# MULTI-AGENT AUTOMATED MARKET MAKING

**Chris Slaughter**
LVL
Los Angeles, CA 900036
chris@lvl.co

December 14, 2020

**ABSTRACT**

In this white paper we discuss the background and design decisions of our new AMM v2 "Yield Farming" algorithm. We provide a quick review of constant function pricing models (CFPMs). We propose a new CFPM scoring rule designed to promote inventory management, while also providing price protections for retail market orders. We touch on ways AMM v2 may impact pricing, liquidity, and arbitrage on the exchange. We also share the resuts of running AMM v2 in simulated markets.

***K*eywords** Exchange · Liquidity · Automated Market Making

## 1 Background

LVL is a US-based financial institution and mobile app that offers free cryptocurrency trading and other consumer financial. LVL operates its own centralized exchange, or spot market, and does not route order flow to institutional market makers or allow them to trade on LVL. To ensure the availability of liquidity on LVL's spot market, we developed and deployed automated market making on LVL in 2019 [1]. Automated market making is available to all LVL Premium members as part of the Autopilot feature of the mobile app. LVL does not participate in automated market making on its own spot market. The automated market making algorithm, which we refer to in this paper as AMM v1, implements the marketing making strategy of Avellaneda and Stoikov [2], the optimal strategy for marketing making on limit order books under the standard brownian motion model of market price. AMM v1 modifies this algorithm in several ways to simplify user configuration and manage asset inventory. Since March 2019, Autopilot accounts running AMM v1 has provided liquidity for millions of dollars of market order flow, while earning a potential income for users. While AMM v1 has demonstrated efficient pricing to market, it has experienced numerous instances where inventory has been either sold or bought out on a pair. When inventory is sold or bought out, buy and sell market orders can no longer be executed, respectively. We refer to this as the inventory management problem, and it is the central motivation for AMM v2.

Since we introduced automated market making on LVL, there has been an explosion of interest in automated market making in decentralized finance, or DeFi. DeFi is a system of censorship-resistant, decentralized financial primitives including lending [3], stable assets [4], and exchanges [5]. DeFi exchanges provide a mechanism of exchange of digital assets between interested parties similar to exchanges and marketplaces found in traditional. These digital assets can include online ad units, prediction market bets, or cryptocurrencies. Because digital assets are an emerging class of assets, DeFi exchanges have historically suffered from low liquidity, which has led to the introduction of automated market makers to "bootsrap" liquidity in nascent digital asset markets [6]. Automated market makers allow market participants to passively allocate their digital assets to the provision of liquidity on an exchange. In return, these participants earn a potential yield on their assets through either trading commissions or a spread on the marginal exchange price [7]. Automated market making algorithms use a *scoring rule* to map liquidity pool reserves and/or a reference market price provided by an oracle, to a marginal asset price. A popular scoring rule for prediction markets is Hanson's *Logarithmic Market Scoring Rule* (LMSR) [8]. Our AMM v1 market making algorithm uses a scoring rule based on stochastic optimal control to price limit orders on a spot market. Bancor introduced a scoring rule based on bond curves [9]. The most well known scoring rule for DeFi exchanges is the *constant product pricing model*, first introduced in the Uniswap Protocol. Constant product pricing is part of a broader family of *constant function pricing* *models* (CFPMs) which require any trade to result in reserves in such a way that some function of the reserves is always equal to some constant $k$. In CPPM exchanges, this function is $xy = k$. CPPMs appear to work exceptionally well at solving the inventory management problem despite their simplicity [10]. CPPMs, and the broader CFPM family have been demonstrated to closely track the reference price under common conditions.

In this white paper we discuss the background and design decisions of our new AMM v2 "Yield Farming" algorithm. We provide a quick review of constant function pricing models (CFPMs). We propose a new CFPM scoring rule designed to promote inventory management, while also providing price protections for retail market orders. We touch on ways AMM v2 may impact pricing, liquidity, and arbitrage on the exchange. We also share the resuts of running AMM v2 in simulated markets.

