A Quantitative Risk Model for crvUSD

prepared by Xenophon Labs

January 2024

Disclaimer

The authors do not own `CRV` or `crvUSD` tokens, nor are they affiliated with Curve Finance or any of its affiliates. This research was funded by the Curve Research and Llama Risk teams. Any opinions and results stated here are those of the authors, not of Curve or its affiliates. Henceforth any mention of “Curve” is in reference to the Curve Protocol, unless explicitly stated otherwise.

This report is for informational purposes only, and does not constitute an offer to sell, a solicitation to buy, or a recommendation for any security. Nothing in this report should be construed as financial advice or trading advice. All information is subject to possible correction. Information may quickly become unreliable for various reasons, including changes in market conditions or economic circumstances. The material is based on information that we consider reliable, but we do not represent that it is accurate, complete and/or up to date, and it should not be relied on as such.

Contact

Prepared by Xenophon Labs.

All Inquiries: contact@xenophonlabs.com

Website: xenophonlabs.com

Changelog

v1.0	2024-01	Published by Xenophon Labs.

Table of Contents

Appendix

1

Executive Summary

Curve is an Automated Market Maker (AMM) that enables traders to exchange stablecoins and other cryptocurrencies. On May 2023, Curve deployed its own highly anticipated stablecoin, crvUSD, on Ethereum mainnet. crvUSD quickly climbed to over 100M in issued crvUSD debt, and $200M in deposited collateral. Like most stablecoins, borrowers may deposit blue-chip collateral assets (BTC, ETH) or their derivatives (wstETH, sfrxETH, tBTC), in exchange for crvUSD.

crvUSD differentiates itself from other decentralized stablecoins (such as DAI, LUSD, and FRAX) with its special-purpose AMM, dubbed the Lending-Liquidating Automated Market Maker Algorithm (`LLAMMA`). crvUSD borrowers deposit their collateral in `LLAMMA` in exchange for crvUSD. As collateral prices fluctuate, arbitrageurs will re-balance `LLAMMA` reserves by exchanging collateral for crvUSD, or vice-versa. This process, termed "soft-liquidation", allows arbitrageurs to slowly liquidate collateralized debt positions (CDPs) as collateral prices decrease by swapping collateral assets for crvUSD, and conversely allows them to de-liquidate positions by selling the collateral back to `LLAMMA` as prices climb up[1]. Through this process, arbitrageurs are incentivized to partially repay CDPs as their collateralization ratios drop, while in theory mitigating the need for full liquidations.

Like most lending platforms, crvUSD is subject to the usual market risks involved with pricing and liquidating volatile assets. For example, the protocol must ensure that market liquidity is sufficient to liquidate underwater positions under expected and worst-case asset price volatility. Unlike most lending platforms, `LLAMMA`, along with other innovations within the crvUSD protocol, also introduce new risk vectors that must be modeled and properly understood as the protocol continues to grow[2].

We have developed an Agent-Based Model (ABM) for simulating such market risks for crvUSD borrowers. Using this model, we have simulated the crvUSD system under a variety of stressed market conditions. Throughout each simulation, we track several key metrics, with added focus on the system's ability to liquidate under-collateralized positions. Taking a Monte Carlo approach, we aggregate our results over thousands of simulations and dozens of different "stress scenarios" to analyze Curve's exposure to market and liquidity risk.

The primary contribution of this project is to develop a Curve-specific ABM for simulating the crvUSD protocol, and use it to identify key risks and how they may be mitigated. In this investigative process, we hope to de-mystify certain key mechanisms in the system and provide a useful framework for the Curve DAO to understand the crvUSD protocol from a market risk perspective. In particular, we focus on de-mystifying the effect of soft liquidations on borrower profits, which we will refer to as Losses-Versus-Rebalancing (LVR) throughout this report[3], as well as providing a better understanding of crvUSD's Peg Keepers and Oracles in times of extreme asset price volatility.

[1] For details on how `LLAMMA` manages to buy volatile assets as prices go up, and sell them as prices go down (the opposite of traditional AMMs), please refer to the crvUSD whitepaper or our FAQ.

[2] For an introduction to crvUSD Risks, please refer to Curve's support page.

[3] For an introduction to LVR, please refer to *Automated Market Making and Loss-Versus-Rebalancing* by Milionis et al.

Overview of Results

We simulate the crvUSD system under a battery of stress scenarios that pose adverse yet realistic market conditions. This report summarizes the results of this initial risk exploration. We have identified extreme market volatility, stablecoin depegs, and fluctuations in crvUSD liquidity as the primary sources of risk. That said, we find that under reasonable market stress, the crvUSD system adequately incentivizes liquidators to close underwater positions and avoid potential insolvencies.

We simulate how liquidations perform as the amount of crvUSD liquidity accessible to liquidators decreases. Notice that this liquidity is largely concentrated in Curve's Peg Keeper pools, which are also the price sources for crvUSD's oracles. We have found that, if Curve fails to incentivize crvUSD deposits in Curve's Peg Keeper pools, the resulting lack of liquidity may lead to missed liquidations as well as cascading liquidations (i.e., deflationary price spirals). We suggest the protocol should aim to maintain at least 20% of crvUSD debt in its Peg Keeper pools[4]. We discuss some strategies for maintaining crvUSD liquidity in §6.2.

[4] Throughout Q4 2023, an average of 40% of crvUSD debt was deposited in Peg Keeper pools.

Furthermore, crvUSD uses special-purpose oracles that point to a small subset of high-TVL Curve pools and apply Exponential Moving Average (EMA) smoothing to minimize borrower losses due to transient price fluctuations. However, our simulations have identified a few scenarios where these underlying Curve pools may exhibit price distortions that lead to missed or excessive liquidations. The main reason these distortions may occur is if either USDC or USDT momentarily (or permanently) lose their peg. To address this concern, we consider reinstituting the Chainlink "guard-rails" in our simulations, which allow crvUSD's oracles to default to Chainlink aggregator prices when their EMA prices deviate by a preset limit.

Throughout our simulations, we also measure borrower losses from soft-liquidations. Using the LVR framework proposed by Milionis et al., we measure the Mark to Market (MTM) value extracted from `LLAMMA`s as they buy and sell assets at worse-than-market prices. We compare these LVR to the fees accrued by the `LLAMMA`s to estimate net borrower losses. We find that under moderate volatility regimes, high-leverage borrowers may suffer meaningful LVR that is not entirely offset by fee income, and which leads to unnecessary liquidations. We simulate slight increases in `LLAMMA`'s fees to address this concern, such that fees may more closely offset expected LVR without dis-incentivizing arbitrageurs from re-balancing positions.

All simulation results may be viewed using our Risk Dashboard. The underlying risk model is open-sourced under an MIT Licence in this GitHub repository.

Assumptions, Limitations, and Future Work

This is a preliminary risk modeling effort where we introduce an open-sourced, Curve-specific ABM for simulating the crvUSD protocol. All results are constrained by the modeling assumptions and limitations discussed in §7, as well as the scenario assumptions outlined in §4.

It is worth noting that the crvUSD protocol is very intricate, and relies on a combination of several complex mechanisms, including novel AMMs, health calculations, and specialized oracles. A large part of this modeling effort was to investigate and understand these intricacies, and incorporate them into a high-fidelity model. However, this complexity increases the probability of unknown unknowns that may pose a risk to the system and undermine the accuracy of our results.

Future work may involve applying the model to different market conditions and investigate the impact of changing the protocol's parameters, or modifying the logic of select smart contracts. For example, the model may be applied to investigate the impact of deploying different `Peg Keeper` and `Oracle` mechanisms.

2

Introduction

We begin with an overview of the crvUSD protocol, with an added focus on key mechanisms such as `LLAMMA`, the `Peg Keeper`, the `Oracle`s, and their relevant parameters. We then provide a definition for relevant market risks. We describe each risk vector in detail, how such risks may affect borrowers or the protocol at large, and introduce the metrics used to measure them in our simulations.

2.1 crvUSD Background

crvUSD is a stablecoin issued permissionlessly by a set of smart contracts and soft-pegged to the U.S. dollar. Users deposit collateral (such as WETH) in exchange for crvUSD tokens. The general schematic for crvUSD is depicted in Fig 1. We will briefly overview each of crvUSD's smart contracts, which we will refer to throughout the report. For more detail, please refer to the crvUSD whitepaper, our FAQ, or these articles by 0xReviews and Albert Lin.

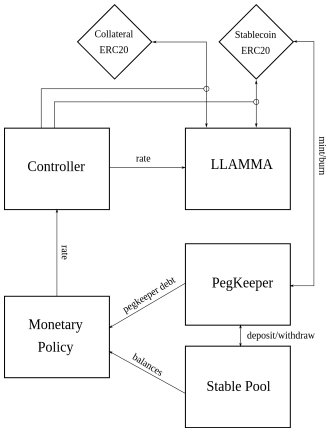

Figure 1: crvUSD Schematic, taken from the whitepaper.

2.1.1 Borrower Workflow

Users borrowing crvUSD will deposit their collateral in the `Controller` contract, which tracks the collateralized debt position (CDP) and in turn deposits the collateral in the corresponding `LLAMMA`. While in the `LLAMMA`, the user's collateral may be slowly converted to crvUSD by traders (henceforth referred to as arbitrageurs) as prices fluctuate, which may prevent the user from being liquidated but might result in losses to borrower (described later). While the CDP remains active, the user is charged an interest rate determined by the `Monetary Policy` contract.

Although purely optional, users may then take their borrowed crvUSD and deposit it into other AMMs, often Curve's `StableSwap` pools. In doing so, users earn CRV rewards directly from Curve, as well as any fees accrued to the pools. This behavior is crucial to the operation of crvUSD, as we will see with the Peg Keeping and Oracle mechanisms.

2.1.2 Peg Keeping Mechanism

There are four `StableSwap` pools[5] that are directly used by the crvUSD smart contracts to stabilize crvUSD's peg against the dollar. These `StableSwap` pools are all Curve AMMs that pair crvUSD to another stablecoin, primarily USDC and USDT. Attached to each of these pools is a `Peg Keeper` contract, whose job it is to ensure crvUSD is always priced 1:1 with its corresponding stablecoin. At any point, if crvUSD is more expensive than its paired stablecoin in one of these `StableSwap` pools, a user may update the `Peg Keeper` and mint more crvUSD into that pool. Conversely, if crvUSD is cheaper than its paired stablecoin, a user may update the `Peg Keeper` and burn off some of its existing crvUSD deposits in the pool. In either case, updates to the `Peg Keeper` will always push the exchange rate between crvUSD and the given stablecoin to 1.

There are three limits to the`Peg Keeper`'s updates. First, the `Peg Keeper` may not mint more crvUSD than its specified `debt ceiling`. Second, the`Peg Keeper` may not burn more crvUSD

[5] `StableSwap` pools refer to Curve's original AMMs, which minimize slippage for trading pegged assets using the StableSwap Invariant.

than its existing crvUSD deposited in the `StableSwap` pool. Finally, the `Peg Keeper` may only perform an `update` depending on the current aggregated crvUSD/USD price reported by the `Aggregator`, discussed in the following subsection.

Throughout this report, we will refer to these `StableSwap` pools as `Peg Keeper Pools`.

2.1.3 Oracles

crvUSD has two types of oracles, the stablecoin price `Aggregator` and collateral `Oracles`. The `Aggregator` contract aggregates prices from the `Peg Keeper Pools` to approximate a crvUSD/USD price. The `Aggregator` acts as the *de facto* crvUSD/USD oracle, and determines whether or not a `Peg Keeper` may perform an `update`. For example: if a `Peg Keeper Pool` is reporting a crvUSD price > 1, but the `Aggregator` reports a price < 1, the `Peg Keeper` will not be allowed to mint new crvUSD. This prevents a `Peg Keeper` from minting excessive crvUSD due to a price distortion in a specific `Peg Keeper Pool`. As we will see in §5, this mechanism is the key to preventing crvUSD's `Peg Keepers` from contributing to a "death spiral", even if one of its paired stablecoins (like USDC) depegs. We dive deeply into this in §5.4.

The second type of oracle is the collateral `Oracle`. The collateral `Oracle` is complex, and relies on price feeds from several Curve pools, depicted in Fig 2. In the figure, we see that the `Oracle` has three primary price feeds: prices from Curve's TriCrypto pools, prices from Curve's StableSwap pools (i.e. `Peg Keeper Pools`) and the `Aggregator` price. It combines these price feeds to produce a collateral/USD price, which is then fed into `LLAMMA`. The two primary sets of pools used by the `Oracles` are the USDC/WBTC/WETH and USDT/WBTC/WETH TriCrypto pools, and the USDC/crvUSD and USDT/crvUSD `Peg Keeper Pools`.

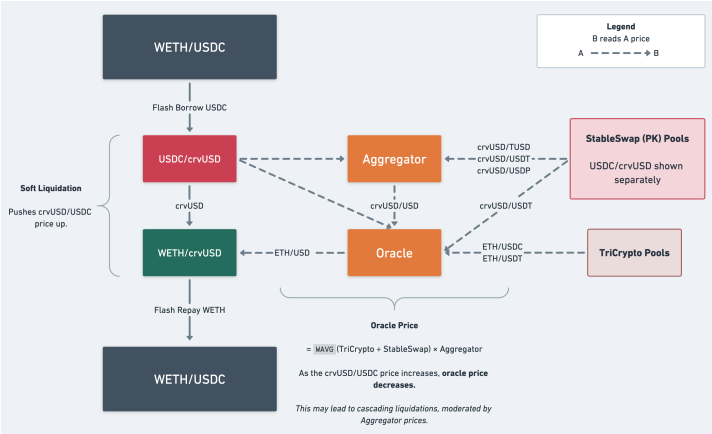

Figure 2: This figure illustrates how soft liquidations may increase the crvUSD/USDC price consumed by `LLAMMA`'s oracle, which decreases the oracle's reported ETH/USD price. This process may lead to cascading liquidations under certain conditions. The `Aggregator`,`Peg Keepers`, and EMA mechanisms all serve to mute the price impact of individual trades on the crvUSD/USDC pool and prevent cascading liquidations.

