Gnosis LMSR Automated Market Maker

Yasin Seifi

October 27, 2024

Gnosis Formula for LMSR

Presented below is the fundamental Gnosis formula for the LMSR Automated Market Maker (AMM), where we set the invariant to 1:

$$invariant = 2^{-\frac{Yes\_shares}{funding}} + 2^{-\frac{No\_shares}{funding}}$$

In this formula, "funding" denotes the liquidity, and "Yes_shares" and "No_shares" represent the quantities of each type of share in the liquidity pool. This formula constitutes the cornerstone of our analysis in this document, serving as the foundation for all derived functions.

How Do Pricing, Buying, and Selling Functions Work?

The logic underpinning these functions is straightforward.

Pricing

The pricing of Yes and No shares is determined as follows:

$$Yes\_price = 2^{-\frac{Yes\_shares}{funding}}$$

$$No\_price = 2^{-\frac{No\_shares}{funding}}$$

It is evident that the sum of these prices always equals 1.

Buying

The buying function accepts two inputs:

1. *Money for trading*

2. *Desired share type for purchase*

For instance, let the amount of money be M_{tr}, and suppose the trader opts to buy Yes shares. If "Yes_shares" and "No_shares" are the quantities of each share in the pool, the new share counts after the transaction would be $Yes\_shares + M_{tr}$ for Yes shares and $No\_shares + M_{tr}$ for No shares. This results from minting M_{tr} units of money into an equivalent amount of Yes and No shares added to the pool. Consequently, the trader receives x units of Yes shares, calculated as:

$$1 = 2^{-\frac{Yes\_shares+M_{tr}-x}{funding}} + 2^{-\frac{No\_shares+M_{tr}}{funding}}$$

Thus, x is given by:

$$x = \frac{\ln\left(1 - 2^{-(No\_shares+M_{tr})/funding}\right)}{\ln(2)} \times funding + Yes\_shares + M_{tr}$$

Selling

The selling function operates similarly to the buying function and also requires two inputs:

1. *Number of shares to be sold*
2. *Desired share type for sale*

For example, consider that a trader wishes to sell a quantity N_{Yes} of Yes shares. Let y represent the monetary value owed to the trader. Initially, we have:

$$1 = 2^{-\frac{Yes\_shares}{funding}} + 2^{-\frac{No\_shares}{funding}}$$

After compensating the trader, the new condition is:

$$1 = 2^{-\frac{Yes\_shares+N_{Yes}-y}{funding}} + 2^{-\frac{No\_shares-y}{funding}}$$

After simplification, the expression for y is:

$$y = -\frac{\ln\left(2^{-\frac{Yes\_shares+N_{Yes}}{funding}} + 2^{-\frac{No\_shares}{funding}}\right)}{\ln(2)} \times funding$$

Adding Liquidity, Initial Seeding, Remove Liquidity

Here, the funding variable is changed unlike the previous functions. also we explain how to handle the cases which market creator wants a non $.5 - .5$(in most of times we don't have enough data to approximate the probability of the outcomes so in this cases they adjust the initial pool to have equal number of each share. but in some cases we have enough data to approximate the probabilities. so in this case we may need a non $.5 - .5$ pricing). we explain this situations in Initial seeding function.

Adding Liquidity

In this section, we elucidate the liquidity addition function through an illustrative example. Assume that Yes_shares *and* No_shares *represent the quantities of each type of share in the liquidity pool, and let the total volume of liquidity be denoted by* $funding$*. Consider a scenario where a liquidity provider wishes to contribute* M_{LP} *dollars to the pool. This contribution is divided into two segments: one portion for augmenting the liquidity volume and another for purchasing shares of the higher-priced type to maintain stable pricing. Without loss of generality, suppose* $Yes\_shares < No\_shares$*. The relationship between the money allocated for share purchases (*M_{buy}*) and the addition to liquidity (*$M_{funding}$*) is given by:*

$$M_{LP} = M_{buy} + M_{funding}$$

where M_{buy} *is the amount used for purchasing shares, and* $M_{funding}$ *is the amount added to the liquidity. Additionally, let* N_{buy} *represent the number of shares purchased on behalf of the liquidity provider to keep the prices constant. The state of the pool after these transactions is represented by the equation:*

$$1 = 2^{-\frac{Yes\_shares + M_{LP} - N_{buy}}{funding + M_{funding}}} + 2^{-\frac{No\_shares + M_{LP}}{funding + M_{funding}}}$$

