AMM Optimization

Oakley, Trevor Lee

May 17, 2024

Abstract

AMMs have significant problems with capital efficiency and slippage. We demonstrate how these problems can be solved with a variation on sum and product invariants.

1 Introduction

A number of AMM solutions from Uniswap (v1 .. v4), Balancer (weighted product invariant [1]), Curve (v1 .. v2), Wombat, Platypus and more.

We examine some of these and improve upon them.

2 Balancer

Balancer protocols are described as follows [4]:

The Balancer AMM is based on *Balancer's value function*, which is the function $V : (\mathbb{R}_{>0})^N \to \mathbb{R}_{>0}$ that is defined by

$$V(b_1, b_2, \ldots, b_N) = \prod_{j=1}^{N} b_j^{w_j}$$

or:

$$V = \prod_t Q_t^{W_t}$$

Where V is the invariant value function, t ranges the tokens in the pool, Q_t represents the quantity of token t in the pool, and W_t is the constant weight of token t, such that $\sum_t W_t = 1$.

3 Curve

Introduction

We consider the analysis from [3] to diagnose the qualities and value of using the Curve invariants.

Curve v1 was called StableSwap and we consider still the algorithm to be the same name, v2 is for trading with non-pegged assets.

Version 1 is most relevant to RWA tokenization as the RWA value is fixed for long periods of time.

$$\prod_{j=1}^{N} b_j = C \quad \text{and} \quad \sum_{j=1}^{N} b_j = D$$

Sum Invariants

First we consider the sum of all tokens and consider this is an invariant and from this it is clear that no slippage happens, ie the spot price is the same as the effective price.

Consider an amount b of token i and receive an amount a of token o, from the constant sum invariant, we obtain that

$$\sum_{j=1}^{N} b_j = (b_i + b) + (b_o - a) + \sum_{\substack{j=1 \\ j \neq i,o}}^{N} b_j,$$

and hence, $b_i + b_o = b_i + b + b_o - a$, from where it follows that a = b, which implies that the effective price paid per unit of token o in terms of token i is

$$\frac{b}{a} = 1.$$

There is no slippage (spot price and the effective price are the same).

Combining Sum and Product Invariants

We wish to combine the sum and product invariants. Knowing the values for C and D, we simplify the problem by assuming all tokens have equal value:

$$\{b_i = b_j \quad \forall i, j\}$$

Then for a token of value b, we have:

$$\sum_{j=1}^{N} b_j = Nb = D => b = \frac{D}{N}$$

And for the product invariant:

$$\prod_{j=1}^{N} b_j = b^N = C = \left(\frac{D}{N}\right)^N$$

Therefore under the assumption all tokens have the same value, we have the product and sum invariants expressed in terms of D.

$$\prod_{j=1}^{N} b_j = \left(\frac{D}{N}\right)^N \quad \text{and} \quad \sum_{j=1}^{N} b_j = D$$

We can apply AM-GM to get the following, with the equality holding when all the tokens are equally priced:

$$\frac{D}{N} \geq \left(\prod_{j=1}^{N} b_j\right)^{\frac{1}{N}}$$

The Arithmetic Mean-Geometric Mean Inequality states that for any non-negative real numbers $a_1, a_2, \ldots, a_n$, the following inequality holds:

$$\frac{a_1 + a_2 + \cdots + a_n}{n} \geq \sqrt[n]{a_1 \cdot a_2 \cdots a_n}$$

where equality holds if and only if $a_1 = a_2 = \cdots = a_n$.

Proof of the Inequality of Arithmetic and Geometric Means

Proof for Two Numbers

Let a and b be non-negative real numbers. The arithmetic mean (AM) of these numbers is given by:

$$AM = \frac{a+b}{2}$$

The geometric mean (GM) is:

$$GM = \sqrt{ab}$$

We need to show that:

$$\frac{a+b}{2} \geq \sqrt{ab}$$

Start by manipulating the inequality:

$$a + b \geq 2\sqrt{ab}$$

Square both sides to remove the square root (this is valid because both sides are non-negative):

$$(a+b)^2 \geq 4ab$$

Expanding and simplifying:

$$a^2 + 2ab + b^2 \geq 4ab$$

$$a^2 - 2ab + b^2 \geq 0$$

$$(a-b)^2 \geq 0$$

This is always true, as the square of any real number (including zero) is non-negative. This completes the proof for two numbers.

General Proof by Induction

For n non-negative real numbers $a_1, a_2, \ldots, a_n$, the claim is:

$$\frac{a_1 + a_2 + \ldots + a_n}{n} \geq \sqrt[n]{a_1 a_2 \ldots a_n}$$

Base Case: For $n = 1$, the inequality holds trivially as the arithmetic mean and geometric mean of one number are equal to that number itself.

Inductive Step: Assume the inequality holds for $n = k$; that is:

$$\frac{a_1 + a_2 + \ldots + a_k}{k} \geq \sqrt[k]{a_1 a_2 \ldots a_k}$$

Now consider $n = k + 1$ numbers. We need to show that:

$$\frac{a_1 + a_2 + \ldots + a_k + a_{k+1}}{k+1} \geq \sqrt[k+1]{a_1 a_2 \ldots a_k a_{k+1}}$$

Using the inductive hypothesis, and a clever manipulation involving the AM-GM for two numbers (the mean of the first k numbers and a_{k+1}), one can show this holds, completing the inductive step.

