Dynamic Automated Market Making via Model Predictive Control

Bryan Chiang and Alan Yang

June 17, 2022

1 Introduction

Unique to decentralized finance (DeFi) is a derivative trading product known as the perpetual swap. In contrast to spot trading, perpetuals enable traders to gain exposure to an asset's price movement without needing to custody the actual asset. However, unlike other derivative products such as futures and options, perpetuals do not have an expiration date, eliminating the need for traders to constantly reestablish their positions. Instead, the perpetual price (mark price) is anchored to some underlying spot price (oracle) via a funding rate mechanism: if the mark is below the oracle, shorts pay the longs a fee proportional to their position. This results in shorts closing their positions and traders opening new long positions, driving the mark price up to match the oracle price. The reverse situation can also take place: if the mark price is above the oracle, longs pays shorts a fee.

Due to on-chain computation and storage constraints, decentralized perpetual exchanges including Drift protocol [Lab21] rely on automated market makers (AMMs) [AKC+19] for mark price discovery. An AMM pair between two assets, such as USD Coin (USDC) and Bitcoin (BTC), consists of *virtual* reserve amounts y for the quote asset (vUSDC) and x for the base asset (e.g. BTC). To execute a swap, a trader tenders Δy of the quote asset to the AMM and receives a corresponding amount Δx of the base asset. The amount received is determined by a function referred to as a *constant product curve*.

We consider a dynamic AMM (dAMM), where the AMM has control over the parameters of the constant product curve. The AMM can, at a cost, adjust the parameters over time to improve market conditions and trading experience on its platform, while increasing its revenue. Currently, parameters are modified according to ad-hoc heuristics. Optimal control based strategies have recently been proposed for the related Olympus protocol [CKA+22], but have yet to be explored for perpetual swaps.

In this work, we develop a model predictive control (MPC) scheme for achieving this. At each time step, we find parameter adjustments by solving a convex optimization problem based on forecasts of various signals. MPC, also known as receding horizon control, has been applied to similar problems in finance [BBD+17].

The rest of this report is organized as follows. We describe the basic operating principles of dAMMs we consider in Section 2. Next, we develop our MPC control scheme in Section 3 and present experimental results in Section 4. Finally, Section 5 concludes.

2 Dynamic Automated Market Makers

The following model is based on the Drift protocol [Lab21] and the documentation provided by the Drift team [1]. We assume a uniform time discretization. For times s between t and $t+1$, the values of the base and quote reserves x_s and y_s must be related by $x_s \cdot y_s = k_t$, where k_t is the curve invariant. Suppose that at time s a trader buys an amount Δx of the base asset. She gains a virtual quote asset amount of Δy, where

$$(x_s - \Delta x) \cdot (y_s + \Delta y) = k_t. \tag{1}$$

The value of Δy in the non-virtual quote asset amount is determined by $C_t \cdot \Delta y$, where C_t is the current price multiplier (peg). A swap in the other direction (base asset to quote asset) works similarly. Since C_t and k_t can be changed over time, the AMM is referred to as a *dynamic* AMM (dAMM).

Between time t and $t+1$, the dAMM collects trading fees, sets aside funds for changing C_t (also known as repegging) and changing k_t, and providing funding for the long and short positions in the market. In the following, we denote to the number of long and short positions in the market by l_t and s_t, respectively, and the net market position is given by $l_t - s_t$. Traders essentially place bets on the market price p_t relative to the *oracle price* $p_t^\star$, which is the market price of the non-virtual base asset (e.g. market price of Bitcoin). The dAMM promises a certain amount of funding for position holders in each time period, depending on the price difference $p_t - p_t^\star$ and the type of position held (long or short). If the long and short positions are equal, the losing side funds the winning side. Otherwise, the dAMM may have to make up the difference to meet the funding promise.

The dAMM has several objectives, some of which are competing.

1. Maximizing trading volume. The higher the trading volume, the more the dAMM makes through trading fees. The dAMM may be able to encourage more volume by ensuring that the market is in a healthy state. We consider two main aspects of market health.