The `Oracle`'s primary purpose is to provide real-time price updates to `LLAMMA`, compared to Chainlink oracles which update less frequently (e.g. hourly or daily). Furthermore, crvUSD's oracles apply a smoothing factor to spot prices using an Exponential Moving Average (EMA), which dampens the effect of short-term fluctuations in prices. As shown in previous modeling efforts, this EMA smoothing generally leads to an improvement in borrower profits (both by reducing borrower LVR, and preventing unnecessary liquidations).[6]

[6] This EMA smoothing is actually applied in each individual Curve pool, and then fed into the `Oracle`.

2.1.4 LLAMMA

The Lending-Liquidating Automated Market Maker Algorithm (`LLAMMA`) is a special-purpose AMM where all crvUSD collateral is deposited. In that sense, all crvUSD borrowers are Liquidity Providers (LPs) in a volatile AMM. It follows that they suffer similar losses to LPs on AMMs such as Uniswap, and share in similar benefits, like fees.

`LLAMMA`, the key innovation behind crvUSD, is an oraclized AMM. This means that the price reported by `LLAMMA` depends directly on the prices it receives from its `Oracle`. As the `Oracle` price decreases, `LLAMMA`'s price actually decreases *faster*, allowing it to "invert" the traditional behavior of an AMM by setting a lower-than-market price, incentivizing arbitrageurs to buy collateral from `LLAMMA` instead of sell it. This means that as collateral prices decrease, `LLAMMA` sells more collateral, instead of the usual behavior of an AMM, where the AMM buys whichever asset decreases in market price. This relationship between `LLAMMA`'s price and the `Oracle` price is illustrated in Fig. 3. Notice that as the `Oracle` price rises, `LLAMMA`'s price rises faster.

This allows `LLAMMA` to "liquidate" positions by incentivizing arbitrageurs to purchase `LLAMMA`'s collateral in exchange for crvUSD, effectively partially repaying the debt. If prices rebound, `LLAMMA` then incentivizes arbitrageurs to "de-liquidate" positions by selling collateral back to the AMM. These swaps are referred to as "soft liquidations", which distinguishes them from "hard liquidations", where a liquidator repays the entire debt position and takes the user's collateral as a reward. Soft liquidations can be loosely interpreted as "partial and impermanent" liquidations.

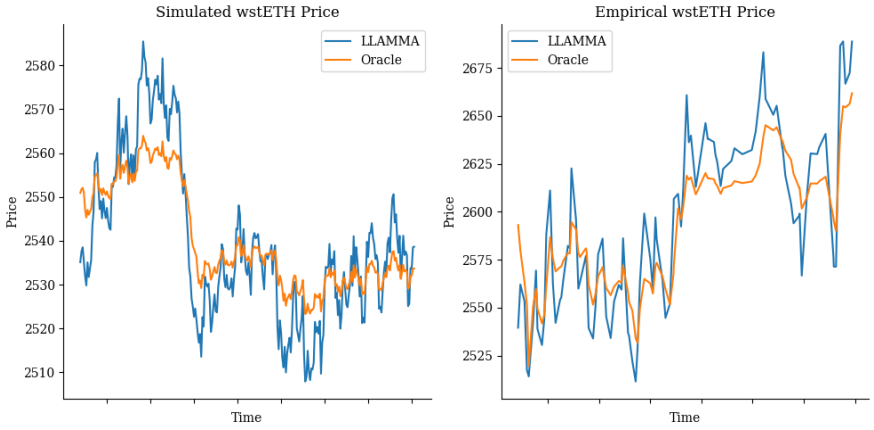

Figure 3: Left: Simulated `LLAMMA` and oracle prices for the wstETH market. **Right:** Empirical `LLAMMA` and oracle prices from the crvUSD subgraph for the wstETH market. Notice that the `LLAMMA` price moves *faster* than the oracle price in both plots, and confirms that our simulated arbitrageurs approximate the expected behavior.

`LLAMMA` is somewhat similar to a Uniswap v3 AMM, in that it is composed of several smaller AMMs called "bands" (referred to as "ticks" in Uni v3 pools)[7]. When creating their loan, users determine the number of bands they would like their collateral to be spread over. When computing borrower healths, collateral placed in bands farther below the current price is "worth less" than collateral placed in bands closer to the current price. This is intuitive: collateral may only be exchanged by arbitrageurs into crvUSD once prices reach those bands, at which point the collateral would be exchanged at a lower price. For example, if an LP places 1 WETH at the current `LLAMMA` band (say, at 2500 WETH/crvUSD price), and places 1 WETH at a lower band (say, at 2000 WETH/crvUSD price), then it is clear that the WETH at the higher band is worth "more" crvUSD from the AMM's perspective, than the WETH at the lower band. Collateral at lower bands is also "safer", in that it is farther away from potential soft liquidations (arbitrages).

[7] Refer to this article for a direct comparison between the two AMMs.

Like most AMMs, `LLAMMA` charges a fee for all swaps performed against it. This accrues entirely to `LLAMMA`'s LPs (the crvUSD borrowers) and may offset or exceed the losses incurred by the fact that the LPs are trading at worse-than-market prices.

2.1.5 Health and Hard Liquidations

Finally, we describe how crvUSD calculates loan and health and incentivizes liquidations. crvUSD's health calculations are significantly more nuanced than most lending platforms, and rely heavily on the math behind `LLAMMA`'s bonding curve. At a high level, health is calculated by aggregating the value of a user's collateral at each band. As previously mentioned, collateral in lower

bands is treated as less valuable than collateral in higher bands. The `Controller` contract calculates the user's health by summing the value of the user's collateral in each band, and then dividing it by the user's debt. Finally, a `liquidation_discount` is applied, which acts as a minimum over-collateralization ratio.

If the user's health falls below 0, then the value of the user's collateral does not sufficiently over-collateralize their outstanding debt. At this point, any user may call the `Controller`'s `liquidate` function. To perform a liquidation, the liquidator must first repay the targeted user's debt (minus whatever crvUSD the targeted user has accrued in `LLAMMA`). The liquidator then receives all of the user's collateral token as a reward. The difference between the dollar value the liquidator paid to repay the CDP and the dollar value they receive from selling the collateral is the liquidation incentive. We describe how we model this in §3.

2.2 Risk Vectors

From this overview of the crvUSD protocol we derive the key risk vectors being investigated in this report. This is not an exhaustive list of risks; it includes those risks we believed were most important in this initial modeling phase. We begin with two standard sources of risk in lending protocols, followed by three novel risk vectors resulting from crvUSD's design.

2.2.1 Missed Liquidations and Bad Debt

If prices move too quickly, or market liquidity dries up, then the protocol might experience "missed" liquidations. This occurs when liquidators are unable to profitably liquidate an underwater position. The primary reason liquidations may be missed is due to a lack of market liquidity. In this case, this lack of liquidity manifests itself in two distinct ways:

1. There is insufficient sell-side liquidity for crvUSD, preventing liquidators from being able to repay a CDP's debt.
2. There is insufficient buy-side liquidity for the collateral token, preventing liquidators from being able to sell the collateral received from liquidations and lock in a profit.

We measure the impact of missed liquidations using a metric termed "Bad Debt", which has been popularized on many lending protocols, most notably Maker. For crvUSD's case, we define "Bad Debt" as any crvUSD debt belonging to an account with sub-zero health. That is, "Bad Debt" equates to debt that should have been liquidated, but wasn't. We track the maximum amount of bad debt observed in each of our simulations as our primary risk metric, and consider the mean, median, and p99 values observed across all simulations.

As we discuss in §5, the p99 bad debt metric should be interpreted as the **worst-case daily bad debt**, as we run our simulations on a 24 hour horizon. Intuitively, excessive bad debt may lead to the protocol being under-collateralized (some refer to this as "insolvency"), which in turn threatens crvUSD's ability to hold its peg.

2.2.2 Cascading Liquidations

Liquidations rely on both purchasing the debt token (crvUSD) and selling the collateral token (e.g. WETH). In this process, the liquidator both inflates the price of the debt, and deflates the price of the collateral. Both may have a deleterious impact on oracle prices and therefore lead to additional losses and liquidations. This phenomenon is known as cascading liquidations.

As discussed in §3, cascading liquidations may be modelled by incorporating the price impact of liquidations into the simulated prices themselves. In our case, we simulate each Curve pool directly using the `curvesim` and `crvUSDsim` packages, developed by the Curve Research sub-DAO. All trades simulated against these pools will move the price of the crvUSD token, and may therefore trigger further liquidations at worse prices. This is partially addressed by the EMA

smoothing applied by the `Oracle`.

However, as discussed in §7, we model collateral prices as *exogenous* to the system. This means that, although we do capture the effect of liquidations on crvUSD price, trades involving collateral assets do not affect the future simulated market prices for those assets[8]. This is a clear limitation of our modeling approach (although currently not a major limitation given crvUSD's small size relative to its primary collateral tokens: wstETH, WETH, and WBTC), and should be addressed in future work.

[8] This is not to say we don't model the slippage incurred when exchanging collateral assets. This is done according to the methodology is §3.2.4.

2.2.3 Oracle Risk

A potential risk for crvUSD is that `LLAMMA`'s `Oracle` price deviates meaningfully from the spot price for any of its collateral tokens. This may occur for two reasons:

1. The EMA applied by the `Oracle` causes prices to go stale. By the time risky positions become eligible for liquidation, market prices may have moved past the point where the liquidations are profitable.
2. The price sources used by the `Oracle`, namely the USDC and USDT Curve pools (both TriCrypto and StableSwap pools) experience unexpected price distortions, such as a depeg.[9]

[9] There is an additional risk that the oracle price may be intentionally manipulated to trigger profitable liquidations. We do not model this behavior.

Although somewhat unexpected, we have found that (2) poses a meaningful risk to borrowers when either USDC or USDT depegs, implying that a safeguard may be put in place to protect the `Oracle` against such distortions. We measure this oracle risk using two metrics: the amount of debt liquidated, and the percentage error between the simulated market price and the reported `Oracle` price. In either case, distortions in the `Oracle` price lead to greater debt being liquidated, an increase in the error metric, and an increase in bad debt.

2.2.4 Peg Keeper Risk

The `Peg Keeper` modules are the only contracts capable of "minting" crvUSD without first depositing collateral in `LLAMMA`. A natural risk is that the `Peg Keeper` mints an infinite amount of crvUSD, leading to a collapse in its value. There are two primary mechanisms preventing this from occurring, both of which have been previously discussed in §2.1.2.

In our analysis, we have found that both mechanisms are crucial in preventing a collapse in crvUSD's value. In particular, we have identified that a depeg in any of crvUSD's `Peg Keeper` counterparts (USDC, USDT, TUSD, and USDP) may lead to the `Peg Keeper` minting excessive crvUSD. However, in the vast majority of simulations, the `Aggregator` mechanism prevents the `Peg Keeper` from minting any crvUSD at all, and in the rare cases where the `Peg Keeper` does mint excessive crvUSD, the amount it can mint is capped by a debt ceiling (controlled by the DAO). In each simulation, we track the net amount of debt minted by each `Peg Keeper`.

2.2.5 LVR Risk

Finally, we describe how borrowers may hemorrhage capital due to continuous soft liquidations. We have referred to this concept as LVR (loss-versus-rebalancing) as it is identical to the concepts described by Milionis et al. in this paper. As arbitrageurs rebalance `LLAMMA`'s reserves against the market, `LLAMMA`'s LPs end up trading at worse-than-market prices. For example, if the market price of ETH increases, `LLAMMA` will report an even higher price, at which point arbitrageurs will sell ETH to the `LLAMMA` to bring prices down. In this process, `LLAMMA`'s LPs have purchased ETH at higher-than-market prices. This is shown very clearly in Fig. 3. We formalize the LVR metric in §5.2.1.

LVR is undesirable for borrowers for two reasons: (1) it may lead to material losses depending on the exit price at which the borrower repays their loan, and (2) it may lead to a reduction

in health, and eventually a hard liquidation. LVR is both mitigated by, and (as we will discuss) potentially exacerbated by, the `Oracle` mechanism.

As shown by Milionis et al., LVR increases proportionally to asset price volatility. The counterpart to LVR is fee income: arbitrageurs also pay a fee to the AMM when executing a swap. In a perfectly competitive market, the amount paid in fees would approximately equal the amount lost through LVR. In each simulation, we track both the fee income and the LVR that accrued to all `LLAMMA`s.

We measure the difference between LVR and Fee income in our simulations to investigate whether LVR does or does not lead to unnecessary liquidations[10]. We consider how increasing `LLAMMA`'s swap fee affects the soft liquidation process, and whether it meaningfully reduces losses to borrowers.

[10] We define "unnecessary" liquidations as liquidations where the original collateral deposited by the borrower would have otherwise kept their loan healthy, but due to losses from soft-liquidations is no longer sufficient to keep their health above zero.

Of course, LVR and fees are not a complete picture for the borrower's PnL: they also pay a borrow rate, and may be "hard" liquidated. Losses from either of these is not tracked by our LVR − Fees metric.

2.3 Summary of Metrics

To address the aforementioned risks we focus on the following key metrics **for each simulation**:

1. **Bad Debt:** the maximum amount of debt that should have been liquidated, but wasn't.
2. **Liquidated Debt:** the total amount of debt that was hard liquidated at the end of the simulation.
3. **Net PK Debt:** the maximum amount of`Peg Keeper` debt minted over a simulation.
4. **LVR:** the dollar value coming out of `LLAMMA` minus the dollar value going into `LLAMMA`, accumulated over all timesteps in each simulation. This simple metric tracks the loss in value to `LLAMMA`'s LPs due to them trading at worse-than-market prices.
5. **Fee income:** the dollar value coming into `LLAMMA` from fees, accumulated over all timesteps in each simulation. For direct comparison to LVR, we also dollarize the fee income at each timestep.
6. **Oracle error:** we track the error between the reported oracle prices and the simulated market prices. This allows us to identify if and when the oracle might deviate from spot prices.

The metrics are then aggregated across all simulations. We generally consider the mean, median, and p99 values.

The Bad Debt, LVR, and Fee metrics are calculated as a percentage of the total initial debt for each simulation. This allows us to normalize results across all scenarios, which might begin with different amounts of debt in the system. It also allows us to map results onto the current state of `LLAMMA`, despite the fact that it would have slightly different debt amounts than those in the simulation.

