algorithms

MDPI

Article

ESS-LP: An Effective Slippage Scheme Based on Liquidity Pools for Data Trading

Huayou Si [1], Mengyang Li [1], Yuanyuan Qi [1,*], Neal N. Xiong [2,*], Wei Chen [1], Loc Nguyen The [3] and Shichong Wang [1]

[1] The School of Information Engineering, China Jiliang University, Hangzhou 310018, China; sshily@cjlu.edu.cn (H.S.); p23030854081@cjlu.edu.cn (M.L.); wei.chen@cjlu.edu.cn (W.C.); s23030812006@cjlu.edu.cn (S.W.)
[2] The Department of Computer Science, Southern New Hampshire University, Manchester, NH 03106, USA
[3] The School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi 100000, Vietnam; locnt@hnue.edu.vn
* Correspondence: qiyuanyuan@cjlu.edu.cn (Y.Q.); xiongnaixue@gmail.com (N.N.X.)

Abstract

This paper proposes a decentralized data trading approach based on the Automated Market Maker (AMM) mechanism, aiming to address the bottlenecks in data trading efficiency and fairness via the collaborative innovation of market-oriented pricing mechanisms and automated trading processes. We focus on constructing an automatic pricing and matching mechanism based on liquidity pools. Subsequently, mathematical modeling and simulations reveal the slippage generation mechanism in data liquidity pools under trading shocks and imbalances. To address these issues, a novel dual slippage optimization mechanism integrating dynamic trade splitting and alternating order sorting is proposed, which decomposes orders into sub-orders and reorganizes their timing, establishing a dynamic equilibrium model. Experiments show that the method reduces the average slippage amplitude from 2.1% to 0.5%, representing a 76.2% reduction, and significantly enhances price stability and market efficiency. This research explores the mechanism of applying AMM to data asset trading and mitigates the limitations of AMM in this scenario, providing a theoretical and empirical foundation for building low-cost, high-fairness data trading systems through mechanism innovation and technical optimization.

Keywords: data trading; automatic market maker; slippage optimization; trading systems

Check for updates

Academic Editor: Sally McClean

Received: 21 April 2026
Revised: 30 May 2026
Accepted: 5 June 2026
Published: 7 June 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) license.

1. Introduction

In recent years, the digital economy has grown at an unprecedented scale and speed. It has become a driving force in reshaping the global economic structure and competitive landscape. Data has gradually transformed from a supportive resource into a core resource driving production and innovation. The efficient circulation and value realization of data increasingly rely on standardized and orderly trading mechanisms, and the establishment of a data asset market allocation system has become a critical pathway for advancing the digital economy. To facilitate data circulation, various data trading platforms have been successively established both domestically and internationally [1], such as the domestic Guiyang Global Big Data Trading Platform, the Central China Big Data Trading Platform, and the Shanghai Data Trading Platform, etc., alongside the international data trading platforms such as Factual, BDEX, and Data Plaza [2]. These platforms aim to bridge data suppliers and customers. While data exchanges promote cross-hierarchical data

circulation across society, their centralized architectures frequently manifest single point of failure vulnerabilities. Persistent challenges include security risks from centralized data storage, suboptimal pricing mechanisms, and inefficient matching capabilities, collectively rendering these platforms inadequate to satisfy the flexible, real-time, and high-frequency trading requirements characteristic of data asset markets.

Decentralized digital currency trading platforms based on the Automated Market Maker (AMM) model, such as Bancor [3], Uniswap [4], and Curve [5], have been widely applied in blockchain financial systems recently. These platforms utilize smart contracts on the blockchain to enable liquidity pool trading, allowing users to complete trades in a decentralized, trustless way that partially resolves issues inherent in traditional exchanges. The AMM mechanism relies on liquidity pools and predefined mathematical functions for dynamic pricing, facilitating decentralized trading without order matching, which provides a novel idea for overcoming the liquidity bottleneck and inefficient matching in centralized platforms. Although the mechanism has been validated in homogeneous financial assets, its applicability to data asset trading remains under-explored. Few existing studies have introduced the AMM mechanism into data trading, and the non-homogeneity, non-standardization, and sparsity of data assets pose new modeling and trading challenges. However, the trading slippage problem is particularly prominent under the liquidity pool-based trading mechanism. Slippage typically denotes the discrepancy between executed and expected prices [6], and is influenced by factors such as trade volume, market depth, and supply-demand dynamics. Data assets exhibit strong time-sensitivity and uniqueness [7], with slippage amplification frequently occurring during block trades or asymmetric trades. This phenomenon significantly increases trading costs and undermines market fairness.

Existing research on slippage control mainly focuses on financial trading scenarios, typically employing mitigation strategies such as increasing market depth, limiting trading volume, optimizing trading strategies for dynamic curve market makers, or introducing price oracles, such as those used in Bancor and Uniswap. However, these methods mainly rely on the structure of high-frequency and highly symmetric financial assets, making it difficult to adapt them to the characteristics of data trading. The new mechanistic causes underlying slippage in data trading scenarios still lack systematic research and effective solutions. Within the domain of data trading platforms, some studies have attempted to optimize price discovery and data quality assessment mechanisms, such as automatically negotiating pricing through smart contracts, supply-demand matching based on data quality priorities [8], fair exchange agreements incentivizing honest behavior [9] and privacy protection based on zero-knowledge proofs [10]; however, a comprehensive slippage analysis framework and control mechanism remain unrealized. Crucially, the precise optimization of trade execution pathways in response to real-time platform liquidity conditions, particularly within dynamic trading environments, still constitutes an unexplored research frontier.

Overall, existing studies have provided important foundations for decentralized data trading and AMM-based automated trading. Blockchain-based data trading mechanisms have improved transaction transparency, privacy protection, incentive compatibility, and fair exchange through smart contracts, data quality-aware matching, and cryptographic protocols [8–10]. Meanwhile, AMM-based liquidity pool mechanisms have demonstrated strong advantages in automatic pricing and continuous liquidity provision in decentralized financial markets [3–5]. However, these two research streams have not yet been sufficiently integrated. Existing data trading studies generally lack continuous market-oriented pricing and dynamic execution optimization, while existing AMM-based slippage control methods are mainly designed for standardized financial assets and are difficult to directly

adapt to data asset trading scenarios characterized by non-standardized rights, sparse transactions, limited liquidity, and supply-demand disequilibrium. Therefore, the central problem addressed in this study is how to adapt the AMM mechanism to multi-source homogeneous data asset trading while effectively controlling slippage caused by block trades and transaction imbalance.

To tackle these challenges, this paper proposes a decentralized data trading approach based on the AMM mechanism for multi-source homogeneous data asset trading scenarios, aiming to overcome the bottlenecks in data trading efficiency and fairness through the collaborative innovation of market-oriented pricing mechanisms and automated trading processes. In response to the slippage issues, we propose a novel dual slippage optimization mechanism integrating dynamic trade splitting and alternating order sorting to effectively alleviate the impacts of block trades and matching disequilibrium. The following are the main contributions of this paper:

- Proposes a decentralized data trading approach based on the AMM mechanism, aiming to overcome the bottlenecks in data trading efficiency and fairness through the collaborative innovation of market-oriented pricing mechanisms and automated trading processes. This establishes a novel decentralized trading paradigm for data assets, extending the application boundaries of key Decentralized Finance (DeFi) mechanisms to non-financial trading scenarios.
- Innovatively constructs an automated pricing and trading matching mechanism based on a liquidity pool. Through mathematical modeling and simulation experiments, it quantitatively analyzes the generation mechanism of slippage in data liquidity pools when coping with transaction shocks and buy-sell disequilibrium. The findings reveal that liquidity shocks induced by block trades and temporal mismatches arising from buy-sell disequilibrium constitute the primary root causes of pronounced slippage volatility in data trading.
- Innovatively proposes a dual slippage optimization mechanism integrating dynamic trade splitting and alternating order sorting, featuring a liquidity pool state-aware dynamic trade splitting and alternating order sorting execution engine. This achieves stable control of slippage in block trade scenarios.

The rest of the paper is organized as follows: Section 2 introduces and discusses related work in the data trading field. In Section 3, we propose a data trading model based on liquidity pools and analyze the factors influencing trading slippage. Building upon this analysis, we introduce our slippage optimization methodology for liquidity pool-based data trading. In Section 4, we evaluate the efficacy of the proposed slippage optimization approach through comprehensive experiments. Finally, Section 5 presents the principal conclusions and research contributions.

2. Related Work

2.1. Development of Data Trading Market

Currently, most domestic and international data trading platforms adopt a centralized architecture, facilitating trades between data buyers and sellers. While these platforms improve the efficiency of data resource aggregation and sharing, they commonly rely on a high degree of trust in the platform, are plagued by structural deficiencies such as the absence of flexible pricing mechanisms and low negotiation efficiency, and fail to meet the market-oriented allocation needs of data assets. For example, China's Zhongguancun Data Sea Big Data Trading Platform and the international platform RapidAPI [11] both utilize service brokerage or API interface authorization to avoid direct data storage issues, yet they still face fundamental challenges like a lack of pricing mechanism flexibility. Some platforms offer data packaging and authorized distribution services, standardizing

raw datasets into structured commodities with clear usage rights, allowing buyers to acquire data packages under specific license terms [12]. Other platforms act solely as information intermediaries, facilitating discovery and trade matching between data buyers and sellers without directly storing or processing data. Subscription-based trading models are also prevalent in data markets, where buyers continuously receive updated data streams through regular payments. Crowdsourcing market models aggregate user-contributed data assets and sell them centrally to interested enterprise clients [13,14]. Despite the variations in their implementation, these trading models fundamentally depend on a centralized architecture. Consequently, they are inherently susceptible to a range of issues, including data leakage, single points of failure, opaque trading processes, and a high reliance on trust among participants. These limitations have prompted researchers to explore more open and decentralized data trading mechanisms.

