# Improved Price Oracles: Constant Function Market Makers

Guillermo Angeris
angeris@stanford.edu

Tarun Chitra
tarun@gauntlet.network

March 2020

**Abstract**

Automated market makers, first popularized by Hanson's Logarithmic Market Scoring Rule for prediction markets, have become important building blocks (often called 'primitives') for decentralized finance. A particularly useful primitive is the ability to measure the price of an asset, a problem often known as the pricing oracle problem. In this paper, we focus on the analysis of a very large class of automated market makers, called constant function market makers, which includes popular market makers such as Uniswap and Balancer. We give sufficient conditions such that, under fairly general assumptions, agents who interact with these constant function market makers are incentivized to correctly report the price of an asset. We also derive several other useful properties including liquidity provider returns in the path independent case.

## 1 Introduction

As natively digital assets continue to proliferate, there is an increasing need for mechanisms of exchange similar to those found in traditional financial markets. These digital assets, which include online ad impressions, cryptocurrencies, and prediction market bets, are often complex to interact with and suffer from low liquidity [33]. Over the last decade, a number of designs for automated market makers (AMMs) have been proposed to alleviate this complexity. AMMs encourage passive market participants with low time preference to lend their digital assets to asset pools. The assets are then priced via a *scoring rule* which maps the current pool sizes to a marginal price. One of the most popular scoring rules is Hanson's *logarithmic market scoring rule* (LMSR) [21]. This rule has been implemented in numerous online settings including online ad auctions [20], prediction markets [6, 34], and instructor rating markets [9].

Recently, the cryptocurrency community has constructed alternative automated market makers to the LMSR (and its counterparts), known as the *constant function market makers* (CFMMs). Examples of CFMMs include Uniswap's constant product AMM [48] and the constant mean AMM [29]. Most applications of CFMMs have been to construct *decentralized exchanges*, which allow for the exchange of security-like assets without the need for a trusted third-party. One main concern for users of these markets is whether prices on decentralized exchanges accurately follow those on centralized exchanges, which currently have more liquidity. If the price of a decentralized exchange prices matches external prices, then such an exchange is said to be a good *price oracle* that other smart contracts can query as a source of ground truth.

**The oracle problem.** Of the aforementioned AMM use-cases, the most studied and controversial applications of markets implementing the LMSR are prediction markets. As prediction markets have repeatedly been shut down or discontinued due to intervention by governments, large-scale stress testing of these mechanisms has been limited [36]. The advent of smart contract systems, such as Ethereum [47] and Tezos [19], has allowed for the design and implementation of decentralized and censorship resistant versions of prediction markets. In particular, Augur [35] and Gnosis [6] are smart contracts that implement LMSR-AMMs that are deployed to Ethereum's live network. This means that anyone who owns Ethereum's native asset (ETH) can participate in these prediction markets and bet without the risk of government intervention and asset seizure (*e.g.*, akin to 'Black Friday' within the online poker community [40]).

However, the main difficulty in using these systems in the decentralized setting is the ability to query data external to the smart contract. For example, a prediction market betting on the future weather in Seattle would need to know the resulting weather report, once the event has happened. Doing this in a trustless manner is difficult, as participants who have a losing bet are encouraged to try to dupe the smart contract, *i.e.*, to manipulate the response of the contract's query. In our previous example, if the prediction market bets on the question "will the weather in Seattle be greater than 25°C?" then a malicious participant with an active bet on "no" is incentivized to manipulate the query response to say that the temperature is $< 25$°C. In the cryptocurrency community, the problem of providing external data to a blockchain is known as the *oracle problem*, as an homage to oracles queried in theoretical computer science [27].

**Decentralized oracles.** Formally, an *oracle* refers to any computational device that provides the smart contract data external to the underlying blockchain [35]. There are two types of oracles in smart contract prediction markets: centralized and decentralized. Centralized oracles involve a trusted individual or organization that provides data to the smart contract. Examples of centralized oracles include Provable/Oraclize [37], Chainlink [30, 8], Wolfram Alpha [46], and the MakerDAO oracle [44]. If these oracles are used to decide on LMSR prediction market events, they still rely on participants trusting that the centralized authority will not manipulate the data determining the final outcome of the market. For highly valuable markets, such as the prediction of the US presidential election, it is usually untenable to trust a single individual or organization and one defers to *decentralized oracles*.

Decentralized oracles are smart contracts that rely on users voting on particular prediction market outcomes. A final outcome is chosen via a social choice function [32], similar to how majority or weighted majority voting is used to decide outcomes in elections. In the case of smart contracts, the social choice function is usually significantly more complicated as the voting mechanism needs to account for adverse selection, bribery, and collusion amongst voters. In order to reduce the likelihood of such *oracle manipulation*, decentralized oracle designs often have exit games and/or complicated multiparty games that allow for certain users to challenge votes that they dispute. Moreover, to encourage a large swath of potential users to participate in a vote, prediction market smart contracts usually provide users with a reward. This reward is disbursed in a manner similar to how cryptocurrency rewards are given to miners and/or validators [10]. These oracles are difficult to design as one has to balance mechanism complexity with provable defenses against collusion between prediction market participants. Examples of decentralized oracles include Augur, Astraea [3], Gnosis, and UMA [28]. We note that a stated design goal of Augur is for the prices implied by the LMSR to be used as an oracle input into *other* smart contracts. For instance, if another smart contract relies on the probability of whether Seattle's temperature is greater or less than 25°C, then that contract simply has to subscribe to the data and pricing provided by an Augur market. In this way, prediction market smart contracts aim to serve as the single source of off-chain data that is accessible to an arbitrarily large number of on-chain smart contracts.

**Decentralized exchanges.** Decentralized exchanges (DEXs) provide a method for participants to trade pairs of on-chain assets without ever needing to trust a centralized authority [45, 24], while additionally providing a means for measuring the relative price of this pair of assets (by simply reporting the price of the last trade). Currently, there are roughly $100 million of digital assets locked in DEXs with daily trading volumes often surpassing $10 million per day [38, 15]. A design for a secure decentralized exchange for cryptocurrencies has been desired almost since the advent of Bitcoin, since centralized exchanges such as Mt. Gox [14], Quadriga [42], and Bitfinex [25], have had catastrophic losses that aggregate to billions of dollars of depositors' funds.

Many different decentralized exchanges have been proposed using different mechanisms, ranging from classic order book mechanisms [45] to other, more complicated cases [22]. Yet, Uniswap [48, 2], an AMM whose pricing mechanism is relatively simple in both theory and practice [5], has become very popular as can be seen from the large trading volumes within these markets and total funds in their reserves [15]. This has led protocols such as Celo [26] to use the Uniswap mechanism as a price oracle, in order to measure the relative price of two assets.

**Generalizations of Uniswap.** The success of Uniswap, which required far fewer resources than competing decentralized exchanges,[1] has led to a number of generalizations. A natural

[1]Uniswap was created with a $100,000 grant from the Ethereum Foundation [2], whereas 0x [23] and Bancor raised $24 million and $153 million in Initial Coin Offerings. Uniswap currently has roughly 5-10 times the liquidity of 0x and Bancor [13] combined, and both 0x and Bancor now route orders to Uniswap [11].

| | | | | | |
|---|---|---|---|---|---|
| 1. | Maker | Ethereum | Lending | $704.9M | 11.5% |
| 2. | Compound | Ethereum | Lending | $143.4M | 8.5% |
| 3. | Synthetix | Ethereum | Derivatives | $134.8M | -4.1% |
| 4. | InstaDApp | Ethereum | Lending | $92.0M | 15.1% |
| 5. | Uniswap | Ethereum | DEXes | $67.7M | 9.3% |
| 6. | dYdX | Ethereum | Lending | $23.1M | 13.8% |
| 7. | bZx | Ethereum | Lending | $13.4M | 1.1% |
| 8. | Bancor | Ethereum | DEXes | $12.7M | 8.0% |
| 9. | Lightning Network | Bitcoin | Payments | $9.1M | 1.9% |
| 10. | WBTC | Ethereum | Assets | $7.2M | 0.9% |
| 11. | Set Protocol | Ethereum | Assets | $5.4M | 13.1% |
| 12. | Kyber | Ethereum | DEXes | $4.8M | 5.4% |
| 13. | Nuo Network | Ethereum | Lending | $4.3M | 5.5% |
| 14. | DDEX | Ethereum | Lending | $3.6M | 28.9% |
| 15. | Nexus Mutual | Ethereum | Derivatives | $3.4M | 12.7% |
| 16. | Dharma | Ethereum | Lending | $1.3M | -2.2% |
| 17. | Augur | Ethereum | Derivatives | $686.0K | 11.3% |
| 18. | Melon | Ethereum | Assets | $348.7K | 7.9% |
| 19. | xDai | Ethereum | Payments | $38.1K | -1.6% |
| 20. | Veil | Ethereum | Derivatives | $34.1K | 12.7% |

Figure 1: Market Sizes of protocols utilizing Constant Function Market Makers (in red) and LMSR-based AMMs (in blue) on February 12, 2020. There are $129.7 million of digital assets held by CFMM contracts and $0.721 million of assets held by LMSR-based market makers. Data taken from DeFi Pulse [38].

first question to ask is whether the Uniswap constant product formula is optimal for all types of assets. For instance, assets that have a common mean price but different volatilities can have too steep of a loss with the Uniswap bonding curve.

