PRIMEFI AMM Whitepaper

February 8, 2023

Abstract

This paper elucidates the technical specification for the PrimeFi Automated Market Maker (AMM) that will first serve as a stableswap. In particular, we described an AMM design that is single-sided, enables “open liquidity pool”, and the first in kind to construct a single-variant slippage curve instead of using invariant curves.

1 Existing Designs and Problems

Existing Automated Market Makers (AMM), such as Uniswap, Curve, require liquidity providers to provide liquidity of tokens in pair or in a bundle prepared to swap within the provided liquidity pool. The liquidity providers are equivalent to short the volatility of the tokens being supplied and suffer from impermanent loss as a consequence. In the context of stableswap, this will mean that the liquidity provider may get back asset as a combination of tokens that are different from what he / she has original provided.

1.1 Invariant Curve

Uniswap, Curve, etc, relies on the use of invariant curves to govern the relationship between tokens in a particular liquidity pool. Most of them utilize Constant Function Market Maker (CFMM) (Angeris Chitra, 2020). Define the token space T to be a collection of sets of token amounts, the function $f : T \to \mathbb{R}$ maps the vectors of token amounts to a real number. Under CFMM, assume no one provide extra liquidity or withdraw liquidity, $f(x_1, x_2, ..., x_n) = k$ for some $k \in \mathbb{R}$ after any swap within the liquidity pool of tokens $x_1, x_2, ..., x_n$. AMM of different protocols mostly focus on varying the invariant curve to achieve certain desirable properties.

mStable specifies their invariant curve with a Constant Sum Market Maker (CSMM), that is, $f(x_1, x_2, ..., x_n) = \sum_{i=1}^{n} x_i = k$ (mStable, 2021). This design enables users to swap tokens with a fixed ratio. Uniswap, however, allows swapping between tokens with high volatility. Therefore, they utilize an invariant curve with a Constant Product Market Maker (CPMM), that is $f(x_1, x_2, ...x_n) = \prod_{i=1}^{n} x_i = k$ (Adams, 2019). The curve introduces slippages into the design. The marginal cost of swapping into a certain token would increase with the amount swapped to that token. Specializing in stablecoins swap, Curve defines an invariant curve with a mix of CSMM and CPMM, which allows users to swap under almost a constant 1:1 rate. Slippages are significant only when a significant amount is traded. The function is specified as $A_n n^n \sum x_i + D = ADn^n + \frac{D^{n+1}}{n^n \prod x_i}$, where D is a constant and A is an amplification coefficient, which will be adjusted according to the current pool status (Egorov, 2019).

1.2 Shortcomings

All of the aforementioned designs require liquidity providers to prepare tokens in pair or in bundle for swapping to happen within pools. The exchange rate risks are shifted to the liquidity providers (i.e. divergence / impermanent loss). To the liquidity providers, this is not ideal and not convenient to provide a bundle of tokens simultaneously.

Besides, CFMM typically are comprised of separated, closed pools, resulting in liquidity fragmentation. For instance, while USDT may exist in both ”ETH-USDT” and ”USDT-USDC” pool, the USDT among the two pools cannot be shared with each other, which reduced the system’s capital efficiency.

An additional shortcoming for stableswap like Curve or Saddle is that their stableswap invariant requires all tokens within the same pool to have the same amount of liquidity to reach its equilibrium state, making the least popular token the bottleneck for the growth of the pool. Moreover, it is impossible to add new tokens into existing pool (particularly if the new token’s liquidity is not comparable to other existing tokens in the pool), which hindering the scalability of the protocol.

2 PrimeFi Design

PrimeFi allows users to provide unilateral liquidity. Instead of pools of token pairs and bundles, PrimeFi uses accounts of token to record assets and liabilities. Such design allows singled-sided liquidity provision.

PrimeFi also uses coverage ratio as the input parameter for our AMM (instead of liquidity), hence removes the same liquidity equilibrium constraint from Curve's stableswap invariant, allows token to grow organically based on its nature demand and supply.

This design also allows new tokens to be added into existing pool, greatly enhance the scalability and capital efficiency of the protocol. For example, while the protocol may start with the base 3 tokens (USDT, USDC, DAI), we may gradually add more tokens into the big pool (such as TUSD, FRAX) for shared liquidity. We refer this design as ”Open Liquidity Pool”.

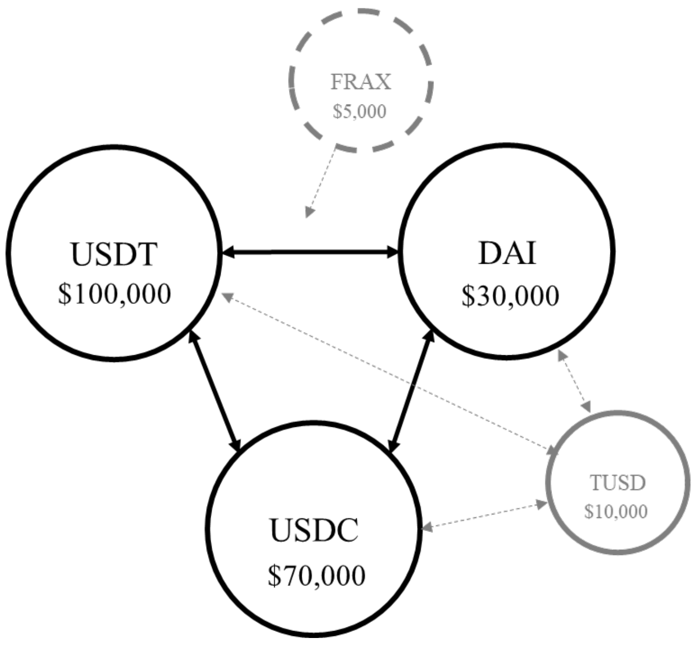


Figure 1: Open liquidity pool

2.1 Price Oracle

While the exchange rate for tokens in stableswap should be pegged, it is possible that some of them becomes unpegged due to unforeseen reasons. Hence, PrimeFi will track each token's exchange rate via price oracle like Chainlink. Define the exchange rate f_i to be the price of token i in terms of USD. For a swap from token i to token j, the price of token i in terms of token j is defined to be $f_{i\to j} = \frac{f_i}{f_j}$. Under a perfect competition market, selling 1 token i would give the seller $f_{i\to j}$ of token j.