Recently, the academic community has extensively explored decentralized data trading. The emergence of blockchain decentralized technology has effectively addressed issues such as data theft by trading platforms and single points of failure [15–17]. For example, Li et al. [18] applied blockchain technology to virtual power plant trades, verifying the feasibility of decentralized dispatch and smart contracts. Dai et al. [19] proposed a data secure trading method based on blockchain, which prevents both the trading platform and data buyers from obtaining the original data of sellers, only allowing access to the analysis results of the required data, thereby ensuring the security of the original data. In [8], Hu et al. designed a blockchain-based big data trading system architecture called DataTBC. In DataTBC, data requestors publish data demands via contracts, and data providers can respond to these demands. During the data matching process completed by the contract, high-quality data has a higher priority when participating in data demands. Furthermore, trading parties complete the price negotiation process through smart contracts, and data rewards are automatically distributed according to the negotiated price. In [9], Jiang et al. proposed a blockchain-based data trading system called BDTS, implementing a fair exchange protocol where benevolent behavior of traders is rewarded, and dishonest behavior is punished. This encourages rational traders to faithfully participate in data trading activities to obtain maximum incentive benefits. In this research, Jiang et al. [9] also analyzed the strategies of sellers, buyers, and dealers in trades, pointing out that if every participant is honest and rational, the trading market can reach a Nash equilibrium, ensuring the maximization of traders' interests. In [10], Xue et al. proposed a fair and accurate blockchain-based medical data trading scheme. Using zero-knowledge proofs and encryption techniques, data sellers can sell partial data without disclosing personal information, and data buyers can only obtain the necessary data, thus ensuring fairness and privacy protection in trades.

The aforementioned methods primarily focus on ensuring fairness, protecting privacy, and designing incentive mechanisms in the process of data trading. These methods alleviate problems such as data leakage, information asymmetry, and dishonest trading behavior to some extent. However, these methods generally suffer from a lack of practicality, relying on fixed rules or static negotiation processes that fail to meet the market-oriented allocation needs of data assets. Especially in scenarios with block trades, high frequencies, or strong heterogeneity among participants, they often exhibit limitations such as low matching efficiency and long response delays. At the same time, current mechanisms largely lack dynamic pricing capabilities based on market conditions, making them unable to respond sensitively to real-time supply and demand relationships, which can lead to pricing deviations and market disequilibrium. In this context, there is an urgent need to introduce decentralized trading mechanisms with automatic pricing and continuous matching capabilities to enhance the flexibility and efficiency of data trading systems. The

AMM model, due to its mechanistic characteristics, offers a new possible path to address these issues.

2.2. AMM Mechanism

The market maker mechanism originated in the stock and foreign exchange markets, aiming to maintain market liquidity by providing buy and sell quotes. As trading scales expanded and markets became high-frequency, traditional manual quoting methods gradually failed to meet efficiency demands, leading to the emergence of algorithmic market making. AMMs [20,21], as representative mechanisms, were initially used in prediction markets to mitigate pricing distortions and insufficient liquidity caused by irrational participants. The core features of AMMs lie in their ability to achieve continuous liquidity provision and automatic dynamic pricing without order book matching through predefined pricing functions and liquidity pool mechanisms, thus significantly simplifying the trading execution process. A typical example is the AMM model built on the Logarithmic Market Scoring Rule [22], which not only controls maximum losses but also maintains continuous trading capability under any market conditions, demonstrating high stability and flexibility.

In recent years, with the popularity of cryptocurrencies and the growing demand for decentralized exchanges, AMMs have again become more important as a viable alternative to centralized order books. In the AMM model, liquidity providers supply single or multiple types of tokens to designated liquidity pools, and traders trade based on these token pools rather than relying on traditional order matching. The liquidity pools in these AMM models track predefined mathematical functions, which dynamically adjust token prices based on the supply and demand of assets in the pool, thereby determining the provision of various token types to traders in exchange for a certain quantity of another token. Compared to traditional order book models, curve-based AMMs provide a continuous supply of liquidity [23]. Additionally, depending on the mathematical function used, trading prices can better reflect market supply and demand [24]. To meet various trading demands and asset characteristics, various AMM models have been developed in the industry. As shown in [3], Bancor implemented an AMM-based DEX called Bonding Curve, which provides continuous liquidity for buying and selling individual tokens. In a curve-based AMM, any two currencies can be traded directly with each other. The common curve is a mathematical function of the token supply used to calculate the token price, satisfying the law of supply and demand and ensuring unlimited liquidity. Drawing on solutions from the prediction market model, Buterin [25] proposed an AMM for decentralized exchanges, with the constant product curve presented in Equation (1):

$$x \times y = k, \tag{1}$$

where x and y represent the total supply of the two tokens in the liquidity pool, and k is a constant. Subsequently, as shown in [4], Adams et al. created Uniswap based on this. Since then, a series of constant function AMM variants have emerged. For example, as presented in [5], Curve combines constant sum and constant product AMMs for stablecoin exchange. Balancer [26] uses a constant geometric mean AMM, which is the first DEX to allow liquidity pools with more than two tokens.

Liquidity pools are a core component of an AMM and an innovative market structure in DeFi that allows assets to be traded freely without the need for traditional market makers. By constructing homogeneous liquidity pools, they provide dynamic pricing and liquidity support for both trading parties, and automatically adjust asset prices through algorithms, maintaining equilibrium asset ratios and improving data trading efficiency. Regarding research on liquidity pools, scholars have proposed a series of new methods and mechanisms to address market liquidity issues and malicious trading. Othman et al. [27]

pointed out that traditional market makers, unable to adapt to liquidity changes, caused their trading prices to fluctuate similarly in high-volume and low-volume markets, and their trades were always at a loss in typical markets. Therefore, a liquidity-sensitive market maker was constructed based on a variant of LMSR. This market maker generates bounded losses at any initial liquidity level, and in the worst case, its loss is zero even when the initial liquidity level is close to zero. Thus, for any initial liquidity level, a boundary can be established in the market state space, and if the market terminates within this boundary, the market maker can achieve profitability. Mohan et al. [28] proposed a neoclassical black-box-based decentralized AMM algorithm analysis method. Their method can not only visually analyze the geometric shape of a given AMM algorithm but also examine the differences caused by manipulating a given exchange function. For instance, analysis shows that any monotonic transformation of the exchange function in the Constant Product Market Maker (CPMM) does not change a given reserve price. Jiang et al. [29] proposed a Dynamic Curve Automated Market Maker (DCAMM) model that uses a Price Oracle to track market prices in real time and dynamically adjust the price curve. These models are used to solve arbitrage and sandwich attack problems in traditional AMM models and ensure trading fairness.

To further position AMM-based mechanisms within the broader context of digital asset markets and automated financial trading, recent studies have also highlighted the increasing complexity of digital asset markets and data-driven trading systems. Erdoğan [30] analyzed the interconnectedness and risk structure among major digital assets during 2020–2025 using a Generalized R^2 approach, showing that digital assets may exhibit time-varying interactions and risk transmission characteristics. Erdoğan and Deniz [31] further examined the asymmetric connectedness between Ethereum and sustainable digital assets, indicating that digital asset markets may present heterogeneous and dynamic interdependence patterns. In addition, recent studies on high-frequency trading and financial data science have emphasized the role of financial technology, machine learning, real-time data processing, and network optimization in supporting automated and data-driven trading systems [32,33]. These studies provide useful background for understanding digital asset trading and automated financial markets. However, they mainly focus on financial assets, cryptocurrency markets, or high-frequency trading systems, where tradable assets, market data, and price discovery mechanisms are relatively mature. In contrast, the application of AMM-based liquidity pools to data asset trading remains underexplored, especially under the conditions of non-standardized data rights, sparse transactions, limited liquidity, and supply-demand disequilibrium.

Despite the superior performance of AMM mechanisms in the digital asset domain, they rely on assumptions such as asset standardization, high-frequency trading, strong asset correlation, and sufficient market information, making them difficult to adapt directly to data asset trading scenarios with non-standardized rights, limited liquidity, sparse transactions, and uncertain valuation. For example, Curve is mainly designed for stablecoins or highly correlated homogeneous assets, and its low-slippage advantage relies on the assumption that asset prices remain within a narrow range. DCAMM generally depends on external price oracles to track market prices and dynamically adjust pricing curves, which may introduce oracle manipulation risks when reliable external reference prices are unavailable. Predictive AMM approaches usually require sufficient historical market data for model training and market dynamics prediction, whereas data asset trading often faces sparse transactions, limited historical records, and uncertain valuation. Therefore, although these AMM variants provide valuable inspiration, directly applying them to data trading scenarios would introduce additional assumptions that are not fully consistent with the characteristics of data asset circulation. This also means that the exploration of the AMM

mechanism in the data trading field is still in a preliminary stage, and the adaptability of its key mechanisms and risk control issues require further in-depth research.

2.3. Slippage in Data Trading

Slippage is one of the main factors prompting the proposal and adoption of different AMMs, and it is directly related to traders' losses. Slippage is the difference between the expected trading execution price and the actual trading execution price [6]. AMMs exhibit highly volatile, complex, and dynamic issues similar to futures prices [34]. The asset exchange price users see during exchanges often deviates from the actual exchange price, and slippage is commonly used to measure this difference. Slippage occurs whenever the token price changes during a trade. In liquidity pool-based data trading, the slippage problem is particularly prominent, especially with block trades or insufficient liquidity. For instance, in the constant product model, the larger the block trade, the more significant the impact on the asset ratio in the pool, resulting in price deviations from expectations and increased slippage [35].

