Thinking Fast and Slow: Data-Driven Adaptive DeFi Borrow-Lending Protocol

Mahsa Bastankhah
Princeton University, NJ, USA

Viraj Nadkarni
Princeton University, NJ, USA

Chi Jin
Princeton University, NJ, USA

Sanjeev Kulkarni
Princeton University, NJ, USA

Pramod Viswanath
Princeton University, NJ, USA

Abstract

Decentralized finance (DeFi) borrowing and lending platforms are crucial to the decentralized economy, involving two main participants: lenders who provide assets for interest and borrowers who offer collateral exceeding their debt and pay interest. Collateral volatility necessitates over-collateralization to protect lenders and ensure competitive returns. Traditional DeFi platforms use a fixed interest rate curve based on the utilization rate (the fraction of available assets borrowed) and determine over-collateralization offline through simulations to manage risk. This method doesn't adapt well to dynamic market changes, such as price fluctuations and evolving user needs, often resulting in losses for lenders or borrowers. In this paper, we introduce an adaptive, data-driven protocol for DeFi borrowing and lending. Our approach includes a high-frequency controller that dynamically adjusts interest rates to maintain market stability and competitiveness with external markets. Unlike traditional protocols, which rely on user reactions and often adjust slowly, our controller uses a learning-based algorithm to quickly find optimal interest rates, reducing the opportunity cost for users during periods of misalignment with external rates. Additionally, we use a low-frequency planner that analyzes user behavior to set an optimal over-collateralization ratio, balancing risk reduction with profit maximization over the long term. This dual approach is essential for adaptive markets: the short-term component maintains market stability, preventing exploitation, while the long-term planner optimizes market parameters to enhance profitability and reduce risks. We provide theoretical guarantees on the convergence rates and adversarial robustness of the short-term component and the long-term effectiveness of our protocol. Empirical validation confirms our protocol's theoretical benefits.

2012 ACM Subject Classification Theory of computation → Market equilibria

Keywords and phrases Defi borrow-lending, adaptive market design, decentralized finance

Digital Object Identifier 10.4230/LIPIcs.AFT.2024.27

Related Version *Full Version*: https://arxiv.org/pdf/2407.10890 [6]

Funding This work was supported by the National Science Foundation via grants CNS-2325477, the Army Research Office via grant W911NF2310147, a grant from C3.AI, a SEAS Innovation grant from Princeton University and a gift from XinFin Private Limited.

1 Introduction

Decentralized Finance (DeFi) has revolutionized lending and borrowing by eliminating centralized intermediaries. The main paradigm shift has been around moving away from opaque financial entities such as banks, that use proprietary models and data to match deposits with borrowers [31], to transparent pools with published algorithms to change interest rates, and borrowing conditions. Major DeFi lending platforms like Aave [2] and Compound [14] function through these liquidity pools, where lenders provide capital that borrowers can access. These protocols ensure that borrowers pledge enough collateral to cover their debt, along with an additional safety buffer.

The simplest variable of interest that any lending protocol seeks to control is the supply-demand ratio of the pool, referred to as "utilization." The objective is to maintain a stable utilization around a designated "optimal utilization" threshold. A coarse rule of thumb is that when the utilization is low, interest rates remain low to encourage borrowing [5]. As utilization increases, interest rates rise to balance demand with supply and to prevent excessively high utilization, which could restrict lenders' ability to withdraw their funds, thus rendering the market less attractive.

Besides the interest rate, other parameters like the over-collateralization ratio, also known as the "collateral factor," govern the long-term risks and profits of the market [30]. In particular, the cash flow that any lender gets from the protocol is at risk of liquidation, and in the more severe cases, default. This risk can be minimized by demanding a large amount of collateral from borrowers, which makes the risk vanishingly small while making the lending market incredibly inefficient and unattractive for borrowers, especially if the asset used as the collateral does not suffer frequent price fluctuations. Thus, the collateral factor needs to be determined based on a careful analysis of recent historic behavior of the collateral asset price, and the risk appetite of the lender.

Present DeFi platforms fix the interest rate as a static function of the utilization [3, 15]. Choosing utilization as the primary indicator of both supply/demand dynamics and market risk/attractiveness and employing a fixed interest rate curve to manage these aspects is very arbitrary and manually determined. Furthermore, traditional DeFi borrowing and lending markets set the collateral factor through a comprehensive process involving community proposals and review phases [1, 13]. However, this method is notably slow and struggles to adapt quickly to rapid market changes, potentially leading to losses and excessive risks due to the delayed adjustment of parameters in response to market fluctuations and experiencing long periods of extreme low liquidity or market inefficiencies. For instance, the authors of [21] have found that the markets for DAI and USDC frequently exhibit periods of extreme low liquidity with utilization exceeding 80% and 90%, respectively, which further highlights the inadequacy of current interest rate models.

In this study, we propose an adaptive, automated, and data-driven approach for designing a borrowing/lending protocol. We begin by modeling the behaviors of borrowers and lenders based on their incentives in a principled manner and examine how external factors alter these behaviors over time. We then define market equilibrium (Definition 2), where the market remains stable within a broader external market, and rates offered by our protocol do not allow borrowers or lenders to gain an advantage over external market rates. Achieving equilibrium is crucial in a market with conflicting interests, as instability tends to disproportionately benefit one group over another, reducing overall fairness and attractiveness of the market [23]. If a protocol cannot dynamically adjust to achieve equilibrium, it risks losing liquidity and users. Market stability must be promptly restored after disruptions, which may be caused by changes in external market conditions or shifts in price distributions that affect the market's risk and profit structure.

Traditional stationary-curve borrowing/lending markets depend on user interactions to push towards equilibrium – for instance, high interest rates prompt borrowers to repay loans, reducing utilization and interest rates. However, this process is slow and often results in impermanent loss [36], especially for users with less flexibility in managing their assets. In order to address this problem, our protocol includes an "interest rate controller" submodule that learns the equilibrium interest rate from user behaviors, providing a faster convergence rate, even in the presence of uninformed users who lack precise information on competitive rates. Unlike traditional methods, our approach does not solely rely on user actions but actively learns from them to accurately assess and adapt to market conditions. Moreover, we provide an adversarial robust version of our interest rate controller as well which learns the equilibrium interest rate as long as the adversary controls less than 50% of the borrowing demand.

In addition to promptly restoring stability following market disruptions, it is essential for a protocol to adaptively optimize hyperparameters that enhance long-term system efficiency and manage risk. Our protocol features a long-term planner that uses the collateralization ratio as a control variable to adapt to market changes and stabilize the market at a desired level (Section 3.2). The collateralization ratio is critical for managing long-term risks and rewards in the system. This ratio has a complex relationship with user behavior and the overall risk and profitability of the market, which we explore in detail in our paper (Section 4). The objectives of the protocol within this long-term planner can be defined in many different ways; In this paper our focus is on maintaining long-term utilization at a target level and controlling default, however more complex objective functions could be implemented to address specific market needs or objectives. Our approach provides a general framework for designing adaptive markets with heterogeneous users who may have varying incentives.

Additionally, we implemented our protocol and tested it with simulated borrowers and lenders, empirically comparing its performance against fixed-curve baselines. We evaluated the correctness of our theoretical guarantees in practice and demonstrated that our interest rate controller can quickly learn the equilibrium interest rate after each market disruption, regardless of borrowers' and lenders' elasticity. In contrast, the baseline protocol fails to find the equilibrium interest rate when user elasticity is low due to its reliance on user reactions to push the market toward equilibrium. Moreover, we showed that in the presence of major market changes, our protocol's collateral factor planner adaptively activates. By learning new price and market parameters, it sets the collateral factor to maintain utilization near a predefined optimal level in the long term.

Related work

Various models on lender and borrower behavior and their equilibria have been explored. [12] assume parametrized supply and demand curves based on interest rates, approximating the curve around equilibrium to recommend rates, but they ignore external markets and default risk minimization. [33] consider external markets, measuring protocol efficiency by interest rate differences, but their models lack long-term decision-making and liquidation considerations. [10] examines Nash equilibrium in a model with independent quality shocks, showing that exogenous asset prices yield one equilibrium, while protocol-influenced prices cause oscillation and propose ad-hoc contract adjustments. Empirical studies on lender/borrower behavior [19, 20, 34] inform our parameter values. [35] discusses borrower trading strategies with market makers. Adversarial attacks on lending protocols have also been highlighted [9, 11, 8].

Several recent works in the mechanism design of financial systems have been advocating for the use of automated adaptivity [40]. The presence of impermanent loss and arbitrage loss in the design of market makers has also spawned multiple works in adaptive market making [18, 27, 29]. We seek to bring similar automated methods to lending in DeFi. Platforms such as Morpho [28] and Ajna [4] provide lenders and borrowers more flexibility when it comes to equilibrium interest rate discovery via an order-book like structure. However, such protocols require constant monitoring on the part of the participants for fairness and optimality. Our objective is to bring these notions of fairness/optimality to more passive pool-based lending protocols.

The methods used in this work are based on optimal control and filtering literature using the least squares method [22]. This method has been used in the estimation of underlying dynamics, given a noisy access to measurements [25, 39]. Several recent works have provided extensions of this algorithm to ensure adversarial robustness [7, 37, 26], which use hard thresholding and concentration inequalities to weed out adversarial data.

2 Problem formulation

2.1 Market actors

The DeFi borrowing and lending market includes four key participants: lenders, borrowers, liquidators, and the protocol, here called $\mathcal{P}$. These actors interact within a shared pool. This subsection briefly clarifies each participant's role and the mechanisms protocols use to regulate their interactions.