To mark the LVR and Fee metrics to market, we use the simulated `Aggregator` price as a dollar equivalent for crvUSD.

3

Risk Model Overview

Our risk model rests on a high-fidelity representation of all crvUSD smart contracts and how simulated agents might interact with them. We simulate how 3 key agents, the Arbitrageur, Liquidator, and Keeper, interact with each crvUSD contract based on simple decision functions. We track several key metrics using our Agent-Based model and aggregate them over thousands of simulations. In this section, we overview the key components of our model.

The model combines three common modeling strategies among financial risk managers:

1. **Agent-Based Modeling:** we develop models for each of the system's key agents and how they interact with crvUSD's contracts. The agents, therefore, define how the system transitions from one state to another, largely dependent on simulated prices and liquidity.
2. **Stress Testing:** we run our simulations under varying degrees of market stress. The exact configuration for each model run is termed a "Stress Scenario", and defines how volatile prices will be, how much debt will be in the system, etc..[11]
3. **Monte Carlo:** we aggregate metrics over one thousand simulations for each scenario. We consider the mean, median, and p99 values for each metric to draw insights about the system.

At a high level, this means we are running thousands of simulations where agents are acting under varying degrees of market stress. We collect metrics about how the system performs in each simulation, aggregate them over all our model runs, and use these results to draw meaningful insights. Each simulation spans **24 hours** of price updates at a 5 minute granularity[12].

We first describe how we model the crvUSD system itself, meaning its various smart contracts. We then describe how we generate the inputs to each simulation (such as prices) and how we encode agent behaviors. A simplified model architecture is shown in Fig. 9.

3.1 Modeling crvUSD's Smart Contracts

To model crvUSD's smart contracts we use two Python packages developed by the Curve Research subDAO: `curvesim` and `crvUSDsim`. Both packages provide convenient Python classes for interacting with any Curve smart contract. For example, we may simulate exactly how an arbitrageur's trade will affect simulated crvUSD prices by simulating the underlying `LLAMMA` pool itself.

All modules are instantiated using on-chain data from Curve's subgraphs. As we will see, we then randomize and resample some of this data to eliminate any recency bias and widen the scope of our model.

The risk model is open-sourced under an MIT license here.

[11] Our Stress Testing framework is inspired by the Fed's DFAST Stress Testing guidelines for banks, which defines baseline, adverse, and severely adverse economic conditions. In our case, severely adverse economic conditions resemble volatile prices, low liquidity, and large amounts of risky debt.

[12] The coarse 5 minute granularity is a model limitation and strikes a balance between computational cost and model accuracy. Future work may address computational bottlenecks and increase the granularity of price updates.

3.2 Model Inputs

3.2.1 Prices

The primary independent variable to most financial simulations is asset prices. We simulate collateral prices using Geometric Brownian Motion (GBM) processes, and we simulate stablecoin prices using tailored mean-reverting Ornstein-Uhlenbeck stochastic processes.

In addition, we incorporate a technique known as Cholesky decomposition to simulate correlated asset prices. This is critical to modeling risk for crvUSD, whose collateral prices are either explicitly pegged to one another (as is the case with WETH, wstETH, and sfrxETH) or are highly correlated (WETH, WBTC).

Finally, we also consider a number of scenarios where we apply "jumps" to simulated prices. Jumps are sudden extreme movements in market prices, which are helpful in simulating moments of extreme price volatility, such as "Flash Crashes" or stablecoin depegs[13].

[13] A model that combines jumps and a diffusion process (such as GBMs or OUs) is termed a Merton Jump-Diffusion Model.

All generative parameters for these processes, including GBM drifts and volatilities, OU mean-reversion speeds, and Jump sizes are approximated from empirical data over the last three years. These parameters are tabulated in §4. An example of simulated prices (without jumps) is shown in Fig. 4. Some of our analysis on asset prices is displayed in this Jupyter notebook.

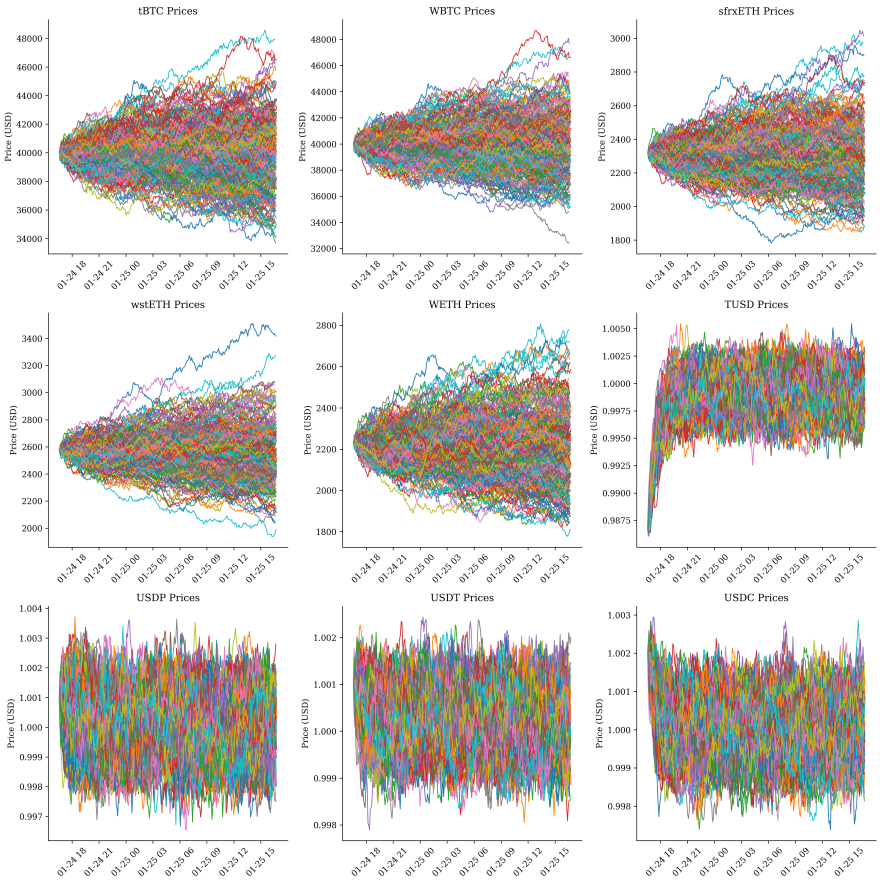

Figure 4: Snapshot of simulated prices for a zero-drift, severe volatility scenario. Notice how collateral assets behave as volatile Brownian motions, whereas stable assets mean-revert tightly around $1. The starting price for our simulations is taken from Coingecko at the time the model is run. For this particular run on January 17th, 2024, TUSD had just dipped below $0.99, which is why we see this unexpected shape on the TUSD panel.

3.2.2 Debt

In our simulations, borrowers are treated as *passive*. This is a conservative assumption: borrowers will not repay their loans proactively, increasing the burden on liquidators to keep the system solvent. Given a simulation horizon of 24 hours, this is a reasonable assumption. For longer time horizons, a more sophisticated borrower model may be developed. For this reason, we do not treat borrowers as "agents" in the system, and instead treat debt distributions as an input.

At the beginning of each simulation, we sample a random but realistic debt distribution based on historical user data from the crvUSD `Controllers`. Under the hood, we leverage a simple Gaussian Kernel Density Estimator (KDE) to sample realistic debt distributions (both in terms of leverage and number of bands). We sequentially create new loans until a target amount of debt has been reached for the simulation (which we vary for different scenarios). This allows us to both control the amount of simulated debt in the system, and ensures we capture as many realistic distributions for the specified debt as possible. The Kernel for the wstETH market is pictured in Fig. 5.[14]

[14] More specifically, we create a Gaussian KDE on the log of collateral and debt amounts, as well as the number of bands. As shown in Fig. 5, collateral (and debt) are log-normally distributed.

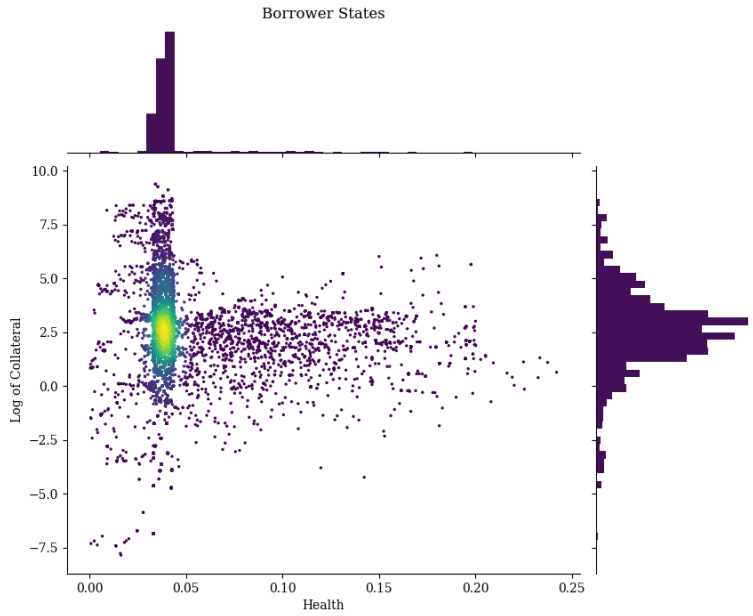

Figure 5: Empirical borrower health and collateral distribution throughout Q4 2023 for the wstETH market taken from on-chain data. Simulated debt is sampled from the Gaussian kernel, whose density gradient is colored in the figure. Not pictured: the distribution for the number of bands, N.

Additional investigative work may be performed to better understand why borrower healths are so starkly concentrated between 0.03 and 0.045. Based on our preliminary analysis of this distribution, borrower healths on all markets are concentrated within this range.

3.2.3 Internal Liquidity

When describing the risk of missed liquidations in §2 we distinguished between crvUSD liquidity and collateral liquidity. We refer to internal liquidity as all liquidity contained in Curve pools that are being explicitly simulated as described in §3.1, and includes all crvUSD liquidity. This crvUSD liquidity is largely concentrated in the `Peg Keeper Pools`. As we will see, crvUSD liquidity is also the primary liquidity-limiting step for liquidations, and as the ratio of crvUSD debt to available crvUSD liquidity in the simulated Curve pools increases, the system risks insolvency.

We use empirical data from Q4 2023 to fix a debt to liquidity ratio in our simulations. The liquidity for each pool (e.g. the amount of USDC and crvUSD deposited in the crvUSD/USDC pool) is sampled from a multi-variate normal distribution based on empirical data. The sampled amounts are then scaled such that the target ratio of debt to liquidity is met for the simulation[15]. For example, if we use a 2:1 debt to liquidity ratio and we initialize a simulation with 100M crvUSD debt, then we will sample the liquidity in the `Peg Keeper Pools` such that they have 50M crvUSD in total deposits.

[15] Most of our analysis on crvUSD debt and liquidity is contained in these Jupyter notebooks (1, 2).

We then stress test this ratio as described in §4. The statistical data used to to analyze this ratio is summarized in Table 3, and is illustrated in Fig. 12.

3.2.4 External Liquidity

The final input to our simulations is the amount of external liquidity available for each collateral asset. We define this liquidity as a price impact curve $f(x)$, where x is the input trade size. Intuitively, $f(x)$ is a monotonically increasing function that starts at 0 for an infinitesimally small trade size, and increases asymptotically to 100% as trade size goes to infinity.

We define $f(x)$ by performing a regression on historical trading data. We gather this data using 1Inch's quotes API for each possible trading pair within the crvUSD protocol. We designed a system that requests quotes for all trading pairs and for all reasonable trade sizes to build an empirical slippage curve for relevant assets[16]. Using these quotes, we construct the curve $f(x)$.

[16] Please refer to this repo for greater details on how this is done

We perform our regression using an Isotonic Regressor, which constrains the regression problem such that the fitted trend line is non-decreasing. This ensures that $f(x)$ is strictly non-decreasing and bounded within $[0, 1]$. This regression is pictured for the USDP, WETH pair in Fig. 7, and the raw data is shown in Fig. 6.

In Fig. 7 we can see clearly where the DEX liquidity for USDP $\rightarrow$ WETH dries up (at around 2M USDP). Below this amount, our regression yields more reasonable price impacts in the $0 - 3\%$ range. This is key to modeling how much an arbitrageur or liquidator can sell of each asset, before it becomes unprofitable to do so.[17]

[17] Refer to this Jupyter notebook for an example on fetching the 1Inch quotes and fitting an Isotonic Regressor to predict price impact.

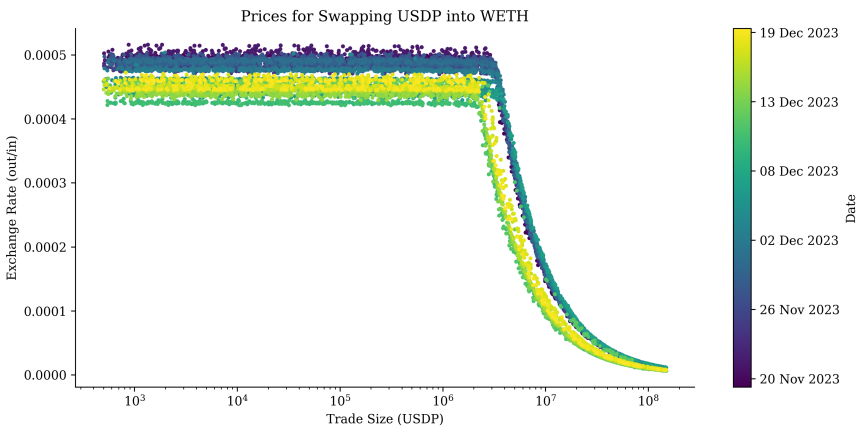

Figure 6: Historical 1Inch quotes fetched for the USDP $\rightarrow$ WETH pair. Notice how at around 2M USDP, the quoted exchange rate drops to zero, indicating that price impact is rising to 100%.