This inductive proof shows that the AM-GM inequality holds for any number of non-negative real numbers, proving the statement universally.

Convex Combination

We start by considering the general form of a convex function:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$$

Applying this concept to the product and sum invariants, we have:

$$t\sum_{j=1}^{N} b_j + (1-t)\prod_{j=1}^{N} b_j = tD + (1-t)\left(\frac{D}{N}\right)^N$$

For t in $[0,1]$, we consider the expression. For this when t=0, we have the constant sum invariant and when t=1, we have the constant product invariant.

We wish to choose a function st: $g(0) = 0$ and $\lim_{x\to+\infty} g(x) = 1$.

The function $g(x) = \frac{x}{x+1}$ described in the text has specific characteristics: it approaches 1 as x approaches infinity, and it starts from 0 at $x = 0$.

This is designed so the product invariant and the sum invariant have different weights, and the product invariant dominates for low x and for high x then the sum invariant dominates.

Here are some examples of other functions that also exhibit these properties:

1. **Hyperbolic Tangent Function**:

$$g(x) = \tanh(x)$$

For large x, $\tanh(x)$ approaches 1. This function smoothly transitions from -1 to 1, but if shifted and scaled, it can be made to start from 0 and saturate at 1. For example:

$$g(x) = \frac{1+\tanh(x)}{2}$$

This modification starts at 0 when $x = 0$ and approaches 1 as $x \to \infty$.

2. **Arctangent Function**:

$$g(x) = \frac{2}{\pi}\arctan(x)$$

The arctangent function asymptotically approaches $\frac{\pi}{2}$ as $x \to \infty$. Scaling it by $\frac{2}{\pi}$ ensures it starts at 0 and approaches 1.

3. **Logistic Function**:

$$g(x) = \frac{1}{1+e^{-x}}$$

This is the standard logistic function often used in logistic regression and neural networks. It starts from 0 for large negative x and asymptotically approaches 1 as x becomes large.

4. **Exponential Decay Function of Reciprocal**:

$$g(x) = 1 - e^{-x}$$

This function also starts at 0 and asymptotically approaches 1 as x increases. It represents an exponential decay of the quantity e^{-x} subtracted from 1.

Each of these functions can be used as alternatives to $\frac{x}{x+1}$ where the requirement is to have a function that starts at 0 and approaches 1 as x increases.

3.1 Combinations of Sum and Product Invariants

$$\frac{x}{x+1}\sum_{j=1}^{N} b_j + \frac{1}{x+1}\prod_{j=1}^{N} b_j = \frac{x}{x+1}D + \frac{1}{x+1}\left(\frac{D}{N}\right)^N$$

or equivalently,

$$\boxed{x\sum_{j=1}^{N} b_j + \prod_{j=1}^{N} b_j = xD + \left(\frac{D}{N}\right)^N}$$

In addition, we want the parameter x to be dimensionless. To this end, we will substitute $x = \chi D^{N-1}$ and therefore:

$$\chi D^{N-1}\sum_{j=1}^{N} b_j + \prod_{j=1}^{N} b_j = \chi D^N + \left(\frac{D}{N}\right)^N.$$

Finally, we want the parameter χ to be dynamic rather than a constant chosen value. To this end, we substitute

$$\chi = A \cdot \frac{\prod_{j=1}^{N} b_j}{\left(\frac{D}{N}\right)^N},$$

where A is a positive number that is called *amplification coefficient*. In the actual code, there is a parameter labeled A, which is a positive integer and is equal to AN^{N-1}. Therefore, from now on, we will assume that AN^{N-1} is a positive integer—and this will indeed be needed later.

Two token model

For the pool which has two tokens:

$$f(b_1, b_2) = \frac{b_1 b_2}{\left(\frac{b_1+b_2}{2}\right)^2}$$

This shows maximum values when $b_1 = b_2$.

