The Conceptual Flaws of Constant Product Automated Market Making

Andreas Park[*]

August 15, 2021

Abstract

Blockchain-based decentralized exchanges (DEXes) are a pre-requisite for the nascent decentralized finance ecosystem. The most successful form of DEX in terms of trading volume are swap exchanges, a smart contract which pools liquidity and prices transactions with a deterministic function. Almost all swap exchanges use a pricing rule that is conceptually problematic as it gives rise to intrinsically profitable front-running opportunities as well as cross-DEX arbitrage. In practice, these shortcomings cause significant network congestion, a negative externality that raises costs for all users and that threatens the long-term viability of the entire blockchain ecosystem. Calibrated to a low liquidity but frequently used trading pair on UniSwap, the largest DEX for much of 2020/21, about 14% of transactions see an implicit theoretical excess cost of at least 50bps, which is orders of magnitude larger than the common trading costs for this pair on centralized exchanges. I further present an alternative pricing rule, based on insights from the market microstructure literature, that does not suffer from these flaws.

[*]University of Toronto `andreas.park@rotman.utoronto.ca`. I am grateful for financial support from the *Global Risk Institute* and the *Social Sciences and Humanities Research Council* and to Cam Harvey, Julien Prat, Agostino Capponi, and Fas Qureshi for insightful discussions and feedback.

The plumbing of the traditional world of finance is problematic: Assets are stored in silos; transfers are complex, slow, and require the involvement of and coordination among many parties; access to the value transfer infrastructure is limited to a small number of trusted institutions, allowing these to extract rents. Permissionless blockchains such as Ethereum are a fundamentally different type of plumbing and promise to solve many of these problems: they are a digital resource infrastructure for the storage and transfer of items of value on a single and open system.

A storage and transfer infrastructure by itself, however, is not a marketplace. In the first half decade of its existence, trades of blockchain-based items or tokens occurred mostly on centralized, "off-chain" exchanges, thus reducing a blockchain to a settlement infrastructure. However, most blockchains (Bitcoin being the exception) can do more: execute code. By mid 2020, blockchain developers have found a way to use this functionality to build so-called decentralized finance applications (DeFi dapps) for borrowing and lending, creating and managing derivatives and digital representations of fiat currencies, and trading of tokens. These dapps run directly on the blockchain and do not require the involvement of an off-chain institution. This paper studies the functionality and design of a dapp that's key for DeFi: decentralized trading platforms.

Although it is straightforward to organize trading directly on a blockchain, running a traditional limit order market on a network such as Ethereum is not practical. A limit order is simply a set of instructions —transfer an asset if someone is willing to pay a specific price— and these instructions can be translated into code that runs on a blockchain. However, each new limit order submission involves a fee to blockchain miners, and thus running a market with high message volumes where users monitor and often-amend their orders would not only be cumbersome but very expensive. Such an approach would also be resource intensive because in contrast to a centralized exchange where a single server collects and processes all orders, in a blockchain every one of the 10,000+ nodes must execute the same code.

Decentralized *swap* exchanges such as UniSwap, SushiSwap, or Balancer follow a different approach. Liquidity providers deposit pairs of assets into a smart contract. Liquidity takers can then trade against this pooled set of assets at a price that is determined by the quantities of the existing deposited assets. The promise and innovation of decentralized swap exchanges is that they use blockchain smart contracts to *aggregate* and *automate* the provision of liquidity, or, put differently, that an algorithm implements an otherwise decentralized market. Since 2020, these swap exchanges have been extraordinarily successful as they process more trading volume than many prominent centralized exchanges, including the recently IPO-ed Coinbase.

Yet there is a problem. As I outline in this paper, the hard-coded pricing function that all of the swap exchanges use creates persistent disincentives and arbitrage opportunities, which imposes excess costs on users and liquidity providers. I also propose a pricing approach based on a traditional economic model that overcomes these issues.

All swap exchanges use a "bonding curve" approach in the sense that the price is a deterministic function of the supply of assets in the smart contract. The pricing formula is also referred to as "constant product (automated) market making" (henceforth: CPMM) and best explained by example.[1] Suppose X units of token A and Y units of token B have been deposited in a swap exchange smart contract. The ratio Y/X is

[1]The first mention of automated decentralized market making is in a 2016 Reddit post by Vitalik Buterin. Martin Koppelmann of Gnosis expands on this idea and proposes a constant product automated market making (henceforth: CPMM) pricing scheme.

the implicit marginal price of an A token measured in B tokens. If the B token is a stablecoin, i.e. a digital representation of a fiat currency, then the exchange rate Y/X is the dollar-price of an infinitesimal amount of A tokens. Constant product pricing means that for the constant $c \equiv X \times Y$, the price for any quantity x of A tokens is the amount y of the B tokens such that $c = (X - x) \times (Y + y)$.

Why is this functional form problematic? The way blockchain settlement is organized (almost always) makes it possible for anyone to front-run a trade (I explain the details of the mechanics in the next section). Although being front-run is a nuisance and costly for an investor, it is only an economic concern if the pricing function itself guarantees the front-runner a profit. The first main contribution of this paper is to show this is *always* the case for CPMM pricing[2] but not for the alternative pricing function that I derive from a financial market microstructure model (Biais (1993)).

I then take a broader view and describe three additional properties that make an automated market making price function desirable in a blockchain environment. The key consideration behind these additional conditions is that the pricing function should not create an incentive to generate economically meaningless trades. Such trades would congest the network, which creates particularly high social cost in blockchains because each trade needs to be processed by all blockchain nodes.

The second[3] condition is "size consistency:" the pricing mechanism should not incentivize the splitting of a large trade into many small ones. This feature is desirable because of operational efficiency: there would be no economic value created by breaking

[2]I am not the first to highlight this problem — see, e.g., Vitalik Buterin's post here. Angeris and Chitra (2021) formalizes the concept as "path deficiency." Some DEX protocols limit price "slippage," but this does not eliminate the arbitrage problem. My contribution is to develop a functional form of pricing based on a economic model that not have these deficiencies.

[3]The first is the absence of intrinsically profitable front-running.

up an order into smaller units. This requirement is not trivially satisfied: many microstructure models have uniform pricing (each unit of an order costs the same), e.g., Kyle (1985). Such pricing does not satisfy this property. Discriminatory pricing as found in limit order books satisfies size consistency (see also Glosten (1994)). CPMM pricing satisfies size consistency; the original Biais (1993) framework that I base my model on does not, but my modified framework does.

The final two properties relate to multi-venue pricing. The third requirement is that splitting liquidity across venues should not be advantageous. Both pricing models that I consider satisfy this requirement.

For the fourth condition, consider a scenario with two identical swap exchanges. If liquidity is split between the two, then, intuitively, an investor should split her order (evenly) between the venues. In practice, users will often not split perfectly (e.g., because of limited attention or technological constraints), and that will lead to a price dislocation after the trade. One way to resolve this price disparity is to submit a trade that offsets the over-traded amount and thus equalizes the price across both venues. Specifically, if the original trade was a buy for quantity $2x$ on venue A, buying x on B and selling x on A would equalize prices across both venues. The fourth requirement is that any trade in excess of x should be unprofitable. The proposed alternative model satisfies this condition, whereas CPMM violates it. That is a problem because it would lead to "ping-pong trading" with economically meaningless transactions.

In the last part of the paper I discuss how mining (and trading) fees can mitigate some of the undesirable features of CPMM pricing because they reduce front-running profits and I then establish some empirical stylized facts, based on a small dataset from UniSwap, the largest decentralized swap exchange. For very liquid contracts, the potential loss for investors from CPMM pricing is negligible. For less liquid trading pairs, however, the loss is substantial: in January 2021, more than 10% of investors who traded “wrapped” Bitcoin against USDT stood to lose (theoretically) 53 bps or more of their trade size by using a DEX compared to traditional pricing.