An example of such assets are *stablecoins*, which are digital assets whose value is (approximately) pegged to be equal to \$1. The price of these assets naturally fluctuate around \$1, but tend to stay within a bounded range of $[1-\epsilon, 1+\epsilon]$ for some $\epsilon > 0$. The fluctuations of these assets is dictated by their natural sources of demand and can vary greatly, even though these digital assets are all meant to represent the same real world asset. For instance, a stablecoin that is popular in Venezuela will likely have different demand characteristics than one that is popular in China. In order to incentivize traders, the curve should instead charge lower fees when two stablecoins are near \$1 and higher fees when the stablecoins are farther from \$1. This approach has been implemented in Curve/StableSwap [17], which uses a scoring rule whose curvature is flatter around \$1 and parabolic as coins trade away from one another (see Figure 2).

**Multi-coin generalizations.** Another generalization of the Uniswap scoring rule involves pricing multiple assets simultaneously. Instead of providing a scoring rule that is a function of the quantities of two assets, these scoring rules take are able to price $m$ assets in terms of a set of $n$ other assets. This allows for users to exchange portfolios of assets for other portfolios, reducing the number of transactions that the network has to handle. On an exchange that only allows for pairwise trades, a participant would need to do $m$ trades to a numéraire asset (*e.g.*, Bitcoin or USD) and then $n$ trades from the numéraire to the output assets. Multi-asset generalizations of Uniswap, such as Balancer [29], would execute such a trade atomically, reducing fees and price slippage. The choice of scoring rule affects how easy it is for arbitrageurs to keep the portfolio prices synchronized with the prices of the underlying components.

While this mechanism might seem arbitrary, there are a number of examples of similar assets from traditional finance that involve trading baskets of good for other baskets of goods. For instance, an Exchange Traded Fund (ETF) is a single equity instrument $S_E$ that represents a weighted set of shares $S_1, \ldots, S_n$. There are currently over \$5 trillion of assets locked in ETFs [31]. Most ETFs represent a share of $S_E$ bia weighted linear combination of the shares, $S_E = \sum_{i=1}^n w_i S_i$ for some positive integer weights $w_i$. If any of the prices of shares $S_i$ change, leaving $S_E$ mispriced, an arbitrageur can perform creation-redemption arbitrage [18]. This arbitrage works due to two steps:

- Creation: A market participant can create a single share $S_E$ by providing the ETF underwriter with $w_i$ shares of $S_i$ for all $i$
- Redemption: A market participant can redeem a single share $S_E$ by giving the ETF underwriter $S_E$ and receiving $\{w_i S_i : i \in [n]\}$.

If the price of $S_E$ is higher than the weighted sum of the prices of $S_i$, then an arbitrageur can buy the basket $\{w_i S_i : i \in [n]\}$ for less than price$(S_E)$, create a share of $S_E$ and sell it for a profit of price$(S_E) - \sum_{i=1}^n w_i \cdot$ price$(S_i)$. Similarly, the arbitrageur can use a redemption to

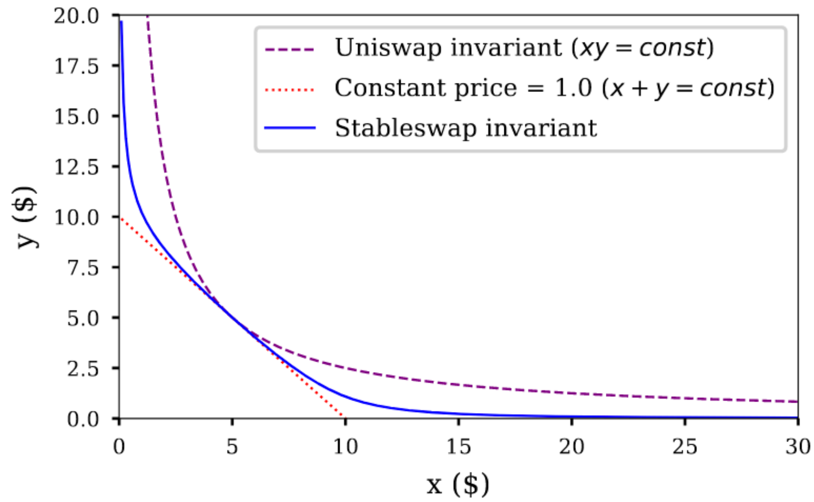


Figure 2: A comparison of the scoring rules for StableSwap / Curve and Uniswap. The price of asset $x$ in terms of asset $y$ can be viewed as the slope of the curve. The StableSwap curve is flatter when the price is near \$1 (dotted line), which allows for Stablecoins to be traded cheaply when they are near their mean. Figure taken from [16].

arbitrage a low $S_E$ price. Multi-asset generalizations of Uniswap allow for precisely this type of arbitrage to occur without the need for a trusted intermediary (*e.g.*, the ETF underwriter).

**Summary.** In this paper, we show that many of the above generalizations are special cases of a large family of *constant function market makers*, or CFMMs, which all satisfy relatively similar and useful theoretical properties, under mild conditions.

In §2, we give a mathematically simple, but relatively complete characterization of the CFMMs in the two-coin case, while in §3 we give a complete, but somewhat more mathematical generalization to the $n$ coin case, leading to a nearly complete characterization of all constant function market makers that appear to be useful in practice. In particular, the properties proved for CFMMs expand upon those given in [5] and we provide necessary and sufficient conditions for a larger class of scoring rules to be valid market makers by satisfying many of these properties. The generalizations provided here take advantage of convexity properties of a scoring rule and are similar in spirit to axiomatic formulations of LMSRs via optimization problems, such as [1]. These generalizations also leads to simple expressions for liquidity provider returns in the path independent CFMM case.

# 2 Constant function market makers

A two-coin *constant function market maker* (CFMM) is a market maker defined by a *trading function* $\varphi : \mathbf{R}_+ \times \mathbf{R}_+ \times \mathbf{R} \times \mathbf{R} \to \mathbf{R}$ whose current state is defined by its coin *reserves*

$R_\alpha, R_\beta \in \mathbf{R}_+$, which are simply reserves of some pair of coins available to the market contract. Agents who supply coins to the reserves $R_\alpha, R_\beta$ are known as *liquidity providers*.

A trader can exchange $\Delta_\alpha$ of coin $\alpha$ for $\Delta_\beta$ of coin $\beta$ only when the chosen coin amounts satisfy

$$\varphi(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) = \varphi(R_\alpha, R_\beta, 0, 0),$$

in which case we say a trade is *feasible*. In practice, every time a feasible trade is executed, a fraction of the trade quantities $\Delta_\alpha, \Delta_\beta$ is taken as a fee and paid to liquidity providers. Most practical models of transaction fees can be absorbed into the definition of $\varphi$, which generalizes the formulation of [5]. (In §4.1 we show a simple method for introducing fees to any given CFMM.)

If the trade is indeed feasible, the market maker then updates its reserves in the following way: $R_\alpha \leftarrow R_\alpha + \Delta_\alpha$ and $R_\beta \leftarrow R_\beta + \Delta_\beta$ and takes $\Delta_\alpha$ of coin $\alpha$ and $\Delta_\beta$ of coin $\beta$ from the trader (negative values for these quantities indicate that the AMM gives coin to the trader, instead). In other words, the market maker only accepts trades which keep the trading function $\varphi$ constant at the current reserve values. Additionally, for notational convenience, we will write $\varphi_R = \varphi(R_\alpha, R_\beta, \cdot, \cdot)$ when the reserves $R = (R_\alpha, R_\beta) \in \mathbf{R}^2_+$ are taken to be arbitrary.

In order to simplify the exposition, we will only consider trades from $\alpha$ to $\beta$ (*i.e.*, we will assume that the traded amount $\Delta_\alpha \geq 0$ while $\Delta_\beta \leq 0$), though the results hold generally (by swapping $\alpha$ with $\beta$ along with its corresponding reserves). We will also assume that $\varphi_R$ is continuously differentiable for each $R$ and that it has nonzero derivative everywhere for all components, though there is a generalization to the nondifferentiable case via convex analysis. We present this more general case (along with an $n$ coin generalization) in §3, though we note that the generalization requires much more involved mathematical machinery.

While apparently very strong, many of the CFMMs currently found in practice satisfy the assumptions presented in this section. For example, in the specific case of constant product markets, we have that

$$\varphi(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) = R_\alpha R_\beta - (R_\alpha + \Delta_\alpha)(R_\beta + \Delta_\beta), \tag{1}$$

which we will use as our canonical example, throughout.

