Sustainability of Pendle V2 and its incentive mechanisms (vePENDLE)

Vu Nguyen
vu@pendle.finance

Nghia Pham
ngfam@pendle.finance

October 14, 2022

Abstract

Pendle V2 is a permission-less system for tokenising and trading of yields for any yield-generating mechanisms in DeFi. This paper discusses the sustainability of Pendle V2 and describes a system of incentive mechanisms inspired by the VotingEscrow system in CurveDAO [1].

1 Sustainability of Pendle V2

1.1 Overview

Pendle V2 consists of 2 components:

- Standardized Yield Stripping [3]: a protocol to strip any SY into Yield Token (YT) and Principal Token (PT)
- PT AMM [2]: a protocol to trade PT against SY, which is used as the backbone to trade PT and YT against any token. Each pool of the PT AMM is called a Pendle market.

1.2 Protocol revenue in PendleV2

There are two sources of protocol revenue in PendleV2:

- **YT fees**: The protocol takes a portion of the yields being split into YT
- **Swap fees**: The protocol takes a portion of the swap fees in the PT AMM

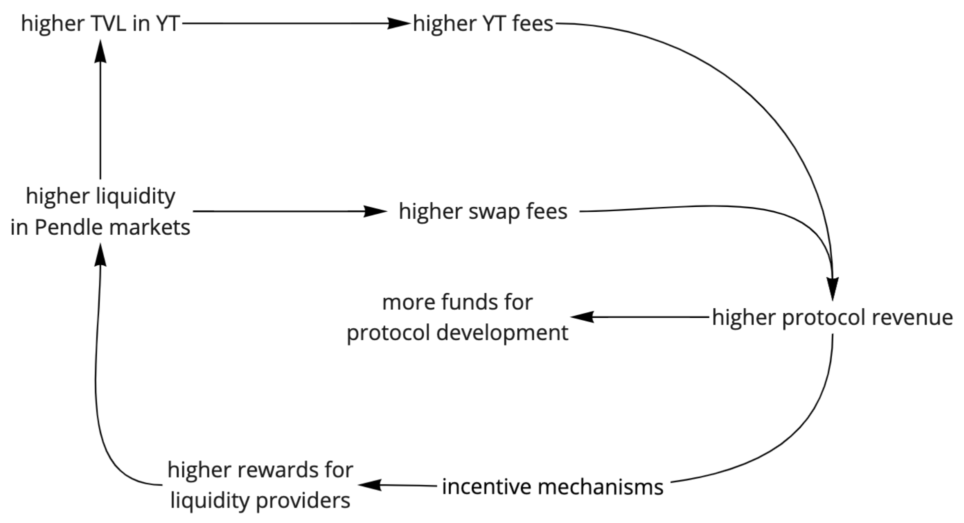
1.3 Sustainability and incentive mechanisms

In the long term, there are two major factors in increasing protocol revenue:

- **Protocol development** This involves product development, marketing and education to increase protocol utilization.
- **Pendle market liquidity** Deeper liquidity encourages more trading.

To ensure the sustainablity and growth of the protocol, the actors' interests should be aligned to create a virtuous cycle leading to the protocol's growth.

- Protocol development will be funded by a developer fund that takes a fixed x% of the protocol revenue



higher TVL in YT
higher YT fees
higher liquidity in Pendle markets
higher swap fees
more funds for protocol development
higher protocol revenue
higher rewards for liquidity providers
incentive mechanisms

- On Pendle market liquidity, there needs to be incentive mechanisms to align liquidity providers (as the main source of liquidity) to protocol revenue.

In this paper, we will describe Pendle V2's incentive mechanism, which comprises of:

- The PENDLE token, which is an ERC20 token used to incentivize liquidity providers for Pendle markets
- A mechanism to lock PENDLE to generate Vote-escrowed PENDLE (vePENDLE), which receives protocol revenue. vePENDLE holders are incentivized to act in the best interests of the protocol.

2 Vote-escrowed PENDLE

In this section, we describe the core mechanics of Vote-escrowed PENDLE (vePENDLE), a system to lock PENDLE to mint an amount of vePENDLE. This enables various utilities within the Pendle protocol.