    (a) Ensuring that p_t is close to $p_t^\star$. This encourages a good balance between longs and shorts, meaning that traders would hopefully be more willing to place bets. This also reduces the risk of losing a lot of money through funding payments, as discussed in Subsection 2.1.

    (b) Ensuring low slippage by increasing k. This means that the trade execution price experienced by a trader is very close to the market price at the time of execution. This is explained in more detail in Subsection B.

[1] `https://driftprotocol.notion.site/Drift-dAMM-deep-dive-ff154003aedb4efa83d6e7f4440cd4ab`

2. Minimizing operation costs from adjusting C and k. See Subsection 2.2.

3. Minimizing funding payments to position holders. See Subsection 2.1.

Minimizing costs while achieving the objectives outlined in point 1 above generally requires solving a sequential planning problem. For example, we may have a situation where we want to immediately adjust C to better match the oracle price, but the net operation costs may be lower if we first decreased k to reduce the cost of changing C.

2.1 Funding payments

The dAMM promises that position holders earn (or lose) a set amount of *funding* in each time period based on the price difference $p_t - p_t^\star$. When the long and short positions are equal, i.e. $l_t = s_t$, the losing side of the bet funds the winning side. When $l_t \neq s_t$, the dAMM either has to make up the difference or gets to collect the excess. The amount paid by the dAMM at time t is

$$(l_t - s_t) \cdot (p_t - p_t^\star)/24. \tag{2}$$

The sign of the funding payments can be either positive or negative, depending on the sign of the net position and the current mark price relative to the oracle price. For example, when $l_t - s_t \geq 0$, i.e. there are more longs than shorts, there are two cases:

$p_t^\star \leq p_t$. The longs are losing the bet, and have to provide funding for the shorts. Since there are more longs than shorts, the dAMM collects the excess funds provided by the longs.

$p_t^\star > p_t$. The shorts are losing the bet, and have to provide funding for the longs. However, since there are more longs than shorts, the perp needs to make up the difference in order to provide the promised amount of funding for the longs.

2.2 Cost of adjusting C and k

Operation costs are calculated by looking at the change in base asset value of the net market position before and after trading. Since adjusting C and k can increase the assets' trade value, we need to set aside additional funds to pay out traders if they sell their positions. In reality, the operation costs are rarely fully realized, since traders may not all immediately close their positions. Moreover the operation costs can be negative (the dAMM profits) if changing C and k reduces trade value and the traders close. In any case, the positive part of the operation cost represents the funds that need to be set aside in a given period to pay out traders in the case of a sell-off.

The base asset value of the net market position is given by $C_t \cdot \Delta y$, where Δy is computed using (1) at time $s = t$, using $\Delta x = l_t - s_t$. The cost of changing C_t to $C_{t+1} = C_t + u_{t,C}$ and k_t to $k_{t+1} = k_t + u_{t,k}$ is the difference in base asset value of the current net market position

$\Delta x = l_t - s_t$ before and after changing C_t and k_t. It is given by

$$g_t(C_t, k_t, u_{t,C}, u_{t,k}) = \frac{u_{t,C} k_t (l_t - s_t)/x_t - u_{t,C} u_{t,k} - C_t u_{t,k}}{x_t + l_t - s_t}, \tag{3}$$

and a derivation is provided in Appendix A.

2.3 Reserve Dynamics

At time t, the reserves are characterized by the constant product curve $x_t y_t = k_t$. Between time t and $t+1$, the value of x may change due to trading activity and adjustments to k, in that order. Let w_t denote a random variable that represents the net change in x due to trading activity. In our constant-product market, scaling k by a factor of η is equivalent to scaling both x and y by a factor of $\sqrt{\eta}$ each. If we additively perturb k_t by a control $u_{t,k}$, x_t is therefore scaled by a factor of $\sqrt{(k_t + u_{t,k})/k_t}$. This leads to the update equation