**Equivalence of CFMMs.** For the remainder of the section, we will say that two CFMMs are *equivalent* whenever they generate the same set of feasible trades. In other words, two CFMMs with trading functions $\varphi$ and $\varphi'$, respectively, are equivalent if, for any $(\Delta_\alpha, \Delta_\beta)$ and $R \in \mathbf{R}^2_+$, we have

$$\varphi_R(\Delta_\alpha, \Delta_\beta) = \varphi_R(0, 0) \quad \text{if, and only if,} \quad \varphi'_R(\Delta_\alpha, \Delta_\beta) = \varphi'_R(0, 0). \tag{2}$$

This notion of equivalence comes from the fact that there are many functions $\varphi$ which have the same feasible trades. One simple example is to take any invertible function $\zeta : \mathbf{R} \to \mathbf{R}$ and note that a (new) CFMM with trading function $\zeta \circ \varphi$ is equivalent in the above sense to the original CFMM, even though their trading functions may not be equal.

## 2.1 Basic requirements

We will first state some basic assumptions of the trade function $\varphi$.

**No free trades.** We will assume that there exist no trades such that

$$\varphi_R(0, \Delta_\beta) = \varphi_R(0, 0),$$

when $\Delta_\beta < 0$ (*i.e.*, an agent cannot freely remove coin from the reserves without paying). This additionally lets us choose a 'direction' for our function, since this implies that either $\varphi_R(0, \Delta_\beta) > 0$ or $\varphi_R(0, \Delta_\beta) < 0$ for all $\Delta_\beta > 0$, by the continuity of $\varphi_R$.

For the remainder of this section, we will assume that $\varphi_R(0, \Delta_\beta) > 0$ for any $\Delta_\beta < 0$ (otherwise, we can simply exchange $\varphi_R$ for $-\varphi_R$).

**Existence of a feasible trade.** We will assume that, for each possible strictly positive reserve amount, $R \in \mathbf{R}^2_{++}$, there exists a feasible trade. In other words, there exists some pair $\Delta_\beta < 0$ and $\Delta_\alpha > 0$ satisfying

$$\varphi_R(\Delta_\alpha, \Delta_\beta) = \varphi_R(0, 0).$$

This assumption prevents 'trivial' functions $\varphi_R$ which do not allow trading, even though reserves are available. This condition can be weakened to hold only on some set $\mathcal{R} \subseteq \mathbf{R}^2_{++}$ with the obvious replacements (requiring $R \in \mathcal{R}$) in the conditions below. For example, in the case of (1), we can assume that $\mathcal{R} = \{(R_\alpha, R_\beta) \in \mathbf{R}^2_{++} \mid R_\alpha R_\beta = k\}$ for some constant $k > 0$, since, if a constant product market begins with reserves in the set $\mathcal{R}$, by definition, every feasible trade will keep the reserves in $\mathcal{R}$.

## 2.2 Properties

Using the above, we can prove some simple properties of the trading function $\varphi$.

**Zero point.** We can always assume that the function $\varphi_R(0, 0) = 0$, without any loss of generality, since we can always take the equivalent CFMM with trading function

$$\tilde{\varphi}_R(\Delta_\alpha, \Delta_\beta) = \varphi_R(\Delta_\alpha, \Delta_\beta) - \varphi_R(0, 0).$$

We will therefore assume that $\varphi_R(0, 0) = 0$, throughout.

**Monotonicity of the trading function.** We can show that the trading function $\varphi$, given the requirements above, is monotonically decreasing in $\Delta_\alpha$ and $\Delta_\beta$.

Due to the "no free trades" property, we know that $\varphi_R(0, \Delta'_\beta) > 0$ for every $\Delta'_\beta < 0$. This implies that, since there exists a feasible nonzero trade $(\Delta_\alpha, \Delta_\beta)$, by assumption, we must have that $\varphi_R(0, \Delta_\beta) > 0 = \varphi_R(\Delta_\alpha, \Delta_\beta)$, so the function $\varphi_R(\cdot, \Delta_\beta)$ must be decreasing at some point, by the mean value theorem. Since the derivative is nonzero everywhere, the function $\varphi_R(\cdot, \Delta_\beta)$ must therefore be decreasing for every $\Delta_\beta \le 0$, as the derivative is continuous.

Because the function $\varphi_R(\cdot, \Delta'_\beta)$ is decreasing for every $\Delta'_\beta \le 0$, then for a feasible trade $(\Delta_\alpha, \Delta_\beta)$ with $\Delta_\alpha > 0$ and $\Delta_\beta < 0$, we must have $\varphi_R(\Delta_\alpha, 0) < 0$. By a similar argument, since $\varphi_R(\Delta_\alpha, \Delta_\beta) = 0$, we must therefore have $\varphi_R(\Delta_\alpha, \cdot)$ also decreasing (recall that $\Delta_\beta < 0$). Combining this with the above implies the function $\varphi_R$ is decreasing, componentwise.

## 2.3 Definitions

In this section, we will define a few properties of CFMMs which will be used later to prove some important results.

### 2.3.1 Path independence

In a similar way to classical AMMs (such as LMSRs), we might ask if there a notion of path independence for the function $\varphi$.

A natural way of phrasing this property in the context of CFMMs is to say that an AMM is *path independent* if it satisfies the following property: given a feasible trade $(\Delta_\alpha, \Delta_\beta)$ then an aggregate trade $(\Delta_\alpha + \Delta'_\alpha, \Delta_\beta + \Delta'_\beta)$ is feasible if and only if sequentially trading $(\Delta_\alpha, \Delta_\beta)$ and $(\Delta'_\alpha, \Delta'_\beta)$ is also feasible. In other words, given a trade $(\Delta_\alpha, \Delta_\beta)$ satisfying

$$\varphi(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) = 0$$

then

$$\varphi(R_\alpha, R_\beta, \Delta_\alpha + \Delta'_\alpha, \Delta_\beta + \Delta'_\beta) = 0 \quad \text{if, and only if,} \quad \varphi(R_\alpha + \Delta_\alpha, R_\beta + \Delta_\beta, \Delta'_\alpha, \Delta'_\beta) = 0. \tag{3}$$

Note that path independence is a *very* strong condition for CFMMs, unlike in classical AMMs, where it is almost universally accepted as a requirement. In particular, a CFMM with initial reserves $R^0$ is path independent if, and only if, it can be written in the following form:

$$\varphi(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) = \psi(R_\alpha + \Delta_\alpha, R_\beta + \Delta_\beta),$$

for some (decreasing) function $\psi$ (which may depend on $R^0$). Said another way, if the CFMM given by $\varphi$ is path independent, then there exists an equivalent CFMM whose trading function depends only on the final state of the reserves at each trade. This function $\psi$ is qualitatively similar to the existence of a potential or cost function under the path independence assumption in the traditional automated market making literature [1, §3.2.1].

To show this, let $(\Delta^i_\alpha, \Delta^i_\beta)$ be the accepted transactions for $i = 1, \dots, n$ and $\varphi$ be a path independent trading function, then, by definition of a CFMM, we have

$$R_\alpha = R^0_\alpha + \sum_{i=1}^n \Delta^i_\alpha, \quad R_\beta = R^0_\beta + \sum_{i=1}^n \Delta^i_\beta. \tag{4}$$

We can rewrite

$$\varphi(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) = 0, \quad \text{if and only if,} \quad \varphi\left(R_\alpha^0, R_\beta^0, \Delta_\alpha + \sum_{i=1}^n \Delta_\alpha^i, \Delta_\beta + \sum_{i=1}^n \Delta_\beta^i\right) = 0,$$

which follows from (4) and applying (3) inductively on the valid transactions $(\Delta_\alpha^i, \Delta_\beta^i)$ starting from $i = n$ down to $i = 1$. Using (4) once more, we can then write

$$\varphi\left(R_\alpha^0, R_\beta^0, \Delta_\alpha + \sum_{i=1}^n \Delta_\alpha^i, \Delta_\beta + \sum_{i=1}^n \Delta_\beta^i\right) = \varphi(R_\alpha^0, R_\beta^0, \Delta_\alpha + R_\alpha - R_\alpha^0, \Delta_\beta + R_\beta - R_\beta^0),$$

and setting

$$\psi(R_\alpha + \Delta_\alpha, R_\beta + \Delta_\beta) = \varphi(R_\alpha^0, R_\beta^0, \Delta_\alpha + R_\alpha - R_\alpha^0, \Delta_\beta + R_\beta - R_\beta^0),$$

finishes the claim. The converse can be easily verified.

It is enlightening (and not very difficult) to show that, for constant product CFMMs (1), this $\psi$, as constructed above, is the constant product formula given in [5, §2], where the constant $k = R_\alpha^0 R_\beta^0$ and $\gamma = 1$, *i.e.*, where $k$ is the product of the initial reserve amounts and with no percentage fees. (This differs from the original definition in (1), where we are instead using the product of the *current* reserve amounts, $R_\alpha R_\beta$.)

