Volatility AMM: Technical Design Document

Pravin and Nikhil Bezwada

May 17, 2025

1 Introduction

This document provides a comprehensive technical design for a Decentralized Finance (DeFi) Automated Market Maker (AMM) tailored specifically for options trading, termed the Volatility AMM. Unlike traditional AMMs focused on spot assets, this system is engineered to handle the complexities of options markets by leveraging a volatility-driven bonding curve. The Volatility AMM facilitates efficient price discovery, robust liquidity provision, and secure collateral management through synthetic volatility tokens (vTokens). By integrating implied volatility dynamics with a constant product invariant, the system addresses critical challenges in decentralized options trading, including fragmented liquidity, complex pricing models, and counterparty risk. The design ensures continuous and transparent pricing, maintains collateral safety under volatile market conditions, and creates arbitrage opportunities to align AMM prices with broader market dynamics. This framework establishes a scalable and resilient infrastructure for decentralized options markets, suitable for both retail and institutional participants.

2 Volatility Bonding Curve Fundamentals

2.1 Core Concept

The volatility bonding curve is the cornerstone of the Volatility AMM, defining a mathematical relationship between key variables to ensure consistent pricing and liquidity. These variables include:

- **vToken Supply (x)**: Represents synthetic volatility points, where 1 vToken corresponds to 1% annualized implied volatility (IV).
- **USDC Reserves (y)**: The collateral pool, denominated in USDC, that backs the vTokens and ensures system solvency.
- **Invariant (k)**: A constant product that maintains the balance between vToken supply and USDC reserves, adhering to the AMM's core principle.

The relationship is governed by the constant product formula:

$$x \cdot y = k \tag{1}$$

This invariant ensures that as vTokens are bought or sold, the reserve adjusts dynamically to reflect changes in implied volatility and maintain market equilibrium.

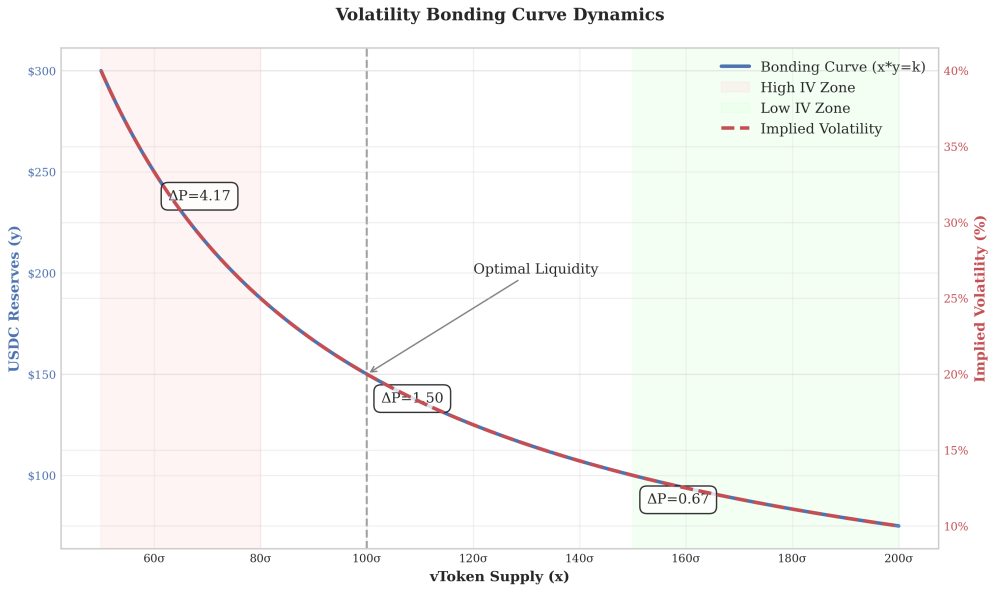

Figure 1: Volatility bonding curve illustrating price impact and collateral zones. The curve shows how vToken trades affect USDC reserves and implied volatility.