To prevent defaults during price declines, the protocol employs *liquidation*. This occurs when a borrower's *loan-to-value* ratio (debt-to-collateral value) exceeds a threshold liquidation threshold (LT), set between 0 and 1 and higher than the initial loan-to-value ratio c. When this threshold is surpassed, liquidators can claim a portion of the borrower's collateral to repay the debt, reducing the loan-to-value ratio. Liquidators receive a fee LI from the borrower's collateral. Liquidations enhance system safety but are unfavorable for borrowers due to the incentive fee. This prompts borrowers to increase their collateral preemptively. Despite liquidation mechanisms, defaults can occur if collateral prices drop abruptly or if liquidators lack sufficient incentives to act.

The protocol must adjust parameters $\{r_t, c_t, LT_t, LI_t\}$ over time to stabilize the pool. Objectives include stabilizing loan supply and demand by setting an interest rate r_t and optimizing parameters to minimize defaults and liquidations while maintaining an ideal utilization rate. Our paper focuses on creating a competitive DeFi protocol with efficient rates, not on revenue maximization. The openness of DeFi protocols and minimal fees should ensure that the most competitive protocol eventually dominates the market.

The lending pool consists of two assets: a stable asset, $\mathcal{A}_l$, provided by lenders for interest, and a volatile asset, $\mathcal{A}_c$, used by borrowers as collateral. Borrowers can only borrow a fraction collateral factor (c) of their collateral, set by the protocol. At timeslot t, the overall pool's assets of type $\mathcal{A}_l$, considering both lent-out funds and available liquidity, are denoted by L_t. Note that L_t increases over time as lenders accrue interest on the lent-out portion. The asset of a particular lender i is represented by $L_t(i)$.

The interest rate r_t is set by the protocol at each block. Borrowers pay this rate, but lenders earn interest only on the utilized fraction U_t of their deposit, defined as $U_t = \frac{B_t}{L_t}$, where B_t represents the overall debt across all borrowers, and $B_t(i)$ represents the debt of borrower i. The debt amount also increases over time due to accrued interest. The quantity of the overall collateral posted by all borrowers is denoted by C_t, and $C_t(i)$ denotes the

collateral of borrower i. Hence, $p_t \cdot C_t$ determines the value of the collateral in terms of the lent-out asset, $\mathcal{A}_l$ (for a thorough list of the notations and their description refer to Appendix D of the full paper [6]). When borrowers repay, they return the loan plus interest and retrieve their collateral. Lenders receive interest based on the protocol rate and fund utilization. Defaults can affect the final interest rate for lenders. If collateral value drops below the debt, the protocol cannot compel repayment of the insolvent debt, resulting in a loss that impacts the lenders' interest rate.

2.2 Environment model

Asset price model. We operate within discrete time intervals, denoted as Δ, each corresponding to one blocktime. We use a discrete price model to monitor the collateral asset's price from one block to the next. For simplicity, we assume the lent-out asset is a stablecoin with a relatively stable price, while the collateral asset's price follows an exogenous geometric Brownian motion with volatility σ. We assume constant volatility over short periods of time, with occasional sporadic jumps, but no fluctuations from one timeslot to the next. In particular, we assume that the price volatility is constant within timescales denoted by $T_m > 1$ (m for market, denoting the timeframe within which the market is stable) which consists of multiple timeslots and can change arbitrarily every T_m timeslots.

The price at time t, denoted as p_t, follows a Geometric Brownian motion with drift μ_{price} and volatility σ. The initial price p_0 is the starting point, for notation simplicity we normalize and consider $\Delta = 1$ and hence formally, the model is:

$$p_t = p_{t-1} \exp\left(\mu + \sigma\varepsilon_t\right), \quad \varepsilon_t \sim \mathcal{N}(0,1) \tag{1}$$

where ε_t is the innovation term for the volatility.

External market competition. We assume the existence of an external competitor market which offers risk-free borrow rate r_o^b and risk-free lend rate r_o^l. These rates are constant during each market period T_m and can change arbitrarily between periods. This assumption accounts for the competition and the broader market within which our protocol operates. r_o^l and r_o^b might represent the existence of complex alternatives rather than simple risk-free rates. In Appendix B of the full paper [6], we discuss interpreting these parameters based on real-world strategies and competitors in the Defi ecosystem. Throughout the paper, we abstract these concepts into r_o^l and r_o^b.

2.3 Protocol behaviour and pool logic

In this section, we establish the structure of a decentralized borrowing-lending protocol, denoted by $\mathcal{P}$. The protocol fulfills two primary roles: 1) $\mathcal{P}$ sets the pool's parameters for each block, denoted as $\{r_t, c_t, LT_t, LI_t\}$, by transmitting a transaction to the underlying blockchain. These parameters govern the pool's logic. 2) $\mathcal{P}$ updates the state variables L_t, B_t, and C_t every block to apply interest rate accumulation and liquidation or default due to price fluctuations.

Handling default. At the beginning of each timeslot t, $\mathcal{P}$ receives the latest price of $\mathcal{A}_c$, p_t, from an oracle. The protocol calculates potential defaults accrued in the last timeslot for each user. The default for borrower i at timeslot $[t-1, t]$ is:

$$\pi_{t-1}^i(p_t) \coloneqq \max\left\{0, B_{t-1}(i) - C_{t-1}(i) \cdot p_t\right\} \tag{2}$$

The overall default, normalized by L_{t-1}, is:

$$\pi_{t-1}(p_t) \coloneqq \frac{1}{L_{t-1}} \sum_{i \in \text{borrowers}} \max\left\{0, B_{t-1}(i) - C_{t-1}(i) \cdot p_t\right\} \tag{3}$$

The protocol seizes the remaining collateral of defaulted positions, exchanges it for $\mathcal{A}_l$, and sets the debt of defaulted borrowers to zero. The gained $\mathcal{A}_l$ assets are added back to the pool. Moreover the underwater debt is deduced from the lender's deposit accordingly:

$$L_t(i) = L_{t-1}(i) - \pi_{t-1}(p_t) \cdot L_{t-1}(i), \quad \forall\, i \in \text{Lenders} \tag{4}$$

For a more thorough explanation of why we handle defaults in this way rather than using a safety reserve similar to most of the current working Defi borrow-lending platforms refer to Appendix A of the full paper [6].

Interest update. The debt of non-defaulted borrowers is updated by:

$$B_t(i) = B_{t-1}(i) \cdot (1 + r_{t-1}), \quad \forall\, i \in \text{Borrowers}$$

The interest rate on the utilized portion of the pool applies to the lenders as well:

$$L_t(i) = L_t(i) \cdot \left(1 + r_{t-1} \frac{B_{t-1}}{L_{t-1}}\right), \quad \forall\, i \in \text{Lenders}$$

Liquidation. $\mathcal{P}$ tracks borrow positions exceeding the liquidation threshold. A position i is eligible for liquidation if $LT_{t-1} < \frac{B_t(i)}{C_t(i) \cdot p_t} < 1$. Liquidators reduce the user's debt by purchasing collateral, restoring the loan-to-value ratio below LT_{t-1}. The debt and collateral after liquidation are updated accordingly. The minimum liquidation amount that reduces the user's loan-to-value below LT_{t-1} is

$$\lambda^i_{t-1}(p_t) \coloneqq \max\left\{0, \frac{B_t(i) - LT_{t-1} \cdot C_t(i) \cdot p_t}{1 - LT_{t-1}(1 + LI_{t-1})}\right\}$$

Setting new parameters. $\mathcal{P}$ sets new parameters for the next timeslot: $\{r_t, c_t, LT_t, LI_t\}$. These parameters determine the interest rate, maximum loan-to-value, and liquidation parameters for the next timeslot.

Admitting new users. The protocol accepts all new lend and repay requests. Borrow requests are accepted only if they adhere to the maximum collateralization factor c_t and there is sufficient $\mathcal{A}_l$ in the pool to satisfy the request. Additionally, it processes withdrawal requests from lenders as long as the withdrawal amount does not exceed the available $\mathcal{A}_l$ in the pool.

2.4 User behavior model

In this section, we establish the model that a rational user would use to interact with the protocol. We consider a continuum of lenders and borrowers, each controlling a single unit of demand or supply. In each timeslot, lenders can deposit or withdraw their unit, and borrowers can borrow, repay, or adjust their loan-to-value ratio by sending a transaction to the smart contract.

2.4.1 Lender

We assume that a continuum lender with one unit of supply, planning for the next timeslot, will calculate the following utility function at time t to decide whether to deposit into the pool or, if already deposited, to withdraw and invest in another external alternative offering r_o^l.

$$\text{Utility}_t^l \coloneqq r_t U_t - \mathbb{E}\left[\pi_t(p_{t+1})\right] - r_o^l \tag{5}$$

where the expectation is over the price at timeslot $t+1$ price. The first term in 5 represents the interest earned by the lender on the utilized portion of their deposit. Although the interest rate is compounded, we approximate it linearly for simplicity. The second term denotes the normalized defaulted debt deducted from the lender's deposit. Finally, we subtract the external interest rate r_o^l to account for the lender's opportunity cost.

We now describe the dynamics by which lenders add or withdraw their deposits based on utility. We introduce a new parameter η_l, which reflects the average elasticity of lenders at time t. This value can change over time (not faster than T_m) and is unknown to the protocol. We assume that the relative rate at which lenders deposit or withdraw from the pool is governed by the following model:

$$\begin{aligned}\frac{L_{t+1}-L_t}{L_t} &= \eta_l \cdot \text{Utility}_t^l + \varepsilon_t \\ &= \eta_l \cdot \left(r_t U_t - \mathbb{E}[\pi_t(p_{t+1})] - r_o^l\right) + \varepsilon_t, \qquad \varepsilon_t \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,\zeta^2)\end{aligned} \tag{6}$$

The noise term accounts for the behavior of uninformed or less informed users.