**Path deficiency.** In fact, in many cases, path independence may not be a desirable property for CFMMs. For example, assuming that a CFMM pays liquidity providers as a proportion of its pool, then a 'good' CFMM would incentivize liquidity providers to not withdraw their position. A way of doing this would be to charge a simple percentage fee for each trade, yet doing this within a contract could prevent path independence (for a simple example, see [5, §2.3]).

In light of this, we can give a relaxation of the path independence property. A CFMM is *path deficient* if, for any valid trade $(\Delta_\alpha, \Delta_\beta)$, we have that, for any (potentially infeasible) trade $(\Delta'_\alpha, \Delta'_\beta)$,

$$\varphi(R_\alpha, R_\beta, \Delta_\alpha + \Delta'_\alpha, \Delta_\beta + \Delta'_\beta) = 0, \quad \text{implies} \quad \varphi(R_\alpha + \Delta_\alpha, R_\beta + \Delta_\beta, \Delta'_\alpha, \Delta'_\beta) \geq 0,$$

and

$$\varphi(R_\alpha + \Delta_\alpha, R_\beta + \Delta_\beta, \Delta'_\alpha, \Delta'_\beta) = 0, \quad \text{implies} \quad \varphi(R_\alpha, R_\beta, \Delta_\alpha + \Delta'_\alpha, \Delta_\beta + \Delta'_\beta) \leq 0.$$

We will say that a CFMM is *strictly path deficient* if the inequalities hold strictly for nonzero trades. Intuitively, the first condition says that, if an aggregate trade is feasible, then splitting it up into two trades can come up short (this follows from the fact that $\varphi$ is decreasing), while the second condition says that if a split trade is feasible, then aggregating it into a single trade implies that the agent has potentially overpaid. Note that a path independent CFMM is also path deficient (but not strictly so) given our definition.

### 2.3.2 Marginal price

For a given trading function $\varphi$, the expression for the marginal price is very simple. Given reserves $R_\alpha, R_\beta$ and a feasible trade $\Delta_\alpha$, $\Delta_\beta$, we can compute the marginal price of buying a slightly larger quantity of $\Delta_\beta$ (*e.g.*, $-d\Delta_\beta/d\Delta_\alpha$, as $\Delta_\beta$ is negative) by implicitly differentiating the trading conditions:

$$0 = \frac{d\Delta_\beta}{d\Delta_\alpha}\frac{\partial \varphi_R(\Delta_\alpha, \Delta_\beta)}{\partial \Delta_\beta} + \frac{\partial \varphi_R(\Delta_\alpha, \Delta_\beta)}{\partial \Delta_\alpha}.$$

This gives

$$m_a(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) = -\frac{d\Delta_\beta}{d\Delta_\alpha} = \frac{\frac{\partial \varphi_R(\Delta_\alpha, \Delta_\beta)}{\partial \Delta_\alpha}}{\frac{\partial \varphi_R(\Delta_\alpha, \Delta_\beta)}{\partial \Delta_\beta}}. \tag{5}$$

Correctness of (5) follows since $\varphi_R$ has partial derivatives which are never zero, so the implicit function theorem holds [43].

We show in §2.4.2 that, given a marginal price function $m_a$, we can always construct a CFMM whose marginal price is equal to $m_a$, and that this CFMM is unique up to equivalence in the sense of (2).

**Arbitrary price differences.** A useful condition of a CFMM is that it should be able to achieve arbitrary price jumps, *i.e.*, that for any reserves there exists a feasible trade that can set the current marginal price to a desired amount.

A sufficient condition for this to happen is to require

$$\mathbf{R}_{++} \subseteq \{m_a(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) \mid \varphi(R_\alpha, R_\beta, \Delta_\alpha, \Delta_\beta) = 0\},$$

for any given reserves $R_\alpha, R_\beta \in \mathbf{R}_{++}$. We will assume this throughout our presentation.

## 2.4 Implications

### 2.4.1 Marginal price after trade

If the CFMM is path deficient, an arbitrageur can perform arbitrage in one trade and this action is optimal. This follows from §2.3.1, since a path deficient CFMM has the property that trades are no cheaper when subdivided into several steps. If the CFMM is strictly path deficient, then subdividing a trade into more than one step is always suboptimal.

In this case, given an infinitely liquid reference market (*e.g.*, when an agent is free to trade $\Delta_\alpha$ coins for $\Delta_\beta$ coins and $\Delta_\alpha = m_p\Delta_\beta$ for some market price $m_p > 0$), an arbitrageur seeks to solve a the problem (*cf.*, [5, §2.1]),

$$\begin{array}{ll} \text{maximize} & \Delta_\beta - m_p\Delta_\alpha \\ \text{subject to} & \varphi_R(\Delta_\alpha, -\Delta_\beta) = 0. \end{array} \tag{6}$$

Note that, since the function $\varphi_R$ is decreasing with respect to $\Delta_\alpha$ and increasing with respect to $-\Delta_\beta$, it has an implicit function $\xi : \mathbf{R} \to \mathbf{R}$ which is also monotonic, in the sense that $\varphi(\Delta_\alpha, -\xi(\Delta_\alpha)) = 0$ and $\xi$ is increasing. An arbitrageur can easily solve the resulting problem,

$$\text{maximize} \quad \xi(\Delta_\alpha) - m_p \Delta_\alpha$$

in polynomial time, as it is a single parameter problem.

The resulting marginal price (right before the trade is performed) can be derived from the optimality conditions of (6), whose Lagrangian $\mathcal{L}$ is [7, §5.1.1],

$$\mathcal{L}(\Delta_\alpha, \Delta_\beta) = \Delta_\beta - m_p \Delta_\alpha + \lambda \varphi_R(\Delta_\alpha, -\Delta_\beta).$$

The conditions at an optimal point $(\Delta_\alpha^\star, \Delta_\beta^\star)$ are given by

$$\begin{aligned} \lambda \partial_{\Delta_\alpha} \varphi_R(\Delta_\alpha^\star, -\Delta_\beta^\star) &= m_p \\ \lambda \partial_{\Delta_\beta} \varphi_R(\Delta_\alpha^\star, -\Delta_\beta^\star) &= 1, \end{aligned}$$

and dividing both expressions (assuming $\lambda \neq 0$, which can be shown by monotonicity and regularity) yields that the market price $m_p$ is the same as the resulting marginal price $m_a$ before the trade is performed $(\Delta_\alpha^\star, \Delta_\beta^\star)$.[2] If the marginal price after the trade is the same as the marginal price right before the trade is performed, an arbitrageur then gets the highest payoff when correctly setting the CFMM's price relative to a reference market. Since the CFMM is further assumed to be path deficient, then the arbitrageur does not make any profit by subdividing their trade, implying that this is an optimal action.

### 2.4.2 Generating a CFMM given a marginal price

First, we will assume that we are given a positive marginal price function $p_R : \mathbf{R}_+ \to \mathbf{R}_{++}$ which, for fixed reserves $R \in \mathbf{R}_+^2$, maps the amount of given coin $\Delta_\beta$ to its marginal price. In other words, to *purchase* some amount of coin $-\Delta_\beta > 0$ (keeping with our notation that negative amounts are those removed from the reserves), we will require $\Delta_\alpha$ amount put into reserves, such that

$$\Delta_\alpha = \int_0^{-\Delta_\beta} p_R(\Delta_\beta') \, d\Delta_\beta', \tag{7}$$

is satisfied.

In this case, finding a decreasing trading function $\varphi_R$ such that (7) is satisfied if, and only if $\varphi_R(\Delta_\alpha, \Delta_\beta)$ is zero, is nearly trivial since the following function suffices:

$$\varphi_R(\Delta_\alpha, \Delta_\beta) = \left( \int_0^{-\Delta_\beta} p_R(\Delta_\beta') \, d\Delta_\beta' \right) - \Delta_\alpha. \tag{8}$$

Additionally, note that $\varphi_R$ is also decreasing with respect to both of its arguments, as required.

[2] This also gives a new way of solving the problem: namely changing the reserves in such a way that the new marginal price is the same as the market price.

