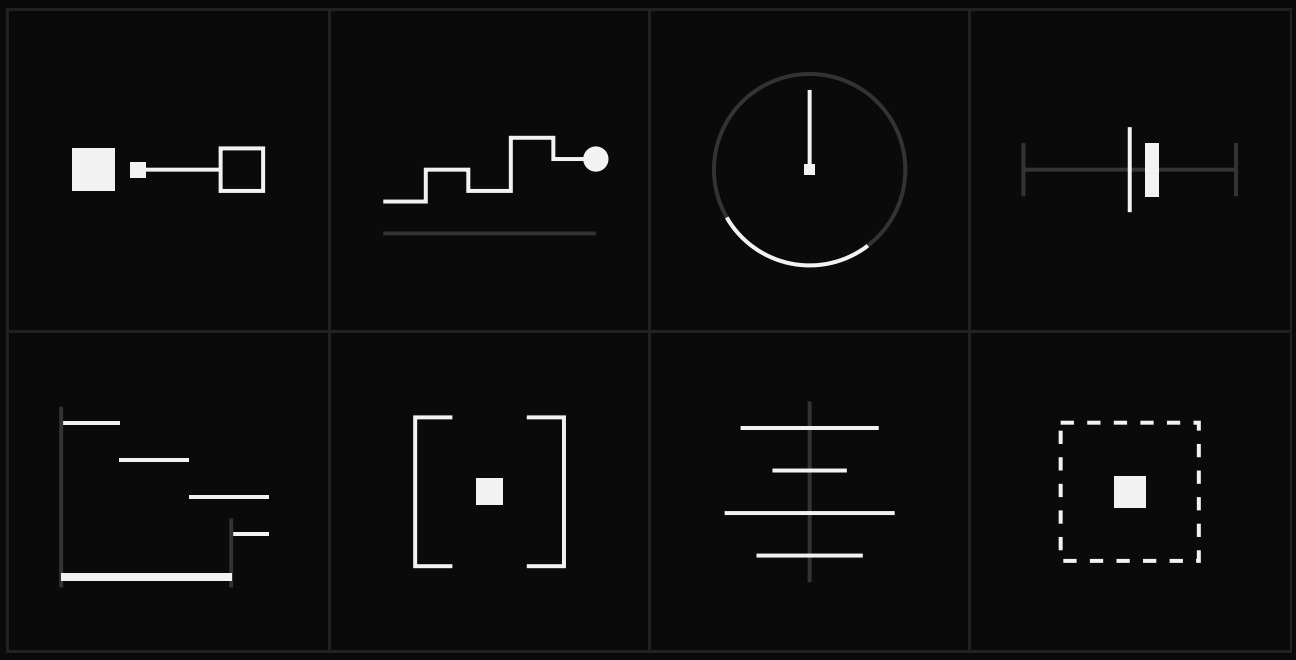
z poke



Whitepaper

A proprietary automated market maker for tokenized equities: why passive curves fail the asset, a four-term quote with proofs, signed-quote settlement that binds the maker on chain, and a measured comparison.

Version 1.0 · September 2026

ABSTRACT

Tokenized equities trade on a chain that never closes, while the shares they track trade six and a half hours a day. On-chain liquidity for these tokens today is almost entirely passive and, for most names, smaller than a single retail order. We show that this is a consequence of the mechanism: a constant-product pool charges a taker $f + 2N/V$, is stale by up to its fee, pays $\sigma^2 V/8$ a year to be corrected, and cannot know whether the underlying market is open.

z poke is a proprietary automated market maker built for that asset. Its reference is the price of the underlying share from Pyth's signed equity feeds, verified and converted into the token's units by the settlement program itself. Its quote is a half-spread of published terms, for the instrument, the order's size, the session of the underlying market and the stated uncertainty of the reference, around a centre that the pool's inventory moves, computed in integer arithmetic with proved ordering, monotonicity, bounded skew, and no profitable round trip. The maker commits to exact amounts in a 509-byte signed message, and an onchain program settles both legs atomically and exactly once, re-checking at execution every condition the quote was priced under and enforcing a price band, size and turnover limits that bind the maker itself. We specify the message and its verification, characterise the maker's risk, report measurements, and compare the design, with the live pools' parameters and fee but without the confidence term, with a constant-product pool of equal capital on a common price path. It tracks its reference four to eleven times more closely in every scenario. Against a curve charging 30 basis points it is about two and a half to three times cheaper for the taker; a curve with a fee under about 9 basis points is cheaper at the live order sizes.

Version 1.0 runs this design on ten markets, NVDAx, SPYx, QQQx, GLDx, TSLAx, AAPLx, MSFTx, METAx, COINx and CRCLx, from pools holding about 32,000 USDC of value and owned by their liquidity providers through on-chain shares. On those pools quotes are given only while a US equity session is open, the session term is set to zero, orders are capped at 100 USDC on NVDAx and 20 USDC on the others, and inventory is not hedged. The other 31 of the 41 listed tokens trade through the same terminal by routing to Jupiter.

CONTENTS

The problem

1 Introduction

2 The liquidity that is actually there

3 What a curve costs

4 Equities keep a calendar

The quote

5 The quote

6 Where the terms come from

7 Inventory and the maker's risk

8 The reference price

9 Units: tokens, shares and the multiplier

Settlement

10 Settlement: one signed quote, one transaction

11 The program: accounts and authorities

12 Token-2022 and the issuer's powers

The system

13 The off-chain system

14 Routing and distribution

15 Security analysis

16 Performance

Evaluation

17 Comparison

18 Economics

19 Risks and disclosures

20 Admission, growth and conclusion

Appendices

A Notation

B Markets

C Program errors

D Formats

E References

1

Introduction

Tokenized equities put a share's economics on a chain that never closes. That is a genuine advance in who can hold an asset and when it can move. It has not yet produced a market in which those tokens can be traded at a fair price, and the reason is structural.

1.1 Two clocks

A tokenized stock is a bearer instrument on a public ledger. It can be transferred at any second of any day, held in self-custody, used as collateral, and settled with finality in about a second. The share it tracks has none of these properties. It trades on an exchange in New York for six and a half hours on weekdays, with thinner sessions either side, and not at all at night, at weekends or on holidays.

Every difficulty in making a market in these tokens comes from running those two clocks against each other. While the exchange is open, the token has an obvious fair value and anyone holding the wrong amount of it can correct their position in seconds. When the exchange is shut, neither is true. The fair value is a belief rather than an observation, and a position taken on is a position carried.

1.2 What exists today

On Solana these tokens trade almost entirely on passive automated market makers: pools in which anyone deposits the token and USDC and a formula sets the price. Chapter 3 derives what such a pool costs a taker and what it loses to arbitrage, and shows that both are governed by the amount of capital in the pool. Chapter 2 measures that capital. For most listed names it is smaller than a single retail order.

Meanwhile the most successful venues for crypto pairs on Solana are no longer passive at all. Proprietary market makers, quoting from their own inventory and repricing from external data many times a second, now account for a large share of spot volume on the chain. They exist because a quote that is told the price beats a curve that has to be shown it. Nothing about that argument is specific to crypto, and for an asset with a closing bell it is considerably stronger.

1.3 What z poke is

z poke is a proprietary automated market maker for tokenized equities and ETFs. Version 1.0 runs it on ten markets, NVDAx, SPYx, QQQx, GLDx, TSLAx, AAPLx, MSFTx, METAx, COINx and CRCLx, each a pool of the token and USDC held in on-chain vaults and owned by its liquidity providers through shares. It prices each request from the price of the underlying share, signed by Pyth and verified on chain, together with the size requested, the state of the underlying market, the stated uncertainty of that price, and the pool's inventory. It quotes only while the underlying market is in an open session. It commits to exact amounts with a signature, and an onchain program settles both legs atomically, enforcing limits that bind z poke itself. A taker sees the exact amounts, the fee and the price's distance from the reference before signing, receives exactly what was shown or nothing, and keeps custody throughout. The other 31 of the 41 listed tokens trade through the same terminal by routing to Jupiter, at the route's own prices.

Reference	Session	Pool	Engine	Signer	Ledger
Pyth, verified on chain	the New York clock	inventory against target	integer arithmetic	exact amounts, up to 10 s	atomic, exactly once

1.4 Contributions and plan of the paper

The paper makes five claims and supports each.

1. Passive curves are structurally unsuited to tokenized equities, and the gap can be quantified (Chapters 2 to 4).
2. A quote with explicit session, confidence and inventory terms is well defined in integer arithmetic, monotone in size, bounded in skew, and cannot be round-tripped for profit (Chapter 5). Each term has an economic derivation from which it can be re-estimated (Chapter 6).
3. The maker's risk under that quote is characterised and bounded by inventory mean reversion (Chapter 7).
4. Signed-quote settlement onchain gives exactly-once, exact-amount execution with a verification template that closes the known signature-offset attacks, against a reference the program verifies from a signed price report rather than one an operator key publishes, and bounds the damage of a compromised maker key on chain (Chapters 8, 10 to 12, 15).
5. On the same price path, the quote tracks its reference four to eleven times more closely than a constant-product pool holding the same capital. Whether it is also cheaper depends on the curve's fee: before the confidence term, it costs the taker less than half as much as a curve charging 30 basis points, and more than one charging 5 at the live order sizes (Chapter 17).

Chapters 8, 9 and 13 cover the reference price, units, and the off-chain system. Chapter 14 covers routing and distribution. Chapters 18 to 20 cover economics, risks, and how markets are admitted and the design extends. Appendices give notation, parameters, the markets, and the byte-level formats.

2

The liquidity that is actually there

Arguments about market structure are easier to have with numbers. This chapter reports the on-chain liquidity available for each of the forty-one tokens z poke lists, as an aggregator sees it, and draws the consequence.

2.1 Measurement

For each token we record the liquidity figure reported by the Jupiter price service, which aggregates the pools it can route through. The observation was made on 2026-09-20 20:47 UTC. It is a snapshot, it depends on one provider's definition of liquidity, and it will be different tomorrow. The distribution, however, is not subtle.

