DeFi: The Mathematics Behind Trustless Finance

Extended Slides – BSc Digital Finance Course

Digital Finance

BSc Digital Finance Course

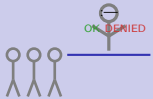
What Will You Be Able to Do After This Lecture?

1. Derive AMM pricing from the constant product invariant and quantify slippage
2. Model utilization-dependent interest rates and liquidation cascades
3. Decompose DeFi yield into sustainable and inflationary components
4. Quantify impermanent loss and model smart contract risk as a jump process
5. Measure governance concentration using Gini and Nakamoto coefficients
6. Evaluate DeFi protocols against the MiCA regulatory framework

---

Six objectives: AMM mechanics (1), lending protocols (2), yield farming (3), risk quantification (4), governance measurement (5), and regulatory mapping (6). Rigorous theory with working code and 12 data visualizations.

Traditional Finance



vs.

The equation doesn't care who you are. That's the point – and the problem.

$$x \cdot y = k$$

Banks discriminate. Equations don't. But equations don't forgive mistakes either.

How Does a Constant Product Market Maker Set Prices Without an Order Book?

Invariant. A pool with reserves (x, y) enforces the constant product rule before and after every swap:

$$x \cdot y = k \quad \text{(invariant, } k > 0\text{)}$$

Marginal price. Differentiating implicitly: $y + x\frac{dy}{dx} = 0$, so:

$$P = -\frac{dy}{dx} = \frac{y}{x}$$

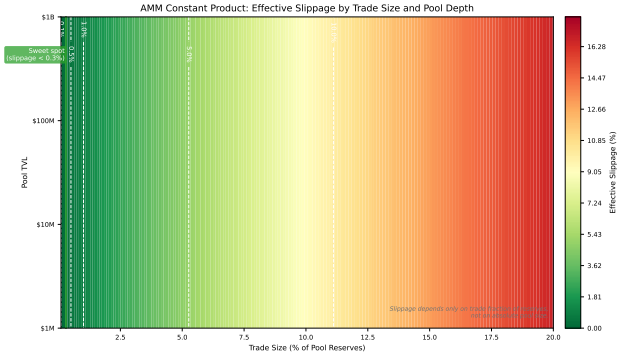
Execution after trade Δx. The buyer sends Δx tokens and receives:

$$\Delta y = \frac{y \cdot \Delta x}{x + \Delta x}, \qquad \text{execution price} = \frac{\Delta y}{\Delta x} = \frac{y}{x + \Delta x}$$

Price impact: $\text{PI} = 1 - \frac{\text{execution price}}{\text{marginal price}} = \frac{\Delta x}{x+\Delta x}$
Kyle's lambda (AMM analogue): $\lambda_{\text{AMM}} = \frac{1}{2\sqrt{k}}$ – the larger the pool, the lower the price impact per unit traded.

The AMM replaces the order book with a deterministic curve. Price is not discovered – it is computed. The trade-off: simplicity and guaranteed liquidity vs. quadratic price impact.

How Does Price Impact Scale with Trade Size and Pool Depth?



- **Green zone (slippage $<$0.3%):** Trades smaller than 0.3% of pool reserves. Most retail swaps operate here.
- **Yellow zone (0.3–1%):** Institutional-sized trades. MEV bots monitor this zone for sandwich attacks.
- **Red zone ($>$5%):** Whale trades or thin pools. Price impact exceeds typical CEX spreads.
- Slippage depends only on trade size as a fraction of reserves, not on absolute pool TVL.
- A **$**1M trade in a **$**100M pool and a **$**10 trade in a **$**1,000 pool both suffer 1% slippage.

Key: AMM slippage is a function of one variable: $\Delta x/x$. Pool depth determines the dollar cost, not the percentage cost.

Constant product slippage is purely geometric: PI $= \Delta x/(x + \Delta x)$. Deep pools do not reduce percentage slippage – they reduce the dollar size at which a given percentage is reached.

How Does Concentrated Liquidity Multiply Capital Efficiency by 10x – or 4000x?

Uniswap V3 virtual reserves. Liquidity is provided only in range $[p_a, p_b]$:

$$x_v = x + \frac{L}{\sqrt{p_b}}, \qquad y_v = y + L\sqrt{p_a}$$

where $L = \sqrt{k}$ is the liquidity parameter.
Capital efficiency multiplier:

$$e = \frac{1}{1 - \sqrt{p_a/p_b}}$$

Example 1 (ETH/USDC, range $1,800–$2,200):

$$\sqrt{1800/2200} = 0.9045, \quad e = \frac{1}{1 - 0.9045} \approx 10.5\times$$

Example 2 (USDC/DAI, range $0.99975–$1.00025):

$$\sqrt{0.99975/1.00025} \approx 0.99975, \quad e = \frac{1}{1 - 0.99975} = 4{,}000\times$$

Note: 4,000x is achievable only for stablecoins with tight pegs. Volatile pairs typically reach 5–20x.

Concentrated liquidity transforms passive LPs into active market makers. The efficiency gain is the inverse of the range width – narrower ranges mean more capital per tick, but higher rebalancing frequency and IL exposure.