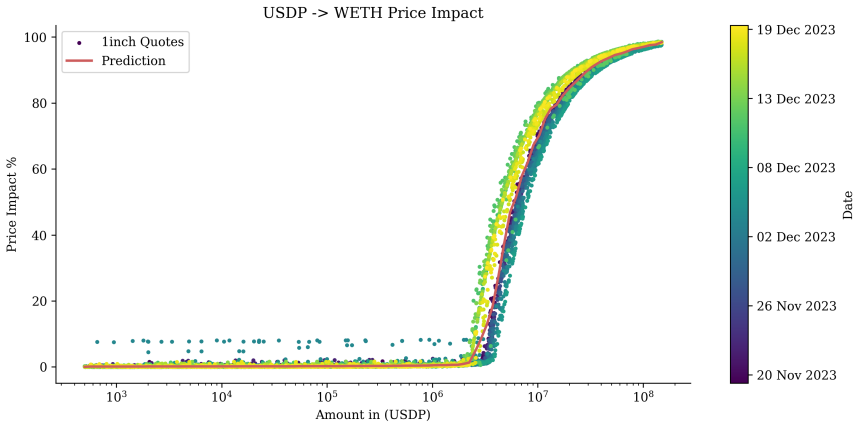

Figure 7: Fitted regression on the price impact from the 1Inch quotes for USDP → WETH pair. Notice how at around 2M USDP, the modeled price impact rises to 100%.

3.3 Agents

The three key agents in the crvUSD protocol are the Arbitrageur, who arbitrages prices across all Curve pools, the Liquidator, who hard-liquidates positions in all crvUSD markets, and the Keeper, who updates all `Peg Keepers`.

3.3.1 Arbitrageur

Modeling liquidity-limited arbitrages is key to adequately modeling risk in the crvUSD protocol. crvUSD relies on arbitrageurs profitably soft-liquidating borrowers by swapping collateral and crvUSD tokens in all `LLAMMAs`. We simulate all possible cyclic arbitrages (of length 3) within the crvUSD protocol, including all `LLAMMAs`, `Peg Keeper Pools`, and "external markets". In this case, external markets represent all external liquidity venues (including other Curve pools), which we model using the price impact curve described in §3.2.4. We illustrate an example soft liquidation in Fig. 8, which is a 3-step cyclic arbitrage.

The algorithm we use to simulate soft liquidations, as well as all other cyclic arbitrages, is as follows:

1. Instantiate all `LLAMMAs`, `Peg Keeper Pools`, and external markets. There is one external market for each pair of tokens being simulated, including collateral tokens such as WETH and stablecoins such as USDC.
2. Identify all possible cycles of length 3 between these markets using a simple depth-first search.
3. Each cycle defines a function $f(x)$ where x is the amount of token going into the first trade of the cycle, and $f(x)$ is the amount of token coming out. Since these are cycles, x and $f(x)$ are in the units of the same token. In this step, we find the x that maximizes $f(x) - x$ for each cycle.
4. Execute the most profitable cycle if it exceeds the input profitability threshold. Notice that profits are dollarized against market prices for comparison.
5. Repeat steps (3) and (4) until no profitable cycles remain (defined by some profitability threshold).

At each timestep, arbitrageurs will execute the above algorithm to ensure prices are equilibrated, as long as it is profitable to do so. We sanity-check our arbitrage algorithm by comparing LLAMMA and oracle prices in our simulations and from empirical data in Fig. 3.

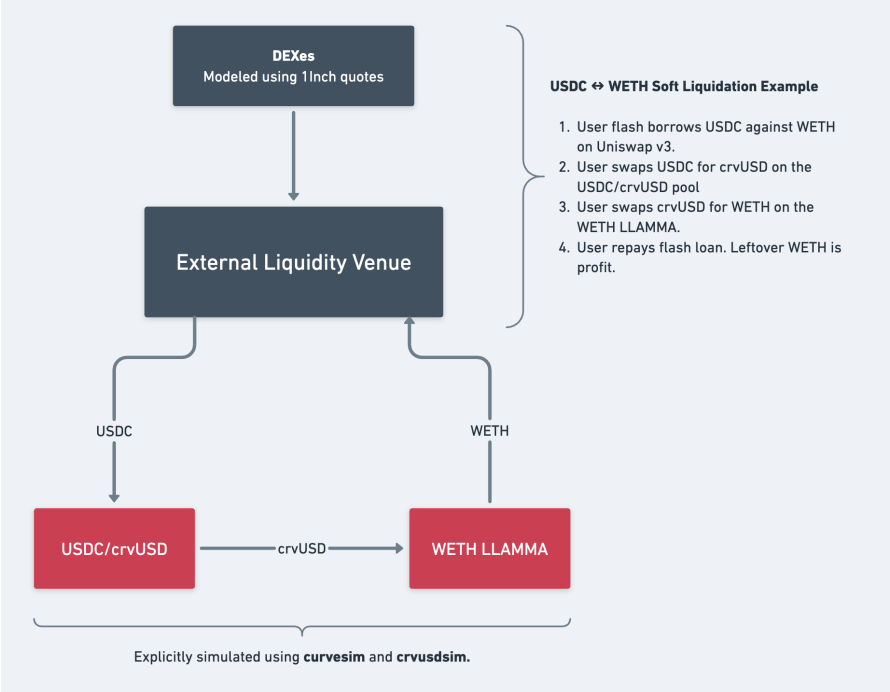

Figure 8: The archetypal cyclic arbitrage we consider in our model.

3.3.2 Liquidator

Similar to arbitrages, liquidations are also assumed to be cyclic. The liquidator agent must first purchase crvUSD from a `Peg Keeper Pool` (such as USDC/crvUSD), they then use this crvUSD to repay the borrower's loan, and finally sell the corresponding collateral in the external market for a profit. This model for liquidations resembles flash swaps, where on the last leg of the liquidation the liquidator swaps the collateral for the original stablecoin (e.g. USDC) to lock in a profit[18].

[18] Example: liquidator purchases crvUSD using USDC, repays the loan and receives WETH, swaps WETH for USDC and locks in a USDC profit.

Like arbitrageurs, our liquidator agents will only perform liquidations if the profit from closing the liquidation cycle exceeds their profit threshold. This ensures that liquidations will not occur if market prices or liquidity make it prohibitively expensive to procure the necessary crvUSD to perform the liquidation.

3.3.3 Keeper

The simplest agent in our simulations is the Keeper. The Keeper's task is to update the `Peg Keeper` whenever doing so is profitable. This is done by calling the `update` function in the `Peg Keeper` contract, which can only be done when such updates would result in a profitable deposit or withdrawal of liquidity in a `Peg Keeper Pool`.

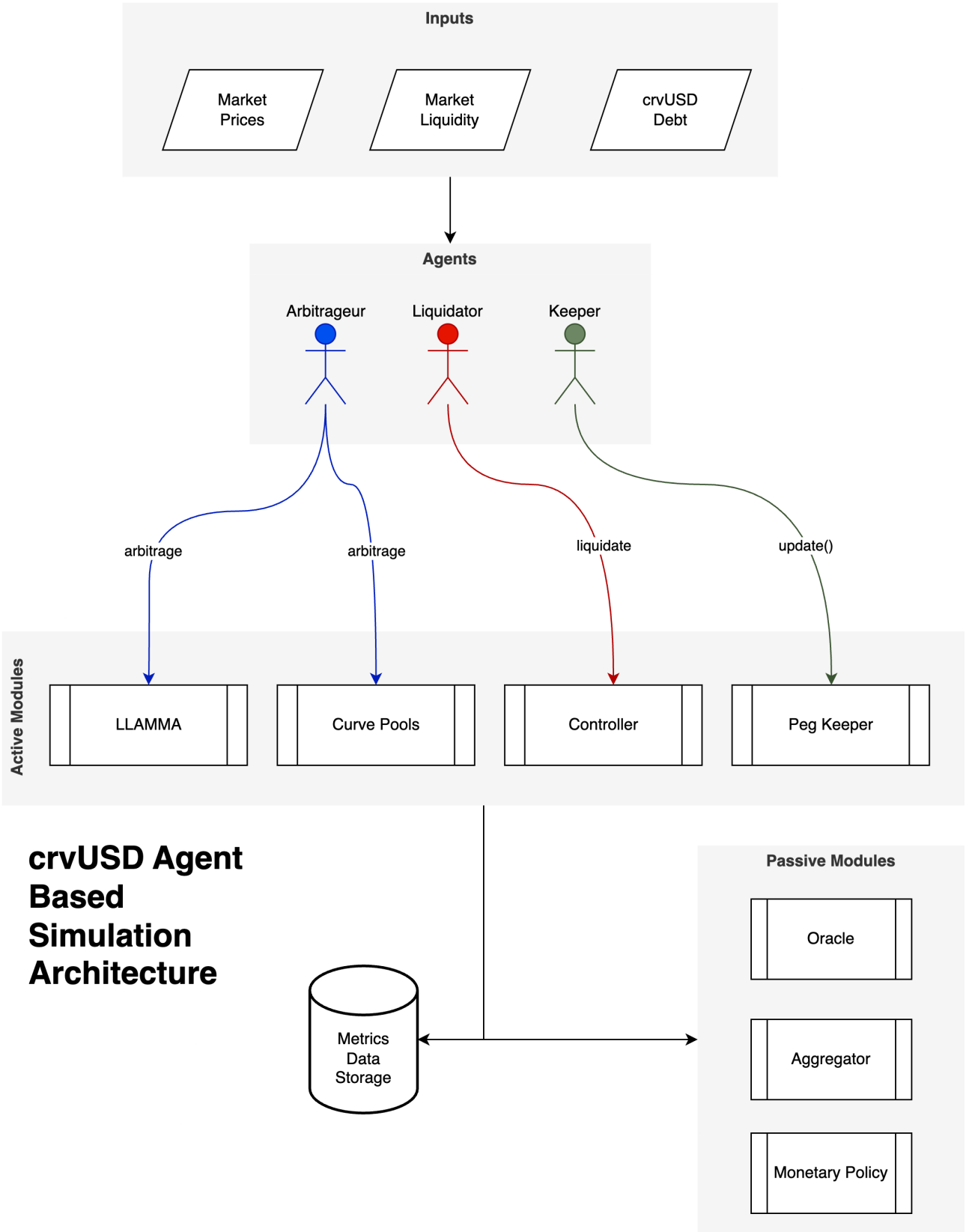

Figure 9: A simple depiction of the agents in our model architecture and their relationship to crvUSD's modules.

4

Stress Testing

Our Agent-Based Simulations are organized by stress scenarios. Each scenario aims to investigate the effect of different assumptions regarding asset prices, market liquidity, and outstanding debt, which influence agent behavior on the crvUSD system. We overview the several scenarios we have simulated with our model, the motivation behind each scenario, and how the corresponding parameters were gleaned from empirical data.

4.1 Market Stressors

We focus on three sources of market stress:

1. **Price volatility**: The expected variance in asset returns.
2. **Debt**: The amount of debt in the system at the beginning of the simulation.
3. **Liquidity**: The amount of crvUSD liquidity available for liquidations.

All our stress scenarios define some combination of these three stressors, generally organized as *baseline*, *adverse*, or *severe* conditions for each variable.

We may additionally apply a drift or jumps to market prices, however volatility is the dominant price term that we focus on throughout our analysis. For most scenarios, we enforce a zero-drift condition and no jumps, although we also consider some scenarios with very negative drift and large jumps to stress test the system.

4.1.1 Volatilities

We consider three volatility regimes gleaned from empirical data since 2020. For adverse scenarios we analyze BTC and ETH prices and assume that the parameters learned from these assets approximate the volatility and drift of their derivatives: sfrxETH, wstETH, and tBTC. We do this as there is significantly more data for BTC and ETH, and these have become less volatile since their derivatives launched a few years ago. Our volatility configuration, shown in Table 1, is based on data from Fig. 10.

	WETH	wstETH	sfrxETH	WBTC	tBTC
baseline	0.46	0.45	0.47	0.43	0.45
adverse	2.45	2.45	2.45	1.88	1.88
severe	3.67	3.67	3.67	2.82	2.82

Table 1: Annualized volatility configurations for each collateral asset for each scenario. Baseline volatilities are the average intraday volatility observed over the sampling period. Adverse volatilities represent the p99 intraday volatilities observed over the sampling period. Severe volatilities are 50% worse than adverse volatilities are represent extreme market turmoil.

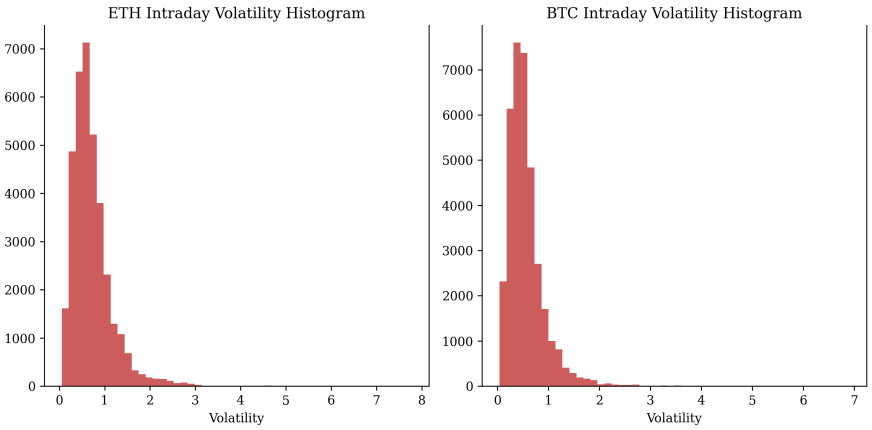

Figure 10: A histogram of intraday volatility for ETH and BTC since 2020. This is the standard deviation of log returns on a rolling, 24 hour basis, and is annualized.

4.1.2 Debts

Similar to volatility, we consider the crvUSD system under three debt conditions. As the system grows in debt, the stress on external collateral liquidity grows, increasing the probability that liquidators are unable to profitably liquidate CDPs due to the price impact they suffer when selling the appropriated collateral. The exact configurations we test in our simulations is shown in Table 2, and is based on data illustrated in Fig. 11.[19]

[19] As discussed in §7, we don't explicitly model the tBTC market due to peculiarities with its oracle, and that it is a fraction of the size of other markets.

