

Understanding Slippage in Automated Market Makers: A Unified Deviation Framework and Survey

Xukang Shang ¹, Jian Zheng ², Jianwen Yuan ² and Huawei Huang ^{2,*}

¹ School of Philosophy, Fudan University, Shanghai 200433, China; xkshang22@m.fudan.edu.cn

² School of Software Engineering, Sun Yat-sen University, Zhuhai 519082, China; zhengj79@mail3.sysu.edu.cn (J.Z.); yuanjw26@mail.sysu.edu.cn (J.Y.).

* Correspondence: huanghw28@mail.sysu.edu.cn

Abstract

As a major cost in decentralized asset swapping, slippage is commonly measured as the relative deviation between realized and expected outcomes. However, existing studies adopt different benchmarks, leading to ambiguity in terminology and the positioning of prior works. We propose a unified deviation-based framework for understanding slippage, which reconciles existing definitions and enables further developments. Within this framework, the overall deviation can be decomposed into two components: endogenous deviation, arising from the swap itself, and exogenous deviation, which is induced by intervening swaps during the latency period. We review the existing literature along with the two components, covering their theoretical properties, further decompositions, and mitigation approaches. Finally, we outline several directions for future research.

Keywords: decentralized finance (DeFi); decentralized exchange (DEX); automated market maker (AMM); slippage; maximal extractable value (MEV)

1. Introduction

Decentralized asset swapping is enabled by blockchain technology with automated market makers (AMMs) emerging as the dominant realization in practice. Although AMMs offer advantages over centralized exchanges in terms of transparency, auditability, and accessibility, they also incur inherent costs [1,2].

Slippage, as a major cost of AMMs, has attracted significant attention from both scholars and practitioners. However, we find that existing studies adopt different benchmarks to measure slippage, leading to confusion about the definition and difficulties in positioning existing works within a unified framework.

We aim to bridge this gap by proposing a unified deviation framework and organizing existing works accordingly. The proposed deviation formulation reconciles different benchmark choices within a common analytical framework. Furthermore, it admits a decomposition into two separable components: endogenous deviation and exogenous deviation. Beyond a common formal language, the proposed framework accommodates a broader range of slippage-related studies that are often treated separately under existing classifications. The framework also suggests a new perspective for measuring slippage, allowing different sources to be distinguished and analyzed explicitly.

We adopt several simplifying assumptions at the cost of placing certain topics outside the scope. First, we focus exclusively on swap events and do not explicitly model liquidity provision and withdrawal. As a result, mechanisms such as Just-in-Time liquidity [3] are



Academic Editors: Keke Gai and Liehuang Zhu

Received: 4 May 2026

Revised: 22 June 2026

Accepted: 24 June 2026

Published: 1 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

beyond the scope. Second, we focus on exact-input swaps, while exact-output orders are omitted for simplicity. Finally, transaction fees are abstracted away in the derivation of the core framework to emphasize the fundamental sources of deviation. But when fees play an important role, such as in optimal routing problems, we will discuss their effects.

To identify the relevant literature, we conducted a broad search of academic databases including Google Scholar, ACM Digital Library, IEEE Xplore, SSRN, and arXiv. Keywords included “automated market maker”, “slippage”, “MEV”, “DEX routing”, and related terms. The survey primarily covers the literature published between 2020 and 2026, which is supplemented by protocol whitepapers and technical documentation.

The rest of the paper is organized as follows. Section 2 introduces the preliminaries on slippage. Section 3 presents a bottom-up modeling of slippage and introduces the unified definition. Section 4 and Section 5 analyze endogenous and exogenous deviations, respectively, covering their properties and corresponding mitigation strategies. Section 6 discusses open issues, and Section 7 concludes. Figure 1 illustrates the main body of this paper.

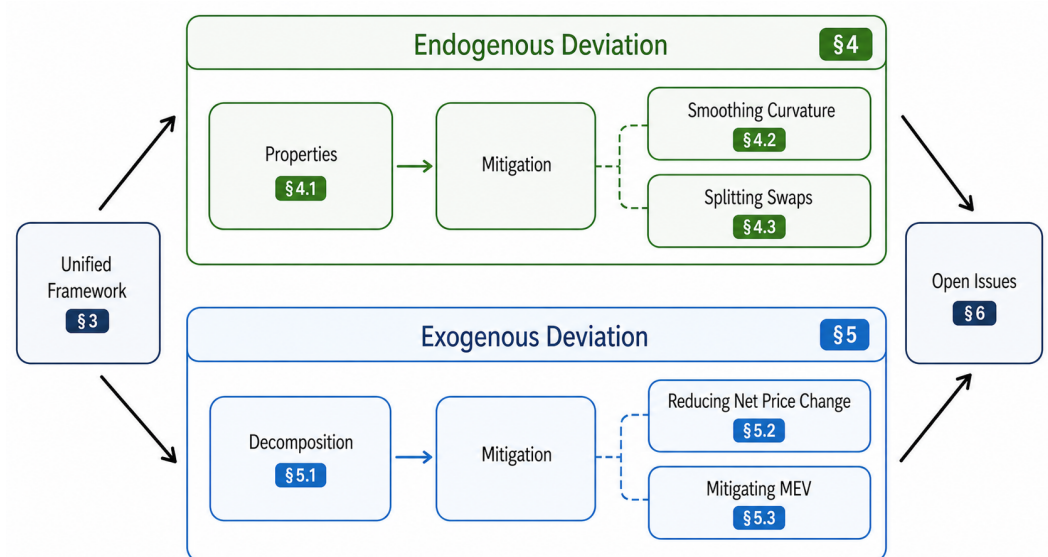


Figure 1. Overview of the proposed framework and organization of the literature.

2. Preliminaries

DeFi (Decentralized Finance). Decentralized finance refers to blockchain-based financial systems that operate without centralized intermediaries. DeFi protocols provide services such as lending, borrowing, derivatives, and asset swapping through smart contracts. The permissionless and transparent nature of DeFi allows anyone to interact with financial instruments directly but also introduces unique execution considerations. The total value locked in DeFi protocols exceeded USD 85 billion as of May 2026 [4].

DEX (Decentralized Exchange). A decentralized exchange enables users to swap digital assets without relying on a central authority [2]. Trades are executed on-chain via smart contracts, making DEXs non-custodial while exposing them to latency and transaction ordering effects. DEXs can be broadly classified into three types: order-book-based, AMM-based, and hybrid [5]. Among these, AMM-based DEXs are the most widely adopted in practice.

AMM (Automated Market Maker). An AMM is a type of DEX that determines asset prices algorithmically. In a typical AMM, liquidity is provided by users known as liquidity providers (LPs), who deposit pairs of assets (e.g., wETH and USDC) into a liquidity pool. In return, LPs receive shares of the pool and earn fees from swaps.

AMMs support several fundamental operations. Swap is the core trading operation, where users exchange one asset for another against the pool. Mint refers to adding liquidity to the pool, where LPs deposit assets and receive pool shares. Burn refers to removing liquidity, where LPs redeem their shares for the underlying assets. Prior work has shown that mint and burn events occur much less frequently than swap events [6]. Figure 2 provides an illustrative example.

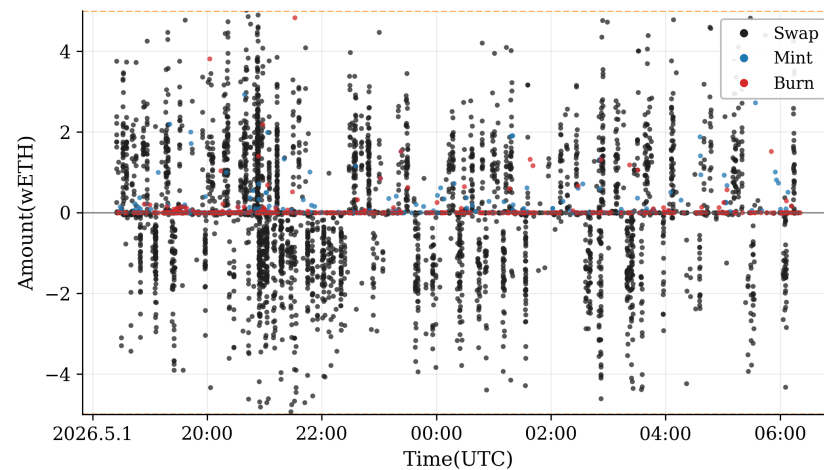


Figure 2. Event amounts and temporal distribution of swap, mint, and burn in the Arbitrum wETH–USDC pool (fee tier: 0.05%), which is the largest pool on Uniswap V3 by trading volume in April 2026. Pool address: 0xC6962004f452bE9203591991D15f6b388e09E8D0; block range: 458,350,000–458,520,000. The display is clipped to $[-5, 5]$ for readability. In total, 120 out of 4299 events are clipped. During this period, swaps account for 85.8% of events, while mint and burn account for 5.6% and 8.7%, respectively. For swap events, positive amounts correspond to swaps from wETH to USDC, while negative amounts indicate swaps in the opposite direction. The data extraction and plotting scripts are provided in the Supplementary Materials.

