, regardless of the population of participating agents, the market's ability to estimate the true probability of an event's occurring also decreases:

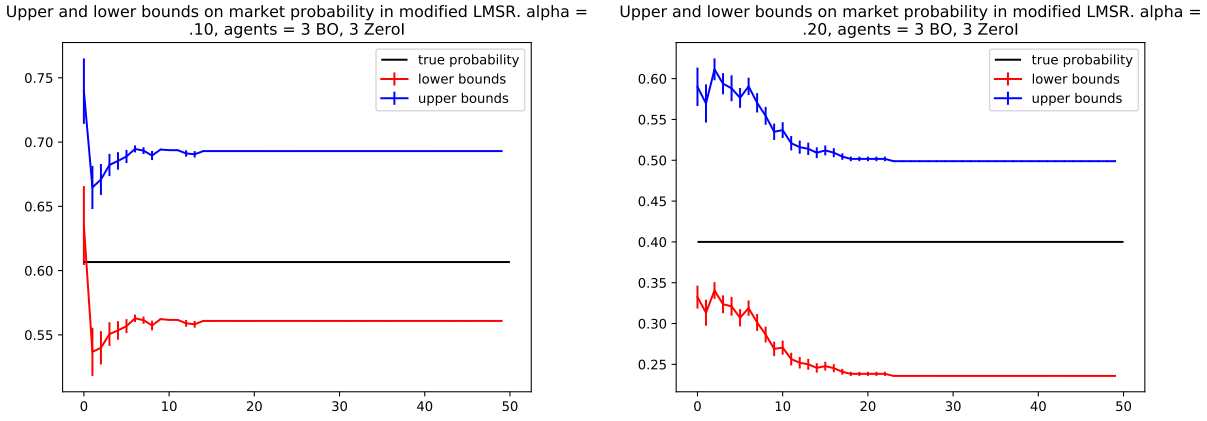


Figure 3: **Prediction accuracy under increasing alpha**(a) left, (b) right

Interestingly, if we consider 4(a-c) we note that with $\alpha = 0.01$, the market's ability to approximate the true probability accurately decreases, suggesting that ZeroI traders (who never update their prior beliefs based on observed signals) begin to have an outsize impact at low levels of $\alpha$ because trading with the market is so cheap. When we remove the Zero Intelligence traders and give the BuyOne traders sufficiently high initial allocations, the market once again returns to closely approximating the true probability, even with low $\alpha$ 4(b).