2.2 Solvency Risk

For a unilateral liquidity provision, the liquidity provided would become liability to the protocol. We define this amount to be L_i for token i account. The current amount of token i held by the protocol would be the protocol's asset in token i account. We define this amount to be A_i for token i account. The coverage ratio r_i is defined to be $\frac{A_i}{L}$, which is a measure of default risk of token i account. A higher coverage ratio indicates a lower default risk. When $r_i = 1$, all of the liability in token i account could be settled by the asset held by the protocol. The coverage ratio is an important parameter to our protocol since it needs to be maintained above certain level to avoid default. In particular, the amount of a withdrawal request exceeds the amount of asset in a token account.

2.3 Terminal Exchange Rate

Slippage is not built-in under PrimeFi AMM design. It is however necessary to penalize huge volume swaps which deviate our financial position and pose a default risk to PrimeFi. PrimeFi AMM therefore would take away parts of the receivables from the swaps. The terminal exchange rate f^* is defined to be

$$f^*_{i\to j} = f_{i\to j}(1 - S_{i\to j})(1 - h),$$

where $f_{i\to j}$ is obtained from external price oracles, $S_{i\to j}$ and h are the average slippage and haircut respectively, to be defined later.

In order to discourage huge transaction that substantially lowers the coverage ratio of any pool, the slippage would be a function of coverage ratio of the two corresponding pools. After a swap from Δ_i token i to token j, $A'_i = A_i + \Delta_i$ and $A'_j = A_j - f^*_{i\to j}\Delta_i$. Then, we have $r'_i = \frac{A_i + \Delta_i}{L_i} > \frac{A_i}{L_i} = r_i$ while $r'_j = \frac{A_j - f^*_{i\to j}\Delta_i}{L_j} < \frac{A_j}{L_j} = r_j$. An extremely distributed coverage ratios would harm the solvency of the AMM, the slippage is therefore designed to penalize actions that deviate coverage ratios of two pools and incentivize actions that converge two coverage ratios.

2.4 Account Slippage

PrimeFi is the first in kind to use a single-variant slippage function instead of invariant curves. Define account slippage function to be $g : \mathbb{R}^+ \to [0, 1]$ that maps coverage ratio of a token account to a slippage value which lies between 0 and 1. The function g should satisfy the below properties:

1. g is continuously differentiable
2. $g' \leq 0$ if g' exists
3. g is a weakly convex function

Its derivative g' is called marginal slippage in our context. Note that g' indicates the slippage for an extra coverage ratio change at a particular coverage ratio. g is a decreasing function since slippage is a disincentive to reduce coverage ratio. When the coverage ratio of an ac-

6

count recovers, the protocol could offer a lower slippage to the users. g is a weakly convex function since the cost of reducing coverage ratio would increase when coverage ratio of an account is low. Convexity is also essential to some of the desirable properties of AMM to be discussed later.

Notice that for any function that is continuous almost everywhere and satisfied the aforementioned properties of g', we could generate an account slippage function by Riemann integrating the function g' according to Lebesgue's Theorem. Hence, we first attempt to define a marginal slippage function, that is

$$g'(r) = \begin{cases} -1, & \text{for } -\frac{kn}{r^{n+1}} < -1 \\ -\frac{kn}{r^{n+1}}, & \text{for } -\frac{kn}{r^{n+1}} \in [-1, 0] \end{cases}$$

$$\Rightarrow g(r) = \int g'(r)dr = \begin{cases} -r + C_1, & \text{for } -\frac{kn}{r^{n+1}} < -1 \\ \frac{k}{r^n} + C_2, & \text{for } -\frac{kn}{r^{n+1}} \in [-1, 0] \end{cases}$$

k and n are fixed parameters to be specified. Our research found that $k = 0.00002$ and $n = 7$ could be a competent choice of parameters.

In the function $-\frac{kn}{r^{n+1}}$, there exists some $r > 0$ that makes the function value smaller than -1. This is however not optimal since any extra reduction in coverage ratio would reduce the amount of token received. More concretely, swapping more token might result in less receivable token. Marginal slippage therefore bounded below by -1. For simplicity, we define threshold coverage ratio to be r^* such that