2.2 Mechanics Breakdown

2.2.1 Initialization

The bonding curve is initialized when a liquidity provider (LP) deposits collateral to mint vTokens. For example:

1. An LP deposits 150 USDC to mint 100 vTokens at an initial implied volatility of 20%. The invariant is calculated as:

$$k = x \cdot y = 100 \cdot 150 = 15,000 \tag{2}$$

2. The implied volatility is derived from the vToken supply using a calibration constant (C), which aligns the system with market volatility:

$$IV = \frac{C}{x} \tag{3}$$

This setup ensures that the AMM starts with a balanced pool, ready for trading activity.

2.2.2 Trading Impact

Trading vTokens directly affects the bonding curve, adjusting implied volatility and reserves. The following table summarizes the effects:

Action	Formula	Market Effect
Buy vTokens	$x' = x - \Delta x$	Reduces vToken supply, increasing IV due to higher demand.
Sell vTokens	$x' = x + \Delta x$	Increases vToken supply, decreasing IV due to increased supply.

Table 1: Impact of vToken trades on the bonding curve and implied volatility.

3 Advanced Dynamics

3.1 Collateral Rebalancing

The bonding curve dynamically adjusts USDC reserves during trades to maintain the constant product invariant. The change in reserves is calculated as:

$$\Delta y = \left| \frac{k}{x'} - \frac{k}{x} \right| \tag{4}$$

Consider a purchase of 10 vTokens from the initial state ($x = 100$, $y = 150$, $k = 15,000$):

$$x' = 100 - 10 = 90 \tag{5}$$

$$y' = \frac{15,000}{90} \approx 166.67 \tag{6}$$

$$\Delta y = 166.67 - 150 = 16.67 \text{ USDC added} \tag{7}$$

This ensures that the system remains fully collateralized, with reserves increasing to reflect the reduced vToken supply.

3.2 Volatility Smile Integration

To account for the volatility smile (variation in IV across strike prices), the bonding curve incorporates a moneyness adjustment:

$$k_{\text{adjusted}} = k \cdot \left(1 + \alpha \cdot \frac{\partial^2 P}{\partial K^2}\right) \tag{8}$$

Here, α is a convexity factor calibrated for each trading bucket, ensuring that the AMM accurately reflects market-implied volatility curves for different strike prices.

4 Implementation

4.1 Smart Contract Logic

The core trading functionality is implemented in a Solidity smart contract, handling vToken swaps and volatility updates:

```
function swap(
    uint bucketId,
    bool isBuy,
    uint amount
) external {
    Bucket storage b = buckets[bucketId];
    uint newX = isBuy ? b.vTokens - amount : b.vTokens + amount;
    uint newY = k[bucketId] / newX;

    // Update IV oracle with dynamic adjustment
    iv[bucketId] = BASE_IV * (b.vTokens / newX);

    // Transfer assets with fee handling
    if (isBuy) {
        USDC.transferFrom(msg.sender, address(this), amount + fee);
        b.vTokens -= amount;
    } else {
        USDC.transfer(msg.sender, amount - fee);
        b.vTokens += amount;
    }
}
```

This function ensures atomic updates to the bonding curve, reserves, and implied volatility oracle.

4.2 Key Properties

The Volatility AMM guarantees several critical properties:

Property	**Guarantee**
Continuous Pricing	The price of vTokens is always defined, with $\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = \frac{dy}{dx}$.
Collateral Safety	Reserves always satisfy $y \geq x \cdot IV_{\min}$, ensuring solvency.
Arbitrageable	AMM prices converge to market prices ($P_{\text{AMM}} \to P_{\text{market}}$) through arbitrage incentives.

5 Synthetic vTokens

5.1 Token Specification

vTokens are synthetic assets representing volatility exposure, with the following properties:

$$\text{vToken} = \begin{cases} 1 \text{ vToken} \equiv 1\% \text{ annualized volatility} \\ \text{ERC-1155 standard for efficient batch management} \\ \text{Non-transferable during active bucket phase to prevent manipulation} \end{cases} \tag{9}$$

5.2 Minting Mechanics

The minting process converts USDC deposits into vTokens based on current implied volatility: [H] vToken Minting

Require: $USDC_{\text{deposit}} \geq 0$, $bucket_{\text{id}}$ active

0: $iv \leftarrow \text{currentIV}(bucket_{\text{id}})$

0: $vTokens \leftarrow \frac{USDC_{\text{deposit}}}{iv \cdot \text{collateralFactor}}$

0: $\textsc{mint}(msg.sender, vTokens)$

0: $\textsc{lockCollateral}(USDC_{\text{deposit}}) = 0$

6 vToken Trading Mechanics

6.1 Overview

Synthetic vTokens are designed to represent volatility exposure within the Volatility AMM, with specific constraints on their tradability to ensure system stability and prevent market manipulation. This section details the mechanisms governing vToken trading, including internal AMM transactions and restrictions on external transfers.