MEV (Miner/Maximal Extractable Value). MEV refers to profits that validators or block builders can extract by reordering, including, or excluding transactions within a block [7]. In AMMs, MEV often manifests as adversarial slippage [1], where the execution order of swaps can significantly affect the realized output amounts. A canonical example is the sandwich attack [7], in which an adversary observes a pending swap, inserts a transaction before it to move the price unfavorably, and then places another transaction after it to revert the price and capture profit. As a result, the victim trade is executed at a worse price, leading to increased slippage.

3. Definition and Decomposition

3.1. AMMs and Swaps

For all AMMs that enable swaps between asset X and Y , denote their set by I . For AMM $i \in I$, denote its reserves of X and Y by x_i and y_i , respectively, where $x_i, y_i \in \mathbb{R}_+$. Suppose their relationship is governed by a constant function (the literature uses different terminology for this function, e.g., “conservation function” [2,5] and “constant function” [8,9]). $Z_i(x_i, y_i) = 0$, which can be equivalently represented as $y_i = f_i(x_i)$. For simplicity, we focus on a single AMM for now and drop the subscript i , writing x, y , and $f(x)$ instead.

The state of the AMM can be represented by $(x, f(x))$. The state evolves with each event (swap, mint, or burn), which updates the reserves and potentially the function $f(x)$. Although the function may change, we assume that $f'(x) \leq 0$ holds and $f(x)$ is twice-

differentiable [10,11]. Thus, $-f'(x)$ represents the marginal exchange rate or the marginal price of X . Define $p(x) = -f'(x)$ as the pricing function.

The evolution of AMM states is driven by a sequence of events indexed by $\mathbb{N}$, where each $j \in \mathbb{N}$ corresponds to the j -th event. For simplicity, we consider the swap event only. Additionally, we ignore the potential impact of swap event on $f(x)$ to avoid notational complexity. Under these assumptions, we can denote the state before the n -th swap by $(x_n, f(x_n))$, which transitions to $(x_{n+1}, f(x_{n+1}))$ after the swap. We assume the swap is from X to Y , and we denote the input as Δx_n and output as Δy_n . Ignoring fees, we have

$$\Delta x_n = x_{n+1} - x_n \geq 0, n \in \mathbb{N}, \quad (1)$$

$$\Delta y_n = f(x_n) - f(x_n + \Delta x_n) = \int_{x_n}^{x_n + \Delta x_n} p(t) dt \geq 0, n \in \mathbb{N}. \quad (2)$$

For this swap to be executed as the n -th event, it must have been submitted earlier. We assume that both the query and the submission occur at the time of event o , where $o \in \mathbb{N}$. The quoted output from the query, denoted by $\widehat{\Delta y}_n$, is therefore determined by the state right before event o . Formally,

$$\widehat{\Delta y}_n = f(x_o) - f(x_o + \Delta x_n), o, n \in \mathbb{N}, o < n. \quad (3)$$

3.2. Slippages

For the swap $(\Delta x_n, \Delta y_n)$, slippage arises from the deviation between the realized price and the baseline price. However, existing works differ in the choice of these prices, leading to different definitions of slippage. Before formalizing the definition, Figure 3 illustrates the lifecycle of a swap—query, submission, and execution—and how slippage arises during this process.

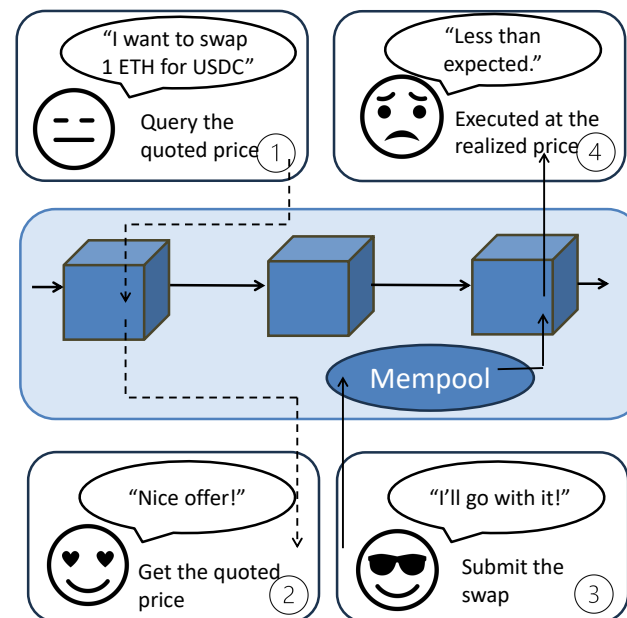


Figure 3. The lifecycle of a swap and the emergence of slippage. ① The user queries the AMM for a quoted price. ② The AMM simulates the swap based on the latest on-chain state, which we denote by $(x_o, f(x_o))$, and returns a quote. ③ If the quote is acceptable, the user submits the swap transaction, which is then placed in the mempool. ④ The transaction is later selected and executed by a block proposer, at which point the AMM state has evolved to $(x_n, f(x_n))$, resulting in a deviation between the quoted and realized outcomes. Dashed arrows indicate operations that do not modify the on-chain state, while solid arrows indicate state-changing operations.

The theoretical literature [2,6,10] typically ignores the latency between query and execution and thus treats the quoted price as the realized price while taking the spot price at query as the baseline. That is,

$$\text{slippage} = \frac{\text{quoted price}}{\text{spot price}} - 1 = \frac{\widehat{\Delta y_n} - p(x_o)\Delta x_n}{p(x_o)\Delta x_n}, o, n \in \mathbb{N}, o < n. \quad (4)$$

This definition abstracts away the impact of other swaps, thereby focusing on the intrinsic constant function of AMMs. However, it differs from the trader's perception, limiting its applicability in practice. Practical works [1,12] often refer to the former definition as "price impact" and instead define slippage as

$$\text{slippage} = \frac{\text{realized price}}{\text{quoted price}} - 1 = \frac{\Delta y_n - \widehat{\Delta y_n}}{\widehat{\Delta y_n}}, n \in \mathbb{N}. \quad (5)$$

This definition aligns with the trader's perspective and is well suited for empirical analysis. However, it lacks a tractable mathematical formulation, which limits its use in theoretical modeling. In addition, beyond the two definitions above, the Loss-Versus-Rebalancing literature [13] takes CEX prices as the baseline, which is relatively uncommon and is therefore not discussed further in this paper.

To unify Equations (4) and (5) and address their respective limitations, we propose the following definition.

Definition 1 (Slippage). *For a swap $(\Delta x_n, \Delta y_n)$ that is queried and submitted at event o and executed as event n , we define its slippage s_n as*

$$s_n = \frac{\text{realized price}}{\text{spot price}} - 1 = \frac{\Delta y_n - p(x_o)\Delta x_n}{p(x_o)\Delta x_n}, o, n \in \mathbb{N}, o < n. \quad (6)$$

Note that all three definitions of slippage, although defined in terms of prices, can be equivalently reformulated in terms of amounts. Slippage can then be interpreted as the relative deviation of the realized amount from its baseline value. To avoid unnecessary algebraic complexity, we focus on the deviation term (the numerators) rather than the ratio in the following analysis.