It is noted that the prices remain unchanged, hence the price of No-shares must be equivalent post-transaction:

$$2^{-\frac{No\_shares}{funding}} = 2^{-\frac{No\_shares + M_{LP}}{funding + M_{funding}}}$$

From this equality, $M_{funding}$ *can be calculated as follows:*

$$M_{funding} = \frac{funding \cdot M_{LP}}{No\_shares}$$

Moreover, assuming the Yes_price remains constant:

$$2^{-\frac{Yes\_shares}{funding}} = 2^{-\frac{Yes\_shares + M_{LP} - N_{buy}}{funding + M_{funding}}}$$

This leads to the following determination for N_{buy}*:*

$$N_{buy} = \frac{(No\_shares - Yes\_shares) \cdot M_{LP}}{No\_shares}$$

Thus, by contributing M_{LP} *to the pool, the Liquidity Provider acquires* $M_{funding}$ *in LP tokens and* N_{buy} *shares of the more likely outcome.*

Initial Seeding

This scenario parallels the addition of liquidity, except that initially there are no shares in the pool, and the funding is zero. Here, the market creator injects M_{Mc} dollars into the market as initial seeding. If the intended initial pricing probability for Yes-shares is greater than 0.5, designated by p, where $p > 0.5$, the equations adapt accordingly:

$$No\_price = 1 - p = 2^{-\frac{No\_shares}{funding}} = 2^{-\frac{M_{Mc}}{M_{seeding}}}$$

This results in:

$$M_{seeding} = -\frac{M_{Mc} \cdot \ln(2)}{\ln(1-p)}$$

Given that:

$$Yes\_price = p = 2^{-\frac{No\_shares}{funding}} = 2^{-\frac{M_{Mc}-N_{buy}}{M_{seeding}}}$$

We derive:

$$N_{buy} = M_{Mc} + M_{seeding} \cdot \frac{\ln(Yes\_price)}{\ln(2)}$$

Thus, upon adding M_{Mc} to the liquidity pool as initial seeding, the market creator receives $M_{seeding}$ in LP tokens and N_{buy} Yes-shares. The pool's new state is defined by $funding = M_{seeding}$, $Yes\_shares = M_{Mc} - N_{buy}$, and $No\_shares = M_{Mc}$.

Remove Liquidity

Consider a liquidity pool with previously defined states. Suppose a Liquidity Provider decides to withdraw a portion of his LP tokens. The process mirrors that of adding liquidity. Let LP_{remove} denote the tokens he intends to remove, which are divided into two parts:

$$LP_{remove} = M_{paid} + M_{buy}$$

Maintaining constant prices during the removal necessitates the following conditions:

$$1 = 2^{-\frac{Yes\_shares-M_{paid}+M_{buy}}{funding-M_{paid}}} + 2^{-\frac{No\_shares-M_{paid}-N_{buy}}{funding-M_{paid}}}$$

From this hypothesis, we affirm the constancy of prices:

$$Yes\_price = 2^{-\frac{Yes\_shares}{funding}} = 2^{-\frac{Yes\_shares-M_{paid}+M_{buy}}{funding-M_{paid}}}$$

$$No\_price = 2^{-\frac{No\_shares}{funding}} = 2^{-\frac{No\_shares-M_{paid}-N_{buy}}{funding-M_{paid}}}$$

Thus, we have three equations and three unknowns, leading to:

$$M_{paid} = \frac{funding \cdot LP_{remove}}{2 \cdot funding - Yes\_shares}$$

And:

$$N_{buy} = (No\_share - funding) \cdot \frac{M_{paid}}{funding}$$

Therefore, a Liquidity Provider who removes LP_{remove} tokens receives M_{paid} dollars and N_{buy} shares of No-shares.