The stylized facts aside, that CPMM shortcomings create profitable trades is not a hypothetical problem. CPMM arbitrage/front-running profits are one form of so-called Miner Extractable Value (MEV) (see the appendix for more information), and as Daian, Goldfeder, Kell, Li, Zhao, Bentov, Breidenbach, and Juels (2019) outline in detail, bots scan and exploit profit opportunities. The presence of such profit opportunities is associated with so-called priority gas auctions (PGAs) that have raised the costs of transactions on Ethereum substantially since mid 2020 (coincidentally, after the emergence of decentralized swap exchanges in 2020).

In the nascent economic literature on decentralized finance there are currently three other papers that study decentralized swap exchanges. Lehar and Parlour (2021) describe the returns to liquidity provision, and they study the competition of CPMM with a limit order book. Aoyagi and Ito (2021) study the interaction of trading and prices when centralized and decentralized swap exchanges co-exist in a model with asymmetric information. Capponi and Jia (2021) examine the impact of price volatility, caused by trading on centralized exchanges, on the welfare of liquidity providers in automated, constant product swap exchanges. My focus in this paper is on not on competing systems, but on the intrinsic properties of automated, DEX pricing rules.

I. Market Making Models and Pricing

A. Constant Product Automated Market Makers

An investor wants to trade quantity x of token A and receive (or make a payment in) token B. Under constant product market maker pricing (CPMM), liquidity providers make deposits for the pair of tokens A and B in a smart contract. Specifically, they provide aggregate quantities X units of tokens A and Y units of tokens B and define $c \equiv X \cdot Y$. Pricing is such a liquidity demander who wants to purchase $x \in (-\infty, X]$ of the A token, has to pay y units of the B tokens such that $(X - x) \times (Y + y) = c$ (and the reverse for $y \in [-Y, \infty)$). Therefore, one can think of this exchange rate as the fundamental value of an A token measured in B tokens. The direct implication is that the marginal price of an A token measured in B tokens is Y/X. The cost $p^{\mathsf{cmm}}(x)$ of purchasing quantity x, where I use superscript cmm to signify the CPMM pricing schedule, is derive in the following lemma.

Lemma 1 (CPMM Pricing): *To trade x units of token A, an investor pays*

$$y(x) = x \times \frac{Y}{X - x} := p^{\mathsf{cmm}}(x, X, Y). \tag{1}$$

The proof follows by rearranging the equation $(X - x) \times (Y + y) = c$ and is omitted. After the trade, the contract contains $X - x$ units of token A, and $XY/(X - x)$ units of B. In practice, users have to pay an additional fee to liquidity providers, which I will discuss in more detail in Section III.; I abstract from such a fee here in order to limit the number of free model parameters.

The major appeal of this formulation is that the price is a simple, code-able rule that is based only on information contained in the contract at the time when a trade settles. From an economist's perspective, however, the formulation is problematic. The key inputs are the quantities of the supplied tokens, and these depend on the liquidity providers' willingness to provide liquidity, conditional on the expected compensation. This key decision is not modeled, and it is unclear whether the particular pricing function aligns with the costs of liquidity providers.

B. Traditional Market Making with Inventory Risk

My goal is to develop a simple economic model of market making that accounts for the liquidity providers' decisions and that can be used as a reference point for constant product market making.

The simplest and, in my opinion, most applicable class of frameworks that apply to swap exchanges are those that model market maker pricing as driven by inventory risk: when accepting a trade, liquidity providers need to absorb a risky inventory, and prices are set to compensate liquidity providers for their risk. Such a model would explicitly account for the liquidity providers' decision to supply funds. Prominent models in this literature originate from Ho and Stoll (1981); I use a variation of Biais (1993), because it is a very tractable framework. I will henceforth refer to the pricing model that I develop here as the "traditional" market maker pricing model.

Specifically, I assume that the risky token A has a fundamental value that is normally distributed with mean V and variance σ^2. For simplicity, I assume that security B is riskless (i.e., it is a nummeraire such as cash). Information regarding the distribution of the fundamental value is public knowledge. The asset is infinitely divisible.

All trades go through a group of N intermediaries, who require compensation in exchange for taking a risky inventory in token A. The intermediaries are risk averse and have negative exponential utility of wealth w (which displays constant absolute risk aversion (CARA)) with risk aversion coefficient $\kappa > 0$, and inventory of A tokens I_i, with $\sum_i I_i =: I$. If $p^{\mathsf{tmm}}(x_i)$ denotes the payment that the intermediary receives for quantity x_i, then with CARA utility intermediaries have utility of terminal wealth $u(w) = -e^{-\kappa w}$, where $w = -(vx_i - p^{\mathsf{tmm}}(x_i)) + I_i \cdot v$ and v denotes the value of an A token measured in B tokens, x_i is the quantity of A tokens that they sold at price p.

In the original Biais (1993) model, that a trader arrives at the market once, makes a single trade, and all tokens trade at a uniform price.[4] As I discuss in the introduction, however, some of the concerns such as front-running have an implicitly dynamic flavor in the sense that there may be multiple related trades and that in blockchains we need to avoid scenarios where a single trade gets split into multiple ones to improve operational efficiency. With uniform pricing, investors would have an incentive to split their trades into infinitesimal orders sizes and trade "along" the price curve; this would allow them to reduce the total cost by half compared to trading the whole quantity in one go.[5] Such behavior would create network congestion. I therefore amend the Biais model and assume that market makers post a pricing function $p(\cdot)$ and that when purchasing quantity x, investors pay $\int_0^x p(t)dt$. In other words, the marginal unit may cost more than the first unit.

For simplicity I assume that the supply of the riskless token B is ubiquitous, i.e.,

[4]One way to think about this is that there is an implicit assumption that traders are not anonymous so that market makers can detect repeated trading accept trades only at prices that reflect the full size of an order.

[5]See van Kervel, Kwan, and Westerholm (2020) for a model where order splitting across time is intrinsically beneficial, even in the absence of asymmetric information.

that intermediaries can borrow these tokens at an interest rate of 0. Namely, intermediaries each specify for each price p of an A token measured in the B token, how many tokens they are willing to buy (or sell) $\forall p \in \mathbb{R} \;\; x_i(\cdot) : p \to x_i(p)$. Markets clear at a uniform price such that $\sum_{i=1}^{N} x_i(p) = x$.

Lemma 2 (Equilibrium Price in Traditional Market Making): *To buy x units of token A from the N intermediaries investors pay a marginal price*

$$p(x, I) = V + \frac{2\kappa\sigma^2}{N}(x - I). \tag{2}$$

To condense the exposition, I define parameter $\ell := \kappa\sigma^2$; it signifies the liquidity or the price impact cost in this intermediated market.

Proof. Intermediaries choose quantities x_i given price p, to maximize their expected utility, $\max_{x_i} \mathsf{E}U[-vx_i(p) + \int_0^{x_i} p(t)dt + I_i \times v]$. For CARA-normal frameworks, this task reduces to maximizing the certainty equivalent:

$$\max_{x_i}[I_i \times V - (Vx_i - \int_0^{x_i} p(t)dt)] - \frac{\kappa}{2}\sigma^2[-x_i + I_i]^2,$$

where V denotes the expected value of the asset. The maximization problem results in the following first order condition:

$$V - p(x_i) - \kappa\sigma^2 \times I_i + \kappa\sigma^2 x_i = 0.$$

Solving for x_i yields the (inverse) demand schedule

$$x_i(p) = -\frac{V-p}{\kappa\sigma^2} + I_i.$$

The market clearing condition

$$\sum_{i=1}^{N} x_i(p) = x$$

implies, substituting for x_i, and simplifying, that

$$\sum_{i=1}^{N} \left(-\frac{V-p}{\kappa\sigma^2} + I_i \right) = x$$

which I solve for p to be

$$\Leftrightarrow \quad p(I,x) = V + \frac{\kappa\sigma^2}{N} \, (-I+x), \tag{3}$$

where I denotes the combined inventory of the intermediaries: $I = \sum_{i=1}^{N} I_i$. $\square$

As with CPMM pricing, this pricing formula is mechanical and can be programmed as a smart contract (subject to sufficient availability of B tokens in the contract). Price changes in this model occur for two reasons: changes in the fundamental value V, and trades which require intermediaries to absorb a risky inventory. When an investor approaches the intermediaries who hold total inventory I in order to buy x units, and the investor pays, the total cost for buying x is

$$p^{\mathsf{tmm}}(x,I) := \int_0^x V + \frac{\ell}{N}(t-I) \; dt = x\left(V - \frac{\ell}{N} I\right) + \frac{\ell}{2N} \, x^2,$$

where I use superscript tmm to signify the traditional market maker pricing. Although the marginal price is linear, the cost is a convex function of the quantity.