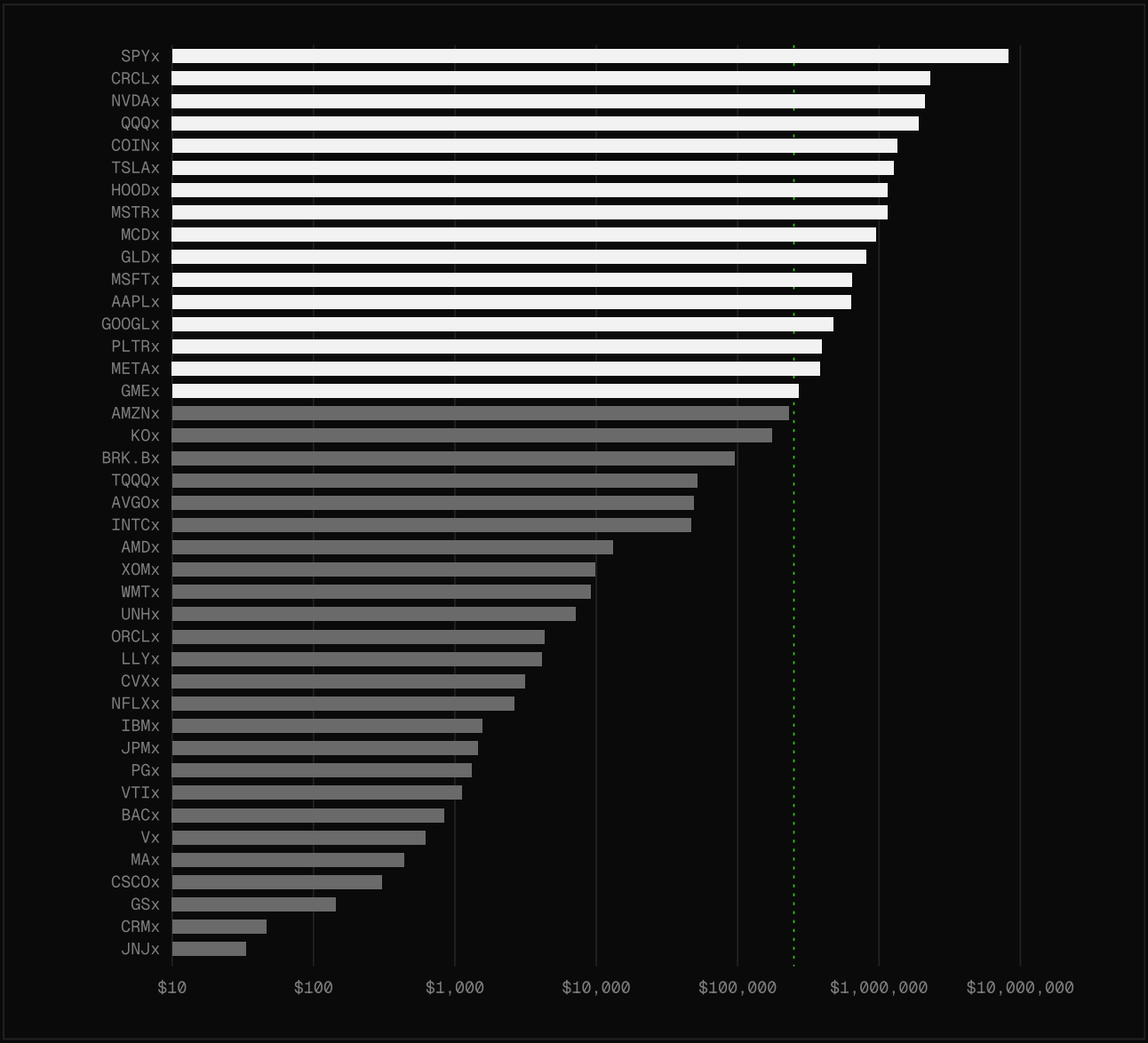


Figure 2.1 On-chain liquidity for each listed token, in USDC, on a logarithmic axis from $10 to $10,000,000. The rule marks 250,000 USDC, the threshold above which the public reference used for Quick and the charts accepts a token's own on-chain price (Section 8.6); ZPOKE prices from the underlying share instead. Source: Jupiter Price API v3, liquidity field, 2026-09-20 20:47 UTC.

Table 2.1 Summary of Figure 2.1.

Statistic	Value
Tokens observed	41
Total liquidity across all of them	$24,659,038
Share of that total in the five deepest	64%
Median	$49,087
Tokens under 250,000	25
Tokens under 100,000	23
Tokens under 10,000	18
Shallowest	JNJx, $33

2.2 Reading it

The shallowest token in the table tracks a company worth $651 billion. Its on-chain liquidity would not cover a restaurant bill. This is not a criticism of anyone: it is what equation (3.3) predicts. Passive liquidity is capital that earns a fee in exchange for bearing arbitrage losses, and outside a handful of names the fee income is not there to attract it. The capital goes where the volume is, the volume goes where the prices are good, and for most of the list neither has arrived.

Put the median into equation (3.2). With V equal to the median above, a 1,000 USDC order moves a constant-product pool by roughly

$$\frac{2N}{V} = \frac{2 \times 1{,}000}{49{,}087} \approx 4.1\% \tag{2.1}$$

before any fee. A venue on which a thousand-dollar order costs several percent is not a market in any useful sense; it is a price display. The quote of Chapter 5, with the parameters z poke 's live pools use, prices the same order at a half-spread of 9 basis points at target, plus a five basis point fee and a confidence term of twice the reference's relative uncertainty, 2 basis points when that uncertainty is 1 basis point. The live pools accept at most 100 USDC per order on NVDAx and 20 USDC on the others, and quote a 100 USDC order at 5.4 basis points before the confidence term.

What this chapter establishes

Outside the few deepest names, on-chain liquidity for tokenized equities is two to five orders of magnitude short of what a passive curve needs to price a retail order competitively. The shortfall is a consequence of the mechanism, not of the market being young.

3

What a curve costs

Most on-chain liquidity for tokenized stocks sits on constant-product and concentrated-liquidity curves. This chapter derives what such a pool charges a taker, what it loses to arbitrage, and how much capital it needs to be competitive. The results set the bar the z poke quote has to clear.

3.1 The constant-product pool

A constant-product pool holds x tokens and y dollars and lets anyone trade along the curve $xy = L^2$. Its marginal price is the ratio of its reserves, and both reserves are functions of that price alone:

$$P = \frac{y}{x}, \qquad x(P) = \frac{L}{\sqrt{P}}, \qquad y(P) = L\sqrt{P}, \qquad V(P) = xP + y = 2L\sqrt{P}. \tag{3.1}$$

Half the pool's value is always in each asset. A taker who pays N dollars, of which a fraction f is kept as the pool fee, receives $\Delta x = x\,\frac{N(1-f)}{y+N(1-f)}$ tokens. The average price paid, relative to the marginal price, is therefore

$$\frac{N/\Delta x}{P} - 1 \;=\; \frac{1}{1-f}\left(1 + \frac{N(1-f)}{y}\right) - 1 \;\approx\; f + \frac{2N}{V}. \tag{3.2}$$

The approximation holds for small f and $N \ll V$, and it is the whole story of curve pricing: a fixed fee, plus an impact term that depends only on the ratio of the order to the pool. It does not depend on volatility, on the time of day, or on anything the pool could know about the asset. Figure 3.1 plots (3.2) for a pool of two million dollars, the capital given to each venue in the model comparisons of Chapter 17.

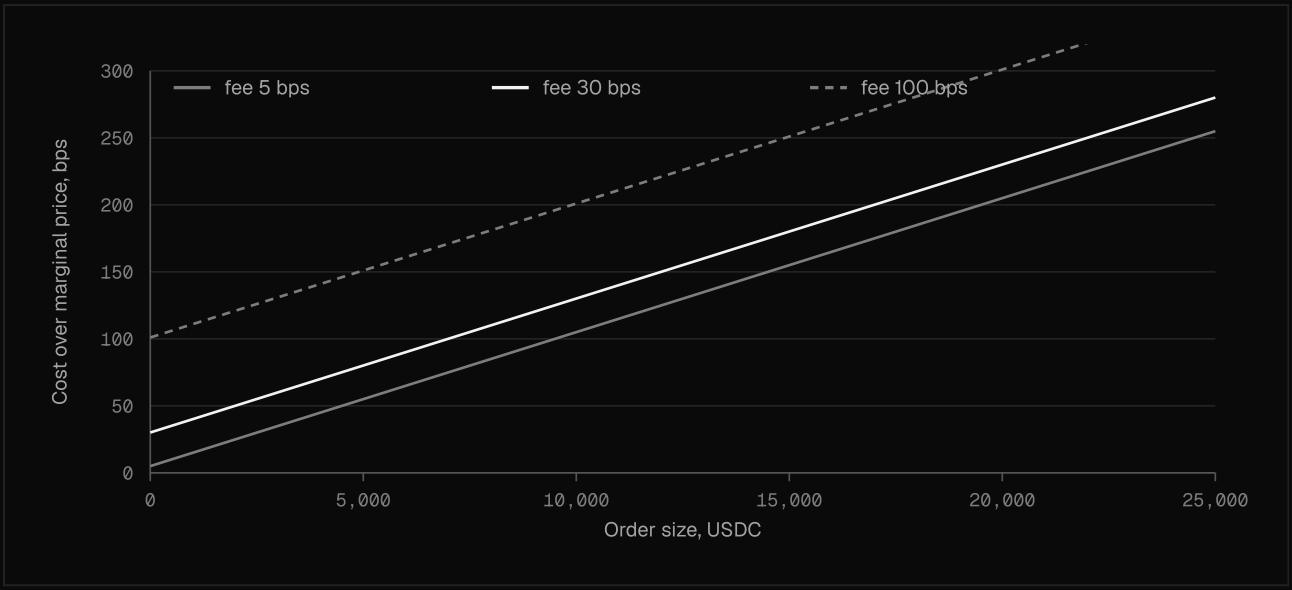


Figure 3.1 Cost of a buy on a constant-product pool with 2,000,000 USDC of total value, from equation (3.2), for three fee tiers. The slope is 2/ V : one basis point per 100 USDC.

> **Proposition 3.1. Capital needed to match a quoted spread**
>
> For a constant-product pool to fill an order of N dollars at a total cost of at most c, with pool fee $f < c$, its total value must satisfy
>
> $$V \;\geq\; \frac{2N(1-f)}{(1+c)(1-f)-1} \;\approx\; \frac{2N}{c-f}. \tag{3.3}$$

> **Proof.**
>
> Set the left side of (3.2) to at most c and solve for $y = V/2$. ■

The number is sobering. To fill 10,000 USDC at 15 basis points all-in with a 5 basis point fee, a pool needs $V \approx 2 \times 10^4/10^{-3} = 2 \times 10^7$ dollars: twenty million, per market. Forty-one markets would need over eight hundred million dollars of passive capital to offer that price. Chapter 2 shows how far real pools are from that figure.

3.2 Concentration does not remove the cost, it moves it

Concentrated liquidity restricts a position to a price range $[P_a, P_b]$. Inside the range the position behaves like a constant-product pool with virtual reserves, so the same capital supports a larger L. For a symmetric range $[P/r,\ Pr]$ the amplification over a full-range position is

$$A(r) = \frac{1}{1 - r^{-1/2}}, \qquad A(1.05) \approx 41.5, \qquad A(1.01) \approx 201. \tag{3.4}$$

Impact falls by the factor A while the price stays in range. Two things follow that matter for equities. First, every loss that scales with L, including the arbitrage loss derived below, is amplified by the same factor per dollar of capital. Second, a stock that gaps through the range at the open leaves the position holding only the asset that fell, quoting nothing, until someone re-centres it. Re-centring is

active market making. A provider who does it well has rebuilt a worse version of the system this paper describes: one that adjusts its quote only by paying gas to move a range, and only after the price has already left.

3.3 A curve reprices by being arbitraged

A curve has no input other than trades. When the reference price moves from P to P', the pool still offers P until a trader moves it. With a fee f, that trade is profitable only once the gap exceeds the fee, so the pool's marginal price P_{pool} is confined to a band around the reference rather than pinned to it:

$$\frac{P_{\text{ref}}}{1+f} \;\le\; P_{\text{pool}} \;\le\; P_{\text{ref}}\,(1+f). \tag{3.5}$$

Inside the band the pool is simply stale, and a taker pays or receives the stale price. A larger fee protects the pool and widens the band by exactly as much. Outside the band, the correction is made by an arbitrageur at the pool's expense. The cleanest way to measure that expense is loss-versus-rebalancing (Milionis, Moallemi, Roughgarden and Zhang, 2022): the shortfall of the pool against a portfolio that holds the same quantities but rebalances at the reference price.

> **Proposition 3.2. Arbitrage loss of a constant-product pool**
>
> Let the reference follow $dP = \sigma P\,dW$ and let arbitrage be frictionless. The pool loses to arbitrageurs at the rate
>
> $$\frac{d\,\text{LVR}}{dt} \;=\; \tfrac{1}{2}\sigma^2 P^2\,|x'(P)| \;=\; \frac{\sigma^2}{8}\,V(P). \tag{3.6}$$
>
> For a single jump from P to P' with log-size $g = \ln(P'/P)$, the loss is
>
> $$\frac{\text{loss}}{V(P)} \;=\; \frac{\left(\sqrt{P'/P}-1\right)^2}{2} \;=\; \frac{g^2}{8} + O(g^3). \tag{3.7}$$

> **Proof.**
>
> The rebalancing portfolio holds $x(P)$ tokens and trades at the reference, so over dt it differs from the pool by the convexity of the pool's holdings: $\frac{1}{2}\sigma^2 P^2|x'(P)|\,dt$. From (3.1), $x'(P) = -\frac{1}{2}LP^{-3/2}$, which gives $\frac{1}{4}\sigma^2 L\sqrt{P} = \frac{\sigma^2}{8}V$. For the jump, holding the old reserves through the move is worth $x(P)P' + y(P) = L\,(P'/\sqrt{P} + \sqrt{P})$, while the arbitraged pool is worth $2L\sqrt{P'}$. The difference is $L(\sqrt{P'} - \sqrt{P})^2/\sqrt{P}$; divide by $V(P) = 2L\sqrt{P}$. ■

At an annual volatility of 35 percent, (3.6) is a drag of 1.5 percent of pool value a year for a full-range pool, paid before any fee is earned, and $A(r)$ times that per dollar for a concentrated one. Equation (3.7) says what a single overnight gap costs: a 5 percent earnings move takes 3.1 basis points of a

full-range pool's value, and about 1.3 percent of the capital in a position concentrated to a ±5 percent range, which the gap also pushes out of range. These losses are not paid to takers as better prices. They are paid to whoever arrives first with the new price.