## 2 Constant Function Pricing Models

The section explains the motivtion and mathematical formulation of several well known CFPMs. In exploring the mathematical formulations, we deviate from the reference work in formulas and variable names so that the reader may benefit from a more notationally cohesive exploral of multiple diffent CFPMs. Recall that CFPMs require that the asset reserves, represented in their notional value of a single currency (typically USD), $R_1, R_2, \ldots, R_n \in \mathbb{R}^+$ satisfy at all times the relation $\varphi(R_1, R_2, \ldots, R_n) = k$. Here, $k$ is the constant and $\varphi$ is the *constant function*. The constant function is not to be confused with the constant map, which mapst the entire domain too a constant. The set of all valid reserve allocions $(R_1, R_2, \ldots, R_n)$, the preimage of $k$ under $\varphi$. Not all valid reserve allocations may be reachable based on choice of constant function and market model.

### 2.1 Constant Product Pricing Model

We begin with Constant Product Pricing Model from Uniswap [5]. This model uses the constant function $xy = k$. We denote the constant function as $\varphi_\times(R_1, R_2)$, and valid reserve allocations satisfy:

$$\varphi_\times(R_1, R_2) = R_1 R_2 = k \tag{1}$$

An easy way to visualize the set of valid asset allocations $(R_1, R_2)$ is to plot them as a curve on a graph where the $x$ and $y$ axes represent the value of reserves $R_1$ and $R_2$ respectively. In this case, the curve is $R_2(R_1) = k/R_1$, the familiar inverse linear function $y = a/x$. Because any trade must result in another point on this curve, it is useful to charactize which trades of a given asset allocation $(R_1, R_2)$ result in a new asset allocation $(R_1', R_2')$ that satisfies the invariant:

$$\varphi_\times(R_1', R_2') = \varphi_\times(R_1, R_2) = k \tag{2}$$

Trades which satisfy this equation are considered valid because they preserve the constant product $k$. We can characterize a trade as a pair of fills $(\Delta_1, \Delta_2)$. W.l.o.g., we can assume that the market order being executed is trading asset $\Delta_1$ for asset $\Delta_2$. Execution of the trade $(\Delta_1, \Delta_2)$ results in a new asset allocation $(R_1, R_2) \to (R_1 + \Delta_1, R_2 - \Delta_2)$. For this asset allocation to be a valid, it must satisfy $\varphi_\times(R_1 + \Delta_1, R_2 - \Delta_2) = k$. Through manipulation, we can see that $\Delta_1$ can be expressed as a function of $(\Delta_2, R_1, R_2)$:

$$\Delta_2 = \frac{R_1 \Delta_2}{R_2 - \Delta_2} \tag{3}$$

$\Delta_1$ is the amount of asset that must be *tendered* to purchase $\Delta_2$ units of another asset. We can see that as $\Delta_2$ approaches $R_2$, the total amount of $\Delta_1$ that must be tendered tends towards infinity. As a result, it is impossible for either $R_1$ or $R_2$ to sell out, provided that $R_1, R_2 > 0$ at the outset. We can also consider the marginal price to purchase $\Delta_2$, $P_{\text{buy}}(\Delta_2) = \Delta_1/\Delta_2$, which we can see from the above equation is:

$$P_{\text{buy}}(\Delta_2) = \frac{R_1}{R_2 - \Delta_2} \tag{4}$$

In this way, as a buyer attempts to purchase more $\Delta_2$ from the liquidity pool, both the price and in turn the amount of $\Delta_1$ that must be tendered become unbounded. We can also see that the marginal cost of

### 2.2 Old Work

We can define the executed price, $p_{\Delta_2}(\Delta_1)$, on this trade as $\Delta_1/\Delta_2$, which represents the units of $\Delta_1$ required to buy one unit of $\Delta_2$. The price $p_{\Delta_1}(\Delta_2)$ can be solved using the above result, leading to:

$$p_{\Delta_2}(\Delta_1) = \Delta_1/\Delta_2 = \frac{R_1\Delta_1 + \Delta_1^2}{R_2\Delta_1} \tag{5}$$

This reveals two important properties of Constant Product Pricing Models. First, the price to buy $\Delta_2$ scales quadratially in the amount of $\Delta_1$ that is traded, a process called slippage. CPPM's quadratic slippage creates a strong disincentive for traders to buy out $R_2$, and in fact $R_2$ will not ever sell out, as we will see shortly. Secondly, by evaluating the limit of $p_{\Delta_2}(\Delta_1)$ as $(\Delta_1, \Delta_2) \to (0, 0)$ we can see that the marginal price on the liquidity pool is $R_1/R_2$.