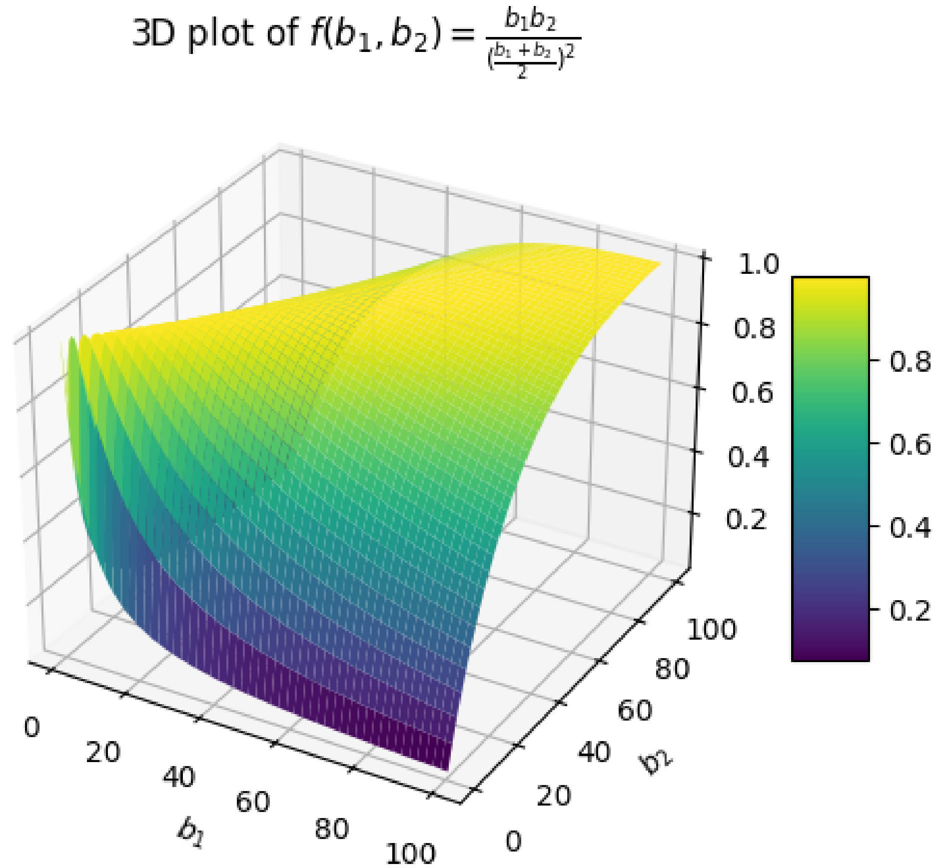

Figure 1: $f(b_1, b_2)$

$$A \cdot \left(\frac{D}{N}\right)^N \prod_{j=1}^{N} b_j + D^{N-1} \sum_{j=1}^{N} b_j + \prod_{j=1}^{N} b_j = A \cdot \left(\frac{D}{N}\right)^N + \left(\frac{D}{N}\right)^N$$

$$\boxed{AN^N \sum_{j=1}^{N} b_j + D = ADN^N + \frac{D^{N+1}}{N^N \prod_{j=1}^{N} b_j}}$$

We can optimize for A and in this example A is found to be 12.5

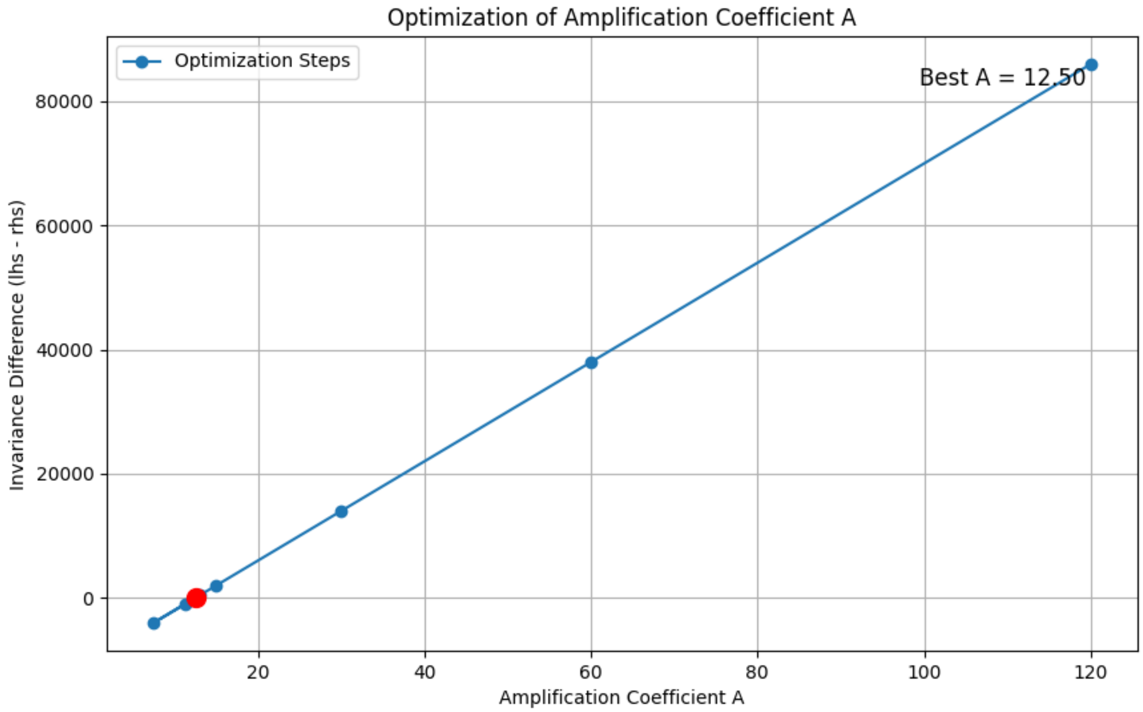

Figure 2: Amplification Factor

4 Wombat

The Wombat algorithms seeks to compete with the Stableswap one [2]. The analysis below shows some of the analysis.