Can You Simulate an AMM Swap and Measure the Slippage?

```python
import numpy as np
class ConstantProductAMM:
    def __init__(self, x, y):
        self.x, self.y = float(x), float(y)
        self.k = self.x * self.y
    def price(self):
        return self.y / self.x
    def swap_x_for_y(self, dx):
        """Sell dx of token X, receive dy."""
        new_x = self.x + float(dx)
        new_y = self.k / new_x
        dy = self.y - new_y
        self.x, self.y = new_x, new_y
        return dy
    def slippage(self, dx):
        mid = self.price()
        dy = self.swap_x_for_y(dx)
        slip = 1 - (dy / dx) / mid
        self.x -= dx; self.y += dy # reset
        return slip
pool = ConstantProductAMM(1e6, 2e9)
for pct in [0.1, 1, 5, 10, 20]:
    dx = pool.x * pct / 100
    s = pool.slippage(dx)
    print(f"{pct:5.1f}% -> slip {s:.4%}")
```

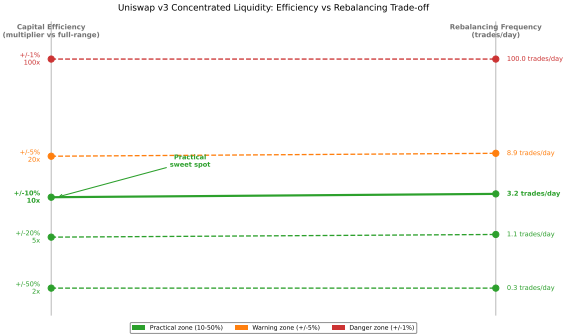
What the code computes

- A pool with 1M token X and 2B token Y (marginal price = 2,000).
- For each trade size (0.1% to 20% of reserves), compute slippage as deviation from midpoint.
- At 0.1%: slippage is $\sim$0.1%. At 20%: slippage reaches $\sim$16.7%.
- The `swap_x_for_y` method updates reserves – sequential trades show path dependence.
- The `slippage` method resets state for repeated measurement without side effects.

Extension: Add a 0.3% fee by reducing Δx before the swap. Fees reduce effective slippage for small trades.

The simulator confirms the analytic formula: PI $= \Delta x/(x + \Delta x)$. Building the simulator from scratch makes the mechanics visceral – students can experiment with pool sizes and fee tiers.

How Does Concentrated Liquidity Reshape the Capital Efficiency Frontier?



- **Slope chart:** Left axis shows capital efficiency multiplier; right axis shows rebalancing frequency required.
- **$\pm$50% range:** 2x efficiency, $<$0.1 rebalances/day. Passive and safe, but capital-inefficient.
- **$\pm$10% range (sweet spot):** $\sim$10x efficiency with $\sim$2 rebalances/day. Practical for active LPs.
- **$\pm$1% range (danger zone):** $\sim$100x efficiency but 30+ rebalances/day. Gas costs and IL can exceed fee income.
- Professional market makers with off-chain hedging dominate narrow ranges. Retail LPs should stay wide.

Key: Capital efficiency is free only in the model. In practice, narrow ranges impose gas costs, monitoring overhead, and amplified impermanent loss.

The efficiency-rebalancing trade-off is the central tension of V3 LP management. The $\pm$10% zone balances capital efficiency against operational burden for most pairs.

What Makes StableSwap Different from Constant Product – and When Does Each Break?

Three bonding curves govern AMM design. Constant product ($xy = k$) guarantees infinite liquidity but with quadratic slippage. Constant sum ($x + y = c$) has zero slippage but drains to one asset at depeg. StableSwap interpolates: flat near peg, curved far away.

	Slippage near peg	**Slippage far**	**Efficiency**	**Best for**
Constant Product ($xy = k$)	High (quadratic)	Moderate	Low	Volatile pairs
StableSwap (Curve)	Near-zero	Cliff	High near peg	Stablecoin pairs
Constant Sum ($x + y = c$)	Zero	Drains pool	Perfect but fragile	Theoretical only

StableSwap invariant: $An^n \sum x_i + D = ADn^n + \frac{D^{n+1}}{n^n \prod x_i}$, where A = amplification coefficient.

- High A flattens the curve near the peg (low slippage for stablecoin swaps). $A = 0$ reverts to constant product.
- **Failure mode:** Depeg causes a "cliff" – slippage explodes from near-zero to extreme in one step. UST/LUNA (May 2022) demonstrated this catastrophically.
- **Capital efficiency:** StableSwap LPs earn 2–5x more fees per dollar vs. constant product for same-peg pairs, but impermanent loss is catastrophic at depeg.

StableSwap is optimal for pegged assets but fragile at depeg. The amplification parameter A is the key design choice: too high risks cliff behavior; too low wastes capital efficiency.

How Do Lending Protocols Set Interest Rates Without a Committee?

Utilization rate. For a pool with total borrows B and total supply S:

$$U = \frac{B}{S}$$

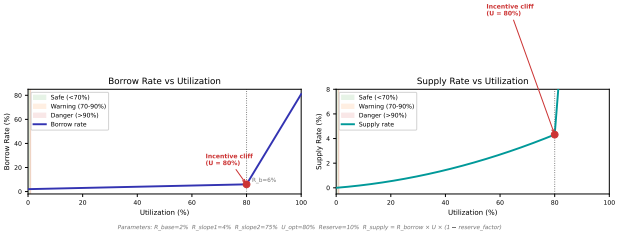
Kink model (Compound/Aave). Two regimes separated by optimal utilization U^*:

$$r(U) = \begin{cases} r_0 + \frac{r_1 - r_0}{U^*} \cdot U & \text{if } U \leq U^* \\ r_1 + \frac{r_2 - r_1}{1 - U^*} \cdot (U - U^*) & \text{if } U > U^* \end{cases}$$

where r_0 = base rate, r_1 = rate at kink, r_2 = rate at 100% utilization.
Typical parameters (Aave V3, USDC): $r_0 = 0\%$, $U^* = 90\%$, $r_1 = 4\%$, $r_2 = 75\%$.
Intuition: Below the kink, rates rise gently to attract borrowers. Above the kink, rates spike to incentivize repayment and attract new deposits. The kink acts as a soft circuit breaker.