In Bancor, Bancor v2 relies on the Chainlink Price Oracle to reduce divergence losses caused by arbitrage, and dynamically updates the pricing curve of the liquidity pool, allowing the asset price in the AMM liquidity pool to track its market price, significantly reducing slippage within a specific price range by tracking market prices [3]. In Uniswap, similar to Bancor, prices in the liquidity pool are unstable, and trading volumes related to the liquidity pool size affect the liquidity pool price to varying degrees, larger block trades may result in greater slippage [4]. A constant-sum curve has zero slippage, but due to its fixed price and limited liquidity, and the ease with which one of its pooled assets can be exhausted, it is only suitable for stablecoins and has mainly theoretical significance [36]. Furthermore, Wang et al. [37] proposed a dynamic curve AMM model to derive optimal trading strategies for dynamic curve market makers to minimize slippage. StableSwap/Curve [38] introduced a hybrid invariant that allows trading on a constant-sum curve when the portfolio is relatively balanced and switches to a constant-product curve when it is in disequilibrium. This design achieves lower slippage, but is only suitable for stablecoins because the price of the desired trading range remains close to 1. Wang et al. [39] proposed the Constant Elliptic Curve AMM, whose general form is shown in Equation (2):

$$(x - a)^2 + (y - a)^2 + b \times xy = C, \tag{2}$$

where a b and C are constants. Traders can choose between concave and convex curves in the first quadrant. Compared with traditional AMM models, the proposed constant elliptical curve supports a fixed price interval; thus, the slippage range is also bounded.

With the rapid development of deep learning technology, deep learning-based methods have been widely used in data fusion tasks [40]. Lim et al. [41] proposed a predictive AMM architecture based on deep reinforcement learning. By combining deep learning algorithms and market dynamics prediction, this architecture can dynamically adjust the price curve during trading to reduce slippage losses, while improving the stability of liquidity pools. It effectively optimizes the liquidity provision and price adjustment mechanisms of AMMs.

Unlike financial asset scenarios, data asset trading is characterized by high heterogeneity, discontinuity, and value sensitivity, where block trades and buy-sell disequilibrium are more likely to cause severe price fluctuations. Slippage in such scenarios not only occurs frequently but also stems from complex causes, making traditional AMM slippage control methods ineffective. Therefore, there is an urgent need to develop a slippage identification and control mechanism specifically for the characteristics of data asset trading.

2.4. Research Gaps and Breakthroughs

Based on the above literature review, several research gaps can be identified. First, existing blockchain-based data trading studies mainly focus on transaction transparency, privacy protection, incentive mechanisms, data quality assessment, and fair exchange protocols. Although these studies improve the trustworthiness and security of data trading to some extent, they generally lack continuous market-oriented pricing and automated transaction execution mechanisms. As a result, their ability to support flexible, real-time, and liquidity-driven data asset circulation remains limited.

Second, existing AMM-based liquidity pool mechanisms have been extensively studied and applied in cryptocurrency and financial asset trading. However, these mechanisms are mainly designed for standardized financial assets with relatively high liquidity, frequent trading activities, and mature price discovery mechanisms. Their applicability to data asset trading remains insufficiently explored. In particular, data assets usually involve non-standardized rights, limited liquidity, sparse trading activities, and supply-demand disequilibrium, which may lead to more pronounced slippage when AMM-based trading is directly introduced into data circulation.

Third, current slippage control studies mainly focus on financial trading scenarios, such as dynamic curve adjustment, oracle-assisted pricing, or predictive AMM strategies. These methods often depend on reliable external reference prices, sufficient historical market data, or high-frequency trading environments. However, in data asset trading scenarios, such assumptions may not always hold. More importantly, the slippage generation mechanism caused by block trades and buy-sell disequilibrium in liquidity pool-based data trading has not been systematically analyzed, and execution-level optimization strategies remain underexplored.

To address these gaps, this paper introduces the AMM mechanism into multi-source homogeneous data asset trading and constructs a liquidity pool-based data trading framework. The proposed ESS-LP mechanism further integrates dynamic trade splitting and alternating order sorting to optimize the execution process. By decomposing block trades and reorganizing buy and sell orders according to the liquidity pool state, ESS-LP actively controls slippage caused by liquidity shocks and transaction imbalance. In this way, the study contributes to both the exploration of AMM-based data asset circulation and the development of slippage control mechanisms for liquidity pool-based data trading.

3. Data Trading Model and Slippage Mechanism Analysis Based on AMM

This section presents the liquidity pool-based data trading model and analyzes the slippage mechanism under the AMM framework. The model construction is based on the constant-product AMM mechanism, in which the reserves of data warrants and payment tokens determine the pool price, while trade volume and trading direction determine the disturbance to the liquidity pool state. Accordingly, liquidity pool reserves, trade size, buy/sell direction, expected price, actual execution price, and slippage are selected as the key variables because they directly reflect the AMM price formation and price impact process. Based on this modeling logic, Section 3.1 constructs the data trading liquidity pool model, Section 3.2 analyzes the main factors influencing slippage, and Section 3.3 introduces the proposed ESS-LP slippage optimization mechanism.

3.1. Data Trading Liquidity Pool Model Construction

The essence of the data trading market lies in establishing a networked trading ecosystem that supports multi-agent participation, with its core characteristic manifested in a many-to-many trading model. This paradigm aggregates a large number of homogeneous

data providers and geographically dispersed data consumers within a unified marketplace. Driven by dynamic demands, consumers typically procure large-scale homogeneous data assets to fulfill domain-specific analytical objectives. A representative application scenario involves constructing high-precision supermarket consumption behavior models using customer purchase records. To achieve this, data consumers such as market researchers, suppliers, advertisers, and academic institutions require multidimensional, large-scale datasets to support machine learning training, in-depth data mining, and analytical activities. Concurrently, numerous supermarkets capable of supplying such data act as data providers.

In this operational framework, data generated by Supermarket A is fragmented and distributed across multiple consumers, each of whom imposes distinct purposes, processing methods, retention periods, and re-licensing terms for data usage. Such heterogeneity complicates the traceability of data flows and post-transaction usage for providers like Supermarket A. Conversely, individual consumers must consolidate fragmented datasets from multiple supermarkets to assemble comprehensive, application-specific datasets. This operational complexity underscores the intrinsic attributes of data trading markets: dynamic supply-demand alignment and interwoven data circulation pathways. This scenario exemplifies how decentralized data trading requires both fine-grained contractual governance and cross-source integration mechanisms to reconcile scalability with compliance. The specific scenario is illustrated in Figure 1.

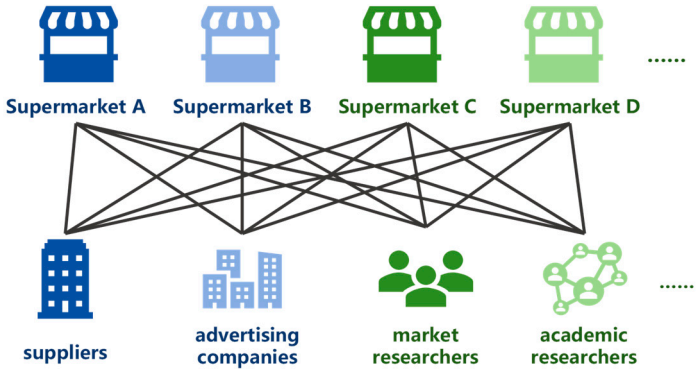


Figure 1. Data Trading Ecosystem.

To address the multi-agent dynamic trading scenario illustrated in Figure 1, this study proposes a multi-party data trading mechanism based on an AMM liquidity pool architecture, leveraging smart contracts to achieve decentralized pricing and trading. Furthermore, we design a slippage optimization algorithm to mitigate the market impact costs caused by block data trading.

3.1.1. Trading Model

Under the liquidity pool-based data trading model, the data trading process is divided into three stages: data assetization, asset certification, and entitlement transactionalization. Data assetization refers to transforming raw data into a standardized asset object, enabling the management of corresponding data assets in the form of digital assets. Asset certification refers to converting the ownership rights of a data asset into a set of entitlement objects, enabling the management of data rights in the form of digital assets according to the ownership system. Entitlement transactionalization refers to the actual trading of data through the exchange of data rights, with digital assets representing data asset rights as the object of exchange. In this process, we define the data meta-certificate as the trading subject of the original data. The data warrant is the subject matter of the ownership of the

data asset represented by the data meta-certificate. This design facilitates the separation of exercisable data ownership from the underlying data asset used as the transaction subject, and simplifies the governance of data assets, as shown in Figure 2.

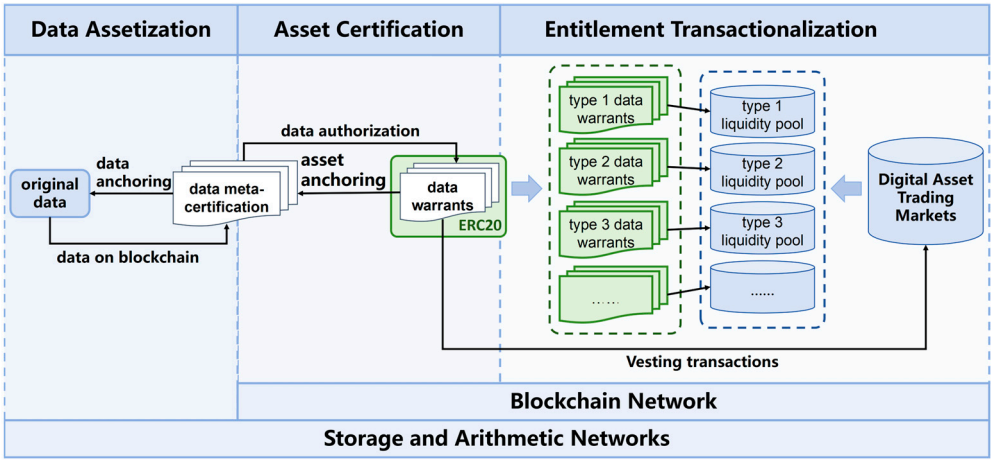


Figure 2. Warrant-oriented data trading ideas.

Based on this concept, this paper utilizes ERC20-based data warrants as the equity targets for data assets. ERC20 is a widely used token standard on the Ethereum blockchain, defining a set of standardized rules for the creation, transfer, and management of tokens on the Ethereum network. By employing ERC20-based digital warrants, the existence and uniqueness of data asset ownership can be represented, while also providing cross-platform interoperability and flexibility, thereby facilitating data trading management. During data trading, similar types of data can generate corresponding data warrants of the same type, which are then traded as trading objects in specific types of liquidity pools.

