Silkswap: An asymmetric automated market maker model for stablecoins

Nicola Cantarutti* , Alex Harker† , Carter Woetzel‡
shadeprotocol.io

July 31, 2022

Abstract

Silkswap is an automated market maker model designed for efficient stablecoin trading with minimal price impact. The original purpose of Silkswap is to facilitate the trading of fiat pegged stablecoins with Silk, an overcollateralized stablecoin pegged to a basket of global currencies and commodities; but it can be applied to any pair of stablecoins. The Silkswap invariant is a hybrid function that generates an asymmetric price impact curve. We present the derivation of the Silkswap model and its mathematical properties. We also compare different numerical methods used to solve the invariant equation. Finally, we compare our model with the well known Curve Finance model.

1 Introduction

Decentralized EXchanges (DEXs) are currently the most popular application of Decentralized Finance (DeFi). Unlike centralized exchanges, DEXs are not based on a single centralized entity acting as custodians or intermediaries. Instead, on DEXs traders retain full control of their funds and private keys, and smart contracts execute trades for users in a neutral and automated fashion. Most DEXs use Automated Market Maker (AMM) models to define the rules of trading, rather than relying on the order book model. Liquidity providers can deposit their tokens into liquidity pools in exchange for rewards coming from swap fees and token farming. The AMM algorithmically computes the price of the tokens inside a liquidity pool only based on the token balance. A technical introduction on the mechanics of AMMs can be found in [Angeris and Chitra, 2020] and [Mohan, 2022].

One of the first AMMs to appear in DeFi is Uniswap [Adams, 2018], that together with its upgraded version, Uniswap v2 [Adams et al., 2020], is based on the Constant Product Market Maker (CPMM) model. This model assumes that the product of the quantities of tokens in a liquidity pool is constant, which guarantees infinite liquidity inside the pool. CPMM works quite well for volatile

*Contact: https://twitter.com/Canta86
†Contact: https://twitter.com/AlexHarker19
‡Contact: https://twitter.com/l_woetzel

tokens, as it promptly adapts the price in response to new trades. Nevertheless, it is not very suitable when trading low volatility or stable assets.

Another important AMM model is the Constant Sum Market Maker (CSMM), that assumes that the sum of the quantities of tokens inside the pool is constant. The advantage of this model is that any trade has zero price impact, however it also has the inconvenience that it permits the complete drain of the entire liquidity inside the pool. For this reason, it is not used in practical applications.

Low price impact is a good property when swapping between fiat pegged stablecoins. Curve Finance [Egorov, 2019], formerly Stableswap, and the new version Curve v2 [Egorov, 2021] implement a Hybrid Function Market Maker (HFMM) model with the exact purpose of facilitating swaps between stablecoins. This model combines CPMM and CSMM together in order to take advantage of both the infinite liquidity property of CPMM and the zero price impact property of CSMM.

The main goal of the Silkswap model is to facilitate the trades of Silk with other fiat pegged stablecoins. However this model can be used to trade any pair of stablecoins. Silk [Duniya, 2021] is a privacy-preserving overcollateralized stablecoin developed by Shade Protocol and native to Secret Network. The main feature of Silk is that it is a stablecoin pegged to a basket of global currencies and commodities using decentralized price feeds, creating a digital currency that serves as a hedge against global volatility. We developed Silkswap as an HFMM model inspired by the Curve Finance v2 model. Additionally we introduced more flexibility in the shape of the invariant function, allowing for an asymmetric price impact curve. In this way it is possible to discourage possible imbalances within the liquidity pool.

In the next sections we derive the Silkswap invariant and show its mathematical properties. We present numerical results for three different zero-finder algorithms and finally we compare our model with the Curve finance model.

2 Silkswap model

Let us consider a liquidity pool containing only two stablecoins. The two tokens in focus are token X, that represents any fiat pegged stablecoin, and token Y that represents Silk. We use lower-case letters x and y to indicate the quantities of X and Y. Without loss of generality, let us quote the price of X and Y in US dollars, although any other fiat currency would work. Let us introduce the conversion factors p_X and p_Y, such that the values in dollars of x and y are simply $p_X \times x$ and $p_Y \times y$ respectively. The conversion factor p_X has units [USD]/[X] and p_Y has units [USD]/[Y]. We also introduce $p := \frac{p_Y}{p_X}$, the conversion factor between Y and X, which has units [X]/[Y].

The conversion factors p_X and p_Y represent the market prices of one unit of X and Y respectively, while p is the price of one unit of Y in terms of X. These values must be provided by an oracle, which is an external source of information that is fed into the AMM smart contract with a certain frequency.

Example:

Let us consider a liquidity pool containing USDC (token X) and Silk (token Y). If the oracle price of Silk is 1.05$ then $p_Y = 1.05\,\text{USD/SILK}$, while for USDC the oracle price is exactly 1$, resulting in a perfect peg, then we have $p_X = 1\,\text{USD/USDC}$. The direct conversion between Silk and USDC is therefore

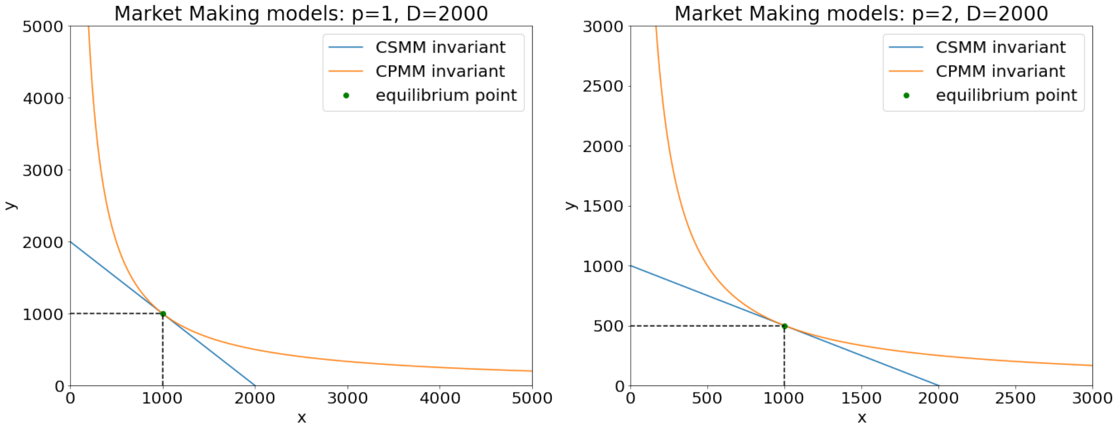

Figure 1: Graph of the CPMM and CSMM models with $D = 2000$. On the left $p = 1$. On the right $p = 2$.