In contrast to the CPMM pricing, the traditional formulation accounts for the decision of liquidity providers and the pricing function compensates them fairly for their services. In Section IV. I will compare the two pricing functions further to establish whether market makers receive fair compensation.

II. Desirable Properties of Pricing Functions

A. Trading and Institutional Features of Blockchain Operations

Blockchain-based Trading. At its core, a blockchain is a computer network that provides the guaranteed execution of code. For Bitcoin the code is primitive and only does one thing: transfer Bitcoins. Second generation blockchains such as Ethereum are Turing complete, meaning that they can process essentially any code, and a block in the chain, therefore, consists of a set of pieces of code that the network executes. This feature allows the organization of trading on the blockchain, because an order is the proposal of a contingent contract and a trade is an atomic swap of one token for another that can be expressed in code (more details are in the appendix). However, blockchains to date are not efficient in the execution of code because all of the thousands of network nodes have to execute every piece of code. Running a full exchange on the blockchain would therefore be resource inefficient and expensive (users would have to pay not just for trades but for every order). Some decentralized exchanges use workarounds such as keeping the limit orders off-chain and using the "mainnet" only for the settlement of trades. Even with that feature, however, it is critical that the trading institutions are designed to not create an incentive to submit economically valueless trades, defined as trades beyond the transfer of the quantity that the end-user desires that create a utility transfer from one user to another without increasing overall welfare.

Blockchains and Front-Running. The possibility of front-running is a consequence of the organization of public blockchains: verified but non-settled transactions are added to "mem-pools" from which miners select transactions for inclusion in a new block. Transactions in mem-pools are publicly visible and can therefore be front-run. Figure 1 illustrates the mechanics of front-running. Most mining protocols mechanically prioritize transactions by the (gas) fees that they offer.[6] Any user who monitors the mempool can submit the same trade but with a higher mining fee so that the trade will have higher priority of getting included in the next block. The price of a DEX swap trade is determined only at the time of settlement —not at the time when the trade was agreed upon— because the price depends on the number of tokens that are in the contract at the time when a miner adds the trade to a block. Trades are ordered even within a block and they execute in sequence, and therefore the number of tokens in a contract can change even while the different transactions in a block execute. The front-run person's trade would execute later than that of the front runner and it would therefore have a worse price because the front-runner has absorbed liquidity from the pool. At this point, the front-runner holds a position and has not yet earned a profit. For that, the front-runner needs to submit a second trade that reverses the position.

Although being front-run is costly for the front-run person, is it a genuine concern?

[6]See, for instance, this link for the part of the Parity mining protocol that describes the ordering of transactions by fees.

The answer is yes, if the front-runner can always guarantee a profit when closing the position and the answer is no when the front-runner cannot profit of her actions (because, why do it?). The critical issue is therefore the pricing rule of the DEXes. If the pricing rule of the DEX is such that the two round-trip trades of the front-runner guarantee her a profit (at the expense of the original liquidity demander) then this lowers the incentive for people to participate in the market. As I show below, for CPMM trading this is always the case.

B. *What are desirable features of pricing curves?*

The first property is the absence of *guaranteed front-running profits.* Namely, front-running refers to a situation where a trader Alice sees the trading intention of another trader Bob, mimics Bob's trade but acts before him, and then does an offsetting trade to close her original position. Front-running is usually costly for the person who is front-run, but it is not clear whether front-running will happen just because it's possible. To be a concern, front-running would have to be intrinsically profitable. If that's the case, then the underlying pricing function is highly undesirable for a setting with a public blockchain. I consider only cases where the original amount x is less than $X/2$, so that front-running is possible (for the total quantity bought would be $2x$).

Definition (Absence of Profitable Front-running): *For trade size $x \in (0, X)$ and pricing function $\mathsf{p}(size, inventory)$ holds $-\mathsf{p}(x, I) + \mathsf{p}(-x, I + 2x) \leq 0$.*

The second property is additivity or *size consistency* which says that splitting an order into two (or more) and trading one after the other should cost no less as sending a single order of the same size "in one go," ignoring blockchain processing costs. Formally, Definition (Size Consistency): *For trade size* $x \in (0, X)$*,* $\alpha \in (0,1)$*, and pricing function* $\mathsf{p}(size, inventory)$ *holds* $\mathsf{p}(\alpha x, I) + \mathsf{p}((1-\alpha)x, I + \alpha x) \geq \mathsf{p}(x, I)$.

If this condition does not hold for a theoretical model, then users would split their order into several parts, and these additional trades would create network congestion without economic gain. In practice, traders commonly break large "parent" orders into many small "child" orders to reduce costs, but such behavior is often driven by considerations other than the pricing function such as shading their private information (which requires a different model altogether).

The next set of properties applies to situations when there are multiple trading venues with the same pricing model.

The third desirable property relates to the splitting of liquidity (e.g., because a new venue opens for business). The question is whether *liquidity is divisible*, i.e., are trading costs the same when liquidity is concentrated on one system compared to it being split across two different ones? Modern markets are often fragmented and it is important to understand whether the pricing function itself generates an explicit benefit or cost when liquidity is fragmented (beyond the obvious fact that splitting trades involves twice the miner fees).

Definition (Additivity of Liquidity): *For trade size* $x \in (0, X)$*,* $\alpha \in (0,1)$*,* $k > 0$*, and pricing function* $\mathsf{p}(size, inventory)m$ *holds* $k \cdot \mathsf{p}(x/k, I)|_{N \to N/k} = \mathsf{p}(x, I)|_{N \to N}$.

The fourth property relates to (accidental) single-venue over-trading. Namely, when liquidity is split across multiple venues, a trader should use all venues for a trade.[7] In

[7]Generally speaking, if liquidity is divisible, then traders should split their trades between venues in proportion to their liquidity. For simplicity I look at the case where the liquidity is the same on two venues, so the trader should have traded half her volume on each venue.

practice, in the DeFi world there are order routing services such as 1inch and Paraswap that perform this service and break up large orders across protocols.[8] Suppose, however, that a trader doesn't do this and instead uses only one of two identical venues (for instance, because s/he is unsophisticated or impatient). This will move the price for subsequent trades on one venue while leaving the other unchanged. Multi-venue arbitrage occurs when the selling of an amount larger than half of the last trade on the first venue and buying it back on the other venue is intrinsically profitable.[9]

Definition (Absence of Multi-Venue Arbitrage): *For trade size $x \in (0, X)$, $\alpha \in (0, 1)$, and pricing function $\mathsf{p}(size, inventory)$ holds: $\forall \alpha > 1/2$, $\mathsf{p}(-\alpha x, I + x) - \mathsf{p}(\alpha x, I) < 0$.*

If this condition is violated, we may see "ping-pong" trading where arbitrageurs repeatedly trade on different venues to exploit profitable opportunities.

C. Properties of Traditional Market Making Prices

Without loss of generality, in the proofs for traditional pricing, I set $V = 0$ and $I = 0$. All results are based on buys; a symmetric argument holds for sells. I use $p^{\mathsf{tmm}}(x, 0)|_{N \to N'}$ to signify the pricing when the N market makers are reduced to $N' \leq N$ market makers (e.g., because N is split between two parallel systems). As the result below shows, traditional pricing satisfies all four desirable properties.

Proposition 1 (Properties of Standard MM Pricing):

For $x \in (0, X)$, $\alpha \in (0, 1)$ and $k > 0$:

[8]I thank Julien Prat for pointing out these services.

[9]Why more than half the original order? Effectively, the original trader over-traded with one venue. Trading half the original amount as described would align liquidity on both venues to be as it should have been had the original trader split the trade equally among the two venues.