**Marginal price functions.** While we have made an active distinction between the marginal price function $p_R$ as given in (7) and the marginal price function $m_a$ for a trading function $\varphi$, given in (5), the two are easily related. For fixed reserves $R = (R_\alpha, R_\beta) \in \mathbf{R}^2_+$, there exists an implicit function $\xi : \mathbf{R} \to \mathbf{R}$ such that

$$\varphi_R(\xi(\Delta_\beta), \Delta_\beta) = 0,$$

for any $\Delta_\beta \leq 0$, by monotonicity. Therefore, for this CFMM with trading function $\varphi$, the marginal price function $p_R$ is equal to

$$p_R(-\Delta_\beta) = m_a(R_\alpha, R_\beta, \xi(\Delta_\beta), \Delta_\beta).$$

**Constant product markets.** As a simple example, we can recover the constant product CFMM directly by using (8). In the case of constant product CFMMs, it is not hard to show that

$$p_R(\Delta_\beta) = \frac{R_\alpha R_\beta}{(R_\beta - \Delta_\beta)^2},$$

for some reserves $R_\alpha, R_\beta$. Using (8), then

$$\varphi_R(\Delta_\alpha, \Delta_\beta) = \left(\int_0^{-\Delta_\beta} \frac{R_\alpha R_\beta}{(R_\beta - \Delta'_\beta)^2}\, d\Delta'_\beta\right) - \Delta_\alpha = \frac{R_\alpha R_\beta}{R_\beta + \Delta_\beta} - (R_\alpha + \Delta_\alpha),$$

which is easily seen to be equivalent to (1) by multiplying the RHS by the (positive) quantity, $R_\beta + \Delta_\beta$, giving an equivalent CFMM.

# 3 Generalizing to $n$-coin market makers

We can generalize the two-coin CFMMs given in §2 to ones where we can trade $n$ coins. Additionally, we can drop the requirements of nonzero derivative everywhere, though they will be replaced with a convexity-type requirement on the feasible trades. Though the presentation here is decidedly more general, it is also quite a bit more complex. As a note, we suspect that, in most practical cases, the definitions and notions presented in the previous section suffice.

In the $n$ coin case, as before, a CFMM is defined by a function $\varphi : \mathbf{R}^n_+ \times \mathbf{R}^n_+ \times \mathbf{R}^{n \times n}_+ \to \mathbf{R}$, such that the current state of the AMM is given by the reserves $R \in \mathbf{R}^n_+$. The AMM fulfills the trade only when the input $\Delta$ for output $\Lambda$ satisfies,

$$\varphi(R, \Delta, \Lambda) = 0, \tag{9}$$

while the reserves are updated in the following way,

$$R \leftarrow R + \Delta - \Lambda \mathbf{1}.$$

There are many AMMs of this form, with the most notable case being Balancer [29]. As before, we will sometimes write the function $\varphi_R$ instead of $\varphi(R, \cdot, \cdot)$ in what follows.

In this CFMM, $\Delta \in \mathbf{R}^n_+$ is the amount of coins that the trading agent inputs ($\Delta_i$ in this case represents the amount of the $i$th coin that the agent is putting into the CFMM's reserves) which we will just call the *input* while $\Lambda \in \mathbf{R}^{n \times n}_+$ is a matrix whose $ij$th entry specifies how much of (input) coin $i$ the user would like to trade for coin $j$. We will call this matrix $\Lambda$ the *output*. This implies that a feasible trade should always satisfy $\Lambda^T \mathbf{1} \leq \Delta$ (where the inequality is elementwise) and that $\Lambda \mathbf{1}$ is a vector whose $i$th entry specifies the amount of coin $i$ the agent wishes to receive. Note the inequality rather than the equality since we may have a nonzero input fee.

**Feasible trades.** We define the *feasible trades* for an $n$-coin trading function $\varphi$ slightly differently than in §2. Here, we define the feasible trades as the family of sets $T$ where $T(R) \subseteq \mathbf{R}^n_+ \times \mathbf{R}^{n \times n}_+$ for each possible reserve $R \in \mathbf{R}^n_+$, defined as

$$T(R) = \{(\Delta, \Lambda) \mid \varphi(R, \Delta', \Lambda') = 0 \text{ for some } \Delta' \leq \Delta \text{ and } \Lambda' \geq \Lambda\}.$$

(This is similar in spirit to the notion of an epigraph [7, §3.1.7] in optimization.)

We will consider the family of sets $T$ as opposed to $\varphi$ directly since, as before, there are potentially many functions $\varphi$ which yield equivalent AMMs, whereas the family of sets $T$ is essentially unique over the possible trading functions.

The idea behind the construction of feasible trades $T$ comes from the fact that agents are rational. If, for some desired output $\Lambda$ there exists two inputs $\Delta \neq \Delta'$, with $\Delta \leq \Delta'$, a rational agent will never opt to input $\Delta'$, since there exists a cheaper alternative $\Delta$. A similar reasoning applies for the output.

**Feasibility and minimality.** For fixed reserves $R$, we say an input-output pair $(\Delta, \Lambda)$ is *feasible* if $(\Delta, \Lambda) \in T(R)$. We will say that an output $\Lambda$ is feasible if there exists an input portfolio $\Delta$ such that the pair $(\Delta, \Lambda)$ is feasible.

We say an input $\Delta$ is *minimal* for an output $\Lambda$ if there is no smaller input with the same output, *i.e.*, there is is no $\Delta' \neq \Delta$ with $(\Delta', \Lambda)$ feasible and $\Delta' \leq \Delta$. In a similar way, we will say an output $\Lambda$ is *maximal* if for some input $\Delta$ if there is no feasible output portfolio which is larger.

## 3.1 Basic requirements

We will require a few properties of any AMM defined by $\varphi$.

**Zero is always feasible.** The trade $(0, 0)$ always feasible, *i.e.*, for any reserve $R \in \mathbf{R}^n_+$, we always have $(0, 0) \in T(R)$. Equivalently, the option to not trade is always feasible.

**Mixtures are never worse.** If $(\Delta, \Lambda)$ and $(\Delta', \Lambda')$ are both feasible input-output pairs, then $\gamma\Lambda + (1-\gamma)\Lambda'$ is also a feasible output with minimal price at most $\gamma\Delta + (1-\gamma)\Delta'$ for $0 \le \gamma \le 1$ and any reserve $R \in \mathbf{R}^n_+$.

There are many sufficient conditions which imply this property; for example, it is enough to have $\varphi_R$ be a quasiconvex function, though we note that this is not a necessary condition. (For more information on quasiconvex functions, see [4].)

## 3.2 Properties

These assumptions on $T$ (or, equivalently, on $\varphi$) have several important consequences.

**Monotonicity.** Surprisingly (and in parallel to the previous section), defining the set of feasible trades as $T(R)$ implies it suffices to consider $\varphi_R$ to be a function which is nonincreasing in its first argument and nondecreasing in its second, since the function

$$\tilde{\varphi}_R(\Delta, \Lambda) = \inf_{(\Delta', \Lambda') \in T(R)} \|\Delta - \Delta'\|_2^2 + \|\Lambda - \Lambda'\|_F^2,$$

where $\|\cdot\|_F^2$ is the squared Frobenius norm [7, §A], generates the same feasible trades $T(R)$ as $\varphi_R$ for any reserve $R \in \mathbf{R}^n_+$. While it is clear that $\tilde{\varphi}$ generates the same feasible trades as $\varphi$, its monotonicity is not obvious, so we prove it in appendix 6.1. Additionally, this implies that we can relax the equality of (9) to an inequality constraint:

$$\varphi(R, \Delta, \Lambda) \le 0,$$

since this inequality will always hold at equality if the function $\varphi$ is continuous around 0 (see appendix 6.2).

**Convexity of $T(R)$.** Given any two feasible input-output pairs $(\Delta, \Lambda)$ and $(\Delta', \Lambda')$, note that, for each $0 \le \gamma \le 1$, the output portfolio $\gamma\Lambda + (1-\gamma)\Lambda$ has a minimal input is no more than $\gamma\Delta + (1-\gamma)\Delta$ and therefore $(\gamma\Delta + (1-\gamma)\Delta, \gamma\Lambda + (1-\gamma)\Lambda) \in T(R)$. This implies that the set $T(R)$ is convex, and therefore that we may consider $\varphi_R$ to be a convex function that is increasing in its first argument and decreasing in its second, which we assume for the rest of the section.

**Arbitrage is easy.** The arbitrage problem is convex, assuming there is an infinitely liquid reference market such that it costs $c^T\Delta$ to purchase the $n$ coins $\Delta$ with respect to some numéraire. This is nearly immediate from the convexity of $\varphi_R$ and its monotonicity, as we have, in the same vein as (6),

$$\begin{array}{ll} \text{maximize} & c^T \Lambda \mathbf{1} - c^T \Delta \\ \text{subject to} & \varphi_R(\Delta, \Lambda) \le 0. \end{array} \tag{10}$$

where the variables in this problem are $\Lambda \in \mathbf{R}^{n \times n}_+$ and $\Delta \in \mathbf{R}^n_+$. This problem is convex as it is optimizing a linear function of the variables in the problem, subject to a convex inequality constraint. Note that we could have written the inequality constraint in an equivalent form, $(\Delta, \Lambda) \in T(R)$.