$p = \frac{p_Y}{p_X} = 1.05\,\text{USDC/SILK}$. Later we will often identify X with USDC and Y with SILK to make the discussion clearer.

Let us define the **equilibrium point** in a liquidity pool as the point (x, y) where the dollar value of x equals the dollar value of y. The equilibrium point satisfies the **equilibrium equation**

$$p_Y\, y \;=\; p_X\, x. \tag{1}$$

From now on, we will assume that X is our numeraire, and we will express the value of Y in terms of X. The equilibrium equation becomes

$$p\, y = x. \tag{2}$$

This choice is due to the fact that it is more natural for the user to evaluate an asset in terms of a fiat pegged stablecoin. Because Silk is pegged to a basket, its value expressed in terms of any fiat currency is not stable, but it fluctuates with a very low volatility.

2.1 Silkswap invariant

Under the CPMM model, the quantities x and y must satisfy the following equation:

$$x\, p\, y = K \tag{3}$$

with $K > 0$. The equilibrium point, satisfying both (2) and (3), is $(x, y) = (\sqrt{K}, \frac{\sqrt{K}}{p})$. Under the CSMM model instead, the following equation holds:

$$x \,+\, py = D \tag{4}$$

with $D > 0$. The equilibrium point, satisfying (2) and (4), is $(x, y) = (\frac{D}{2}, \frac{D}{2p})$. The equilibrium point is a fundamental point where the ratio of the quantities

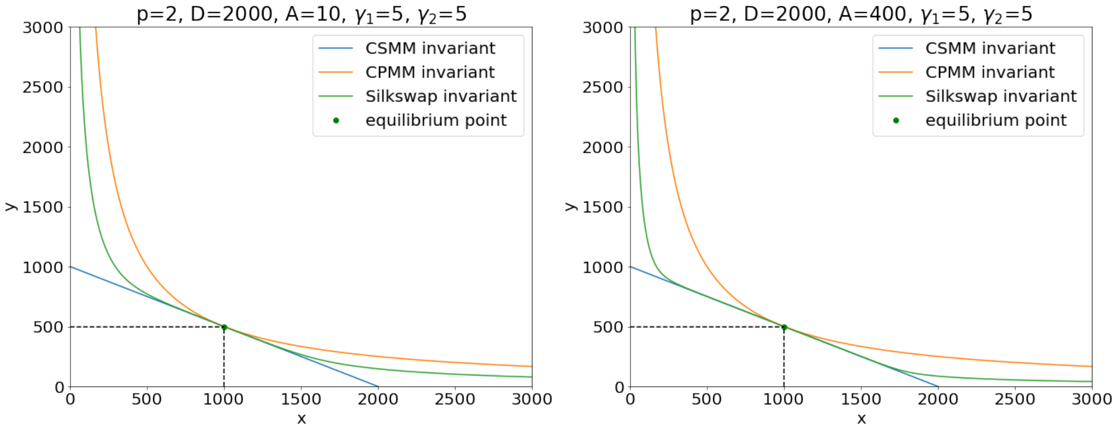

Figure 2: Graph of the Silkswap invariant. On the $A = 10$. On the right $A = 400$. We can see that the parameter A indicates the closeness of the Silkswap curve to the CSMM line.

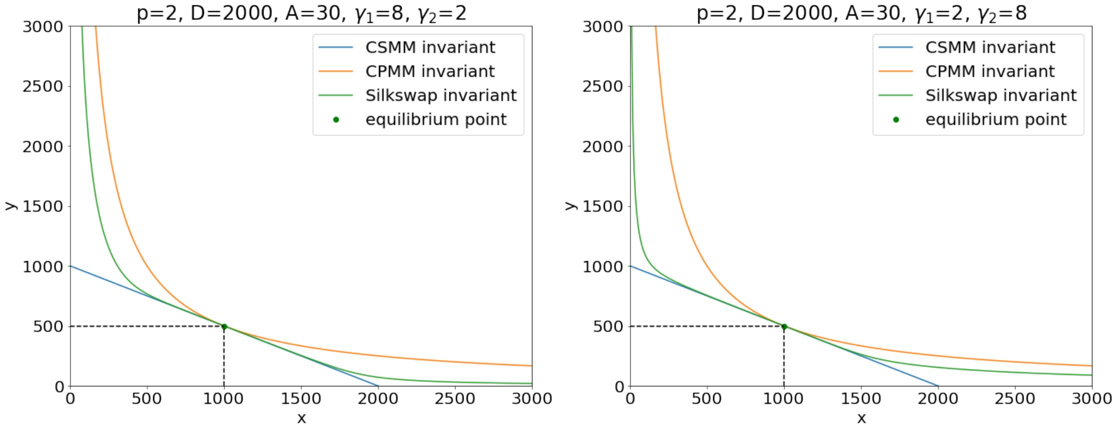

Figure 3: Graph of the Silkswap invariant. We inverted the values of γ_1 and γ_2 to show how these parameters can control the asymmetry of the curve.

of tokens in the pool is equal to p, and any invariant curve should contain this point. Since this point is unique, it follows that $K = \frac{D^2}{4}$. In figure 1 we can see the graph of these two models, with different values of p.

We can now multiply Eq. (4) by χAD, and sum it with eq. (3):

$$(\chi AD)\,(x + py) + xpy \;=\; (\chi AD)D + \frac{D^2}{4} \tag{5}$$

where χ is a function of x and y, defined as

$$\chi := \left(\frac{4xpy}{D^2}\right)^{\gamma} \tag{6}$$

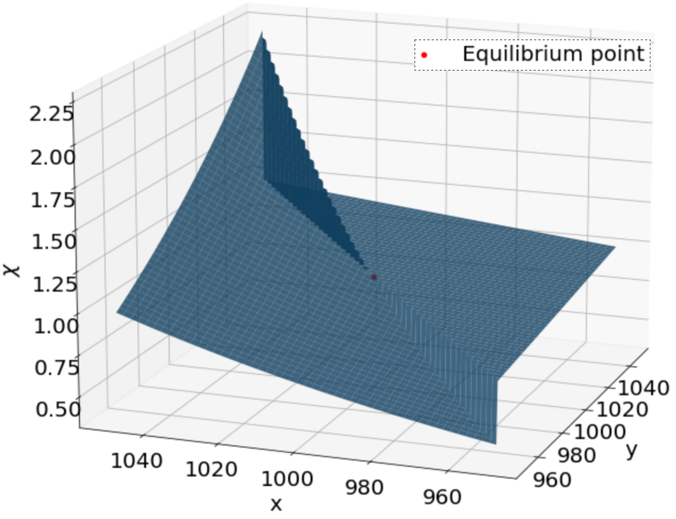

Figure 4: Function χ calculated with $D = 2000$, $p = 1$, $\gamma_1 = 2$, $\gamma_2 = 8$. The equilibrium point is at $(x, y) = (1000, 1000)$ and $\chi(x, y) = 1$. The function χ is continuous at this point, but not differentiable.