$$x_{t+1} = \sqrt{\frac{k_t + u_{t,k}}{k_t}} \cdot (x_t + w_t)\,. \tag{4}$$

Since we must have $x_{t+1} y_{t+1} = k_{t+1}$, y may be updated as $y_{t+1} = k_{t+1}/x_{t+1}$. Note that if $u_{t,k}$ is small relative to k_t, the dynamics are well approximated by the linear dynamics

$$x_{t+1} \approx x_t + w_t. \tag{5}$$

While this simplification neglects the effect of k on x, it is a reasonable approximation for the purposes of planning, since we would like $u_{t,k}$ to be small to minimize arbitrage opportunities and k is a product of two large asset reserve amounts.

3 MPC for dAMMs

At each time t, we solve a planning problem over the next H timesteps, using forecasts of the values of various signals at times $t, \ldots, t+H$. This gives a sequence of controls $u_t, u_{t+1}, \ldots, u_{t+H}$; we keep execute u_t at time t and discard the rest. In particular, we solve:

$$\begin{array}{ll} \text{minimize} & \sum_{\tau=t}^{t+H} \ell_\tau(C_\tau, k_\tau, u_{\tau,C}, u_{\tau,k}) \\ \text{subject to} & C_{\tau+1} = C_\tau + u_{\tau,C}, \\ & k_{\tau+1} = k_\tau + u_{\tau,k}, \quad \tau = t, \ldots, t+H. \end{array} \tag{6}$$

We treat the reserve amounts x_τ as fixed quantities that may be forecasted using the dynamics simplification (5) and predictions of the trading activity w_τ, i.e., using $\mathbb{E}[w_t]$ in the dynamics. We also assume that we have access to a forecast of the oracle price $p^\star_\tau$ and the net market position $l_t - s_t$. We now discuss our choice of the stage cost function ℓ_t.

The operating cost (9) and funding cost (2) consist of bilinear terms in the variables. For the latter, note that we may write the price as $p_\tau = C_\tau k_\tau / x_\tau^2$. We form a convex quadratic stage cost function by adding the following cost terms: $\alpha_\tau(C_\tau - \hat{C}_\tau)^2$, $\beta_\tau(k_\tau - \hat{k}_\tau)^2$, $\gamma_\tau u_{t,C}^2$, and $\delta_\tau u_{t,k}^2$. The first two terms track fixed trajectories $\hat{C}_\tau$ and $\hat{k}_\tau$, chosen to encourage some desirable behavior based on a forecast of the oracle price $p_t^\star$; an example is used in the experiments and described in Appendix D. The latter two terms discourage large changes in the price, which can lead to arbitrage opportunities. Adding the cost terms, we obtain the stage cost

$$\ell_\tau(C_\tau, k_\tau, u_{\tau,C}, u_{\tau,k}) = \frac{1}{2}\begin{bmatrix} C_\tau \\ k_\tau \\ u_{\tau,C} \\ u_{\tau,k} \end{bmatrix}^\top \begin{bmatrix} 2\alpha_\tau & a & 0 & b \\ a & 2\beta_\tau & c & 0 \\ 0 & c & 2\gamma_\tau & b \\ b & 0 & b & 2\delta_\tau \end{bmatrix} \begin{bmatrix} C_\tau \\ k_\tau \\ u_{\tau,C} \\ u_{\tau,k} \end{bmatrix} - 2(\alpha C_\tau \hat{C}_\tau + \beta k_\tau \hat{k}_\tau), \quad (7)$$

where

$$\begin{aligned} a &= -(\ell_\tau - s_\tau)/(24x_\tau^2), \\ b &= -1/(x_\tau + \ell_\tau - s_\tau), \\ c &= (\ell_\tau - s_\tau)/(x_\tau(x_\tau + \ell_\tau - s_\tau)). \end{aligned}$$

In order for the stage cost to be convex, α_τ, β_τ, γ_τ, and δ_τ should be chosen large enough to ensure that the symmetric matrix in (7) is positive semidefinite.