6.2 Internal Trading via Bonding Curve

During the active bucket phase, vTokens can be bought or sold within the Volatility AMM using the bonding curve mechanism. The smart contract `swap` function (Section 4.1) facilitates these trades:

- **Buying vTokens**: Users deposit USDC to reduce vToken supply ($x' = x - \Delta x$), increasing implied volatility and adjusting reserves ($y' = k/x'$).
- **Selling vTokens**: Users receive USDC to increase vToken supply ($x' = x + \Delta x$), decreasing implied volatility.

These transactions are atomic and maintain the constant product invariant ($x \cdot y = k$).

6.3 Non-Transferability Constraints

To ensure bucket stability and prevent speculative manipulation, vTokens are non-transferable during the active bucket phase. This is enforced through:

- **ERC-1155 Implementation**: The vToken contract restricts `safeTransferFrom` calls during the active phase.
- **Bucket Status Checks**: The `Bucket.isActive` flag (Section 8.1) gates transfer functions.

Non-transferability ensures that vTokens remain tied to the AMM's collateral pool, preserving the integrity of the bonding curve.

6.4 Post-Active Phase Considerations

In the rolling and settled phases (Section 7.1), vTokens are either frozen for settlement or burned. Currently, the design does not support external trading post-settlement, as vTokens are burned to release collateral. Future iterations may explore:

- Limited transferability post-settlement for secondary market integration.
- Wrapping vTokens into transferable ERC-20 tokens for external trading.

Such extensions would require additional smart contract logic and governance mechanisms to maintain collateral safety.

6.5 Comparison with Other DeFi Tokens

Unlike freely transferable ERC-20 tokens (e.g., UNI or AAVE), vTokens prioritize AMM-specific functionality over external tradability. This design aligns with options-focused DeFi protocols, where tokens are often locked to specific contracts during active trading periods.

7 Bonding Curve Dynamics

7.1 Core Equation

The bonding curve operates on the constant product principle:

$$x \cdot y = k \tag{10}$$

Where:

- x: vToken supply in the trading bucket
- y: USDC reserves backing the bucket
- k: Dynamic invariant, recalculated after fee collection to account for system growth

7.2 Price Impact

The price of a vToken is determined by the marginal change in reserves:

$$P_{\text{vToken}} = \frac{dy}{dx} = \frac{k}{x^2} \tag{11}$$

This ensures that price increases with demand and decreases with supply, aligning with market dynamics.

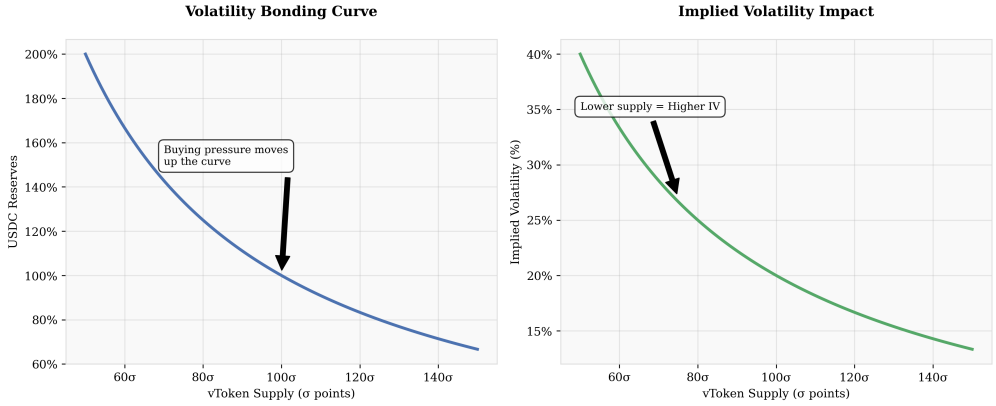

Figure 2: Bonding curve with highlighted liquidity zones, showing stable and volatile regions.