and

$$\gamma := \begin{cases} \gamma_1, & \text{if } x \leq py \\ \gamma_2, & \text{if } x > py \end{cases} \tag{7}$$

and the other parameters are constants satisfying $A > 0$, $\gamma_1 \geq 0$, $\gamma_2 \geq 0$. Let us remark that χ is an adimensional quantity and D has the same dimension of x. We call the equation (5), together with (6) and (7), the **Silkswap invariant**. The graph of the Silkswap invariant is shown in figures 2 and 3 under different set of parameters. We can see that the CPMM hyperbola is always greater than the CSMM line, and they touch each other at the equilibrium point. The Silkswap invariant graph lies in between them. In the Appendix we prove these facts in the theorems (A.1) and (A.3). The parameter A tells us how close the Silkswap graph is to the CPMM or to the CSMM. From Eq. (5) we can easily see that for $A \to 0$ the invariant converges to the CPMM curve, while for $A \to \infty$ it converges to the CSMM curve.[1]

The two dimensional function χ in (6) is a discontinuous function. We show the surface plot in Fig. (4). In the Appendix we prove that, although the Silkswap invariant contains a discontinuous function, the points of our interest are regular points where the invariant is continuously differentiable.

2.2 Pricing with the Silkswap invariant

In order to compute the price of a token, it is convenient to express the invariant in the explicit form $y = f(x)$, where $f : \mathbb{R}_{>0} \to \mathbb{R}_{>0}$ is continuously differentiable, decreasing and convex [2]. For the CPMM the explicit form is the

[1] The parameter A has the same meaning of the parameter A in the Stableswap model [Egorov, 2019].

[2] The set $\mathbb{R}_{>0}$ is the set $\{x \in \mathbb{R} | x > 0\}$.

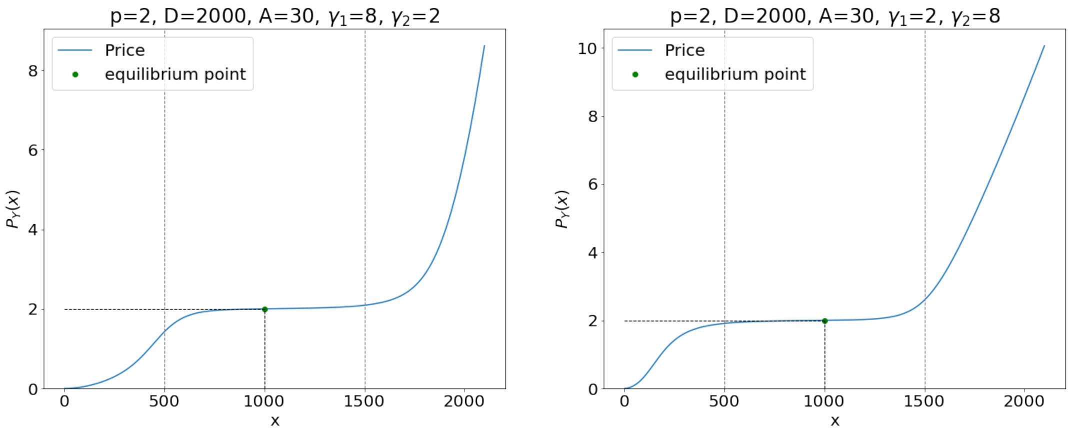

Figure 5: Graph of the price of the token Y as function of x, obtained from the Silkswap invariant. Same parameters used in Fig (3). The vertical lines at 500 and 1500 only serve to better understand the asymmetry in the graph.

hyperbola $f(x) = \frac{K}{px}$, while for the CSMM the explicit form is the straight line $f(x) = \frac{1}{p}(-x + D)$, defined for $0 \leq x \leq D$.

If we trade an infinitesimal quantity dy, using the first order Taylor approximation $dy \approx \frac{df(x)}{dx}dx$, it results equal to trade the quantity $\frac{df(x)}{dx}dx$. Let us recall that by definition $\frac{df(x)}{dx} < 0$, so we need to use the absolute value to define a positive price. We define the **current price** of the token X as function of x

$$P_X(x) := \left|\frac{df(x)}{dx}\right| = \left|\frac{dy}{dx}\right|. \tag{8}$$

The price of the token Y is defined as:

$$P_Y(x) := \left|\frac{dx}{dy}\right| = \frac{1}{P_X}. \tag{9}$$

Under the CPMM model, the current price is $P_Y(x) = \frac{px^2}{K}$, and at equilibrium $P_Y(\sqrt{K}) = p$. Under the CSMM model, the current price is a constant value, $P_Y(x) = p$, for any $0 < x < D$.

Unfortunately the Silkswap invariant cannot be written in an explicit form. We can define the function $F : \mathbb{R}^2_{>0} \to \mathbb{R}_{>0}$ as

$$F(x, y) := AD\left(\frac{4xpy}{D^2}\right)^{\gamma}(x + py - D) + xpy - \frac{D^2}{4} \tag{10}$$

with γ as in (7). The Silkswap invariant (5) is the set of points that satisfy

$$F(x, y) = 0. \tag{11}$$

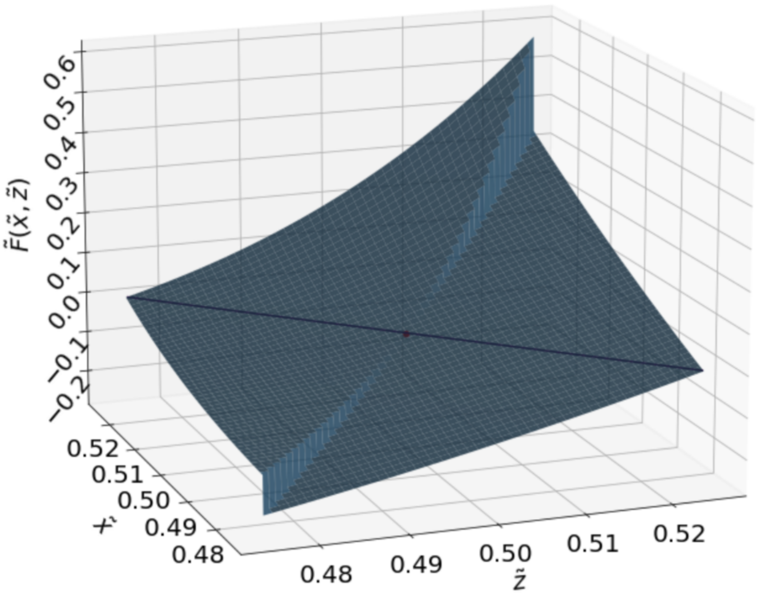

Figure 6: Scaled function $\tilde{F}(\tilde{x}, \tilde{z})$, with, $\gamma_1 = 2$, $\gamma_2 = 8$. The equilibrium point is at $(\tilde{x}, \tilde{z}) = (0.5, 0.5)$ and the function is continuously differentiable in this point. The solid line represents the zero level of the invariant.