> **What this chapter establishes**
>
> A passive curve charges $f + 2N/V$, cannot see volatility or the clock, is stale by up to f at all times, and pays $\sigma^2 V/8$ a year to be corrected. Making it competitive takes either capital in the tens of millions per market or concentration that multiplies the arbitrage loss and fails on gaps. Tokenized equities have the gaps and do not have the capital.

3.4 Order books and request-for-quote

An on-chain central limit order book solves staleness by letting makers cancel and replace. On Solana this is feasible, and it is how the most liquid crypto pairs trade. For forty-one equity tokens it needs forty-one sets of resting orders, each updated on every tick of the underlying, with the maker paying for every update whether or not anyone trades. The cost of quoting grows with the number of markets and the tick rate, not with volume.

A request-for-quote system prices only when asked. The maker computes one price for one taker, for an exact size, valid for seconds. Nothing rests on chain to go stale and nothing is paid for until a trade happens. What it gives up is passive discoverability: a router cannot read a quote that does not exist yet. In version 1.0 z poke 's quotes are reached only through its own terminal. Chapter 14 describes how the design would recover discoverability, through an aggregator's request-for-quote interface and with a published-state pool whose parameters, rather than whose orders, live on chain; neither is part of 1.0.

Table 3.1 Four ways to make a market on chain, judged on what matters for an asset that stops trading every evening.

	Reprices from	Stale by	Knows the session	Capital for 10 bps at 10k
Constant product	arbitrage	up to the fee	no	≈ 20,000,000
Concentrated ±5%	arbitrage	up to the fee	no	≈ 480,000, while in range
Order book	maker updates	one update	if the maker does	resting size per level
z poke quote	reference tick	one tick	yes, by construction	inventory for the order

4

Equities keep a calendar

A token trades whenever a block is produced. The share it represents trades six and a half hours a day on a venue the chain cannot see. Everything that distinguishes a market maker in tokenized equities from one in crypto follows from that mismatch.

4.1 Four states of the underlying

The underlying market is in one of four states, fixed by the New York clock and the exchange calendar. In the regular session the share trades with full depth and a maker can hedge a fill within seconds at a known cost. In the extended sessions it trades thinly: a hedge is possible but wider and less certain. Overnight and at weekends it does not trade in size, and a fill is carried unhedged until the next session opens. On exchange holidays and early closes the boundaries move, which is why the calendar is data and not a rule of thumb (Chapter 8).

Table 4.1 Session states by the New York clock, and the share of a calendar week each one occupies in a week without holidays.

State	New York time	Hedge	Hours per week	Share of week
Regular	09:30 – 16:00, Mon – Fri	immediate, full depth	32.5	19.3%
Extended	04:00 – 09:30, 16:00 – 20:00	possible, thin	47.5	28.3%
Overnight and closed	20:00 – 04:00, weekends, holidays	none until the next session	88.0	52.4%

The regular session is under a fifth of the week. A venue that quotes the same way in all four states is either overcharging for nineteen percent of the time or underpricing risk for the other eighty-one.

The price feeds the live pools use (Chapter 8) label each observation with one of four open sessions: regular, pre-market from 04:00 to 09:30, post-market from 16:00 to 20:00, and overnight from 20:00 to 04:00. Version 1.0 quotes in any of them, but only when a freshly verified market-hours calendar and the signed price report agree that the session is open. At weekends, on exchange holidays, and whenever the session is closed or cannot be confirmed, the live pools give no new quotes at all.

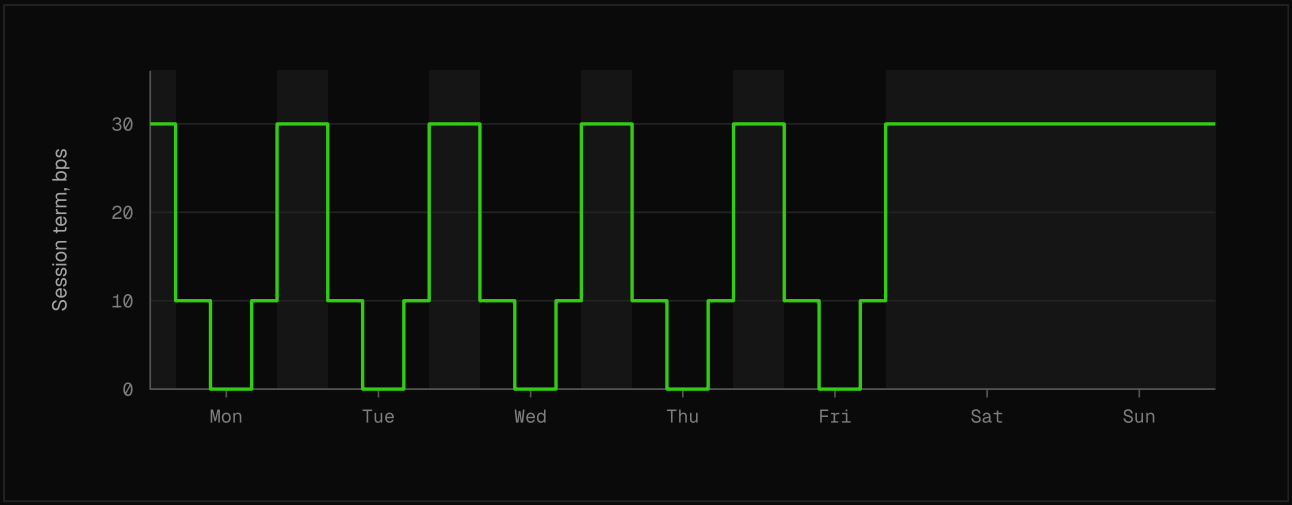


Figure 4.1 The session premium over one week, in basis points of half-spread. Shaded spans are hours in which the underlying cannot be hedged at all. The steps are an illustrative schedule; the engine takes the session term as an input with each quote, and the ten live pools of version 1.0 set the session term to zero in every session in which they quote, and do not quote at weekends or on exchange holidays (Chapters 5 and 8).

4.2 Risk accrues while the market is shut

Write the annual variance of a stock as the sum of what accrues while its market is open and what arrives between the close and the next open. With θ the overnight share,

$$\sigma^2 = \sigma^2_{\text{open}} + \sigma^2_{\text{gap}}, \qquad \sigma^2_{\text{gap}} = \theta\,\sigma^2. \tag{4.1}$$

The gap component arrives in lumps: earnings after the close, macro news before the open, everything that happens over a weekend. A maker who buys from a taker at 21:00 on Friday holds that position through the entire lump. If the inventory acquired is q dollars and the next hedge is available after n closed intervals, the position's standard deviation at the moment it can first be hedged is

$$\text{sd}(\text{P\&L}) = q\,\sigma_{\text{gap}}\sqrt{\frac{n}{252}}. \tag{4.2}$$

A maker that prices this risk at λ units of expected return per unit of standard deviation needs an extra half-spread of

$$\pi_{\text{session}} = \lambda\,\sigma\sqrt{\frac{\theta\,n}{252}}. \tag{4.3}$$

> **Example. Reading the schedule**
>
> Take $\sigma = 35\%$ and suppose, for illustration only, that a third of variance arrives in the gap ($\theta = 1/3$). One overnight interval then carries a standard deviation of $0.35\sqrt{1/756} = 1.27\%$. The schedule's closed-market premium of 30 basis points corresponds to $\lambda \approx 0.24$: the maker asks for about a quarter of a standard deviation. That is a deliberate, conservative starting point, not an estimate. Equation (4.3) is how the parameter is meant to be re-estimated per name as fills and markouts

> accumulate, and it says the right premium is not one number: it should be larger into a weekend (n is effectively larger), larger before earnings, and smaller for an index ETF than for a single volatile stock.

4.3 Why the premium is charged to both sides

It might seem that only the side that increases the maker's exposure should pay. It does not work that way when the maker cannot observe a price. With the market shut there is no reference to be wrong against, only the last one, and information arrives without moving it. A taker who wants to sell at 02:00 on the news everyone has read is as costly as one who wants to buy. The inventory skew of Chapter 5 already discriminates between the two sides on the basis of what the pool holds; the session term prices the fact that the reference itself has stopped being informative, which is symmetric. The live pools avoid the case in which the reference is only the last close: a report whose session is closed is rejected, and no quote is given until the next session opens.

What this chapter establishes

For four fifths of the week a maker in tokenized equities is carrying risk it cannot lay off, and the fair price of that risk is given by (4.3). A quote that does not vary with the session is mispriced most of the time. Making the session an explicit, published term of the price is the central design decision of opcode.

5

The quote

This chapter defines the z poke quote exactly as the engine computes it, in integers, and proves the properties a taker and a maker each rely on.

5.1 State and parameters

A pool is described by a reference price p in micro-dollars per raw token, an inventory I and a target $I^* > 0$ in token atoms, and four parameters in parts per million: a base half-spread b, an impact coefficient κ per 1,000 USDC of notional, a session premium s, and a maximum skew $k_{\max}$. A fifth input is not a parameter but a measurement: the confidence cost c, twice the reference's stated uncertainty relative to its price (Section 6.6). A cap $\bar{h} = 10^5$ ppm, ten percent, bounds every component, the half-spread and each price's distance from the reference. All arithmetic is in 128-bit integers with explicit overflow checks; no floating-point value touches a price.

The inventory and the target are measured from what the pool can actually deliver. Let I and Y be the token and USDC balances of the two vaults, in atoms, less the outgoing amounts reserved by quotes that are still outstanding (Section 13.2). Then

$$I^* = \left\lfloor \tfrac{1}{2}\left(I + \left\lfloor \frac{Y \cdot 10^8}{p} \right\rfloor\right) \right\rfloor,$$

half the pool's available value, expressed in the token at the reference. The target is therefore a 50/50 split by value. It moves with the price and with deposits, redemptions and retained fees; a deposit in the vaults' current ratio raises inventory and target together and leaves the skew where it was. Incoming amounts from a quote count only once the fill has settled.

Table 5.1 Parameters of the ten pools z poke 1.0 runs, NVDAx, SPYx, QQQx, GLDx, TSLAx, AAPLx, MSFTx, METAx, COINx and CRCLx. Pricing parameters are identical across the ten; limits are set per market and enforced by the program.

Parameter	Symbol	Value
Base half-spread	b	500 ppm
Impact per 1,000 USDC	κ	400 ppm
Session premium	s	0 ppm in every session in which the pools quote
Confidence cost	c	twice the reference's confidence over its price, measured per quote
Maximum skew	k_max	1,500 ppm, reached at a 10% deviation from target
Target inventory	I*	half the pool's available value, in the token at the reference
Inventory bounds		the token's share of the pool's value kept between 25% and 75% by the quote signer
Fee	f	5 basis points of the input, rounded up; 80%, rounded down, stays in the pool, the rest to the protocol
Quote lifetime	τ	at most 10 seconds as issued, shown only with at least 5 seconds left; 30 seconds and 64 slots on chain
Largest order	N_max	100 USDC on NVDAx · 20 USDC on each of the other nine
Turnover per fixed 60-second window	W	500 USDC on NVDAx · 100 USDC on each of the other nine
Price band around the on-chain reference		100 basis points

The engine takes the base and the session premium as inputs with every quote, so a base tiered by name, or a premium that steps with the New York session (Figure 4.1), needs no change to it. The live pools use neither: every pool carries a base of 5 basis points and a session term of zero, and none quotes while the underlying market is closed. In the implementation the confidence cost is passed to the engine inside the session input, so the engine sees $s + c$ and, on the live pools, just c. What bounds the position these pools can carry across a close is the per-order and turnover limits and the inventory bounds of Table 5.1, not a premium.