$$-\frac{kn}{r^{*n+1}} = -1$$

$$\Rightarrow r^* = \sqrt[n+1]{kn}$$

Figure 2: Slippage Function

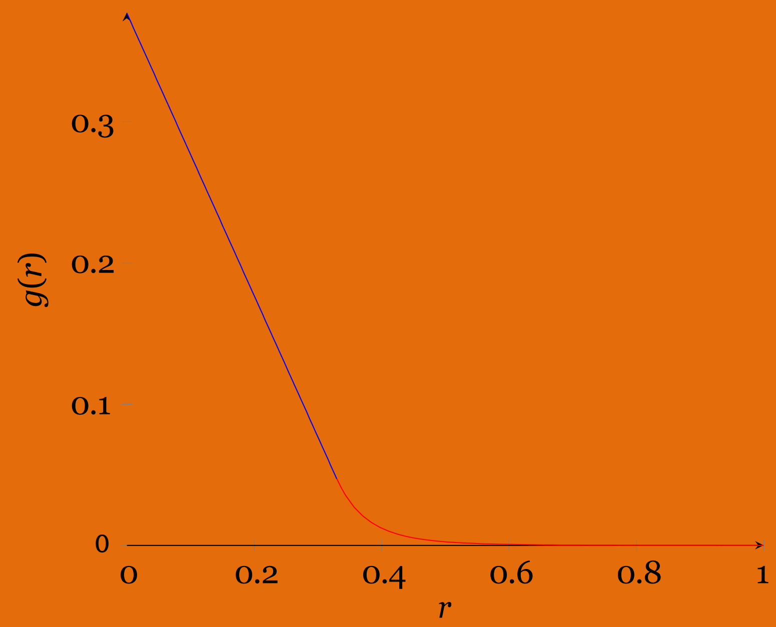
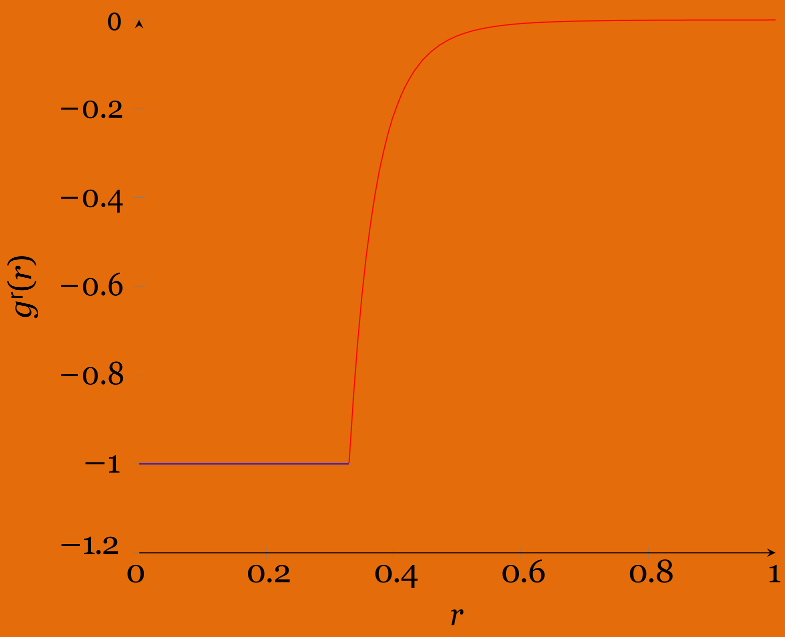


Figure 3: Marginal Slippage Function



8

Any action that attempts to further reduce coverage ratio of an account with coverage ratio equal to r^* would result in 100% slippage, excluding the compensation effect from increasing the coverage ratio of the from token account. Let j to be the from token and $r_j \to \infty$, then we have the swap slippage to be -1, which would be discussed in section 2.5.

Setting C_2 in $g(r)$ to be 0, we have

$$g(r) = \begin{cases} -r + C_1, & \text{for } r < r^* \\ \dfrac{k}{r^n}, & \text{for } r \geq r^* \end{cases}$$

To preserve differentiability of g at the point $r = r^*$, it is necessary to have

$$\lim_{r \uparrow r^*} g(r) = \lim_{r \downarrow r^*} g(r) = g(r^*)$$

$$\Rightarrow -\sqrt[n+1]{kn} + C_1 = \frac{k}{(kn)^{\frac{n}{n+1}}}$$

$$\Rightarrow C_1 = \frac{k^{\frac{1}{n+1}}}{n^{\frac{n}{n+1}}} + \sqrt[n+1]{kn} = r^*\left(1 + \frac{1}{n}\right)$$

$$\Rightarrow g(r) = \begin{cases} -r + r^*\left(1 + \dfrac{1}{n}\right), & \text{for } r < r^* \\ \dfrac{k}{r^n}, & \text{for } r \geq r^* \end{cases}$$

The slippage function g is a fixed reference point for slippage. In any swap that account for token i involved, there will be an initial coverage ratio r_i and final coverage ratio r_i'. The account slippage incurred in account for token i should be defined by the definite integral along the swap path S on token i normalized by the change in coverage ratio, that is

$$S_i := \frac{\int_S g'(r)dr}{r_i' - r_i} = \frac{g(r_i') - g(r_i)}{r_i' - r_i}$$

9

2.5 Swap Slippage

The account slippage defined above is not the final slippage $S_{i\to j}$ that users would encounter when swapping token i to token j. A swap path from token i to token j should at least involve two tokens. As proved in section 2.3, when coverage ratio of the account of a token worsened in a swap, the coverage ratio of the swap counterpart's account should improve. The account slippage on token i is a compensation to the user while the account slippage on token j is a punishment to the user since $r'_i > r_i$ and $r'_j < r_j$. Therefore, we define the swap slippage $S_{i\to j}$ to be

$$S_{i\to j} = S_i + (-S_j) = S_i - S_j$$

2.6 Incentives for Convergence of Coverage Ratio

The swap slippage defined above could incentivize convergence of coverage ratios and disincentivize divergence of coverage ratios between accounts of two tokens. Mathematically, for any swap from token i to token j, if $r'_i, r'_j \in (r_i, r_j)$, we have $S_{i\to j} < 0$. If $r'_i, r'_j \notin (r_j, r_i)$, we have $S_{i\to j} > 0$.