2.1 Definitions

PENDLE is an ERC20 token.

max lock duration is the maximum duration of 2 years that a user can lock their PENDLE for.

locked position of a user u comprises of the $lockAmount_u$ which is the amount of PENDLE locked by user u, and the $lockExpiry_u$ which is the timestamp that user u has locked their PENDLE until. $lockExpiry$ must be in exact weeks from UNIX timestamp 0.

vePENDLE value of a user u at time $t < lockExpiry_u$ is:

$$vePendleValue_u(t) = \frac{lockAmount_u(lockExpiry_u - t)}{MaxLockDuration} \tag{1}$$

If $t >= lockExpiry_u$, $vePendleValue_u(t) = 0$

vePENDLE balance is a line $bias - slope \times t$ that represents vePENDLE value over time, for a certain user, or a group of users. vePENDLE balance of a user is $bias_u - slope_u \times t$
Combined with (1), we can convert from a user's locked position into the bias and slope of the user's vePendle balance:

$$\begin{aligned} slope_u &= \frac{lockAmount_u}{MaxLockDuration} \\ bias_u &= \frac{lockAmount_u \times lockExpiry_u}{MaxLockDuration} \\ &= slope_u \times lockExpiry_u \end{aligned}$$

vePENDLE total supply is the total amount of vePENDLE of all users

$$\begin{aligned} veTotalSupply(t) &= bias_{total} - slope_{total} \times t \\ bias_{total} &= \sum_u bias_u \\ slope_{total} &= \sum_u slope_u \end{aligned}$$

2.2 State changes

2.2.1 Increase lock position

When a user's locked position is $(lockAmount_u, lockExpiry_u)$ (could be $(0,0)$ when the user has no lock, or the previous lock position has already expired), they can increase their position to $(lockAmount'_u, lockExpiry'_u)$, where

$$\begin{aligned} lockAmount'_u &\geq lockAmount_u \\ lockExpiry'_u &\geq lockExpiry_u \end{aligned}$$

As a result, bias and slope of the user's vePENDLE balance will increase accordingly due to the additional locked position:

$$\begin{aligned} bias_{additional} &= \frac{lockAmount'_u \times lockExpiry'_u - lockAmount_u \times lockExpiry_u}{MaxLockDuration} \\ slope_{additional} &= \frac{lockAmount'_u - lockAmount_u}{MaxLockDuration} \end{aligned}$$

The vePENDLE total supply's bias and slope will increase by the same amount as well:

$$\begin{aligned} bias_{total} &= bias_{total} + bias_{additional} \\ slope_{total} &= slope_{total} + slope_{additional} \end{aligned}$$

There are 3 different ways to increase a user's lock position:

- Create a new non-zero locked position: when $(lockAmount_u, lockExpiry_u) = (0,0)$, user u can create a new locked position $(lockAmount'_u, lockExpiry'_u)$ $(lockAmount'_u > 0, lockExpiry'_u > 0)$
- Extend a locked position at time t: when $t < lockExpiry_u$, user u can extend their locked position from $(lockAmount_u, lockExpiry_u)$ to $(lockAmount_u, lockExpiry'_u)$ ($lockExpiry'_u > lockExpiry_u$)

- Add to a locked position at time t: when $t < lockExpiry_u$, user u can add PENDLE to their locked position $(lockAmount_u, lockExpiry_u)$ to get a new locked position $(lockAmount'_u, lockExpiry_u)$ ($lockAmount'_u > lockAmount_u$)

2.2.2 Expiry of a locked position

At $lockExpiry_u$, the position of user u expires and u's position becomes $(lockAmount_u, lockExpiry_u) = (0, 0)$. The vePENDLE total supply's slope and bias will decrease accordingly:

$$\begin{aligned} bias_{total} &= bias_{total} - bias_u \\ slope_{total} &= slope_{total} - slope_u \end{aligned}$$

Implementation details
We can process the expiry of all locked positions on the same expiry at once, by storing $slopeChanges(t) = \sum slope_u^t$ for all users u with a position expiring at t.
We will have:

$$\begin{aligned} \sum bias_u^t &= \sum slope_u^t \times t \\ &= slopeChanges(t) \times t \end{aligned}$$

Then, we adjust the total bias and total slope accordingly:

$$\begin{aligned} bias_{total} &= bias_{total} - slopeChanges(t) \times t \\ slope_{total} &= slope_{total} - slopeChanges(t) \end{aligned}$$

3 Cross-chain vePENDLE

While vePENDLE locking will happen entirely on the main chain (Ethereum), information about each user's locked position will be broadcast to other side chains.

3.1 Definitions

Main chain is a smart contract blockchain (Ethereum in Pendle's case) where PENDLE is deployed and where PENDLE is locked to generate vePENDLE.