5.2 Definition

Definition 5.1. Size, skew and half-spread

For an order of notional N micro-dollars,

$$z(N) = \left\lfloor \frac{N\,\kappa}{10^3 \cdot 10^6} \right\rfloor, \qquad k(I) = \mathrm{clamp}\left(\frac{(I - I^*)\,k_{\max} \cdot 10}{I^*},\ -k_{\max},\ k_{\max} \right), \tag{5.1}$$

$$h = b + z + c + s, \qquad c = 2\left\lceil \frac{10^6\, C}{S} \right\rceil, \tag{5.2}$$

where S is the price of the underlying share and C its confidence, as stated in the latest report Pyth has published for it (Chapter 8). The division in k truncates toward zero, like every division in the engine not written as a floor or a ceiling. There is no clamp on the half-spread: if z or h exceeds $\bar{h}$, no quote is given, rather than a quote with part of its required spread removed.

Definition 5.2. Prices, fee and output

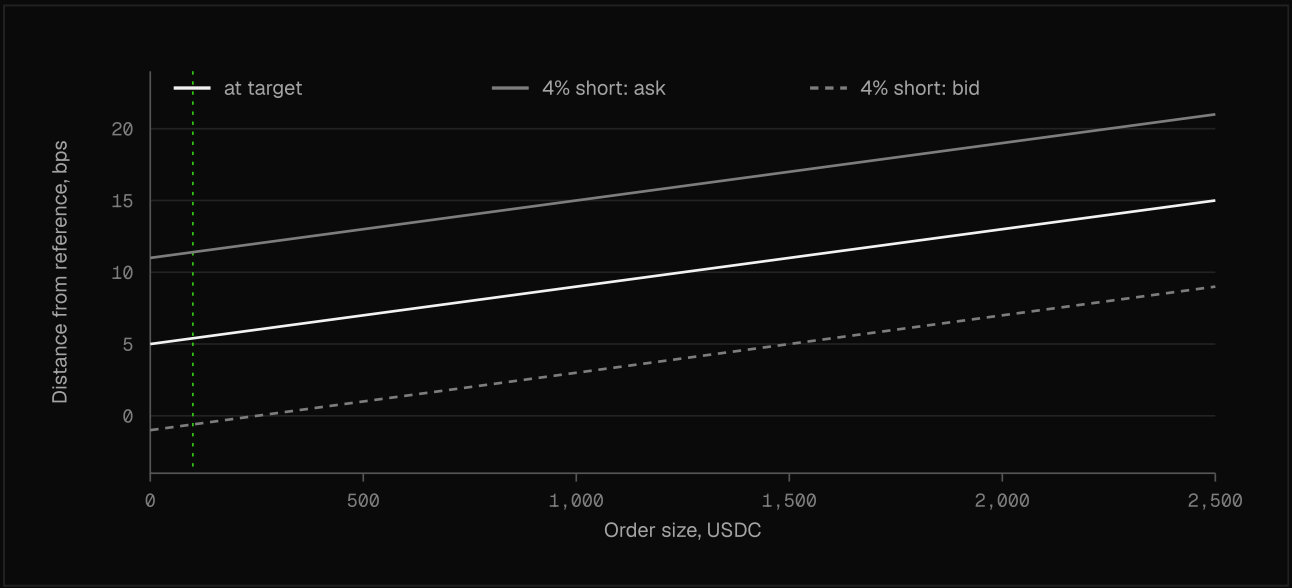
$$P_{\text{ask}} = \left\lceil \frac{p\,(10^6 - k + h)}{10^6} \right\rceil, \qquad P_{\text{bid}} = \left\lfloor \frac{p\,(10^6 - k - h)}{10^6} \right\rfloor, \tag{5.3}$$

and no quote is given if $|h - k|$ or $|h + k|$ exceeds $\bar{h}$. Skew moves the centre of both prices to $p\,(1 - k/10^6)$; the half-spread h sets their distance from that centre.

For an input of a atoms the fee is $\varphi = \lceil af/10^4 \rceil$ and the net input is $a' = a - \varphi$. With d the token's decimals, the output is

$$\text{buy:} \quad q = \left\lfloor \frac{a' \cdot 10^d}{P_{\text{ask}}} \right\rfloor \text{ atoms}, \qquad \text{sell:} \quad v = \left\lfloor \frac{a'\, P_{\text{bid}}}{10^d} \right\rfloor \text{ micro-dollars.} \tag{5.4}$$

Every rounding goes against the taker by less than one unit: the ask up, the bid down, the fee up, the output down. A quote whose output rounds to zero is refused rather than filled. Figure 5.1 shows how far each price lies from the reference as a function of size, for two states of the pool.



`Figure 5.1` Distance of the ask above the reference (solid) and of the bid below it (dashed), with the parameters of Table 5.1 and the confidence cost taken as zero, against order size. A confidence of 1 basis point would raise every line by 2. White: at target, where both distances equal the half-spread. Grey: 4% short of target, so skew raises the centre of the quote by six basis points: the ask moves six further from the reference and the bid six closer, crossing above the reference for orders under 250 USDC, where its distance is negative. The vertical distance between a solid and a dashed line of the same colour is twice the skew; the slope is the impact coefficient. The axis runs past the live limits to show the shape; the rule marks the largest order the live NVDAx pool accepts.

5.3 Properties

Proposition 5.1. Ordering, and where the reference lies

$P_{\text{bid}} < P_{\text{ask}}$ for every state and every size. The reference lies inside the quote, $P_{\text{bid}} \le p \le P_{\text{ask}}$, whenever $|k| \le h$. Otherwise both prices lie on the same side of p: below it when the pool holds too much of the token, above it when it holds too little.

Proof.

The two prices are h either side of the centre $p(1 - k/10^6)$ before rounding, with $h \ge b > 0$, and the two roundings move away from each other. The centre is within h of p exactly when $|k| \le h$. ■

Proposition 5.2. Monotone in size, bounded in skew

The half-spread h is non-decreasing in N, and the distance between the two prices is $2h$ before rounding whatever the inventory: skew moves the midpoint of the quote by $-k$ and leaves its width unchanged, and $|k| \le k_{\max}$.

Proof.

Immediate from (5.1) to (5.3): z is non-decreasing, k is clamped, and k enters both prices with the same sign. ■

The second statement is the one that distinguishes skew from a wider spread. A pool that is short does not charge everyone more. It charges buyers more and pays sellers more, by the same amount, which is what draws the flow that restores it. Because the half-spread is never clamped at zero, the shift carries through in full even when it exceeds the half-spread: a pool at its maximum skew sells below the reference, or buys above it, to the flow that restores it. The price band of Chapter 10 still applies to both prices.

The quote signer adds one bound outside the engine. With w the token's share of the pool's available value at the reference, a trade is refused if it would leave w below 25 percent and falling, or above 75 percent and rising. Flow in the other direction is always allowed. This bound is enforced off chain; on chain, only the vault floors of Chapter 11 limit inventory.

Proposition 5.3. A round trip cannot profit

Let a taker buy with notional N and immediately sell the tokens received, the pool's inventory having fallen by the purchase in between. Write $u = b + z(N) + c + s$ and let $\Delta k \ge 0$ be the fall in skew caused by the purchase. Ignoring rounding, which only helps the maker, the taker ends with less than they started whenever

$$\Delta k \;\le\; 2u + 2f, \qquad \text{for which it suffices that} \qquad p\,I^* \;\ge\; \frac{10\,k_{\max}}{2\kappa/10^3}. \tag{5.5}$$

> **Proof.**
>
> The purchase is made at $p(1+u-k_0)$ and the sale at $p(1-u-k_1)$ with $k_1 = k_0 - \Delta k$, each reduced by the fee. The sale price is below the purchase price when $-u-k_1 \le u-k_0$, that is $\Delta k \le 2u$; the two fees add $2f$ of margin. From (5.1) the skew falls by at most $10\,k_{\max}\,N/(pI^*)$, while $2u \ge 2\kappa N/10^3$; comparing coefficients of N gives the sufficient condition. The target of Section 5.1 does not fall with the purchase: the pool receives at least the value it pays out, so the target rises by half the spread and fee retained. That enlarges Δk by at most about $(u+f)/2$ times the bound above, a second-order term that the margin of the binding condition below covers. ■

With the parameters of Table 5.1 the sufficient condition reads $pI^* \ge 18{,}750$ USDC: a pool whose target inventory is worth at least that much cannot be round-tripped at any size. Below it, the base spread and the fee supply the margin, and the binding requirement becomes $N_{\max}\,(10\,k_{\max}/(pI^*) - 2\kappa/10^3) \le 2b + 2f$, before counting the confidence cost, which only adds to the right side. This is a constraint on how small a pool may be relative to the largest order it accepts. The live pools are below the sufficient condition, with targets, half of each pool's value, worth about 2,500 USDC on NVDAx and 1,500 USDC on each of the others, and meet the binding one with room to spare: the left side is 520 ppm on NVDAx and about 180 ppm on the others, against 2,000 ppm on the right. The engine's test suite verifies, over a grid of spreads, fees and multipliers, that a buy followed by a sale never returns more than was paid, in integers; that grid runs under the engine's earlier policy, and for the live policy the tests fix exact prices, fees and outputs with the pool at maximum skew in each direction.

5.4 One quote, one pool

Nothing in (5.1) to (5.4) depends on any other pool, and each evaluation is a handful of integer operations. In version 1.0 firm quotes exist only on the ten ZPOKE pools, and each is priced by one engine call when a taker asks for it, against the pool's on-chain reference and inventory at that moment. If the underlying has moved more than 5 basis points from that reference, or the reference's source is more than 5 seconds old, no quote is given: the request asks the relayer for a fresh on-chain reference, and the taker asks again once it has landed (Section 8.4). Every quote is therefore priced from a verified source at most 5 seconds old; Chapter 16 reports the measured times.

6

Where the terms come from

A pricing rule with four free parameters could be tuned to say anything. This chapter gives each term an economic meaning and a way to measure it, so that the values in use can be judged and, as data accrues, replaced.

6.1 Base: the cost of being picked off

A maker who quotes both sides loses to takers who know more than the reference does. In the classical models of Glosten and Milgrom and of Kyle, the spread that breaks even is the expected loss to informed flow, spread over all flow. If a fraction α of takers are informed and the informed move the price by δ on average after trading, the break-even half-spread is $\alpha\,\delta$. The base term is this quantity. A tiered base would follow how much private information a name tends to carry: least for a broad index, most for a single volatile stock or a leveraged product. The live pools carry the same base on every name (Table 5.1).

It is measured after the fact by *markout*: the move in the reference over a fixed horizon following each fill, signed by the taker's direction. A base term that is too low shows up as persistently negative markouts net of spread.

6.2 The firm quote is an option, and base must pay for it

An z poke quote is firm for up to $\tau = 10$ seconds as issued, inside an on-chain ceiling of 30 seconds and 64 slots, and is shown to the taker only if at least 5 of those seconds remain. During that time the taker may execute or walk away, and will walk away when the market has moved against them. The maker has written a short-dated option. For a buy quoted at h above the reference, with the reference diffusing at volatility σ, its expected cost is that of a call struck h out of the money:

$$C(h) = \sigma_\tau\big[\phi(d) - d\,(1 - \Phi(d))\big], \qquad \sigma_\tau = \sigma\sqrt{\tau/T}, \qquad d = \frac{h}{\sigma_\tau}, \tag{6.1}$$

where T is a trading year in seconds and ϕ, Φ are the standard normal density and distribution.

Table 6.1 Expected cost to the maker of a 10-second firm quote, in basis points, from (6.1).

Annual volatility	σ over 10 s	h = 3 bps	h = 5 bps	h = 10 bps
20%	2.6 bps	0.16	0.03	0.00
35%	4.6 bps	0.70	0.31	0.02
60%	7.8 bps	1.84	1.23	0.37
90%	11.7 bps	3.33	2.59	1.28

At 35 percent volatility a 5 basis point quote gives away about 0.3 basis points of option value: about a sixteenth of the base term. The price behind a quote may already be up to 5 seconds old when the quote is issued (Chapter 8), so measured from the observation the horizon is up to 15 seconds, and the same quote gives away about 0.6. At 90 percent volatility a 10-second quote gives away about half of the base term, which is why a tiered base would give volatile names 10. The ten live pools, seven single stocks and three ETFs, all carry 5, including volatile names such as TSLAx, COINx and CRCLx; on those it is the 20 USDC order limit, not the base, that keeps the cost small in dollars. The table also shows why the lifetime is seconds and not minutes: the standard deviation grows with the square root of τ, so a five-minute quote carries more than five times that of a 10-second one, and its cost grows faster still because the strike is fewer standard deviations away. The on-chain price band of Chapter 10 caps the option from the other side: a quote that has drifted outside the band around the current verified reference cannot be executed at all.