Proof We first prove the convergent part. First, $r_i < r_j$, otherwise convergence of two ratios could not be achieved by swapping token i to token j. By convexity of g, if $r'_i, r'_j \in (r_i, r_j)$, we have

$$\frac{g(r'_i) - g(r_i)}{r'_i - r_i} \le \frac{g(r_j) - g(r_i)}{r_j - r_i} \le \frac{g(r'_j) - g(r_j)}{r'_j - r_j} < 0$$

$$\Rightarrow S_{i\to j} = S_i - S_j = \frac{g(r'_i) - g(r_i)}{r_i - r'_i} - \frac{g(r'_j) - g(r_j)}{r'_j - r_j} < 0$$

We then prove the divergent part. Note that even if $r_i < r_j$, we could have $r^{\mathrm{r}}_{i}, r^{\mathrm{r}}_{j} \notin (r_i, r_j)$ in the opposite direction by swapping token i to token j. That involves a combination of convergence, reaching the equilibrium state $r'_i = r'_j$ and then divergence. We can achieve the same result even for that case. Here, we only focus on pure divergence, that is, $r'_i > r_i$ and $r'_j < r_j$. Again, by convexity of g, we have

$$\frac{g(r'_j) - g(r_j)}{r'_j - r_j} \le \frac{g(r_i) - g(r_j)}{r_i - r_j} \le \frac{g(r'_i) - g(r_i)}{r'_i - r_i} < 0$$

$$\Rightarrow S_{i\to j} = S_i - S_j = \frac{g(r'_i) - g(r_i)}{r'_i - r_i} - \frac{g(r'_j) - g(r_j)}{r'_j - r_j} > 0$$

2.7 Equilibrium State

It is very important to note that there is a pairwise equilibrium state $r_i = r_j$ for any account pairs i, j.

Proof For any dynamic swap from Δ_i token i to token j, if $r_i < r_j$, we have

$$r'_i = \frac{A_i + \Delta_i}{L_i}, \quad r'_j = \frac{A_j - \Delta_i[f_{i\to j}(1 - S_{i\to j})(1 - h)]}{L_j}$$

Note that r'_i is a continuously increasing function of Δ_i and r'_j is a continuously decreasing

i j

function of Δ_i. $r'_j - r'_i$ is therefore a continuously decreasing function with 0 in its co-domain. By intermediate value theorem, there exists a state where $r'_i = r'_j = r_e$. Any swap prior to this state should receive a marginal compensation for convergence and any swap later than this state should receive a marginal punishment for divergence. Hence, this state is concluded to be the equilibrium state.

A pairwise equilibrium state $r_i = r_j = r_e$ would imply the global equilibrium state to be $r_1 = r_2 = r_3 = ... = r_n = r_e$.

3 Desirable Properties of AMM

There are three desirable properties to be achieved by an AMM: path-independent, liquidity sensitive and no arbitrage. Othman et al. (2013) have shown that it is impossible to achieve three of them simultaneously. PrimeFi AMM achieves path-independent and liquidity sensitive and utilize extra mechanism to block arbitrage to be discussed in later sections.

3.1 Preliminary

The discussion below would be quite complex with the use of different notations. This section is to help readers to develop a thorough understanding of the use of notations.

3.1.1 Direction of Swap

Direction of swap indicates the from token and the to token. For example, a swap from token i to token j is denoted as

$$i \rightarrow j$$

3.1.2 Swap Amount

Swap amount consists of the amount of the from token and the amount of the to token. We always view it from the perspective of the protocol. That is, when a user swap Δ_i amount of token i to Δ_j amount of token j, since the from protocol perspective, there will be $|\Delta_i|$ more token i asset and $|\Delta_j|$ less token j asset, we have $\Delta_i > 0$ and $\Delta_j < 0$. Moreover, given an exchange rate $f_{i\to j}$. We have $\Delta_j = -f_{i\to j}\Delta_i$. Hence, $r_i^{\text{r}} = \frac{A_i + \Delta_i}{L_i}$ and $r_j^{\text{r}} = \frac{A_j + \Delta_j}{L_j} = \frac{A_j - f_{i\to j}\Delta_i}{L_j}$.

3.1.3 Swap Path

Swap path is defined to be a path that indicates the starting and ending vectors of coverage ratios (usually of 2 tokens). We denote it as

$$S : (r_i, r_j) \to (r'_i, r'_j),$$

where (r_i, r_j) is the starting ratios and (r'_i, r'_j) is the ending ratios.

3.1.4 Swapping fee

Swapping fee is the amount that paid by the user as slippage for a swap. We define swapping fee in terms of token j with swap direction $i \to j$ and swap path $(r_i, r_j) \to (r'_i, r'_j)$ to be

$$F_{s,i\to j}((r_i, r_j) \to (r'_i, r'_j)) := \Delta_j S_{i\to j} = -f_{i\to j}\Delta_i S_{i\to j}$$

Note that $\Delta_j < 0$. If $S_{i\to j} > 0$, we will have the swapping fee to be a negative number and vice versa. This is to indicate how much token j is deducted as slippage.
If we want to calculate swapping fee in terms of other tokens, we should first calculate the slippage in terms of to token to preserve universal representation of the positive and negative sign. Hence, the slippage fee in terms of token i for $F_{s,i\to j}((r_i, r_j) \to (r'_i, r'_j))$ is

$$f_{j\to i}F_{s,i\to j}((r_i, r_j) \to (r'_i, r'_j)) = -\Delta_i S_{i\to j}$$

13

3.2 Path-independence

Swapping fee is path-independent given a fixed exchange rate.

Proof For any swap path $S : ((r_i, r_j) \to (r'_i, r'_j))$ from token i to token j, we have

$$
\begin{aligned}
F_{s,i\to j}((r_i, r_j) \to (r'_i, r'_j)) &= \Delta_j S_{i\to j} \\
&= \Delta_j \left(\frac{L_i \int_S g'(r_i) dr_i}{\Delta_i} - \frac{L_j \int_S g'(r_j) dr_j}{\Delta_j} \right) \\
&= -f_{i\to j} L_i \int_S g'(r_i) dr_i - L_j \int_S g'(r_j) dr_j \\
&= \int_S (-f_{i\to j} L_i g'(r_i), -L_j g'(r_j)) \cdot (dr_i, dr_j)
\end{aligned}
$$

Let $\mathbf{F}(r_i, r_j | i \to j) = (-f_{i\to j} L_i g'(r_i), -L_j g'(r_j))$, notice that $\mathbf{F}(r_i, r_j | i \to j) = \nabla h(r_i, r_j)$, where $h(r_i, r_j) = -f_{i\to j} L_i g(r_i) - L_j g(r_j)$ and h is continuous. Together with $(r_i, r_j) \in \mathbb{R}^2_+$, we have F_s is path-independent of S.