In the data trading model proposed in this paper, a data market maker liquidity pool model is constructed to isolate the trading parties, forming an indirect data circulation and trading model through interactions between data buyers/sellers and the liquidity pool. For example, a smart device collects supermarket sales records (i.e., customer shopping receipt datasets containing N purchase records) for a specific period and publishes them for sale as a data asset. A data meta-certificate can be created to characterize its asset attributes. For a data meta-certificate, a data warrant or a set of data warrants can be automatically generated, allowing the supermarket as a data seller to deposit these warrants into the supermarket receipt liquidity pool. If no such liquidity pool exists, a supermarket shopping receipt warrant liquidity pool can be created. At the same time, the supermarket deposits supermarket receipt data warrants and tokens to form a supermarket shopping receipt data warrant market maker, and receives the corresponding reward tokens. Data analysts or market researchers as data buyers will enter the liquidity pool to purchase data warrants based on their needs and complete data trading. The data trading process is automatically executed by smart contracts, where purchasers pay corresponding data tokens to obtain the required data. The liquidity pool dynamically adjusts data prices based on supply, demand, and trading volume. For example, if shopping receipt data for a certain period are highly sought after and in high demand, the liquidity pool will raise its price accordingly to balance supply and demand. Furthermore, it is also necessary to design optimized trading strategies based on the slippage issues inherent in liquidity pools to ensure trading efficiency and fairness. Figure 3 illustrates the workflow of the proposed data market maker liquidity pool model.

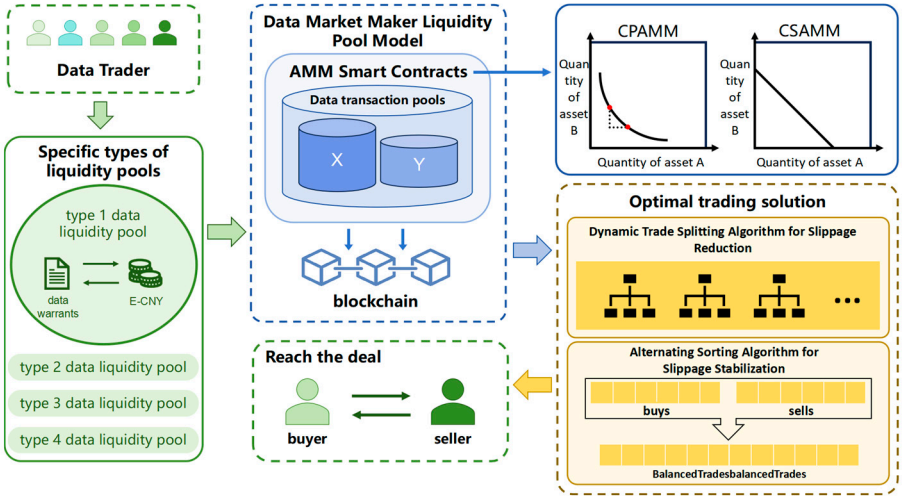


Figure 3. The system architecture of our proposed ESS-LP.

Data trading is supported by different types of liquidity pools, and data traders interact with specific liquidity pools via smart contracts to buy or sell data assets. This model adopts commonly used liquidity pool models, such as the Constant Product Automated Market Maker (CPAMM) and the Constant Sum Automated Market Maker (CSAMM), as references and applies them to liquidity management in the data market. To optimize the trading process, the model introduces a dynamic trade splitting algorithm and an alternating order sorting algorithm to find the optimal trading solution, thereby reducing slippage and improving trading efficiency.

The data trading pool is a liquidity pool consisting of data warrants, Digital Currency Electronic Payment (e-CNY), and a data trading access validation mechanism, as shown in Figure 4. When a new data trading pool is created, data access verification rules are first set, and the initial ratio of data warrants to the payment currency in the liquidity pool is also determined. To incentivize liquidity providers to provide liquidity and ensure the depth of data and monetary assets in the data liquidity pool, the liquidity pool implements a strategy that rewards liquidity providers with trading fees. Once data trading is completed, the liquidity pool calculates the trading price through an AMM algorithm, thus completing the data trading activity.

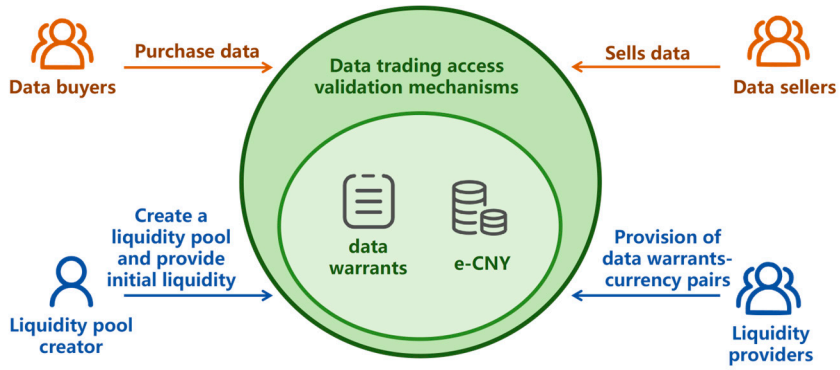


Figure 4. Data trading pools.

3.1.2. Pricing Mechanisms

Consider a data trading pool containing two types of tokens: the data warrant and e-CNY, referred to as MTK and ECNY, respectively, with quantities denoted x_{MTK} and

y_{ECNY}. The AMM allows one token to be exchanged for the other according to a given function, this paper uses constant product curves, as shown in Equation (3):

$$x_{MTK} \times y_{ECNY} = k. \tag{3}$$

Users holding data warrants can choose to simultaneously deposit MTK and ECNY asset pairs into the liquidity pool to provide liquidity and earn fee income directly. Data sellers can generate MTK from their data assets and then sell MTK to obtain ECNY, while data buyers pay ECNY to the trading pool to purchase MTK, through which they can access the corresponding data assets. In the process of mutual exchange between ECNY and MTK, when a user purchases data assets (ignoring trading fees), the amount of data assets purchased is calculated as shown in Equation (4). Similarly, when a user sells data assets, the obtained ECNY is calculated as shown in Equation (5).

$$\Delta x_{MTK} = x_{MTK} - \frac{k}{y_{ECNY} + \Delta y_{ECNY}}, \tag{4}$$

$$\Delta y_{ECNY} = y_{ECNY} - \frac{k}{x_{MTK} + \Delta \mathrm{x_{MTK}}}. \tag{5}$$

In these equations, Δx_{MTK} represents the data warrants purchased, Δy_{ECNY} represents the ECNY spent.

3.1.3. Definition of Slippage

For curve-based AMMs, slippage refers to the loss incurred by a trader due to a mismatch between the pool price at which a trade is initiated and the effective price obtained during the trade. The AMM mechanism in the trading pool is based on the constant product formula, and each trade changes the number of tokens in the pool, which in turn affects the price of subsequent trades. Therefore, slippage directly influences the costs and benefits of both parties involved in data trading. Taking the purchase of MTK as an example, the actual price of purchasing MTK is shown in Equation (6):

$$P_{trade} = \frac{\Delta y_{ECNY}}{\Delta x_{MTK}}, \tag{6}$$

where P_{trade} denotes the actual trading price.

The expected price of purchasing MTK is shown in Equation (7):

$$P_{\mathrm{exp}\,ected} = \frac{y_{ECNY}}{x_{MTK}}, \tag{7}$$

where $P_{\mathrm{exp}\,ected}$ denotes the expected trading price.

The calculation of the slippage H is detailed in Equation (8):

$$H = \left(\frac{P_{trade} - P_{\mathrm{exp}\,ected}}{P_{\mathrm{exp}\,ected}}\right) \times 100\%. \tag{8}$$

The lower the slippage, the more favorable it is for users. In data trading, lower slippage means that traders can trade closer to the expected price, which will increase users' willingness to participate in data trading and further improve the efficiency and attractiveness of the data trading market.

3.2. Analysis of Slippage Mechanisms and Influencing Factors

In this study, we quantitatively and qualitatively analyze the impact of different factors on slippage in the data trading pool through simulation experiments. The experiment

simulates a real trading environment, assuming the existence of the aforementioned data trading pool. In the initialization phase of the data trading pool, a user deposits 1000 data warrants and 1000 tokens into the pool to provide initial liquidity, and a certain amount of initial funds is allocated to each trading user's account. The data trading pool adopts a constant product pricing model, meaning that the product of the number of data warrants and the number of tokens is constant, which can effectively reflect market demand. The total trading duration is 10 min. During this period, three data traders continuously perform buy and sell operations at specific intervals: User 1 trades every 10 s, User 2 every 30 s, and User 3 every 1 min. During the trading process, users' trading behavior occurs randomly and is determined by a random generator configured in the JavaScript simulation scripts. Specifically, the trading direction of each request is generated using the JavaScript *Math.random*() function. In the buy-sell equilibrium scenario, a trade is classified as a buying operation when the generated random value is less than 0.5; otherwise, it is classified as a selling operation. Therefore, the probabilities of buying and selling are both set to 50%. The trading volume is generated according to the predefined volume range of each experiment, and one of these candidate values is randomly selected for each trade. This random decision model is designed to simulate the uncertainty and behavioral differences in users in real trading. During the experiment, key data for each trade, including trading time, price, volume, and slippage, is recorded. Appropriate statistical analysis methods are used to analyze the data to study the impact of different factors on slippage.

3.2.1. Impact of Block Trade on Slippage

To gain a deeper understanding of the impact of block trades on slippage, we designed experiments to observe slippage variation over different trading volume intervals. Specifically, we defined six different trading volume intervals ranging from 0.1% to 20% of the number of tokens in the liquidity pool, as shown in Table 1.

Table 1. Range of trading volumes.

Serial Number	Volume Ranges
1	0.1–0.5%
2	0.5–1%
3	1–5%
4	5–10%
5	10–15%
6	15–20%

Figure 5 illustrates the variation in slippage over time for different volume ranges. It can be observed that for smaller volume ranges, particularly at the 1% level, the market absorbs trades more easily, resulting in lower slippage. However, as the trading volume increases, slippage also increases, indicating that block trades significantly affect slippage, with larger trades leading to higher slippage.