1. In the two-token case for the Wombat formula, we have:

$$D = \sum(x_k - \frac{A}{x_k})$$

which simplifies to:

$$x - \frac{A}{x} + y - \frac{A}{y} = D$$

2. Consider the initial deposit when $x = y$:

$$D = x - \frac{A}{x} + y - \frac{A}{y} = 2x - \frac{2A}{x} \text{ substituting } y = x$$

which simplifies to:

$$Dx = 2x^2 - 2A$$

$$2x^2 - Dx - 2A = 0$$

3. Solving for a quadratic equation, we have $a = 2$, $b = -D$, $c = -2A$:

$$2x^2 - Dx - 2A = 0$$

4. Substituting $x = 50$, $y = 50$:

$$2 \cdot 2500 - 50D - 2A = 0$$

$$5000 - 50D - 2A = 0$$

$$\frac{5000 - 2A}{50} = D$$

This is how A and D are related.
For $A = 100, 200, 300$, calculate D values:
For $A = 100$:

$$D = \frac{5000 - 2 \cdot 100}{50} = \frac{5000 - 200}{50} = \frac{4800}{50} = 96$$

For $A = 200$:

$$D = \frac{5000 - 2 \cdot 200}{50} = \frac{5000 - 400}{50} = \frac{4600}{50} = 92$$

For $A = 300$:

$$D = \frac{5000 - 2 \cdot 300}{50} = \frac{5000 - 600}{50} = \frac{4400}{50} = 88$$

This graph plots for varying A and D is calculated, showing that the slippage reduces for a lower A.

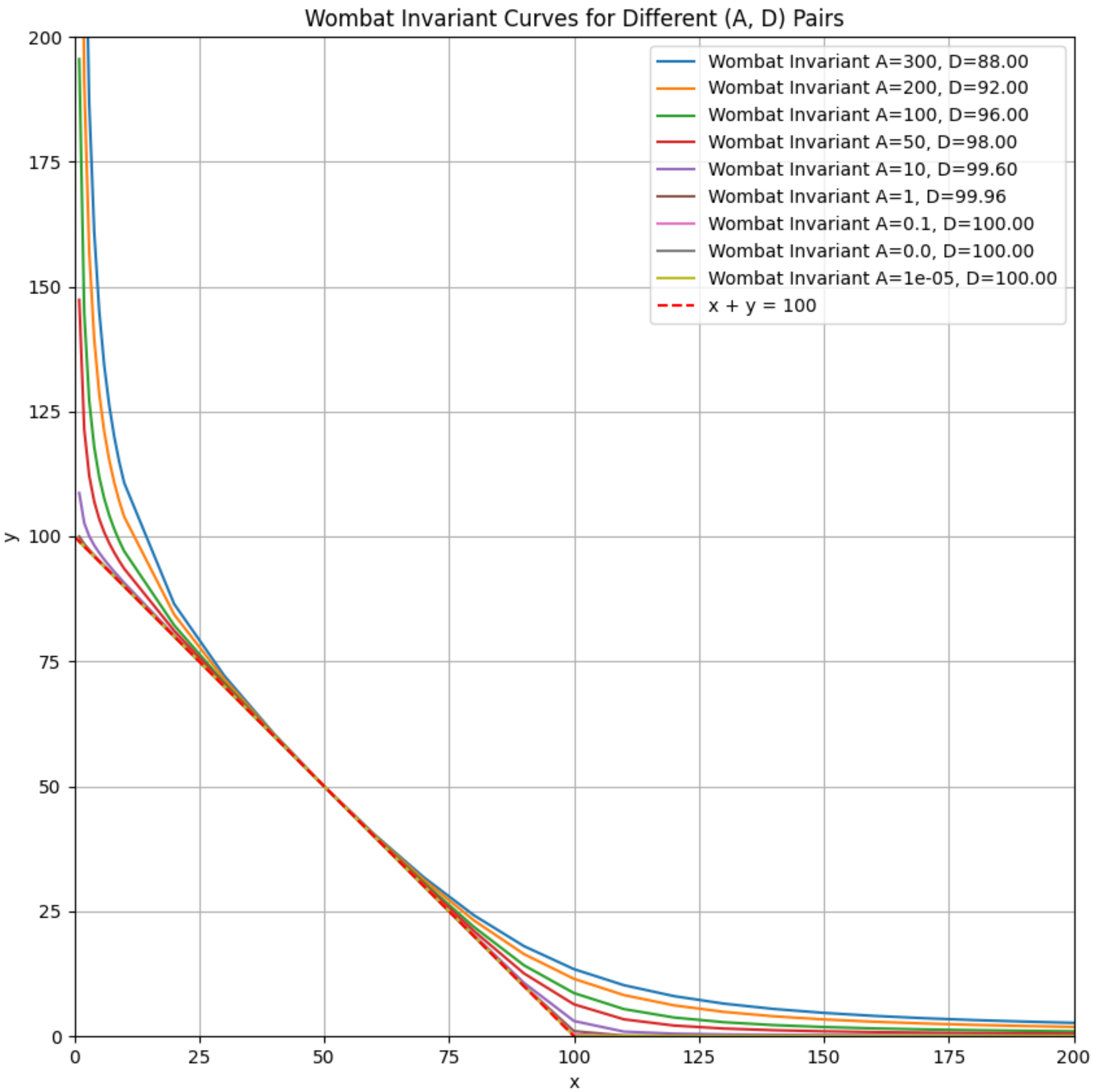

Figure 3: Wombat Plot

Results for slippage.