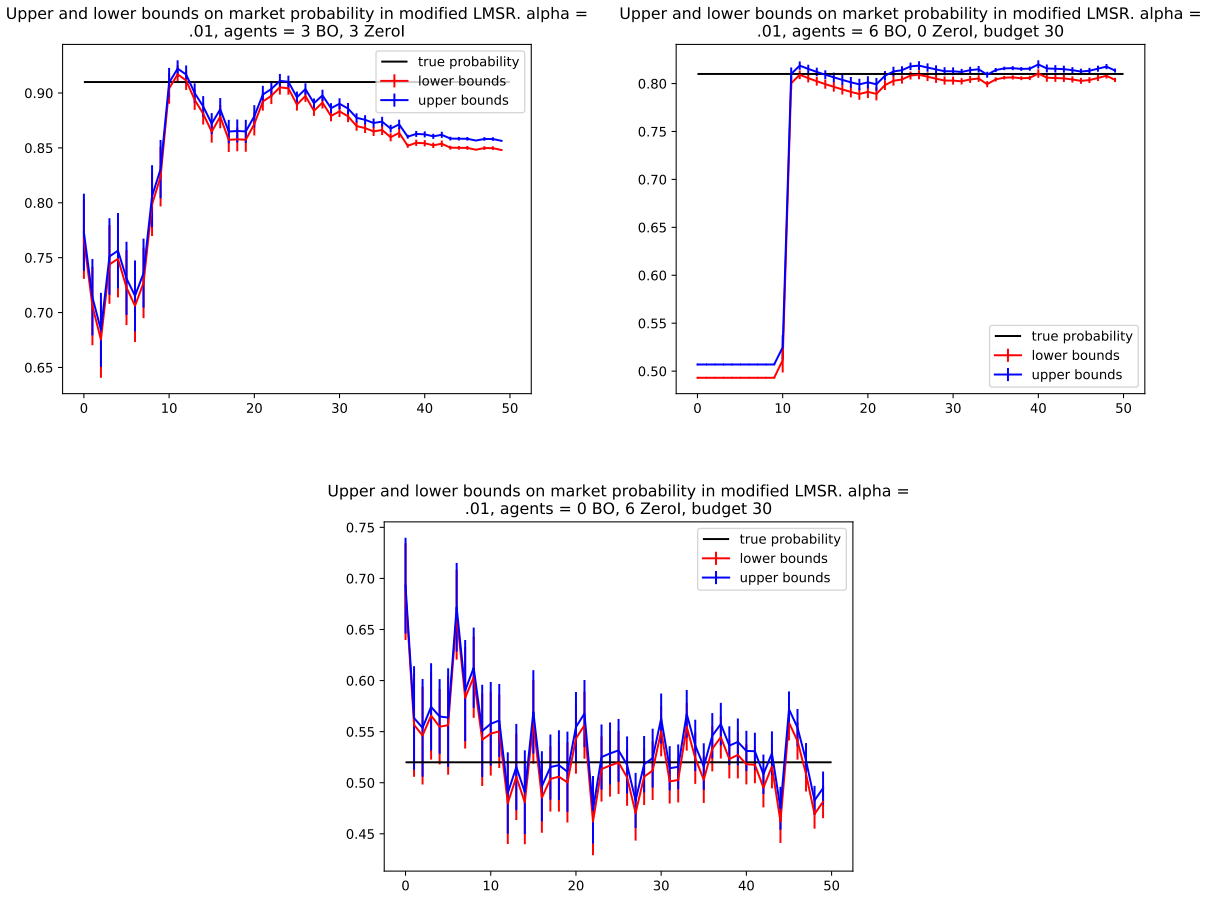


Figure 4: **Prediction accuracy under low alpha**(a) left, (b) right, (c) low

If we instead remove all BuyOne traders and include only Zero Intelligence traders under low $\alpha$ and high budget, market prices remain tightly bounded, but fluctuate much more than they do under the BuyOne condition 4(b).

## 6.3 Profits and alpha

In investigating the relationship between profits and $\alpha$, we notice that profits seem to peak around $\alpha = 0.07$ before falling off steeply and finally declining to 0, when trading with the market becomes prohibitively expensive due to high commissions. Using the equation from section 4.4, we see that $\alpha = 0.07$ roughly corresponds to a commission of $2\log(2) * .07 = .10$, or 10%. It is not immediately clear why there should be such a steep decline in profits at exactly this level of commission, so this is something worth investigating further.

Interestingly, if we restrict the market to only Zero Intelligence agents, the "dropoff" becomes a more linear, gradual decline in profitability. This would seem reasonable because Zero Intelligence agents do not update their priors and may imagine themselves to be getting a good deal even for higher levels of $\alpha$. By

contrast, if we run a market of only BuyOne traders, then the "dropoff" remains, and we see a sharp initial jump in profits followed steadily decreasing profits with increasing $\alpha$. For a visual representation please refer to 5 below.