In addition, we use standard deviation as a metric for slippage volatility and analyze the data for trades in different volume ranges. Figure 6 shows the results of comparing the standard deviation of slippage across different volume ranges. A significant positive correlation is observed between the standard deviation of slippage and the trading volume range. Specifically, the standard deviation of slippage increases significantly as the trading volume range grows, showing a clear increasing trend. The relatively small standard deviation in lower volume ranges (e.g., 0.1–0.5% and 0.5–1%) indicates more stable slippage changes. In contrast, higher volume ranges (e.g., 15–20%) have larger standard deviations, reflecting more dramatic slippage fluctuations. Therefore, in block trades, the uncertainty

of slippage changes is stronger, and the stability of the trading environment decreases, indicating higher risks and a potentially unfairer trading environment for data traders.

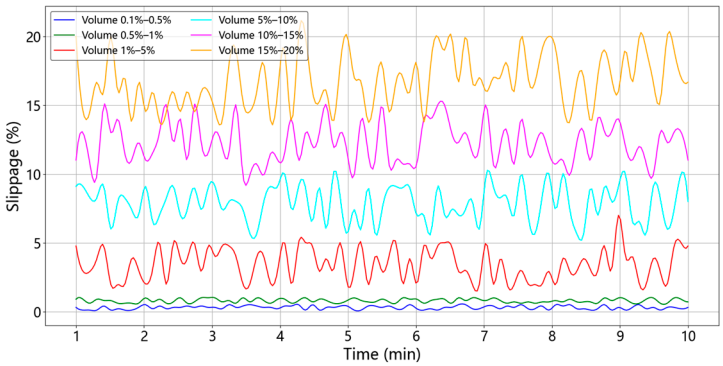


Figure 5. Changes in slippage over time for various volume ranges.

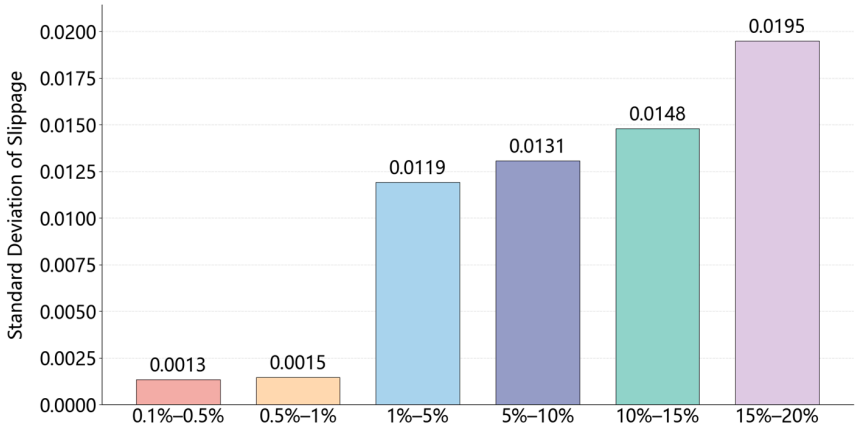


Figure 6. Standard deviation of slippage over a range of volumes.

Figure 7 further quantifies the impact of block trades on slippage, showing the trend in average slippage across trading volume ranges. It can be observed that as the trading volume range increases, the average slippage shows a clear upward trend, consistent with theories of market liquidity and price impact of trading pairs.

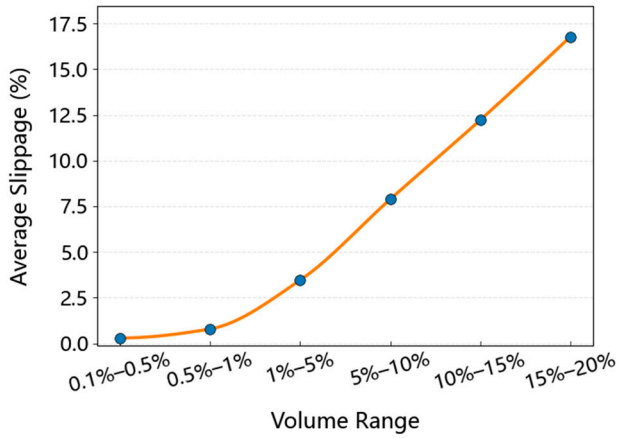


Figure 7. Average slippage over a range of volumes.

Within the 0.1–1% volume range, average slippage is close to 0%, indicating that at smaller trade volumes, the market can efficiently absorb trades with low slippage,

benefiting traders. In the 0.5–1% range, slippage remains relatively low, but begins to show a tendency to increase as the trading volume increases. However, once the trading volume reaches 1%, the market begins to experience the pressure of block trades, causing average slippage to rise significantly, with an accelerated growth rate. When the trading volume reaches the 15–20% range, average slippage increases significantly to nearly 17%, highlighting the substantial impact of block trades on market prices and slippage. In addition, the increase in slippage is non-linear: it grows more slowly for smaller ranges of trading volumes and faster for larger block trades. This suggests that for block trades, the liquidity of the pool appears insufficient, and slippage significantly increases.

3.2.2. Impact of Buy-Sell Equilibrium on Slippage

In financial markets, buy-sell equilibrium refers to a state where the trading intentions and capabilities of buyers and sellers are balanced, meaning that buyers' willingness to purchase matches the sellers' willingness to sell. In the second part of the experiment, we examine the impact of buy-sell equilibrium on slippage by simulating both equilibrium and disequilibrium trading environments.

To simulate a buy-sell equilibrium market environment, we set the probability of traders buying and selling to be equal, with each accounting for 50%, thereby ensuring an equilibrium power between buyers and sellers. In contrast, for buy-sell disequilibrium, the probability of buying or selling activity is set to 80%, simulating situations of oversupply or undersupply in the market. Through multiple repeated simulations, we obtain slippage data under different conditions.

Figure 8 illustrates the trend of slippage over time under both buy-sell equilibrium and disequilibrium conditions. The blue curve represents slippage in the equilibrium state, where the probabilities of users buying and selling are both 50%. It can be observed that the fluctuation amplitude is moderate, with overall stability in the range of 10–20%, indicating that slippage is relatively smooth under equilibrium conditions. The green curve corresponds to buying-dominated behavior, where the buying probability is 80% and the selling probability is 20%, while the red curve corresponds to selling-dominated behavior. It can be seen that the magnitude of slippage fluctuations is significantly higher under disequilibrium conditions, indicating a higher slippage risk.

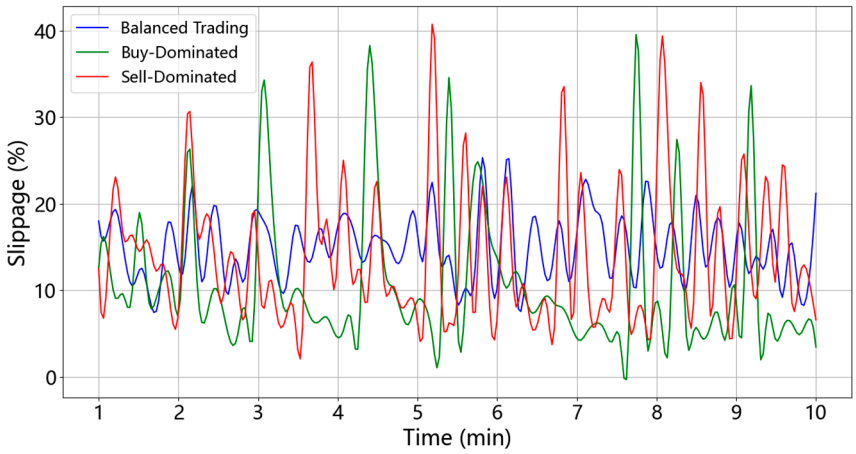


Figure 8. The variation in the slippage points over time for the equilibrium and disequilibrium conditions of buying and selling.

Figure 9 shows the comparison of the standard deviation of slippage under different conditions through a bar chart. The lowest standard deviation of slippage is 0.0394 under the buy-sell equilibrium condition, suggesting low slippage volatility and relatively smooth

market trading under the equilibrium condition. The slippage standard deviation under disequilibrium conditions is significantly higher than that under equilibrium conditions, further confirming the significant impact of buy-sell disequilibrium on slippage volatility.

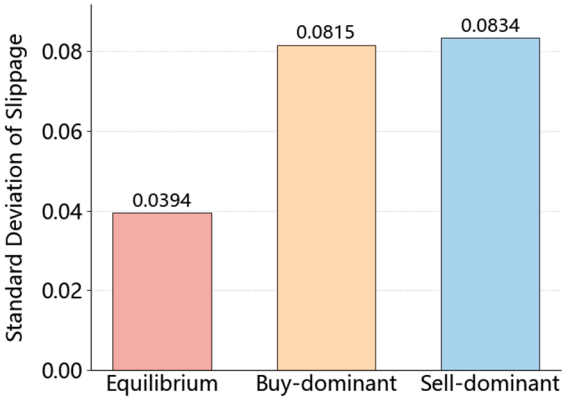


Figure 9. Standard deviation of buying and selling.

In summary, block trades and the degree of market buy-sell equilibrium are key factors influencing slippage. Block trades, due to their impact on market liquidity, often trigger significant price movements, leading to increased slippage. Similarly, the degree of market buy-sell equilibrium directly affects slippage, with implications for the fairness of the trading environment. Therefore, it is crucial for traders to incorporate these factors when formulating and implementing trading strategies, so as to minimize the adverse effects of slippage and effectively improve the overall efficiency and fairness of trading.