1. *Unprofitable Front-running:* $-p^{\mathsf{tmm}}(x, I) + p^{\mathsf{tmm}}(-x, I + 2x) = 0$.
2. *Non-Profitable Order Splitting: for* $\alpha \in (0, 1)$*:* $p^{\mathsf{tmm}}(\alpha x, I) + p^{\mathsf{tmm}}((1-\alpha)x, I + \alpha x) = p^{\mathsf{tmm}}(x, I)$.
3. *Additivity with split liquidity,* $k \cdot p^{\mathsf{tmm}}(x/k, I)|_{N \to N/k} = p^{\mathsf{tmm}}(x, I)|_{N \to N}$.
4. No *Multi-venue arbitrage:* $\forall \alpha > 1/2$, $p^{\mathsf{tmm}}(-\alpha x, I + x) - p^{\mathsf{tmm}}(\alpha x, I) < 0$.

In the traditional model, market makers price "along" the marginal price curve. Splitting an order x into quantities αx and $(1-\alpha)x$, therefore, simply means that after trading αx, the market maker accepts quantity $(1-\alpha)x$ at an inventory of αx.

Proof. of 1.: As a front runner, the investor buys x, sees the other party buy x from the market makers, and then sells x. Payoffs are the amount received from selling minus the amount paid for buying. As before, without loss of generality, I set the initial inventory to $I = 0$. The front runner pays $p^{\mathsf{tmm}}(x, 0)$ for the initial position and receives $p^{\mathsf{tmm}}(-x, 2x)$ when liquidation (the inventory has increased by $2x$ from the front-runner's as well as from the front-run's trade). Then

$$\begin{aligned} -p^{\mathsf{tmm}}(x, 0) + p^{\mathsf{tmm}}(-x, 2x) &= -\int_0^x \ell t dt + \int_0^x \ell(-t + 2t) dt \\ &= \int_0^x \ell(-t(-t + 2t) dt = 0. \end{aligned}$$

Proof of 2.: After selling αx to an investor, the aggregate inventory of intermediaries changes by $-\alpha x$. Then,

$$\begin{aligned} p^{\mathsf{tmm}}(\alpha x, I) + p^{\mathsf{tmm}}((1-\alpha)x, I+\alpha x) &= \int\limits_0^{\alpha x} (V + \ell(t-I))dt + \int\limits_0^{(1-\alpha)x} (V + \ell(t - I + \alpha x))dt \\ &= \int_0^x \times (V + \ell(t-I))dt. \end{aligned}$$

Proof of 3.: Additivity with split liquidity follows directly from the market setup because the model assumes a single market clearing price, and it is irrelevant where market makers are. Formally, note first, that when the N market makers are split into k equal groups then $\ell = \kappa\sigma^2/N \to \kappa\sigma^2/(N/k) = k \cdot \ell$. Therefore

$$\begin{aligned} & k \cdot p^{\mathsf{tmm}}(x/k, I)\big|_{N\to N/k} = k \cdot \frac{\ell k}{2}(x/k)^2 = \frac{\ell}{2}x^2 \\ = \quad & p^{\mathsf{tmm}}(x, I)\big|_{N\to N} . \end{aligned}$$

Proof of 4.: The result follows:

$$\begin{aligned} & p^{\mathsf{tmm}}(-\alpha x, x) - p^{\mathsf{tmm}}(\alpha x, 0) \\ = \quad & \alpha x \frac{\ell}{2}(-\alpha x + x) - \alpha x \frac{\ell}{2} \alpha x = \frac{\ell}{2}\, \alpha(1-2\alpha)x^2. \end{aligned}$$

The last expression is negative if $\alpha > 1/2$. $\square$

D. Properties of Constant Product Automated Market Making Prices

As a next step, I examine the required conditions for constant product market maker pricing. Recall that $p^{\mathsf{cmm}}(x, X, Y)$ signifies the amount of token B that one has to pay for x many of the A tokens when the contract contains X of the A tokens and Y of the B tokens.

Proposition 2 (Properties of CPMM Pricing):

For $x \in (0, X)$, $\alpha \in (0, 1)$ and $k > 0$:

1. *Persistent Profitable Front-running:* $-p^{\mathsf{cmm}}(x, X, Y) + p^{\mathsf{cmm}}(-x, X - 2x, Y + y(2x)) > 0$.
2. *Size Additivity:* $p^{\mathsf{cmm}}(\alpha x, X, Y) + p^{\mathsf{cmm}}((1 - \alpha)x, X - \alpha x, Y + y(\alpha x, X, Y)) = p^{\mathsf{cmm}}(x, X, Y)$.
3. *Additivity of liquidity:* $p^{\mathsf{cmm}}(x, X, Y) = k \cdot p^{\mathsf{cmm}}(x/k, X/k, Y/k)$.
4. *Presence of Ping-Pong Trading:* $p^{\mathsf{cmm}}(-\alpha x, X - x, Y - y(x)) - p^{\mathsf{cmm}}(\alpha x, X, Y) > 0$ $\forall \alpha \in (0, 1)$.

In other words, CPMM pricing satisfies price consistency and additivity of liquidity, but allows persistent front running profits and leads to “ping-pong” trading after accidental price dislocations.

Proof. of 1.: The front runner pays $p^{\mathsf{cmm}}(x, X, Y)$ and later sells x where there have been two purchases of x each before (one by the front-runner, and one by the front-run person). I use $y(x) := xY/(X - x)$ for the additional quantity of B tokens that the contract contains after a buyer buys x of the A tokens. For selling x, the front runner then receives

$$p^{\mathsf{cmm}}(-x, X - 2x, Y + y(2x)) = \frac{xXY}{(X - x)(X - 2x)}.$$

Then I simplify to get

$$p^{\mathsf{cmm}}(-x, X-2x, Y+y(2x)) - p^{\mathsf{cmm}}(x, X, Y) \quad = \quad \frac{2x^2Y}{(X-x)(X-2x)} > 0. \tag{4}$$

Proof of 2.: Since $y(x) = xY/(X-x)$, I have that after purchase of x, there are $XY/(X-x)$ of the type B tokens. Therefore

$$p^{\mathsf{cmm}}((1-\alpha)x, X-\alpha x, Y+y(\alpha x)) = \frac{(1-\alpha)xXY}{(X-\alpha x)(X-x)}.$$

Adding the costs of the two trades I get

$$\begin{aligned}
&p^{\mathsf{cmm}}(\alpha x, X, Y) + p^{\mathsf{cmm}}((1-\alpha)x, X-\alpha x, Y+y(\alpha x)) \\
&= \frac{\alpha xY}{X-\alpha x} + \frac{(1-\alpha)xXY}{(X-\alpha x)(X-x)} \\
&= \frac{xY(\alpha(X-x)+(1-\alpha)X)}{(X-\alpha x)(X-x)} = \frac{xY}{X-x} \\
&= p^{\mathsf{cmm}}(x, X, Y).
\end{aligned}$$

Proof of 3.: I have

$$kp^{\mathsf{cmm}}(x/k, X/k, Y/k) = \frac{kx/kY/k}{X/k - x/k} = \frac{xY}{X-x} = p^{\mathsf{cmm}}(x, X, Y).$$

Proof of 4.: After a trade of x on venue 1, an arbitrageur sells $\alpha x < x$ on venue 1 and buys αx on venue 2. The cost for this trade is

$$\begin{aligned} & p^{\mathsf{cmm}}(-\alpha x, X-x, Y-y(x)) - p^{\mathsf{cmm}}(\alpha x, X, Y) \\ = \; & \frac{\alpha x Y}{(X-\alpha x)(X-(1-\alpha)x)(X-x)} \\ & \qquad \times \big(2X(X-x) + x^2(1-\alpha)\big) > 0. \end{aligned}$$

□

The possibility of profitable front-running as well as ping-pong trading (or multi-venue arbitrage) are therefore intrinsic concerns under the given pricing model. This problem of front-running is indeed well-known, which is why all swap trading venues include an option for traders to limit the "slippage", i.e., when asking to arrange a trade, traders can limit the price impact/change that their order will face. As Proposition 1 shows, however, a different pricing system can eliminate this problem altogether.