8 Integrated System Workflow

8.1 vToken Lifecycle

vTokens progress through distinct phases, each with specific operational constraints.

Phase	vToken Status	Operations
Active	Mintable/Redeemable	Trading, liquidity provision, and redemption.
Rolling	Frozen	Settlement calculations, no new trades.
Settled	Burnable	Final withdrawals and vToken burning.

Table 2: vToken lifecycle phases and permitted operations.

8.2 Competitive Advantages

The Volatility AMM offers significant improvements over existing solutions:

Feature	Our Solution	Alternatives
Price Discovery	Dynamic bonding curve with IV feedback	Static oracles with delayed updates
Liquidity	Curve-determined spreads, adaptive to volatility	Fixed-width orders, prone to slippage
Settlement	Automated vToken burn mechanism	Manual exercise, high operational overhead

9 Competitive Landscape & Strategic Advantages

9.1 Feature Comparison

Feature	vToken	Opyn	Lyra	Premia	Deribit
Pricing Model	Bonding Curve	Black-Scholes	BS + Skew	BS + Surface	Centralized
Collateral	150% Dynamic	100% Static	Pooled	100% Static	10x Leverage
Liquidity	Isolated Buckets	Fragmented	Pooled	Fragmented	Central Book
Expiry Handling	Auto-Rolling	Manual	Static	Manual	Weekly/Monthly
Oracles	IV Feed Free	Vol Oracle Needed	Spot Only	Vol Oracle Needed	Internal
Composability	vToken ERC-1155	SLP Tokens	LP Shares	NFTs	None

Table 3: Protocol Feature Matrix

9.2 Why Competitors Can't Replicate

Protocol	Structural Limitation
Opyn	Static collateral can't adapt to volatility spikes
Lyra	Pooled risk creates LP dilution
Premia	Manual expiry management
Deribit	No composability for DeFi

10 Implementation Details

10.1 Smart Contract Architecture

The smart contract is designed for modularity and security:

```
contract VolatilityAMM {
    struct Bucket {
        uint256 vTokenSupply;  // x in x*y=k
        uint256 usdcReserves;  // y in x*y=k
        uint256 iv;           // Implied volatility
        bool isActive;        // Bucket status
    }
```

```
    function mintvTokens(uint bucketId, uint usdcAmount) external {
        // Implements bonding curve math
        uint newK = calculateNewK(bucketId, usdcAmount);
        // Updates IV oracle
        updateImpliedVolatility(bucketId);
    }
}
```

This structure supports scalable bucket management and efficient state updates.

11 Key Innovations

- **Hybrid Collateralization**: vTokens serve dual purposes as both volatility exposure and a claim on collateral, streamlining capital efficiency.
- **Convexity Adjustment**: The bonding curve dynamically adjusts for the volatility smile using:

$$k_{\text{adjusted}} = k \cdot \left(1 + \frac{\partial^2 P}{\partial K^2}\right) \tag{12}$$

- **Fee Recycling**: 30% of trading fees are redirected to buffer undercollateralized buckets, enhancing system stability during market stress.

12 Option Book Generation from vToken Liquidity

12.1 Overview

The Volatility AMM leverages vToken liquidity to construct call and put option books, enabling decentralized options trading. vTokens, representing annualized implied volatility, are mapped to option prices using a modified Black-Scholes framework, adjusted for the bonding curve's implied volatility dynamics. This section details the process of generating option books from vToken liquidity, including pricing, liquidity allocation, and smart contract implementation.

12.2 Pricing Mechanism

The implied volatility (IV) derived from vToken supply (x) serves as the primary input for option pricing. The Black-Scholes model is adapted as follows:

$$IV = \frac{C}{x} \tag{13}$$

where C is the calibration constant (Section 2.2.1). For a given strike price K and time to expiry T, call and put prices are calculated:

$$c = S \cdot N(d_1) - K \cdot e^{-rT} \cdot N(d_2) \tag{14}$$

$$p = K \cdot e^{-rT} \cdot N(-d_2) - S \cdot N(-d_1) \tag{15}$$

where:

- S: Underlying asset price (sourced from an external oracle)
- r: Risk-free rate (assumed constant or oracle-fed)
- $N(\cdot)$: Cumulative normal distribution
- $d_1 = \frac{\ln(S/K)+(r+IV^2/2)T}{IV\sqrt{T}}$
- $d_2 = d_1 - IV\sqrt{T}$

The bonding curve's IV ensures that option prices dynamically reflect vToken supply changes.