The kink model is the DeFi equivalent of a central bank rate corridor. Below the kink: accommodative. Above the kink: punitive. The key design variable is U^* – too low wastes capital, too high risks illiquidity.

What Does the Interest Rate Kink Look Like – and Why Does It Exist?



- **Below kink ($U < 90\%$):** Borrow rates rise linearly from 0% to $\sim$4%. This is the normal operating zone.
- **Above kink ($U > 90\%$):** Rates spike steeply to 75% APR. This penalizes borrowers and attracts emergency deposits.
- **Why the kink exists:** At 100% utilization, depositors cannot withdraw. The spike ensures this never happens in equilibrium.
- **Supply rate:** $r_s = r_b \cdot U \cdot (1 - \text{reserve factor})$. Supply rates are always below borrow rates.
- Protocol revenue = spread between borrow and supply rate $\times$ utilization.

Key: The kink is an automatic stabilizer. When borrowing exceeds 90%, rates become punitive, driving the system back to equilibrium.

The kink model is elegant precisely because it is simple: two slopes, one breakpoint. Governance sets only four parameters. The market does the rest through utilization-driven price discovery.

Can You Model a Liquidation Cascade in Five Lines of Logic?

```python
import numpy as np
def liquidation_cascade(price, positions,
        liq_thresh=1.5, penalty=0.10, depth=1e6):
    """Simulate cascade liquidations."""
    log = [("Initial", price, 0)]
    rnd = 0
    while True:
        rnd += 1; liq_vol = 0
        for coll, debt in positions:
            hf = (coll * price) / debt
            if hf < liq_thresh:
                liq_vol += debt * (1 + penalty)
        if liq_vol == 0:
            break
        price -= liq_vol / depth * price
        log.append((f"Round {rnd}", price, liq_vol))
        positions = [(c,d) for c,d in positions
                     if (c*price)/d >= 1.0]
    return log

pos = [(1.0,1200),(1.0,1300),(1.0,1400),
       (1.0,1500),(1.0,1600)]
for step in liquidation_cascade(2000, pos):
    print(f"{step[0]:>10}: ${step[1]:.0f}")
```

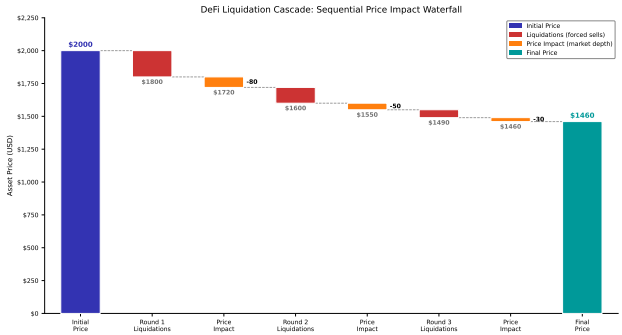
What the code reveals

- Five positions with varying debt levels simulate a heterogeneous borrower pool.
- When price drops, the most leveraged position hits the liquidation threshold first.
- Forced selling pushes price lower, triggering the next tranche – a recursive feedback loop.
- Each round's volume depends on remaining positions and current price.
- The cascade terminates only when no remaining position is undercollateralized.

Extension: Add oracle latency (stale prices cause larger forced-sell volumes) and partial liquidation (50% of position, not 100%).

Liquidation cascades are the DeFi analogue of margin call spirals. The five-line core logic – check health, compute sell volume, update price, repeat – captures the essential positive feedback mechanism.

How Do Liquidation Cascades Amplify a 10% Price Drop into a 40% Loss?



- **Waterfall chart:** Each bar shows a step in the cascade. Red = liquidation volume, orange = market impact.
- **Round 1:** Initial 10% drop triggers **$**200M in forced selling, pushing price down another 4%.
- **Round 2:** New price triggers second-tier liquidations (**$**120M), adding 2.5% impact.
- **Round 3:** Residual positions liquidated (**$**60M), adding 1.5%.
- **Final price:** **$**1,460 – a 27% total drop from a 10% trigger. The cascade amplified the shock by 2.7x.
- Thin markets (low depth parameter) amplify cascades exponentially.

Key: The amplification ratio (total drop / initial drop) depends on leverage concentration and market depth. May 2022: UST depeg triggered a 3.5x amplification cascade across DeFi.

The waterfall makes visible what users experience as a sudden crash. Each round is individually rational (liquidators earn the penalty) but collectively destructive. This is the pro-cyclicality paradox of DeFi lending.

How Does the Health Factor Determine Whether You Get Liquidated?

Health Factor. For a position with collateral value C and outstanding debt D:

$$\text{HF} = \frac{C \cdot \text{LTV}_{\max}}{D}$$

Liquidation occurs when $\text{HF} < 1$. Safe zone: $\text{HF} > 1.5$.
Collateral ratio (CR): $\text{CR} = C/D$. Liquidation threshold: $\text{CR} < 1/\text{LTV}_{\max}$.
How much can the collateral drop before liquidation?

$$\text{Max drawdown} = 1 - \frac{D}{C \cdot \text{LTV}_{\max}} = 1 - \frac{1}{\text{HF}}$$

Safe CR for volatile collateral. With annualized volatility σ and horizon T days:

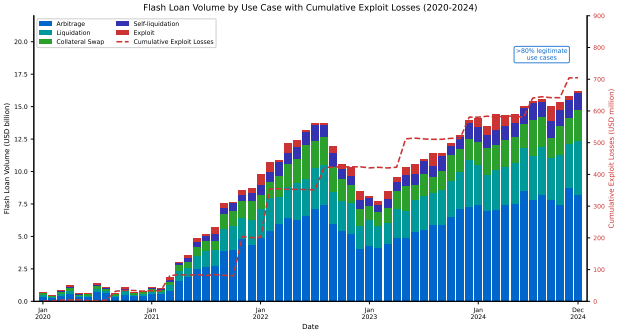
$$\text{CR}_{\text{safe}} \geq \frac{1}{\text{LTV}_{\max}} \cdot \exp\Big(z_\alpha \cdot \sigma \sqrt{T/365}\Big)$$