2.4.2 Borrower

From observing the real lending markets, we identify two types of DeFi borrowers, financing and leveraged trading borrowers (see Appendix B.2 of the full paper [6] for more details).

The first type borrows an asset to use elsewhere, gaining value by leveraging it, e.g., for yield farming or real-world purposes. This group's value from the borrowed asset is represented as r_o^b, measured as an interest rate. The utility function of a borrower of this type controlling one unit of demand, considering the opportunity cost of locked collateral, is

$$\begin{aligned}\text{Utility}_t^{b,1} \coloneqq r_o^b - r_t + \mathbb{E}\left[\pi_t^i(p_{t+1})\right] - \mathbb{E}\left[\lambda_t^i(p_{t+1})\right] \cdot LI_t \\ + C_t(i) \cdot \mathbb{E}\left[\mathbb{1}_{p_{t+1}-p_t<0}(p_{t+1}-p_t)\right]\end{aligned} \tag{7}$$

This includes inherent value (r_o^b), interest rate ($-r_t$), default value ($\mathbb{E}\left[\pi_t^i(p_{t+1})\right]$), liquidation cost ($-\mathbb{E}\left[\lambda_t^i(p_{t+1})\right] \cdot LI_t$), and opportunity cost of locked collateral $\mathbb{E}\left[\mathbb{1}_{p_{t+1}-p_t<0}(p_{t+1}-p_t)\right]$.

The second type aims to take a long position on $\mathcal{A}_c$. They borrow $\frac{1}{p_t}$ units of $\mathcal{A}_c$ from an external provider $\mathcal{Z}$, add more collateral to meet the over-collateralization requirement c_t, and borrow one unit of $\mathcal{A}_l$ from $\mathcal{P}$, then exchange it for $\mathcal{A}_c$ to repay $\mathcal{Z}$. This borrowing strategy is studies in details in [35]. Their utility function is:

$$\text{Utility}_t^{b,2} = -r_t + \mathbb{E}\left[\pi_t^i(p_{t+1})\right] - \mathbb{E}\left[\lambda_t^i(p_{t+1})\right] \cdot LI_t + \mathbb{E}\left[\frac{p_{t+1}-p_t}{p_t}\right] \tag{8}$$

This includes interest rate ($-r_t$), default value ($\mathbb{E}\left[\pi_t^i(p_{t+1})\right]$), liquidation cost ($-\mathbb{E}\left[\lambda_t^i(p_{t+1})\right] \cdot LI_t$), and gain from a price change of the $\frac{1}{p_t}$ investment in $\mathcal{A}_c$ which was possible through interacting with $\mathcal{P}$ i.e.,($\mathbb{E}\left[\frac{p_{t+1}-p_t}{p_t}\right]$. Refer to Appendix B.2 of the full paper [6] for more details.

Assuming α fraction of borrowers are type 1 and the rest type 2, the rate of borrowing or repaying is a linear function of borrower's elasticity η_b and their average utility plus noise:

$$\begin{aligned}\frac{B_{t+1}-B_t}{B_t} &= \eta_b \cdot \left(\alpha \cdot \text{Utility}_t^{b,1} + (1-\alpha)\cdot \text{Utility}_t^{b,2}\right) + \varepsilon_t \\ &= \eta_b \cdot \Bigg(\alpha \cdot r_o^b - r_t + \mathbb{E}\left[\pi_t^i(p_{t+1})\right] - \mathbb{E}\left[\lambda_t^i(p_{t+1})\right]\cdot LI_t \\ &\quad + \alpha \cdot C_t(i)\mathbb{E}\left[\mathbb{1}_{p_{t+1}-p_t<0}(p_{t+1}-p_t)\right] + (1-\alpha)\cdot \mathbb{E}\left[\frac{p_{t+1}-p_t}{p_t}\right]\Bigg) + \varepsilon_t, \end{aligned} \tag{9}$$

where $\varepsilon_t \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,\zeta^2)$.

Throughout this paper, we assume that $\mathcal{P}$ efficiently sets the buffer between the collateral factor and the liquidation threshold to ensure that the expected liquidation, $\mathbb{E}[\lambda_t^i(p_{t+1})]$, for a borrower i who maintains the posted collateral factor, is negligible. This assumption differs from the current borrowing and lending platforms, which experience significant liquidations even among borrowers who adhere to the posted collateral factor. However, we believe that a competent borrowing and lending protocol should set risk parameters to minimize this risk. In Section 3.2, we explain how we determine the liquidation threshold and collateral factor to ensure that the expected liquidations remain negligible. Moreover, throughout this paper, we assume that the liquidation incentive, LI_t, is set sufficiently high by the protocol to encourage the borrowers to maintain the collateral factor, c_t, and avoid liquidations.

▶ **Lemma 1** (Maximum loan-to-value adoption). *Consider the following conditions:*

- *The collateral factor, c_t, and the liquidation threshold, LT_t, are chosen such that for a given LT_t, c_t is the maximum collateral factor that ensures the expected liquidation, $\mathbb{E}[\lambda_t^i]$, is approximately zero for a user i who maintains c_t.*
- *The liquidation incentive, LI_t, is set high enough to incentivize rational borrowers to avoid liquidation by ensuring that $\frac{B_t(i)}{C_t(i)p_t} \leq c_t$.*

Then rational borrowers will adopt the maximum loan to value allowed by the protocol i.e., c_t.

Hence from now on, we assume that $\mathbb{E}[\lambda_t^i]$ for a rational continuum borrower is negligible and for ease of notation, we denote it by $\lambda(c_t, LT_t)$. The proof of all the lemmas and theorems can be found in Appendix C of the full paper [6].

2.4.3 Liquidator

We assume that the liquidation incentive LI is set at a level that consistently incentivizes liquidators, ensuring their prompt engagement and immediate liquidation up to the limit allowed by $\mathcal{P}$. Additionally, since we use an exogenous price model for collateral, phenomena like liquidation spirals studied in previous works [24] are not considered in our analysis.

2.5 Equilibrium analysis

In this section, we will formally define the concept of equilibrium in the borrow-lending framework we established. And we will analytically identify the set of equilibria of this market when users follow the behaviour outlined in 2.4.

▶ **Definition 2** (Market equilibrium). *A lending pool governed by protocol $\mathcal{P}$ parameterized by $\{r_t, c_t, LT_t, LI_t\}$, and lender's and borrower's behavior respectively governed by 6, and 9 is in equilibrium if and only if:*

$$\mathbb{E}\left[\frac{B_{t+1}-B_t}{B_t}\right] = 0 \quad \textit{and} \quad \mathbb{E}\left[\frac{L_{t+1}-L_t}{L_t}\right] = 0$$

where the expectation is over the noise term in the user behavior model.

▶ **Lemma 3** (Simplified default and price change terms). *In the presence of rational continuum lenders and rational continuum borrowers who follow collateral factor c_t (due to Lemma 1) we have:*

$$\begin{aligned}\pi(c_t) &:= \mathbb{E}\left[\pi_t^i(p_{t+1})\right] \\ &= \Phi\left(\frac{\log(c_t)-\mu}{\sigma}\right) - \frac{\exp\left(\frac{\sigma^2}{2}+\mu\right)}{c_t}\cdot\Phi\left(\frac{-\mu+\log(c_t)-\sigma^2}{\sigma}\right)\end{aligned} \tag{10}$$

$$\mathbb{E}\left[\pi_t(p_{t+1})\right] = U_t\pi(c_t) \tag{11}$$

$$C_t(i)\mathbb{E}\left[\mathbb{1}_{p_{t+T_u}-p_t<0}(p_{t+T_u}-p_t)\right] = \frac{1}{c_t}\left(e^{\mu+\frac{\sigma^2}{2}}\frac{\Phi\left(\frac{-\mu-\sigma^2}{\sigma}\right)}{\Phi\left(\frac{-\mu}{\sigma}\right)}-1\right) \tag{12}$$

$$\mathbb{E}\left[\frac{p_{t+1}-p_t}{p_t}\right] = e^{\mu+\frac{\sigma^2}{2}}-1$$

Where $\phi(.)$ and $\Phi(.)$ denote the PDF and CDF, respectively, of the standard normal distribution.

Throughout the rest of the paper, we denote the expected default for one unit of debt by $\pi(c_t)$. Conceptually, Lemma 3 implies that if the pool's loan-to-value ratio is maintained close to c_t in each timeslot, the probability of default remains memoryless. This is due to the Brownian motion of price, where the price ratio between consecutive timeslots follows a stationary distribution. Therefore, the probability that the next timeslot's price falls below the default threshold is stationary and memoryless, changing only when the price distribution itself changes.

▶ **Theorem 4.** *Let the lender behavior be described by Equation 6, the borrower behavior by Equation 9 (for $\eta_b > 0$), and the assumptions of Lemma 1 hold. A protocol with parameters $\{r_t, c_t, LI_t, LT_t\}$, achieves a non-trivial equilibrium if and only if $r_t = r^*$:*

$$r^* = \alpha\, r_o^b + \pi(c_t) + \frac{\alpha}{c_t}\left(e^{\mu+\frac{\sigma^2}{2}}\frac{\Phi\left(\frac{-\mu-\sigma^2}{\sigma}\right)}{\Phi\left(\frac{-\mu}{\sigma}\right)}-1\right) + (1-\alpha)\left(e^{\mu+\frac{\sigma^2}{2}}-1\right) \tag{13}$$

Furthermore, if $\eta_l > 0$, then the unique equilibrium utilization is:

$$U^* = \begin{cases}\frac{r_o^l}{r^*-\pi(c_t)} & \text{if } r^* > \pi(c_t) \\ 1 & \text{otherwise}\end{cases} \tag{14}$$

Equilibrium dynamics. In order to achieve the equilibrium point, The protocol first should find the equilibrium interest rate, r^*, and set $r_t = r^*$ to prevent borrowers from leaving or joining the system. Once r^* is set, if the utilization U_t is above U^*, lenders' utility is positive, so they keep lending until U_t reaches U^*, stabilizing the system. Conversely, if U_t is below U^*, lenders' utility is negative, causing them to withdraw until $U_t = U^*$. At this point, lenders are indifferent between the pool and external competitors and remain fixed. The equilibrium utilization is $U^* = 1$ if, regardless of utilization, lenders' utility under $r_t = r^*$ remains non-positive. Consequently, they withdraw fully, yielding $U^* = 1$, and they remain trapped in an unfavorable equilibrium where they prefer external rates but since their fund is being borrowed, they cannot leave the system.