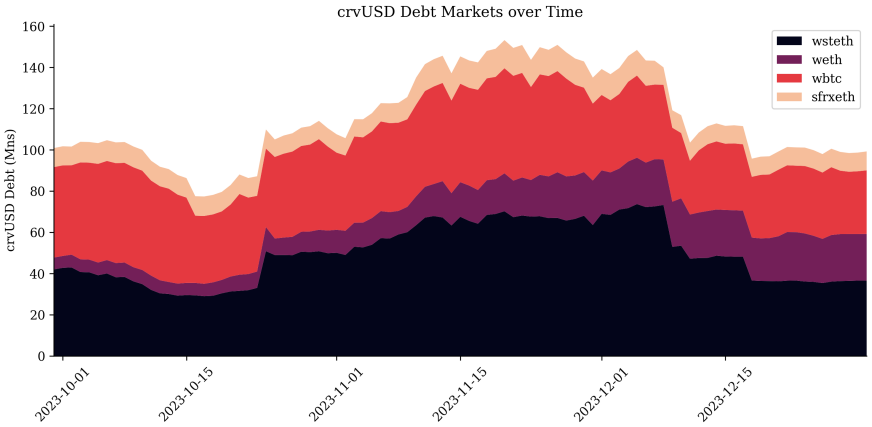

Figure 11: Empirical data on the outstanding debt for each market throughout Q4 2023.

	wstETH	WETH	WBTC	sfrxETH	Total (Mns)
baseline	0.33	0.07	0.20	0.20	115
adverse	0.50	0.12	0.28	0.30	170
severe	0.99	0.99	0.99	0.99	594

Table 2: The fraction of the debt ceiling being instantiated in each market is shown, as well as the total amount of debt assuming current debt ceilings. The baseline conditions represent the average debt ceiling utilization throughout Q4 2023, whereas the adverse conditions represent the p99 utilization.

4.1.3 Liquidities

The ratio of crvUSD debt to crvUSD liquidity in Peg Keeper Pools is relatively constant between 2:1 and 3:1. This distribution is shown in greater detail in Fig. 12. The configuration used in our stress scenarios is shown in Table 4. We include a fourth stress condition to provide further insight on the impact of crvUSD liquidity on the system's risk.

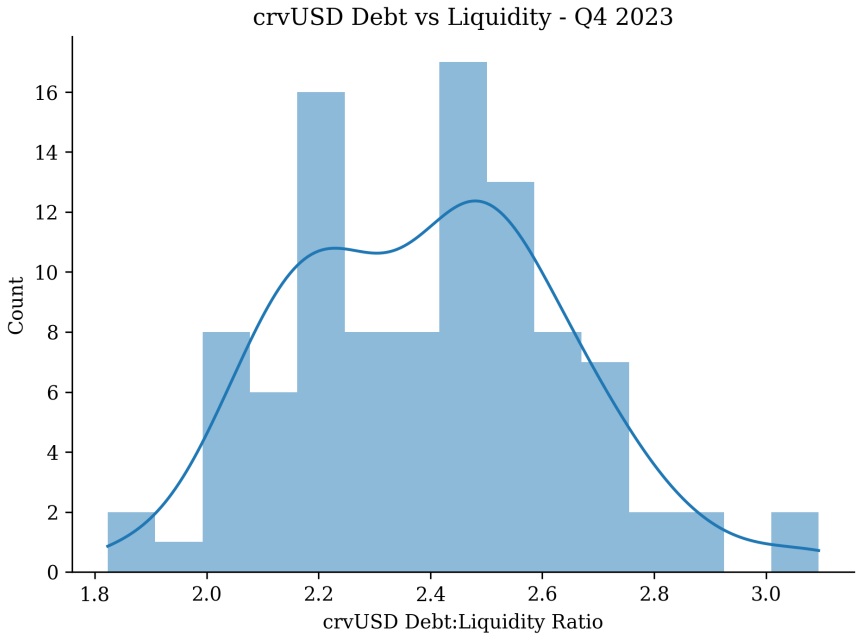

Figure 12: Empirical data on the ratio of crvUSD debt vs crvUSD liquidity in `Peg Keeper Pools` throughout Q4 2023. Notice that this ratio has been relatively tight between 2:1 and 3:1.

Statistic	Value
mean	2.36
std	0.36
min	1.55
1%	1.62
25%	2.17
50%	2.44
75%	2.61
99%	3.02
max	3.16

Table 3: Historical data on the crvUSD debt to liquidity ratio.

	Debt:Liquidity Ratio
baseline	2.36
adverse	3.50
severe	5.00
very severe	10.00

Table 4: The ratio of crvUSD debt to liquidity used for our stress scenarios. Baseline conditions represent the average ratio, and adverse conditions are slightly above the maximum ratio observed in Q4 2023. The severe and very severe conditions were chosen arbitrarily.

4.2 Scenarios

We enumerate the key scenarios used in our simulations in Table 5. Notice that we focus our analysis on the severe volatility scenario as this is when the system is most vulnerable to extreme losses and potential insolvencies. We also provide the configuration for some additional scenarios (which include jumps and negative drifts) in §B. A key scenario not shown in Table 5 is the "depeg scenario", which we introduce in §5.3.1.

	vol	debt	liquidity
baseline	baseline	baseline	baseline
adverse vol	adverse	baseline	baseline
severe vol	severe	baseline	baseline
adverse growth	baseline	adverse	baseline
severe growth	baseline	severe	baseline
adverse crvusd liquidity	baseline	baseline	adverse
severe crvusd liquidity	baseline	baseline	severe
very severe crvusd liquidity	baseline	baseline	very severe
severe vol and adverse growth	severe	adverse	baseline
severe vol and severe growth	severe	severe	baseline
severe vol and adverse crvusd liquidity	severe	baseline	adverse
severe vol and severe crvusd liquidity	severe	baseline	severe
severe vol and very severe crvusd liquidity	severe	baseline	very severe

Table 5: The key scenarios used in our simulations and their stressor configuration.

We stress that this is not an exhaustive list of stress scenarios; there are other relevant market conditions we might want to test with our model which would help identify potential vulnerabilities and improvements. An obvious example not discussed here is network congestion, where Ethereum gas prices make arbitrages and liquidations unprofitable.

However, given limited time and resources, we have selected the scenarios in Table 5 as the highest priority scenarios to simulate.

5

Results

In this section, we present an aggregate view of simulation results. We illustrate the simulated metrics, in particular Bad Debt, for several of the scenarios described in §4. We investigate how select parameter changes may meaningfully reduce risk in the system and reduce expected borrower losses. We also describe key sources of risk where the system begins to demonstrate potential insolvencies.

The raw data used throughout this section can be found in our GitHub repository, and results can be viewed interactively using our dashboard.

Note: *Throughout this section we refer to "baseline", "adverse", and "severe" conditions for volatility, debt, and liquidity. The conditions for any of these variables is "baseline", unless stated otherwise. Please refer back to the tables in §4 for details on what these conditions mean. In general, "severe" volatility implies much higher volatility than "baseline", "severe" liquidity implies less liquidity, and "severe" debt implies more debt.*

5.1 Protocol Solvency

The primary purpose of this risk assessment is to test whether the protocol can adequately incentivize liquidations, even in periods of major market stress. To that end, we ran thousands of simulations for each of the scenarios described in §4, and used the Bad Debt metric described in §2.2.1 to identify if and when simulated liquidators were unwilling to liquidate underwater positions. More specifically, the Bad Debt metric tracks the maximum amount of underwater debt observed in each simulation. We then calculate the mean, median, and p99[20] amounts of bad debt observed across all stress scenarios.

[20] p99 represents the 99th percentile of measurements. The p99 metric is critical to risk modeling. Using a p99 metric we can provide some statistical confidence that the protocol's bad debt, under the simulated conditions, will only exceed the p99 metric 1% of the time.

Notice that the p99 value is often referred to as the **Value at Risk** (VaR) metric in many similar risk models across DeFi protocols. Value at Risk approximates the worst expected outcome for a given scenario. Since we simulate prices as stochastic processes (most of them with zero drift), only a relatively small subset of model runs will simulate rapidly declining market prices. These rapid declines tend to result in the bulk of simulated losses, and constitute VaR. In this context, VaR (or p99 bad debt) should be interpreted as the **worst-case daily bad debt**, since we run our simulations on a 24 hour horizon.

Recall that we run 1000 simulations per scenario, so the p99 value refers to the 10th worst simulation per scenario.

5.1.1 Bad Debt and Volatility

Notice that bad debt increases with volatility, as expected. Furthermore, even under extreme volatility conditions we see that the p99 bad debt is kept below 1%, meaning the worst-case bad debt we expect over the course of a day is below 1% of the outstanding debt. Furthermore, the majority of simulations, even under extreme volatility, observe zero bad debt.

Volatility	Bad Debt (Median)	Bad Debt (Mean)	Bad Debt (p99)
baseline	0.00	0.00	0.00
adverse	0.00	0.01	0.18
severe	0.00	0.04	0.88

Table 6: Bad debt as a percentage of initial debt, aggregated over all simulations for each scenario. The three scenarios shown in the table have varying volatility conditions for collateral tokens.

Figure 13: Bad debt for different simulated volatilities. Under a baseline volatility regime (approximately 45% annualized volatility for all collateral assets), the protocol's risk of accruing bad debt is minimal. However, as volatility climbs up to historical highs, a growing percentage of simulations end up accruing non-negligible amounts of bad debt. Given this insight, we will focus on simulating the protocol under the *adverse* and *severe* volatility regimes to ensure crvUSD is resilient against market stress.

5.1.2 Bad Debt and crvUSD Liquidity

Although the protocol appears resilient against extreme market volatility in Table 6 and Fig. 13, this resiliency depends on the amount of crvUSD available to liquidators[21]. We run our severe volatility scenario under varying ratios of crvUSD debt to crvUSD liquidity in the `Peg Keeper Pools`. We plot the results in Fig. 14 and show aggregate values in Table 7.

[21] Recall that a ratio of 5:1 implies that 20% of crvUSD debt is deposited in the simulated `Peg Keeper Pools`

Notice that as the ratio of debt to liquidity increases, the protocol's **Value at Risk** begins to increase past 1%. This increase in the debt : liquidity ratio may occur for several reasons, including if the demand for crvUSD across other DeFi protocols increases. This is especially risky if these liquidity sinks apply some kind of lock-up period to crvUSD (such as bridges or staking pools), making the crvUSD inaccessible to liquidators.

In §6.2, we discuss how Curve can continue to incentivize crvUSD liquidity in these pools to mitigate this risk and suggest keeping the ratio below 5:1 as a general rule of thumb. Preferably, the protocol will maintain a ratio of at most 3:1 across all of its Curve pools, meaning approximately 33% of all crvUSD is deposited on Curve. It follows that as the amount of crvUSD debt grows, the amount of CRV incentives to these pools must grow accordingly.

Starting Debt:Liquidity Ratio	Bad Debt (Median)	Bad Debt (Mean)	Bad Debt (p99)
2.25	0.00	0.04	0.88
3.39	0.00	0.06	1.62
4.92	0.00	0.15	2.89
10.23	0.00	0.30	8.59

Table 7: Bad debt as a percentage of initial debt, aggregated over all simulations for each scenario. The four scenarios shown in the table have varying ratios of crvUSD debt to crvUSD liquidity. All scenarios apply a severe volatility shock.

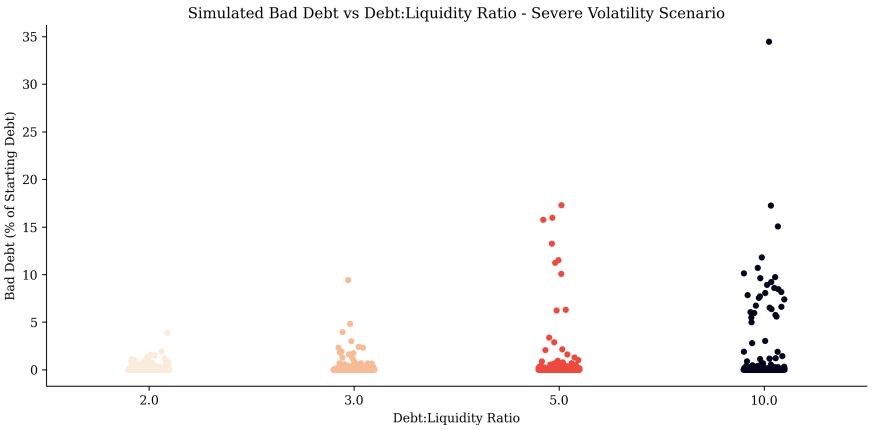

Figure 14: Bad debt for different simulated liquidity ratios. A higher ratio indicates more crvUSD debt than crvUSD liquidity available to liquidators. As we can see, at a ratio of 5:1 the bad debt metric begins to accumulate for a small percentage of runs, and at 10:1 the risk that liquidations are not properly incentivized is substantive. All scenarios apply a severe volatility shock.

5.1.3 Bad Debt and the Debt Ceiling

As discussed in §4.1.2, we also test the system at varying utilizations of the current debt ceiling. In our baseline debt scenario we set the initial debt to the average debt ceiling utilization observed in Q4 2023. For the adverse scenario we use the p99 historical utilization, and for the severe scenario we set a 99% utilization. In Table 8 we show the bad debt metric for these three scenarios under a severe volatility regime.

Notice that increasing the utilization of the debt ceiling does not meaningfully increase the risk in the system at this scale, indicating the debt ceilings have been set conservatively.

Starting Debt (Mns)	Bad Debt Pct Median	Bad Debt Pct Mean	Bad Debt Pct p99
107.38	0.00	0.04	0.88
159.27	0.00	0.05	0.62
558.25	0.01	0.06	0.82

Table 8: Bad debt as a percentage of initial debt under a severe volatility regime. Each scenario shows the bad debt as the starting debt in the simulation increases.

5.2 LVR and Fees

A key source of uncertainty around the crvUSD protocol is the potential losses resulting from "soft liquidations". As previously stated, soft liquidations are simply arbitrages against `LLAMMA`, in which `LLAMMA`'s LPs (the borrowers) end up trading at worse-than-market price. The exact price mechanics behind `LLAMMA` were illustrated in Fig. 3. However, the fees charged to the arbitrageur accrue entirely to the borrowers, and may offset or potentially exceed these LVR losses.