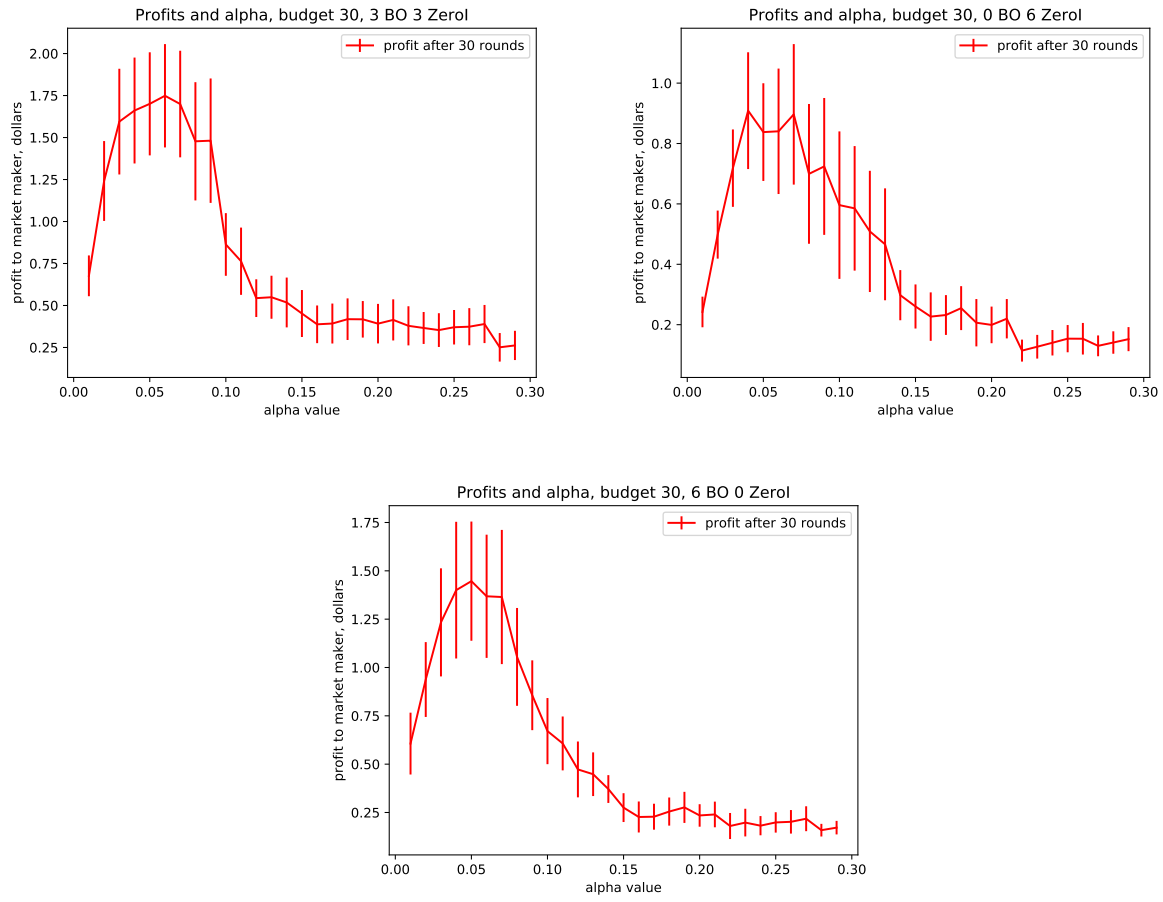


Figure 5: **Profit vs Alpha**. Top-left is mixed population between BO and ZI Agents; top-right is population of only ZI agents; bottom is population of BO agents

Reassuringly, the shape of the profit curve remains fairly constant even as the noise parameter increases, indicating that the choice of $\alpha$ is fairly robust to the level of agent sophistication. We can observe this phenomenon on 6 below. Only at a noise value of 0.60 do we see significant outward shift in the "dropoff" point of profitability. Unexpectedly, as noise increases and traders become less sophisticated, the profit to the market maker tends to increase only up until a certain point. At a noise level of 0.4, when 4 in 10 signals is misinterpreted by market participants, the profit to the market maker actually falls.