3.3. Design and Implementation of Slippage Optimization Algorithm

3.3.1. Design Methodology

Building upon the experimental analysis presented earlier, we identify the key factors that influence slippage and accordingly propose an Effective Slippage Scheme Based on Liquidity Pools (ESS-LP) to minimize slippage. As illustrated in Figure 10, the data trading pool is divided into a pending trading pool and an execution pool. When a trader initiates a new trade, it first enters the pending trading pool. This pool continuously collects trades and performs splitting and sorting operations, after which the processed trades enter the execution pool to achieve optimized trading execution. The pending trading pool and the execution pool operate alternately. This paper focuses on the splitting and sorting mechanisms within the pending trading pool, which include the following key steps:

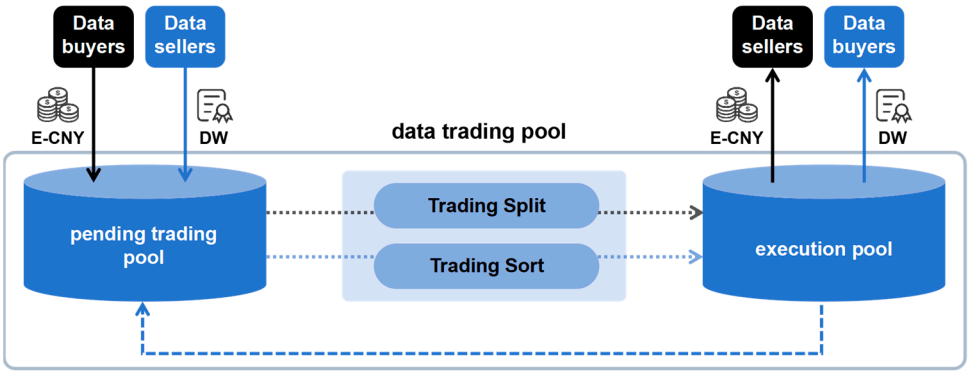


Figure 10. The method of optimizing the slippage. The arrows indicate the processing flow of trading requests among different modules.

- Collection of Trading Requests: Data traders may continuously submit trading requests to the data trading pool within a specified trading time window, and all requests initiated within this window are collected.
- Processing of Trading Requests: Once the time window closes, the system begins processing all collected trading requests. The processing involves two main steps: block trade splitting and trading sorting. First, block trades, defined as those exceeding 1% of the current token supply in the trading pool, are split. Each split is performed according to the 1% liquidity threshold of the trading pool to minimize the impact of block trades on the market price. Second, all trades will be sorted to achieve buy-sell equilibrium. This is achieved by using two arrays to store buy and sell orders respectively, and the system alternately selects trades from the buy and sell order arrays to populate the result array.
- Execution of Trading: After processing the trading requests, the system executes all trades in the result array sequentially, ensuring orderly trade execution while minimizing slippage.

3.3.2. Dynamic Trade Splitting Algorithm for Slippage Reduction

After the trading time window closes, we first identify and process block trades. We define block trades as those whose trading volume exceeds the 1% liquidity threshold of the trading pool. For these block trades, this paper adopts a splitting strategy, dividing them into multiple smaller trades, each not exceeding 1% of the pool's balance, to reduce the impact of a single block trade on market prices.

Algorithm 1 calculates the ratio of trading volume to the token pool balance and compares it with the threshold.

Algorithm 1: Check and Split

Input: *simpleSwapAddress, token, tokenAddress, user, volumeToTrade, balanceToken*
Output: *trades*
$onePercentPoolBalance \leftarrow \frac{balanceToken}{100}$
if $volumeToTrade > onePercentPoolBalance$ **then**
    $remainingVolumeToTrade \leftarrow volumeToTrade$
    while $remainingVolumeToTrade > 0$ **do**
    $currentBalanceToken \leftarrow token.balanceOf(simpleSwapAddress)$
    $currentOnePercentPoolBalance \leftarrow \frac{currentBalanceToken}{100}$
    $splitVolume \leftarrow min(remainingVolumeToTrade, currentOnePercentPoolBalance)$
    $trades.push\begin{pmatrix} user, \\ token, \\ tokenAddress, \\ splitVolume \end{pmatrix}$
    $remainingVolumeToTrade \leftarrow remainingVolumeToTrade - splitVolume$
    end while
else
    $trades.push\begin{pmatrix} user, \\ token, \\ tokenAddress, \\ volumeToTrade \end{pmatrix}$
    end if

If the trading volume exceeds 1% of the pool's balance, the trade enters the splitting process. During splitting, the algorithm iteratively reduces the trading volume through a loop until the remaining volume does not exceed 1% of the pool balance. In each iteration, the algorithm first obtains the current token balance and calculates the current 1% pool balance. Then, it determines the split volume, which is the lesser of the remaining trading volume and the current 1% pool balance. The split trading information is then added to

the trading list, and the remaining trading volume is updated. If the trading volume does not exceed 1% of the pool balance, the original trading information is added directly to the trading list. Finally, the algorithm terminates and returns the updated trading list. This algorithm effectively reduces the impact of block trades on trading pool slippage by dynamically adjusting the trading volume.

3.3.3. Alternating Order Sorting Algorithm for Slippage Stabilization

Next, Algorithm 2 sorts all trades, including split trades, to achieve buy-sell equilibrium, thereby reducing the volatility of slippage within the trading pool.

Algorithm 2: Balance Buys and Sells

```
Input: trades, ECNYTokenAddress, MTKTokenAddress
Output: BalancedTrades
buys ← [], sells ← []
for trade in trades do
  if trade.tokenAddress = ECNYTokenAddress then
    buys.append(trade)
  else if trade.tokenAddress = MTKTokenAddress then
    sells.append(trade)
  end if
end for
BalancedTrades ← []
i ← 0, j ← 0
while i < length(buys) and j < length(sells) do
  BalancedTrades.append(buys[i])
  BalancedTrades.append(sells[j])
  i ← i + 1, j ← j + 1
end while
while i < length(buys) do
  BalancedTrades.append(buys[i])
  i ← i + 1
end while
while j < length(sells) do
  BalancedTrades.append(sells[j])
  j ← j + 1
end while
return BalancedTrades
```

Initially, the algorithm creates two empty lists to store buy and sell orders respectively. Then, through a loop, trades are alternately taken from the buy and sell order lists and added to a new list called *BalancedTrades*. This step ensures the equilibrium execution of both buy and sell orders. This process minimizes slippage caused by one-sided market pressure. If any trades remain in either the buy or sell lists, the algorithm adds them all to the *BalancedTrades* list, ensuring that all trades are executed, even in the case of disequilibrium. Finally, the algorithm returns the *BalancedTrades* list with the optimized sequence of trades.

To further evaluate the computational efficiency of the proposed ESS-LP mechanism, we analyze the complexity of its two core procedures. Let n denote the number of original trading requests collected in the pending trading pool, and let m denote the total number of executable trades after dynamic splitting. For the dynamic trade splitting algorithm, each original trading request is checked once. If the trading volume exceeds the 1% liquidity threshold, the request is divided into multiple sub-orders. Therefore, the time complexity of this procedure is $O(n + m)$, which can be simplified as $O(m)$, since m includes all original and split trading units after preprocessing.

For the alternating order sorting algorithm, the m trades after splitting are first traversed and divided into buy and sell sequences, which requires $O(m)$ time. Then, two index pointers are used to alternately append buy and sell orders to the final execution sequence. Since each trade is accessed and appended once, this procedure also has a time complexity of $O(m)$. Therefore, the overall preprocessing time complexity of ESS-LP is $O(m)$. The space complexity is also $O(m)$, because the algorithm needs to store the split trade list, the buy and sell sequences, and the final balanced trading sequence. This linear complexity indicates that the proposed ESS-LP mechanism can maintain computational efficiency as the number of executable trading units increases.

4. Experimental Evaluation

To validate the efficacy of the proposed ESS-LP slippage optimization mechanism, this study conducts a comparative analysis of system slippage under unified trading environments and simulation parameters. We systematically examine the variation trends in slippage before and after optimization, volatility characteristics, and overall controllability. This section includes three parts: experimental settings, comparative slippage results, and evaluation conclusions, which demonstrate the comprehensive performance of the optimization algorithm in improving trading stability and reducing trading costs.

4.1. Experimental Setting

The experiments were conducted in a local blockchain simulation environment to evaluate the slippage characteristics and optimization performance of the proposed ESS-LP mechanism. The liquidity pool, trading execution, dynamic trade splitting, and alternating order sorting mechanisms were implemented using Solidity smart contracts. The simulation and contract interaction scripts were developed in JavaScript based on the Hardhat framework, which was used for smart contract compilation, deployment, testing, and local blockchain network simulation. The trading records generated from the simulation were further processed using Python version 3.12.0, and the experimental figures were plotted using Matplotlib version 3.8.3. All simulations were executed on a personal computer with a 64-bit operating system.

This experiment replicates the data trading environment described in Section 3. A 10 min trading window was set, during which traders submitted trading requests at fixed intervals. Trade volumes and trading directions were generated probabilistically to simulate behavioral differences among real-world traders. The conventional constant-product AMM with natural execution order is selected as the primary baseline because it shares the same liquidity pool structure and data trading setting as the proposed ESS-LP mechanism, allowing the effect of dynamic trade splitting and alternating order sorting to be evaluated under a fair and consistent experimental environment. The trading mechanisms were divided into two groups: the control group executed trades using a conventional AMM mechanism without applying any slippage optimization strategy; whereas the experimental group implemented the proposed dual slippage optimization mechanism, incorporating liquidity pool state-aware dynamic trade splitting and alternating order sorting. For each experimental scenario, the average slippage and slippage volatility were calculated based on all completed trades within the simulation window.

Slippage evaluation employs three key metrics: Average Slippage (AS), Slippage Standard Deviation (SSD) and Maximum Slippage (MS). The formulas are given in Equations (9)–(11):

$$AS = \frac{1}{n}\sum_{i=1}^{n}|H_i|, \tag{9}$$

$$SSD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(H_i - AS)^2}, \tag{10}$$

$$MS = \max(|H_1|, \ldots, |H_n|). \tag{11}$$

Specifically, AS measures systematic trading cost through the mean price deviation from expectations; SSD quantifies the volatility and stability of slippage; and MS identifies extreme deviations in individual trades to reveal risk exposure. Here, n represents the total number of trading orders in the evaluation period, i denotes the transaction index, and H_i refers to the slippage deviation of each individual trade.