We are particularly interested in the case in which $R_1$ represents US dollars so that $R_1 = R_D$, and $R_2$ is a cryptocurrency so that $R_2 = R_B = p_0 B$. $B$ is the reserve cryptocurrency holdings, and $p_0$ is the prevailing market price for the cryptocurrency, measured in dollars. We immediately see that $k = R_D R_B = R_D(p_0 B)$. As in the previous example, we assume a market order buy arrives represented by $(\Delta_D, \Delta_B)$. This can also be written as $(\Delta_D, \delta_B)$, where $\delta_B$ represents the fill size in the cryptocurrency, and the notional value of the cryptocurrency fill, $p_0\delta_B$ is implicit. Again, $\delta_B$ can be closed for offer size $\Delta_D$:

$$\delta_B = \frac{R_B\Delta_D}{p_0(R_D + \Delta_D)} \tag{6}$$

Further, we can calculate the price $p_{\delta_B}(\Delta_D)$ of this trade as:

$$p_{\delta_B}(\Delta_D) = \Delta_D/\delta_b = \frac{p_0(R_D\Delta_D + \Delta_D^2)}{p_0 B\Delta_D} = \frac{R_D\Delta_D + \Delta_D^2}{B\Delta_D} \tag{7}$$

Again, we see that the price to buy $\delta_B$ scales quadratically in the amount of $\Delta_D$ on offer by the market order. By taking the limit of $p_{\Delta_D}(\delta_B)$ as $(\Delta_D, \delta_B) \to (0, 0)$, we can the marginal price for the liquidity pool is $R_D/B$. Furthermore, the market discovery price, which is the marginal price when $R_D = R_B = p_0 B$, is seen to be $R_D/B = R_D/(R_D/P_0) = p_0$!

## 3 Multi-Agent CFPM

## References


[1] Chris Slaughter and Brandon Eng. Automated Market Making. *White Paper*, Samsa Technologies Inc. dba LVL, 2019. `https://github.com/chriscslaughter/amm/blob/master/paper/AMM.pdf`

[2] Marco Avellaneda and Sasha Stoikov. High-frequency trading in a limit order book. In *Quantitative Finance, Vol. 8, No. 3*, pages 217–224. Routledge, 2008.

[3] R. Leshner and G. Hayes. Compound: The money market protocol *Technical Report*, Compound Labs, 2019. `https://compound.finance/documents/Compound.Whitepaper.pdf`

[4] The Maker Protocol: MakerDAO's Multi-Collateral Dai (MCD) System *White Paper*, MakerDAO, 2015-2020. `https://makerdao.com/en/whitepaper`

[5] Hayden Adams, Noah Zinsmeister, and Dan Robinson. Uniswap v2 Core *White Paper*, Uniswap, 2020. `https://uniswap.org/whitepaper.pdf`

[6] A. Othman, D. M. Pennock, D. M. Reeves, and T. Sandholm. A practical liquidity- sensitive automated market maker. In *ACM Transactions on Economics and Computation (TEAC), vol. 1, no. 3*, pp. 1–25, 2013.

[7] Nikolai Kuznetsov. DeFi yield farming, explained. *Website Article*, CoinDesk, September 2020, accessed December 2020. `https://cointelegraph.com/explained/defi-yield-farming-explained`

[8] R. Hanson. Combinatorial information market design. In *Information Systems Frontiers, vol. 5, no. 1*, pp. 107–119, 2003.

[9] Eyal Hertzog, Guy Benartzi, Galia Benartzi. Continuous Liquidity for Cryptographic Tokens through their Smart Contracts. *White Paper*, Bancor Protocol, 2018. `https://storage.googleapis.com/website-bancor/2018/04/01ba8253\protect\discretionary{\char\hyphenchar\font}{}{}bancor_protocol_whitepaper_en.pdf`

[10] Guillermo Angeris, Hsien-Tang Kao, Rei Chiang, Charlie Noyes, and Tarun Chitra. An analysis of Uniswap markets. *ePrint*, arXiv:1911.03380. `https://arxiv.org/abs/1911.03380`