4 Experiments

We evaluated our approach against several baseline policies on simulated data. Our dAMM simulator and dAMM operating strategies were implemented in Julia, using `Convex.jl`[2].

Setup. The true oracle price $p_t^\star$ was generated as a random walk with normally distributed intervals. To generate "predictions" of the oracle price, we added noise to the true oracle price. We modeled the net market position to be inversely proportional to $p_t - p_t^\star$, plus noise. That is, traders are actively trying to improve their positions. We took $\mathbb{E}[w_t]$ to be the inverse of the mean net market position model (without noise added). Appendix D contains additional details, including the market parameter values used.

Baselines. We compared against several baselines based on heuristics currently used by the Drift dAMM team, as described in their documentation. The static baseline keeps C and k fixed. Ad-hoc (C only) and Ad-hoc (k only) perturb C and k when certain conditions are met; see Appendix C for details. Ad-hoc (both) employ both heuristics simultaneously. We compare the MPC approach described in Section 3 with a simplified version where the operating costs and funding payments are neglected, i.e. we set a, b and c in (7) to zero. We refer to this approach as MPC (price tracking only). The same lookahead H, cost terms α, β, γ, and δ, and market forecasts were used; see Appendix D.

[2]Code available at: `https://github.com/syanga/dAMMOpt`

Metrics. For each method, we computed the average funding and operation costs across timesteps. We only considered the positive part of the operation cost, i.e. $g_t(C_t, k_t, u_{t,C}, u_{t,k})_+$, since we assumed that traders would generally avoid immediately selling at a loss. Even if they did, the dAMM would profit. The operation costs correspond to the funds that need to be allocated in each period in case positions are closed and the dAMM needs to make up the cost; in reality they may not be fully realized. We also measured the mark-oracle price divergence $|p_t - p_t^\star|/p_t^\star$ to evaluate price tracking performance. Finally, we measured the slippage (lower is better); see Appendix B for details.

Policy	Cost (\$)	Price Divergence (%)	Slippage (%)
Static	13.447	2.990	0.333
Ad-hoc (C only)	15.838	0.175	0.332
Ad-hoc (k only)	37.421	3.058	**0.308**
Ad-hoc (C and k)	28.431	**0.171**	0.332
MPC (price tracking only)	6.73	0.786	0.333
MPC	**-0.103**	0.808	0.333

Table 1: Comparison of MPC and baseline methods.

Results. Table 1 compares the six strategies on three metrics. Each metric was computed by averaging the performance on 100 different generated oracle price trajectories, over 500 timesteps. The static baseline achieves moderate cost but mediocre price tracking. The Ad-hoc (C, both) approaches greatly improve the price tracking at the price of higher cost. The Ad-hoc (k only) approach gave consistently lower slippage, but at greater cost and poor mediocre tracking.

The MPC approaches brought significant cost savings, at the expense of a slightly higher price divergence relative to Ad-hoc (C, both). The MPC approach actually made a profit from funding payments, on average. A visual comparison of the price tracking performance between the static and MPC methods is illustrated in Figure 1.

5 Conclusion

In this work, we derived an MPC method for managing dynamic automated market makers, and showed that it can deliver strong performance while providing great cost savings. In future work, we hope to test our approach on publicly available trading data. In addition, we expect that data-driven hyperparameter tuning would be necessary for meeting the objectives and constraints associated with a real dAMM. We believe that optimization-based control schemes such as the one we proposed can be used to build more effective dAMMs in the future.