III. Impact of Trading Fees for CPMM

A. Mining and Trading Fees

Trading Fees are best thought of as compensation for liquidity providers. In addition to the pricing function, in practice CPMM users also pay a fee to liquidity providers and so far I have abstracted from this fee to simplify the exposition and to provide a comparison between the pricing models that is not muddled by an additional variable. The slope of the pricing curve in the traditional pricing model is an exact representation of the compensation that liquidity providers obtain for taking on risk under market clearing. No other compensation is warranted in a competitive market.

A trading fee makes trading more costly and add costs to front-running and arbitrage. A fee therefore reduces the set of trades to which Proposition 2 (1) and (4) apply, which means that not all trades create a risk-free profit opportunity, only sufficiently large ones do.

The same insight applies to **mining fees**: uses need to pay miners for each transaction so that it gets added to the blockchain. Mining fees therefore also reduce the set of trades for which front-running and arbitrage is profitable. The difference between mining and trading fees is that trading fees are set by the protocol whereas mining fees can be set by the trader herself, and in what follows I will focus on how users can use mining fees to thwart front-running.

B. Defensive Fees

Equation (4) specifies the front-running profits. These profits do not account for the mining fees or other transaction fees. To be successful, the front-runner has to outbid the original trader for the mining fees and she has to pay mining fees for the second leg of the trade to establish a round-trip transaction. The mining fees for the return transaction therefore have to be high enough to ensure that the second leg of the trade gets included in the same block (otherwise, the front-runner faces a possibly significant execution risk, because of 4.) but not higher to ensure that the front-run trade is processed first. With a sufficiently high fee, the original traders can make front-running unprofitable.

Proposition 3: *For each quantity x there exists a mining fee f that the original trader* *can submit to render front-running unprofitable; the fee is increasing in x.*

Proof. If the original trader pays f, the front-runner will pay $f + \epsilon_1$ for the first leg and $f - \epsilon_2$ for the return trip. For front-running to be profitable, equation (4) with mining fees has to satisfy

$$\frac{2x^2Y}{(X-x)(X-2x)} > 2f + (\epsilon_1 - \epsilon_2). \tag{5}$$

For ϵ_1, ϵ_2 arbitrarily close to zero, there exists an $\bar{f}$ such that for any mining fee $f > \bar{f}$, front running is unprofitable. Furthermore, the left hand side of (5) is an increasing function of x. $\square$

In other words, although front running is intrinsically profitable, when taking fees into account, traders do not have to accept arbitrarily large prices because of the possibility of front running. The left hand side of (5), however, is a convex function of x and therefore for large quantities, "protective" fees increase more than proportionally.

A second source of fees are the explicit trading fees to liquidity providers. There is a subtle difference to transaction/mining fees in that these fees are usually proportional to the amount traded (in UniSwap V2, they were 30bps of the transaction amount; the details differ between protocols). However, the principle is the same: these fees can prevent front-running as they make each leg of the front-running trade more expensive.

Fees would not change the key features for traditional pricing: there is still no multi-venue arbitrage, and there is still no profitable front-running.

IV. Cost Comparison of the Two Pricing Functions

A conceptual shortcoming of CPMM pricing is that the deposited quantities of the tokens in the contract are exogenous and that it is unclear if liquidity providers obtain fair compensation for their willingness to take an inventory risk. In the traditional model, this problem does not arise because the price function precisely compensates liquidity takers for their risk. Consequently, if both models have the same liquidity providers, then unless the prices are identical, market makers will always either win or lose under CPMM for almost any quantity. In what follows I formally describe when they lose and when they receive too much compensation for the risk they take.

As a first step, I describe how to interpret the parameters of the different pricing models to enable a viable comparison. For traditional market maker pricing, there are four key variables: the liquidity factor ℓ, the number of market makers N, aggregate inventory I, and the true value (or, rather, its mean) V. Variables ℓ and N determine the slope of the linear function, and inventory I and true value V shift the intercept of the pricing function. An inventory in the traditional market maker model is always considered to be costly whereas presumably some users agree to provide liquidity to use their buy-and-hold assets while earning additional yield through lending out their securities (akin to mutual funds). In that sense, any inventory I should be seen as *excess* amount of inventory beyond a target amount. To compare the two pricing functions, I therefore set $I = 0$ and interpret X as an implicit target amount of inventory and assume that market makers deposit single units of A tokens so that $N = X$.

Next, the constant product market maker model is not concerned with a "true" value for the asset. However, liquidity providers for the smart contract would not want to trade at "wrong" prices. Therefore, to compare CPMM and traditional market making, bar any trading, the implied marginal price for A tokens must coincide with the true value V, i.e. $V = Y/X$. Rewritten as $Y = VX$, this is the total amount of cash or B tokens that market makers contribute, and this amount is equivalent to the cash value of their inventory of A tokens.

Next, substituting the simplification and setting $p^{\mathsf{tmm}} = p^{\mathsf{cmm}}$, I find that the two price functions coincide for two values of x. One is $x = 0$, by construction. The other is

$$x^* \equiv \frac{X}{\ell}(\ell - 2V).$$

Consequently, for any trade size other than x^* (or 0), market makers either get over- or under-compensated under CPMM.

Figures 2 and 4 plot the two cost (not price) functions p^{cmm} and p^{tmm} for $V = 1$ and for several values of X and ℓ as indicated in the respective plot legends.

Figure 2 provides a plot for the entire range of feasible values, Panel A in Figure 4 focuses on the subset of the plot for which $x^* < 0$, and Panel B in Figure 4 focuses on the subset of plot for which $x^* > 0$. Dotted lines are for p^{cmm}, dashed lines are for p^{tmm}. Negative values of x signify a sale and the cost, therefore, indicates revenue earned. For the CPMM case, in the limit for $x \to -\infty$, $p^{\mathsf{cmm}} \to -VX$ or $p^{\mathsf{cmm}} \to -Y$, because the investor can at most extract all the cash that has been deposited by the liquidity providers. Likewise, as $x \to X$, i.e., the investor tries to buy all the A tokens in the contract, the cost rises without bound.

The two extreme scenarios highlight the different underlying ideas: for traditional market making, the cost of a buy increases quadratically, but there is no explicit limit and the assumption is that market makers can provide or absorb any quantity of the desired token. Likewise, they are willing to buy arbitrarily large quantities, but at some point, they are no longer paying the seller, they require to be paid to accept further assets.

The plots further indicate that the two cost functions intersect at x^* and touch at 0. The following result summarizes the relationship of the pricing functions.

Proposition 4 (Comparison of Prices): *For $x > x^*$, CPMM pricing over-compensates liquidity providers, and for $x < x^*$ liquidity providers are under-compensated.*

Proof. With two points of intersection, there are two scenarios: $x^* < 0$ and $x^* > 0$.

- When $x^* < 0$, then $\forall x > x^*$, $p^{\mathsf{tmm}}(x) < p^{\mathsf{cmm}}(x)$, i.e., investors pay less in the standard model for $x > 0$ and they receive more for $x < 0$. Furthermore, $x < x^*$, $p^{\mathsf{tmm}}(x) > p^{\mathsf{cmm}}(x)$ such that investors receive less in the standard model.

- When $x^* > 0$, then for all values $x \in (0, x^*)$, $p^{\mathsf{tmm}}(x) > p^{\mathsf{cmm}}(x)$, which means that CPMM provides a "better" price for investors because investors receive a higher price for their sales. For $x > x^*$, $p^{\mathsf{tmm}}(x) < p^{\mathsf{cmm}}(x)$ and standard pricing is better for investors. Finally, for $x < 0$, $p^{\mathsf{tmm}}(x) > p^{\mathsf{cmm}}(x)$ which means that investors receive less for what they sell in the standard model.