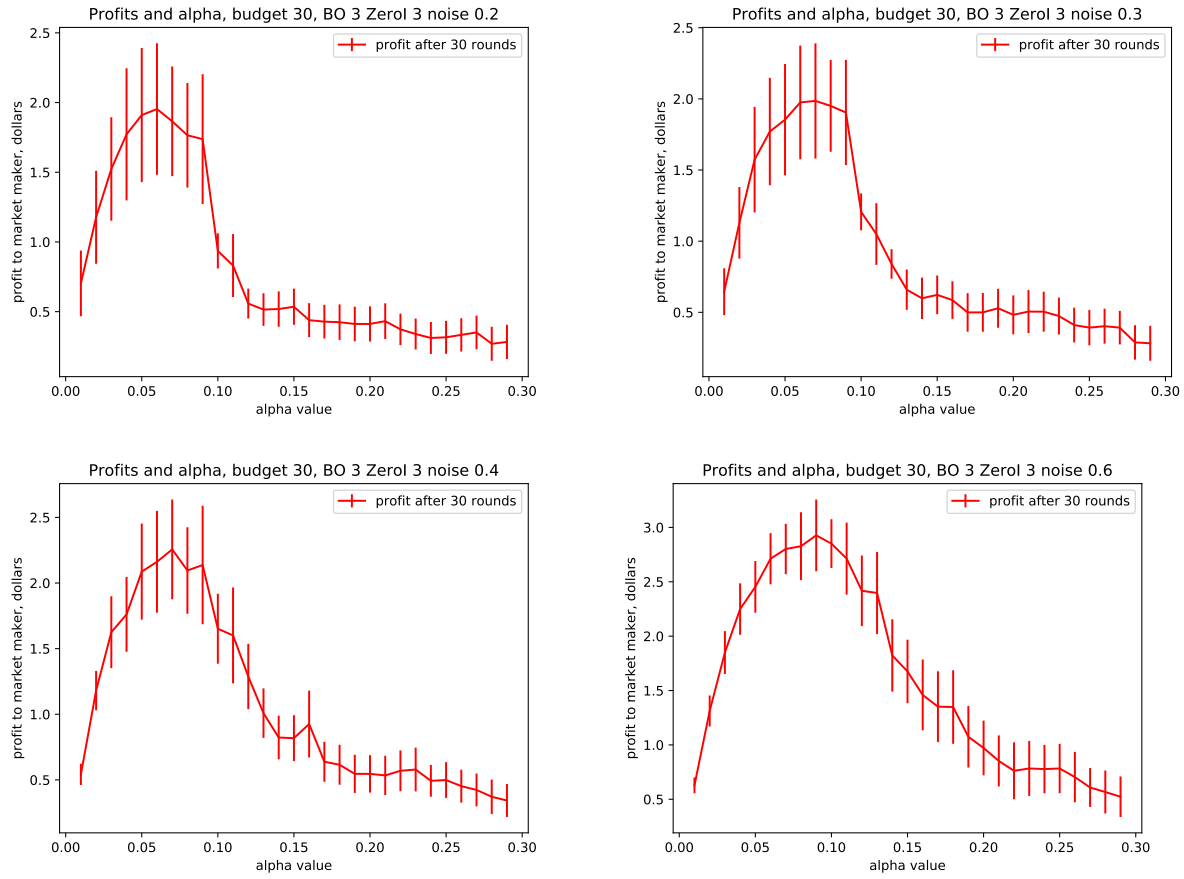


Figure 6: **Profit vs Noise**. testing for various levels of noise in a mixed population of ZI and BO agents

## 6.4 Profits and Initial Conditions

We also want to investigate the profits from the standpoint of analyzing initial conditions. In 7, we varied the initial conditions from the vector (1,1) to the vector (29, 29), and noted that for all combinations of agents, we see the following trend, on average - as the initial conditions increase, the profits generated by the markets tend towards zero, with a more noticeable shift for the modified LSMR market, which seems to be a result of the fact that its profits, in absolute value, are considerably greater than those for the regular LSMR market. Additionally, as noted in (Othman, 2013), prices in the modified LMSR are fairly sensitive to initial conditions, and these results confirm that.