Example: ETH with $\sigma = 60\%$, 30-day horizon, $z_{0.99} = 2.33$, $\text{LTV}_{\max} = 0.80$:

$$\text{CR}_{\text{safe}} \geq \frac{1}{0.80} \cdot \exp(2.33 \times 0.60 \times \sqrt{30/365}) = 1.25 \times 1.493 \approx 187\%$$

A 187% collateral ratio for ETH means depositing $1,870 to borrow $1,000. Most users over-collateralize to 200%+ for safety. The formula links traditional risk metrics (σ, VaR) to DeFi liquidation mechanics.

How Does Flash Loan Volume Reveal the Hidden Economy of Atomic Arbitrage?



- **Flash loans:** Uncollateralized loans that must be repaid within a single transaction. If not repaid, the entire transaction reverts.
- **Arbitrage:** The dominant use case. Borrow $10M, exploit price discrepancy across DEXs, repay with profit – all in one block.
- **Liquidations:** Flash loans fund liquidation bots that need no starting capital. Borrow, liquidate, sell collateral, repay.
- **Collateral swaps:** Refinance a lending position by swapping collateral atomically without closing the position.
- Flash loan volume spikes during high-volatility events (depegs, liquidation cascades).

Key: Flash loans enable capital-free arbitrage – a concept impossible in traditional finance. They reduce market inefficiency but also power governance attacks and sandwich exploits.

Flash loans are the most alien DeFi primitive: zero-capital, zero-risk arbitrage bounded by transaction atomicity. They compress what takes days in TradFi (borrow, trade, repay) into a single 12-second block.

Where Does DeFi Yield Actually Come From – and When Is It Real?

Yield decomposition. Total DeFi yield for an LP position can be separated into:

$$\text{APY}_{\text{total}} = \underbrace{\text{APY}_{\text{fee}}}_{\text{sustainable}} + \underbrace{\text{APY}_{\text{emission}}}_{\text{inflationary}} + \underbrace{\text{APY}_{\text{leverage}}}_{\text{amplified risk}}$$

Fee yield (sustainable): Directly from trading volume.

$$\text{APY}_{\text{fee}} = \frac{\text{Volume} \times \text{fee rate}}{\text{TVL}} \times 365$$

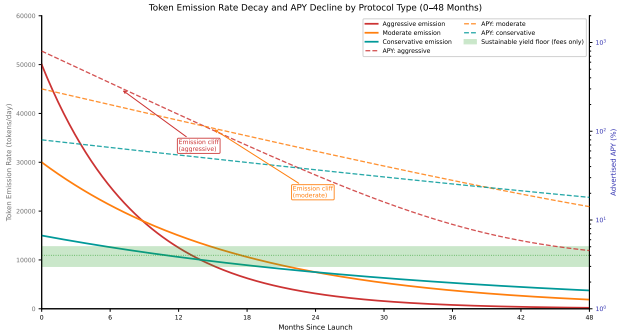
Emission yield (inflationary): Funded by newly minted governance tokens.

$$\text{APY}_{\text{emission}} = \frac{\text{tokens/day} \times \text{token price}}{\text{TVL}} \times 365$$

Sustainability test: If $\text{APY}_{\text{emission}} > \text{APY}_{\text{fee}}$, the protocol is paying more in incentives than it earns. This is a subsidy, not a return.
Terminal yield: When emissions end, $\text{APY}_{\text{total}} \to \text{APY}_{\text{fee}}$. For most DEXs this is 2–8% APY.

The yield decomposition is the single most important DeFi analysis. Emission yield is a marketing cost disguised as a return. When the marketing budget runs out, 90% of yield farms collapse to single-digit APY.

How Do Token Emission Schedules Predict the Yield Collapse?



- **Emission schedule:** Most protocols emit 60–80% of tokens in the first 2 years, then taper to zero by year 4–5.
- **APY decay:** As emission rate drops and TVL grows, emission yield decays hyperbolically.
- **Fee yield floor:** Once emissions end, yield converges to fee-only APY (typically 2–8%).
- **The "yield cliff":** Protocols with steep emission frontloading experience a sharp APY drop 6–12 months after launch.
- Curve and Aave survived the emission taper because their fee yield was already material. Most imitators did not.

Key: Read the token emission schedule before depositing. If 70% of tokens emit in year 1, the yield you see today is temporary.

Every emission schedule is a countdown clock for yield compression. The protocols that survive are those where fee revenue can sustain LP interest after the emissions subsidies end.

Can You Detect a Ponzi-like Protocol from Its On-Chain Cash Flows?

```python
import numpy as np
def ponzi_score(tvl, revenue, emissions, withdrawals):
    """Score 0-1: higher = more Ponzi-like."""
    scores = []
    # 1. Emission/revenue ratio (>5 = red flag)
    if revenue > 0:
        er = min(emissions / revenue, 10)
        scores.append(er / 10.0)
    else:
        scores.append(1.0)
    # 2. Revenue sustainability
    rev_ratio = revenue / max(emissions, 1)
    scores.append(1 - min(rev_ratio, 1.0))
    # 3. Withdrawal pressure
    if tvl[-1] > 0:
        scores.append(min(withdrawals/tvl[-1]/2, 1))
    else:
        scores.append(1.0)
    return np.mean(scores)

tvl = np.linspace(1e8, 5e8, 365)
print(f"Healthy: {ponzi_score(tvl,2e7,5e6,3e7):.2f}")
print(f"Suspect: {ponzi_score(tvl,1e5,8e7,9e7):.2f}")
```