$\square$

This finding shows that from a pricing perspective it is not straightforward to order the two pricing models, except for the knife edge case when $X\ell = 2V$. Generally, the intuition is that the more liquid the traditional market is (ℓ is small), the larger is the region of prices for which traditional pricing is better. Likewise, the more liquidity market makers are willing to provide forin the CPMM model (X is large), the larger is the region of prices for which CPMM prices are superior for investors.

The values X and ℓ should be connected. Namely, I assume that the starting inventory of market makers is $I = 0$ and that the market makers assess inventory other than their "comfort zone" to be costly. When X is large, market makers are willing to hold the A token, which implies that ℓ is small. In other words, X and ℓ are conceptually negatively related.

V. A Calibration Exercise

In this section, I establish a few stylized facts to link the model to market data. First, the traditional market maker model requires a choice for the risk aversion coefficient κ. Using a meta-analysis, Babcock, Choi, and Eli Feinerman (1993) (Table 1) report that the CARA coefficient κ falls anywhere between .00001 and 0.5, where most of the values in the literature are at the small end of the spectrum; in my subsequent calibration, I will use $\kappa = 0.0005$.

Second, I will compare calibrated trading costs for three common trading pairs: Bitcoin–USD, Ether-USD, and USDT-USDC, where the latter is the exchange rate of two common stablecoins, USDT, issued by Tether Ltd., and USDC, issued by Coinbase Inc. Here, USD and USDC are the nummeraire tokens Y.

Table I lists the estimates that I use for the calibration. For the fundamental values of these tokens V I will use the January 17, 2020 approximate prices of \$36,000, \$1,200, and \$1 for BTC, ETH, and USDT. To compute their standard deviations I collected data from CoinDesk (for BTC and ETH) as well as Nomics for USDT-USDC for the time from January 16, 2020 to January 16, 2021.

To determine the values X, the capital at risk, I use information on the liquidity provision in UniSwap trading pairs, currently the most heavily used CPMM protocol, where I used either Tether's USDT or Coinbase's USDC as the baseline token Y, and I use so-called wrapped Bitcoin wBTC as a proxy for Bitcoin. On January 17, 2021, the liquidity provision contract for the USDT-ETH pair contained roughly 80K ETH, the USDC-wBTC contained 120 wBTC, and the USDC-USDT contract contained 17M USDT (and a similar amount of USDC). Therefore, the three contracts reflect medium, high, and low liquidity as well as low, medium and high volatility.

For all these parameters, the value x^* for which traditional and CPMM costs coincide is negative (and quite large in absolute value). Therefore, for a very large set of reasonable trade sizes, traditional pricing offers "better" terms to investors. Using these figures, I then compute the excess trading costs as well as the mining fee in (5) that investors have to submit to avoid being front-run. I compute these items for three standard-sized trades of \$1,000, \$5,000, and \$10,000.

Table I summarizes the amounts. For small trade sizes (\$1,000) the excess cost of CPMM are, arguably, negligible. The same holds generally for the highly liquid contract ETH-USD. However, the wBTC-USD pair is much less liquid and implicit costs to avert front-running can be substantial: trading \$10,000 or more of wBTC-USD comes at a combined excess cost of 53 bps for the trade — which is a large cost in the trading world and it is a large cost given that BTC can be traded with bid-ask spread of an order of magnitude less on centralized exchanges. To put this into perspective, for this type of highly traded pair at a price of \$30K, a \$1 bid-ask spread is less than a basis point of a 1 BTC transaction value, and on most centralized markets, spreads are smaller than \$0.10.

Using the first 4,000 transactions on Jan 14 and thereafter from the UniSwap contract for the wBTC-USDC pair, about 14% of transactions exceed \$10K in value, and another 13% have transaction value between \$5K and \$10K. Although it is not clear whether any of these users have suffered from front-running, the possibility implies that they would have been better served with a traditional market maker pricing protocol.

VI. Discussion and Conclusion

Ad-hoc, exogenous pricing functions are unsatisfactory for economists, because such prices are unlikely to reflect optimization and risk-taking behavior. CPMM prices are thus unlikely to reflect demand and supply or to aggregate information, the core functions of market prices.[10] In an economic model, liquidity providers compete to provide the best price. In this paper I modify the Biais (1993) model of inventory-based liquidity provision, which allows me to describe the price function in closed form. I use the model to demonstrate that a pricing model that's derived from economic primitives has desirable properties.

In contrast, the ubiquitous constant product automated market making function

[10]Notably, Aoyagi (2020) develops a theoretical model of investor and liquidity provider behavior around constant product pricing as an exogenous rule. In his model there are three main market participants: noise traders, who trade for personal reasons; informed traders, who know the true value of the underlying asset and who maximize their trading profits, taking account of the effect of their trade on prices (as in Kyle (1985)); and liquidity providers who maximize their profits taking into account the pricing rule and the behavior of the latter two types of traders. Aoyagi (2020) solves the model in terms of traders and liquidity providers' behavior and he shows that the scheme allows liquidity providers to extract rents. Moreover, although users can infer the security's fundamental value from prices, the price usually not coincide with it.

that most decentralized swap exchanges use is not just economically arbitrary. I show here that it also has highly undesirable properties that matter in practice, at least for less liquid tokens (arguably, the part of the market for which the pooling of liquidity holds the most promise). In summary, although swap exchanges have desirable features, such as the institutionalized pooling of liquidity, there are major concerns that need to be (and can be) addressed before these systems can realize their potential.

It is important that DEXes are designed well. Profitable CPMM front-running has been recognised as early as March 2018 and as Daian, Goldfeder, Kell, Li, Zhao, Bentov, Breidenbach, and Juels (2019)[11] show, it leads to a bigger problem: front-running bots that identify profitable transactions, including (but not only) CPMM trades, regularly bid up mining fees is so-called so-called priority gas auctions. These auctions lead to higher income for miners, and, in some cases, miners themselves can extract the revenue from the profitable transactions, which is why these excess costs are summarized under the name *Miner Extractable Value* (MEV). All the economically valueless transactions therefore not only congest the network, they also create a negative externality by raising the price of Ethereum transactions for all other users. Indeed, UniSwap alone, usually accounts for a very large fraction of the computational cycles (or the "gas" usage) of the Ethereum blockchain.[12] A Paradigm Research blog post by Noyes (2021) describes this issue in more detail and estimates that in December 2020, realized MEV amounted to about $120M per day; the author states that UniSwap arbitrage is the most common form, though the data source and data generating process is not clear from the post. There are a number of projects that try to remedy this issue; Aune,

[11]See also the Flashbot Project.

[12]Running data is available at ETH Gas Station; UniSwap's V2's main contract address is 0x7a250d5630b4cf539739df2c5dacb4c659f2488d.

O'Hara, and Slama (2017), for instance, propose a technique to guarantee time priority in private blockchains.

The overall issue, however, is that bots take advantage of arbitrage opportunities and that their actions congest the blockchain, raise prices for all users, and thus hamper the development of the ecosystem. As I argue in this paper, bot activity could be stymied and MEV be eliminated by a better, economics-based design of DEX pricing.

A Background: The Trading of Blockchain Tokens

Prior to the summer of 2020, most crypto-tokens that had been issued on the various decentralized platforms could only be traded on centralized venues such as Polionex, Binance, Kraken, Coinbase, or the now defunct QuadrigaX. These transactions were usually trades of crytocurrencies such as Bitcoin and Ether, the cryptocurrency of the Ethereum network, in exchange for either fiat currencies or other tokens that represented fiat currencies, so called "stablecoins" such as Tether. To trade, users have to register with the platform and transfer their blockchain asset to the "wallet" of an exchange, before they can use the exchange's system to make their trades. Consequently, as part of the process, custody of the asset moves from the user to the exchange. This arrangement is risky, as demonstrated by the numerous hacking and fraud scandals such as Mt. Gox, QuadrigaX, or Thodex.