## 3.3 Definitions

### 3.3.1 Path independence

As given in §2.3.1, nearly the same definition carries through. We will say that a CFMM is path independent whenever it satisfies the following. For any feasible trade $(\Delta, \Lambda)$, and any (potentially infeasible) trade $(\Delta', \Lambda')$ we have that the aggregate trade $(\Delta + \Delta', \Lambda + \Lambda')$ is feasible if, and only if, the trades $(\Delta, \Lambda)$ and $(\Delta', \Lambda')$ are feasible when performed sequentially. So, given a trade $(\Delta, \Lambda)$ which satisfies

$$\varphi(R, \Delta, \Lambda) = 0,$$

then, for any $(\Delta', \Lambda')$, the following must be satisfied:

$$\varphi(R + \Delta - \Lambda\mathbf{1}, \Delta', \Lambda') = 0, \quad \text{if, and only if,} \quad \varphi(R, \Delta + \Delta', \Lambda + \Lambda') = 0.$$

As before, and by the same proof, this is true if, and only if, $\varphi$ can be written as a function depending only on the state of the reserves of the AMM, *i.e.*,

$$\varphi(R, \Delta, \Lambda) = \psi(R + \Delta - \Lambda\mathbf{1}), \tag{11}$$

which is similar in spirit to the cost functions of LMSRs and other scoring rules [21]. Note that this function $\varphi$ is nonincreasing in all of its arguments.

**Path deficiency.** The same relaxation from path independence to path deficiency works as well. Due to the monotonicity property of $\varphi$, we will similarly define path deficiency. We will say that an $n$-coin CFMM is path deficient if, for any feasible trade $(\Delta, \Lambda)$ then any arbitrary (and potentially infeasible) trade $(\Delta', \Lambda')$ satisfies

$$\varphi(R, \Delta + \Delta', \Lambda + \Lambda') = 0 \quad \text{implies} \quad \varphi(R + \Delta - \Lambda\mathbf{1}, \Delta', \Lambda') \geq 0,$$

and,

$$\varphi(R + \Delta - \Lambda\mathbf{1}, \Delta', \Lambda') = 0 \quad \text{implies} \quad \varphi(R, \Delta + \Delta', \Lambda + \Lambda') \leq 0.$$

As before, the monotonicity of $\varphi$ implies that, if $\varphi(R, \Delta, \Lambda) > 0$, then the agent has not given enough input for the amount of output they desire, and vice versa. If a CFMM is path deficient, then the requirement above simply states that performing an aggregate trade is never more expensive than performing the trade in a piecemeal fashion.

**Notes on path independence.** Unlike the two coin case, path independence in the $n$ coin case makes the analysis quite a bit simpler. The intuitive reason for this is that, in the general case, there may be many possible output trades $\Lambda$ which lead to the desired output portfolio $\Lambda\mathbf{1}$, so specifying the price of an output portfolio requires somehow picking the cheapest or best of the possible trades. In the path independent case, there is no need to optimize over the possible $\Lambda$, as any two outputs $\Lambda$ and $\Lambda'$ which satisfy $\Lambda\mathbf{1} = \Lambda'\mathbf{1}$ will always have the same cost, which follows immediately from (11).

### 3.3.2 Marginal price

**Path independent derivation.** It is not hard to show that the path independent price at some given reserves $R$ is proportional, with positive proportionality constant, to $-\nabla\psi(R)$, which follows from the same argument as (5). (The proportionality constant comes from the fact that the numéraire is arbitrary.)

Additionally, we can give a simple sufficient condition in this case for a path independent trading function $\psi$ to be valid for any cost vector:

$$\mathbf{R}^n_{++} \cap S^{n-1} \subseteq \{\nabla\psi(R) \mid R \in \mathbf{R}^n_+\},$$

where $S^{n-1}$ is the $(n-1)$-sphere; *i.e.*, that, for every direction on the positive orthant, there exists some reserves $R$ such that the cost vector points in that direction.

**General derivation.** The general definition of the marginal price of coin for the $n$ coin case is slightly trickier than that of the marginal price for two coins or the path independent one. In this case, there are many possible routes to receiving some desired output portfolio (*e.g.*, there are many possible $\Lambda$ such that $\Lambda\mathbf{1}$ gives the desired output), so the definition of the marginal price must take this into account.

To solve this, we will consider the following scenario: a trader wishes for some desired coin output $\Pi \in \mathbf{R}^n_+$, such that the trade $(\Delta, \Lambda)$ performed on this CFMM minimizes the cost to the trader, subject to the output received by the trader being equal to some desired output $\Pi = \Lambda\mathbf{1}$. We can set this up as the following convex optimization problem,

$$\begin{array}{ll} \text{minimize} & c^T\Delta \\ \text{subject to} & \Pi = \Lambda\mathbf{1} \\ & \varphi_R(\Delta, \Lambda) \le 0, \end{array} \tag{12}$$

with variables $\Lambda \in \mathbf{R}^{n\times n}_+$ and $\Delta \in \mathbf{R}^n_+$ and data $\Pi \in \mathbf{R}^n_+$ and $c \in \mathbf{R}^n_+$. We will write the optimal value of this problem as $p_R(c, \Pi)$. The marginal price of all coins (as a vector) in the CFMM with trading function $\varphi$ is then simply the gradient of $p_R$ at some per-unit cost $c$ and some desired output $\Pi$; *i.e.*, the marginal price of coin $i$ is given by

$$\left(\nabla_\Pi\, p_R(c, \Pi)\right)_i,$$

if $p_R$ is differentiable. (It may be the case that $p_R$ is not, but, as it is convex in $\Pi$, the function always has a subgradient [7, §5.6], which serves a similar role for our purposes.)

## 3.4 Implications

### 3.4.1 Marginal price under no arbitrage

It is not difficult to show that the optimal conditions for the arbitrage problem (10) imply that the (minimal) marginal price of the given output are equal to the price given by the reference market.

**Sensitivity conditions and equality.** As before, it is not hard to prove that, under the optimality conditions of (10), this marginal price is equal to the market price. To show this, first write the optimality conditions for (10) (we will assume that $\varphi$ is differentiable for simplicity, but a similar argument can be carried out with subgradients),

$$\mathbf{1}c^T = \lambda \nabla_\Lambda \varphi_R(\Delta^\star, \Lambda^\star), \quad \text{and} \quad c = -\lambda \nabla_\Delta \varphi_R(\Delta^\star, \Lambda^\star),$$

where $\lambda \in \mathbf{R}_+$ is the Lagrange multiplier for the inequality constraint of (10), and $\Delta^\star$, $\Lambda^\star$ are optimal for (10). Since we wish to look at the marginal price after the no-arbitrage trade, we then set $\Pi = \Lambda^\star \mathbf{1}$, which would then imply that $\Lambda^\star$ and $\Delta^\star$ are also optimal for (12). (This is easy to see: note that the objective of (10) depends only on $\Lambda\mathbf{1}$, so constraining $\Lambda\mathbf{1}$ to be equal to $\Lambda^\star\mathbf{1}$, a constant, yields the claim.)

Now, the optimality conditions for (12) are the following:

$$c = -\eta \nabla_\Delta \varphi_R(\Delta^\star, \Lambda^\star), \quad \text{and} \quad \mathbf{1}\nu^T = \eta \nabla_\Lambda \varphi_R(\Delta^\star, \Lambda^\star),$$

for Lagrange multipliers $\eta \in \mathbf{R}_+$ corresponding to the inequality constraint and $\nu \in \mathbf{R}^n$ corresponding to the equality constraints of (12). This easily shows that $\lambda = \eta$ and that $\nu = c$. Finally, using the fact that the derivative of the optimal value, with respect to the sensitivity conditions, is equal to the Lagrange multiplier corresponding to those conditions [7, §5.6.3] yields the final result, that

$$\nabla_\Pi \, p_R(c, \Pi) = \nu = c,$$

as required. It is possible that the derivative of $p_R$ may not be defined, but, in this case we can replace gradient with subgradient and the corresponding equalities with set inclusion (since $c$ is therefore a possible subgradient; *cf.*, [7, §5.6.2]).

### 3.4.2 LP returns for path independent CFMMs

We can derive the liquidity provider returns in a general form whenever we know that the CFMM is path independent. In particular, path independence implies that the trading function, $\psi$, can be written only in terms of its reserves, and, assuming no-arbitrage, we can then write

$$-\lambda \nabla \psi(R) = c, \tag{13}$$

where $c \in \mathbf{R}^n_+$ is a vector containing the current market price of all coins with respect to some numéraire, and $\lambda \in \mathbf{R}_+$ is a positive proportionality constant. (The more general case replaces the gradient with subgradients.) There is a very simple relationship between the current price of the coins and the returns for the given reserves, which depends on the convex conjugate of $\psi$.

The idea is to interpret the no arbitrage condition given in (13) as the first-order optimality conditions for an optimization problem. One such (simple) problem is

$$\begin{array}{ll} \text{minimize} & c^T R \\ \text{subject to} & \psi(R) \leq 0, \end{array} \tag{14}$$

with the variable being the reserves $R \in \mathbf{R}^n$ (we will ignore positivity constraints and assume they are included in the domain of the function $\psi$, for simplicity). The Lagrangian for problem (14) is [7, §5.1.1],

$$\mathcal{L}(R, \lambda) = c^T R + \lambda \psi(R),$$

and the KKT optimality conditions are [7, §5.5.3],

$$c + \lambda \nabla \psi(R) = 0, \quad \lambda \geq 0,$$

which are exactly the conditions for (13). So, if there exists exactly one unique set of reserves $R$ such that (13) holds, then the optimal objective value of (14) is exactly the current reserve value.