2.6 Protocol objectives and evaluation metrics

In this section, we define three key metrics to evaluate a borrow-lending protocol. The protocols considered follow the behavior outlined in 2.3. Each protocol selects a deterministic or randomized interest rate function $r_t : \mathcal{H}_1^t \to \mathbb{R}^+$, a collateral factor function $c_t : \mathcal{H}_1^t \to [0,1]$ and a liquidation threshold function $LI_t : \mathcal{H}_1^t \to [0,1]$ mapping the pool's history to an interest rate, a collateral factor and a liquidation threshold.

2.6.1 Rate of equilibrium convergence

Achieving market equilibrium is crucial for any DeFi application, as equilibrium points represent the market's most competitive state. At equilibrium, no participant is overpaid or underpaid, ensuring all users are equally satisfied with $\mathcal{P}$ as with any external alternative. In two-sided markets like borrow-lending, the time to achieve stability can disproportionately benefit one side. Setting interest rates below what borrowers are willing to pay results in lenders receiving less than in a competitive market. Conversely, if interest rates exceed the equilibrium rate, borrowers pay more than in a stable market. The more elastic user benefits at the expense of the less elastic user, who incurs an impermanent loss.

Adapting to market changes allows the protocol to stabilize the pool when user behavior or price volatility changes. As these parameters change over time, the protocol must respond dynamically to maintain system stability within a reasonable timeframe. In the borrow-lending market framework, each time user behavior parameters (e.g., r_o^b, r_o^l, α) or price volatility (σ) change, the market stabilizes at a new r^* (refer to Theorem 4). One objective of $\mathcal{P}$ is to rapidly identify and set this new equilibrium rate. We formalize this concept as the rate of convergence, using it as a metric to evaluate our protocol against non-learning baselines.

▶ **Definition 5** (Rate of equilibrium convergence)**.** *Consider a stable borrow-lending pool with the interest rate $r_t = r^*$, where r^* is the equilibrium interest rate determined by Equation 13. At time $t+1$, user behavior models adapt to new parameters $\{\bar{\eta_l}, \bar{r}_o^l, \bar{\eta_b}, \bar{r}_o^b, \alpha\}$, and price volatility changes to $\bar{\sigma}$. Let r_τ represent the interest rate set by $\mathcal{P}$ at time $\tau > t$, and let $\bar{r}^*$ denote the new equilibrium interest rate for the updated market parameters. We define the* rate of equilibrium convergence *of $\mathcal{P}$, denoted by $\mathcal{R}_\mathcal{P}(\delta, \tau)$, as the infimum of functions $f(\delta, \tau)$, such that there exists some $\mathtt{T}(\delta)$ for which, with probability $1-\delta$,*

$$|r_\tau - \bar{r}^*| < f(\delta, \tau), \quad \forall\, \tau > \mathtt{T}(\delta),$$

for any initial and secondary set of parameters in the user model and price model. Probabilities are calculated on the randomness of the protocol and the noise in the user behavior model.

2.6.2 Equilibrium Optimality Index

Any Defi borrow-lending market aims to meet specific long-term objectives. For instance, the pool should maintain utilization at an optimal level; Because low utilization reduces capital efficiency, requiring the protocol to pay higher interest rates to lenders, and high utilization can make it difficult for lenders to withdraw and borrowers to secure loans. Moreover, expected defaults and liquidations are risk metrics that the protocol aims to control. These objectives operate on a different timescale than the interest rate adjustments discussed in the convergence rate. For instance, we care about the average utilization or expected default over a long period rather than their local values in each timeslot. To address this, we define a metric called optimality index, which evaluates the desirability of the system's equilibriums during the timeslots when the pool has reached its equilibrium, denoting the set of these timeslots by $\mathcal{T}_e$.

▶ **Definition 6** (Optimality Index). *The Optimality Index of a protocol, denoted by $OI_{\mathcal{P}}$, is defined as follows:*

$$OI_{\mathcal{P}} := \frac{1}{|\mathcal{T}_e|} \sum_{t \in \mathcal{T}_e} \mathbb{E}\left[-(U_t - U_{opt})^2 - \gamma \left(U_t\, \pi(c_t) + \lambda(c_t, LT_t)\right)\right], \tag{15}$$

The expectation is taken over the protocol's randomness and user behavior. U_{opt} and λ are constant parameters.

While the first term in the $OI_{\mathcal{P}}$ ensures that utilization is kept near a desired point, the second term acts as a regularization term, controlling the expected default and liquidation. Our definition of Optimality Index evaluates a specific notion of optimality, but it is just one of many possible definitions. Our protocol design methodology can be applied to other objective functions beyond Function 15. In this paper, we showcase our ideas for this specific objective function and discuss how to extend the methodology to other objectives.

2.6.3 Adversarial robustness

Protocols in DeFi are always susceptible to adversarial behavior that can be used to manipulate an adaptive algorithm to respond in a suboptimal manner. This behavior is usually observed when some borrower/lender agents interact with protocols outside of $\mathcal{P}$ in conjunction with $\mathcal{P}$ to achieve a profit. This can involve oracle manipulations attacks used to run away with valuable assets while providing worthless collateral, or attacks that move interest rates in the opposite direction of the equilibrium rates. We focus on the latter type of adversarial behavior in this work. Moving interest rates away from equilibrium leads to market inefficiencies. Further, undue hikes in these rates can be used to trigger unexpected liquidations for profit.

We thus propose that the susceptibility of $\mathcal{P}$ to adversarial manipulation should also be measured. To do that, we first assume that a fraction β of the population of lenders/borrowers are adversarial. Thus, $\mathcal{P}$ will face an adversarial lender/borrower for an approximately β fraction of time slots. In those time slots, the adversary can manipulate the borrow/lend reserves arbitrarily. Let T_β denote the set of time slots that the protocol faces an adversary.

We measure how susceptible a protocol is to adversarial manipulation using the following metric.

▶ **Definition 7** (Adversarial Susceptibility). *The adversarial susceptibility of a protocol, denoted by $AS_{\mathcal{P}}$, is defined as follows:*

$$AS_{\mathcal{P}} := \mathbb{E}\left[\sum_{t \in \mathcal{T}_e} r^t_{\mathcal{P}} - r^t_{\mathcal{P}|T_\beta}\right].$$

The expectation is taken over the protocol's randomness, user/adversarial behavior, where $r^t_{\mathcal{P}}$ denotes the interest rate recommended by the protocol and $r^t_{\mathcal{P}|T_\beta}$ is the same, when the indices of adversarial actions are known.

Since an adversary can manipulate the protocol arbitrarily, the above measure signifies how adept the protocol is in weeding out historical data that has been manipulated, thus ensuring a cleaner convergence to the true interest rate equilibrium.

2.7 Baseline

For the baseline, we consider protocols akin to Compound, which utilize a piecewise linear interest rate curve to ensure stability. These protocols dynamically adjust interest rates at each block according to the model:

$$R_t = \begin{cases} R_0 + \frac{U_t}{U_{\text{opt}}} R_{\text{slope1}}, & \text{if } U_t \leq U_{\text{opt}} \\ R_0 + R_{\text{slope1}} + \left(\frac{U_t - U_{\text{opt}}}{1 - U_{\text{opt}}}\right) R_{\text{slope2}}, & \text{if } U_t > U_{\text{opt}} \end{cases}$$

In contrast to our proposed approach, these platforms generally set collateral factors and other market parameters through offline simulations that attempt to forecast near-future market conditions. Parameters are selected based on simulation outcomes and are subject to decentralized governance voting. Since this phase happens in an offline and opaque manner by centralized companies, we cannot compare this aspect of their protocol with ours.

Due to the piecewise linear nature of the interest rate as a function of utilization, these protocols achieve market stability only when either borrowers or lenders, but not both, exhibit elasticity [17]. To see why this is the case, note that according to Theorem 4, if borrowers are elastic ($\eta_b > 0$), the equilibrium interest rate can be uniquely determined. And if lenders are elastic too, the equilibrium utilization is determined by Equation 14 which is not a linear function of r^*, hence a piecewise linear interest rate curve cannot satisfy the equilibrium conditions if both sides are elastic. Traditional DeFi platforms typically monitor the pool to identify the more elastic side of the market (usually borrowers) and design the curve accordingly.

3 Fast-slow thinker protocol

In this section, we outline a design for the interest rate and collateral factor function of $\mathcal{P}$ that aims to achieve both the best possible convergence rate and the optimal optimality index. The protocol has the following two components.

A least squares estimator detects market disruptions that lead to instability and learns the equilibrium interest rate from borrowers' reactions, setting it agilely.

The long-term parameter planner consists of three parts: 1) A user behavior parameter estimator, which estimates the user behaviour model parameters that are required to optimize the optimality index. 2) An optimization module, which selects the optimal collateral factor to maximize optimality index, assuming negligible expected liquidation. 3) A liquidation threshold determination module, which sets the liquidation threshold as a function of the collateral factor to ensure zero expected liquidation.