On top of the proof, for the propose of rigour, denote $\mathbf{F}(r_i, r_j | i, j)$ to be the integrand of the line integral without specifying the swap direction but the involved token i and j.

3.2.1 Symmetry of Slippage

Next, we check the symmetry of the slippage. For a swap from token i to token j which sends (r_i, r_j) to (r'_i, r'_j), the slippage fee in terms of token j is

$$
\begin{aligned}
F_{s,i\to j}((r_i, r_j) \to (r'_i, r'_j)) &= \int_S (-f_{i\to j} L_i g'(r_i), -L_j g'(r_j)) \cdot (dr_i, dr_j) \\
&= -\int_{r_i}^{r'_i} f_{i\to j} L_i g'(r_i) dr_i - \int_{r_j}^{r'_j} L_j g'(r_j) dr_j
\end{aligned}
$$

For a swap from token j to token i which sends (r'_i, r'_j) to (r_i, r_j), the slippage fee in terms of token j is

$$F_{s,j\to i}((r'_i, r'_j) \to (r_i, r_j)) = f_{i\to j} \int_{S'} (-L_i g'(r_i), -f_{j\to i} L_j g'(r_j)) \cdot (dr_i, dr_j)$$
$$= -\int_{r'_i}^{r_i} f_{i\to j} L_i g'(r_i) dr_i - \int_{r'_j}^{r_j} L_j g'(r_j) dr_j$$
$$= -F_{s,i\to j}$$

Hence, for any closed swap path O, the total slippage is 0.

Proof For any closed swap path O, there exists at least one swap turning point (i.e. $i \to j$ to $j \to i$), otherwise a closed loop could not formed. Let the last turning point to be (r'_i, r'_j). Denote the two swap path $S : (r_i, r_j) \to (r'_i, r'_j)$ and $S' : (r'_i, r'_j) \to (r_i.r_j)$. Then, by path independence and symmetry,

$$F_s(O) = \oint_O \mathbf{F}(r_i, r_j | i, j) \cdot (dr_i, dr_j)$$
$$= F_{s,i\to j}\left((r_i, r_j) \to (r'_i, r'_j)\right) + F_{s,j\to i}\left((r'_i, r'_j) \to (r_i, r_j)\right)$$
$$= 0$$

3.2.2 Slippage Resistance

Although in the previous section, we have shown that the slippage fee is independent of swap path S. Notice that the S here is given by

$$(r_i, r_j) \to (r'_i, r'_j) = \left(\frac{A_i + \Delta_i}{L_i}, \frac{A_j - \Delta_i f_{i\to j}}{L_j}\right)$$

15

However, in our AMM system, the swap from token *i* to token *j* is not using exchange rate from the external price oracle but the terminal exchange rate given by

$$f^*_{i\to j} = f_{i\to j}(1 - S_{i\to j})(1 - h)$$

Hence the terminal swap path derived from S would become S' given by

$$(r_i, r_j) \to \left(r'_i, r''_j\right) = \left(\frac{A_i + \Delta_i}{L_i}, \frac{A_j - \Delta_i f^*_{i\to j}}{L_j}\right)$$

From Section 2.6, we know that for any swap from token *i* to token *j* that intends to diverge slippage ratios of two account would have $S_{i\to j} > 0$. Hence $f^*_{i\to j} < f_{i\to j}$. As a result, we have $r''_j > r'_j$. Similarly, for swaps that intend to converge two coverage ratios, we have $r''_j < r'_j$.

We revisit the closed path swap S. For any closed path swap S that diverge two ratios then converge by swapping token *i* to token *j*, given by

$$(r_i, r_j) \to \left(r'_i, r'_j\right) \to (r_i, r_j)$$

The first-level actual swap path, in which r''_i is the expected coverage ratio change after the second swap. The swap path is given by

$$(r_i, r_j) \to \left(r'_i, r''_j\right) \to (r''_i, r_j)$$

Note that $r''_i > r_i$.

Proof Note that since $f_{i\to j}$ remains unchanged. Given the amount of token *i* expected to be received by the later swap $\left(r'_i, r''_j\right) \to (r''_i, r_j)$ is $f_{j\to i}\Delta^*_j = -f_{j\to i} f^*_{i\to j}\Delta_i$, where Δ_i is the initial swap amount of the first part of the path $(r_i, r_j) \to \left(r'_i, r'_j\right)$ and $f^*_{i\to j}$ is the terminal

exchange rate of the first swap. Then we have

$$-f_{j\to i}f^{*}_{i\to j}\Delta_i \;=\; -\frac{f^{*}_{i\to j}}{f_{i\to j}}\Delta_i \;<\; |\Delta_i|$$

$$\Rightarrow r''_i = \frac{A_i + \Delta_i - f_{j\to i}f^{*}_{i\to j}\Delta_i}{L_i} > \frac{A_i}{L_i} = r_i$$