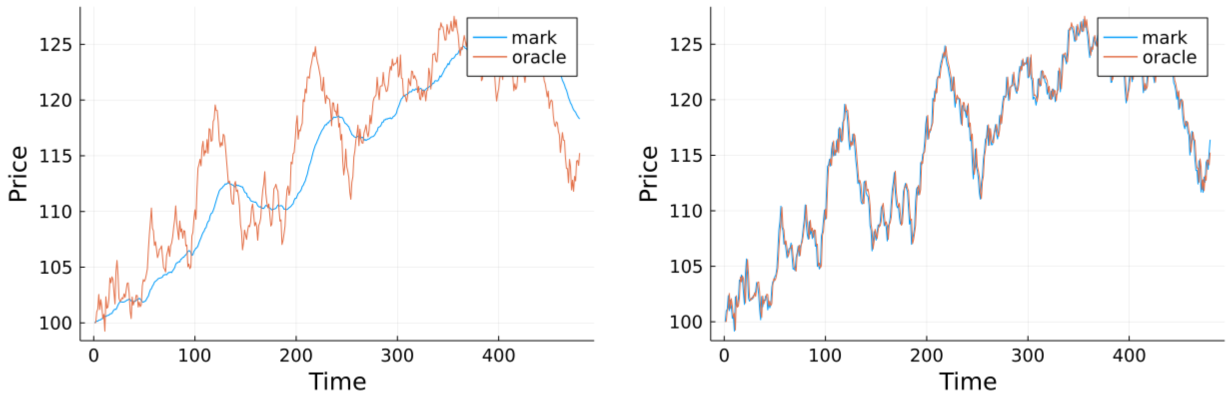

Figure 1: Oracle price tracking comparison between the static baseline (left) and MPC (right).

References

[AKC+19] Guillermo Angeris, Hsien-Tang Kao, Rei Chiang, Charlie Noyes, and Tarun Chitra. An analysis of uniswap markets, 2019.

[BBD+17] Stephen Boyd, Enzo Busseti, Steven Diamond, Ronald N Kahn, Kwangmoo Koh, Peter Nystrup, and Jan Speth. Multi-period trading via convex optimization. *arXiv preprint arXiv:1705.00109*, 2017.

[CKA+22] Tarun Chitra, Kshitij Kulkarni, Guillermo Angeris, Alex Evans, and Victor Xu. Defi liquidity management via optimal control: Ohm as a case study. 2022.

[Lab21] Drift Labs. Drift: Perpetual swaps on Solana, 2021.

A Derivation of Operation Costs

The base asset value of a quantity Δx may be written as

$$C_t \Delta y = C_t \frac{y_t(x_t + \Delta x) - k_t}{x_t + \Delta x}, \tag{8}$$

by rearranging (1).

The cost of changing C_t to $C_{t+1} = C_t + u_{t,C}$ and k_t to $k_{t+1} = k_t + u_{t,k}$ is the difference in base asset value of the current net market position $\Delta x = l_t - s_t$ before and after changing C_t and k_t. In other words,

$$\begin{aligned} g_t(C_t, k_t, u_{t,C}, u_{t,k}) &= (C_t + u_{t,C}) \frac{y_t(x_t + l_t - s_t) - k_t - u_{t,k}}{x + l_t - s_t} - C_t \frac{y_t(x_t + l_t - s_t) - k_t}{x_t + l_t - s_t} \\ &\overset{(a)}{=} (C_t + u_{t,C}) \frac{y_t(l_t - s_t) - u_{t,k}}{x + l_t - s_t} - C_t \frac{y_t(l_t - s_t)}{x_t + l_t - s_t} \\ &= u_{t,C} \frac{y_t(l_t - s_t) - u_{t,k}}{x + l_t - s_t} - \frac{C_t u_{t,k}}{x + l_t - s_t} \\ &\overset{(b)}{=} \frac{u_{t,C} k_t (l_t - s_t)/x_t - u_{t,C} u_{t,k} - C_t u_{t,k}}{x_t + l_t - s_t}. \end{aligned} \tag{9}$$

where in (a) and (b) we have used the fact that $x_t y_t = k_t$.

B Slippage

When traders enter the market, their realized execution price is generally worse than the market price at the time of execution. This phenomenon is known as slippage, and results from the properties of the constant product curve.