The canonical scenario we use to evaluate our protocol is the following: at the beginning of the timeslot t, one or some of the market parameters (e.g., $\sigma, r_o^l, r_o^b, \alpha$) change to a new level and remain constant for a period T_m. The protocol must adapt to these new parameters by setting the equilibrium interest rate and the optimal collateral factor that maximizes the optimality index when the pool reaches equilibrium. Refer to Figure 1 for a visual representation of the protocol and its interaction with the pool.

3.1 Online interest rate controller

We model the problem of finding r^* using the linear regression method, where the borrow/repay rate depends on the difference $r_t - r^*$ (motivating factor for borrowers) and η_b (borrower elasticity), with an added noise component; And use this model to estimate r^* adaptively. Here is the linear regression problem formulation:

$$\frac{B_{t+1}-B_t}{B_t} = \eta_b \cdot \left(\alpha\, r_o^b - r_t + \pi(c_t) + \frac{\alpha}{c_t}\,\mathbb{E}\left[\mathbb{1}_{p_{t+1}-p_t<0}\left(\frac{p_{t+1}-p_t}{p_t}\right)\right] + (1-\alpha)\cdot\mathbb{E}\left[\frac{p_{t+1}-p_t}{p_t}\right]\right) \overset{13}{=} \eta_b(r^* - r_t)$$

$$\mathbf{\Delta B} = \mathbf{P}_b^t \cdot \mathbf{\Theta}_b + \boldsymbol{\varepsilon} \tag{16}$$

where:

- $\mathbf{\Delta B} = \left[\frac{B_1-B_0}{B_0}, \frac{B_2-B_1}{B_1}, \ldots, \frac{B_{t+1}-B_t}{B_t}\right]^T$ is the vector of normalized changes.
- $\mathbf{P}_b^t$ is the matrix of pool variables that affect borrower's behaviour:

$$(\mathbf{P}_b^t)^\top = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ r_0 & r_1 & \cdots & r_t \end{bmatrix}$$

- $\mathbf{\Theta}_b = [\eta_b\, r^*\, , -\eta_b]^T$ is the parameter vector
- $\boldsymbol{\varepsilon} = [\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_t]^T$ is the noise vector.

The interest rate controller algorithm, outlined in 1, activates when ΔB_t exceeds a threshold δ, indicating $r_t \neq r^*$. The algorithm collects $\mathbf{\Delta B}$ and $\mathbf{P}_b^t$ to estimate $\hat{\mathbf{\Theta}}_b$, setting r_t as the estimated $\hat{r}^*$ according to $\hat{r}^* = -\frac{\hat{\mathbf{\Theta}}_b(0)}{\hat{\mathbf{\Theta}}_b(1)}$.

We cannot use vanilla LSE in this setting and simply output $r_t = \hat{r}^* = -\frac{\hat{\mathbf{\Theta}}_b(0)}{\hat{\mathbf{\Theta}}_b(1)}$ because feeding back the estimated $\hat{r}^*$ to the protocol may cause the rows of the $\mathbf{P}_b$ matrix to become very close to each other. This happens when the estimator outputs an r_t that has been seen before, leading to redundant data points. Consequently, the theoretical guarantee of LSE to converge to the correct $\mathbf{\Theta}_b^*$ as the number of data points increases is impaired. To mitigate this problem, with a small probability, we sample a random interest rate. Finally, after running for a few iterations and accumulating a sufficiently large number of samples, the estimator stops and outputs the estimated r^*.

Theoretical analysis

▶ **Theorem 8** (LSE convergence rate). *Assuming $\eta_b > 0$, the interest rate controller described in Algorithm 1 with stopping time τ (taking τ samples), satisfies the following:*

$$\mathcal{R}_{\mathcal{P}}(\delta, \tau) \sim \mathcal{O}\left(\frac{\log\frac{1}{\delta}}{\sqrt{\tau}}\right).$$

Moreover, if η_b is known: $\mathbb{E}[r_\tau] = r^*,\ \forall\tau$

▶ **Theorem 9** (Baseline convergence rate). *Under a piece-wise linear interest rate function*

- *In the presence of elastic borrowers and inelastic lenders, we have:*

$$\mathbb{E}[r_\tau] = r^* + D(1 - K\,\eta_b)^\tau, \qquad 0 < 1 - K\,\eta_b < 1$$
$$\mathcal{R}_{\mathcal{P}}(\delta, \tau) \sim o\left(\frac{1}{\sqrt{\delta}}\right).$$

  where K, D are constants determined from the specifications of the interest rate curve.
- *If lenders are elastic as well, the protocol never stabilizes with rate r^*.*

Our interest rate controller provides an unbiased estimation of the equilibrium interest rate, with the estimation variance decreasing over time. In contrast, the baseline algorithm is a biased estimator of the equilibrium rate, becoming unbiased only as $\tau \to \infty$; And the rate of convergence of the average error to zero is proportional to η_b, as the baseline protocol relies heavily on user actions to adjust the rate toward equilibrium rather than actively learning from user behavior. Additionally, the baseline algorithm maintains a constant error relative to the equilibrium rate due to the noise in user behavior and its inability to filter out this noise.

Algorithm 1 Interest rate controller, utilizing a least squares optimization approach.

Initialize: $t \leftarrow 0$, $\delta \leftarrow$ stability threshold, $t_{\text{sleep}} \leftarrow$ sleep time, ν exploration probability
while True **do**
  if $\frac{B_t - B_{t-1}}{B_{t-1}} < \delta$ **then**
    Sleep for t_{sleep}
    Reset matrices $\mathbf{\Delta B}$ and $\mathbf{P}_b$
  else
    Add the new row $[1, r_{t-1}]$ to $\mathbf{P}_b^{t-2}$ to construct $\mathbf{P}_b^{t-1}$ and
    Add the new column $[\frac{B_t - B_{t-1}}{B_{t-1}}]$ to $\mathbf{\Delta B}$
    Perform least squares estimation to find $\hat{\mathbf{\Theta}}_b \leftarrow ((\mathbf{P}_b^{t-1})^T \mathbf{P}_b^{t-1})^{-1} (\mathbf{P}_b^{t-1})^T \mathbf{\Delta B}$
    Parse $\hat{\mathbf{\Theta}}_b$ as $[\hat{\eta_b}\hat{r}^*, -\hat{\eta_b}]$ and extract $\hat{r}^*$ and set $r_t = \hat{r}^*$
    With probability ν, choose a random $r_t \in [r_{\min}, r_{\max}]$
  end if
  $t \leftarrow t + 1$
end while
end algorithm

3.2 Risk parameters planner

First, we examine the conditions that ensure zero expected liquidation. These conditions provide the planner with the necessary bounds for setting the liquidation threshold.

▶ **Lemma 10.** *The expected liquidation incurred at time $t+1$, given that the loan-to-value ratio at time t is c_t, can be expressed as*

$$\lambda(c_t, LT_t) \coloneqq E[\lambda_t^i(p_{t+1})]$$
$$= \frac{1}{1 - LT_t} \left(\Phi\left(\frac{\ln\left(\frac{c_t}{LT_t}\right) - \mu + \sigma^2}{\sigma} \right) - \frac{LT_t}{c_t} e^{\mu+\sigma^2} \Phi\left(\frac{\ln\left(\frac{c_t}{LT_t}\right) - \mu - \sigma^2}{\sigma} \right) \right)$$

where Φ denotes the cumulative distribution function of the standard normal distribution.

To maintain the expected liquidation below a small threshold (nearly zero), this lemma provides bounds on LT_t and $\frac{c_t}{LT_t}$. These bounds are used by the planner to set LT_t and constrain c_t. With negligible expected liquidation, the optimality index is simplified to include only the utilization error and the default term.

▶ **Corollary 11.** *Given a fixed set of parameters $r_o^l, r_o^b, \sigma, \mu$ and assuming that $\lambda(LT_t, c_t) \approx 0$, the maximization problem of $OI_{\mathcal{P}}$ with respect to c_t can be formulated as follows:*

$$\max_{c_t} OI_{\mathcal{P}} = \min_{c_t} \left(\frac{r_o^l}{b + \frac{a}{c_t}} - U_{opt} \right)^2 + \gamma \frac{r_o^l}{b + \frac{a}{c_t}} \Phi\left(\frac{\log(c_t) - \mu}{\sigma} \right) \tag{17}$$

$$- \gamma \frac{r_o^l}{b + \frac{a}{c_t}} \frac{\exp\left(\frac{\sigma^2}{2} + \mu\right)}{c_t} \cdot \Phi\left(\frac{-\mu + \log(c_t) - \sigma^2}{\sigma} \right) \tag{18}$$

where $a \coloneqq \alpha \left(e^{\mu + \frac{\sigma^2}{2}} \frac{\Phi\left(\frac{-\mu - \sigma^2}{\sigma}\right)}{\Phi\left(\frac{-\mu}{\sigma}\right)} - 1 \right)$ *and* $b \coloneqq \alpha r_o^b + (1 - \alpha)\left(e^{\mu + \frac{\sigma^2}{2}} - 1\right)$.

We can derive this corollary by substituting the default, expected price fall, and price change terms from Lemma 3 into the definition of optimality index stated in Definition 6.