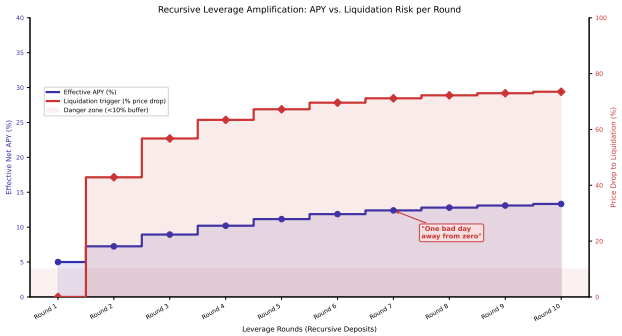
What the code detects

- **Metric 1:** Emission-to-revenue ratio. Healthy protocols earn more than they pay in incentives. Ponzi-like protocols pay 10–100x their revenue.
- **Metric 2:** Revenue sustainability. If fee revenue is a tiny fraction of token emissions, yields depend entirely on new capital.
- **Metric 3:** Withdrawal pressure. If withdrawals exceed TVL, the protocol is in a bank-run state.
- The composite score (0–1) flags protocols where returns are funded by new deposits rather than economic activity.

Extension: Add token price trend (falling price + rising emissions = death spiral) and depositor concentration (whale-dominated TVL is fragile).

The Ponzi score is a heuristic, not a classifier. But the three metrics capture the core pathology: when emissions dwarf revenue and withdrawals pressure TVL, the protocol is paying old investors with new investors' money.

How Does Recursive Leverage Amplify Yield – and Risk – Simultaneously?

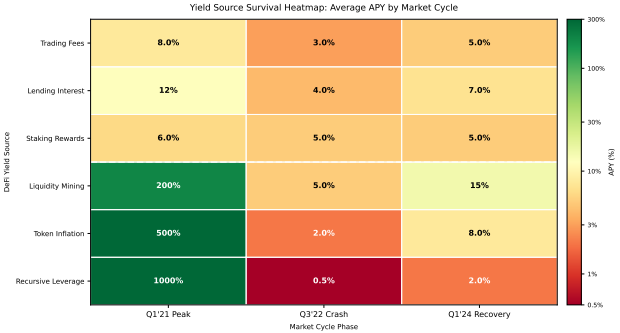


- **Recursive leverage:** Deposit **$**1,000 ETH, borrow **$**750 stablecoin, buy more ETH, deposit again. Repeat n times.
- **Effective leverage:** After n loops with LTV ℓ: total exposure $= \frac{1-\ell^{n+1}}{1-\ell}$.
- **At LTV = 75%, 5 loops:** Effective leverage = 3.8x. Yield amplified from 5% to 19%, but liquidation triggers at a 7% price drop.
- **The trap:** Each loop narrows the margin of safety. The 5th loop adds only 0.24x exposure but brings the liquidation threshold 2% closer.
- Anchor Protocol (Terra) users commonly ran 4–6 loops. When UST depegged, cascading liquidations wiped out **$**40B.

Key: Recursive leverage is a geometric series with a cliff at the end. Each loop adds less return and more risk.

Recursive leverage is the single most common way DeFi users blow up. The yield looks linear in the number of loops; the risk is convex. The last loop always has the worst risk-return ratio.

Which Yield Sources Survived the 2022 Crash – and Which Evaporated?



- **Heatmap:** Rows are yield sources, columns are time periods. Color intensity = APY magnitude.
- **Survivors (green):** Stablecoin lending (Aave/Compound), LP fees on major pairs (ETH/USDC), and real-world asset yields.
- **Casualties (red):** Algorithmic stablecoin yields (UST/Anchor), high-emission farms, and leveraged yield aggregators.
- **Pattern:** Yields backed by real economic activity (trading fees, lending interest) survived. Yields backed by token emissions or reflexive mechanisms collapsed.
- Post-crash DeFi yields converged to 2–6% APY – comparable to traditional fixed income, validating the sustainable floor.

Key: The 2022 crash was a natural experiment in yield sustainability. The survivors are the protocols with genuine product-market fit.

The heatmap is a post-mortem of DeFi yield. Sustainable yield sources barely flickered during the crash. Inflationary sources went to zero. The distinction is now the most important due diligence question in DeFi.

Why Does Impermanent Loss Follow a Square Root Law?

Setup. LP deposits equal value of token A and token B. Price of A relative to B changes by ratio $r = P_1/P_0$.
LP value after price change. From constant product, when price ratio moves to r:

$$V_{\text{LP}} = V_0 \cdot \sqrt{r}, \qquad V_{\text{hold}} = V_0 \cdot \frac{1+r}{2}$$

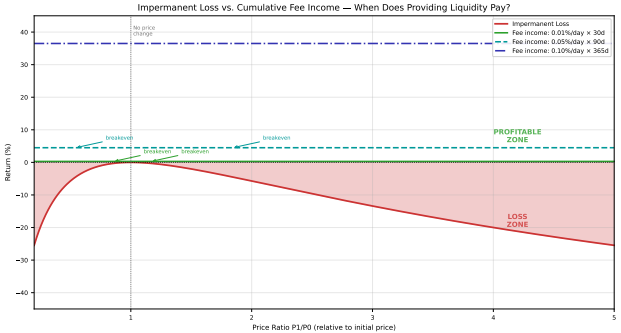
The LP position scales as $\sqrt{r}$ (geometric mean of reserves); the hold position is the arithmetic average.
Impermanent loss:

$$\text{IL}(r) = \frac{V_{\text{LP}}}{V_{\text{hold}}} - 1 = \frac{2\sqrt{r}}{1+r} - 1$$

Taylor expansion around $r = 1$: $\text{IL} \approx -\frac{(r-1)^2}{8}$ for small deviations.
Key values: $r = 1.25 \to \text{IL} = -0.6\%$, $r = 2 \to \text{IL} = -5.7\%$, $r = 5 \to \text{IL} = -25.5\%$.