Furthermore, to improve the reliability of the evaluation, each method was independently repeated 10 times under the same experimental scenario and parameter settings. For each run, the average slippage was calculated based on all completed trades within the simulation window. Since the two groups of repeated experiments were independently generated, Welch's t-test was used to evaluate the statistical significance of the difference between the conventional AMM and ESS-LP based on the average slippage values obtained from the repeated runs, with the significance level set to 0.05.

4.2. Comparative Slippage Results

Figure 11 illustrates the temporal evolution of slippage for both control and experimental cohorts throughout the trading session. As can be seen, the pre-optimized mechanism (blue trace) exhibits pronounced slippage spikes with substantial volatility, particularly during the 4–5 min, 6–7 min, and 8–9 min intervals where slippage exceeds the 5.0% threshold. In the data trading context, such elevated slippage signifies heightened price volatility and increased trading costs for block trade executions. In contrast, the proposed dual slippage optimization mechanism (red trace) demonstrates significantly attenuated fluctuations over the same trading session. The maximum observed slippage was 1.1% throughout the whole trading window. Consequently, the optimized platform achieves minimal price volatility during block trade executions, significantly reduces trading costs and improves the efficiency of the data trading market.

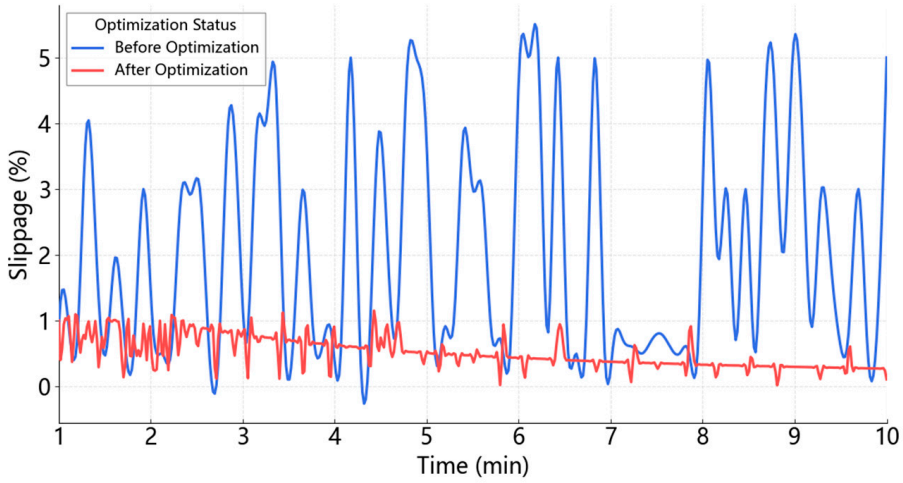


Figure 11. Slippage comparison before and after optimization.

The statistical results shown in Table 2 demonstrate substantial improvements. First, the AS decreases from 2.1% pre-optimization to 0.5% post-optimization, representing a 76.2% reduction. Second, the SSD decreases from 1.66% to 0.25%, indicating an 84.9% reduction in volatility. Third, the MS decreases from 5.5% to 1.1%. These quantitative

metrics confirm the optimization algorithm's significant efficacy in controlling both slippage magnitude and uncertainty.

Table 2. Experimental statistical results.

Indicator Name	Unoptimized	Optimized	Level of Improvement
Average Slippage (%)	2.1	0.5	↑76.2%
Slippage Standard Deviation (%)	1.66	0.25	↑84.9%
Maximum Slippage (%)	5.5	1.1	↑80%

Note: The upward arrows indicate the percentage improvement after optimization.

To further verify the robustness and statistical reliability of the proposed ESS-LP mechanism, repeated experiments were conducted under the same experimental scenario. Specifically, the conventional AMM and ESS-LP were each independently repeated 10 times, and the average slippage of each run was calculated based on all completed trades within the simulation window. The repeated experimental results are summarized in Table 3. The conventional AMM achieved a mean average slippage of 1.9110% with a standard deviation of 0.4609%, whereas ESS-LP achieved a mean average slippage of 0.5924% with a standard deviation of 0.0400%. Compared with the conventional AMM, ESS-LP reduced the mean average slippage by approximately 69.00%. Furthermore, Welch's *t*-test was conducted on the average slippage values obtained from the repeated runs. The result shows that the reduction achieved by ESS-LP is statistically significant at the 0.05 level, with $p = 7.62 \times 10^{-6}$.

Table 3. Repeated experimental results and statistical significance test.

Method	Repeated Runs	Mean Average Slippage (%)	Standard Deviation (%)	Reduction (%)	*p*-Value
Conventional AMM	10	1.9110	0.4609	—	—
ESS-LP	10	0.5924	0.0400	69.00	7.62×10^{-6}

Note: The em dash (—) indicates that the corresponding value is not applicable.

4.3. Evaluation Conclusions

The experimental findings indicate that the proposed dual slippage optimization mechanism not only significantly reduces trading costs throughout the trading session, but also achieves excellent stability in controlling slippage volatility. This enhances the predictability and fairness of the data trading environment. In block trade scenarios, the traditional AMM mechanism suffers from substantial slippage amplification due to liquidity pool impacts. However, the proposed optimization algorithm efficiently mitigates slippage accumulation by splitting block trades into micro-trades and staggering bidirectional execution between buyer and seller orders. This mechanism has been proven to substantially strengthen platform robustness and increase market attractiveness in data trading.

In summary, the dual slippage optimization mechanism proposed in this paper effectively mitigates the slippage problem caused by impacts of single trades and matching disequilibrium. This mechanism integrates dynamic trade splitting and alternating order sorting, which demonstrates theoretical coherence with data asset characteristics and provides empirically validated adaptability. This mechanism provides robust technical support for automated pricing systems in high-frequency or block trade scenarios.

5. Conclusions and the Future Work

At present, the digital economy has become an important engine driving technological innovation, industrial transformation, and economic growth. As a foundational strategic resource, the efficient circulation and trading of data are essential for realizing data value and supporting the development of the digital economy. To address the limitations of traditional centralized data trading platforms in efficiency, transparency, and dynamic pricing, this study introduces the AMM mechanism into data asset trading and proposes the ESS-LP framework. By constructing a liquidity pool-based data trading model, this study provides a preliminary exploration of automated pricing and execution for multi-source homogeneous data assets.

The empirical results provide direct evidence for the effectiveness of ESS-LP in reducing slippage under liquidity pool-based data trading. The analysis shows that block trades and buy-sell disequilibrium are two major sources of slippage volatility. Block trades disturb the reserve ratio of the liquidity pool and increase execution price deviation, while buy-sell disequilibrium causes cumulative directional pressure on the pool state. To address these problems, ESS-LP combines dynamic trade splitting and alternating order sorting. Dynamic trade splitting reduces the price impact of large individual trades, while alternating order sorting alleviates the cumulative imbalance between buy and sell orders.

The experimental findings further demonstrate that ESS-LP improves both slippage magnitude and trading stability. In the representative comparison, the average slippage decreases from 2.1% before optimization to 0.5% after optimization, representing a 76.2% reduction. The slippage standard deviation decreases from 1.66% to 0.25%, indicating a substantial reduction in volatility, and the maximum slippage decreases from 5.5% to 1.1%, showing that ESS-LP can effectively control extreme slippage risk. In addition, the repeated experimental results further support the robustness of the proposed mechanism. The conventional AMM achieves a mean average slippage of $1.9110\% \pm 0.4609\%$, whereas ESS-LP achieves $0.5924\% \pm 0.0400\%$. Welch's t-test confirms that the reduction is statistically significant at the 0.05 level, with $p = 7.62 \times 10^{-6}$. These results indicate that ESS-LP can reduce trading costs, improve price stability, and provide a more predictable execution environment for data trading participants.

These findings also respond to the research gap identified in the literature. Existing AMM and slippage control studies mainly focus on financial assets, cryptocurrency markets, or high-frequency trading scenarios, where assets are more standardized and market data are relatively sufficient. In contrast, data asset trading faces limited liquidity, sparse transactions, non-standardized rights, and supply-demand disequilibrium. By adapting AMM-based liquidity pools to multi-source homogeneous data asset trading and proposing an execution-level slippage optimization mechanism, this study extends the application boundary of AMM mechanisms and provides empirical evidence for the feasibility of decentralized and automated data asset circulation.

Although the proposed ESS-LP mechanism demonstrates promising performance in reducing slippage in liquidity pool-based data trading, several limitations remain. First, the current study mainly validates the proposed mechanism through mathematical modeling and simulation experiments. In real-world data trading markets, its practical applicability may be affected by data ownership verification, privacy protection, regulatory compliance, participant incentives, and the standardization of data products. Therefore, future work will further investigate how ESS-LP can be integrated with practical data exchanges, data compliance frameworks, and data access control systems.

Second, the current study mainly focuses on multi-source homogeneous data trading scenarios, where data warrants of the same type are traded within a specific liquidity pool. This setting helps simplify the modeling of liquidity pool pricing and slippage optimiza-

tion. However, real-world data markets often involve heterogeneous data assets with differences in quality, timeliness, completeness, granularity, usage rights, and commercial value. Therefore, extending ESS-LP from multi-source homogeneous data trading to heterogeneous data trading remains an important direction for future work. Future research will investigate data quality assessment, metadata-based pricing, reputation mechanisms, and multi-factor valuation models to support more accurate valuation and trading of heterogeneous data assets.

Finally, several technical risks should be further considered in practical deployment. If external information such as reference prices or data quality indicators is introduced, oracle manipulation risk may affect pricing fairness and system reliability. In addition, implementing dynamic trade splitting and alternating order sorting through smart contracts may increase gas costs, especially under large-scale trading scenarios. As the number of participants, liquidity pools, and trading requests grows, scalability may also become a key challenge. Future work will explore decentralized oracle designs, gas-efficient smart contract implementation, Layer-2 solutions, hybrid on-chain/off-chain execution, and scalable transaction scheduling mechanisms. We will also further investigate dynamic slippage optimization strategies based on real-time market conditions and liquidity perception, as well as the interaction between slippage optimization and Impermanent Loss (IL), to improve the robustness, fairness, and practicality of ESS-LP in data asset markets.