Side chains are smart contract blockchains that are not the main chain

Locked position (side chain) of a user u on side chain X is $(lockAmount_u^X, lockExpiry_u^X)$

vePENDLE value and vePENDLE balance (side chain) of a user u on side chain X are defined in the same way as 2.1, based on the locked position on the side chain

$$\begin{aligned} vePendleValue_u^X(t) &= \frac{lockAmount_u^X(lockExpiry_u^X - t)}{MaxLockDuration} \quad (2) \\ slope_u^X &= \frac{lockAmount_u^X}{MaxLockDuration} \quad (3) \\ bias_u^X &= \frac{lockAmount_u^X \times lockExpiry_u^X}{MaxLockDuration} \quad (4) \\ &= slope_u^X \times lockExpiry_u^X \quad (5) \end{aligned}$$

vePENDLE total supply (side chain) is a vePENDLE balance defined by $bias^X_{total}$ and $slope^X_{total}$. Note that $bias^X_{total} = \sum_u bias^X_u$ and $slope^X_{total} = \sum_u slope^X_u$ are not necessarily true (because some users' increased locked positions might not be broadcasted)

3.2 State changes

3.2.1 Broadcast vePENDLE total supply

At time t, anyone can broadcast the vePENDLE total supply from the main chain to all the side chains. As a result, for all the side chains X at time t

$$\begin{aligned} bias^X_{total} &= bias_{total} \\ slope^X_{total} &= slope_{total} \end{aligned}$$

3.2.2 Broadcast user locked position and vePENDLE total supply

At time t, anyone can broadcast the locked position of a user u and the vePENDLE total supply to a side chain X.
As a result, at time t:

$$\begin{aligned} bias^X_{total} &= bias_{total} \\ slope^X_{total} &= slope_{total} \\ lockAmount^X_u &= lockAmount_u \\ lockExpiry^X_u &= lockExpiry_u \end{aligned}$$

3.3 Implications

3.3.1 Users would naturally want to broadcast their positions by themselves

There are only two state changes (2.2.1 and 3.2.2) that change a user u's locked position on the main chain or a side chain. 3.2.2 will equate the lock position on the side chain with that on the main chain. 2.2.1 will simply increase the lock position on the main chain.
As such, the lock positions on side chains are always less than or equal to that on the main chain:

$$\begin{aligned} lockAmount^X_u(t) &\le lockAmount_u(t) \\ lockExpiry^X_u(t) &\le lockExpiry_u(t) \end{aligned}$$

As a result, the bias, slope and vePENDLE value of user u are all less than or equal to that on the main chain

$$\begin{aligned} vePendleValue^X_u(t) &\le vePendleValue_u(t) \\ slope^X_u(t) &\le slope_u(t) \\ bias^X_u(t) &\le bias_u(t) \end{aligned}$$

Since vePENDLE value dictates how much benefits users get, a rational user would always want to trigger state change 3.2.2 (broadcast their lock position). In other words, users would only stand to lose if their updated lock position is not broadcasted to the rest of the chains.

3.3.2 Inaccurate vePENDLE total supply at the start of the week

After every week, vePENDLE total supply on the main chain will be higher than on the side chains, because the slope changes are not immediately reflected on the side chains, unless there is a broadcast transaction, while the remaining locked period is reflected on the side chains.
There are 3 ways this issue is mitigated:

1. The stakeholders who want the Pendle protocol to work properly could run the broadcast transactions at the start of the week. The Pendle team will trigger these transactions as part of our operation.
2. From the previous implications, users will naturally broadcast their newly increased positions, which will broadcast the vePENDLE total supply as well
3. Users will naturally want to broadcast a higher vePENDLE total supply and update other users' *activeBalance*, which will be elaborated subsequently.

4 Voting for PENDLE incentives

4.1 Definitions

pools are Pendle markets, that could be on the main chain or a side chain. There are n pools in total from p_1 to p_n that are eligible for incentives voting.

global reward speed is the rate of PENDLE per second in total that will be given as liquidity incentives. This rate can only be set by the governance role.

governance role is a role to govern the important parameters of the protocol. Initially it will be a multisig controlled by the Pendle team, which could be replaced by smart contracts for on-chain voting in the longer future.

user vote of a user u for a pool p_i is a weight $0 \leq weight_i^u \leq 1$. This voting weight is equivalent to a vePENDLE balance $biasForVote_i^u - slopeForVote_i^u \times t$. Sum of user weights for all pools must not exceed 1.

$$\sum_i weight_i^u \leq 1$$

pool vote for a pool p_i is a vePENDLE balance $bias_{p_i} - slope_{p_i} \times t$, such that:

$$bias_{p_i} = \sum_u biasForVote_i^u$$
$$slope_{p_i} = \sum_u slopeForVote_i^u$$

accounting weeks are weeks that start and end at round number of weeks from UNIX timestamp 0. These weeks are identified by the start of the week $t_{weekStart}$. The accounting for how PENDLE incentives are distributed is based on accounting weeks.

pool vote count of a pool p_i for a week starting at $t_{weekStart}$ is simply the vePENDLE value of the pool vote at $t_{weekStart}$

$$poolVoteCount^i_{t_{weekStart}} = bias_{p_i} - slope_{p_i} \times t_{weekStart}$$

total vote count for a week starting at $t_{weekStart}$ is the sum of all pool vote counts for the same week

$$totalVoteCount_{t_{weekStart}} = \sum_i poolVoteCount^i_{t_{weekStart}}$$

pool weekly reward speed of a pool p_i on a week starting at $t_{weekStart}$ is how much PENDLE per second pool p_i is entitled to get for the week.

$$poolWeeklySpeed_{t_{weekStart}} = globalRewardSpeed \times \frac{poolVoteCount_{t_{weekStart}}}{totalVoteCount_{t_{weekStart}}}$$

4.2 State changes

4.2.1 Add or remove a pool

Pools can be added or removed by a Governance role. When a new pool is added, all users' votes for the pool are 0 by default.