Additionally, there is a very simple connection between the current price of the reserves and the convex conjugate of $\psi$. Assuming strong duality holds for (14) then its optimal value is equal to

$$\sup_{\lambda \geq 0} \inf_R \left(c^T R + \lambda \psi(R)\right) = \sup_{\lambda \geq 0} \left(-\lambda \sup_R \left(-\frac{c^T R}{\lambda} - \psi(R)\right)\right) = \sup_{\lambda \geq 0} \left(-\lambda \psi^* \left(-\frac{c}{\lambda}\right)\right), \quad (15)$$

where $\psi^*$ is the convex conjugate [7, §3.3] of $\psi$. We write the term

$$-\lambda \psi^* \left(-\frac{c}{\lambda}\right),$$

slightly loosely, as this expression's value is not defined at zero. Very generally, we write this expression to mean the *perspective* transform of $\psi^*$ [39, §8], which is well-defined even when $\lambda$ is zero, even though the expression written above is not, but we keep this presentation for simplicity.

We show how to use this to easily derive the return for Uniswap and Balancer liquidity providers in the no-fee case in appendix 6.3.

**Notes on returns.** We suspect strong duality holds in practice for most instances of (14), since Slater's condition [7, §5.2.3] holds whenever there exists some reserves $R$ such that $\psi(R) < 0$. We note this as strong duality may not hold if the rewriting of the trading function in §3.2 is used, since the given function is always nonnegative. We can also rewrite the above conditions in a slightly more general form such that Slater's condition always holds if the set of feasible trades has nonempty interior. In this case, the convex conjugate of $\psi$ will be replaced with the support functions [7, prob. 2.26] for the set of feasible trades (and no perspective transformation is needed).

A second important point to note is that there may many reserves $R$ which yields a given marginal price vector. In this case, the optimal value of (13) likely will still, in practice, be equal to the true portfolio value since any arbitrageur is incentivized to rebalance the reserves in such a way that they earn positive payoff, even though the marginal prices reported by the CFMM remain unchanged. This gives an important distinction between the two coin case versus the $n$ coin case.

# 4 Discussion of CFMMs and future work

## 4.1 Trading fees

We can easily introduce trading fees to any given CFMM. A very simple but effective approach, such as the one taken by Uniswap, is to introduce fees on the input trade. Given a trading function $\varphi$ for a CFMM, we can then write a new trading function $\varphi_f$ with some fee constant $0 < \gamma \leq 1$ defined as

$$\varphi_f(R, \Delta, \Lambda) = \varphi(R, \gamma\Delta, \Lambda),$$

for all reserves $R$, inputs $\Delta$, and outputs $\Lambda$ where $(1 - \gamma)$ is the percentage fee required. In this case, the trader is required to put in $1/\gamma$ more of input $\Delta$ for a trade to be feasible. Additionally, this will turn path independent CFMMs into path deficient ones, keeping many of the required conditions, but potentially making the analysis of returns for LPs more difficult. This method also has the nice property that the LP returns are bounded from below by the solution to (14).

There are several more possible methods, some of which may include variable input and/or output fees which vary in such a way as to keep other desirable properties of the CFMMs on a case-to-case basis. We suspect that there are many approaches for charging trading fees, each with their own useful properties, but leave the possibility of finding a suitable class of these trading fees for future work.

## 4.2 Comparison to LMSRs

While it is tempting to ask about potential comparisons or equivalences to classic, algorithmic game theory automated market makers and scoring rules such as Hanson's LMSR, we note that, under the framework used in this paper, there doesn't appear to be a simple way of doing so.

A simple thought experiment shows why our current framework cannot be used for this type of comparison. Given an infinitely liquid reference market with a fixed price $p_1$ from time $[0, T)$, where $T > 0$, and price $p_2 > p_1$ at time $T$ (known by all agents), then the reported price of a prediction market which seeks to predict the price of the asset at time $T$ and any valid CFMM will always diverge by $p_2 - p_1$. Rational agents will always be incentivized to correctly report the (known) future price $p_2$ for the prediction market, while arbitrageurs will always make positive payoff from any CFMM which diverges from the current market price $p_1$ at all times $[0, T)$, by setting the reported price to be $p_1$. Sending $p_2 - p_1 \to \infty$ then shows that these two AMMs can diverge by any desired amount.

The idea here is that any framework which can compare the two will require assumptions about the market price dynamics, *i.e.*, what the current market price might say about the future market price, which we do not assume at any point. We leave this potentially very interesting research avenue of finding a suitable framework for comparison for future work.

### 4.3 Optimization over possible CFMMs

Note that the conditions given above define a family of CFMMs which are likely to be useful in practice. This implies that, given any performance metric (such as LP returns for a given market model) that a market maker designer wishes to optimize, one could find an (approximately) optimal CFMM to accomplish this task. The problem is likely to be computationally difficult in most important cases, but we suspect that many commonly-used heuristics will likely find good results. Though this is likely not possible for the general $n$ coin case except for the smallest possible $n$ or for a very specific subclass of trading functions, we imagine that the two coin case can be quickly optimized on modern hardware for many useful performance metrics.

## 5 Conclusion

The increase in usage and participation in automated market makers has led to a vast set of new scoring rules and pricing mechanisms. Analyzing these mechanisms, which range from LMSR style market makers and CFMMs to scoring rules for rates [12], from the perspective of optimization provides insight into why certain mechanisms are more popular than others. In particular, we generalize the results of [5] to demonstrate that CFMMs provide an easy optimization problem for arbitrageurs to synchronize off-chain and on-chain pricing data. This generalization encompasses all live CFMMs [48, 29, 16, 22] and provides guidance on how one can design CFMMs that are better for certain asset types and volatilities. Further work on CFMMs is necessary to connect how to choose the optimal scoring rule given the volatilities of the external price feeds. As the StableSwap/Curve example demonstrates, adjusting the curvature of the rule can lead to improved performance and participation for different assets. Finally, we leave the study of how to estimate the cost of corruption [5, 28], which can be thought of as a relaxation of the price of anarchy [41], for future work.

## Acknowledgements

We would like to thank John Morrow and Tim Roughgarden for feedback on this paper.

## References

[1] Abernethy, J., Chen, Y., and Wortman Vaughan, J. An optimization-based framework for automated market-making. In *Proceedings of the 12th ACM conference on Electronic commerce* (2011), pp. 297–306.

[2] Adams, H. Uniswap birthday blog - v0, Nov 2019.

[3] Adler, J., Berryhill, R., Veneris, A., Poulos, Z., Veira, N., and Kastania, A. Astraea: A decentralized blockchain oracle. In *2018 IEEE International Confer-*

*ence on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)* (2018), IEEE, pp. 1145–1152.

[4] Agrawal, A., and Boyd, S. Disciplined Quasiconvex Programming. *arXiv:1905.00562 [cs, math]* (June 2019).

[5] Angeris, G., Kao, H.-T., Chiang, R., Noyes, C., and Chitra, T. An analysis of uniswap markets. *arXiv preprint arXiv:1911.03380* (2019).

[6] Bene, N. Getting to the core, Apr 2018.

[7] Boyd, S. P., and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004.

[8] Chainlink Foundation. Welcome to chainlink.

[9] Chakraborty, M., Das, S., Lavoie, A., Magdon-Ismail, M., and Naamad, Y. Instructor rating markets. In *Twenty-Seventh AAAI Conference on Artificial Intelligence* (2013).

[10] Chen, X., Papadimitriou, C., and Roughgarden, T. An axiomatic approach to block rewards. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies* (2019), pp. 124–131.

[11] Cheng, Y. 0x developers release liquidity aggregation tool for ethereum-based exchange protocol, Jan 2020.

[12] Chitra, T. Competitive equilibria between staking and on-chain lending. *arXiv preprint arXiv:2001.00919* (2019).

[13] Cuen, L. Bancor raised $153 million and found actual users. why did its price tank?, Nov 2019.

[14] Decker, C., and Wattenhofer, R. Bitcoin transaction malleability and mtgox. In *European Symposium on Research in Computer Security* (2014), Springer, pp. 313–326.

[15] Dune Analytics. Dune analytics decentralized exchange dashboard, Jan 2020.

[16] Egorov, M. Stableswap — efficient mechanism for stablecoin liquidity. unpublished, Nov 2019.

[17] Egorov, M. Curve, Jan 2020.

[18] Gastineau, G. L. An introduction to exchange-traded funds (etfs). *Journal of Portfolio Management and Economics 27*, 3 (2001), 88–96.