Author Contributions: Writing—original draft, M.L.; Writing—reviewing and editing, H.S., Y.Q., N.N.X., W.C., L.N.T. and S.W.; Experimental design, H.S. and M.L.; Methodology, H.S. and W.C.; Conceptualization, Y.Q. and N.N.X.; Funding acquisition, H.S. and Y.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported by Zhejiang Province Leading Geese Plan (Grant No. 2024C01107 and Grant No. 2025C02028).

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Ren, M.; Wang, W.; Zhou, W.; Liu, Y.; Yao, A.; Wen, L.; Gong, Y. Promoting the Data Element Marketization: Progress, Problems and Solution. *Doc. Inf. Knowl.* **2024**, *41*, 25–35. [CrossRef]
2. Wang, W.; Zhang, M.; Wang, J. Research and Analysis of Big Data Trading Platforms at Home and Abroad. *J. Intell.* **2019**, *38*, 181–186+194.
3. Hertzog, E.; Benartzi, G. Bancor Protocol Continuous Liquidity for Cryptographic Tokens Through Their Smart Contracts. Available online: https://cryptopapers.info/assets/pdf/bancor.pdf (accessed on 4 June 2026).
4. Adams, H.; Zinsmeister, N.; Salem, M.; Keefer, R.; Robinso, D. Uniswap v3 Core. Available online: https://github.com/Uniswap/v3-core (accessed on 4 June 2026).
5. Curve Finance. Curve Finance. Available online: https://curve.finance/ (accessed on 4 June 2026).
6. Auer, R.A.; Haslhofer, B.; Kitzler, S.; Saggese, P.; Victor, F. The technology of decentralized finance (DeFi). *Digit. Financ.* **2024**, *6*, 55–95. [CrossRef]
7. Zhu, Y.; Ye, Y. Defining data assets based on the attributes of data. *Big Data Res.* **2018**, *4*, 65–76.
8. Hu, D.; Li, Y.; Pan, L.; Li, M.; Zheng, S. A blockchain-based trading system for big data. *Comput. Netw.* **2021**, *191*, 107994. [CrossRef]
9. Jiang, E.; Qin, B.; Wang, Q.; Wu, Q.; Li, S.; Shi, W.; Bi, Y.; Tang, W. BDTS: Blockchain-Based Data Trading System. In Proceedings of the Information and Communications Security (ICICS 2023), Berlin, Germany, 20 October 2023; pp. 645–664. [CrossRef]
10. Xue, L.; Ni, J.; Liu, D.; Lin, X.; Shen, X. Blockchain-Based Fair and Fine-Grained Data Trading with Privacy Preservation. *IEEE Trans. Comput.* **2023**, *72*, 2440–2453. [CrossRef]
11. RapidAPI API Marketplace. Available online: https://rapidapi.com/ (accessed on 4 June 2026).
12. Huang, J.; Li, J.; Tang, K. Data trust: A trustworthy data transaction model. *Big Data Res.* **2023**, *9*, 67–78.

13. Azcoitia, S.A.; Laoutaris, N. A Survey of Data Marketplaces and Their Business Models. *arXiv* **2022**, arXiv:2201.04561. [CrossRef]
14. Amazon Mechanical Turk. Amazon Mechanical Turk. Available online: https://www.mturk.com/ (accessed on 4 June 2026).
15. Tsai, W.-T.; He, J.; Wang, R.; Deng, E. Decentralized Digital-Asset Exchanges: Issues and Evaluation. In Proceedings of the 2020 3rd International Conference on Smart BlockChain (SmartBlock), Zhengzhou, China, 23–25 October 2020; pp. 1–6. [CrossRef]
16. Sang, Y.; Shen, H.; Tan, Y.; Xiong, N. Efficient protocols for privacy preserving matching against distributed datasets. In Proceedings of the Information and Communications Security: 8th International Conference (ICICS 2006), Raleigh, NC, USA, 4–7 December 2006; pp. 210–227. [CrossRef]
17. Chen, X.; Zhang, Z.; Qiu, A.; Xia, Z.; Xiong, N. Novel coverless steganography method based on image selection and StarGAN. *IEEE Trans. Netw. Sci. Eng.* **2020**, *9*, 219–230. [CrossRef]
18. Li, D.; Guo, Q.; Bai, D.; Zhang, W. Research and Implementation on the Operation and Transaction System Based on Blockchain Technology for Virtual Power Plant. In Proceedings of the 2022 International Conference on Blockchain Technology and Information Security (ICBCTIS), Huaihua, China, 15–17 July 2022; pp. 165–170. [CrossRef]
19. Dai, W.; Dai, C.; Choo, K.-K.R.; Cui, C.; Zou, D.; Jin, H. SDTE: A Secure Blockchain-Based Data Trading Ecosystem. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 725–737. [CrossRef]
20. Xu, J.; Paruch, K.; Cousaert, S.; Feng, Y. SoK: Decentralized Exchanges (DEX) with Automated Market Maker (AMM) Protocols. *arXiv* **2023**, arXiv:2103.12732v7. [CrossRef]
21. Park, A. The Conceptual Flaws of Decentralized Automated Market Making. *Manag. Sci.* **2023**, *69*, 6731–6751. [CrossRef]
22. Hanson, R. Logarithmic Market Scoring Rules for Modular Combinatorial Information Aggregation. *J. Predict. Mark.* **2007**, *1*, 3–15. [CrossRef]
23. Heimbach, L.; Wang, Y.; Wattenhofer, R. Behavior of Liquidity Providers in Decentralized Exchanges. *arXiv* **2021**, arXiv:2105.13822v2. [CrossRef]
24. Krishnamachari, B.; Feng, Q.; Grippo, E. Dynamic Curves for Decentralized Autonomous Cryptocurrency Exchanges. *arXiv* **2021**, arXiv:2101.02778. [CrossRef]
25. Buterin, V. Let's Run On-Chain Decentralized Exchanges the Way We Run Prediction Markets. Available online: https://www.cleardao.com/docs/Clear%20Litepaper.pdf (accessed on 4 June 2026).
26. Finance, B. Balancer. Available online: https://balancer.fi/ (accessed on 4 June 2026).
27. Othman, A.; Pennock, D.M.; Reeves, D.M.; Sandholm, T. A Practical Liquidity-Sensitive Automated Market Maker. *J. Predict. Mark.* **2013**, *1*, 1–25. [CrossRef]
28. Mohan, V. Automated market makers and decentralized exchanges: A DeFi primer. *Financ. Innov.* **2022**, *8*, 20. [CrossRef]
29. Jiang, S.; Chen, J.; Li, F.; Geng, H.; Chi, H. DCAMM: Dynamic Curve-Based Automated Market Maker. In Proceedings of the GLOBECOM 2023 IEEE Global Communications Conference, Kuala Lumpur, Malaysia, 4–8 December 2023; pp. 4491–4496. [CrossRef]
30. Erdoğan, B. Interconnectedness and Risk Structure Among Digital Assets: Empirical Findings Based on the Generalized R^2 Approach (2020–2025). *Int. J. Bus. Econ. Stud.* **2025**, *7*, 160–171. [CrossRef]
31. Erdoğan, B.; Deniz, G. Asymmetric connectedness between Ethereum and sustainable digital assets: A Quantile-on-Quantile analysis. *J. Sustain. Digit. Futur.* **2025**, *2*, 165–181. [CrossRef]
32. Balaji, K. Revolutionizing High-Frequency Trading: The Impacts of Financial Technology and Data Science Innovations. In *Machine Learning and Modeling Techniques in Financial Data Science*; Chen, H., Ed.; IGI Global Scientific Publishing: Hershey, PA, USA, 2025; pp. 103–124. [CrossRef]
33. Rella, B.P.R.; Asundi, S.; Kumawat, R.; Pujari, T.D.; Kaushik, K.; Arora, A.S. AI-Driven Network Optimization for Real-Time Financial Data Streaming in High-Frequency Trading Systems. In Proceedings of the 2025 4th OPJU International Technology Conference (OTCON) on Smart Computing for Innovation and Advancement in Industry 5.0, Raigarh, India, 14 July 2025; pp. 1–6. [CrossRef]
34. Gu, Q.; Chang, Y.; Xiong, N.; Chen, L. Forecasting Nickel futures price based on the empirical wavelet transform and gradient boosting decision trees. *Appl. Soft Comput.* **2021**, *109*, 107472. [CrossRef]
35. Heimbach, L.; Schertenleib, E.; Wattenhofer, R. Risks and Returns of Uniswap V3 Liquidity Providers. *arXiv* **2022**, arXiv:2205.08904v2. [CrossRef]
36. Berenzon, D. Constant Function Market Makers: DeFi's "Zero to One" Innovation. Available online: https://medium.com/bollinger-investment-group/constant-function-market-makers-defis-zero-to-one-innovation-968f77022159 (accessed on 4 June 2026).
37. Wang, S.; Krishnamachari, B. Optimal Trading on a Dynamic Curve Automated Market Maker. In Proceedings of the 2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Shanghai, China, 2 May 2022; pp. 1–5. [CrossRef]
38. Egorov, M. Stableswap-Efficient Mechanism for Stablecoin Liquidity. Stableswap Foundation. Available online: https://berkeley-defi.github.io/assets/material/StableSwap.pdf (accessed on 4 June 2026).
39. Wang, Y. Automated Market Makers for Decentralized Finance (DeFi). Available online: https://api.semanticscholar.org/CorpusID:221652658 (accessed on 4 June 2026).

40. Sun, C.; Zhang, C.; Xiong, N. Infrared and Visible Image Fusion Techniques Based on Deep Learning: A Review. *Electronics* **2020**, *9*, 2162. [CrossRef]
41. Lim, T. Predictive crypto-asset automated market maker architecture for decentralized finance using deep reinforcement learning. *Financ. Innov.* **2024**, *10*, 144. [CrossRef]

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.