By measuring LVR - Fees, we aim to measure to what extent the LVR at each timestep is offset by fee income. This is key to the safety of `LLAMMA`. If LVR routinely exceeds fee income, then soft liquidations may slowly *decrease* the health of each borrower, leading to further liquidations. Therefore, appropriately setting a fee that offsets (in expectation) a majority of LVR is critical to minimizing unnecessary liquidations and preventing cascades.

5.2.1 Formalizing LVR and Fees

We can derive a simple condition for when fees would offset LVR, and use this as a framework for understanding the fee parameter. Let p_m be the market price for collateral/crvUSD (WLOG,

suppose the collateral is ETH). If p_m increases, then the price on `LLAMMA` increases faster. This leads the arbitrageur to sell collateral (ETH) to the `LLAMMA`. We can calculate LVR as:

$$\text{LVR} := x_{\text{out}} \cdot p_{m,x} - y_{\text{in}} \cdot p_{m,y}, \tag{1}$$

where x is crvUSD and y is the collateral, and $p_{m,x}$ and $p_{m,y}$ are market prices respectively. Notice that this is basically dollar value out minus dollar value in. We can also define fee income as:

$$\text{Fees} := f \cdot y_{\text{in}} \cdot p_{m,y}, \tag{2}$$

where f is the swap fee (currently 0.006). We can identify the condition necessary for LVR to equal fees:

$$\frac{x_{\text{out}}}{y_{\text{in}}} = \frac{p_{m,y}}{p_{m,x}} \cdot (1+f), \tag{3}$$

where $\frac{x_{\text{out}}}{y_{\text{in}}}$ is the trade's execution price $\hat{p}$ and $\frac{p_{m,y}}{p_{m,x}}$ is the market price p_m. Therefore, we have a simple equation for comparing LVR to fees:

$$\hat{p} \stackrel{?}{=} p_m \cdot (1+f). \tag{4}$$

This is intuitive: as the market price p_m increases, `LLAMMA`'s price $\hat{p}$ will increase faster. The LPs will therefore purchase collateral at a higher-than-market price, and this loss is only offset by fees if the fee parameter matches the difference between p_m and $\hat{p}$. For a perfectly competitive market, we would like to set:

$$f = \mathbb{E}\left[\frac{\hat{p}}{p_m}\right] - 1 \tag{5}$$

It follows that LVR - Fees, and therefore the optimal fee parameter, are functions of market volatility, as well as functions of `LLAMMA`'s parameters (which dictate how fast $\hat{p}$ rises relative to the market).

5.2.2 Simulating LVR versus Fees

We plot the mean percentage of LVR covered by fee income in Fig. 15. Under baseline volatility conditions and the current swap fee of 0.6%, we find that approximately 50% of LVR is covered by fee income on average. Notice that we do not observe a linear relationship between volatility and the percentage of LVR covered by fees. Under adverse and severe volatility conditions, between 80% and 100% of LVR is covered by fees in our simulations.

5.2.3 Increasing the Fee Parameter

As we can see in Fig. 15, the percentage of LVR covered by fees increases linearly with the swap fee regardless of volatility. Although this might be desirable for borrowers at a first glance, we must also consider how increasing swap fees would affect arbitrageur behavior, and therefore soft liquidations. If soft liquidations do not occur, then the protocol risks debt positions becoming under-collateralized, and therefore unprofitable to liquidate.

We experimented with fee parameters ranging in $[0.005, 0.006, 0.007, 0.008, 0.009, 0.1]$ and measured the LVR, fee income, and bad debt across many simulations. We find that increasing the

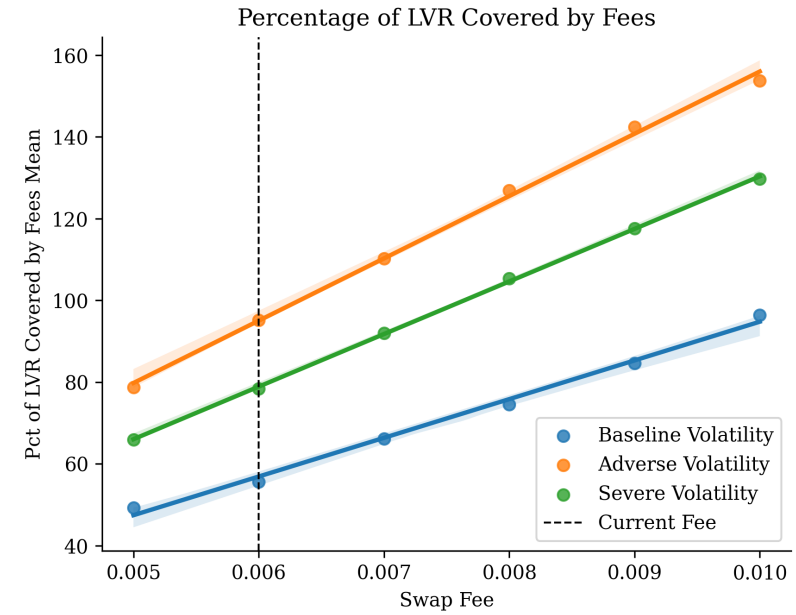

Figure 15: The percentage of simulated LVR covered by fees as the swap fee increases for varying volatility conditions.

fee income consistently improves the LVR - Fee income trade-off, as expected, and introduces no substantive increase in liquidations or bad debt for the adverse or severe volatility scenarios.

Furthermore, increasing the swap fee seems to meaningfully reduce the number of simulated liquidations in the baseline scenario, as depicted in Fig. 16. This is intuitive: at lower volatilities there is a minimal risk of liquidations to borrowers since prices do not tend to dip below their liquidation price. However, this low volatility still incurs some LVR, which may sometimes lead to liquidations. Our simulations confirm that, by increasing the swap fee and offsetting a greater percentage of LVR, many of these liquidations no longer occur.

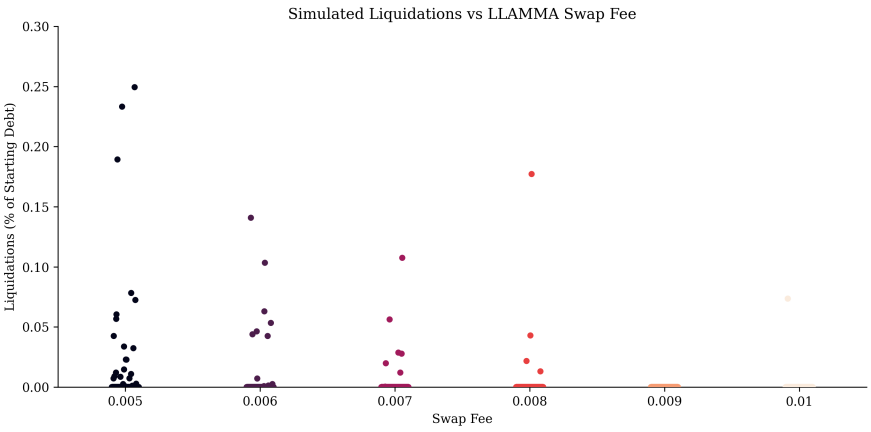

Figure 16: Simulated liquidations (as percentage of starting debt) for varying swap fees under baseline volatility conditions. Notice that the fraction of runs where users are liquidated decreases as the swap fee increases, indicating a reduction in liquidations due to LVR.

5.2.4 Active Debt

It is critical to note that the only users affected by LVR and fees are those in `LLAMMA`'s "active bands" throughout each simulation. As previously described, `LLAMMA` is a composition of several

smaller AMMs (like Uni v3). Borrowers whose collateral is deposited in bands far below the simulated market prices are therefore unlikely to participate in any trades, leading to net-zero LVR and fee income. We define the "active debt" as any debt that participated in soft liquidations throughout a given simulation.

The percentage of debt that was "active" at any point throughout a simulation increases with market volatility. We show these percentages for varying volatilities Table 9. Notice how only a small fraction of debt is involved in these trades.

vol	Active Debt Pct Mean	Active Debt Pct Median
baseline	0.88	0.44
adverse	2.48	1.37
severe	3.92	2.47

Table 9: The percentage of simulated debt that participated in any arbitrage on `LLAMMA`s.

5.2.5 LVR Disclaimer

All of these simulations assume zero drift in collateral prices. They are not representative of expected borrower returns and should not be interpreted as such. A borrower's payoff is largely a consequence of how they structure their loan, whether they enter soft liquidation, and how many bands they spread their collateral over. Furthermore, borrowers are also charged interest, and may be liquidated. We have not tracked losses from interest rates as these are largely negligible over the course of a 24 hour simulation horizon, and our LVR metric does not track positions once they are liquidated. The actual returns experienced by borrowers will be a function of fees, LVR, interest rates, hard liquidations, exit prices, and more.

Furthermore, the granularity of simulations may affect the magnitude of observed LVR or fee income, as higher granularity simulations will lead to arbitrages on smaller price swings. This is further discussed in §7.

Optimal fee setting for AMMs is an area of active and deep research. Further analysis may uncover better fee-setting strategies. However, the key conclusion from model results is that the fee parameter plays a key role, not just in borrower profitability, but in the economic risks of the crvUSD system. In a sense, the LLAMMA fee acts as the opposite of the "liquidation incentive" described in other lending protocols. Setting too wide of a fee may prevent smooth soft liquidations from keeping the system healthy, but setting too low of a fee may result in unintended reductions to borrower health.

5.3 Chainlink Limits

The oracles used in the crvUSD protocol have an option to default to Chainlink Aggregator prices if their internal EMA prices deviate from Chainlink by a preset limit[22]. The reasoning behind this is simple: `LLAMMA`'s oracles apply an EMA smoothing that may sometimes become stale if prices move very quickly in either direction. Therefore, the Chainlink limit may prevent `LLAMMA` and the Controller from using outdated prices to create and manage loans.

[22] **Example:** Suppose that `LLAMMA`'s oracle price for the WETH market is $2500 and it has a Chainlink limit of 1%. If the Chainlink Aggregator price for WETH/USD falls below $2475 or above $2525, then `LLAMMA`'s oracle will default to the Chainlink price instead of using it's oracle price of $2500.

5.3.1 The Depeg Scenario

Originally, all crvUSD oracles used a 1.5% Chainlink limit. However, existing simulations by the Curve team have shown, using granular block-by-block data, that such a tight Chainlink limit may result in unnecessary borrower losses and liquidations as the oracle tends to default to Chainlink too often[23]. For this reason, the DAO voted to remove these limits entirely from most crvUSD markets.

[23] Refer to this governance post.

However, we have identified one scenario where the Chainlink oracle limits are key to *mitigat-*

ing unnecessary borrower losses and liquidations instead of causing them. To understand why this is the case, we must first understand how these oracles work. In Fig. 2 in §2 we provided the general schematic for a crvUSD oracle. Notice that the oracle approximates a collateral/USD price by using USDC and USDT as proxies for the dollar.

It follows that if USDC or USDT are no longer reliable proxies for the dollar (i.e. if they lose their peg), then the crvUSD oracles may provide incorrect prices. The exact impact of a depeg on the `LLAMMA` oracle prices is not obvious: it depends on the relative liquidities between the TriCrypto pools and the Stableswap pools, as well as on the reflexivity between USDC and other stablecoins, and on the perceived risk of holding onto crvUSD as USDC depegs. We have attempted to model this scenario with general arbitrage algorithms detailed in §3.3.1.

We simulate the crvUSD protocol assuming that USDC momentarily depegs by 20%, and then mean-reverts back to $1. In all simulations for this scenario, the depeg happens exactly halfway through the simulation horizon. In this process, we measure how far the crvUSD oracle prices deviate from simulated collateral/USD prices. We also measure the effect the USDC depeg has on the simulated crvUSD price (i.e. the `Aggregator` price) as well as the effect on borrower losses and liquidations.

5.3.2 The Effect on Oracle Prices

In most simulations a USDC depeg causes the price of crvUSD to drop relative to other stablecoins (excluding USDC). This causes the crvUSD aggregator price to drop. The simulated crvUSD/USDC price does increase, but usually not by as much as it drops in other pools. Taken in aggregate, this effect generally causes an increase in the simulated oracle prices for all markets, shown in Fig. 17.

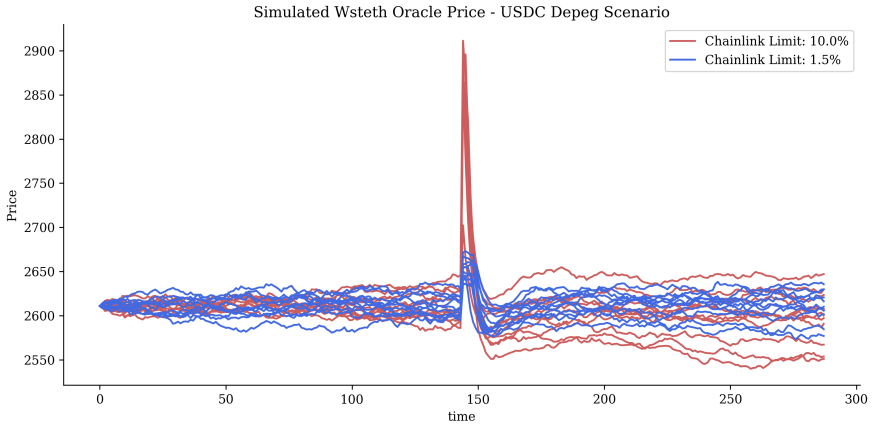

Figure 17: Simulated oracle prices for a subset of model runs. We plot the oracle price assuming a 10% Chainlink limit and a 1.5% Chainlink limit. Notice that the tighted Chainlink limit prevents the simulated oracle price from rising past $2650.

Although this sharp increase in prices may seem benign, there are situations where such a sharp increase may result in losses as prices increase and lead to some accounts being liquidated. This is discussed briefly in this help page from the Curve team. This risk is corroborated by our risk model, which shows a flurry of liquidations at the time USDC depegs on Fig. 18.