3.3. Deviations

The deviation term in our definition Equation (6), denoted by d_n , can be decomposed as

$$d_n = \underbrace{(\Delta y_n - \widehat{\Delta y_n})}_{\text{from Equation (5)}} + \underbrace{(\widehat{\Delta y_n} - p(x_o)\Delta x_n)}_{\text{from Equation (4)}}, o, n \in \mathbb{N}, o < n. \quad (7)$$

This shows that the proposed deviation consists of two components corresponding to the two existing definitions of slippage. Empirical evidence on their relative magnitudes can be found in Adams et al. [1].

Although this decomposition aligns with existing works, it does not provide a clean separation based on the origins of the deviation. The second term arises purely from the swap itself, whereas the first term, while primarily driven by other swaps, also contains a component induced by the swap itself (intuitively, the market never actually realizes the second term; thus, the impact of the swap itself must be absorbed into the first term).

We therefore propose an alternative decomposition:

$$d_n = (\Delta y_n - p(x_n)\Delta x_n) + (p(x_n)\Delta x_n - p(x_o)\Delta x_n), o, n \in \mathbb{N}, o < n. \quad (8)$$

The first term is generated at execution and is induced purely by the swap itself. The second term depends on both Δx_n and the change in spot price; however, since the swap size is an intrinsic parameter of swaps and is fixed upon submission, variations in this term can be attributed to exogenous price movements.

Definition 2 (Exogenous and Endogenous Deviation). *For a swap queried and submitted at event o and executed as event n , with input Δx_n of asset X and output Δy_n of asset Y , we define $d_n = d_n^{\text{exo}} + d_n^{\text{endo}}$, where*

- *Exogenous deviation:*

$$d_n^{\text{exo}} = (p(x_n) - p(x_o))\Delta x_n, o, n \in \mathbb{N}, o < n,$$

which is induced by other swaps during the latency period and scaled linearly by the size of this swap;

- *Endogenous deviation:*

$$d_n^{\text{endo}} = \Delta y_n - p(x_n)\Delta x_n, n \in \mathbb{N},$$

which arises purely from the swap itself.

Compared with existing definitions, Definition 2 offers several advantages. Equation (4) ignores the effects of intervening swaps, which actually constitute a large part of trading costs in practice [1]. Alternatively, if we take Equation (5) as the slippage and Equation (4) as the “price impact”, then the swap’s own impact is counted twice.

While the distinction in Definition 2 provides a useful perspective, it should be interpreted with caution. For a given swap, its exogenous deviation is generated by the endogenous deviations of intervening swaps during the latency period, mechanically coupling the two components across the swap sequence. Consequently, minimizing either endogenous or exogenous deviation should not be regarded as a welfare objective per se. For example, splitting a swap into multiple sequential swaps can relabel part of the endogenous deviation as exogenous without changing the realized output. This example is illustrated explicitly in Section 4.3.

Nevertheless, the distinction remains analytically useful as a framework for classifying and reviewing the literature according to the underlying sources of deviation.

4. Slippage from Inside: Endogenous Deviation

This section analyzes the properties of endogenous deviation and summarizes two primary approaches to mitigate it: smoothing curvature and splitting swaps.

4.1. Local and Global Properties

The properties of endogenous deviation can be revealed through further mathematical transformation.

Cartea et al. [6] apply a second-order Taylor expansion to characterize d_n^{endo} , yielding

$$\begin{aligned} d_n^{\text{endo}} &= \Delta y_n - p(x_n)\Delta x_n \\ &= \left(-f'(x_n)\Delta x_n - \frac{1}{2}f''(x_n)(\Delta x_n)^2 + o((\Delta x_n)^2) \right) - p(x_n)\Delta x_n \\ &\approx -\frac{1}{2}f''(x_n)(\Delta x_n)^2, \end{aligned} \quad (9)$$

where $x_n, \Delta x_n \in \mathbb{R}_+, n \in \mathbb{N}$. This shows that the effect of Δx_n is quadratic, implying that larger trades incur disproportionately higher endogenous deviation. Another factor is $f''(x_n)$, which is a measure of the local curvature of the constant function. (Strictly speaking,

curvature has a precise geometric definition. For simplicity, we use curvature in this paper to refer to the second derivative, which captures the local bending of the pricing function.) Since Δx_n must be nonzero for valid swaps, d_n^{endo} arises essentially from the convexity of the constant function (i.e., $f''(x_n) > 0$). In particular, if the function is designed to be linear (i.e., $f''(x_n) = 0$), then d_n^{endo} vanishes. Pool depth, the commonly used proxy for liquidity, does not explicitly appear in Equation (9): it influences d_n^{endo} only insofar as it affects the curvature of the constant function.

While Equation (9) shows the local properties of d_n^{endo} , Engel and Herlihy [11] provide a global characterization. They proved that the impact of a swap on itself:

- No AMM can bound this impact for bounded-size trades. For example, for an AMM $(x, \frac{1}{x})$, the endogenous deviation of a swap grows without bound when x approaches 0.
- The geometric representation of such impact—given by the angle between the tangent and the x -axis—has a total variation of $\frac{\pi}{2}$ across the price range. Since angles are additive, reducing the impact over one interval necessarily increases it elsewhere, which is independent of the AMM design.

They demonstrate these by introducing the concepts of linear and angular slippage, which capture the same impact as our concept of endogenous deviation, though in different mathematical form.

4.2. Smoothing Curvature

Essentially, curvature is determined by two types of variables: internal variables, which only exist as part of AMM's mechanism, and external variables, which exist regardless of its presence. We thus classify existing curvature-smoothing approaches into two categories: the internal approach, which relies solely on optimized designs of internal variables, and the hybrid approach, which integrates external information.

The internal approach constitutes the mainstream practice in the industry. Table 1 summarizes representative designs. Uniswap V2 [14] adopts the simplest constant product (CP) invariant, serving as a baseline for AMMs. Uniswap V3 [8] introduces concentrated liquidity, which enlarges the denominator of the curvature through virtual reserves, thereby smoothing the curve over the price range (P_a, P_b) . The presence of P_b in its curvature formulation also indicates that curvature is defined piecewise across different price ranges. When a swap spans such ranges, the resulting deviation must be accumulated over segments with distinct curvatures. Further, Balancer V1 [15] generalizes the CP invariant to a multi-asset setting where curvature can be adjusted through asset weights.

In contrast, the linear pool in Balancer V2 [16] takes a constant sum (CS) approach and achieves zero curvature. Although it eliminates endogenous deviation, it applies only to pairs that can be assigned a predefined exchange rate R_{adj} ex ante, such as ETH–wETH. Another design represented by Curve V1 [17], combining CP and CS, achieves low curvatures when the reserve ratio $\frac{x}{y}$ is close to 1, while the curvature increases rapidly as the ratio deviates from 1 (see Appendix A.5). Its application is therefore limited to assets with stable exchange rates, such as stablecoins.

In addition to the internal approach, the hybrid approach that integrates external information with internal variables has been explored in both industry and academia.

External information can help concentrate liquidity around an appropriate price range, resulting in a flatter curve within it. Curve V2 [18], building upon a CP+CS architecture, concentrates liquidity around the external price provided by an oracle. DODO [19] also exploits external price as a reference but is based on a distinct algorithm called Proactive Market Maker, which actively shifts the pricing curve toward the market price. Beyond price anchoring, Predictive Market Makers [20] leverage reinforcement learning to forecast future price ranges and reallocate liquidity ex ante within a CP framework.

Table 1. Representative internal curvature designs.