12.3 Liquidity Allocation

vToken liquidity is allocated across option books for multiple strike prices and expiries within a trading bucket:

- **Bucket Segmentation**: Each bucket corresponds to a specific underlying asset and expiry. Within a bucket, vToken supply (x) is split into sub-pools for different strikes based on moneyness.
- **Volatility Smile Adjustment**: The convexity adjustment (Section 3.2) modifies IV for each strike:
$$IV_{\text{adjusted}} = IV \cdot \left(1 + \alpha \cdot \frac{\partial^2 P}{\partial K^2}\right) \tag{16}$$
- **Reserve Mapping**: USDC reserves (y) back the option book, with collateral requirements calculated as:
$$y_{\text{option}} = \max(c, p) \cdot \text{collateralFactor} \tag{17}$$

This ensures sufficient collateral for both call and put options, maintaining system solvency.

12.4 Smart Contract Implementation

The option book is generated via a dedicated smart contract function that maps vToken liquidity to option prices:

```
function generateOptionBook(uint bucketId, uint[] memory strikes, uint expiry) external
    Bucket storage b = buckets[bucketId];
    uint iv = BASE_IV * (INITIAL_VTOKENS / b.vTokens);

    for (uint i = 0; i < strikes.length; i++) {
        uint strike = strikes[i];
        // Calculate adjusted IV for strike
        uint ivAdjusted = adjustIVForSmile(iv, strike);

        // Black-Scholes pricing
        (uint callPrice, uint putPrice) = blackScholes(
            getSpotPrice(),
            strike,
            ivAdjusted,
            expiry,
            RISK_FREE_RATE
        );

        // Allocate reserves
        uint collateral = max(callPrice, putPrice) * COLLATERAL_FACTOR;
        require(b.usdcReserves >= collateral, "Insufficient reserves");

        // Update option book
        options[bucketId][strike] = Option(callPrice, putPrice, collateral);
    }
}
```

This function ensures that option prices are consistent with vToken liquidity and bonding curve dynamics.

12.5 Trading and Settlement

Options are traded as synthetic assets within the AMM:

- **Buying/Selling**: Users trade options by interacting with the option book, with prices updated based on vToken trades.
- **Settlement**: At expiry, options are settled automatically. Call options pay $\max(S_T - K, 0)$, and put options pay $\max(K - S_T, 0)$, with payouts drawn from USDC reserves.

The non-transferability of vTokens (Section 6.3) ensures that liquidity remains locked during the active phase, supporting stable option trading.

12.6 Advantages

This approach offers:

- **Dynamic Pricing**: Option prices adjust in real-time based on vToken supply and IV.
- **Collateral Efficiency**: Shared USDC reserves across strikes optimize capital use.
- **Scalability**: Multiple buckets support diverse assets and expiries.

13 Bonding Curve Dynamics

13.1 Core Equation

The bonding curve operates on the constant product principle:

$$x \cdot y = k \tag{18}$$

Where:

- x: vToken supply in the trading bucket
- y: USDC reserves backing the bucket
- k: Dynamic invariant, recalculated after fee collection to account for system growth

13.2 Price Impact

The price of a vToken is determined by the marginal change in reserves:

$$P_{\text{vToken}} = \frac{dy}{dx} = \frac{k}{x^2} \tag{19}$$

This ensures that price increases with demand and decreases with supply, aligning with market dynamics.

14 Option Book Generation from vToken Liquidity

14.1 Overview

The Volatility AMM leverages vToken liquidity to construct call and put option books, enabling decentralized options trading. vTokens, representing annualized implied volatility, are mapped to option prices using a modified Black-Scholes framework, adjusted for the bonding curve's implied volatility dynamics. This section details the process of generating option books from vToken liquidity, including pricing, liquidity allocation, and smart contract implementation.