5.3.3 Mitigating Losses with Chainlink

Our results highlight the fact that there are conditions under which the `LLAMMA` oracle could report extreme price distortion due to the peg stability of USDC or USDT. This problem may be mitigated by re-instating the Chainlink limits with wider bounds, balancing the benefits of applying EMA smoothing to the Curve pool prices, and the dangers of aggregating collateral prices from such a small set of price sources. We have simulated a USDC depeg under varying

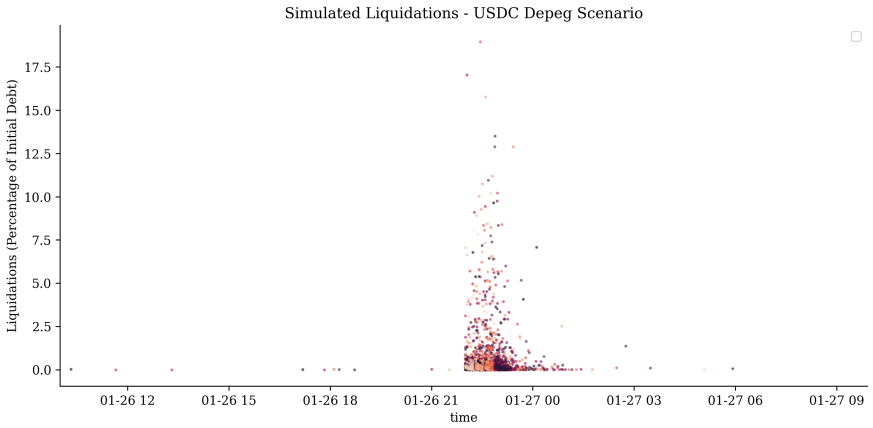

Figure 18: Simulated liquidations for the USDC depeg scenario. Colors denote the different run numbers. Notice that the depeg always occurs halfway through the simulation horizon for all runs, which coincides with the spike in liquidations observed in the graph.

Chainlink oracle limits, from the original 1.5% to 15%. The resulting simulated liquidations and bad debt are shown in Figures 19 and 20. Notice how the application of the Chainlink limits meaningfully reduces simulated liquidations and simulated bad debt.

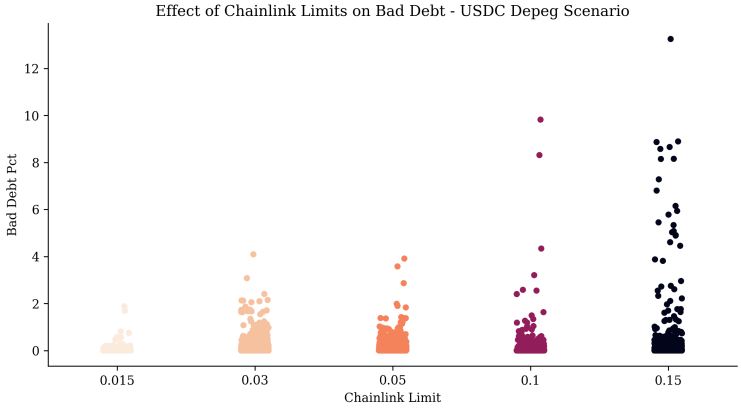

Figure 19: Simulated bad debt as a percentage of initial debt for varying Chainlink limits. Notice how simulated bad debt increases as the Chainlink limit widens.

5.3.4 EMA Smoothing

These results highlight a trade-off between the crvUSD oracles and Chainlink. The crvUSD oracles provide continuous, smooth price feeds that generally minimize borrower losses. However, due to their dependence on a small set of USDC and USDT based pools, issues with either of the stablecoins may result in drastic price swings that lead to borrower losses, liquidations, and potential bad debt.

In §6, we suggest re-instating the Chainlink limits with a higher bound. As seen in Fig. 20, a lot of the benefits can be gained from a 3% limit, which might provide a reasonable trade-off between EMA smoothing and robustness against depegs. We note that our simulations are not granular enough to fully capture the benefit of EMA smoothing on borrower losses. We discuss this further in §7.

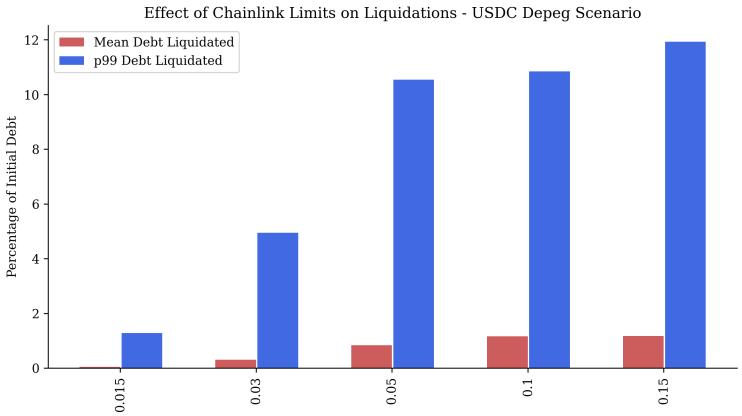

Figure 20: The mean and p99 liquidated debt for varying Chainlink limits. Notice that most of the benefit from enforcing a limit (using a 20% depeg) is achieved only when the limit is set relatively tightly.

5.4 The `Peg Keeper`s and Death Spirals

Another risk that arises if one of crvUSD's paired stablecoins depegs is that its corresponding `Peg Keeper` mints excessive unbacked crvUSD. This occurs because, from the `Peg Keeper`'s perspective, crvUSD is becoming expensive relative to the depegging stablecoin. As previously mentioned, the aggregated price across all other `Peg Keeper Pools` is the primary mechanism that tries to prevent this.

In our simulations, approximately 1% of runs end with the USDC Peg Keeper having minted close to its debt ceiling (25M crvUSD). In all simulations, the Peg Keeper slowly withdraws part or all of its deposits as USDC repegs. Notice that, if USDC does not repeg, this unbacked crvUSD may linger on in the pool indefinately, and become a source of cheap crvUSD for borrowers to repay their debt. This is, of course, undesirable to the system and undermines the crvUSD peg.

Name	Pk Debt Mean	Pk Debt Median	Pk Debt p99
Baseline	12.58	12.33	19.84
Adverse depeg	13.74	12.33	36.49

Table 10: The maximum amount of crvUSD minted by the `Peg Keeper`s throughout the simulations, in millions. In a majority of simulations, the USDC `Peg Keeper` does not mint any additional crvUSD as it is constrained by the `Aggregator` price. However, in approximately 1.1% of simulations, the USDC `Peg Keeper` mints close to its entire debt ceiling (25M crvUSD). Notice that all simulations begin with a total Peg Keeper debt of approximately 12M crvUSD, which represents the recent amount of debt in the Peg Keeper contracts when running the simulations.

In Fig. 21 we illustrate how the `Peg Keeper` begins to mint crvUSD as the `Aggregator` price rises past 1 for a select model run. This confirms a key mechanism behind crvUSD: the `Peg Keeper` can only mint excessive unbacked crvUSD if the `Aggregator` agrees that crvUSD is too expensive.

Recall that the `Aggregator` price is a liquidity-weighted average price across all `Peg Keeper Pools`. This means that the `Aggregator` will only believe that crvUSD is too expensive (in the event of a USDC depeg), if the liquidity in the USDC `Peg Keeper Pool` outweighs the liquidity in all other `Peg Keeper Pools`. As discussed in §3, we sample the relative liquidities between the `Peg Keeper Pools` randomly based on historical distributions, so there are bound to be some simulations where the initial liquidity in the USDC pool outweighs the liquidity in other pools.

However, this is fundamentally unrealistic in the following way: we do not simulate the fact that if USDC depegs, it is likely that a large portion of the liquidity in its `Peg Keeper Pool` is

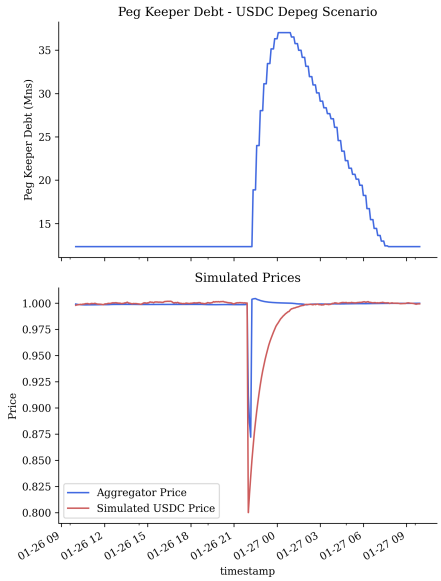

Figure 21: Top: Simulated debt for the USDC `Peg Keeper` as USDC depegs. **Bottom:** Simulated `Aggregator` price relative to the simulated USDC price. Notice that, following the USDC depeg, the aggregated price rises past 1 in accordance with the USDC/crvUSD pool. The high USDC/crvUSD price outweighs the low prices in other `Peg Keeper Pools` as it was [randomly] initialized with more liquidity in this specific simulation.

withdrawn. Therefore, we must take these results with a grain of salt. The key insight is that, as long as there is more combined liquidity in the non-depegging `Peg Keeper Pools` than in the depegging pool, the `Aggregator` is likely to prevent the `Peg Keeper` from minting excessive crvUSD. In that sense, the reflexivity in price between crvUSD and its paired stablecoins works in its favor: if USDC depegs, crvUSD becomes a vehicle to exit USDC into more stable tokens like USDT. In this process, crvUSD becomes less valuable in these other (likely more liquid) pools, resulting in a depressed `Aggregator` price and therefore no excessive `Peg Keeper` minting.

6

Discussion

In this section we summarize our interpretation of the results in §5. We discuss how a moderate increase in `LLAMMA`'s fees may reduce unnecessary liquidations without meaningfully hampering soft liquidations, and how re-instating the Chainlink oracle limits may make the protocol more resilient against depegs in USDC or USDT. Finally, we stress the importance of incentivizing liquidity in the Peg Keeper Pools. We provide a rule of thumb that at least 20% of crvUSD debt should be deposited in these pools to ensure smooth liquidations and mitigate distortions in `LLAMMA` 's oracle prices.

6.1 Potential Improvements

The suggestions below are constrained by the modeling assumptions and limitations in §7. As we eliminate more of these limitations in future work, we may become more confident and precise in recommending changes to the protocol's parameters, and consider more parameters in the process.

6.1.1 `LLAMMA` Fees

As discussed in §5.2, a key trade-off in designing the `LLAMMA` is setting a competitive swap fee. There is a lot of literature on setting competitive swap fees, and the LVR framework referenced throughout this report is but one framework for thinking about this problem.

We have argued that the swap fee is key in mitigating borrower losses from soft liquidations. However, setting a high swap fee may disincentivize arbitrageurs, widening the price difference they demand before executing soft liquidations. By widening this gap, we risk positions becoming under-collateralized before the soft liquidation is executed, which may potentially result in bad debt to the protocol.

In our simulations, we have not identified a meaningful increase in bad debt as we increase the swap fee, even in periods of substantive market volatility. However, we have identified a decrease in liquidations under baseline volatility conditions at higher swap fees, as shown in Fig. 16. The bulk of this improvement is observed as the swap fee approached 0.9%, representing a 50% increase in the current swap fee.

6.1.2 Chainlink Limits

Similar to the swap fee, the Chainlink limit parameter for `LLAMMA` 's oracles also poses a key trade-off: a tight limit reduces the system's exposure to price distortions in the underlying Curve pools, but increases its exposure to issues with the Chainlink oracles, and undermines the benefit of EMA smoothing.

We have shown in §5.3 that there are certain conditions in which setting the Chainlink limit could be beneficial to the system. In particular, a depeg in either USDC or USDT could lead to price distortions in Curve's pools that may result in borrower losses, liquidations, under-

collateralized borrowing, and potential bad debt. This depeg scenario is merely one example for why `LLAMMA` 's oracles may sometimes report prices that deviate meaningfully from the market.

In Figures 20 and 19 we plot the reduction in liquidations and bad debt (respectively) observed by setting Chainlink limits at varying bounds. We can see that the bulk of the improvement is only achieved with relatively tight bounds (1.5% and 3%), which unfortunately comes at the cost of the EMA smoothing which protects borrowers from acute price swings under baseline conditions. As we discuss in §7, our simulation granularity prevents us from capturing the benefit of EMA smoothing on borrower losses. We hope that future versions of this model are able to eliminate this limitation, and allow us to more closely quantify the trade-off in borrower losses for varying Chainlink limits.

6.1.3 Measuring Impact

Insights from risk modeling rely on the fidelity of the model and accuracy of results. Although we believe the the results represented in §5 are reliable, any parameter changes that result from such modeling efforts should be evaluated by tracking changes to real-time metrics.

6.2 crvUSD Liquidity Incentives

A key source of risk we have modeled in the crvUSD protocol is crvUSD liquidity itself. In Table 7, we showed that the maximum daily bad debt observed in our simulations becomes substantive as the debt to liquidity ratio approaches 5:1. This implies that the crvUSD protocol should ensure that at least 20% (and preferably closer to 40%) of crvUSD debt is internalized as crvUSD liquidity in Curve's pools.

We have made the conservative assumption that all of this liquidity is concentrated in the `Peg Keeper Pools`, as these are the price sources that feed into `LLAMMA`'s oracles. However, some of this liquidity is bound to be contained in other Curve pools, such as TriCRV, and would still be accessible to liquidators. Although it is preferable to keep a significant amount of crvUSD liquidity in the `Peg Keeper Pools`, the real danger to the crvUSD protocol is if a significant amount of crvUSD liquidity is diverted to protocols where they become inaccessible to liquidators. This includes any smart contract that locks up the crvUSD, including blockchain bridges, some staking contracts, and most lending platforms[24].

[24] As an example, consider this governance post regarding crvUSD on Osmosis

The primary lever the Curve community has to internalize crvUSD liquidity is the CRV gauges. Curve's gauge system allows CRV holders to vote on how much CRV each Curve pool emits as incentives to LPs. The proposed gauge weights and for Febuary 2024 are shown in Table 11.

Pool	Proposed Gauge Weight	Proposed APY Range	Annual Emissions (CRV Mn)	Annual Emissions ($ Mn)
USDT/crvUSD (`0x390f...7BF4`)	3.28	7.86% to 19.64%	5.37	2.68
USDC/crvUSD (`0x4DEc...D69E`)	2.70	5.49% to 13.71%	4.41	2.20
USDP/crvUSD (`0xCa97...53D0`)	0.55	9.05% to 22.62%	0.90	0.45
TUSD/crvUSD (`0x34D6...8db0`)	0.46	17.61% to 44.03%	0.75	0.37

Table 11: Proposed CRV gauge weights for February 1st, 2024 (as of January 29th, 2024). Weights are shown as a percentage of total CRV emissions that will be directed at each of the pools. The proposed APY range shows the range of expected APYs that LPs in those pools will earn. The annual emissions columns assume 163.4M CRV is emitted (source), and that CRV is approximately $0.5.