The partial derivatives are:

$$\frac{\partial F}{\partial x} = AD\left(\frac{4xpy}{D^2}\right)^{\gamma}\left[1 + \gamma\,\frac{x + py - D}{x}\right] + py \tag{12}$$

$$\frac{\partial F}{\partial y} = AD\left(\frac{4xpy}{D^2}\right)^{\gamma}\left[p + \gamma\,\frac{x + py - D}{y}\right] + px. \tag{13}$$

Thanks to the Theorem (A.6), we know the expression of the slope of the Silkswap invariant, which can be used to compute the prices of the tokens by (8) and (9). The slope has the following expression:

$$\frac{dy}{dx} = -\frac{\frac{\partial F}{\partial x}}{\frac{\partial F}{\partial y}}\,. \tag{14}$$

In Figure 5 we present two examples of price curves obtained from this expression.

2.3 Scaled invariant

If we consider the function (10) depending also on D, for any constant $c > 0$ we have

$$F(x, y, D) = 0 \quad \Longrightarrow \quad F(cx, cy, cD) = 0. \tag{15}$$

This means that the Silkswap invariant is also **invariant by scaling**.
This property is very useful in practice because it reduces the convergence time of some numerical methods and prevents possible overflows when the variables

have very high magnitudes.
Let us introduce the variable $z := py$. We can consider $F(cx, cy, cD)$ with $c = \frac{1}{D}$, equal to $F(\frac{x}{D}, \frac{y}{D}, 1)$, and define the corresponding scaled function:

$$\tilde{F}(\tilde{x}, \tilde{z}) \;:=\; A\left(4\tilde{x}\tilde{z}\right)^{\gamma}(\tilde{x} + \tilde{z} - 1) + \tilde{x}\tilde{z} - \frac{1}{4}. \tag{16}$$

where $\tilde{x} = \frac{x}{D}$, and $\tilde{z} = \frac{z}{D}$. We show in Fig. 6 the graph of this function. The **scaled Silkswap invariant** is given by

$$\tilde{F}(\tilde{x}, \tilde{z}) \;=\; 0. \tag{17}$$

The function (16) is related with (10) by

$$\tilde{F}(\tilde{x}, \tilde{z}) = \frac{1}{D^2} F(x, y). \tag{18}$$

Using the chain rule

$$\frac{\partial \tilde{F}(\tilde{x}, \tilde{z})}{\partial \tilde{x}} \;=\; \frac{\partial \tilde{F}(\tilde{x}, \tilde{z})}{\partial x}\frac{dx}{d\tilde{x}} \;=\; \frac{1}{D}\frac{\partial F(x, y)}{\partial x} \tag{19}$$

$$\frac{\partial \tilde{F}(\tilde{x}, \tilde{z})}{\partial \tilde{z}} \;=\; \frac{\partial \tilde{F}(\tilde{x}, \tilde{z})}{\partial y}\frac{dy}{d\tilde{z}} \;=\; \frac{1}{pD}\frac{\partial F(x, y)}{\partial y}, \tag{20}$$

we can rewrite the price equation (14) as

$$\frac{dy}{dx} = -\frac{1}{p}\,\frac{\frac{\partial \tilde{F}(\tilde{x}, \tilde{z})}{\partial \tilde{x}}}{\frac{\partial \tilde{F}(\tilde{x}, \tilde{z})}{\partial \tilde{z}}}\,. \tag{21}$$

In the numerical calculation of the swap amount, we will make use of the scaled function (16), rather than (10), because it reduces a lot the number of operations, and consequently the run time and gas fees.

3 Numerical implementation

Since this model is defined by an implicit equation, we need numerical methods to compute the variables of interest. If the amounts of tokens in the pool satisfy the equilibrium equation 2, then the parameter D can be quickly computed from Eq. (4). However, it is quite rare that a liquidity pool is in perfect equilibrium and the equation 2 is almost never satisfied. In this cases we need to compute D from Eq. (5) using a numerical method. Let us define $F(D)$ as the function (10) when we consider D variable and x, y fixed. With an abuse of notation we can call it "invariant function dependent of D", but let us recall that the name invariant can be used only when $F(D) = 0$. In Fig 7, we can see that $F(D)$ is a smooth decreasing function. We compare three different numerical methods: the Newton method, Halley method and the bisection method, see Table 1.

The idea of using two times arithmetic mean (AM) and geometric mean (GM) as starting points for Newton and Halley methods comes from Theorem A.4. We can see that 2AM is a better choice. Although Newton method has more loop iterations than Halley, it performs better in terms of time. The reason is that the calculation of the second derivative is more expensive, in terms

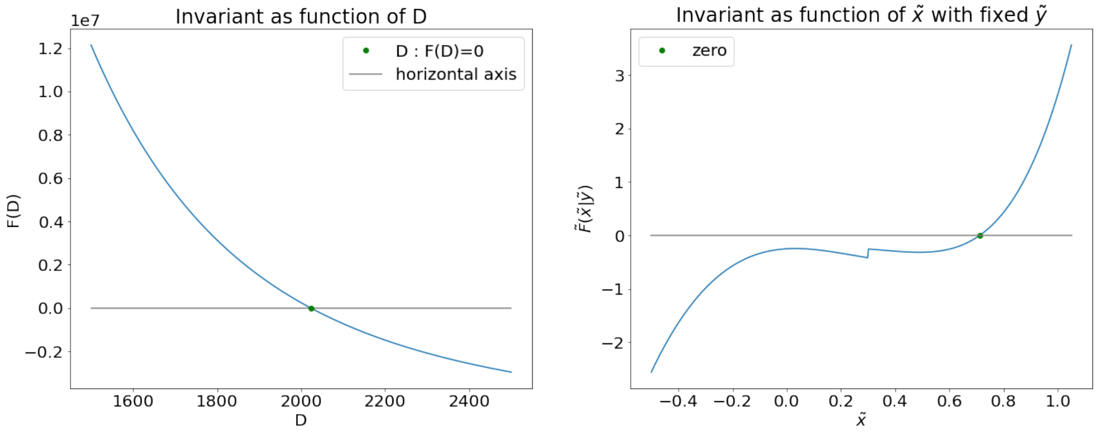

Figure 7: LEFT: Invariant as function of D. We used $x = 1900$, $y = 100$, $p = 2$, $A = 30$, $\gamma_1 = 8$, $\gamma_2 = 2$.
RIGHT: Scaled invariant as function of $\tilde{x}$. We used $y = 600$, $D = 2000$, $p = 1$, $A = 5$, $\gamma_1 = 2$, $\gamma_2 = 3$. The function has only one zero.