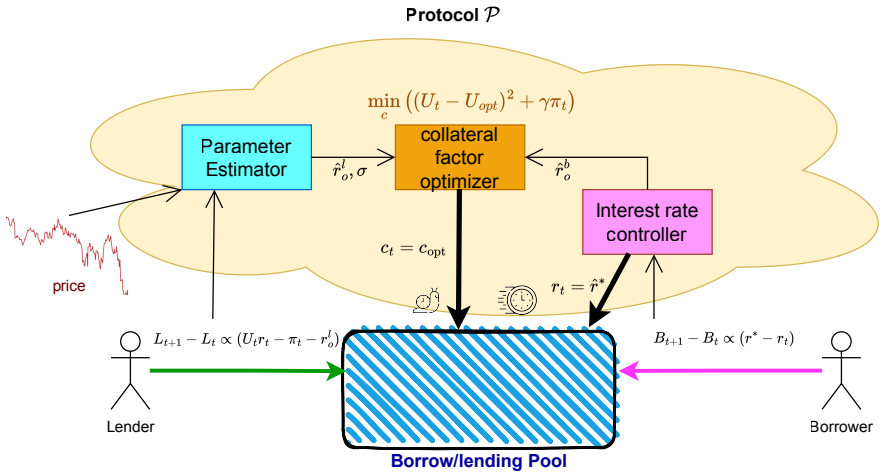


Figure 1 Protocol Overview. The interest rate controller observes borrower actions to estimate r^* and set $r_t = \hat{r}^*$. The collateral factor planner includes a parameter estimator and an optimizer: the estimator finds r_o^l, r_o^b, and σ, while the optimizer uses these estimates to determine the optimal collateral factor for the market.

The estimator module, described in Algorithm 2, uses a least squares estimator to analyze user behavior. It integrates outputs from the interest rate controller to determine the parameters r_o^l and r_o^b. Additionally, it learns the empirical price volatility σ and drift μ from the recent price history. The optimizer component, detailed in Algorithm 3, utilizes these parameters to tackle the optimization problem presented in Corollary 11 and output the optimal collateral factor. To construct the estimator submodule, we model the lender's behavior using a linear regression approach, analogous to that of the borrower's behavior.This relationship is formulated as follows:

$$\frac{L_{t+1} - L_t}{L_t} = \eta_l \cdot \left(r_t U_t - U_t \, \pi(c_t) - r_o^l \right) + \varepsilon_t \tag{19}$$

Algorithm 2 $\hat{r}_o^l, \hat{r}_o^b$ - Estimator, auxiliary to the optimizer procedure.

Initialize: $t \leftarrow 1$, $\hat{\mathbf{\Theta}}_l^0 \leftarrow \mathbf{0}$, and read L_0, U_0, r_0, c from the pool, $\delta_l \leftarrow$ stability threshold
Initialize: $t_{\text{sleep}} \leftarrow$ sleep time, $\delta_\theta \leftarrow$ least square convergence threshold, $T_{\text{optimizer}} \leftarrow$ the minimum time interval between successive executions of the optimizer. $\alpha \leftarrow$ fraction of first type borrowers
while True **do**
    if $\frac{L_t - L_{t-1}}{L_{t-1}} < \delta_l$ **then**
        Reset $\mathbf{\Delta L}, \mathbf{P}_l$
        Sleep for t_{sleep}
    else
        Calculate σ, μ empirically from the recent price history and use them to calculate $\pi(c_t)$
        Add the new row $[1, -U_{t-1}\,\pi(c_t) + r_{t-1}U_{t-1}]$ to $\mathbf{P}_l$ and $[\frac{L_t - L_{t-1}}{L_{t-1}}]$ to $\mathbf{\Delta L}$
        Perform least squares estimation to find $\hat{\mathbf{\Theta}}_l^t \leftarrow ({\mathbf{P}_l}^T \mathbf{P}_l)^{-1}\,\mathbf{P}_l^T\,\mathbf{\Delta L}$
        Parse $\hat{\mathbf{\Theta}}_l^t$ as $[-\hat{\eta}_l\,\hat{r}_o^l\,,\,\hat{\eta}_l]^T$ and extract $\hat{r}_o^l$
        Read the latest $\hat{r}^*$ from Algorithm 1, and extract $\hat{r}_o^b$ as follows:
        $$\alpha\,\hat{r}_o^b \leftarrow \hat{r}^* - \pi(c_t) - \frac{\alpha}{c_t}\left(e^{\mu+\frac{\sigma^2}{2}}\,\frac{\Phi\left(\frac{-\mu-\sigma^2}{\sigma}\right)}{\Phi\left(\frac{-\mu}{\sigma}\right)} - 1\right) - (1-\alpha)\left(e^{\mu+\frac{\sigma^2}{2}} - 1\right)$$
        if $|\hat{\mathbf{\Theta}}_l^t - \hat{\mathbf{\Theta}}_l^{t-1}| < \delta_\theta$ **then**
            $c \leftarrow \textsc{Optimizer}(\hat{r}_o^l, \hat{r}_o^b, \sigma)$
            Reset $\mathbf{\Delta L}, \mathbf{P}_l$
            sleep for $T_{\text{optimizer}}$
        end if
    end if
    $t \leftarrow t + 1$
end while
end algorithm

where $\pi(c_t)$ is defined as per the simplifications in Lemma 3. The linear regression model for this behavior is represented by:

$$\mathbf{\Delta L} = \mathbf{P}_l \cdot \mathbf{\Theta}_l + \boldsymbol{\varepsilon}, \tag{20}$$

with:

- $\mathbf{\Delta L} = \left[\frac{L_1 - L_0}{L_0}, \frac{L_2 - L_1}{L_1}, \ldots, \frac{L_{t+1} - L_t}{L_t}\right]^T$ capturing the normalized changes in lenders' supply.
- $\mathbf{P}_l = \begin{bmatrix} 1 & -U_0\,\pi(c_0) + r_0 U_0 \\ 1 & -U_1\,\pi(c_1) + r_1 U_1 \\ \vdots & \vdots \\ 1 & -U_t\,\pi(c_t) + r_t U_t \end{bmatrix}$ as the matrix of pool variables at each time step.
- $\mathbf{\Theta}_l = [-\eta_l\,r_o^l, \eta_l]^T$ representing the parameter vector.
- $\boldsymbol{\varepsilon} = [\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_t]^T$ as the vector of noise terms.

Refer to Algorithm 2 and 3 to find a detailed description of the planner.

Algorithm 3 Optimizer module.

procedure OPTIMIZER($\hat{r}_o^l, \hat{r}_o^b, \sigma$)
  Initialize: $\alpha \leftarrow$ fraction of first type borrowers, $U_{\text{opt}} \leftarrow$ desired utilization level, initial guess for collateral factor $c(0) \overset{\text{U}}{\sim} [0,1]$, $\gamma \leftarrow$ default regularization factor, $\kappa \leftarrow$ learning rate, $i \leftarrow 0$ the gradient descent iterator, $\delta \leftarrow$ gradient descent stop theshold.
  Set $a = \alpha \left(\exp\left(\mu - \frac{\sigma^2}{2}\right) - 1\right)$ and $b = \alpha\, \hat{r}_o^b + (1-\alpha)\left(\exp\left(\mu + \frac{\sigma^2}{2}\right) - 1\right)$
  $\Psi(c) \coloneqq \left(\left(\frac{\hat{r}_o^l}{b+\frac{a}{c}} - U_{\text{opt}}\right)^2 + \gamma \frac{\hat{r}_o^l}{b+\frac{a}{c}} \Phi\left(\frac{\log(c)-\mu}{\sigma}\right) - \lambda \frac{r_o^l}{b+\frac{a}{c}} \frac{\exp\left(\frac{\sigma^2}{2}+\mu\right)}{c} \cdot \Phi\left(\frac{-\mu+\log(c)-\sigma^2}{\sigma}\right)\right)$
  while $|\Psi(c(i)) - \Psi(c(i-1))| > \delta$ **do**
    $i \leftarrow i+1$
    $c(i) \leftarrow c(i-1) - \kappa \frac{d\Psi(c)}{dc}|_{c=c(i-1)}$
  end while
  return $c(i)$
end procedure

3.3 Adversarial Robustness

In this section, we employ a gradient descent-based approach to perform robust linear regression, adapting the Torrent-GD method from the robust regression literature [7]. This method leverages the resilience of gradient descent to sparse corruptions and is highly efficient for large datasets, and helps us get an improved short-term algorithm for optimizing Adversarial Susceptibility.

The Torrent-GD modification to Algorithm 1 proceeds by updating the regression coefficients iteratively, reducing the influence of corrupt data points effectively. The update rule for the regression coefficients $\hat{\mathbf{\Theta}}_b$ at each iteration t is given by:

$$\hat{\mathbf{\Theta}}_b(t+1) = \hat{\mathbf{\Theta}}_b(t) - \kappa \nabla L_S(\hat{\mathbf{\Theta}}_b(t)), \tag{21}$$

where κ is the learning rate and $\nabla L_S(\hat{\mathbf{\Theta}}_b(t))$ represents the gradient of the loss function computed only over the subset of data points S believed to be uncorrupted. This subset is dynamically determined in each iteration based on the residual errors.

The gradient of the loss function with respect to the regression coefficients is computed as:

$$\nabla L_S(w) = ({\mathbf{P}_b}^S)^T ({\mathbf{P}_b})^S \hat{\mathbf{\Theta}}_b - \mathbf{\Delta B}_S, \tag{22}$$

where ${\mathbf{P}_b}^S$ and $\mathbf{\Delta B}_S$ are the features and responses of the uncorrupted subset, respectively.

The active set S, which includes indices of the data points assumed to be uncorrupted, is updated using a hard thresholding operator that selects the points with the smallest residuals:

$$S^{(t+1)} = \text{HT}\left(r^{(t)}, k\right), \tag{23}$$

where $r^{(t)} = \mathbf{\Delta B} - \mathbf{P}_b \hat{\mathbf{\Theta}}_b(t)$ represents the residuals at iteration t, and k is a threshold parameter controlling the number of points included in S, and the hard thresholding operator HT for a vector $v \in \mathbb{R}^n$ and a threshold τ is defined as follows:

$$\text{HT}(v_i, \tau) = \begin{cases} v_i & \text{if } |v_i| \geq \tau, \\ 0 & \text{otherwise.} \end{cases}$$

This operation retains only the components of v that are greater than or equal to the threshold τ in absolute value, effectively zeroing out smaller coefficients.