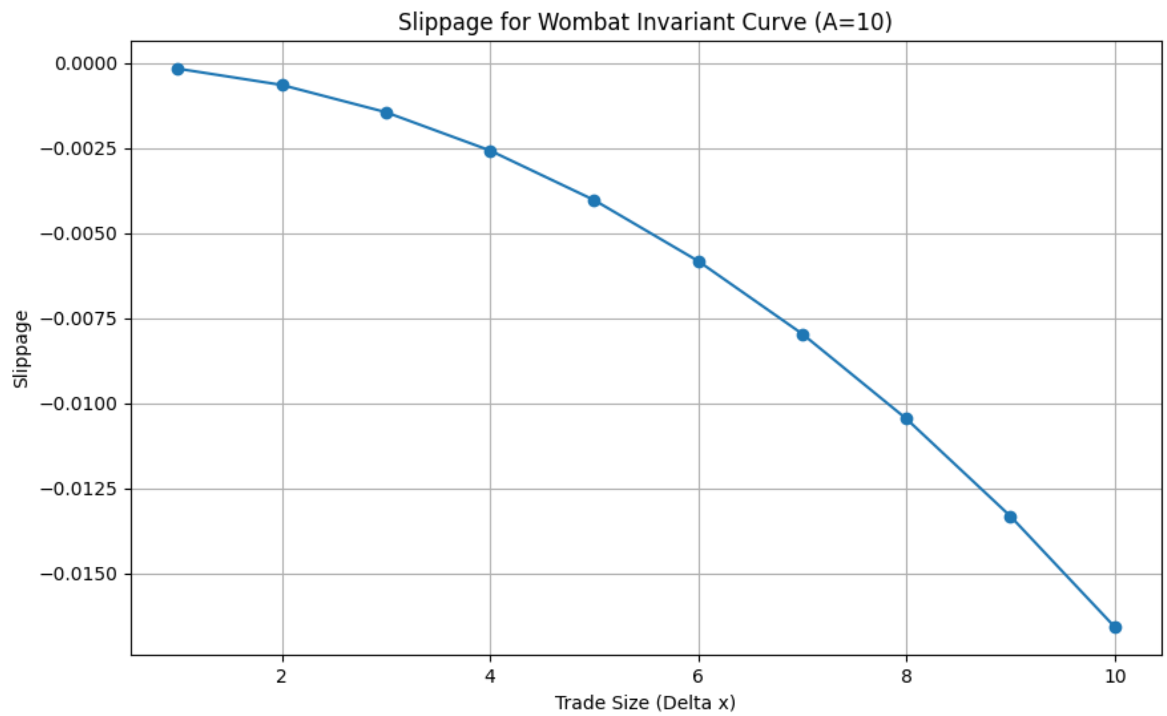

Figure 4: Wombat Slippage A =10

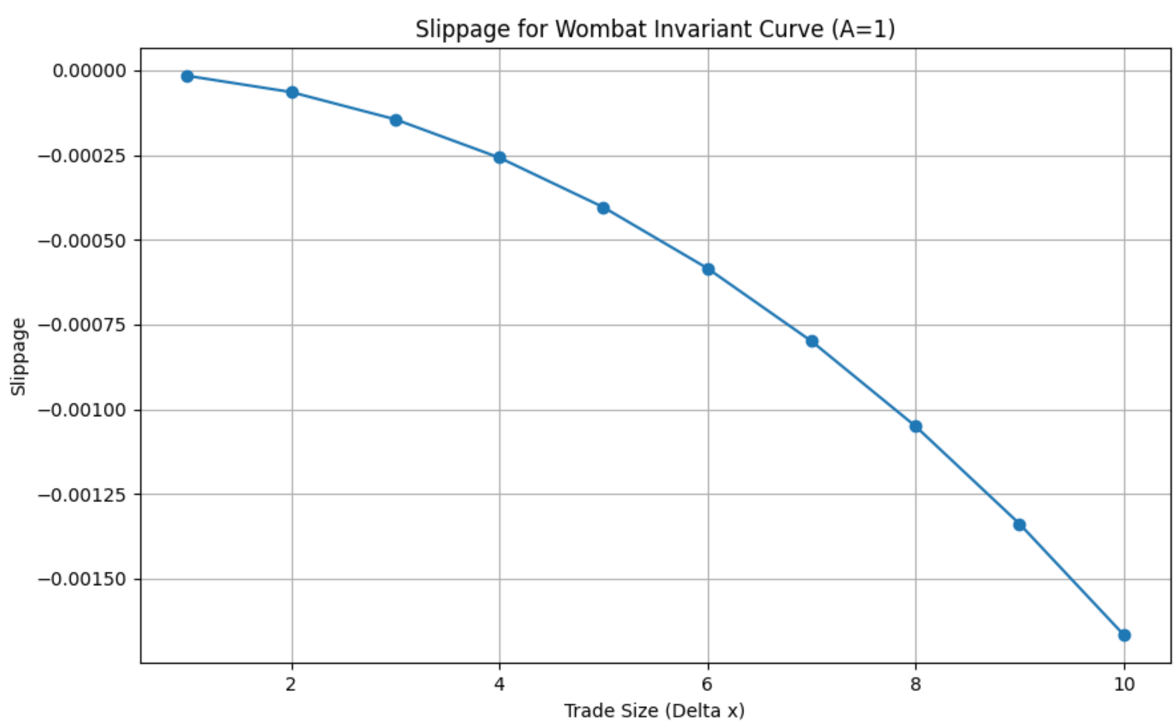

Figure 5: Wombat Slippage A =1

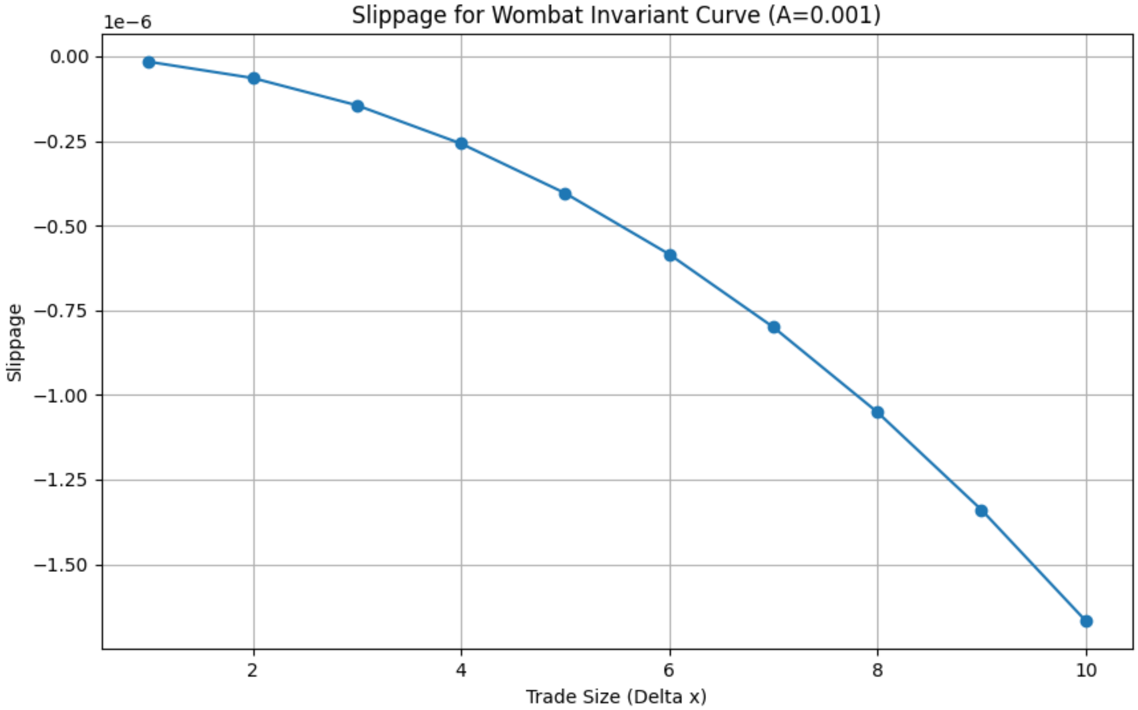

Figure 6: Wombat Slippage A=0.001

5 Test Results

What the results confirm is that for lower A the slippage is reduced when the trades are small but when the trades are larger the y return is 0. Therefore we in effect have no trade after the small trade boundary. That boundary is the intercept when y=0 on the x axis.

Hence in this design we can effect a series of small trades using the Wombat design but there is effectively no trade (i.e. no DEX) for the values outside this range. This is the basic dilemma to solve.

Slippage results for $A = 10$, $D = 99.6$

delta_x	**new_x**	**new_y**	**slippage**	**slippage (%)**
1.0	51.00	49.00	-0.000159	-0.02%
2.0	52.00	48.00	-0.000638	-0.03%
3.0	53.00	47.00	-0.001439	-0.05%
4.0	54.00	46.00	-0.002564	-0.06%
5.0	55.00	45.00	-0.004021	-0.08%
6.0	56.00	44.01	-0.005814	-0.10%

delta_x	new_x	new_y	slippage	slippage (%)
7.0	57.00	43.01	-0.007954	-0.11%
8.0	58.00	42.01	-0.010450	-0.13%
9.0	59.00	41.01	-0.013315	-0.15%
10.0	60.00	40.02	-0.016563	-0.17%