It follows that a potential safety mechanism for the Curve DAO is to track the total crvUSD liquidity contained across all Curve pools, and specifically in the `Peg Keeper Pools`. The DAO may agree on some initial rules of thumb for how much liquidity to internalize in Curve relative to outstanding crvUSD debt. veCRV holders may then use their votes to ensure that crvUSD pools on Curve are sufficiently incentivized to keep some crvUSD liquidity. If the debt to liq-

uidity ratio begins to fall below 30%, then veCRV holders may vote to increase the allocation to these pools. Further interest rate models may be developed to understand how much CRV must be emitted to these pools.

Curve may also choose to divert a portion of the interest paid by crvUSD borrowers to fund incentives into the `Peg Keeper Pools`[25]. The DAO may, for example, choose to use the collected interest to fund a CRV buy-back program, and use this CRV to vote for a larger share of Curve's gauges to be directed at the `Peg Keeper Pools`. Alternatively, the collected fees may be emitted directly to the `Peg Keeper Pools` using custom gauges. Either way, part of the interest rate revenue may be distributed to the crvUSD LPs, who are crucial in keeping the system safe.

[25] Thanks to Nagaking from the Curve research team for this suggestion.

Notice that the total revenue accrued by the crvUSD protocol since May 2023, shown in Fig. 22, is similar to the expected dollar emissions to the `Peg Keeper Pools` shown in Table 11.

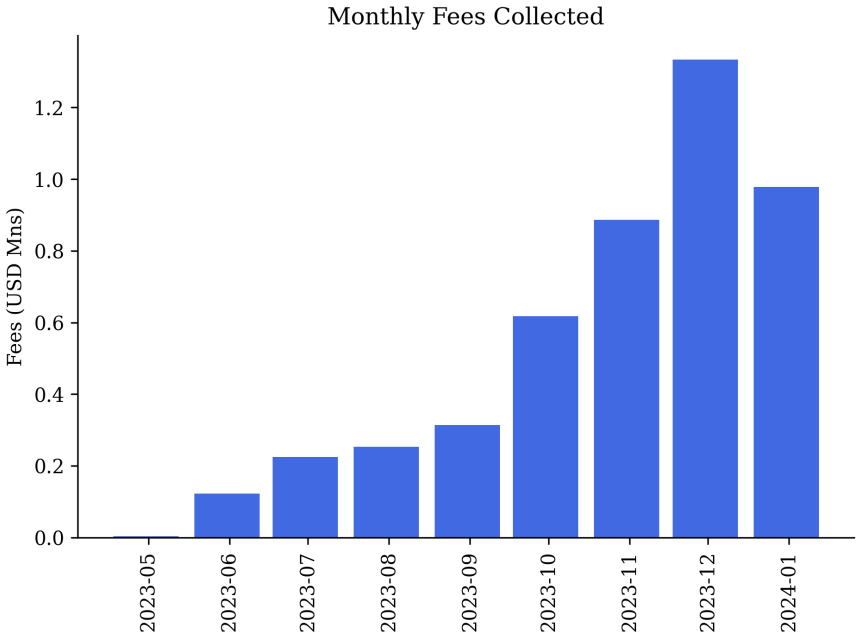

Figure 22: Monthly fees collected from interest charged to crvUSD borrowers based on data from the crvUSD subgraph. **Total: 4.6M USD**.

6.2.1 Diversification

As we discussed in §5.4, the key mechanism that prevents a "death spiral" if one of crvUSD's paired stablecoins (e.g. USDC) depegs is the `Aggregator` price. It is crucial that the `Aggregator` disagrees[26] with the price reported by the depegging pool, preventing it from minting excessive crvUSD. In the extremes, where there is only one `Peg Keeper Pool`, crvUSD would be fully exposed to depegs in the underlying stablecoin.

[26] Recall that the `Peg Keeper` can only deposit unbacked crvUSD if both its `Peg Keeper Pool` *and* the `Aggregator` agree that crvUSD is too expensive.

Currently, the vast majority of `Peg Keeper` liquidity is concentrated around the USDC and USDT pools, with the TUSD and USDP pools having mostly negligible liquidities. Based on our results, having more liquidity in a greater assortment of `Peg Keeper Pools` would make crvUSD more resilient against a depeg in any one of these corresponding stablecoins. Of course, this requires a sophisticated analysis of any new `Peg Keeper Pools` that might be added, or increased incentives into the existing, illiquid pools (TUSD, USDP).

7

Assumptions, Limitations, and Future Work

Agent-Based simulations have several limitations which can ultimately be summarized by: (1) the quality of the inputs, and (2) the flexibility of the agents. In this section, we discuss the key limitations of our modeling approach and how they affect our results. Many of these limitations may be addressed in future work and would serve to make simulation results more accurate.

This is a preliminary analysis of the crvUSD protocol. We do not consider an exhaustive set of risks or scenarios that could affect the system. We have focused on what we believed are the key sources of risk and potential improvement in the system. We list some limitations and future work below.

7.1 Time Granularity

We run our simulations using 5-minute intervals between price updates. This makes our results more conservative since liquidators and arbitrageurs have to respond to greater price swings than they would in actuality. Although more granular sims are higher fidelity, they are much more computationally expensive. We tried running simulations with varying degrees of granularity and found that simulating 5 minute intervals provided reasonable accuracy while keeping our simulation run-times manageable using a 24 hour simulation horizon. One obvious improvement to our simulation harness would be to further optimize our model and allow us to run simulations at a minutely or even per-block granularity. The computational bottleneck in our simulations is identifying optimal arbitrages and computing price impacts, described in §3.3.1 and §3.2.4 respectively.

There are some mechanisms in the crvUSD protocol, particularly around EMA smoothing factors on price feeds and time delays for certain function calls, that have slightly different behaviors if you model smooth price swings at a high granularity versus more coarse price movements at a low granularity. This has a particularly strong effect in the flash crash scenario, where we model a major price swing over a 5 minute timestep, which realistically would be distributed over many block-by-block updates. Our simulations in this scenario meaningfully over-estimate the risk in the system and should be taken with a grain of salt.

7.2 Network Costs

We ran our simulations with fixed gas costs and 0 gas costs. We found that incorporating fixed gas costs does not meaningfully change the metrics we collect from our simulations unless we assume network congestion and very high gas prices. As an extension, incorporating variable gas prices, perhaps as a function of market volatility, would improve the accuracy of our simulation results and allow us to determine crvUSD's risks relative to network congestion.

7.3 Passive Borrowers

In our simulations, borrowers are passive. In a 1-day time horizon this is a reasonable, conservative assumption. It helps answer the question: "can the system (made up of rational arbitrageurs and liquidators) smoothly clear underwater positions even if the borrower herself does not respond to changing prices?" Of course, this means the liquidation metrics (i.e. debt liquidated) are overestimated: in reality many borrowers would rather repay part or all of their position than be liquidated. Prospective crvUSD borrowers should not over-index on liquidation loss metrics, as these are largely concentrated on the riskiest debt positions, which are often actively managed and, therefore, often evade liquidations during periods of market volatility.

7.4 Passive Liquidity Providers

LPs are also passive and we use historical data to generate price impact curves. We stress test this price impact curve in our "Growth" scenarios, which meaningfully increase the amount of debt in the system and therefore stress tests the liquidator's price impact. However, a possible extension is to turn our slippage curves into multi-variate models, which consider both order size and price volatility. Higher price volatility leads to worse price impact as LPs de-risk their positions. Furthermore, higher price volatility may lead to simulated LP withdrawals from the simulated Curve pools.

7.5 Exogenous Prices

Currently we model the price impact for market selling collateral in each timestep using our Isotonic Regression models trained on historical swap quotes from 1Inch. However, we also assume that collateral prices are exogenous to the system: although trades in our model will incur price impact proportional to their size, this does not propagate over time. That is: a trade at t_0 that incurs 10% price impact, will not correspondingly affect the simulated price at time t_1; the price at time t_1 is exclusively determined by the GBM process that generates it. A more realistic model would incorporate this price impact in the simulated prices themselves, such that a 10% price impact from a trade leads to a proportional change in future simulated prices.

This limitation does not apply to trades against the simulated Curve pools (`LLAMMA`s and `Peg Keeper Pools`), since we model how those trades affect the internal state of those pools directly.

7.6 tBTC

We do not model the tBTC market due to the peculiar nature of its oracle. Unlike all other markets, tBTC's oracle points directly at a single TriCrypto pool with relatively little liquidity compared to other TriCrypto pools. Unlike other markets, this pool also includes both the collateral token (tBTC) and the debt token (crvUSD). This means liquidations (hard and soft) may be routed directly against this pool, leading to a greater risk of deflationary price spirals. Furthermore, this pool exhibits potential risks of price manipulation attacks, which are not within the scope of this report.

Notice that the tBTC market represents a small portion of the crvUSD protocol (about 4%). Future work is required to address the specific risks inherent to tBTC.

7.7 Endogenous crvUSD Liquidity

Our model assumes that all crvUSD liquidity is concentrated in `LLAMMA`s and the Peg Keeper pools. This is a conservative assumption, as it both stresses the price impact suffered by our agents and increases the risk of cascading liquidations. Future work may incorporate other major Curve pools that include crvUSD (such as TriCRV).

According to CoinMarketCap (as of January 2024), upwards of 80% of all crvUSD volume is contained in the four Peg Keeper pools and so is the majority of its TVL.

7.8 Partial Liquidations

Currently, all liquidations in the model are full liquidations. Technically, the crvUSD protocol allows for full liquidations via the public `liquidate_extended` method. However, we have found no evidence of partial liquidations when reviewing historical on-chain data. We have made the conservative assumption that all liquidations must be performed in full, which increases the burden on liquidators. This means we are not modeling the effects (positive or negative) that partial liquidations would have on the system. In future work, support for partial liquidations may be included in the model if there is evidence that such liquidations are performed, or to understand how they would affect the system.

7.9 Future Work

Our risk model may be used to better understand and improve the crvUSD protocol. One example for how it might be used is in onboarding new markets. Our model enables a researcher to understand the impact of adding new collateral types on the protocol's risk and optimize parameters.

The model may also be used to investigate the impact of changing the underlying mechanisms in the protocol's smart contracts. We use the `crvusdsim` and `curvesim` Python packages to simulate the underlying Curve pools, which may be easily modified to consider new implementations, such as new `Peg Keeper`s or new `Oracle`s. A modeler may then be able to quantify the impact of changing the underlying code, and use it to decide whether or not the contracts should be updated.

In terms of model improvements, our primary concern is to reduce the computational complexity of the model so it can handle simulations at higher granularities. Ideally, the model should be able to handle block-by-block updates for a one day horizon, without making it untenable to run thousands of simulations. Further improvements include modeling network congestion (i.e., varying gas costs) and incorporating "sticky" liquidity for collateral tokens, meaning that trades against external markets at time t_i will affect simulated prices at times $t > t_i$.

Appendix

A

Authors and Acknowledgements

This report was prepared by Thomas Cintra (@tncintra) at Xenophon Labs under the supervision of the Curve Research and Llama Risk teams.

A special thank you to the following people and teams who have been crucial to the development of this model and analysis:

- Vishesh Choudhry (@visavishesh) for his advisory role in guiding the project, risk methodologies, and analysis.
- Chan-Ho Suh and Nagaking at the Curve Research team for their guidance in understanding the crvUSD protocol, and constant feedback on our progress.
- WormholeOracle and the Llama Risk team for their feedback on our analysis.
- 0xReviews for building the `crvUSDsim` package, which was invaluable in developing our risk model.
- Max at Xenophon Labs for document review and guidance in developing our risk methodologies.

B

Additional Scenarios

We also tested a few scenarios with varying jump sizes and negative drifts. We found little to no risk in the system exhibited by applying a negative drift compared to extreme asset volatility. For this reason, we focus on the high volatility scenarios. We found meaningful risk in the system by applying large negative jumps to collateral prices (referred to as "flash crashes"). However, as we discuss in §7, this is likely due to a limitation in our model than actual systemic risk. The depeg scenario will be key in our analysis in the following section. These scenarios are shown in Table 12.

	vol	debt	liquidity	mu	jump
adverse drift	baseline	baseline	baseline	severe	baseline
severe drift	baseline	baseline	baseline	severe	baseline
adverse flash crash	baseline	baseline	baseline	baseline	adverse
severe flash crash	baseline	baseline	baseline	baseline	severe
adverse depeg	baseline	baseline	baseline	baseline	adverse
severe vol and adverse drift	severe	baseline	baseline	severe	baseline
severe vol and severe drift	severe	baseline	baseline	severe	baseline
adverse flash crash and adverse growth	baseline	adverse	baseline	baseline	adverse
adverse flash crash and severe growth	baseline	severe	baseline	baseline	adverse
adverse flash crash and adverse crvusd liquidity	baseline	baseline	adverse	baseline	adverse
adverse flash crash and severe crvusd liquidity	baseline	baseline	severe	baseline	adverse

Table 12: Additional scenarios used in our simulations and their stressor configuration.

The adverse and severe drifts are computed using the same methodology as done for asset volatilities, discussed in §4.1.1. The adverse and severe jump sizes are computed as follows for the flash crash scenarios:

1. Compute the standard deviation of log returns in your sampling period.
2. Filter out all log returns below N standard deviations.
3. Compute the mean log returns of the remaining samples.

In our case, we consider only negative jumps to calculate jump sizes, and we use $N = 4$ for our adverse scenario and $N = 10$ for our severe scenario. In either case, the probability of a jump occuring is 1 halfway through the simulation, and zero elsewhere. This means we enforce a single large jump. Similarly, we enforce a 20% depeg for the adverse depeg scenario.

Results for these additional scenarios may be viewed alongside other results in our GitHub repository or using our dashboard.

Xenophon Labs
January 2024