AMM Type	Representative DEX	Constant Function	Curvature Formula
CP	Uniswap V2	$xy = k$	$\frac{2k}{x^3}$
CP + concentrated liquidity	Uniswap V3	$(x + \frac{\sqrt{k}}{\sqrt{P_b}})(y + \sqrt{k}\sqrt{P_a}) = k$	$\frac{2k}{(x + \frac{\sqrt{k}}{\sqrt{P_b}})^3}$
CP + multiple assets	Balancer V1	$x^{w_x} y^{w_y} \prod_{i=3}^n x_i^{w_i} = k$	$\frac{w_x}{w_y} \left(\frac{w_x}{w_y} + 1 \right) \frac{y}{x^2}$
CS	Balancer V2 (Linear Pool)	$x + R_{\text{adj}}y = k$	0
CP + CS	Curve V1	$An^n(x + y + \sum_{i=3}^n x_i) + k = Akn^n + \frac{k^{n+1}}{n^n xy \prod_{i=3}^n x_i}$	See Appendix A.5

CP denotes constant product, and CS denotes constant sum. In all equations, k represents the invariant. x and y denote the reserves of asset X (input asset) and asset Y (output asset), respectively. x_i denotes the reserves of asset i in a multi-asset pool, and n is the total number of assets. For specific protocols: in Uniswap V3, P_a and P_b denote the lower and upper bounds of the active price range containing the current price P ; in Balancer V1, w_i represents the weight of asset i ; in Balancer V2, R_{adj} denotes the predefined exchange rate, which is adjusted via fees when the reserve ratio deviates beyond some threshold; in Curve V1, A is the amplification coefficient. Note that we adopt a unified notation to facilitate the comparison of curvatures. For the standard notation and detailed derivations, see Appendix A.

External information can be used to directly determine prices. Dynamic Market Makers [21,22] synchronize the spot price with the external price before each execution by introducing additional parameters into the CP or CS framework. This design does not reduce endogenous deviation but promotes fairness as the pre-swap conditions are synchronized. Its main advantage lies in eliminating exogenous deviation, which will be discussed in the next section.

External information can also be used to dynamically customize the pricing function. Uniswap V4 Hooks [23] provide a general framework for building custom AMMs, allowing different pricing curves to be activated under different market conditions.

4.3. Splitting Swaps

Splitting swaps can also be categorized into two types: (1) Splitting over pools: divide the total input Δx_n across multiple pools and execute simultaneously. (2) Splitting over time: divide Δx_n into sequential swaps within a single pool. The corresponding optimization problems are referred to as optimal routing and optimal execution, respectively [6].

Splitting over pools both mitigates the quadratic effect of Δx_n and exploits curvature discrepancies across AMMs. For the same swapping pair, curvatures across AMMs are not necessarily identical, making it profitable to route swaps into pools with lower curvature. We illustrate the rationales of the two mechanisms by a simple model. Suppose two pools are in the identical state $(x_n, f(x_n))$, where $x_n \in \mathbb{R}_+$. For a swap of size $\Delta x_n \in \mathbb{R}_+$, the trader can either execute the entire swap in a single pool or split it evenly across the two pools. Comparing endogenous deviations, we obtain

$$2 \cdot \left(-\frac{1}{2} f''(x_n) \left(\frac{\Delta x_n}{2} \right)^2 \right) - \left(-\frac{1}{2} f''(x_n) (\Delta x_n)^2 \right) = \frac{1}{4} f''(x_n) (\Delta x_n)^2, n \in \mathbb{N}. \quad (10)$$

Similarly, comparing the realized outputs,

$$2(f(x_n) - f(x_n + \frac{\Delta x_n}{2})) - (f(x_n) - f(x_n + \Delta x_n)) \approx \frac{1}{4} f''(x_n) (\Delta x_n)^2, n \in \mathbb{N}. \quad (11)$$

When $f''(x_n) > 0$, i.e., the constant function is convex, splitting yields lower deviation and higher output. The gains are further amplified when pools differ ex ante. Note that if the constant function is linear, splitting yields no benefit.

To split optimally, a basic framework is to introduce $x_{n,i}$ as the reserve of X in AMM i before the n -th swap, and $\Delta x_{n,i}$ as the amount allocated to AMM i , such that $\Delta x_n = \sum_{i \in I} \Delta x_{n,i}$. It then formulates the following optimization problem:

$$\begin{aligned} \max \quad & \sum_{i \in I} (f_i(x_{n,i}) - f_i(x_{n,i} + \Delta x_{n,i})) \\ \text{s.t.} \quad & \sum_{i \in I} \Delta x_{n,i} = \Delta x_n, \\ & \Delta x_{n,i} \geq 0, \quad \forall i \in I. \\ \text{Variables:} \quad & \{\Delta x_{n,i}\}_{i \in I} \end{aligned} \quad (12)$$

The standard first-order condition for this convex program, as derived by [24–26], is

$$p_i(x_{n,i} + \Delta x_{n,i}) = p_j(x_{n,j} + \Delta x_{n,j}), \quad \forall i, j \in I. \quad (13)$$

In other words, for optimality, the post-swap spot prices across all AMMs must be equal. Based on this insight, Zhou et al. [26] propose a greedy routing strategy: the transaction volume is first allocated to the pool offering the most favorable price, and then it is distributed across multiple pools in a way that maintains price equalization.

The following works consider more general settings. Danos et al. [25] consider a multi-hop situation and thus introduce the concept of an exchange network. This is a directed multi-graph, where nodes are assets and edges are pools. They show that optimal routing can still be formulated as a convex optimization problem, and tractable subproblems can be derived to obtain efficient solutions. Angeris et al. [24] further generalize the framework by allowing for concave utility functions as the objective instead of directly maximizing output. They demonstrate that the resulting problem remains convex and efficiently solvable. They then consider a more practical setting with execution costs, showing that the routing problem becomes a mixed-integer convex program. The introduction of execution costs creates a trade-off between the gains from routing across pools and the additional costs incurred.

Another form of splitting is splitting over time. Its rationale stems from the quadratic effect of the swap size, which suggests that dividing a large swap into smaller pieces could be beneficial even within the same pool [6]. However, this intuition does not necessarily hold, as illustrated by the following model.

Consider an AMM state $(x_n, f(x_n))$ and a swap $(\Delta x_n, \Delta y_n)$. We compare executing the swap in a single transaction with splitting it into two sequential swaps $(\lambda \Delta x_n, \alpha \Delta y_n)$ and $((1 - \lambda) \Delta x_n, \beta \Delta y_n)$ where $\lambda, \alpha, \beta \in (0, 1)$. The difference in endogenous deviations is

$$\begin{aligned} & (\Delta y_n - p(x_n) \Delta x_n) - (\alpha \Delta y_n - p(x_n) \lambda \Delta x_n) - (\beta \Delta y_n - p(x_n + \lambda \Delta x_n) (1 - \lambda) \Delta x_n) \\ &= \Delta x_n (1 - \lambda) (p(x_n + \lambda \Delta x_n) - p(x_n)) + \underbrace{(\Delta y_n - \alpha \Delta y_n - \beta \Delta y_n)}_{=0} \\ &= \Delta x_n (1 - \lambda) (p(x_n + \lambda \Delta x_n) - p(x_n)), \end{aligned} \quad (14)$$

which is negative when the constant function is convex. This confirms that larger swaps lead to greater endogenous deviation. However, note that from Equation (14) we have $\Delta y_n - \alpha \Delta y_n - \beta \Delta y_n = 0$ (see Appendix B), implying that there is no difference in the realized output. It can be shown that the indifference generalizes to an arbitrary number of splits and remains unchanged. This property is referred to as path independence and is considered a desirable feature of AMMs [26,27].

Although splitting over time exhibits lower deviation, it does not change returns (this is because each sub-swap is evaluated at the updated spot price (e.g., $p(x_n + \lambda \Delta x_n)$), rather than the original price $p(x_n)$, resulting in smaller measured deviations). Therefore, once

fees are introduced, splitting over time becomes strictly suboptimal as it incurs additional transaction costs. This is formally shown for Uniswap V2 in Angeris et al. [28].

Nevertheless, when additional factors are taken into account, splitting over time may still be beneficial. Cartea et al. [6] incorporate stochastic price dynamics and arbitrage between DEX and external markets, and they develop an optimal execution strategy. They model the swap amount Δx_n as a stochastic process, which is controlled by the execution speed into the AMM. The problem is thus formulated as an optimal control problem that is solved via PDE methods and approximations. In this setting, price evolution plays a crucial role: waiting may lead to either more favorable or less favorable prices. Hence, splitting over time may improve outcomes. Recent work has explored price dynamics in AMMs [29].

5. Slippage from Outside: Exogenous Deviation

This section decomposes exogenous deviation and organizes the related literature into two themes: reducing net price change and mitigating MEV.