6.3 Size: what it costs to lay the position off

A maker that fills N and wants to return to target has to trade N somewhere else. The size term is the marginal cost of doing so. If the hedge venue's impact is approximately linear in size with slope η, recovering that cost requires $\kappa \geq \eta$. Table 6.2 reports read-only aggregator quotes for buys of 1,000 and 10,000 USDC in ten of the listed tokens, and the implied rise in price between the two sizes.

Table 6.2 Aggregator quotes for buys, tokens received for 1,000 and 10,000 USDC, and the implied increase in average price. Observations were sequential and read-only on 20 September 2026; they are not simultaneous depth, not executable size, cover the buy side only, and include the provider's fee in both columns.

Token	Out for 1,000	Out for 10,000	Price rise, bps	Per 1,000 USDC, bps
NVDAx	4.50614786	45.05700985	0.99	0.110
SPYx	1.30257612	13.02576120	0.00	0.000
TSLAx	2.74243990	27.42439898	0.00	0.000
AAPLx	2.97767656	29.77058376	2.08	0.231
MSFTx	2.01036690	20.09908443	2.28	0.253
GOOGLx	2.85012189	28.49640444	1.69	0.188
AMZNx	3.94613806	39.45239219	2.28	0.253
METAx	1.48661391	14.86277497	2.26	0.251
AVGOx	2.77583824	27.75297120	1.95	0.217
AMDx	1.79173993	17.91334595	2.26	0.251

For these names the measured slope is a small fraction of the live value of 4 basis points per 1,000 USDC, and for two of them it rounds to zero, because the aggregator is itself routing to professional makers. The live value is therefore conservative for the deepest tokens. For the thin ones, where Chapter 2 shows there is little to hedge against on chain at all, the binding hedge would be the underlying share during the regular session. Version 1.0 places no hedges: the size term is what a taker pays for moving the pool away from its target, and the pool recovers the position only through later flow that the skew draws in.

6.4 Session: the price of carrying a position

Derived in Chapter 4: $\pi = \lambda\sigma\sqrt{\theta n/252}$. The schedule of 10 and 30 basis points in Figure 4.1 is a deliberately simple step function. The engine takes the term as an input with each quote, and the live pools set it to zero. The formula says what a refined version looks like: a premium per name proportional to its gap volatility, larger before a weekend or a scheduled announcement, and falling to zero as the open approaches and the expected holding time shrinks.

6.5 Skew: a reservation price, bounded

Avellaneda and Stoikov (2008) show that a risk-averse maker with inventory q should centre its quotes not on the mid price S but on a reservation price

$$r = S - q\,\gamma\,\sigma^2\,(T - t), \tag{6.2}$$

lower when long and higher when short, by an amount proportional to inventory, risk aversion γ, variance and remaining horizon. The z poke skew is this rule with two changes. Inventory is measured as a deviation from a target rather than from zero, because the maker needs stock on hand to sell; the target is half the pool's available value, so the pool leans toward holding equal values of the token and of USDC. And the shift is capped at $k_{\max}$, reached at a ten percent deviation, so that a pool under sustained one-sided flow becomes steadily less attractive but never quotes a midpoint more than 15 basis points from its reference. Identifying coefficients, $10\,k_{\max}/I^*$ plays the role of $\gamma\sigma^2(T-t)$: a larger target means a flatter response.

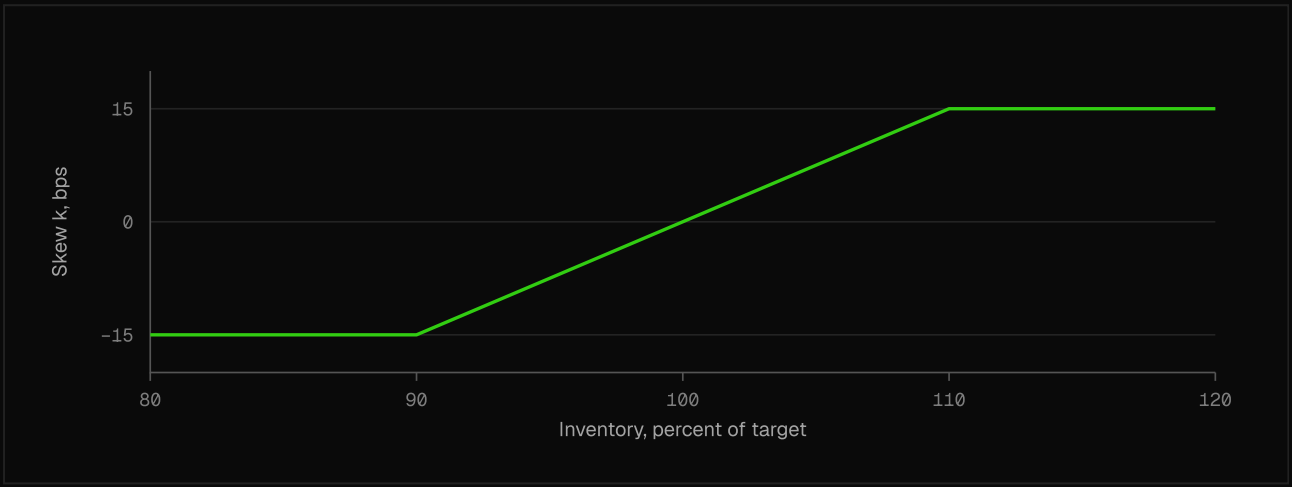


Figure 6.1 Skew as a function of inventory. Linear through the target, flat beyond ten percent either side. Positive skew lowers both bid and ask.

6.6 Confidence: what the reference does not know

Each Pyth price arrives with a confidence interval C, a statement of how far the contributing publishers disagree. It is not a forecast of volatility, but it is the one measure of the reference's own uncertainty that is signed together with the price. The quote charges for it directly: $c = 2C/S$ is added to the half-spread, so a price quoted with a confidence of 1 basis point costs 2 basis points more on each side, and the quote widens in proportion as the publishers disagree.

The term is bounded by the admission rule of Chapter 8: a report is refused if C/S exceeds 25 basis points in the regular session or 50 in the others, so c is at most 50 or 100 basis points. Before that bound is reached, a quote whose price would fall outside the 100 basis point band of Chapter 10 is not given. If the confidence widens between the pricing of a quote and its final check before the quote is returned, the quote is withdrawn and the taker asks for a new one.

7

Inventory and the maker's risk

The quote transfers risk from takers to the maker. This chapter says how much, how it is bounded, and what the maker's profit and loss is made of.

7.1 Inventory mean-reverts under skew

Let buys and sells arrive at rates that respond to price: a side that is one basis point cheaper attracts a fraction ε more flow. With balanced flow of total rate Λ and mean trade size $\bar{q}$, a skew of k tilts net flow by approximately $-\varepsilon k \Lambda \bar{q}$ per unit time. Inside the linear region $k = 10\, k_{\max}\, x$, with $x = (I - I^*)/I^*$ the fractional deviation, so

$$\frac{d\,\mathbb{E}[x]}{dt} \;=\; -\,\rho\, x, \qquad \rho = \frac{10\, k_{\max}\, \varepsilon\, \Lambda\, \bar{q}}{I^*}. \tag{7.1}$$

Deviations decay exponentially at rate ρ. Flow that is more price-sensitive, a steeper skew, or a smaller target all restore inventory faster. Routed flow from aggregators is highly price-sensitive, which is one reason distribution (Chapter 14) matters to risk and not only to volume. Where flow is one-sided regardless of price, skew alone is not enough. A maker can then rebalance actively, trading the excess elsewhere when $|x|$ leaves a band. Version 1.0 does not: the pool carries its inventory, and skew and the 25 to 75 percent value bounds of Section 5.3 are the only controls. A pool under persistent one-sided flow runs to one of those bounds, slowed by the per-order and turnover limits of Table 5.1, and then quotes only in the direction that restores it.

7.2 What the maker earns and what it pays

$$\text{P\&L} \;=\; \underbrace{\sum_i h_i N_i}_{\text{spread}} \;+\; \underbrace{\sum_i f N_i}_{\text{fee}} \;-\; \underbrace{\sum_i m_i N_i}_{\text{markout}} \;-\; \underbrace{\sum_j c_j}_{\text{rebalancing}} \;+\; \underbrace{\int I_t\, dp_t}_{\text{inventory}}. \tag{7.2}$$

The first two terms are known at the moment of each fill. The third is the adverse move after each fill, which the base term exists to cover. The fourth is the cost of hedges. The last is the mark-to-market of whatever is held, which has zero expectation under a fair reference and a variance that the session term exists to pay for. A market is worth quoting when the first two exceed the next two on average and the variance of the last is within the capital allotted to it. None of this can be asserted in advance; it is measured, per market, from fills.

In version 1.0 the fourth term is zero, because nothing is hedged, and the last is carried in full. The pool belongs to its liquidity providers in proportion to their shares, z poke 's founding deposit among them, so this profit and loss is theirs, less the 20 percent of the fee that goes to the protocol (Chapter 18).

7.3 A hard bound on loss

Whatever the flow, a pool's exposure is bounded by what its vaults can hold. The settlement program enforces a floor on each vault (Chapter 11), so inventory lies in $[I_{\min}, I_{\max}]$ and the dollar exposure relative to target is at most $E = p \cdot \max(I_{\max} - I^*,\, I^* - I_{\min})$. A gap of log-size g in the reference then costs at most

$$\text{loss} \;\le\; E\left(e^{|g|} - 1\right). \tag{7.3}$$

> **Example. Sizing a pool**
>
> A pool with a target worth 200,000 USDC, allowed to run 25 percent either side, has $E = 50{,}000$. A 10 percent gap against a fully extended pool costs at most 5,260 USDC. Compare a constant-product pool of the same 400,000 USDC total value: equation (3.7) puts its loss from the same gap at 0.12 percent of value, about 480 USDC, which looks better until one recalls from (3.2) that such a pool charges a taker 50 basis points of impact on a 1,000 USDC order and is therefore not trading at all. The maker's loss is bounded and chosen; the curve's is small only because its prices are too poor to attract flow.

The live pools are far smaller than this example. The quote signer keeps the token's share of each pool's value between 25 and 75 percent, so E is about a quarter of the pool's value: some 1,250 USDC on NVDAx and 750 USDC on each of the nine others. The same 10 percent gap against a pool at either bound then costs at most about 130 USDC on NVDAx and 79 USDC on each of the others; if all ten were at a bound and all gapped 10 percent together, E would total 8,000 USDC and the loss about 840 USDC. That bound is enforced off chain. On chain, the vault floors are one raw token atom and one USDC, so a pool can run almost entirely into either asset only if the signer fails to enforce it; E is then about half the pool's value, 2,500 and 1,500 USDC, and the same gap costs at most about 260 and 160 USDC, or about 1,680 USDC across all ten.

> **What these three chapters establish**
>
> The quote is a precise integer function with proved ordering, monotonicity, bounded skew and no profitable round trip under a stated condition on pool size. Each of its terms corresponds to a cost that can be measured from fills and hedges. The risk it leaves with the maker mean-reverts under skew, is held inside value bounds by the signer, and is capped absolutely by vault limits enforced on chain.

8

The reference price

Every property proved in Chapter 5 is relative to the reference. A quote built on a wrong reference is wrong however carefully it is rounded. This chapter specifies how the reference is formed and the conditions under which z poke concludes it does not have one.

8.1 The price of the underlying share

For each ZPOKE market the reference starts from the price of the share or fund the token tracks, not from the token's own market, which Chapter 2 shows is too thin for most names to carry a price. z poke reads that price from Pyth Pro, Pyth's low-latency feeds for US equities: one feed per market, Equity.US.NVDA/USD, SPY, QQQ, GLD, TSLA, AAPL, MSFT, META, COIN and CRCL (Appendix B), where GLD is the fund and not the spot price of gold. The feeds are streamed at a fixed rate of one update every 200 milliseconds over three redundant connections. Each update carries the price and its exponent, a confidence interval, the number of contributing publishers, the market session the feed is in, and the time at which the price was generated, the source time, together with a report of the same fields signed by Pyth.