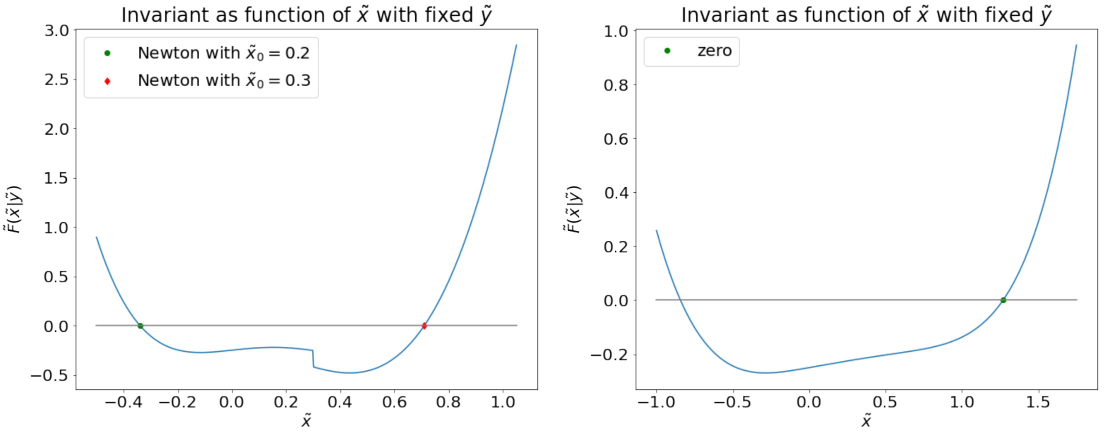

Figure 8: LEFT: Scaled invariant as function of $\tilde{x}$. We used $y = 600$, $D = 2000$, $p = 1$, $A = 5$, $\gamma_1 = 3$, $\gamma_2 = 2$. The graph shows two zeros. The Halley and Newton methods are highly dependent on the initial guess $\tilde{x}_0$.
RIGHT: Scaled invariant as function of $\tilde{x}$. We used $y = 200$, $D = 2000$, $p = 1$, $A = 5$, $\gamma_1 = 3$, $\gamma_2 = 4$. The function is increasing for $\tilde{x} > 0$, therefore the Halley and Newton methods converge to the right solution.

of operations, than the extra iterations. The expression of the derivatives are in Appendix B. Bisection method, as expected from theoretical results, is the slowest.

We will also make use of numerical methods to calculate the amount of tokens that are returned during a swap. When a quantity Δy is introduced into the pool, we need to compute the quantity Δx that is extracted from the pool, and such that the point $(x - \Delta x, y + \Delta y)$ belongs to the Silkswap invariant.

Pool with 2000 USDC and 1000 SILK			
Starting point	Method	Iterations	Time
2AM	Newton	4	312.8µs
2GM	Newton	8	770.4µs
2AM	Halley	2	390.5µs
2GM	Halley	5	820.4µs
/	Bisection	61	2.37ms

Pool with 200000 USDC and 100000 SILK			
Starting point	Newton	Halley	Bisection
2AM	Newton	4	290.0µs
2GM	Newton	7	790.3µs
2AM	Halley	3	399.5µs
2GM	Halley	5	914.2µs
/	Bisection	67	2.6ms

Table 1: Performance tables for the calculation of the parameter D. Newton and Halley methods are computed using two different initial guesses: $2\,AM$ and $2\,GM$. The bisection method is computed inside the interval [2GM, 2AM], see Theorem A.4. The parameters of the AMM are $p = 1$, $A = 100$, $\gamma_1 = \gamma_2 = 8$. We set a very small tolerance $\epsilon_D = 10^{-16}$.

The parameter D is fixed, and it is irrelevant for this calculation, therefore in order to simplify the problem, we can use the scaled function (16). Let us define $\tilde{F}(\tilde{x}|\tilde{y})$ and $\tilde{F}(\tilde{y}|\tilde{x})$ the function with $\tilde{y}$ or $\tilde{x}$ fixed respectively. Again, with an abuse of notation we can call these functions "invariant functions depending on $\tilde{x}$ or $\tilde{y}$". In the following we consider the case when Δx is extracted and therefore we need to find the zero of $\tilde{F}(\tilde{x}|\tilde{y})$, but the same analysis works for $\tilde{F}(\tilde{y}|\tilde{x})$. In the Figures 7 and 8, we can see that the function is discontinuous and can have different shapes depending on the choice of the parameters. So the numerical convergence to the right solution is not always guaranteed.

By definition, the parameter γ can be any non-negative real number. In practice we restrict γ to be a positive integer value in order to simplify the model. The value of γ_1 is particularly important because it controls the behavior of the left tail:

$$\lim_{\tilde{x}\to-\infty} \tilde{F}(\tilde{x}|\tilde{y}) \to -\infty \qquad \text{for even } \gamma_1$$
$$\lim_{\tilde{x}\to-\infty} \tilde{F}(\tilde{x}|\tilde{y}) \to +\infty \qquad \text{for odd } \gamma_1,$$

as we can see if we compare the plot of $\tilde{F}(\tilde{x}|\tilde{y})$ in Fig. 7 with the plots in Fig. 8. When γ_1 is odd, $\tilde{F}(\tilde{x}|\tilde{y})$ has two zeros. Since $\tilde{x} > 0$ by definition, we know that the correct solution must be positive. However, numerical methods such as Newton and Halley can converge to the wrong solution if initialized with an unlucky starting point, see Fig. 8 (LEFT). Under some set of parameters, it is possible that the function $\tilde{F}(\tilde{x}|\tilde{y})$ is increasing for $\tilde{x} > 0$, and the Newton method works fine, see Fig. 8 (RIGHT). However, to avoid this uncertainty, in these cases it is better to use numerical methods that always guarantee the convergence to the right solution, such as the bisection algorithm. If we call $(\tilde{x}_0, \tilde{y}_0)$ the amounts in the pool before the swap, and $(\tilde{x}_1, \tilde{y}_1)$ the amounts after the swap such that $\tilde{F}(\tilde{x}_1|\tilde{y}_1) = 0$, then we know that $\tilde{x}_1 \in [0, \tilde{x}_0]$.