There are a number of projects that operate decentralized limit order books, e.g. EtherDelta. The idea of these decentralized "venues" is simple. Trades are always exchanges of two blockchain tokens, and so users simply register their limit orders as a smart contract on the blockchain, "locking" the token that can be sold subject to someone sending the desired amount of the other token to the contract. When that happens, the contract initiates an atomic swamp: the buyer receives the bought tokens and the seller receives the payment tokens. This atomic swap is processed by the blockchain miners and happens "in one go" so that failures to deliver cannot happen — the trade is the settlement. A decentralized exchange (DEX) provides users with an interface to enter and sign orders using the cryptographic capabilities of their existing blockchain wallet. The system keeps track of these orders (using information from the blockchain, not its own system) and displays available orders on a website.

Although such a mechanism is workable, it requires that individuals continuously monitor and re-submit orders to supply liquidity, a costly activity. Moreover, liquidity provision also involves the ongoing submission of new liquidity providing orders, which absorbs computational power and costs mining fees.[13] Some trading systems such as dx/dy circumvent this approach by organizing limit orders off-chain and they submit only matched orders as a trade to the blockchain for settlement. However, such a system is not decentralized and therefore requires trust.

Swap exchanges take yet another approach. A swap exchange is a smart contract that pools deposits of *pairs* of token from numerous liquidity providers.[14] Liquidity seekers then interact with this pool of liquidity and offer to add one type of tokens to this contract in exchange for the other part of the pair. The contract then determines

[13]This is in contrast to most centralized exchanges or today's stock exchanges, where order submissions are usually free. The exception is Canada where users have to pay a nominal, sub-penny fee for every order that they submit; see Malinova, Park, and Riordan (2013). There are systems that keep the limit order book offline and only execute the trades on-chair.

[14]Balancer is more general and can hold portfolios of more than two tokens. If a particular pair is not available as a separate contract, some swap exchanges such as SushiSwap offer cross-contract trading. For instance, suppose someone whats to trade USDC for USDT but there is no such contract. If, however, there is a contract for ETH and USDT and ETH and USDC, then the system arranges a swap from USDC to USDT by trading USDC$\rightarrow$ETH$\rightarrow$USDT.

the exchanged quantity (and thus the implied price) based on a mechanical rule. Importantly, liquidity providers do not provide a priced contract, they are entirely passive. It's worth noting that the core code for swap contracts has only about 200 lines.

These properties aside, there are other concerns that merit attention. For instance, prices on a swap exchange do not exist in isolation because blockchain securities can be traded on various venues. This implies that the price of the CPMM is often wrong relative to the broader market, unless liquidity suppliers constantly update their quantities such that the marginal price is in line with what's available elsewhere. Capponi and Jia (2021) studies this issue extensively and they show how volatility prices can lead to surges in transactions fees for the entire blockchain –yet another negative externality– as well as to liquidity freezes. All in all, although swap exchanges have very desirable features, such as the pooling of liquidity, there are several major concerns that need to be addressed for these systems to unfold their potential.

Finally, in Section III. I describe how users can use mining fees as a defense against front running. There is, however, a bigger question relating to the workings of mining. Namely, the idea of a defensive fee is predicated on the idea that someone would have to pay a fee to outbid the original trader and then pay a similar fee to reverse the transaction. The implicit assumption is that this "someone" must pay the miner. But what if the front-runner is the miner itself? Then the defensive fees are irrelevant because the miner can choose the order of transactions in the block and put its own transactions ahead of the front-run trade without paying a fee. In other words, mining fees cannot work as a defense against malicious miners.

It is important to emphasize that even though the word suggests otherwise, miners are not people. Mining follows an open-source algorithm, there is no secret in what miners are doing, their actions are transparent and hard-coded, and they don't take decisions pondering each arbitrage case. If they are malicious, it would be plainly visible (including the fact that their blockchain addresses are known). In principle, it may be possible for a protocol to avoid broadcasting transactions to mempools that are operated by malicious miners. As I argue in this paper, a better approach is to employ a pricing function that doesn't enable exploitative behavior.

REFERENCES

Adams, Hayden, Noah Zinsmeister, and Dan Robinson, 2020, Uniswap v2 core, Whitepaper Uniswap Foundation https://uniswap.org/whitepaper.pdf.

Angeris, Guillermo, and Tarun Chitra, 2021, Improved price oracles: Constant function market makers, Working paper Stanford University https://arxiv.org/abs/2003.10001.

Aoyagi, Jun, 2020, Liquidity provision by automated market makers, Discussion Paper, working paper U.C. Berkeley https://ssrn.com/abstract=3674178.

——— , and Yuki Ito, 2021, Liquidity implications of constant product market makers, Working paper UC Berkeley https://ssrn.com/abstract=3808755.

Aune, Rune, Maureen O'Hara, and Ouziel Slama, 2017, Footprints on the blockchain: Information leakage in distributed ledgers, Working paper Cornell University https://ssrn.com/abstract=2896803.

Babcock, Bruce, E. Kwan Choi, and Eli Feinerman, 1993, Risk and probability premiums for cara utility functions, *Journal of Agricultural and Resource Economics* 18, 17–24.

Biais, Bruno, 1993, Price formation and equilibrium liquidity in fragmented and centralized markets, *The Journal of Finance* 48, 157–185.

Capponi, Agostino, and Ruizhe Jia, 2021, The adoption of blockchain-based decentralized exchanges, working paper Columbia University https://ssrn.com/abstract=3805095.

Daian, Philip, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels, 2019, Flash Boys 2.0: Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges, Working paper Cornell Tech.

Glosten, Lawrence R., 1994, Is the electronic open limit order book inevitable?, *The Journal of Finance* 49, 1127–1161.

Ho, Thomas, and Hans Stoll, 1981, Optimal dealer pricing under transactions and return uncertainty, *Journal of Financial Economics* 9, 47–73.

KyberNetwork, 2019, Kyber: An on-chain liquidity protoco, Whitepaper Kyber Network Foundation https://files.kyber.network/Kyber_Protocol_22_April_v0.1.pdf.

Kyle, Albert S., 1985, Continuous auctions and insider trading, *Econometrica* 53, 1315–1336.

Lehar, Alfred, and Christine Parlour, 2021, Decentralized exchanges, Working paper University of Calgary link.

Malinova, Katya, Andreas Park, and Ryan Riordan, 2013, Do retail traders suffer from high frequency traders?, Working paper University of Toronto.

Martinelli, Fernando, and Nikolai Mushegian, 2019, A non-custodial portfolio manager, liquidity provider, and price sensor, Whitepaper Balancer Foundation https://balancer.finance/whitepaper/.

van Kervel, Vincent, Amy Kwan, and P. Joakim Westerholm, 2020, Order splitting and interacting with a counterparty, Working paper Pontificia Universidad Católica de Chile https://ssrn.com/abstract=3516199.