> **Definition 8.1. Normalization into the token**
>
> Let S be the underlying's price in dollars, E the issuer's exposure in shares per displayed token unit, which is one for the listed xStocks, and m the mint's active multiplier in displayed units per raw token, read from the Token-2022 mint at the chain clock. Taking one USDC to be one dollar (Section 8.2), the reference in micro-USDC per raw token is
>
> $$p = \mathrm{round}\big(10^6 \cdot S \cdot E \cdot m\big), \tag{8.1}$$
>
> rounded to the nearest micro-USDC, ties upward, in exact rational arithmetic. The multiplier is applied here and nowhere else; the engine receives p with a multiplier of exactly one (Chapter 9). For example, $S = 200$, $E = 1$ and $m = 1.002$ give $p = 200{,}400{,}000$.

A verified price for the share is not a price for the token. The token can trade away from the share, most of all outside the regular session, and that basis can persist: the pools place no hedges and have no access to the issuer's creation and redemption to close it. The reference fixes the price at which the pool trades, not the price at which the token trades elsewhere.

8.2 One USDC is taken to be one dollar

The feed prices the share in dollars and the pool exchanges it for USDC. z poke converts between the two by an explicit assumption, recorded on chain as the policy `ASSUMED_USDC_PARITY`: one USDC is one dollar. It is not a measured or verified USDC price; Pyth's USDC feed is not available under the equities subscription z poke holds, and the program verifies only the equity price. The assumption is guarded instead. An independent USDC price from Jupiter, whose observation time is taken from the Solana block it reports, must lie between $0.99 and $1.01 inclusive and be no more than 60 seconds old, or no new ZPOKE quote is given. The guard never changes the price. A depeg inside that band, or any difficulty redeeming USDC for dollars, is borne by the user and the pool (Chapter 19).

8.3 Verified on chain

The reference the program settles against is written by the program itself, from the signed report. A publication transaction carries the report, 158 bytes, and two instructions that do the work. Solana's native Ed25519 instruction checks the report's signature. z poke 's `publish_pyth_reference` then parses the report, applies the checks of Table 8.1 and computes (8.1) on chain from the report's price and the mint's multiplier. Before it writes anything it calls the official Pyth Pro verifier program, which confirms that the signing key is one Pyth currently trusts and has not expired, and that the bytes the native instruction verified are exactly the bytes z poke read. Only when the verifier accepts does z poke write the market's reference together with a provenance record, the market's `OracleConfig`.

Table 8.1 Checks made by publish_pyth_reference before it writes a reference. Any failure rejects the whole transaction.

Check	Condition
Signature	valid Ed25519 signature over the exact report, by a signer the Pyth verifier trusts and has not expired, established through a cross-program call to the verifier
Format	exactly one feed and the six required properties, each once, with the exponent inside the pinned range
Feed and channel	the feed identifier pinned in the market's OracleConfig, on the fixed 200 ms channel
Session	the report's own session field is an open session the market's policy allows
Publishers	at least 3 in the regular session and 2 in the others; 3 for COIN in pre-market and post-market
Confidence	at most 25 bps of the price in the regular session, 50 bps in the others
Age	source time no later than the chain clock and at most 5 seconds before it
Order	source time strictly later than the last verified one; sequence exactly one more than the current reference
Validity	expiry after the chain clock, at most 20 seconds after the source time, and a slot range of at most 50 slots
Token	the mint's configuration and active multiplier equal those the market was registered with

`OracleConfig` is a 227-byte account at an address derived from the market. It holds the pinned policy, meaning the feed, channel, permitted sessions, publisher and confidence limits, exponent range, issuer exposure and a policy revision, and the provenance of the current reference: the SHA-256 hash of the report, its source time, the feed's price, confidence, exponent, publisher count and session, the multiplier and normalization epoch it was normalized under, the resulting price, and the reference's sequence number and validity. At every fill the program recomputes (8.1) from those stored fields and requires the reference to match them exactly (Chapter 10).

All ten markets are in the mode `PYTH_ONLY`. Each was switched once, while paused, and the switch is one-way: in this mode the program refuses the older instruction by which an operator key published a reference of its own choosing, and refuses any return to it. Anyone may submit a verified report and pay its fees, and z poke runs a relayer that does. The relayer chooses no price. It can decide only whether and when to submit, and a report it withholds lets the reference expire, which stops quoting. The mode is a property of the deployed program, which the upgrade authority could replace (Chapter 11).

Verification has been live on all ten markets since the program upgrade at slot 450,310,590 on 25 September 2026.

8.4 Freshness, sessions and limits

Table 8.2 Time and quality limits on the ZPOKE reference and on the quotes built from it.

Rule	Limit
Source age when a reference is published	at most 5 seconds
Source age when a quote is prepared and when it is issued	at most 5 seconds
Reference lifetime	at most 20 seconds after the source time and 50 slots; shorter at a session boundary
At every fill	verified source at most 20 seconds old, reference unexpired, provenance matching, 100 bps price band
New quote against the current reference	refused, and a refresh requested, if the underlying has moved more than 5 bps from it or its source is over 5 seconds old
Quote lifetime	at most 10 seconds, and never past the reference's expiry, its slot range or the next session change; shown only with at least 5 seconds left
Sessions	regular 09:30–16:00, pre-market 04:00–09:30, post-market 16:00–20:00, overnight 20:00–04:00, New York time, each confirmed by the calendar and by the report
Confidence	at most 25 bps regular, 50 bps in other open sessions; twice the ratio enters the half-spread (Section 6.6)
Publishers	at least 3 regular, 2 in other sessions; 3 for COIN in pre-market and post-market
USDC guard	independent price within $0.99–$1.01 inclusive, at most 60 seconds old

A report landed two seconds after its source time can therefore support a quote until twelve seconds after it, provided the taker still has five seconds to review it. Publication follows demand. A market that has had a quote request in the last minute is refreshed about every two seconds, each market on its own schedule, so a slow confirmation in one does not hold up another. A market with no request for 60 seconds cools down; the next request asks for a refresh, and the taker sees the quote as being prepared until the new reference is confirmed. These are scheduling targets, not guaranteed landing times.

> **Remark. Fail closed**
>
> A market without a fresh verified reference cannot be quoted. There is no last-known-good price, no fallback to the token's own price, no average, no widened spread as a substitute for knowledge. The cost of this policy is availability: some markets are unquotable some of the time, including every market at weekends and on exchange holidays. The cost of the alternative is that the maker's worst quotes are precisely the ones that get taken.

8.5 The calendar is data

The session in force is derived from Pyth's published market-hours calendar for each underlying, which includes holidays and early closes, refreshed daily. For ZPOKE the calendar must have been verified within the last 48 hours, and the session it gives must agree with the session field of the signed report; neither alone is enough. For what the terminal displays, a calendar is used with a warning for up to seven days, after which the session is shown as unknown, and until a first calendar has ever been verified a weekday clock rule is used and labelled as an estimate. Daylight saving is handled by the time zone database rather than by offsets, so a local time that does not exist is never taken for a boundary.

8.6 The public reference, for Quick and the charts

z poke keeps a second reference for all 41 markets. It prices what the terminal displays for the Quick route and the charts, and it is not ZPOKE 's price. For each market it observes the token's own on-chain price with the liquidity behind it, and the last price of the underlying share. The token price is available at all hours but, by Chapter 2, is meaningless for most names; the share price is authoritative but exists only while the exchange is open.

> **Definition 8.2. Public reference selection**
>
> Let ℓ be the token's on-chain liquidity and $\ell^* = 250{,}000$ USDC. With m the token's active multiplier,
>
> $$p_{\text{public}} = \begin{cases} p_{\text{token}} & \ell \geq \ell^* \\ m \cdot p_{\text{share}} & \ell < \ell^* \end{cases} \tag{8.2}$$
>
> and the source used is published with the price.

A candidate public reference is accepted only if it passes every guard in Table 8.3. Failure of any one marks it stale.

Table 8.3 Guards on the public reference. Bands are wider outside the regular session because the token and the last share price legitimately diverge when the share is not trading.

Guard	Condition	Limit
Freshness of the snapshot	time since the observation	10 s
Freshness of the token price	time since the on-chain observation	60 s
Freshness of the share price	regular session / otherwise	15 min / 7 days
Token against share	relative difference, regular / otherwise	150 bps / 400 bps
Independent venue	relative difference against a second venue's book, for the ten most traded names	same bands
Continuity	move between consecutive observations	10%
Unit consistency	price per displayed share times multiplier equals price per raw token	1 part in 10^9

The independent venue is a centralised exchange's order book in the same token, normalised to USDC through a separately verified stablecoin peg, with a second exchange as fallback. It shares no infrastructure with the primary source, so a fault in one is unlikely to be mirrored in the other. Because the public reference fails closed in the same way, not every market has a public price at every moment; at the time of writing about 35 of the 41 were available. Quick trades at the route's own price, not at this reference.

9

Units: tokens, shares and the multiplier

A tokenized stock has two quantities, raw tokens and the shares they represent, related by a multiplier the issuer changes. Most of the ways to lose money quietly in this domain are unit errors. z poke 's rule is simple: everything is raw until the screen.

9.1 The conversion

Let n be a balance in raw tokens and m_t the mint's active multiplier at time t. The holding represents $n\,m_t$ shares. If S_t is the share price, the price of one raw token is

$$p_t = m_t\,S_t, \qquad \text{so that} \qquad n\,p_t = (n\,m_t)\,S_t. \tag{9.1}$$

Value is invariant to the choice of unit, as it must be. What is not invariant is any computation that mixes them: a raw quantity priced per share is wrong by the factor m_t, and a provider that already reports a per-raw-token price must not have the multiplier applied again.

9.2 The rule

1. References are in micro-dollars per raw token. The ZPOKE reference is normalised on chain by (8.1), from the share price and the multiplier the mint reports at that moment, with the multiplier applied once; the public reference is normalised on arrival, and the unit-consistency guard of Table 8.3 rejects a source whose two price fields disagree with the mint's multiplier.
2. The engine receives raw atoms and a raw-token price, and a multiplier of exactly one. It never converts.
3. Quotes, transfers, receipts and fill events are in raw atoms: eight decimals for the token, six for USDC.
4. Shares appear only in the interface, beside the raw figure, computed from the multiplier in force at that moment, and marked approximate when truncated.
5. Every fill event records the normalization epoch its quote was priced under, so history remains interpretable after the multiplier changes.

9.3 A scheduled change

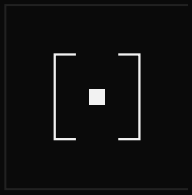
The mint carries the current multiplier, a new multiplier, and the time at which the new one takes effect. The active value is therefore a step function known in advance:

$$m_t = \begin{cases} m^{\text{old}} & t < t^* \\ m^{\text{new}} & t \geq t^* \end{cases} \tag{9.2}$$

Off chain, the step is applied at t^* without waiting for a poll. On chain, the program stores the multiplier it registered the market with and compares it with the mint's active value at every fill; at t^* they differ, fills stop, and the market is refreshed under a new normalization epoch that voids every quote priced in the old unit (Chapter 12). A quote can never straddle the step. Nor can a reference: publication requires the mint's active multiplier to equal the registered one, and the provenance of every reference records the normalization epoch it was computed under. The share price and the multiplier can change on different clocks around a split, so new ZPOKE quotes are withheld across an announced change until the mint, the issuer's exposure mapping and a post-change price agree.

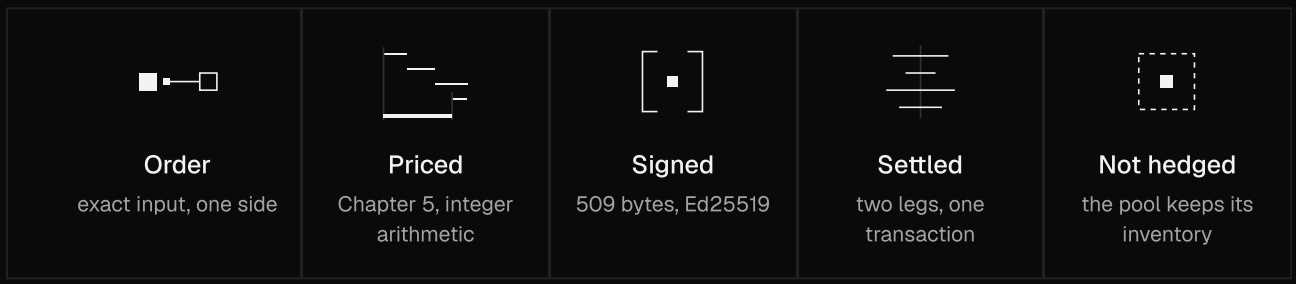
10

Settlement: one signed quote, one transaction



A price is only as good as the guarantee that it will be honoured exactly. z poke settles by request-for-quote: the maker signs fixed amounts for one taker, and an onchain program moves both legs or neither. This chapter specifies the message, the verification, and the properties that follow.