The second-level actual swap path, in which $r'''_i < r''_i$ due to compensation is the actual coverage ratio change after the second swap, is given by

$$(r_i, r_j) \to (r'_i, r''_j) \to (r'''_i, r_j)$$

Define $S' : (r'_i, r''_j) \to (r''_i, r_j)$. The actual slippage fee is then given by

$$\begin{aligned}
F_s(S \cup S') &= \int_S (-f_{i\to j}L_i g'(r_i), -L_j g'(r_j)) \cdot (dr_i, dr_j) + f_{i\to j}\int_{S'} (-L_i g'(r_i), -f_{j\to i}L_j g'(r_j)) \cdot (dr_i, dr_j)\\
&= -\int_{r_i}^{r'_i} f_{i\to j}L_i g'(r_i)dr_i - \int_{r_j}^{r'_j} L_j g'(r_j)dr_j + \left(-\int_{r''_i}^{r'_i} f_{i\to j}L_i g'(r_i)dr_i - \int_{r_j}^{r''_j} L_j g'(r_j)dr_j\right)\\
&= F_{s,i\to j}\left((r_i, r_j) \to (r'_i, r'_j)\right) + F_{s,i\to j}\left((r''_i, r_j) \to (r'_i, r''_j)\right)\\
&< 0
\end{aligned}$$

Hence, the protocol is receiving a positive income from the users over the whole swap. Hence, $r'''_i > r_i$ due to the increment of asset. Note that the initial state would be $r_1 = r_2 = r_3 = \ldots = r_n = 1$. So the first step must be a diverging move. Given that, we would have at least one of the account's coverage ratio is greater than 1 after a series of closed path swap. The equilibrium coverage ratio r_e would be increasing and greater than 1.

3.3 Liquidity Sensitive

Given a fixed amount of swap from Δ_i of token i to token j, if L_j is higher, the slippage would be smaller.

Proof

$$F_s = \int_S (-f_{i\to j} L_i g'(r_i), -L_j g'(r_j)) \cdot (dr_i, dr_j)$$

$$\begin{aligned}
\frac{\partial F_s}{\partial L_j} &= \frac{\partial}{\partial L_j} \int_S (-f_{i\to j} L_i g'(r_i), -L_j g'(r_j)) \cdot (dr_i, dr_j) \\
&= \frac{\partial}{\partial L_j} \int_{r'_j}^{r_j} -L_j g'(r_j) dr_j \\
&= -\int_{r'_j}^{r_j} g'(r_j) dr_j \\
&= g(r_j) - g(r'_j) \\
&< 0 \text{ , since } g \text{ is a decreasing function.}
\end{aligned}$$

3.4 Haircut

Haircut h is a fee to be collected by the protocol to support and encourage the operation of the platform. This amount varies on the swap action carried out by the users. In particular, we are getting a bit more from users whose swap made the coverage ratios of two accounts diverge.

3.5 Retention Ratio

We have retention ratio $r \in (0, 1)$ to be the proportion of the haircut retains in the AMM system. $(1 - r)$ will be the proportion of the haircut will be distributed to the liquidity providers of the protocol as a share of fees. Consider a haircut h_i received for token i account, then $\Delta A_i = h_i$ but $\Delta L_i = (1 - r)h_i$. A higher coverage ratio for token i account is immediately observed after receiving a haircut of h_i and retaining rh_i of that.

4 Withdrawal Arbitrage

Withdrawal Arbitrage exists in our design so it is necessary to create a withdrawal fee to prevent such attack.

4.1 Arbitrage Procedure

Consider the account for token i, assume $r_i < 1$ and a user have deposited or going to deposit a certain amount D_i into the token i account.

1. User could swap y token j to $\bar{y}$ token i. The new coverage ratio for token i would be $r_i' = \frac{A_i - \bar{y}}{L_i}$. User pays the slippage for lowering r_i.

2. User withdraws D_i from token i account. The new coverage ratio decreases further to $r_i'' = \frac{A_i - \bar{y} - D_i}{L_i - D_i}$.

3. User reverses the swap in 1. to gain the slippage of pushing r_i to a higher level. User aims to restore the coverage ratio of token j account. Due to the altered state of coverage ratio, r_i'', the required amount for reversal swap, y', tends to be different from the initial amount $\bar{y}$. The final coverage ratio of token i account would then be $r_i^* = \frac{A_i - D_i - (\bar{y} - y')}{L_i - D_i}$.

The swap and withdrawal path is given by

$$S : (r_i, r_j) \xrightarrow{\text{swap}} (r'_i, r'_j) \xrightarrow{\text{withdrawal}} (r''_i, r'_j) \xrightarrow{\text{swap}} (r^*_i, r_j)$$

There is a key condition required for finding the amount of token i, y', needed for retrieving the target amount of token j, y, through the reversal swap:

$$y = (1 - S'_{i\to j})y'$$

Thus, the slippage from the action in terms of token i would be

$$= -yS_{j\to i} + f_{j\to i}(-y' f_{i\to j})S'_{i\to j}$$
$$= -y'(S_{j\to i} + S'_{i\to j} - S_{j\to i}S'_{i\to j})$$

That means the user could gain from this strategy if the overall slippage is greater than 0.

4.2 Withdrawal Fee

This section illustrates a highly simplified version of withdrawal fee that can outweigh the arbitrage profit.

For simplicity in preliminary construction, the following withdrawal fee neglects the quadratic term in slippage gain as shown above, which gauges the impact of the first slippage exerted on the second slippage. The required condition for restoring the coverage ratio of token j account is neglected here. Also, y is assumed to be equal to y'. These simplifications may cause inconsistency in the ideal arbitrage structure. Thus, the resultant withdrawal fee may not fully block arbitrage by itself. Nevertheless, the total fees required for doing arbitrage (haircut fee plus arbitrage fee) are forbiddingly high to render arbitrage unprofitable in practice. A more rigorous and independent withdrawal fee design will be further developed.