5.1. Decomposition

Exogenous deviation, defined as $(p(x_n) - p(x_o))\Delta x_n$, is driven by the net price change between o and n . Since x_n evolves from x_o through a sequence of intervening swaps $\{\Delta x_o, \Delta x_{o+1}, \dots, \Delta x_{o+n-1}\}$, exogenous deviation can also be viewed as the cumulative impact of these swaps.

This suggests that exogenous deviation is governed by two factors: the pricing function and the sequence of intervening swaps. The former determines the magnitude of price changes, while the latter drives these changes. The pricing function is relatively straightforward and has been largely discussed in Section 4.2, whereas the sequence of intervening swaps is more complex. We therefore first present a taxonomy of these swaps.

Following Adams et al. [1], intervening swaps can be broadly categorized into two classes: benign and adversarial. Adversarial swaps refer to reactive swaps that strategically respond to the target transaction, while benign swaps are those that are simply sequenced between the quote time and the execution time without strategic intent. Furthermore, benign transactions can be classified into within-block and cross-block components, depending on whether they are included in the same block as the target transaction or not.

Adams et al. [1] also provide an empirical landscape of the magnitudes of these components as well as their key determinants. Drawing on data from Uniswap V3, they show that cross-block slippage is generally smaller than within-block slippage. Benign slippage is primarily driven by latency, while adversarial slippage is largely influenced by swap size, as larger swaps are more likely to attract MEV strategies. Moreover, the relative importance of determinants varies across asset types.

Among these determinants, latency requires special consideration. While reducing latency at the individual level does mitigate exogenous deviation, a system-level reduction in latency (e.g., increasing TPS) does not necessarily have the same effect. This is because what ultimately matters is the swap's relative position in the execution ordering rather than its absolute delay. Although improving blockchain throughput may shorten latency in absolute terms, it does not necessarily improve relative ordering. In practice, priority is primarily determined by fee bidding [30] or network advantages.

Therefore, we focus only on the pricing function and the intervening swaps. Note that intervening swaps are primarily associated with front running, whereas not all MEV behaviors (e.g., back running) fall into this category. Nevertheless, following Adams et al. [1], we adopt the broader terminology of MEV to describe such effects.

5.2. Reducing Net Price Change

The net price change from submission to execution is still governed by the pricing function, and thus the analysis in Section 4.2 remains applicable. However, unlike endogenous deviation which is driven by the curvature $f''(x_n)$, exogenous deviation depends on the net price change $p(x_o) - p(x_n)$. This difference gives rise to additional approaches.

We begin by decomposing the total price change induced by a sequence of swaps. The overall price change can be rewritten and expanded as

$$p(x_n) - p(x_o) = \sum_{j=0}^{n-1} (p(x_{j+1}) - p(x_j)) = \sum_{j=0}^{n-1} (-f''(x_j)\Delta x_j), o, n \in \mathbb{N}, \quad (15)$$

where each term represents the impact of one swap generated on subsequent swaps. Compared to the endogenous deviation of this swap, namely $-\frac{1}{2}f''(x_j)(\Delta x_j)^2$, it can be shown that they share similar mechanisms though not the same quantity. In this sense, the smoothing-curvature approach, which is discussed in Section 4.2, still applies to exogenous deviation and will not be repeated here. Moreover, the construction of Equation (15) provides new ways beyond smoothing curvature. In fact, it suffices to require that the prices at the endpoints coincide regardless of the intermediate path. Along this direction, Dynamic AMMs [21,22] provide useful insights. Dynamic AMMs introduce additional parameters into the CP framework and adjust them to align the spot price with an external reference price before each swap. When the external price remains constant, the exogenous deviation is eliminated.

Despite this advantage, it comes with a fundamental cost. By anchoring prices to an external oracle, the AMM relinquishes its role in price discovery and instead becomes a passive follower of external markets.

5.3. Mitigating MEV

MEV mitigation approaches are designed to address adversarial swaps, but some of these approaches can also reduce deviation from within-block benign swaps. MEV mitigation has been extensively studied with a large body of work and survey papers available. We therefore provide only a brief overview of the main approaches, following the narrative of this paper. We refer readers to existing surveys and SoK papers for a comprehensive discussion [31–33].

Representative works are summarized in Table 2. We categorize existing MEV mitigation approaches into three classes based on how they intervene in the execution pipeline: ordering, blinding, and reallocation.

Table 2. Taxonomy of MEV mitigation mechanisms.

Approaches	Key Features	Representative Works
Ordering	Fair ordering	Wendy [34], Themis [35,36]
	Batch ordering	Frequent batch auctions [37]
	Pricing-based ordering	GSP [38], CLVR [39]
Blinding	Private mempool	Flashbots-Protect [40]
	Encrypted mempool	BlindPerm [41], TrX [42], ST ³ [43]
	Commit–reveal	Submarine Commitments [44], FairMM [45], P2DEX [46]
	Privacy-preserving execution	Manta [47]
Reallocation	User-side redistribution	B. Protocol [48], MEV-Share [40], RediSwap [49]
	Validator-side redistribution	MEV-Boost (PBS) [40], EIP-7732 (ePBS) [50]

Ordering mechanisms directly constrain the transaction sequencing rule. Fair ordering ranks transactions according to their arrival time, while batching executes a set of transactions simultaneously. Pricing-based ordering ranks transactions according to price-related objectives: for example, the Greedy Sequencing Rule (GSP) [38] guarantees that each swap receives at least a certain price bound, while Clever Look-ahead Volatility Reduction (CLVR) [39] determines the ordering that maximizes the overall price stability.

Blinking mechanisms hide transaction information, preventing adversaries from exploiting the observable order flow. Private mempools allow users to submit transactions through protected channels that are not publicly visible. Encrypted mempools use threshold or delay encryption to hide transaction contents from block producers until the ordering is finalized. Commit-reveal schemes require users to first publish a commitment and later reveal the transaction details. Finally, privacy-preserving execution leverages zero-knowledge proofs to keep transaction details permanently hidden, revealing only the validity of execution.

Reallocation mechanisms do not directly prevent MEV extraction but instead redistribute it among participants and can therefore mitigate its adverse effects indirectly. Some approaches allocate MEV to users (e.g., traders on DEXs in our setting), while others shift MEV to validators or block builders.

Since 2024, the emergence of cross-chain MEV has led to new mitigation approaches [51]. However, how these mechanisms relate to the setting considered in this paper remains an open question. For a comprehensive review of cross-chain MEV mitigation, see Mancino and Sevim [52].

6. Open Issues

Overall, the literature has studied slippage in great detail, achieving low and even zero slippage in certain settings. Nevertheless, several problems remain to be explored. Figure 4 illustrates the landscape of emerging directions in slippage.

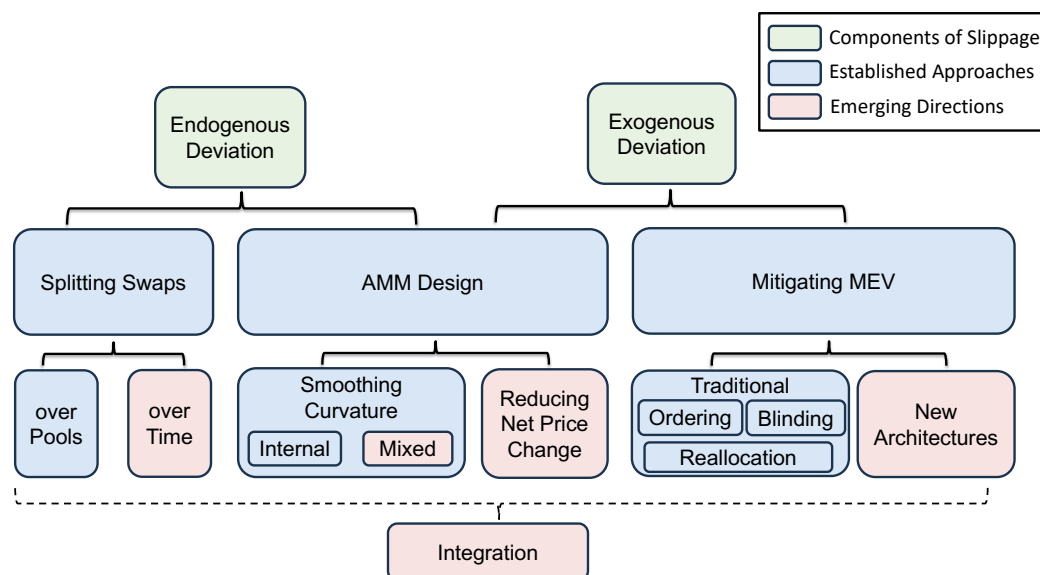


Figure 4. A structured view of slippage and its mitigation landscape. Colors indicate components, established approaches, and emerging directions.