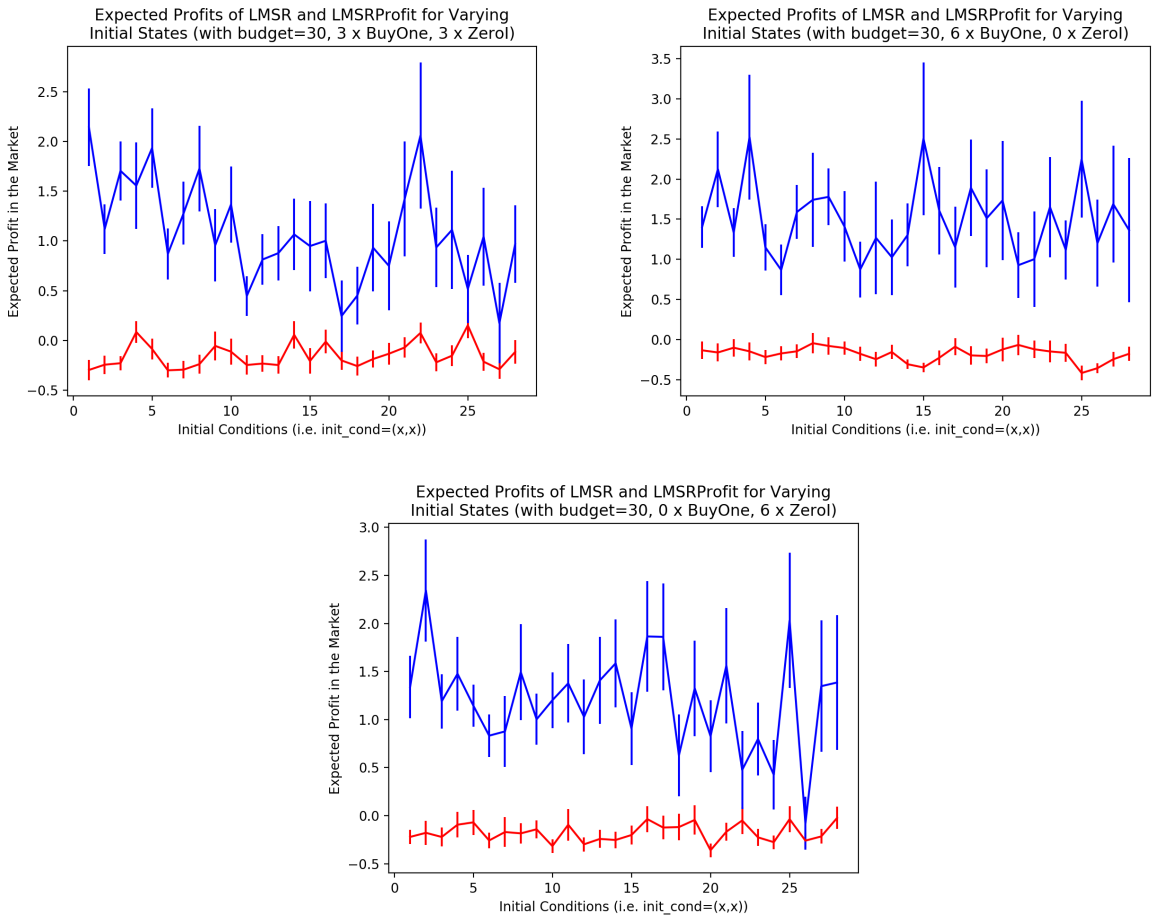


Figure 7: **Profit vs initial conditions**. Testing for impact of initial conditions on profits