Impermanent loss is always negative – the LP always underperforms holding. The square root law means IL is quadratic in price deviation: a 2x move costs 5.7%, but a 5x move costs 25.5%. Fees must exceed IL for the LP to profit.

How Does Impermanent Loss Compare to Fee Income Across Price Scenarios?



- **Filled area chart:** Red area = impermanent loss magnitude. Green area = cumulative fee income at different volume/TVL ratios.
- **Breakeven line:** Where fee income exactly offsets IL. Below this line, LPs lose money relative to holding.
- **High-volume pools (ETH/USDC):** Fees exceed IL for price moves up to $\pm 40\%$. LPs profit in most scenarios.
- **Low-volume pools:** Fees cover only $\pm 10\%$ moves. Any meaningful price change makes the LP position worse than holding.
- The breakeven analysis explains why 50% of Uniswap V3 positions lose money (Loesch et al., 2021).

Key: IL is the cost of providing liquidity. Fees are the revenue. The LP business is profitable only when revenue exceeds cost – which depends on volume, not TVL.

The IL-vs-fee chart is the P&L statement for liquidity provision. Most LPs focus on APY (fee revenue) and ignore IL (cost of goods sold). The profitable LPs are those who understand both sides of the ledger.

Can You Build a Monte Carlo Simulator for LP Profitability?

```python
import numpy as np
def mc_lp_profit(sigma, fee_apy, days=365,
                 n_sims=10000, seed=42):
    """Monte Carlo LP profitability.
    sigma: annual vol; fee_apy: fee yield."""
    rng = np.random.default_rng(seed)
    dt = 1 / 365
    daily_vol = sigma * np.sqrt(dt)
    log_r = rng.normal(-0.5*daily_vol**2,
                       daily_vol,
                       (n_sims, days))
    r = np.exp(np.cumsum(log_r, axis=1))
    il = 2*np.sqrt(r[:,-1])/(1+r[:,-1]) - 1
    fees = fee_apy * days / 365
    net = il + fees
    return net

net = mc_lp_profit(sigma=0.8, fee_apy=0.15)
win = np.mean(net > 0) * 100
print(f"Win rate: {win:.1f}%")
print(f"Mean return: {np.mean(net):.2%}")
print(f"5th pctile: {np.percentile(net,5):.2%}")
```

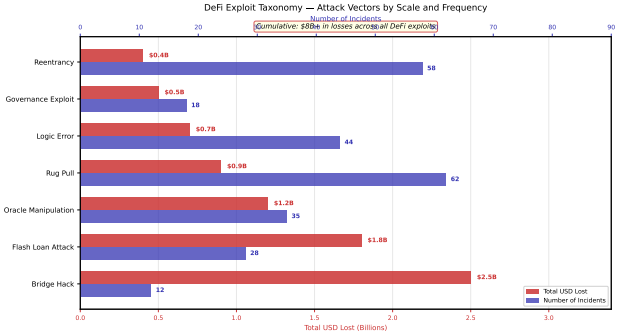
What the simulator reveals

- **Input:** 80% annual vol (typical for ETH pairs), 15% fee APY.
- **Win rate:** $\sim$62% of paths end profitable vs. holding. Not a sure bet.
- **Mean return:** $\sim+5\%$, but right-skewed: large losses in tail scenarios offset by modest gains in most.
- **5th percentile:** $\sim-18\%$. In bad scenarios, IL dominates fees completely.
- The Monte Carlo approach captures what analytic IL formulas miss: path dependence and compounding effects over time.

Extension: Add concentrated liquidity (range-dependent IL amplification), gas costs for rebalancing, and MEV losses from sandwich attacks.

The Monte Carlo simulator turns IL from a formula into a distribution. The key output is not the mean but the win rate and the left tail – how often and how badly can an LP position underperform simple holding?

How Much Value Have Smart Contract Exploits Destroyed – and What Are the Patterns?



- **Reentrancy:** The DAO hack (2016), Cream Finance (2021). Attacker calls back into the contract before state updates.
- **Oracle manipulation:** Mango Markets ($114M, 2022). Attacker inflates collateral price to drain lending pools.
- **Flash loan attacks:** Beanstalk ($182M, 2022). Borrow, manipulate, profit, repay – all in one transaction.
- **Bridge exploits:** Ronin ($625M), Wormhole ($320M). Cross-chain bridges are the weakest link.
- **Governance attacks:** Beanstalk governance takeover. Flash-borrow governance tokens, pass malicious proposal.

Pattern: Bridge exploits account for 50%+ of total losses. Composability is the attack surface – more integrations mean more entry points.

Over $6B lost to smart contract exploits since 2020. The taxonomy shows that DeFi security is not a single problem but five distinct failure modes, each requiring different defenses (formal verification, oracle design, bridge architecture).

How Do You Model Smart Contract Risk as a Jump Process?

Standard GBM is insufficient. Smart contract exploits cause sudden, discontinuous value loss – not gradual decline. We need a jump-diffusion model.

Merton jump-diffusion for DeFi protocol value V:

$$\frac{dV}{V} = \mu\, dt + \sigma\, dW + J\, dN$$

where dW = Brownian motion (normal vol), dN = Poisson process with intensity λ (exploit arrival rate), J = jump size (loss severity, $J < 0$).