6.1. AMM Design

The design of AMMs serves as a fundamental source of slippage. As discussed in our framework, existing approaches can be categorized into internal and mixed designs. While the internal approach has been extensively explored and deployed, the mixed approach

is a rapidly evolving area. By exploiting external information to adjust prices or modify pricing functions, mixed designs introduce new challenges while also opening up a rich space for further innovation and exploration.

6.2. *Splitting Swaps over Time*

The rationale for splitting swaps originates from multiple factors, including price differences across AMMs, the quadratic impact of trade size induced by the convexity of constant functions, and the role of stochastic price dynamics. Splitting swaps can be viewed both as a user-side execution strategy and as a mechanism embedded within AMMs [26], where the protocol itself executes trades along optimized paths.

Two main lines of research arise: splitting over pools and splitting over time. The former is primarily formulated as a convex optimization problem, while the latter is typically modeled as an optimal control problem. The latter requires a deeper understanding of AMM price dynamics, which remains relatively underexplored. Moreover, existing studies offer conflicting perspectives on its effectiveness.

6.3. *MEV and Slippage in Emerging Blockchain Architectures*

While a vast body of literature has studied MEV, relatively less attention has been paid to MEV under emerging blockchain architectures. Recent developments, particularly in cross-chain environments, introduce new challenges. Mitigating MEV in these settings has therefore emerged as a frontier and remains relatively underexplored.

More broadly, these developments call for rethinking AMM design under different system architectures. In settings such as sharded blockchains, the fragmentation of liquidity and asynchronous execution may fundamentally alter trading dynamics. As a result, not only MEV but also slippage may manifest in new forms. Understanding how slippage arises and can be mitigated under these architectures remains an open question.

6.4. *Integration Towards Practical Systems*

Our framework highlights that slippage consists of multiple components, which each have dedicated mechanisms for mitigation. Given the advances along these directions, a natural question is how to integrate state-of-the-art approaches in AMM design, execution strategies, and MEV mitigation into a unified system that delivers user experience comparable to centralized exchanges.

Such integration is nontrivial. From a practical perspective, the resulting system must remain simple and efficient, as it needs to support large-scale usage similar to today's centralized exchanges. This imposes constraints on computational complexity and system design, making it challenging to combine multiple sophisticated mechanisms in a lightweight manner. Therefore, a key open question is not only how to integrate these approaches but also how to do so in a simple and scalable way.

Moreover, such integration may extend beyond the AMM paradigm itself. While AMMs are currently the dominant realization of decentralized swapping, they are not the only possible design. Recent efforts have explored alternative paradigms, such as intent-based markets [53], where users specify desired outcomes and delegate execution to solvers, who ensure that trades are executed in accordance with user-specified constraints. Some designs under this paradigm not only integrate AMM design, execution strategies, and MEV mitigation but also enable practical, system-level implementations. A representative example is CoW Protocol [54].

7. Conclusions

Slippage links AMMs and MEV, two important areas in DeFi and blockchain systems, and is also one of the most user-relevant aspects of these domains. Improving slippage is therefore not only of theoretical importance but also highly relevant to industry practice.

By providing a unified framework, we systematically organize and position the existing literature on slippage. This perspective not only clarifies the current state of the field but also reveals the landscape of research progress, highlighting both well-established directions and emerging frontiers. These insights point to several promising avenues for future work.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/blockchains4030008/s1>, File S1: Figure2_data; File S2: Figure2_plotting.

Author Contributions: Theoretical Analysis, investigation, visualization, and writing—original draft, X.S.; writing—review and editing, J.Z., J.Y. and H.H.; supervision, H.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data supporting this paper are obtained from public blockchain networks via RPC queries. As blockchain data are inherently public and permissionless, all raw data can be independently retrieved by querying the relevant smart contracts and block ranges. The authors also provide related codes. The original data presented in the study are openly available in <https://www.alchemy.com/>.

Acknowledgments: During the preparation of this manuscript, the authors used generative AI tools (specifically, ChatGPT-5.5) only to improve language and readability. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DeFi	Decentralized Finance
DEX	Decentralized Exchanges
AMM	Automated Market Maker
LP	Liquidity Provider
CP	Constant Product
CS	Constant Sum
TPS	Transactions Per Second
PDE	Partial Differential Equation
MEV	Miner/Maximal Extractable Value
(e)PBS	(enshrined) Proposer–Builder Separation

Appendix A. Derivations of Table 1

Table A1. Constant functions (standard notation) of representative DEXs.

Representative DEX	Constant Function (Standard Notation)
Uniswap V2	$xy = k$
Uniswap V3	$(x + \frac{L}{\sqrt{p_b}})(y + L\sqrt{p_a}) = L^2$
Balancer V1	$V = \prod_t B_t^{W_t}$
Balancer V2 (Linear Pool)	$I = \sum_t (B_t * R_t)$
Curve V1	$An^n \sum x_i + D = ADn^n + \frac{D^{n+1}}{n^n \prod x_i}$

Appendix A.1. Uniswap V2

For Uniswap V2,

$$xy = k \Rightarrow f(x) = \frac{k}{x} \Rightarrow f''(x) = \frac{2k}{x^3}.$$

Appendix A.2. Uniswap V3

For Uniswap V3, denote L^2 by k ,

$$\left(x + \frac{\sqrt{k}}{\sqrt{p_b}}\right)(y + \sqrt{kp_a}) = k \Rightarrow f(x) = \frac{k}{x + \frac{\sqrt{k}}{\sqrt{p_b}}} - \sqrt{kp_a} \Rightarrow f''(x) = \frac{2k}{\left(x + \frac{\sqrt{k}}{\sqrt{p_b}}\right)^3}.$$

Appendix A.3. Balancer V1

For Balancer V1, denote the invariant V by k , the balances B_1 and B_2 by x and y , respectively, and the remaining balances by x_i for $i = 3, \dots, n$. Let the corresponding weights be w_x , w_y , and w_i . The constant function becomes

$$x^{w_x} y^{w_y} \prod_{i=3}^n x_i^{w_i} = k.$$

Solving for y ,

$$f(x) = \left(\frac{k}{x^{w_x} \prod_{i=3}^n x_i^{w_i}} \right)^{\frac{1}{w_y}}.$$

Let

$$A = \left(\frac{k}{\prod_{i=3}^n x_i^{w_i}} \right)^{\frac{1}{w_y}},$$

then

$$f(x) = Ax^{-\frac{w_x}{w_y}}.$$

Therefore,

$$f'(x) = A \left(-\frac{w_x}{w_y} \right) x^{-\frac{w_x}{w_y} - 1},$$

and

$$f''(x) = A \left(-\frac{w_x}{w_y} \right) \left(-\frac{w_x}{w_y} - 1 \right) x^{-\frac{w_x}{w_y} - 2}.$$

Substituting $y = Ax^{-\frac{w_x}{w_y}}$ yields

$$f''(x) = \frac{w_x}{w_y} \left(\frac{w_x}{w_y} + 1 \right) \frac{y}{x^2}.$$

Appendix A.4. Balancer V2 (Linear Pool)

For Balancer V2 Linear Pools, the deployed pools are typically two-asset pools, so we let $t = 2$. Denote the invariant I by k and the two asset balances by x and y . Normalizing one exchange rate to 1 and denoting the other by R_{adj} , the invariant can be written as

$$x + R_{\text{adj}}y = k.$$

Hence,

$$f(x) = \frac{k - x}{R_{\text{adj}}},$$

which implies

$$f''(x) = 0.$$

Here, R_{adj} represents an externally adjusted exchange rate. When the relative value between the two assets changes, R_{adj} is updated accordingly, allowing the pool to maintain a near-linear pricing function.