From the above section, the profit from performing the strategy would be

$$\pi(y, A_i, L_i, D_i) = y\left(\frac{g(r'_i) - g(r_i)}{r'_i - r_i} - \frac{g(r^*_i) - g(r''_i)}{r^*_i - r''_i}\right)$$

$$= -L_i(g(r'_i) - g(r_i)) - (L_i - D_i)(g(r^*_i) - g(r''_i))$$

$$= D_i(g(r^*_i) - g(r''_i)) - L_i(g(r'_i) - g(r_i) + g(r^*_i) - g(r''_i))$$

We then attempt to find the maximum profit. Note that at the time the user withdraw deposit from the protocol, we are sitting at state r'_i, that is, the state after a swap with size y. From that, r'_i and r''_i are known. Note that $r^*_i = \frac{r_i L_i - D_i}{L_i - D_i}$, which is a function of r_i. We then consider the derivative of π with respect to r_i.

$$
\begin{aligned}
\frac{\partial \pi}{\partial r_i} &= D_i g'(r_i^*) \frac{L_i}{L_i - D_i} + L_i g'(r_i) - L_i g'(r_i^*) \frac{L_i}{L_i - D_i} \\
&= g'(r_i^*) \frac{L_i}{L_i - D_i} (D_i - L_i) + L_i g'(r_i) \\
&= L_i \left(g'(r_i) - g'(r_i^*) \right)
\end{aligned}
$$

Now, note that

$$
\begin{aligned}
& r_i < 1 \\
\Rightarrow & r_i^* < r_i \\
\Rightarrow & g'(r_i^*) < g'(r_i) \text{, by convexity of } g
\end{aligned}
$$

The other cases of r_i are similar. From that, we have

$$\frac{\partial \pi}{\partial r_i} = L_i\left(g'(r_i) - g'(r_i^*)\right) \begin{cases} > 0 & , \text{ if } r_i < 1 \\ < 0 & , \text{ if } r_i > 1 \\ = 0 & , \text{ if } r_i = 1 \end{cases}$$

Hence, the maximum of π happens at $r_i = 1$ and $r_i^* = 1$. Therefore, we have

$$\begin{aligned} \max_{r_i} \pi(r_i) &= \pi(1) \\ &= D_i(g(1) - g(r_i'')) - L_i(g(r_i') - g(1) + g(1) - g(r_i'')) \\ &= g(r_i'')(L_i - D_i) - g(r_i')L_i + g(1)D_i \end{aligned}$$

Therefore, for any withdrawal that happens at coverage ratio $r'_i < 1$ and after-withdrawal coverage ratio r_i'', the maximum profit and cost to be charged would be

$$C_w(r_i', L_i, D_i) := g(r_i'')(L_i - D_i) - g(r_i')L_i + g(1)D_i$$

This amount is to be charged if and only if $r_i' < 1$.

5 Deposit Arbitrage

This is similar to withdrawal arbitrage but in opposite direction.

5.1 Arbitrage Procedure

Consider the account for token i, assume $r_i > 1$.

1. User could swap y token j to $\bar{y}$ of token i. The new coverage ratio for token i would be $r'_i = \frac{A_i - \bar{y}}{L_i}$. User pays the slippage for lowering r_i.

2. User deposits D_i from token i account. The new coverage ratio decreases to $r''_i = \frac{A_i - \bar{y} + D_i}{L_i + D_i}$.

3. User reverses the swap in 1. to gain the slippage of pushing r_i to a higher level. User aims to restore the coverage ratio of token j account. Due to the altered state of coverage ratio, r''_i, the required amount for reversal swap, y', tends to be different from the initial amount $\bar{y}$. The final coverage ratio of token i account would then be $r^*_i = \frac{A_i + D_i - (\bar{y} - y')}{L_i + D_i}$.

The swap and deposit path is given by

$$S : (r_i, r_j) \xrightarrow{\text{swap}} (r'_i, r'_j) \xrightarrow{\text{deposit}} (r''_i, r'_j) \xrightarrow{\text{swap}} (r^*_i, r_j)$$

There is a key condition required for finding the amount of token i, y', needed for retrieving the target amount of token j, y, through the reversal swap:

$$y = (1 - S'_{i\to j})y'$$

Thus, the slippage from the action in terms of token i would be

$$= -yS_{j\to i} + f_{j\to i}(-y' f_{i\to j})S'_{i\to j}$$
$$= -y'(S_{j\to i} + S'_{i\to j} - S_{j\to i}S'_{i\to j})$$

That means the user could gain from this strategy if the overall slippage is greater than zero.

5.2 Deposit Fee

All the assumptions that are applied to the derivation of the withdrawal fee are also applicable in this section. The following deposit fee neglects the quadratic term in slippage gain as shown above, which gauges the impact of the first slippage exerted on the second slippage. The required condition for restoring the coverage ratio of token j account is neglected here. Also, y is assumed to be equal to y'. Analogous to the withdrawal fee, the resultant deposit fee may not fully block arbitrage by itself. Nevertheless, the total fees required for doing arbitrage (haircut fee plus arbitrage fee) are forbiddingly high to render arbitrage unprofitable in practice. A more rigorous and independent fee design will be further developed.

The profit from the strategy is

$$\begin{aligned}\pi(y, A_i, L_i, D_i) &= y\left(\frac{g(r'_i) - g(r_i)}{r'_i - r_i} - \frac{g(r^*_i) - g(r''_i)}{r^*_i - r''_i}\right)\\ &= -L_i(g(r'_i) - g(r_i)) - (L_i + D_i)(g(r^*_i) - g(r''_i))\\ &= -D_i(g(r^*_i) - g(r''_i)) - L_i(g(r'_i) - g(r_i) + g(r^*_i) - g(r''_i))\end{aligned}$$

Similar to section 4.2, we consider the derivative of π with respect to r_i,

$$\begin{aligned}\frac{\partial \pi}{\partial r_i} &= -D_i g'(r^*_i)\frac{L_i}{L_i + D_i} + L_i g'(r_i) - L_i g'(r^*_i)\frac{L_i}{L_i + D_i}\\ &= -g'(r^*_i)\frac{L_i}{L_i + D_i}(D_i + L_i) + L_i g'(r_i)\\ &= L_i\,(g'(r_i) - g'(r^*_i))\end{aligned}$$