10.1 The shape of a trade



The taker names a market, a side and an exact input amount. The engine prices it from the pool's current state. The maker's quote authority signs a message that fixes the input amount, the output amount, the fee, the taker's wallet and token accounts, and the window in which it is valid. The taker's wallet then signs and sends a transaction that carries the maker's signature and asks the program to execute it. The taker never signs a price it has not seen, and the maker never signs an open-ended commitment.

There is no slippage parameter, because there is nothing to slip. The amounts in the signed message are the amounts that move. If conditions have changed so that the program will not honour them, the transaction fails and nothing moves except the network fee.

10.2 The quote message

The message is 509 bytes: a 13-byte domain tag, thirteen 32-byte identities, and ten 8-byte little-endian integers. There is no length prefix, no padding and no optional field, so there is exactly one encoding of any quote and no parser ambiguity to exploit. Its SHA-256 hash is stored in the receipt and emitted in the fill event.

Table 10.1 Layout of the version 1 quote. Every identity a caller could try to substitute is inside the signed bytes.

Offset	Bytes	Field	Meaning
0	13	domain_tag	ASCII z poke :rfq:v1
13	32	deployment_domain	which deployment this quote belongs to
45	32	program_id	the settlement program
77	32	protocol	protocol configuration account
109	32	market	market configuration account
141	32	maker_id	the maker this market belongs to
173	32	quote_id	unique; seeds the receipt
205	32	trader	the only wallet that may execute
237	32	user_source	token account debited
269	32	user_destination	token account credited
301	32	input_mint	
333	32	output_mint	
365	32	input_token_program	
397	32	output_token_program	
429	8	amount_in	u64, raw atoms, fee included
437	8	amount_out	u64, raw atoms
445	8	fee	u64, in the input asset
453	8	issued_slot	u64
461	8	max_valid_slot	u64
469	8	expires_at	i64, unix seconds
477	8	config_epoch	u64
485	8	normalization_epoch	u64
493	8	risk_epoch	u64
501	8	policy_epoch	u64

The four epochs are the revocation mechanism. Each names a version of something the quote was priced against: the market's configuration, the token's unit normalization, the market's risk state, and the participant policy. If any has moved on since the quote was signed, the quote is dead. A maker who needs to withdraw every outstanding quote at once, for instance on a halt in the underlying, increments the risk epoch with a single instruction.

10.3 Verifying a signature inside a program

Solana programs do not verify Ed25519 signatures themselves. The runtime provides a precompile that verifies signatures named by an instruction's data and fails the whole transaction if any is invalid. A program that relies on it must then prove, by reading the instructions sysvar, that the precompile instruction in this transaction verified *this* message under *this* key. Doing that loosely is a well-known way to lose a vault: the precompile's header contains offsets that may point into other instructions, so a careless program can be shown a valid signature over bytes other than the ones it goes on to read.

The settlement program removes every degree of freedom. Before any economic check it requires all of the following.

Table 10.2 Transaction template enforced by execute_quote. Anything else fails before an amount is read.

Requirement	Why
The instruction runs at the top level of the transaction, never by cross-program invocation.	Another program cannot wrap a fill inside logic the taker did not see.
It is the last instruction in the transaction.	Nothing can run after the transfer that depends on it.
The instruction immediately before it is the Ed25519 precompile, with no accounts and exactly 621 bytes of data.	One header, one key, one signature, one 509-byte message, and nothing else.
The header declares exactly one signature, and its seven offsets equal 48, self, 16, self, 112, 509, self.	Signature, key and message must all live inside that same instruction. Pointing anywhere else is rejected even though the precompile would accept it.
The 32-byte key in the instruction equals the quote authority stored in the market account.	The key comes from chain state, not from the caller.
Every earlier instruction is either a compute-budget setting, at most one of each kind, or an idempotent creation of the taker's own associated token account for one of the two mints in the quote.	Nothing else may share a transaction with a fill.

Only then is the message decoded, and each identity in it compared with the accounts actually supplied: deployment, program, protocol, market, maker, taker, source and destination token accounts, both mints and both token programs. The twenty-one accounts of the instruction, the twenty of the original design and the market's `OracleConfig`, must be pairwise distinct, and no additional accounts are accepted.

10.4 What is checked at execution time

A valid signature is necessary, not sufficient. The program re-establishes at execution time every condition the quote was priced under, because a quote is signed off chain seconds earlier and the chain may have moved.

Table 10.3 Checks performed by execute_quote after the signature is established, in order.

Check	Condition	Failure
Not paused	protocol and market both active	`Paused`
Funded	the market's liquidity pool has shares outstanding	`LiquiditySharesRequired`
Epochs	all four epochs in the quote equal the chain's	`EpochMismatch`
Lifetime	issued ≤ now ≤ max slot; slot span ≤ $\mathrm{max_quote_slots} \leq 128$, 64 on the live markets; expiry in the future and at most $30\ \mathrm{s}$ away	`QuoteExpired`
Fee	fee equals $\lceil \mathrm{amount_in} \times \mathrm{fee_bps}/10^4 \rceil$ exactly	`Amount`
Direction	input and output mints are the market's pair, in one order or the other	`AccountMismatch`
Token state	mint fingerprint and active multiplier unchanged; accounts unfrozen, undelegated	`MintConfiguration`, `NormalizationChanged`, `TokenConfiguration`
Oracle	the market's OracleConfig is canonical and, in PYTH_ONLY mode, its verified report is at most 20 seconds old and still meets the session, publisher and confidence limits; the reference's price, sequence, times, slots and normalization epoch equal the recorded provenance, and the price equals (8.1) recomputed from it	`OracleConfiguration`, `OraclePolicy`, `OracleProvenance`
Reference	the on-chain reference is fresh by both slot and clock and has not expired	`ReferenceUnavailable`
Price band	the maker's price net of fee lies inside the band around that reference	`PriceBand`
Size	gross amounts within per-trade limits	`RiskLimit`
Inventory	vaults stay above their floors after the trade	`RiskLimit`
Turnover	notional in the current fixed 60-second window stays under its cap	`RiskLimit`
Participant	when the market requires it, the taker holds a current permission for this side	`ParticipantPolicy`
Transfers	after three checked transfers, all five token balances changed by exactly the signed amounts	`BalanceDelta`

The price band deserves a closer look, because it bounds the damage of a compromised or faulty maker key. The reference is not the maker's: on the live markets it is written only by the program, from a Pyth report it has verified (Chapter 8), and short of an upgrade of the program no key can set it to a price of its choosing. With p that reference in micro-dollars per raw token, b the band in basis points, and (Q, B) the dollar and token amounts of the fill net of fee, the program requires

$$B\,p\,(10^4 - b) \;\le\; Q \cdot 10^8 \cdot 10^4 \;\le\; B\,p\,(10^4 + b), \tag{10.1}$$

evaluated in 128-bit integers by cross-multiplication, with no division and therefore no rounding in the maker's favour. The factor 10^8 reconciles the token's eight decimals with the dollar's six. A stolen quote key can sign prices, but only prices within b of the verified reference, only up to the per-trade and per-window limits, only while the vaults stay above their floors, and only until the emergency or administrator key pauses the market, which also voids everything already signed. The inventory bounds of Section 5.3 are the signer's own rule and do not constrain a stolen key. On the live markets b is 100 basis points.

10.5 Atomicity and replay

A fill performs three token transfers: the protocol's 20 percent of the fee, rounded up, from the taker to the protocol fee vault; the rest of the input, including the liquidity providers' 80 percent of the fee, from the taker to the market's vault; and the output from the market's vault to the taker, the last signed by the market's program-derived address. Solana executes a transaction completely or not at all, so a failed check at any point leaves every balance as it was.

The same transaction creates a receipt account whose address is derived from the market and the quote's identifier. Account creation fails if the address is already in use, so executing a quote a second time is impossible by construction: not under a new blockhash, not with a different fee payer, not with the instructions rearranged. The receipt records the quote hash, the taker, the amounts, the fee, the slot and the validity bounds. It is the taker's proof of the fill and the indexer's source of truth.

> **Proposition 10.1. Exactly-once, exact-amount execution**
>
> For any signed quote q, in any sequence of transactions, the taker is debited $\mathrm{amount_in}(q)$ at most once, and whenever the taker is debited it is credited exactly $\mathrm{amount_out}(q)$ in the same transaction.

> **Proof.**
>
> A debit occurs only inside a successful `execute_quote`, which creates the receipt at the address derived from $(\mathrm{market}, \mathrm{quote_id})$. A second success would have to create the same address and cannot. Within a success, the three transfers carry amounts read from the signed bytes, and the final balance check compares the actual change in each of the five token accounts with those amounts, so a token extension that altered an amount in flight would fail the transaction rather than shortchange either side. Atomicity of the transaction gives the rest. ■

10.6 Cost on chain

Measured on a local validator running the current program, with the official Pyth verifier and native signature checks in place, a swap against a verified reference consumed 108,037 compute units, a small fraction of the per-transaction budget. The transaction carries a 621-byte signature instruction and twenty-one accounts, which would not fit in Solana's 1,232-byte packet on its own; an address lookup table holding the sixteen accounts that are the same for every fill in a market, the market's `OracleConfig` among them, brings a buy or a sell to 1,119 bytes, and to 1,171 when both of the taker's token accounts must be created. The client refuses to build anything over the limit rather than discover it at submission.

Publishing a reference is a separate transaction, so the quote format and the swap template did not change when verification was added. On the same local validator a verified publication consumed 67,949 compute units. Its packet, with two compute-budget instructions, the native Ed25519 instruction and the publication itself, is 844 bytes without a lookup table. The Pyth verifier charged a fee of 1 lamport per update, separate from Solana's transaction fee, when this was written; the relayer pays both. On mainnet, a read-only simulation of the official verifier alone accepted a real 158-byte report using 12,255 compute units in a 581-byte transaction, and rejected a tampered report and a correctly signed report from an untrusted signer.

Table 10.4 Properties of settlement. Compute figures were measured on a local validator with the real Pyth verifier and native signature checks, not taken from mainnet fills; the sizes are those the current transaction builder produces; the binary is the one deployed on mainnet at slot 450,310,590.

Quantity	Value
Quote message	509 bytes
Signature instruction	621 bytes
Accounts in execute_quote	21, pairwise distinct, none optional
Compute units, swap against a verified reference	108,037
Transaction size with lookup table	1,119 bytes of 1,232
Signed Pyth report	158 bytes
Compute units, verified reference publication	67,949
Publication packet without lookup table	844 bytes
Program binary	750,992 bytes
Quote lifetime on chain, live markets	30 seconds and 64 slots, whichever ends first
Quote lifetime as issued	at most 10 seconds, shown only with at least 5 seconds left

11

The program: accounts and authorities

A settlement program is mostly a statement about who may do what. This chapter lists the accounts the program owns, the keys it recognises, and the limits on each of them, including the limits on z poke itself.

11.1 Accounts

Table 11.1 Program-derived accounts. Sizes include the eight-byte discriminator.