In practical applications we want to take advantage of the speed of the Newton method, and therefore we choose to use only odd values for γ_1. For comparisons between the three different numerical methods under consideration, see Table 2. Let us comment a few points:

1. The number of iterations of the bisection method does not depend on the size of the pool. The reason is that we are searching in the interval $[0, \tilde{x}_0]$,

Pool with 10^3 USDC and 10^3 SILK			
Swap size	Method	Iterations	Time
0	Bisect	0	153µs
10^{-1}	Bisect	52	1.32ms
1	Bisect	52	1.31ms
10	Bisect	52	1.30ms
10^2	Bisect	52	1.37ms
10^3	Bisect	52	1.31ms
10^4	Bisect	52	1.34ms
10^5	Bisect	52	1.47ms
10^6	Bisect	52	1.46ms
0	Halley	0	190µs
10^{-1}	Halley	2	311µs
1	Halley	2	303µs
10	Halley	3	366µs
10^2	Halley	4	436µs
10^3	Halley	9	751µs
10^4	Halley	17	1.30ms
10^5	Halley	26	1.95ms
10^6	Halley	37	2.88ms
0	Newton	0	159µs
10^{-1}	Newton	3	262µs
1	Newton	3	246µs
10	Newton	4	285µs
10^2	Newton	6	358µs
10^3	Newton	16	674µs
10^4	Newton	30	1.20ms
10^5	Newton	49	2.01ms
10^6	Newton	68	2.80ms

Pool with 10^6 USDC and 10^6 SILK			
Swap size	Method	Iterations	Time
0	Bisect	0	141µs
10^{-1}	Bisect	52	1.51ms
1	Bisect	52	1.45ms
10	Bisect	52	1.62ms
10^2	Bisect	52	1.46ms
10^3	Bisect	52	1.48ms
10^4	Bisect	52	1.46ms
10^5	Bisect	52	1.57ms
10^6	Bisect	52	1.47ms
0	Halley	0	150µs
10^{-1}	Halley	1	270µs
1	Halley	1	269µs
10	Halley	2	346µs
10^2	Halley	2	341µs
10^3	Halley	2	341µs
10^4	Halley	3	422µs
10^5	Halley	4	488µs
10^6	Halley	9	885µs
0	Newton	0	155µs
10^{-1}	Newton	2	236µs
1	Newton	2	239µs
10	Newton	2	247µs
10^2	Newton	3	297µs
10^3	Newton	3	284µs
10^4	Newton	4	316µs
10^5	Newton	6	394µs
10^6	Newton	16	1.04ms

Table 2: Performance tables. We compare swap times where a trader swaps SILK for USDC. We consider different sizes and different numerical methods. Initial guess for Halley and Newton methods is 2AM $:= \tilde{x} + \tilde{z}$ scaled by D. The parameters of the AMM are $p = 1$, $A = 100$, $\gamma_1 = \gamma_2 = 8$. We set a very small tolerance $\epsilon_x = 10^{-16}$.

and $\tilde{x}_0 = \frac{x}{D}$. In the example, since we are at equilibrium $D = 2x$ and therefore $\tilde{x}_0 = \frac{1}{2}$.

2. The number of iterations of the bisection method does not depend on the size of the trade either.

3. As for the calculation of D, Newton is slightly faster than Halley. The computation of second order derivatives (see Appendix B) is expensive.

4. The number of iterations of the Newton method increases when the size of the trade is big with respect to the size of the pool. In this case the Newton method is the one that performs worse while the bisection method is the one that performs best. However, in practice it is very unlikely to see transactions bigger than the size of the pool.

5. We used $\tilde{x}_0$ i.e. the value $\frac{x}{D}$ before the swap, as an initial guess. We tried also with $\frac{2\text{AM}}{D}$ and $\frac{2\text{GM}}{D}$ as initial guesses, but the performances are worse.

When considering an even γ_1, it turns out that the Newton method is superior for both the calculations of D and the swap size. In production, we decided to use Newton as main solver, and bisection as fallback in case of failure.

We tested the model with values of A ranging from 1 to 10^5 and values of gamma from 1 to 75, under a variety of conditions including pools sizes from 1 to 10^{11} USDC total value with a similar range of trade sizes. Generally, the Rust implementation can handle trade sizes several orders of magnitude above the size of the pool without overflowing, unless the values for γ_1 or γ_2 are excessively high. Since γ_1 and γ_2 are exponents, large values quickly create unmanageable numbers.

We implemented the Silkswap algorithm inside Rust smart contracts, which do not allow floating point calculations, and this required the use of a few workarounds. Since smart contracts only support integer arithmetics, we stored most numbers as 10^{18} larger than their actual value. This effectively allowed us to store 18 decimal places. Variables are stored as **uint256** i.e. unsigned integers with 256 bits. Additionally, we paired each variable with a boolean representing its sign, allowing us to calculate both positive and negative numbers. The square root in 2GM is computed by the Babilonian method.

The Rust implementation of the Silkswap algorithm on Secret Network generates gas fees of around 30000 gas from the mathematical calculations alone. Since 10^7 gas $\sim$ 1 SCRT, the algorithm costs about 0.003 SCRT.

4 Comparison with the Curve model

Curve finance is currently the most popular DEX for trading stablecoins. The Curve v2 model [Egorov, 2021] is an HFMM model, conceptually not very different from Silkswap. They both take a kind of weighted average between the CSMM and the CPMM models. Since in the development of Silkswap we took inspiration from Curve, we decided to use a similar notation. The parameter A in Curve has the same meaning as in Silkswap. The main difference between these models is the definition of the function χ:

$$\text{Silkswap}: \quad \chi = \left(\frac{4xy}{D^2}\right)^{\gamma} \qquad \gamma = \begin{cases} \gamma_1, & \text{if } x \leq y \\ \gamma_2, & \text{if } x > y \end{cases}$$

$$\text{Curve v2}: \quad \chi = \frac{4xy}{D^2}\left(\frac{\gamma}{\gamma + 1 - \frac{4xy}{D^2}}\right)^2.$$

For simplicity we didn't include the conversion factor p. In both models $0 \leq \chi \leq 1$. If for a moment we do not consider the fact that our gamma can assume two values, the gammas in the two models have different meanings. For any point not at equilibrium, in Silkswap: $\lim_{\gamma\to 0}\chi = 1$ and $\lim_{\gamma\to\infty}\chi = 0$, while for Curve v2 we have the opposite: $\lim_{\gamma\to 0}\chi = 0$ and $\lim_{\gamma\to\infty}\chi = 1$.