Slippage results for $A = 10$, $D = 99.6$

delta_x	new_x	new_y	slippage	slippage (%)
10.0	60.00	40.02	-0.016563	-0.17%
20.0	70.00	30.08	-0.075355	-0.38%
30.0	80.00	20.22	-0.219570	-0.73%
40.0	90.00	10.65	-0.650072	-1.63%
50.0	100.00	3.02	-3.015833	-6.03%
60.0	110.00	0.89	-10.892713	-18.15%
70.0	120.00	0.48	-20.480827	-29.26%
80.0	130.00	0.33	-30.326271	-37.91%
90.0	140.00	0.25	-40.246457	-44.72%
100.0	150.00	0.20	-50.197897	-50.20%
110.0	160.00	0.17	-60.165282	-54.70%
120.0	170.00	0.14	-70.141878	-58.45%
130.0	180.00	0.12	-80.124272	-61.63%
140.0	190.00	0.11	-90.110549	-64.36%
150.0	200.00	0.10	-100.099552	-66.73%
160.0	210.00	0.09	-110.090545	-68.81%

delta_x	new_x	new_y	slippage	slippage (%)
170.0	220.00	0.08	-120.083031	-70.64%
180.0	230.00	0.08	-130.076668	-72.26%
190.0	240.00	0.07	-140.071210	-73.72%
200.0	250.00	0.07	-150.066478	-75.03%

Slippage results for $A = 1$, $D = 99.96$

delta_x	new_x	new_y	slippage	slippage (%)
1.0	51.00	49.00	-0.000016	-0.00%
2.0	52.00	48.00	-0.000064	-0.00%
3.0	53.00	47.00	-0.000144	-0.00%
4.0	54.00	46.00	-0.000258	-0.01%
5.0	55.00	45.00	-0.000404	-0.01%
6.0	56.00	44.00	-0.000584	-0.01%
7.0	57.00	43.00	-0.000799	-0.01%
8.0	58.00	42.00	-0.001050	-0.01%
9.0	59.00	41.00	-0.001339	-0.01%
10.0	60.00	40.00	-0.001666	-0.02%

Slippage results for $A = 1$, $D = 99.96$

delta_x	new_x	new_y	slippage	slippage (%)
10.0	60.00	40.00	-0.001666	-0.02%
20.0	70.00	30.01	-0.007611	-0.04%
30.0	80.00	20.02	-0.022444	-0.07%
40.0	90.00	10.07	-0.070412	-0.18%

delta_x	new_x	new_y	slippage	slippage (%)
50.0	100.00	0.99	-0.985112	-1.97%
60.0	110.00	0.10	-10.098720	-16.83%
70.0	120.00	0.05	-20.049797	-28.64%
80.0	130.00	0.03	-30.033261	-37.54%
90.0	140.00	0.02	-40.024964	-44.47%
100.0	150.00	0.02	-50.019979	-50.02%
110.0	160.00	0.02	-60.016653	-54.56%
120.0	170.00	0.01	-70.014276	-58.35%
130.0	180.00	0.01	-80.012493	-61.55%
140.0	190.00	0.01	-90.011105	-64.29%
150.0	200.00	0.01	-100.009996	-66.67%
160.0	210.00	0.01	-110.009087	-68.76%
170.0	220.00	0.01	-120.008330	-70.59%
180.0	230.00	0.01	-130.007690	-72.23%
190.0	240.00	0.01	-140.007141	-73.69%
200.0	250.00	0.01	-150.006665	-75.00%

Slippage results for $A = 0.1$, $D = 99.99600000000001$

delta_x	new_x	new_y	slippage	slippage (%)
10.0	60.00	40.00	-0.000167	-0.00%
20.0	70.00	30.00	-0.000762	-0.00%
30.0	80.00	20.00	-0.002249	-0.01%
40.0	90.00	10.01	-0.007104	-0.02%

delta_x	new_x	new_y	slippage	slippage (%)
50.0	100.00	0.31	-0.314731	-0.63%
60.0	110.00	0.01	-10.009987	-16.68%
70.0	120.00	0.00	-20.004998	-28.58%
80.0	130.00	0.00	-30.003333	-37.50%
90.0	140.00	0.00	-40.002500	-44.45%
100.0	150.00	0.00	-50.002000	-50.00%
110.0	160.00	0.00	-60.001667	-54.55%
120.0	170.00	0.00	-70.001428	-58.33%
130.0	180.00	0.00	-80.001250	-61.54%
140.0	190.00	0.00	-90.001111	-64.29%
150.0	200.00	0.00	-100.001000	-66.67%
160.0	210.00	0.00	-110.000909	-68.75%
170.0	220.00	0.00	-120.000833	-70.59%
180.0	230.00	0.00	-130.000769	-72.22%
190.0	240.00	0.00	-140.000714	-73.68%
200.0	250.00	0.00	-150.000667	-75.00%

6 Platypus

6.1 Coverage Ratio

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

$$r'_i = \frac{A_i + \Delta_i}{L_i} > \frac{A_i}{L_i} = r_i$$

$$r'_j = \frac{A_j - f^*_{i\to j}\Delta_i}{L_j} < \frac{A_j}{L_j} = r_j.$$

[5] considers the ability of an AMM not to have a pair deposited but a single currency and liquidity is improved by a coverage ratio.

We apply the findings for [5] and define $g(r)$ for slippage and $g'(r)$ for marginal slippage. The principles of Riemann integration and Lebesgue are applied.