Appendix A.5. Curve V1

Denote the invariant D by k and the balances x_1 and x_2 by x and y ; then, the constant function becomes

$$An^n(x + y + \sum_{i=3}^n x_i) + k = Akn^n + \frac{k^{n+1}}{n^n xy \prod_{i=3}^n x_i}$$

Define

$$C_1 = An^n, \quad C_2 = \frac{k^{n+1}}{n^n \prod_{i=3}^n x_i}.$$

The invariant can be rewritten as the implicit equation

$$F(x, y) = C_1 \left(x + y + \sum_{i=3}^n x_i \right) + k - Akn^n - \frac{C_2}{xy} = 0.$$

Since y is implicitly determined by x , we compute the curvature using implicit differentiation. The first derivative is

$$f'(x) = \frac{dy}{dx} = -\frac{F_x}{F_y}.$$

Differentiating once more yields

$$f''(x) = -\frac{F_{xx}F_y^2 - 2F_{xy}F_xF_y + F_{yy}F_x^2}{F_y^3}.$$

For the Curve V1 invariant, the required partial derivatives are

$$F_x = C_1 + \frac{C_2}{x^2y}, \quad F_y = C_1 + \frac{C_2}{xy^2},$$

and

$$F_{xx} = -\frac{2C_2}{x^3y}, \quad F_{xy} = -\frac{C_2}{x^2y^2}, \quad F_{yy} = -\frac{2C_2}{xy^3}.$$

The curvature formula is thus

$$f''(x) = \frac{\frac{2C_2}{x^3y} \left(C_1 + \frac{C_2}{xy^2} \right)^2 - \frac{2C_2}{x^2y^2} \left(C_1 + \frac{C_2}{x^2y} \right) \left(C_1 + \frac{C_2}{xy^2} \right) + \frac{2C_2}{xy^3} \left(C_1 + \frac{C_2}{x^2y} \right)^2}{\left(C_1 + \frac{C_2}{xy^2} \right)^3}.$$

The curvature diverges to $+\infty$ as $x \rightarrow 0^+$. In the symmetric case $x = y$, the expression simplifies to

$$f''(x) = \frac{2C_2}{x(C_1x^3 + C_2)}.$$

Appendix B. Path Independence

Note that Δy_n denotes the output generated by an input Δx_n at state $(x_n, f(x_n))$. Similarly, $\alpha \Delta y_n$ denotes the output generated by an input $\lambda \Delta x_n$ at state $(x_n, f(x_n))$, while $\beta \Delta y_n$ denotes the output generated by an input $(1 - \lambda) \Delta x_n$ at state $(x_n + \lambda \Delta x_n, f(x_n + \lambda \Delta x_n))$. Therefore,

$$\begin{aligned}\alpha \Delta y_n + \beta \Delta y_n &= [f(x_n) - f(x_n + \lambda \Delta x_n)] \\ &\quad + [f(x_n + \lambda \Delta x_n) - f(x_n + \Delta x_n)] \\ &= f(x_n) - f(x_n + \Delta x_n) \\ &= \Delta y_n,\end{aligned}$$

where the intermediate terms cancel out through a telescoping sum.

References

1. Adams, A.; Chan, B.Y.; Markovich, S.; Wan, X. Don't Let MEV Slip: The Costs of Swapping on the Uniswap Protocol. In *Proceedings of the Financial Cryptography and Data Security: 28th International Conference (FC 2024), Willemstad, Curaçao, 4–8 March 2024*; Springer: Cham, Switzerland, 2025; pp. 172–191.
2. Xu, J.; Paruch, K.; Cousaert, S.; Feng, Y. SoK: Decentralized Exchanges (DEX) with Automated Market Maker (AMM) Protocols. *ACM Comput. Surv.* **2023**, *55*, 238. [CrossRef]
3. Adams, A.; Wan, X. Just-In-Time Liquidity on the Uniswap Protocol. Available online: <https://blog.uniswap.org/jit-liquidity> (accessed on 20 June 2026).
4. DefiLlama. DeFi Dashboard. Available online: <https://defillama.com/> (accessed on 4 May 2026).
5. Asef, M.A.; Bamakan, S.M.H. From $x*y=k$ to Uniswap Hooks: A Comparative Review of Decentralized Exchanges (DEX). *arXiv* **2024**, arXiv:2410.10162.
6. Cartea, Á.; Drissi, F.; Monga, M. Decentralised finance and automated market making: Execution and speculation. *J. Econ. Dyn. Control* **2025**, *177*, 105–134. [CrossRef]
7. Daian, P.; Goldfeder, S.; Kell, T.; Li, Y.; Zhao, X.; Bentov, I.; Breidenbach, L.; Juels, A. Flash Boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In *Proceedings of the 2020 IEEE Symposium on Security and Privacy (SP 2020)*, San Francisco, CA, USA, 18–21 May 2020; pp. 910–927.
8. Adams, H.; Zinsmeister, N.; Salem, M.; Keefer, R.; Robinson, D. Uniswap V3 Core. Available online: <https://app.uniswap.org/whitepaper-v3.pdf> (accessed on 12 April 2026).
9. Angeris, G.; Chitra, T. Improved Price Oracles: Constant Function Market Makers. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies (AFT'20)*, New York, NY, USA, 21–23 October, 2020; pp. 80–91.
10. Engel, D.; Herlihy, M. Composing Networks of Automated Market Makers. In *Proceedings of the 3rd ACM Conference on Advances in Financial Technologies (AFT)*, Arlington, VA, USA, 26–28 September 2021; pp. 15–28.
11. Engel, D.; Herlihy, M. Presentation and Publication: Loss and Slippage in Networks of Automated Market Makers. In *Proceedings of the 3rd International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2021)*, New York, NY, USA, 18–19 November 2021; pp. 13:1–13:23.
12. Uniswap Labs. Swaps—Uniswap Docs. Available online: <https://docs.uniswap.org/concepts/protocol/swaps> (accessed on 5 April 2026).
13. Millionis, J.; Moallemi, C.; Roughgarden, T.; Zhang, A. Automated Market Making and Loss-Versus-Rebalancing. *arXiv* **2024**, arXiv:2208.06046.
14. Adams, H.; Zinsmeister, N.; Robinson, D. Uniswap V2 Core. Available online: <https://app.uniswap.org/whitepaper.pdf> (accessed on 12 April 2026).
15. Martinelli, F.; Mushegian, N. Balancer: A Non-Custodial Portfolio Manager, Liquidity Provider, and Price Sensor. Available online: <https://docs.balancer.fi/whitepaper.pdf> (accessed on 12 April 2026).
16. Linear Math. Available online: <https://docs-v2.balancer.fi/reference/math/linear-math.html> (accessed on 12 April 2026).
17. Egorov, M. StableSwap—Efficient Mechanism for Stablecoin Liquidity. Available online: https://docs.curve.finance/pdf/whitepapers/whitepaper_stableswap.pdf (accessed on 12 April 2026).
18. Egorov, M. Automatic Market-Making with Dynamic Peg. Available online: https://docs.curve.finance/pdf/whitepapers/whitepaper_cryptoswap.pdf (accessed on 12 April 2026).
19. DODO Team. DODO—A Next-Generation On-Chain Liquidity Provider Powered by Pro-Active Market Maker Algorithm. Available online: <https://github.com/DODOEX/docs/blob/master/docs/whitepaper.md> (accessed on 12 April 2026).