In Fig. 9 we can see the invariant and price curves produced by these two models. The values of gammas are chosen to make the two curves as close as possible when using same value of A. We notice that the Silkswap model produces a higher curvature, although we did not prove it formally.

Another difference between the models is that currently Silkswap is designed to work with liquidity pools containing only two tokens, while Curve v2 can have pools with any number of coins.

As done by Curve, we too have decided to apply transaction fees not to the token that is inserted into the pool (as Uniswap does) but to the extracted token.

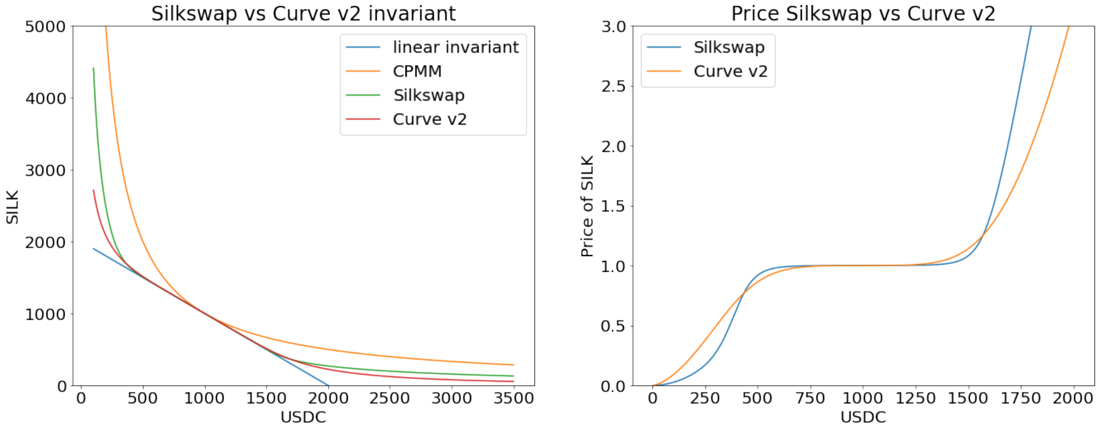

Figure 9: LEFT: Comparison of the Silkswap invariant with the Curve v2 invariant curves. RIGHT: Comparison of price curves of SILK as function of USDC in the pool. We used $D = 2000$, $p = 1$, and $A = 400$ for both models. In Silkswap we use $\gamma_1 = \gamma_2 = 10$. In Curve v2 we use $\gamma = 0.05$

5 Conclusions

The Silkswap invariant is an AMM model that allows users to trade stablecoins with minimal price impact when the state of the pool is close to equilibrium. Away from equilibrium, the price impact tends to that of the CPMM model. The main innovation of this model is the asymmetry of the invariant, which helps to regulate the liquidity in the pool and discourage strong imbalances in the quantity of tokens asymmetrically. For new stablecoins like Silk that are in the process of building up liquidity, the asymmetric curve provides attack protection, discouraging the sale when the amount of Silk in the pool exceeds a certain threshold (dependent on the parameters of the model). This is done by making the price impact grow faster in that direction and slower in the opposite. Figure 5 shows this behavior very well.

Acknowledgements

This research was fully funded by Shade Protocol.

A Properties of the Silkswap invariant

In Figure 1 we can see that the CPMM graph is greater than the CSMM graph, and they intersect each other at the equilibrium point. Let us formally prove this fact.

Theorem A.1. *For $x > 0$, the condition*

$$\frac{1}{p}(-x + D) \leq \frac{D^2}{4px} \tag{A.1}$$

is always satisfied.

Proof. The condition (A.1) can be written as

$$x^2 - Dx + \frac{D^2}{4} = \left(x - \frac{D}{2}\right)^2 \geq 0,$$

which is always verified. □

Theorem A.2. *The value of χ, defined in (6) always satisfy*

$$0 < \chi \leq 1. \tag{A.2}$$

Proof. Since χ is a function of only positive variables, it follows that χ must be positive.
Let us consider the Silkswap invariant (5):

$$\underbrace{(\chi AD)}_{>0} \underbrace{(x + py - D)}_{\geq 0} + \underbrace{xpy - \frac{D^2}{4}}_{\leq 0} = 0 \tag{A.3}$$

Since the sum of two terms is zero, it means that or both terms are zero, or the two terms have opposite sign. The case of both terms equal zero happens only at the equilibrium point.
Let us consider the case of both terms different than zero. We want to prove by contradiction that the first term must be positive.
If the first term is negative, then $(x + py - D) < 0$, and we have that

$$y < \frac{1}{p}(-x + D) \leq \frac{D^2}{4px},$$

by Theorem A.1. This implies that $xpy - \frac{D^2}{4} < 0$, but this is a contradiction because the terms must have opposite sign. The second term is always negative or zero, and we can conclude the proof.

$$xpy - \frac{D^2}{4} \leq 0 \implies \frac{4xpy}{D^2} \leq 1 \implies \chi \leq 1.$$

□

In Figure 2 we can see that the Silkswap invariant always lies between the graphs of the CSMM and CPMM models.

Theorem A.3. *The graph of the Silkswap invariant always lies between the graphs of the CSMM and CPMM models.*

Proof. First we prove that the Silkswap invariant graph is not greater than the CPMM graph. This is a direct consequence of Theorem A.2:

$$\frac{4xpy}{D^2} \leq 1 \implies y \leq \frac{D^2}{4xp}. \tag{A.4}$$

Now we prove that the Silkswap invariant graph is not smaller than the CSMM graph. Let us consider the invariant (5) and divide it by the positive quantity $AD\chi$. We get

$$\begin{aligned} 0 &= (x + py - D) + \frac{xpy - \frac{D^2}{4}}{AD\chi} \\ &\leq x + py - D. \end{aligned}$$

where we used $xpy - \frac{D^2}{4} \leq 0$. It follows that

$$y \;\geq\; \frac{-x+D}{p}. \tag{A.5}$$

□

Theorem A.4. *The parameter D in the Silkswap invariant satisfies*

$$2GM \;\leq\; D \;\leq\; 2AM, \tag{A.6}$$

where GM is the geometric mean of x and py, and AM is the arithmetic mean.