Exploit arrival rate: From historical data, $\lambda \approx 0.02$ per protocol per month (one exploit every 50 months on average for audited protocols).

Jump size distribution: Log-normal with mean loss $\mu_J = -0.30$ (30% of TVL) and $\sigma_J = 0.50$.

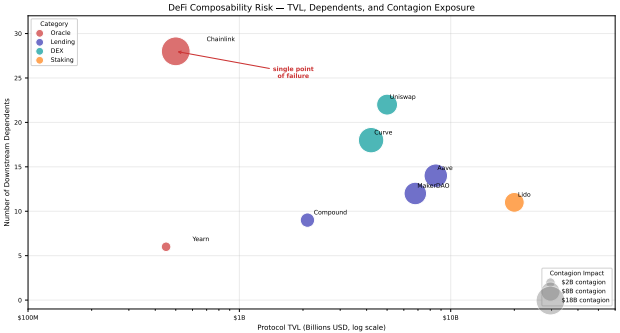
Expected annual loss from exploits:

$$\text{EL} = \lambda \cdot \mathbb{E}[|J|] \cdot 12 = 0.02 \times 0.30 \times 12 = 7.2\% \text{ per year}$$

Implication: A protocol offering 10% APY has a risk-adjusted yield of only 2.8% after accounting for expected exploit losses.

The jump-diffusion model makes smart contract risk quantifiable. The expected annual loss of $\sim$7% is the "insurance premium" that DeFi yields must exceed to compensate for exploit risk. Most protocols fail this test.

How Does Protocol Composability Create Hidden Contagion Pathways?



- **Bubble scatter:** Each bubble is a DeFi protocol. Size = TVL. Position = composability degree (x) vs. exploit exposure (y).
- **High composability, high exposure:** Protocols integrated with many others (Curve, Aave) have more contagion pathways but also more security scrutiny.
- **Low composability, high exposure:** Isolated protocols with poor security (small bridges, new forks). Highest risk per TVL.
- **Contagion:** When Curve was exploited (Jul 2023), pools across Convex, Yearn, and Frax were affected. One vulnerability, multiple protocols.
- The composability graph is DeFi's systemic risk map. Highly connected nodes are "too integrated to fail."

Key: Composability is DeFi's superpower and its Achilles heel. The same interoperability that enables innovation also enables contagion.

Composability creates network effects for both value and risk. The scatter plot reveals that the most systemically important protocols (high composability, high TVL) are also the most dangerous single points of failure.

How Concentrated Is Voting Power in Decentralized Governance?

Nakamoto coefficient. The minimum number of entities N_c that collectively control >50% of voting power:

$$N_c = \min\Big\{k : \sum_{i=1}^{k} s_{(i)} > 0.50\Big\}$$

where $s_{(1)} \geq s_{(2)} \geq \ldots$ are voting shares sorted descending.
Gini coefficient for voting power:

$$G = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}|s_i - s_j|}{2n\sum_{i=1}^{n} s_i}$$

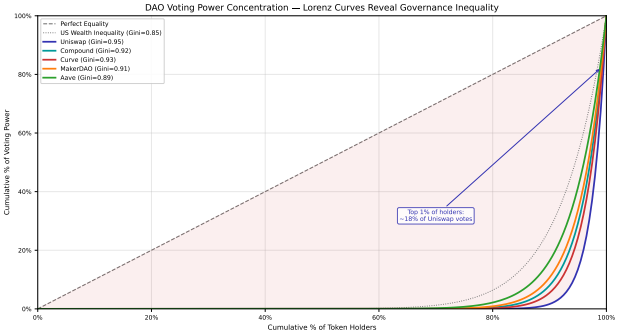
Typical values: Uniswap $G = 0.95$, $N_c \approx 4$. Compound $G = 0.92$, $N_c \approx 6$. For comparison, US wealth inequality: $G \approx 0.85$.
Flash loan governance attack cost:

$$\text{Cost}_{\text{attack}} = \text{flash loan fee} \times \text{tokens needed for quorum} \times \text{token price}$$

Example (Beanstalk, April 2022): Flash loan fee = 0.09%. Tokens needed: 67% quorum of 67M tokens at \$1.50. Cost: \$0.09% × 44.9M × \$1.50 = \$60,615. Attacker stole \$182M.

DAOs are more concentrated than the traditional financial system they aim to replace. Flash loan attacks make governance capture absurdly cheap – Beanstalk was taken over for \$60K to steal \$182M.

How Does Voting Power Distribution Compare Across Major DAOs?



- **Lorenz curves:** The diagonal = perfect equality. The further a curve bows from the diagonal, the more concentrated the power.
- **All DAOs are worse than US wealth inequality** (dashed gray). "Decentralized" governance is more concentrated than one of the most unequal economies.
- **Top 1% of Uniswap holders** control ∼37% of voting power. Top 10 addresses can pass any proposal.
- **Aave (Gini=0.89):** The "least bad" major DAO, partly due to delegation and safety module staking.
- **Delegation helps:** Protocols with active delegation programs have lower effective Gini (voters delegate to informed representatives).

Key: Lorenz curves expose the gap between the marketing narrative ("governed by the community") and reality (governed by 5–10 whales).

The Lorenz curve comparison is the most damning chart in DeFi governance. Every major DAO exhibits more concentration than traditional wealth inequality. Decentralization is aspirational, not actual.