# 7 Results and Conclusions.

In the analysis above we can see that in our simulations the modified LMSR does indeed perform better when it comes to revenue to the market maker, so it is more practical to use in real-money implementations. We observed that for $\alpha \in [0.04, 0.07]$ the AMM seemed to give the best performance with regards to profitability. Depending on the alpha coefficient, the modified LMSR is more or less accurate when it comes to equilibriating around the real probability, but it does provide us with a range of probability estimates.The lower the value of alpha, the closer we got to the true probability. One interesting observation here is that for $\alpha = 0.01$ the approximation accuracy decreases when we run it with Zero Intelligence agents present in the population. We attribute that to these agents never updating their prior beliefs based on the observed signals, so they purchase a contract for an outcome if the instantaneous price is less than their belief in that outcome.

With this liquidity-sensitive modified LMSR market maker, the subsidy can be set arbitrarily low without harming liquidity (except in the initial stages of the trading). Depending on the level of sophistication of

the traders and the way they incorporate prior information, these results may vary.

Given that we are implementing the *no selling* scheme described by Othman et al. (2013), our current implementation would not allow agents to sell for profit back to the market maker, which would not make for a user-friendly implementation in the real world. They also propose a different scheme: *covered short selling* which might be a better fit because it will not penalize real-world users for mistaken bets that they cancel right away. Both schemes produce the same cost function but we chose *no selling* because it was more intuitive to implement in our simulator.

# 8 Discussion and Future Work.

We presented our findings about how relaxing translation invariance from the original LMSR model, and satisfying liquidity sensitivity provides us with a more profitable model that is more suitable to be implemented in real-money environments. This comes at the cost of losing on prediction accuracy. Of course, given the limited scope of our work, there are many more improvements to be made on our simulator.

Othman et al. (2013) claim that for "a broad range of terminal market states our market maker actually makes a profit regardless of the event that gets realized". Since we designed our simulator to work specifically with binary outcomes, it would be interesting to see what the results would show for three or more outcomes. One assumption that we made is that agents would keep buying until the price reaches their true belief, and we are not accounting for that fact in the modified LMSR market. When we lose translation invariance, we lose the direct mapping from prices to probabilities that the original LMSR provides us with. Instead, we are left with a range of possible probability estimates. One future improvement on the accuracy of these simulations could be implementing the model so that agents will only buy up to the above-mentioned probability range. One could also imagine more sophisticated trading strategies than our current model that could be developed to show a more accurate representation of trading behaviour in the real world. There is still much room for improvement to study this modified LMSR market maker, developing models of how to better incorporate prior information or learning into the way the AMM prices contracts.

# 9 Acknowledgements

We would like to acknowledge Dr. David Parkes for the great experience that we had in CS136 and all the guidance that he provided us with when building our simulator. We would also like to acknowledge Dr. Abraham Othman and Dr. David Pennock for answering our questions and helping us clarify some of our ideas.

# 10 References.


1. Robin Hanson, 2003. Combinatorial information market design. Information Systems Frontiers, 5(1), pp.107-119.

2. Robin Hanson, 2007. Logarithmic Market Scoring Rules for Modular Combinatorial Information Aggregation. Journal of Prediction Markets, University of Buckingham Press, vol. 1(1), pages 3-15, February.

3. Abraham Othman, 2008. Zero-Intelligence Agents in Prediction Markets. *Proc. of 7th Int. Conf. on Autonomous Agents and Mul- tiagent Systems (AAMAS 2008)*, Padgham, Parkes, Muller and Parsons(eds.),May,12-16.,2008,Estoril,Portugal,pp. 879-886.

4. Abraham Othman, Tuomas Sandholm, David M. Pennock, and Daniel M. Reeves, 2010. A practical liquidity-sensitive automated market maker. In Proceedings of the 11th ACM conference on Electronic commerce (EC '10). ACM, New York, NY, USA, 377-386.

5. David C. Parkes and Sven Seuken, 2019. Economics and Computation. Cambridge University Press 2019.