Concepts of Riemann and Lebesgue Integration

Riemann Integration

- **Riemann Integral**: The Riemann integral is a method for measuring the area under a curve by approximating it with a sum of areas of rectangles. It is defined for functions that are bounded and integrable over a closed interval.
- **Use in Slippage Function**: In the context of the slippage function $g(r)$, Riemann integration is used to compute the cumulative slippage from the marginal slippage function $g'(r)$. This involves summing up the incremental slippage contributions over the range of the coverage ratio r.

Lebesgue Integration

- **Lebesgue Integral**: The Lebesgue integral generalizes the Riemann integral by allowing integration over more general sets and for more general functions. It is particularly useful for dealing with functions that have discontinuities or are unbounded.
- **Lebesgue's Theorem**: One important aspect of Lebesgue integration is the Dominated Convergence Theorem, which allows for the interchange of limits and integrals under certain conditions. This is relevant when dealing with integrals of functions that change dynamically.

Derivation of the Slippage Function

1. **Marginal Slippage Function** $g'(r)$
   - The marginal slippage function is defined piecewise to capture different behaviors of slippage based on the coverage ratio r:
   $$g'(r) = \begin{cases} -1, & \text{for } -\frac{kn}{r^{n+1}} < -1 \\ -\frac{kn}{r^{n+1}}, & \text{for } -\frac{kn}{r^{n+1}} \in [-1, 0] \end{cases}$$
2. **Integrating** $g'(r)$ **to Obtain** $g(r)$
   - To find the cumulative slippage function $g(r)$, we integrate $g'(r)$:
   $$g(r) = \int g'(r)\, dr$$
   - For $-\frac{kn}{r^{n+1}} < -1$:
   $$g(r) = \int -1\, dr = -r + C_1$$
   - For $-\frac{kn}{r^{n+1}} \in [-1, 0]$:
   $$g(r) = \int -\frac{kn}{r^{n+1}}\, dr = \frac{k}{r^n} + C_2$$
3. **Constants of Integration**

- The constants C_1 and C_2 are determined by boundary conditions or continuity requirements at the points where the pieces meet.

Proof Sketch and Relevance

Applying Riemann and Lebesgue Integration

- **Riemann Integration**: Used to directly integrate the piecewise-defined $g'(r)$ over specific intervals. This approach is straightforward for continuous and piecewise continuous functions.

- **Lebesgue Integration**: Justifies the integration process, especially if $g'(r)$ exhibits discontinuities. Lebesgue's Dominated Convergence Theorem ensures that under certain conditions, the integration and limit processes can be interchanged, making the integral well-defined.

Step-by-Step Proof Sketch

1. **Define $g'(r)$ Piecewise**

   - Ensure that $g'(r)$ meets the criteria for Riemann integrability within each piece.

2. **Integrate Each Piece**

   - Compute the integral for each interval:
     - For $-\frac{kn}{r^{n+1}} < -1$:
     $$g(r) = \int -1\, dr = -r + C_1$$
     - For $-\frac{kn}{r^{n+1}} \in [-1, 0]$:
     $$g(r) = \int -\frac{kn}{r^{n+1}}\, dr = \frac{k}{r^n} + C_2$$

3. **Ensure Continuity at Boundaries**

   - Determine constants C_1 and C_2 to ensure that $g(r)$ is continuous at the boundaries where the pieces meet.

Example of Boundary Condition

- Suppose the transition occurs at $r = r_0$ where:

$$-\frac{kn}{r_0^{n+1}} = -1 \implies r_0 = \left(\frac{kn}{1}\right)^{\frac{1}{n+1}}$$

- Ensure $g(r)$ is continuous at r_0:
$$-r_0 + C_1 = \frac{k}{r_0^n} + C_2$$

  Solve for C_1 and C_2 to maintain continuity.

Summary

The provided formula for $g(r)$ and $g'(r)$ leverages the principles of Riemann and Lebesgue integration to ensure well-defined, continuous, and integrable slippage functions. These functions are designed to manage slippage dynamically, penalizing large coverage ratio changes and maintaining the stability of the Platypus AMM. The theoretical foundations from integration theory provide the mathematical rigor to justify the piecewise definitions and ensure smooth behavior.

7 Pools

7.1 Weighted Pools

7.2 Composable Stable Pools

7.3 MetaStable Pools

7.4 Boosted Pools

References

[1] Weighted pools. https://app.balancer.fi/#/risks#weighted-pools. Accessed: 2023-05-07.

[2] Jen Houng Lie, Tony WH Wong, and Alex Yin-ting Lee. Wombat—an efficient stableswap algorithm. In *The International Conference on Mathematical Research for Blockchain Economy*, pages 233–251. Springer, 2022.

[3] Miguel Ottina, Peter Johannes Steffensen, and Jesper Kristensen. Introduction to amms. In *Automated Market Makers: A Practical Guide to Decentralized Exchanges and Cryptocurrency Trading*, pages 1–12. Springer, 2023.

[4] Neil Pattanaik. Minimizing impermanent loss in asset-pair balancer pools. 2023.

[5] Platypus Team. Platypus amm yellow paper, June 2022.