14.2 Pricing Mechanism

The implied volatility (IV) derived from vToken supply (x) serves as the primary input for option pricing. The Black-Scholes model is adapted as follows:

$$IV = \frac{C}{x} \tag{20}$$

where C is the calibration constant (Section 2.2.1). For a given strike price K and time to expiry T, call and put prices are calculated:

$$c = S \cdot N(d_1) - K \cdot e^{-rT} \cdot N(d_2) \tag{21}$$

$$p = K \cdot e^{-rT} \cdot N(-d_2) - S \cdot N(-d_1) \tag{22}$$

where:

- S: Underlying asset price (sourced from an external oracle)
- r: Risk-free rate (assumed constant or oracle-fed)
- $N(\cdot)$: Cumulative normal distribution
- $d_1 = \frac{\ln(S/K)+(r+IV^2/2)T}{IV\sqrt{T}}$
- $d_2 = d_1 - IV\sqrt{T}$

The bonding curve's IV ensures that option prices dynamically reflect vToken supply changes.

14.3 Liquidity Allocation

vToken liquidity is allocated across option books for multiple strike prices and expiries within a trading bucket:

- **Bucket Segmentation**: Each bucket corresponds to a specific underlying asset and expiry. Within a bucket, vToken supply (x) is split into sub-pools for different strikes based on moneyness.
- **Volatility Smile Adjustment**: The convexity adjustment (Section 3.2) modifies IV for each strike:
$$IV_{\text{adjusted}} = IV \cdot \left(1 + \alpha \cdot \frac{\partial^2 P}{\partial K^2}\right) \tag{23}$$
- **Reserve Mapping**: USDC reserves (y) back the option book, with collateral requirements calculated as:
$$y_{\text{option}} = \max(c, p) \cdot \text{collateralFactor} \tag{24}$$

This ensures sufficient collateral for both call and put options, maintaining system solvency.

14.4 Smart Contract Implementation

The option book is generated via a dedicated smart contract function that maps vToken liquidity to option prices:

```
function generateOptionBook(uint bucketId, uint[] memory strikes, uint expiry) external
    Bucket storage b = buckets[bucketId];
    uint iv = BASE_IV * (INITIAL_VTOKENS / b.vTokens);

    for (uint i = 0; i < strikes.length; i++) {
        uint strike = strikes[i];
        // Calculate adjusted IV for strike
        uint ivAdjusted = adjustIVForSmile(iv, strike);

        // Black-Scholes pricing
        (uint callPrice, uint putPrice) = blackScholes(
            getSpotPrice(),
            strike,
            ivAdjusted,
            expiry,
            RISK_FREE_RATE
        );

        // Allocate reserves
        uint collateral = max(callPrice, putPrice) * COLLATERAL_FACTOR;
        require(b.usdcReserves >= collateral, "Insufficient reserves");

        // Update option book
        options[bucketId][strike] = Option(callPrice, putPrice, collateral);
    }
}
```

This function ensures that option prices are consistent with vToken liquidity and bonding curve dynamics.

14.5 Trading and Settlement

Options are traded as synthetic assets within the AMM:

- **Buying/Selling**: Users trade options by interacting with the option book, with prices updated based on vToken trades.
- **Settlement**: At expiry, options are settled automatically. Call options pay $\max(S_T - K, 0)$, and put options pay $\max(K - S_T, 0)$, with payouts drawn from USDC reserves.

The non-transferability of vTokens (Section 6.3) ensures that liquidity remains locked during the active phase, supporting stable option trading.

14.6 Advantages

This approach offers:

- **Dynamic Pricing**: Option prices adjust in real-time based on vToken supply and IV.
- **Collateral Efficiency**: Shared USDC reserves across strikes optimize capital use.
- **Scalability**: Multiple buckets support diverse assets and expiries.

15 Bonding Curve Dynamics

15.1 Core Equation

The bonding curve operates on the constant product principle:

$$x \cdot y = k \tag{25}$$

Where:

- x: vToken supply in the trading bucket
- y: USDC reserves backing the bucket
- k: Dynamic invariant, recalculated after fee collection to account for system growth

15.2 Price Impact

The price of a vToken is determined by the marginal change in reserves:

$$P_{\text{vToken}} = \frac{dy}{dx} = \frac{k}{x^2} \tag{26}$$

This ensures that price increases with demand and decreases with supply, aligning with market dynamics.