Proof. This is an immediate consequence of Theorem A.3. From the two inequalities (A.4) and (A.5) we obtain

$$2\sqrt{xpy} \;\leq\; D \;\leq\; x + py. \tag{A.7}$$

□

Theorem A.5. *The partial derivatives (12), (13) of the function $F(x, y)$ defined in (10) are always positive on $\big\{(x, y) \,:\, F(x, y) = 0\big\}$.*

Proof. We present a proof for $\frac{\partial F}{\partial y}$ only, since the same arguments can be used for $\frac{\partial F}{\partial x}$. Let us consider the expression:

$$\frac{\partial F}{\partial y} \;=\; AD\chi\left[p(\gamma+1) + \gamma\left(\frac{x}{y} - \frac{D}{y}\right)\right] + px.$$

Since all variables are positive, when $\left(\frac{x}{y} - \frac{D}{y}\right) \geq 0$ then $\frac{\partial F}{\partial y} > 0$. This happens for $x \geq D$.
For $0 < x < D$, let us rearrange the terms inside the square brackets and use (A.5):

$$\left[\gamma p + p - \gamma p \underbrace{\left(\frac{1}{y}\,\frac{-x+D}{p}\right)}_{\leq 1}\right] \;\geq p \;>\; 0.$$

This last inequality proves the theorem. □

Unfortunately, the function $F(x, y)$ defined in (10) is not continuous in the entire $\mathbb{R}^2_{>0}$, and we need to be careful around the points of discontinuity. The following theorem guarantees the validity of (14).

Theorem A.6. *The derivative of the Silkswap invariant can be written as*

$$\frac{dy}{dx} = -\frac{\frac{\partial F}{\partial x}}{\frac{\partial F}{\partial y}}.$$

Proof. The function $F(x, y)$ is continuously differentiable everywhere except on the line $x = py$, where it is discontinuous. The intersection between this line and the Silkswap invariant,

$$\begin{cases} x = py \\ F(x, y) = 0, \end{cases}$$

corresponds to the equilibrium point $(\frac{D}{2}, \frac{D}{2p})$. At the equilibrium point we have

$$\lim_{(x,y)\to\left(\frac{D}{2},\frac{D}{2p}\right)} F(x,y) \;=\; 0$$

from any directions, and therefore $F(x,y)$ is continuous in this point. Also

$$\lim_{(x,y)\to\left(\frac{D}{2},\frac{D}{2p}\right)} \frac{\partial F}{\partial x} \;=\; \big(A+\frac{1}{2}\big)D$$

$$\lim_{(x,y)\to\left(\frac{D}{2},\frac{D}{2p}\right)} \frac{\partial F}{\partial y} \;=\; p\big(A+\frac{1}{2}\big)D$$

with limits from any directions. Therefore the partial derivatives are continuous and $F(x,y)$ is continuously differentiable at $\big(\frac{D}{2}, \frac{D}{2p}\big)$.
Let us differentiate $F(x,y)$ along the direction of the Silkswap invariant

$$0 \;=\; dF(x,y) = \frac{\partial F}{\partial x}dx + \frac{\partial F}{\partial y}dy.$$

In the Theorem (A.5) we prove that $\frac{\partial F}{\partial y} > 0$. We can rearrange the last expression to conclude the proof. □

B Expression of the derivatives

Derivative expressions used for the calculation of D by Newton and Halley methods:

$$\frac{dF(D\,|\,x,y)}{dD} = A\bigg(\frac{4xpy}{D^2}\bigg)^{\gamma}\Big[(-2\gamma+1)\,(x+py-D)-D\Big] - \frac{D}{2}, \tag{B.1}$$

$$\frac{d^2F(D|x,y)}{dD^2} = A\bigg(\frac{4xpy}{D^2}\bigg)^{\gamma}\bigg[4\gamma-2+2\gamma(2\gamma-1)\bigg(\frac{x}{D}+\frac{py}{D}-1\bigg)\bigg] - \frac{1}{2}. \tag{B.2}$$

First derivative expressions for computing $\tilde{x}$ and $\tilde{z}$ by Newton and Halley methods:

$$\frac{d\tilde{F}(\tilde{x}\,|\,\tilde{z})}{d\tilde{x}} = A\big(4\tilde{x}\tilde{z}\big)^{\gamma}\bigg[\gamma\frac{\tilde{x}+\tilde{z}-1}{\tilde{x}}+1\bigg] + \tilde{z}, \tag{B.3}$$

$$\frac{d\tilde{F}(\tilde{z}\,|\,\tilde{x})}{d\tilde{z}} = A\big(4\tilde{x}\tilde{z}\big)^{\gamma}\bigg[\gamma\frac{\tilde{x}+\tilde{z}-1}{\tilde{z}}+1\bigg] + \tilde{x}. \tag{B.4}$$

for

$$\gamma := \begin{cases}\gamma_1, & \text{if } \tilde{x}\le\tilde{z}\\ \gamma_2, & \text{if } \tilde{x}>\tilde{z}.\end{cases} \tag{B.5}$$

Second derivatives:

$$\frac{d^2\tilde{F}(\tilde{x}\,|\,\tilde{z})}{d\tilde{x}^2} = 4\tilde{z}\gamma A\big(4\tilde{x}\tilde{z}\big)^{\gamma-1}\bigg[2+(\gamma-1)\frac{\tilde{x}+\tilde{z}-1}{\tilde{x}}\bigg], \tag{B.6}$$

$$\frac{d^2\tilde{F}(\tilde{z}\,|\,\tilde{x})}{d\tilde{z}^2} = 4\tilde{x}\gamma A\big(4\tilde{x}\tilde{z}\big)^{\gamma-1}\bigg[2+(\gamma-1)\frac{\tilde{x}+\tilde{z}-1}{\tilde{z}}\bigg]. \tag{B.7}$$

References

[Adams, 2018] Adams, H. (2018). Uniswap v1 whitepaper. *https://hackmd.io/@HaydenAdams/HJ9jLsfTz*.

[Adams et al., 2020] Adams, H., Zinsmeister, N., and Robinson, D. (2020). Uniswap v2 core. *https://uniswap.org/whitepaper.pdf*.

[Angeris and Chitra, 2020] Angeris, G. and Chitra, T. (2020). Improved price oracles: Constant function market makers. *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, pages 80 – 91.

[Duniya, 2021] Duniya, S. (2021). Silk: A privacy-preserving algorithmic burn stablecoin. *https://shadeprotocol.io/pdf/Silk_Whitepaper.pdf*.

[Egorov, 2019] Egorov, M. (2019). Stableswap - efficient mechanism for stablecoin liquidity. *https://curve.fi/files/stableswap-paper.pdf*.

[Egorov, 2021] Egorov, M. (2021). Automatic market-making with dynamic peg. *https://curve.fi/files/crypto-pools-paper.pdf*.

[Mohan, 2022] Mohan, V. (2022). Automated market makers and decentralized exchanges: a defi primer. *Financial Innovation*, 8(20).