The convergence of Torrent-GD is guaranteed under conditions that the noise and corruptions are sparse [7]. The algorithm can tolerate arbitrary adversarial corruptions as long as $\beta < 50\%$.

3.4 Blockchain implementation

To implement this protocol on a blockchain, we can utilize an optimistic Rollup solution like Arbitrum or Optimism. The core idea behind these Rollups is that the computation is performed off-chain by a Rollup validator. Only the state update data is posted on the blockchain, allowing challengers to validate the state update with the list of user transactions and challenge the validators in case of discrepancies. Therefore, in an optimistic scenario, no computation is performed on-chain.

Our online algorithm and the parameter estimator modules each have a computational complexity of $\mathcal{O}(W)$, where W is the sliding window used to collect and retain data for estimation. In practice, using $W \approx 50$ was sufficient in our implementation when the user's elasticity is not awfully low (An elasticity of approximately 10 is sufficient when the noise standard deviation does not exceed 1). The optimizer module, in general, can be computationally infeasible since its objective function is not necessarily convex. However, since the range of possible collateral factors is limited, we can discretize the possible values into approximately 100 levels and find the optimal collateral factor through brute force. The calculation of the objective function itself is not computationally intensive.

Gas fee estimation

Even the Roll-up solution might necessitate running the computation on-chain in case of a dispute, therefore we need to estimate the on-chain computation cost. To estimate the gas cost for updating the protocol's slow and fast parts, we consider the least squares estimator (LSE) update of Algorithm 1 and 2. This update requires approximately $12W$ multiplications and $10W$ additions, where W is the length of the history window. Given the gas costs (5 gas per multiplication and 3 gas per addition based on [16]), for $W = 50$, we require 4500 gas for multiplications and additions. Additionally, storing new rows of $\mathbf{P}_b$ and $\mathbf{\Delta B}$ costs $20,000$ gas per 32-byte storage, totaling $40,000$ gas. Thus, the overall gas needed for the LSE update and storing new rows is $44,500$ gas. The slow planner additionally needs to store a table of the optimality index values for different collateral factors and volatilities. Discretizing each into 100 and 10 values respectively, the storage cost for this table is $20,000 \times 1000 = 20,000,000$ gas. Additionally, storing a table of the CDF of Gaussian variables to calculate expected defaults and liquidations requires $20,000 \times 100 = 2,000,000$ gas for a reasonable discretization with 100 points. While more opcodes are involved in each interest rate and collateral factor update, they mainly consist of single arithmetic operations or multiple data storage, making the cost manageable.

3.5 Limitations

User behaviour model. We build upon a specific model of lender and borrower behavior, assuming that no single user controls a significant portion of the supply or demand. However, prior research has shown that this assumption might be unrealistic for many current platforms [32, 38]. In these scenarios, rational user strategies would differ from those assumed in our model. While our model manages to capture the main factors driving both sides of the

market in borrow-lending platforms, it fails to adapt to markets where a few entities control the majority of the funds. Furthermore, we assume that liquidators are always available to liquidate positions when protocols permit. However, real data indicates that this is not always the case, especially when the collateral asset lacks liquidity, resulting in considerable slippage when reselling the collateral [32].

Price distribution. Our choice of price model does not necessarily accurately describe the actual price distribution. However, as long as the price distribution belongs to a parameterized family of distributions with parameters that change slowly over time, the protocol can learn risk terms from user behavior, similar to our designed protocol. However, it may be challenging to analytically infer the price distribution parameters from user actions or to analytically relate the price distribution to key risk metrics, such as expected default or price fall.

Risk neutralily. In defining our utility functions, we assume users are risk-neutral and perceive their utility as the sum of profit minus expected risk, accurately calculating expectations over the price distribution. However, this may not be realistic. Our protocol learns the projection of user behavior onto our specific linear utility function and identifies parameters that best describe this behavior. For more complex user behavior, neural networks can be used to learn a vector representation of user behavior end-to-end. These user representation vectors are then fed to the interest rate controller and collateral factor planner, which are replaced by Deep Reinforcement Learning agents. These agents learn the optimal strategy based on feedback from their reward function. A key challenge is designing reward functions that best meet the protocol's needs.

Adversarial robustness. The adversarial resistance model we use for the LSE algorithm may be inadequate for permissionless blockchains. In such blockchains, any participant, including miners who organize and submit transactions, can act as adversaries. These adversaries can run the regression algorithm off-chain with different sets of transactions and find the set and the order that benefits them the most. Our current notion of adversarial robustness only protects against adversaries who control less than 50% of the funds and occasionally send transactions that deviate from the system's assumed supply and demand dynamics i.e., producing outlier data points.

Single equilibrium point. Our borrower behavior model assumes that there is always a single, unique interest rate that all borrowers are willing to pay, and that this equilibrium interest rate is known to all borrowers with some noise. However, in reality, this assumption may not hold true. Different groups of borrowers may have varying perceptions of the interest rate they are willing to pay. As a result, the system could have multiple equilibrium points, each attracting a different subgroup of borrowers.

4 Evaluation

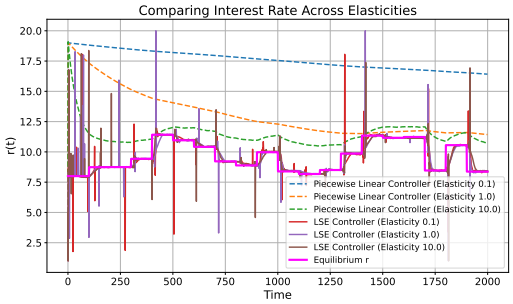
In this section, we test the interest rate controller and collateral factor planner described in section 3 and demonstrate their robustness to the change of market and user behaviours, moreover, we compare our interest rate controller with that of the baseline.

4.1 Interest rate controller

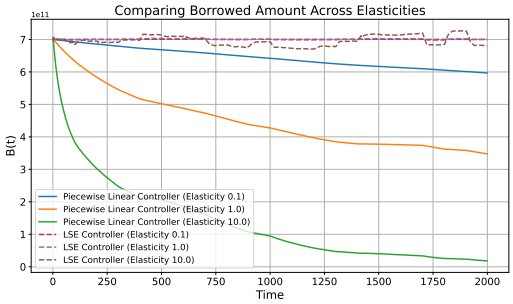
Experiment set-up. In this experiment, we start a borrow-lending pool with an initial borrow supply of 7×10^{11} and an initial lend supply of 10^{12}. Both the elasticity of borrowing and lending are set to 50, and the standard deviation of the borrowed and supply dynamic noise is 0.1. We assume very low price volatility, as it does not affect the determination of the equilibrium interest rate in this experiment. Every 100 time slots, we change r_o^b and allow the interest rate controller to adjust to the new equilibrium interest rate based on borrowers' behavior. During the exploration phases, the interest rate is randomly selected between $r_{min} = 1$ and $r_{max} = 20$ (outliers in figure 2a).

As shown in Figure 2a, the LSE-based controller adapts to the equilibrium interest rate consistently across different user elasticities with a nearly identical convergence rate, aligning with our theoretical guarantees. In contrast, the baseline controller, which sets the interest rate as a piecewise linear function of utilization, heavily relies on elasticity. It struggles to adapt to the new equilibrium interest rate in low elasticity scenarios and performs slightly better as elasticity increases.

While the performance of the baseline controller improves as borrower's elasticity increases, the consequences of misadjustment are more severe, causing significant borrower capital to leave the system when the interest rate is too high and to flood the system when it is too low. Figure 2b illustrates how the borrowed value changes with market conditions. The LSE-based controller consistently finds the stable interest rate, resulting in minimal market disruption whenever changes occur. In contrast, the baseline controller's inability to quickly find the correct rate causes substantial borrower exit from the system. This issue worsens with increased borrower elasticity due to higher repayment rates. Thus, even in high elasticity markets, the baseline controller is ineffective in preventing excessive capital inflows or outflows due to interest rate misadjustments.



(a) The LSE-based controller adapts to the new equilibrium interest rate quickly. In contrast, the piecewise linear interest rate curve fails to track the equilibrium interest rate when the borrower's elasticity is low.



(b) The LSE controller prevents the exit of borrower capital by setting a competitive interest rate, unlike the baseline controller.

Figure 2 Comparing the LSE-based interest rate controller and the piecewise linear curve.

4.2 Collateral factor planner

We conducted an experiment to evaluate the performance of our collateral factor planner. The optimality index is defined as in Definition 6 with $\lambda = 0$ and $U_{\text{opt}} = 0.5$. Initially, the system starts with an unoptimized collateral factor of $c = 0.95$. At timeslot $t = 200$, the optimizer activates and sets a new collateral factor of $c = 0.84$, which adjusts the utilization

to the desired level of 0.5 soon after that (the pace of reaching the equilibrium is a function of lenders' elasticity). At timeslot $t = 3000$, a change in price volatility disrupts the system. By timeslot $t = 3200$, the estimator accurately detects the new volatility and triggers the optimizer. The optimizer then sets a new collateral factor of $c = 0.64$, which stabilizes the utilization around 0.5 once again. This entire process is automated. The resulting utilization curve is shown in Figure 3.