Can You Compute the Cost of a Flash Loan Governance Attack?

```python
def governance_attack_cost(
    total_supply, token_price,
    quorum_pct=0.10, flash_fee=0.0009,
    proposal_type="standard"):
    quorum = total_supply * quorum_pct
    borrow_cost = quorum * token_price * flash_fee
    if proposal_type == "timelock":
        hold_cost = quorum * token_price
        return {"method": "spot_buy",
                "cost": hold_cost,
                "feasible": hold_cost < 1e8}
    return {"method": "flash_loan",
            "cost": borrow_cost,
            "feasible": True}

protocols = {
    "Uniswap": (1e9,  7.50, 0.04, "timelock"),
    "Compound": (1e7, 50.00, 0.04, "timelock"),
    "Beanstalk":(67e6, 1.50, 0.67, "standard"),
}
for name, (sup, px, q, t) in protocols.items():
    r = governance_attack_cost(sup, px, q,
                               proposal_type=t)
    print(f"{name:>10}: ${r['cost']:>12,.0f}"
          f"  ({r['method']})")
```

What the code computes

- **Beanstalk (standard):** Flash loan attack costs ∼$60K. No timelock, no delay. Attacker profits $182M.
- **Uniswap (timelock):** Flash loans are useless – tokens must be held across a 2-day delay. Spot-buy cost: ∼$300M. Economically infeasible.
- **Compound (timelock):** Similar protection. Cost: ∼$20M. Expensive but not impossible for a state actor.
- The key defense: timelocks force attackers to hold tokens, eliminating flash-loan-funded attacks.

Extension: Model vote delegation – if delegated votes can be redirected, effective quorum cost is lower than nominal supply suggests.

Timelocks are the key governance defense. Without them, any protocol with liquid governance tokens on a lending market is vulnerable to a flash loan takeover costing 0.09% of the quorum.

What Does MiCA Actually Regulate – and What Escapes Through the Cracks?

MiCA Category	Definition	Requirements	DeFi Examples	Status
E-Money Token (EMT)	Pegged to one fiat currency	Banking license, 1:1 reserves, audit	USDC, EUROC	Regulated
Asset-Ref. Token (ART)	Pegged to basket/commodity	White paper, 2% reserve buffer, governance rules	DAI (partially), PAXG	Regulated
Utility Token	Access to a service	White paper, no reserve mandate	UNI, AAVE, COMP	Light touch
CASP (Service Provider)	Exchange, custody, brokerage	License, AML/KYC, capital requirements	Uniswap Labs, Aave front-end	Regulated if centralized
Truly decentralized	*No identifiable issuer or operator*	*Exempt under recital 22*	*Uniswap contracts, Bitcoin*	*Unregulated*

The loophole: MiCA exempts "fully decentralized" protocols with no identifiable service provider. But Uniswap has a legal entity (Uniswap Labs), a front-end, and a governance token – making the exemption ambiguous.
The enforcement gap: Smart contracts are borderless; MiCA is EU-only. A protocol can remove its EU front-end while the contracts remain accessible globally via alternative interfaces.

MiCA is the world's first comprehensive crypto regulation (effective June 2024). It clarifies stablecoin rules but leaves DeFi in a gray zone. The "decentralization exemption" will be the most litigated clause in crypto law.

Can You Score How Decentralized a Protocol Actually Is?

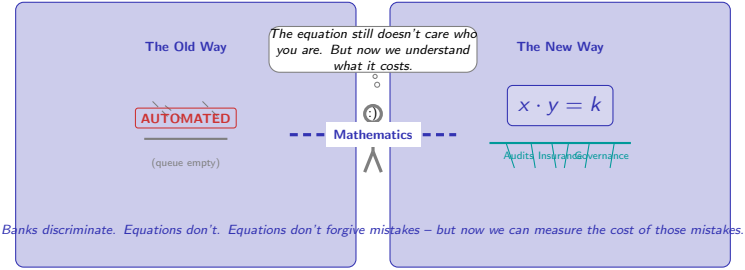
```python
import numpy as np
def decentral_score(nakamoto, admin_keys,
    open_src, upgradeable, frontends, validators):
    """Score 0-100: higher = more decentralized."""
    s = min(nakamoto / 20 * 30, 30) # governance
    if admin_keys == 0: s += 20   # admin risk
    elif admin_keys <= 3: s += 10
    if open_src: s += 10          # open source
    if not upgradeable: s += 20   # immutability
    elif admin_keys == 0: s += 10
    s += min(frontends / 5 * 10, 10) # front-ends
    s += min(validators/100 * 10, 10)# validators
    return min(s, 100)

protocols = {
    "Uniswap V2": (4, 0, True, False, 8, 0),
    "Aave V3":   (6, 5, True, True, 3, 0),
    "Curve":     (3, 1, True, False, 5, 0),
}
for name, args in protocols.items():
    sc = decentral_score(*args)
    print(f"{name:>12}: {sc}/100")
```

What the score captures

- **Six dimensions:** Governance distribution, admin key risk, open source, immutability, front-end diversity, validator diversity.
- **Uniswap V2** scores highest: immutable contracts, no admin keys, 8+ front-ends. Once deployed, nobody can change it.
- **Aave V3** scores lower: upgradeable proxies and 5 admin keys. More features, less decentralization.
- **Curve:** Non-upgradeable but concentrated governance (Nakamoto coeff = 3).
- The score operationalizes what "decentralized" means beyond marketing claims.

Extension: Weight dimensions by MiCA relevance (identifiable operators and upgradeability matter most).

Decentralization is not binary – it is a spectrum across multiple dimensions. The score converts subjective claims into measurable attributes. Regulators could use similar frameworks to determine which protocols qualify for the MiCA exemption.


The Old Way
The equation still doesn't care who you are. But now we understand what it costs.
The New Way
AUTOMATED
(queue empty)
Mathematics
$x \cdot y = k$
Audits Insurance Governance
Banks discriminate. Equations don't. Equations don't forgive mistakes – but now we can measure the cost of those mistakes.