[19] Goodman, L. Tezos—a self-amending crypto-ledger white paper. *URL: https://www. tezos. com/static/papers/white_paper. pdf* (2014).

[20] Google. google/arithmancer, May 2017.

[21] Hanson, R. Combinatorial information market design. *Information Systems Frontiers 5*, 1 (2003), 107–119.

[22] Hertzog, E., Benartzi, G., and Benartzi, G. Bancor protocol.

[23] Higgins, S. Decentralized exchange protocol 0x raises $24 million in ico, Aug 2017.

[24] Juliano, A. dydx: A standard for decentralized derivatives, 2017.

[25] Kaminska, I. Bitcoin bitfinex exchange hacked: The unanswered questions. *Financial Times 4* (2016).

[26] Kamvar, S., Olszewski, M., and Reinsberg, R. Celo: A multi-asset cryptographic protocol for decentralized social payments.

[27] Koblitz, N., and Menezes, A. J. The random oracle model: a twenty-year retrospective. *Designs, Codes and Cryptography 77*, 2-3 (2015), 587–610.

[28] Lambur, H., Lu, A., and Cai, R. *UMA Data Verification Mechanism: Adding Economic Guarantees to Blockchain Oracles*. Jul 2019.

[29] Martinelli, F., and Mushegian, N. Balancer: A non-custodial portfolio manager, liquidity provider, and price sensor.

[30] Mohanty, D. Advanced programming in oraclize and ipfs, and best practices. In *Ethereum for Architects and Developers*. Springer, 2018, pp. 151–179.

[31] Nace, S. Topic: Exchange traded funds.

[32] O'Donnell, R. *Analysis of boolean functions*. Cambridge University Press, 2014.

[33] Othman, A., Pennock, D. M., Reeves, D. M., and Sandholm, T. A practical liquidity-sensitive automated market maker. *ACM Transactions on Economics and Computation (TEAC) 1*, 3 (2013), 1–25.

[34] Othman, A., and Sandholm, T. Automated market-making in the large: the gates hillman prediction market. In *Proceedings of the 11th ACM conference on Electronic commerce* (2010), pp. 367–376.

[35] Peterson, J., and Krug, J. Augur: a decentralized, open-source platform for prediction markets. *arXiv preprint arXiv:1501.01042* (2015).

[36] Phillips, M. What's behind the mysterious intrade shutdown?, Mar 2013.

[37] PROVABLE DATA TEAM. A scalable architecture for on-demand, untrusted delivery of entropy, Mar 2019.

[38] PULSE, D. Defi pulse: The defi leaderboard: Stats, charts and guides.

[39] ROCKAFELLAR, R. T. *Convex Analysis*, vol. 28. Princeton university press, 1970.

[40] ROSE, I. N. Poker's black friday. *Gaming Law Review and Economics 15*, 6 (2011), 327–331.

[41] ROUGHGARDEN, T. *Selfish routing and the price of anarchy*, vol. 174. MIT press Cambridge, 2005.

[42] SHANE, D. A crypto exchange may have lost $145 million after its ceo suddenly died. *CNN Business, https://edition. cnn. com/2019/02/05/tech/quadriga-gerald-cotten-cryptocurrency/index. html* (2019).

[43] SPIVAK, M. *Calculus*. Publish or Perish, Houston, Tex., 2008. OCLC: 1028869960.

[44] TEAM, M. The dai stablecoin system. *URl: https://makerdao. com/whitepaper/DaiDec17WP. pdf* (2017).

[45] WARREN, W., AND BANDEALI, A. 0x: An open protocol for decentralized exchange on the Ethereum blockchain.

[46] WOLFRAM, S. Did stephen wolfram's knowledge engine just become a quantum neural blockchain AI?, Apr 2018.

[47] WOOD, G., ET AL. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper 151*, 2014 (2014), 1–32.

[48] ZHANG, Y., CHEN, X., AND PARK, D. Formal specification of constant product (xy=k) market maker model and implementation.

# Appendix

## 6 Miscellaneous Proofs

### 6.1 Monotonicity of distance to feasible trades

We will prove the monotonicity claim for more general sets $Q \subseteq \mathbf{R} \times \mathbf{R}^n$ of the form

$$Q = \{(t, q) \mid (t', q) \in S \text{ for some } t' \leq t\},$$

for some nonempty closed set $S \subseteq \mathbf{R} \times \mathbf{R}^n$.

Let $d : \mathbf{R} \times \mathbf{R}^n \to \mathbf{R}_+$ be the squared distance to the set $Q$,

$$d(t, q) = \inf_{(t',q') \in Q} \|(t - t', q - q')\|_2^2,$$

we will then show that $d$ is decreasing with respect to its first argument. Consider some pair $(t, q) \in \mathbf{R} \times \mathbf{R}^n$ and let $(t^\star, q^\star) \in \mathbf{R} \times \mathbf{R}^n$ minimize $d(t, q)$. Now, for any $t' \in \mathbf{R}$ with $t' \geq t$, either $t' \geq t^\star$, in which case $d(t', q) \leq \|q - q^\star\|_2^2 \leq d(t, q)$ as $(t', q^\star) \in Q$, or $t' < t^\star$ which implies that $t \leq t' < t^\star$ so clearly,

$$d(t', q) \leq (t' - t^\star)^2 + \|q - q^\star\|_2^2 < (t - t^\star)^2 + \|q - q^\star\|_2^2 = d(t, q),$$

proving the claim.

The complete proof then follows from the fact that we can apply this proof elementwise to the function $\tilde{\varphi}$ which is equal to the squared distance to the set of feasible trades, $T(R)$.

### 6.2 Equivalence of AMM with inequality

It suffices to prove that if $\varphi(R, \Delta, \Lambda) < 0$, then there exists a strictly better input-output pair, $(\Delta', \Lambda')$, with $\varphi(R, \Delta', \Lambda') \leq 0$, which follows nearly immediately by continuity. In particular, by continuity, there are two open balls $B_\Delta \subseteq \mathbf{R}^n$ and $B_\Lambda \subseteq \mathbf{R}^{n \times n}$ such that for any $\delta \in B_\Delta$ and $\lambda \in B_\Lambda$,

$$\varphi(R, \Delta + \delta, \Lambda + \lambda) \leq 0.$$

This implies that exists some strictly positive input change $\delta \in \mathbf{R}^n_{++} \cap B_\Delta$ and strictly positive output change $\lambda \in \mathbf{R}^{n \times n}_{++} \cap B_\Lambda$, such that

$$\Delta' = \Delta + \delta, \quad \Lambda' = \Lambda + \lambda,$$

are both feasible and strictly dominating over $(\Delta, \Lambda)$ since $\Delta' > \Delta$ and output $\Lambda' > \Lambda$.

## 6.3 Uniswap and Balancer LP returns

We can write the Uniswap trading function in a way that satisfies the conditions of $n$-coin CFMMs and use it to derive the LP returns via the equation given in (15) in a very simple way. Generally speaking, we can write any constant mean market without fees [5, §3] by writing

$$\psi(R) = k - \prod_{i=1}^{n} R_i^{w_i},$$

where $w \in \mathbf{R}_{++}^n$ are positive weights satisfying $\mathbf{1}^T w = 1$ and $k \in \mathbf{R}_{++}$ is the constant product. First, we note that [7, prob. 3.36],

$$\psi^*(-c) = \begin{cases} -k & c \geq 0, \ \prod_i c_i^{w_i} \geq 1/n \\ \infty & \text{otherwise.} \end{cases}$$

Recall that the current value of the reserves is given by (15):

$$\sup_{\lambda \geq 0} \left(-\lambda \psi^* \left(-\frac{c}{\lambda}\right)\right).$$

Now, note that

$$-\lambda \psi^* \left(-\frac{c}{\lambda}\right) = \begin{cases} \lambda k & \left(\prod_i c_i^{w_i}\right)/\lambda \geq 1/n \\ -\infty & \text{otherwise,} \end{cases}$$

which we can easily maximize by choosing the largest possible $\lambda$, *i.e.*, $\lambda = n \prod_i c_i^{w_i}$, yielding

$$\sup_{\lambda \geq 0} \left(-\lambda \psi^* \left(-\frac{c}{\lambda}\right)\right) = kn \prod_{i=1}^{n} c_i^{w_i}, \tag{16}$$

where $c \in \mathbf{R}_+^n$ is the cost vector.

Using the special case of $n = 2$ and $w_1 = w_2 = 1/2$ in (16), we can recover the LP returns of Uniswap with no fees given by [5, §2.3]. First set $c_1 = 1$ (*i.e.*, coin 1 is the numéraire) such that $c_2$ is the market price of coin 2 with respect to coin 1. Then, we can write the total value in reserves as,

$$P_V = 2k\sqrt{c_2},$$

where the square root difference comes from the fact that $k$ here is the *square root* of the product constant, *i.e.*, $k = \sqrt{R_\alpha R_\beta}$. (See [5, §3].)