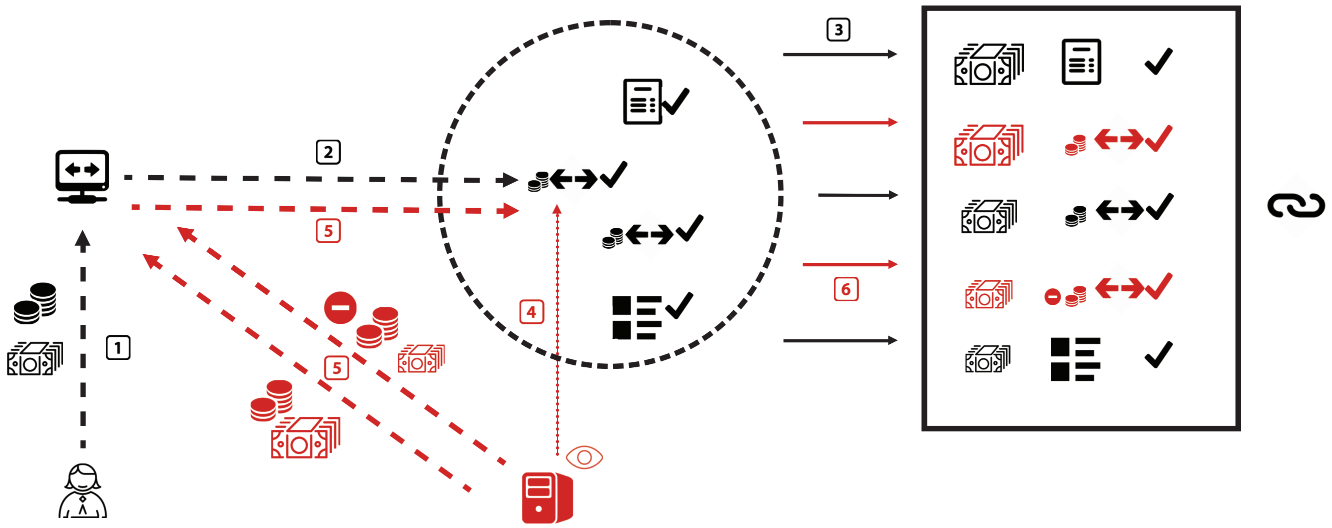


Figure 1
Illustration of Front-Running in Blockchains

The possibility of front-running is an intrinsic feature of blockchains. The schematic works as follows. A user who wants to perform a swap transaction submits the tokens she desires to exchange to the constant product market making contract (1). The contract submits an atomic swap to the blockchain network, and upon verification this transactions enters the mem-pool (2). Verified transactions get ordered in a block based on the fees that they offer (all else equal) (3). An attacker (likely a bot) observes the mempool and see a transaction that can be front-run profitably (4). The bot sends two off-setting swap transactions to the contract, when the front-running trade has a higher fee than the original, front-run transaction (5). In the block, the transactions now get re-ordered according to their mining fees and, upon inclusion on the chain, the transactions in the CPMM contract get executed in the order of the fees (6).

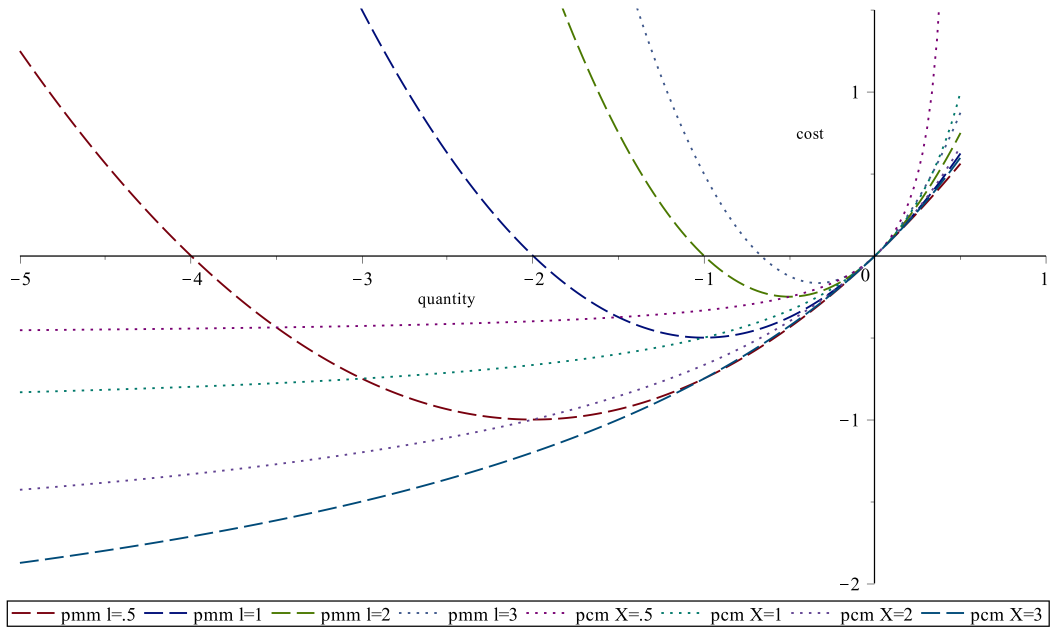
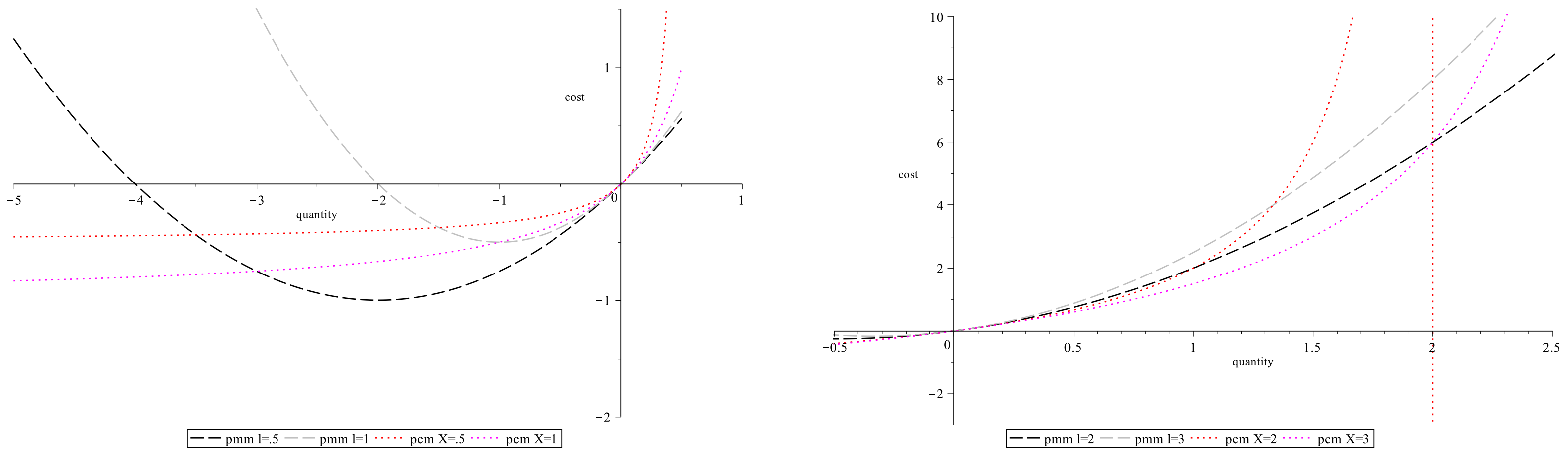


Figure 2
Comparison of the Cost Functions p^{tmm} and p^{cmm}

The figure plots the two cost functions, CPMM (dotted) and traditional (dashed) for parameters $\ell \in \{.5, 1, 2, 3\}$ and target inventory $X \in \{.5, 1, 2, 3\}$ as a function of the trade size x. The curves intersect at value $x^* = X/\ell(\ell - 2V)$, where I set $V = 1$, and they touch at $x = 0$. By construction, the dashed curve for traditional pricing reflects fair compensation for liquidity provision. Therefore, under CPMM pricing, intermediaries are over-compensated if for positive values of x the dotted curve is above the corresponding dashed curve or for negative values of x when the dotted curve is below the dashed curve.



Panel A: $x^* < 0$ Panel B: $x^* > 0$

Figure 4
Comparison of the Cost Functions for $x^* < 0$ and $x^* > 0$

The two panels plot the sames curves as Figure 2, except for sub-cases when $x^* < 0$ (Panel A) and $x^* > 0$, for greater clarity. Under CPMM pricing, intermediaries are over-compensated if for positive values of x the dotted curve is above the corresponding dashed curve or for negative values of x when the dotted curve is below the dashed curve.

Table I
Calibration of Costs and No-Arbitrage Fees

	trade size	USDC-USDT	ETH-USD	BTC-USD
value V		1	1.2K	36K
σ		0.01	220	6.6K
X		17M	80K	120
X in USD		17M	100M	4.3M
x^*		$-3.4e^{15}$	-39.6M	-1.9K
excess cost	1K	0.1	0.0	0.3
CPMM	5K	1.5	0.4	6.4
	10K	5.9	1.5	25.4
arbitrage	1K	0.1	0.0	0.3
prevention	5K	1.5	0.4	6.8
fee	10K	5.9	1.5	27.2
total in bps	1K	0.1	0.0	0.5
of transaction	5K	0.7	1.5	13.1
value	10K	11.8	3.0	52.6