4.2.2 Update vote

A user u can update their vote for a pool p_i to be $weight^u_i$ at time t. At this point, u's $biasForVote^u_i$ and $slopeForVote^u_i$ are calculated based on u's vePENDLE balance at t:

$$biasForVote^u_i = bias_u \times weight^u_i$$
$$slopeForVote^u_i = bias_u \times weight^u_i$$

This means that if a user increases their lock position (and hence their vePENDLE balance), they will need to update their vote for the different pools to reflect their new vePENDLE balance

4.2.3 Expiry of a user vote

At $lockExpiry_u$, all votes by u will expire together with their locked position. The pool vote will decrease accordingly:

$$bias_{p_i} = bias_{p_i} - biasForVote^u_i$$
$$slope_{p_i} = slope_{p_i} - slopeForVote^u_i$$

Implementation details
Similar to the expiry of a locked position in 2.2.2, we can also store the $slopeChanges_i(t)$ for all user votes expiring at t, and adjust the slope and bias of the pool vote in one go.

5 Distribution of PENDLE incentives to each pool

5.1 Definitions

distribution timestamp of an accounting week $distributionTime(t_{weekStart})$ for a pool p_i is when the distribution of the PENDLE rewards reserved for the pool for that accounting week starts, which could be a few hours after the start of the accounting week.

pool actual speed of a pool p_i is how much PENDLE per second pool p_i will actually distribute to the liquidity providers in real time. This is not always the same as $poolWeeklySpeed$

5.2 State changes

5.2.1 Broadcast voting results

At time t during an accounting week starting at $t_{weekStart}$, anyone can broadcast the voting results for accounting week $t_{weekStart}$ (which are already finalised at $t_{weekStart}$) to a particular chain X. The voting results include the pool weekly rewards for all the pools in chain X.

For each pool p_i, the distribution timestamp for accounting week $t_{weekStart}$ is set to be t, and the pool actual speed is set to include the **pool weekly reward speed** and any left over incentives if the distribution for the previous accounting week is not done yet

6 Liquidity mining rewards for each market

6.1 Overview

There are 2 types of rewards for liquidity providers to each Pendle market:

1. PENDLE incentives
2. Reward tokens from the SY tokens in the market

If a user u has more vePENDLE value on the chain Y of the market, u will get higher liquidity mining rewards.

6.2 Definitions and specifications

LP balance is the amount of LP tokens of a user u, $lpBalance_u$

total LP is the total amount of LP tokens

active balance is a unit used to distribute rewards among the users.

$$activeBalance_u = min(lpBalance_u, boostedBalance_u)$$

$$boostedBalance_u = 0.4lpBalance_u + 0.6totalLP\frac{vePendleValue_u^Y}{veTotalSupply^Y}$$

The rewards are distributed among the liquidity providers, proportionally to their *activeBalance*. This boosting of liquidity mining rewards follow the same mechanics as Curve [1].

Users' *activeBalance* are updated whenever there is a transfer (including minting and bunring during liquidity addition or removal) of the LP tokens.

notice The *activeBalance* values used for distributing rewards among the users are the last values when an action was done to update it. As such, each user's boost in liquidity mining rewards is not reflected in real time and is only a cached value. If a user u's locked position has expired, anyone can update *activeBalance* for u to reflect u's 0 vePENDLE value.

6.3 Implications

6.3.1 Users will naturally broadcast vePENDLE totalSupply

At the start of the week, vePENDLE supply is higher. After every week, vePENDLE total supply on the main chain will be higher than on the side chains, unless there is a broadcast transaction.

Naturally, users will want to broadcast the higher vePENDLE total supply and update others' *activeBalance*, which will decrease with a higher vePENDLE total supply.

References

[1] Curve Finance. *CurveDAO*. URL: https://curve.fi/files/CurveDAO.pdf.

[2] Vu Nguyen. *A study on AMMs for trading fixed yield and Pendle V2's Principal Token AMM*. URL: https://pendle.finance/V2_AMM.pdf.

[3] Vu Nguyen. *Standardized Yield Stripping - efficient yield stripping mechanism on DeFi's yield generating assets*. URL: https://pendle.finance/SYS.pdf.