The slippage is defined as follows. Suppose that a trader puts in a single unit of the base asset ($\Delta x = 1$). The realized value of that base asset is, by (8) and the fact that $x_t y_t = k_t$, given by

$$C_t \frac{y_t(x_t + 1) - k_t}{x_t + 1} = \frac{C_t y_t}{x_t + 1}.$$

The slippage is defined as the ratio between the realized value and the market price $C_t y_t / x_t$:

$$\text{slippage} = 1 - \frac{C_t y_t / (x_t + 1)}{C_t y_t / x_t} = 1 - \frac{x_t}{x_t + 1} = \frac{1}{x_t + 1} \tag{10}$$

The slippage can be reduced by increasing x_t. The perp can achieve this by increasing k, since scaling k by a factor of $\eta > 1$ scales both x and y by a factor of $\sqrt{\eta}$ each. While increasing k incurs a cost, the perp is incentivized to reduce slippage, since high slippage discourages traders from engaging with the dAMM.

C Baseline Details

Ad-hoc (C only). The peg C is adjusted such that p_{t+1} matches the forecast of the oracle price $\hat{p}^\star_{t+1}$. Specifically, we set $u_{c,t} = \hat{p}^\star_{t+1} x_t^2/k_t - C_t, u_{k,t} = 0$. This adjustment only occurs if the mark-oracle divergence exceeds a liquidation limit (which we set to 5%), or if the cumulative forecasted funding costs over the horizon (funding rate bias) is greater than the current cost of repegging to the oracle.

Ad-hoc (k only). For the formulaic curve invariant, we first check if there has been persistent bias in k over the past $H' = 10$ timesteps. The bias is defined as the cumulative mark-oracle difference for past timesteps where longs exceeds shorts. If the absolute value of the bias exceeds a limit $B = \$2$ and and the percentage of past timesteps where the funding imbalance is positive is sufficiently large ($> 70\%$), then we set $u_{t,c} = 0, u_{t,k} = \rho b_k$, where b_k is the k-bias for a parameter $\rho = 45$. Intuitively, if there are more longs than shorts, a positive mark-oracle difference means that k was set too small since it the longs moved the price too far up. Conversely, a negative mark-oracle difference means that k was set too large since traders who long the dAMM were unable to match the mark to the oracle due to the deep virtual liquidity.

D Simulation Details

The oracle price was generated using a random walk: $p^\star_{t+1} = p^\star_t + \epsilon_t$, where $\epsilon_t \sim \mathcal{N}(0, 1)$ and $p^\star_0 = 100$. To simulate a forecast of the oracle price, we sampled $\hat{p}^\star_t \sim \mathcal{N}(p^\star_t, 0.1^2)$. To generate the net market position, we used $l_t - s_t \sim \mathcal{N}(5(p_t - p^\star_t)/p^\star_t, 0.1^2)$. The estimated net market position (and estimate of $-\mathbb{E}[w_t]$ was taken to be $5(p_t - p^\star_t)/p^\star_t$.

For MPC, we used a lookahead length of $H = 15$. The estimated trajectory of the base asset reserve was taken to be $\hat{x}_{\tau+1} = \hat{x}_\tau + 5(p_t - p^\star_t)/p^\star_t$, where $\hat{x}_{\tau-1} = x_{t-1}$. The values of $\alpha, \beta, \gamma, \delta$ in (7) were set to be $\alpha = 1$, $\beta = 10^{-6}$, $\gamma = 10^{-4}$, and $\delta = 0.05$. Since values of k tend to be much larger than C, β was set to be much smaller than α.

We chose the nominal trajectories $\hat{k}_\tau$ and $\hat{C}_\tau$ as follows. First, we would like k to increase slowly. This means that the size of the reserves is increasing, and that traders will experience less slippage. Therefore, we chose $\hat{k}_\tau = k_t + \lambda(\tau - t)$, where $\lambda = 1$. Next, we set $\hat{C}_\tau$ such that $\hat{C}_\tau \hat{k}_\tau / \hat{x}_\tau = \hat{p}^\star_\tau$, so that the nominal trajectory tracks the predicted oracle price.