Now, note that

$$
\begin{aligned}
& r_i > 1 \\
\Rightarrow & r_i^* < r_i \\
\Rightarrow & g'(r_i^*) < g'(r_i) \text{ , by convexity of } g
\end{aligned}
$$

The other cases of r_i are similar. From that, we have

$$
\frac{\partial \pi}{\partial r_i} = L_i(g'(r_i) - g'(r_i^*)) \begin{cases} < 0 & \text{, if } r_i < 1 \\ > 0 & \text{, if } r_i > 1 \\ = 0 & \text{, if } r_i = 1 \end{cases}
$$

Hence, the maximum of π happens at $r_i \to \infty$ and $r_i^* \to \infty$, given $r_i > 1$. Note that $\lim_{r_i \to \infty} g(r_i) = 0$, we have

$$
\begin{aligned}
\max_{r_i} \pi &= \lim_{(r_i, r_i^*) \to (\infty, \infty)} -D_i(g(r_i^*) - g(r_i')) - L_i(g(r_i') - g(r_i) + g(r_i^*) - g(r_i')) \\
&= (L_i + D_i)g(r_i'') - L_i g(r_i')
\end{aligned}
$$

Therefore, for any deposit that happens at coverage ratio $r_i' > 1$ and after-deposit coverage ratio r_i'', the maximum profit and cost to be charged would be

$$
C_d(r_i', L_i, D_i) := (L_i + D_i)g(r_i'') - L_i g(r_i')
$$

The deposit fee is to be charged if and only if $r_i' > 1$.

6 To-Amount Calculation

One challenge faced by our protocol would be that our protocol is unable to calculate a from-amount by specifying a to-amount. More precisely, given a swap $i \to j$, if we specify the from-amount to be Δ_i, we are able to get the swap path $(r_i, r_j) \to (r'_i, r'_j)$ and hence $S_{i\to j}$. From that, we are able to calculate the terminal exchange rate $f^*_{i\to j}$. However, given the to-amount, we are unable to calculate the terminal exchange rate $f^*_{i\to j}$.

6.1 Newton's Method Solution

Given a Δ^*_j terminal to-amount, we would like to find the from-amount Δ_i. First, we have the relation

$$\begin{aligned}\Delta^*_j &= \Delta_j(1 - S_{i\to j})(1-h)\\ &= -f_{i\to j}\Delta_i\left(1 - L_i\frac{g(r'_i) - g(r_i)}{\Delta_i} + L_j\frac{g(r'_j) - g(r_j)}{\Delta_j}\right)(1-h)\end{aligned}$$

Rearranging the equation, we have

$$f_{i\to j}\Delta_i - f_{i\to j}L_i g(r'_i) - L_j g(r'_j) = -\frac{\Delta^*_j}{1-h} - f_{i\to j}L_i g(r_i) - L_j g(r_j)$$

$$\Rightarrow \Delta_i - L_i g(r'_i) - \frac{L_j g(r'_j)}{f_{i\to j}} = \frac{\Delta^*_j}{f_{i\to j}(1-h)} - L_i g(r_i) - \frac{L_j g(r_j)}{f_{i\to j}}$$

Note that the right hand side of the equation is a constant, we let it to be $K := \frac{\Delta^*_j}{f_{i\to j}(1-h)} - L_i g(r_i) - \frac{L_j g(r_j)}{f_{i\to j}}$.

$$h(\Delta_i) := \Delta_i - L_i g(r'_i) - \frac{L_j g(r'_j)}{f_{i\to j}} - K$$

We want to solve $h(\Delta_i) = 0$. Consider the derivative of h,

$$h'(\Delta_i) = 1 - L_i g'(r'_i) \frac{1}{L_i} - \frac{L_j g'(r'_j)}{f_{i\to j}} \frac{-1}{L_j f_{j\to i}} = 1 - g'(r'_i) + g'(r'_j)$$

We then iterate through:

$$\Delta_{i_{n+1}} = \Delta_{i_n} - \frac{h(\Delta_{i_n})}{h'(\Delta_{i_n})}$$

When the number converges, we could obtain the to-amount numerically.

7 Risk Management

The following section highlight some risk management strategies we used in PrimeFi.

7.1 Price Oracle

While Price Oracle was mentioned in section 2.1, in practice for stableswap we will always assume the assets inside the pool are always pegged. Instead, upon swap, price oracle will be used to check the price between the two tokens, and the trade will be reverted if the price deviation between them are larger than x% (say, 1%).

7.2 Withdrawal

It is possible that one asset may become highly under-covered while a large liquidity provider wants to withdraw, hence he may not get the amount in full. PrimeFi will handle the situation by two ways:

> We may warn the liquidity providers that he can at most withdraw what the system has in cash, and he may wait until the system's coverage ratio restore back to equilibrium to withdraw;

2. We allow liquidity providers to withdraw in other assets in the pool that are over-covered

Reader should note that as long as the system is solvent, the total amount of assets in the system must be larger than liabilities, hence liquidity providers can always withdraw their liabilities in full, albeit may not be in the same kind of asset as he / she has provided.

8 References

1. Angeris, G., & Chitra, T. (2020). Improved Price Oracles: Constant Function Market Makers. SSRN Electronic Journal.

2. Hayden Adams. (2019). Uniswap birthday blog - v0.

3. Michael Egorov. (2019). stableswap - efficient mechanism for Stablecoin liquidity.

4. mStable. (2021). mStable gitbook.

5. Othman, A., Sandholm, T., Pennock, D. M., amp; Reeves, D. M. (2013). A practical liquidity-sensitive automated market maker. ACM Transactions on Economics and Computation (TEAC), 1(3):1-25.