20. Lim, T. Predictive crypto-asset automated market maker architecture for decentralized finance using deep reinforcement learning. *Financ. Innov.* **2024**, *10*, 144–173. [CrossRef]
21. Jiang, S.; Chen, J.; Li, F.; Geng, H.; Chi, H. DCAMM: Dynamic Curve-Based Automated Market Maker. In Proceedings of the IEEE Global Communications Conference (GLOBECOM), Kuala Lumpur, Malaysia, 4–8 December 2023; pp. 4491–4496.
22. Krishnamachari, B.; Feng, Q.; Grippo, E. Dynamic Automated Market Makers for Decentralized Cryptocurrency Exchange. In Proceedings of the IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Sydney, Australia, 3–6 May 2021; pp. 1–2.
23. Uniswap Labs. Hooks–Uniswap Developers. Available online: <https://developers.uniswap.org/docs/protocols/v4/concepts/hooks> (accessed on 20 June 2026).
24. Angeris, G.; Chitra, T.; Evans, A.; Boyd, S. Optimal routing for constant function market makers. *arXiv* **2022**, arXiv:2204.05238.
25. Danos, V.; El Khallofi, H.; Prat, J. Global order routing on exchange networks. In Proceedings of the International Conference on Financial Cryptography and Data Security, Virtual Event, 1–5 March 2021; pp. 207–226.
26. Zhou, L.; Qin, K.; Gervais, A. A2MM: Mitigating Frontrunning, Transaction Reordering and Consensus Instability in Decentralized Exchanges. *arXiv* **2021**, arXiv:2106.07371.
27. Buterin, V. On Path Independence. Available online: <https://vitalik.eth.limo/general/2017/06/22/marketmakers.html> (accessed on 18 April 2026).
28. Angeris, G.; Kao, H.; Chiang, R.; Noyes, C.; Chitra, T. An analysis of Uniswap market. *arXiv* **2021**, arXiv:1911.03380. Available online: <https://arxiv.org/abs/1911.03380> (accessed on 13 April 2026).
29. Maymin, P. Option Pricing on Automated Market Maker Tokens. *arXiv* **2026**, arXiv:2603.29763.
30. Buterin, V.; Conner, E.; Dudley, R.; Slipper, M.; Norden, I.; Bakhta, A. EIP-1559: Fee Market Change for ETH 1.0 Chain. Available online: <https://eips.ethereum.org/EIPS/eip-1559> (accessed on 18 April 2026).
31. Heimbach, L.; Wattenhofer, R. SoK: Preventing transaction reordering manipulations in decentralized finance. In Proceedings of the 4th ACM Conference on Advances in Financial Technologies (AFT '22), Cambridge, MA, USA, 19–21 September 2022; pp. 47–60.
32. Materwala, H.; Naik, S.M.; Taha, A.; Abed, T.A.; Svetinovic, D. Maximal extractable value in decentralized finance: Taxonomy, detection, and mitigation. *IEEE Trans. Serv. Comput.* **2025**, *18*, 4386–4407. [CrossRef]
33. Yang, S.; Zhang, F.; Huang, K.; Chen, X.; Yang, Y.; Zhu, F. SoK: MEV Countermeasures. In Proceedings of the ACM Workshop on Decentralized Finance and Security (DeFi '24), co-located with the ACM SIGSAC Conference on Computer and Communications Security (CCS '24), Salt Lake City, UT, USA, 14–18 October 2024; pp. 21–30.
34. Kursawe, K. Wendy, the good little fairness widget: Achieving order fairness for blockchains. In Proceedings of the 2nd ACM Conference on Advances in Financial Technologies (AFT '20), New York, NY, USA, 21–23 October 2020; pp. 25–36.
35. Kelkar, M.; Deb, S.; Kannan, S. Order-fair consensus in the permissionless setting. In Proceedings of the 9th ACM Asia Public-Key Cryptography Workshop (APKC '22), Nagasaki, Japan, 30 May 2022; pp. 3–14.
36. Kelkar, M.; Deb, S.; Long, S.; Juels, A.; Kannan, S. Themis: Fast, strong order-fairness in Byzantine consensus. In Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), Copenhagen, Denmark, 26–30 November 2023; pp. 475–489.
37. McMenamin, C.; Daza, V.; Fitzi, M.; O'Donoghue, P. Fairtradex: A decentralised exchange preventing value extraction. In Proceedings of the ACM CCS Workshop on Decentralized Finance and Security (DeFi '22), New York, NY, USA, 6 May 2022; pp. 39–46.
38. Ferreira, M.V.X.; Parkes, D.C. Credible decentralized exchange design via verifiable sequencing rules. In Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC), Orlando, FL, USA, 20–23 June 2023; pp. 723–736.
39. McLaughlin, R.; Chemaya, N.; Liu, D.; Malkhi, D. CLVR Ordering of Transaction on AMMs. *arXiv* **2025**, arXiv:2408.02634.
40. Flashbots. Flashbots Documentation. Available online: <https://docs.flashbots.net/> (accessed on 18 April 2026).
41. Kavousi, A.; Le, D.V.; Jovanovic, P.; Danezis, G. BlindPerm: Efficient MEV Mitigation with an Encrypted Mempool and Permutation. *IACR Cryptology ePrint Archive*, Paper 2023/1061. 2023. Available online: <https://eprint.iacr.org/2023/1061> (accessed on 20 April 2026).
42. Fernando, R.; Policharla, G.-V.; Tonkikh, A.; Xiang, Z. TrX: Encrypted Mempools in High Performance BFT Protocols. *IACR Cryptology ePrint Archive*, Paper 2025/2032. 2025. Available online: <https://eprint.iacr.org/2025/2032> (accessed on 20 April 2026).
43. Chakrabarti, A.; Maitra, M.; Mondal, A.; Sehgal, K. Silent threshold traitor tracing & enhancing mempool privacy. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS '25), Taipei, Taiwan, 13–17 October 2025; pp. 1769–1783.
44. Breidenbach, L.; Daian, P.; Tramèr, F.; Juels, A. Enter the hydra: Towards principled bug bounties and exploit-resistant smart contracts. In Proceedings of the 27th USENIX Security Symposium (USENIX Security '18), Baltimore, MD, USA, 15–17 August 2018; pp. 1335–1352.
45. Ciampi, M.; Ishaq, M.; Magdon-Ismael, M.; Ostrovsky, R.; Zikas, V. FairMM: A Fast and Frontrunning-Resistant Crypto Market-Maker. *IACR Cryptology ePrint Archive*, Paper 2021/609. 2021. Available online: <https://eprint.iacr.org/2021/609> (accessed on 12 April 2026).
46. Baum, C.; David, B.; Frederiksen, T.K. P2DEX: Privacy-preserving decentralized cryptocurrency exchange. In Proceedings of the 19th International Conference on Applied Cryptography and Network Security (ACNS 2021), Kamakura, Japan, 21–24 June 2021; pp. 163–194.

47. Chu, S.; Xia, Y.; Zhang, Z. Manta: A Plug and Play Private DeFi Stack. *IACR Cryptology ePrint Archive*, Paper 2021/743. 2021. Available online: <https://eprint.iacr.org/2021/743> (accessed on 18 April 2026).
48. B.Protocol. B.Protocol Documentation. Available online: <https://docs.bprotocol.org/> (accessed on 18 April 2026).
49. Zhang, M.; Yang, S.; Zhang, F. RediSwap: MEV Redistribution Mechanism for CFMMs. *arXiv* **2024**, arXiv:2410.18434.
50. D’Amato, F.; Flaig, N.; Monnot, B.; Neuder, M.; Potuz, J.; Traglia, J.; Tsao, T. EIP-7732: Enshrined Proposer-Builder Separation. Available online: <https://eips.ethereum.org/EIPS/eip-7732> (accessed on 18 April 2026).
51. Torres, C.F.; Mamuti, S.; Weintraub, B.; Nita-Rotaru, C.; Shinde, S. Rolling in the shadows: Analyzing the extraction of MEV across layer-2 rollups. *arXiv* **2024**, arXiv:2405.00138.
52. Mancino, D.; Sevim, H. SoK: The Evolution of Maximal Extractable Value, From Miners to Cross-Chain *arXiv* **2026**, arXiv:2603.07716.
53. Chitra, T.; Kulkarni, K.; Pai, M.; Diamandis, T. An Analysis of Intent-Based Markets *arXiv* **2024**, arXiv:2403.02525.
54. Nag, A.; Prabhakar, M.; Behl, T. A Coincidence of Wants Mechanism for Swap Trade Execution in Decentralized Exchanges. *arXiv* **2025**, arXiv:2507.10149.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.