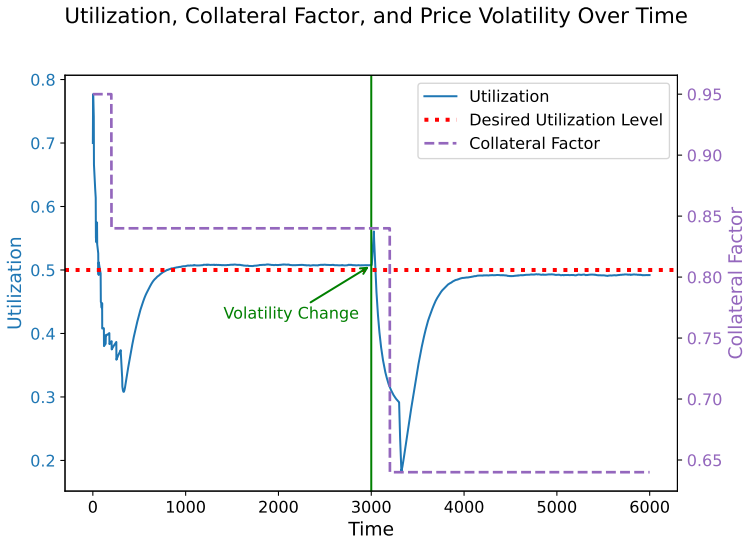


Figure 3 The collateral factor planner adapts to volatility changes and optimizes the collateral factor for achieving optimal utilization.

5 Conclusion and discussion

In this paper, we present a first-principles model of the behavior and incentives of borrowers and lenders in a DeFi market. We consider their alternative strategies and analyze how to achieve market equilibrium in the presence of price volatility. We mention empirical evidence on the validity of our model.

We propose a data-driven, borrow-lending protocol that sets the interest rate and over-collateralization ratio adaptively. By monitoring user reactions and learning from their behavior, our protocol determines a competitive interest rates and optimal collateral factors. The protocol consists of two components, 1) Fast interest rate controller: This component reacts online to user behavior, ensuring competitive interest rates and preventing over- or underpaying users. It has theoretical guarantees for fast convergence to the equilibrium interest rate. 2) Slow collateral factor planner: This component uses accurate market condition estimates to adjust the collateral factor, maintaining utilization at a desired level while controlling default risk. Overall, our protocol ensures rapid convergence to the equilibrium interest rate and optimal tuning of the collateral factor to achieve a desired equilibrium.

We implement our protocol and test its performance using simulated users, including uninformed ones. We compare our protocol with a baseline that uses piecewise linear functions to set interest rates based on utilization. Our protocol demonstrates superior performance compared to the baseline in practice.

References

1 Aave. Aave finance governance. https://app.aave.com/governance/. Accessed: 2023-05.

2 Aave. Aave whitepaper. https://github.com/aave/aave-protocol/blob/master/docs/Aave_Protocol_Whitepaper_v1_0.pdf. Accessed: 2024-02.

3 Aave. Interest rate curve in aave. https://docs.aave.com/risk/liquidity-risk/borrow-interest-rate. Accessed: 2023-05.

4 Ajna. Ajna whitepaper. https://www.ajna.finance/pdf/Ajna_Protocol_Whitepaper_01-11-2024.pdf. Accessed: 2023-05.

5 Jean Barthélemy, Clément Delaneau, Thomas Martignon, Benoît Nguyen, Marten Riho Pallum, and Salim Talout Zitan. Interest rates in decentralised finance. https://www.banque-france.fr/en/publications-and-statistics/publications/interest-rates-decentralised-finance#:~:text=The%20setting%20of%20interest%20rates%20in%20DeFI&text=The%20formula%20is%20simple%3A%20the,interest%20rates%20cannot%20be%20negative., 2023. Accessed: 2023-05.

6 Mahsa Bastankhah, Viraj Nadkarni, Xuechao Wang, Chi Jin, Sanjeev Kulkarni, and Pramod Viswanath. Thinking fast and slow: Data-driven adaptive defi borrow-lending protocol, 2024. arXiv:2407.10890.

7 Kush Bhatia, Prateek Jain, and Purushottam Kar. Robust regression via hard thresholding, 2015. arXiv:1506.02428.

8 Sylvain Carre and Franck Gabriel. Security and efficiency in defi lending. *Université Paris-Dauphine Research Paper*, 2023.

9 Tarun Chitra, Peteris Erins, and Kshitij Kulkarni. Attacks on dynamic defi interest rate curves. *arXiv preprint*, 2023. arXiv:2307.13139.

10 Jonathan Chiu, Emre Ozdenoren, Kathy Yuan, and Shengxing Zhang. On the fragility of defi lending. *Available at SSRN 4328481*, 2022.

11 Samuel N Cohen, Marc Sabate-Vidales, Lukasz Szpruch, and Mathis Gontier Delaunay. The paradox of adversarial liquidation in decentralised lending. *Available at SSRN 4540333*, 2023.

12 Samuel N Cohen, Leandro Sánchez-Betancourt, and Lukasz Szpruch. The economics of interest rate models in decentralised lending protocols. *Available at SSRN*, 2023.

13 Compound. Compound finance governance. https://compound.finance/governance. Accessed: 2023-05.

14 Compound. Compound whitepaper. https://compound.finance/documents/Compound.Whitepaper.pdf. Accessed: 2024-02.

15 Compound. Impermanent loss in defi lending. https://docs.compound.finance/interest-rates/. Accessed: 2023-05.

16 ethereum.org. Opcodes for the evm, 2024. URL: https://ethereum.org/en/developers/docs/evm/opcodes/.

17 Gauntlet. Arfc aave v3 interest rate curve recommendations from gauntlet, 2023. Accessed: 2024-05-20. URL: https://governance.aave.com/t/arfc-aave-v3-interest-rate-curve-recommendations-from-gauntlet-2023-04-27/12921.

18 Mohak Goyal, Geoffrey Ramseyer, Ashish Goel, and David Mazières. Finding the right curve: Optimal design of constant function market makers, 2023. arXiv:2212.03340.

19 Gauntlet Research Group. Aave market risk analysis. URL: https://gauntlet.network/reports/aave.

20 Gauntlet Research Group. Compound market risk analysis. URL: https://assets-global.website-files.com/648bdc0d4b8ce322f27da0af/651f146b6a1c8cc78e6c9cac_Gauntlet%20-%20Compound%20Market%20Risk%20Assessment.pdf.

21 Lewis Gudgeon, Sam Werner, Daniel Perez, and William J Knottenbelt. Defi protocols for loanable funds: Interest rates, liquidity and market efficiency. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, pages 92–112, 2020.

22 Thomas Kailath. *Linears Systems*. Prentice-Hall, 1980.

23 Will Kenton. Market distortion. https://www.investopedia.com/terms/m/marketdistortion.asp. Accessed: 2023-05.

24 Ariah Klages-Mundt and Andreea Minca. (in) stability for the blockchain: Deleveraging spirals and stablecoin attacks. *PubPub*, 2021.

25 Arthur D Kuo. A least-squares estimation approach to improving the precision of inverse dynamics computations. *ASME digital collection*, 1998.

26 Brian McWilliams, Gabriel Krummenacher, Mario Lucic, and Joachim M Buhmann. Fast and robust least squares estimation in corrupted linear models. *Advances in Neural Information Processing Systems*, 27, 2014.

27 Jason Milionis, Ciamac C. Moallemi, and Tim Roughgarden. Automated market making and arbitrage profits in the presence of fees, 2023. arXiv:2305.14604.

28 Morpho. Morpho whitepaper. https://whitepaper.morpho.xyz/, 2023. Accessed: 2023-05.

29 Viraj Nadkarni, Jiachen Hu, Ranvir Rana, Chi Jin, Sanjeev Kulkarni, and Pramod Viswanath. Zeroswap: Data-driven optimal market making in defi. *arXiv preprint*, 2023. arXiv:2310.09413.

30 Pedro M. Negron. Navigating risks in defi lending. https://medium.com/intotheblock/navigating-risks-in-defi-lending-a034dbf83b54. Accessed: 2023-05.

31 Board of governors of the federal reserve systems. Capital planning at large bank holding companies: Supervisory expectations and range of current practice. https://www.federalreserve.gov/bankinforeg/stress-tests/ccar/august-2013-estimation-methodologies-for-losses-revenues-and-expenses.htm. Accessed: 2023-05.

32 Kaihua Qin, Liyi Zhou, Pablo Gamito, Philipp Jovanovic, and Arthur Gervais. An empirical study of defi liquidations: Incentives, risks, and instabilities. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 336–350, 2021.

33 Thomas J Rivera, Fahad Saleh, and Quentin Vandeweyer. Equilibrium in a defi lending market. *Available at SSRN 4389890*, 2023.

34 Kanis Saengchote. Decentralized lending and its users: Insights from compound. *Journal of International Financial Markets, Institutions and Money*, 87:101807, 2023. doi:10.1016/j.intfin.2023.101807.

35 Lukasz Szpruch, Jiahua Xu, Marc Sabate-Vidales, and Kamel Aouane. Leveraged trading via lending platforms. *Available at SSRN*, 2024.

36 Eli Tan. Impermanent loss in defi lending. https://www.coindesk.com/learn/defi-lending-3-major-risks-to-know/. Accessed: 2023-05.

37 Yue Xing, Ruizhi Zhang, and Guang Cheng. Adversarially robust estimate and risk analysis in linear regression. In *International Conference on Artificial Intelligence and Statistics*, pages 514–522. PMLR, 2021.

38 Aviv Yaish, Maya Dotan, Kaihua Qin, Aviv Zohar, and Arthur Gervais. Suboptimality in defi. *Cryptology ePrint Archive*, 2023.

39 Xiao Zhang and Feng Ding. Adaptive parameter estimation for a general dynamical system with unknown states. *International Journal of Robust and Nonlinear Control*, 30(4):1351–1372, 2020.

40 Stephan Zheng, Alexander Trott, Sunil Srinivasa, David C. Parkes, and Richard Socher. The AI economist: Optimal economic policy design via two-level deep reinforcement learning. *CoRR*, abs/2108.02755, 2021. arXiv:2108.02755.