Account	Derived from	Holds	Bytes
Protocol	deployment domain	the four authorities, the global pause	171
Market	protocol, both mints, maker	vaults, quote authority, treasury, limits, the mint fingerprint, three epochs	483
Risk state	market	risk epoch, turnover window, fill count, slot of the last pause	88
Reference	market	the reference price, its sequence number and its validity by slot and by clock; on the live markets written only from a verified report	96
Oracle (OracleConfig)	market	mode, pinned Pyth feed and policy, and the provenance of the current reference: report hash, source time, feed fields, multiplier, normalization epoch, price and sequence	227
Receipt	market, quote id	one per fill: hash, taker, amounts, fee, slot	185
Participant	protocol, wallet	permission bits, expiry, revision	89
Liquidity pool	market	total shares, the providers' share of the fee, fee counters	116
Liquidity position	pool, owner	one provider's shares	82
Liquidity receipt	pool, owner, operation id	one per deposit or redemption: shares and both amounts	138

Each market has four token vaults: inventory in each asset, owned by the market's derived address, and a fee vault in each asset, owned by the protocol's. Registration rejects any aliasing among them and checks each one's mint, owner and token program. Markets are created paused, and cannot trade until a liquidity pool has been opened with shares outstanding and a fresh reference published.

The inventory vaults belong to the pool's shares. The treasury makes the founding deposit, which issues the first shares as the geometric mean, in atoms, of the two vault balances. After that anyone may deposit both assets in the vaults' current ratio, with signed maximum inputs and a minimum number of shares, and anyone holding shares may redeem them for both assets pro rata, with signed minimum outputs. Deposit amounts round up and redemptions round down, so no entry or exit dilutes the remaining shares. A redemption advances the risk epoch, voiding every outstanding quote, and the redemption that empties the pool or takes a vault below its floor pauses the market.

11.2 Separation of powers

Table 11.2 Keys recognised by the program. The administrator, emergency, reference and policy keys must be distinct, and the quote authority must differ from all of them and from the treasury. In version 1.0 the administrator and the treasury are the same owner wallet, which is also the program's upgrade authority.

Key	May	May not
Administrator (owner wallet)	register markets, open liquidity pools, pause and resume, change limits while paused, rotate the quote authority, void quotes, refresh token normalization, collect the protocol's fees; create a market's oracle and switch it to PYTH_ONLY, once, while paused	sign quotes, move pool inventory, publish references, return a market from PYTH_ONLY
Emergency	pause the protocol or a market	resume anything, or do anything else
Reference publisher	publish a reference of its choosing on a market still in the earlier mode; none of the ten live markets is	publish anything on a PYTH_ONLY market, or anything else
Relayer (any fee payer)	submit a Pyth-signed report that passes every check of Table 8.1, paying the fees	choose, alter or back-date the price; it decides only whether and when to submit
Policy	grant and revoke participant permissions	anything else
Treasury (owner wallet)	make a pool's founding deposit; add inventory to the vaults	withdraw inventory: the withdrawal instruction is disabled, and its shares redeem like any provider's
Liquidity provider (anyone)	deposit both assets for shares; redeem its own shares for both assets	anything else
Quote authority (per market)	sign quotes off chain	anything on chain: it cannot withdraw, list, change a limit or grant a permission

Two of these constraints are worth drawing out. The emergency key can only make the system safer, so it can be held somewhere fast and relatively exposed, such as an automated monitor, without widening what an attacker could do with it. And inventory cannot be pulled from under a quote that a taker is in the middle of executing: inventory leaves the vaults only through fills and share redemptions, and a redemption voids every outstanding quote before any later fill can execute.

Above all of these sits the program's upgrade authority. The settlement program is upgradeable, and its upgrade authority is the owner wallet `FyRz7wPjrFqj6RAwk1EcrJfhFaEpnChXAm21MrdGKyw6`, a single key with no multisig and no timelock. Everything this chapter says the program enforces is true of the deployed binary, and an upgrade signed by that key could change it, including the rule that a `PYTH_ONLY` market accepts only verified references.

11.3 Limits the chain enforces on the maker

The limits in a market's configuration are validated when set, and the program refuses configurations outside hard bounds no administrator can exceed: a fee of at most 1 percent, a price band of at most 10 percent, a quote lifetime of at most 30 seconds and 128 slots, and a reference no older than 300 seconds or 512 slots. Changing limits requires the market to be paused and increments the configuration and risk epochs, which voids every quote signed under the old limits.

`Table 11.3` Limits of the ten live markets, set at registration and enforced by the program on every fill. Notional limits are in USDC per market.

Limit	NVDAx	The other nine
Fee	5 bps	5 bps
Price band around the on-chain reference	100 bps	100 bps
Largest order	100	20
Turnover per fixed 60-second window	500	100
Quote lifetime on chain	30 s, 64 slots	30 s, 64 slots
Reference lifetime, as registered	60 s, 160 slots	60 s, 160 slots
Reference lifetime, PYTH_ONLY	20 s from source, 50 slots	20 s from source, 50 slots
Oracle mode	PYTH_ONLY	PYTH_ONLY
Vault floors	1 raw atom, 1 USDC	1 raw atom, 1 USDC
Participant permission required	no	no

> **Remark. Fixed windows that cannot be reset early**
>
> The turnover cap applies to fixed 60-second windows, not a rolling one. When a window has elapsed, its start advances by a whole number of window lengths rather than jumping to the present, so the windows stay on one grid and the cap cannot be refreshed early by timing fills or by a skewed clock. The cost of fixed windows is that fills either side of a boundary draw on two caps: in a short interval that straddles one, up to twice the cap can trade, 1,000 USDC on NVDAx and 200 USDC on the others.

12

Token-2022 and the issuer's powers

Tokenized stocks are Token-2022 mints with extensions that give the issuer real powers over every balance. A venue that ignores them is relying on luck. z poke names the configuration it supports, fingerprints it, and stops the moment it changes.

12.1 What the issuer can do

The tokens z poke lists are issued by Backed under the xStocks name. Each mint carries a metadata pointer, a scaled UI amount configuration that expresses corporate actions as a multiplier, a pausable configuration, a permanent delegate that can move or burn any balance, a default account state, and initialised but inactive transfer-hook and confidential-transfer extensions. These exist so the issuer can meet its regulatory obligations. They also mean that the token a taker receives is subject to the issuer's control in ways a share in a brokerage account is not, which Chapter 19 states plainly as a risk.

12.2 An allowlist, a fingerprint, and a stop

The program accepts a stock mint only if it is owned by the Token-2022 program, has eight decimals, and carries no extension outside an explicit allowlist. Transfer fees are not on it: an asset that skims transfers cannot satisfy exact-amount settlement. The scaled amount and pausable configurations must be present, the mint must not be paused, a transfer hook may be initialised only with no program behind it, and confidential transfers only with auto-approval off and no auditor key.

At registration the program hashes everything that defines the mint's behaviour, including the extension set, the multiplier schedule, the pause authority, the mint and freeze authorities, and the delegate, hook and confidential configurations, and stores the digest in the market. Every fill recomputes it. If the issuer enables a hook, changes an authority or schedules a new multiplier, the digest changes and fills stop with a precise error until the change has been reviewed and the market refreshed, which voids outstanding quotes.

Token accounts are held to the same standard. The taker's and the vaults' accounts must be initialised and not frozen, with no delegate, no close authority, and no extension that would block or alter an incoming transfer. Transfers use the checked form of the instruction with the decimals stated, so a mint substituted at the last moment cannot be mistaken for the expected one.

12.3 Corporate actions

A stock split or dividend does not change anyone's raw token balance. The issuer changes the mint's multiplier on a scheduled date, and wallets display the raw balance times the multiplier. z poke prices and settles in raw tokens throughout and converts to displayed shares only at the edge (Chapter 9). The program treats the moment a scheduled multiplier takes effect as a change of normalization: fills halt at the boundary, the market is refreshed, the normalization epoch advances, and quoting resumes against the new unit. No quote priced in the old unit can execute in the new one.

> **What these three chapters establish**
>
> The taker signs exact amounts and receives exactly those amounts or nothing. A quote executes at most once. The maker's signing key cannot withdraw, cannot exceed published limits, and cannot price outside a band around a reference it does not control. The issuer's powers over the token are detected, not assumed away. Each of these is enforced by the program at execution time, not promised by the operator.

13

The off-chain system

Between a taker's request and a signed quote sits an ordinary web service with some unusual discipline: integer arithmetic in a separate process, reservations under serializable isolation, fills confirmed only from the chain, and a rule that an unknown outcome is never treated as a failure.

13.1 Components

Pyth worker	Pricing engine	Reservations	Quote signer	Receipt check	No hedge
stream, verified publication	Rust, integers, stateless	serializable	one key, no chain powers	confirmed from chain	the pool keeps its inventory

The pricing engine is a separate Rust service that holds no state and performs no I/O beyond its own request. It accepts decimal strings, rejects anything malformed or out of range, and returns a price with its four terms. Because it is pure, a quote can be reproduced exactly from its recorded inputs.

13.2 Reservations

A quote is a promise against inventory. Two quotes must not both promise the last thousand tokens. When a quote is created the output amount is reserved against the vault's available balance inside a serializable database transaction; the reservation is released by the fill, by expiry, or by cancellation. No network call runs inside that transaction. A new quote is refused unless its output fits within the vault's on-chain balance above its floor less every outstanding reservation. The inventory, target and skew of Chapter 5 are computed from the same available balances: the on-chain vaults less the outgoing amounts of outstanding quotes. A quote's outgoing amount therefore moves the skew as soon as the quote is issued, while its incoming amount counts only once the fill lands. The 25 to 75 percent value bounds are checked again against current reservations under the market's lock immediately before the quote is signed.

13.3 Idempotency and unknown outcomes

Every order carries a client-generated key. Repeating a request with the same key returns the same order; reusing a key for a different order is refused. After the taker signs, the exact signed bytes are persisted before they are submitted. If the network does not answer, the order is *unknown*, not failed: a background worker re-examines every unresolved order every five seconds, resubmitting the identical bytes while the quote is live, until the chain shows it confirmed, failed, or expired by block height. This holds whether or not the taker's browser is open, and a pending order survives a reload of the page. A confirmed observation is never overwritten by an older response that arrives late.

13.4 The record of a fill

An order is marked confirmed, and then final, only when the receipt account at the address derived from its quote identifier exists on chain and matches it: the market, the taker, the quote identifier and hash, the input, the output and the fee. Logs, simulations and transport responses never mark a fill. Version 1.0 keeps no separate off-chain ledger of fills: the receipts, the fill events and the vault balances on chain are the record, and anyone can read them.

13.5 What runs once

The public reference poller holds a lease, so exactly one instance is active at a time. The Pyth worker runs beside the web service and holds a leadership lease for the price stream, so replicas never open duplicate streams, and a separate publication lease for each market, so at most one publication per market is in flight. It persists each signed transaction before sending it, and after a restart it retries those same bytes and declares a publication confirmed only when both the reference and its `OracleConfig` provenance on chain match what it intended. The recovery worker runs in every instance of the web service and claims each order before checking it, so no two instances check the same order at once. Web servers scale horizontally and read prices from shared storage. One consequence is that the number of people looking at prices has no effect on the load placed on any external data provider.

14

Routing and distribution

A market maker's prices matter only if flow can reach them. The design admits three ways, in increasing order of reach and of engineering. Version 1.0 runs the first.

14.1 The terminal

z poke has one terminal with two routes. ZPOKE requests a firm quote from z poke 's own pool on the ten markets that have one. Quick routes an order through the Jupiter aggregator on all 41 markets, with exact input, a slippage tolerance of 0.5 percent, the route's own trading fee and no z poke fee. The interface shows the construction of every ZPOKE price and is where the full design is visible. It is also the smallest source of flow any venue has. Most swaps on Solana begin in a wallet or an aggregator, not on a market maker's own site.

14.2 Aggregators

Aggregators route each swap to whoever offers the best price, and now run request-for-quote interfaces alongside their pool routing: a maker answers a quote request within a deadline and, if chosen, the fill settles through the aggregator's own contract. z poke 's quote is already an answer to exactly that question, so the integration would be an adapter, not a redesign. No aggregator routes into z poke 's pools in version 1.0. The integration matters for risk as well as volume. Routed flow is price-sensitive by construction, which is the elasticity ε in equation (7.1): the more of it there is, the faster skew restores inventory. Until then the pools see only the terminal's own takers.

14.3 A published-state pool

Request-for-quote has one structural limit: a router cannot simulate a price that does not exist until asked for. The longer-run design, which version 1.0 does not deploy, publishes the quote's *parameters* on chain, per market: the reference and its scale, the two half-spreads as functions of size, the inventory anchor, per-side capacity, and a validity window. A swap instruction then evaluates the same integer function as Chapter 5 against that state, with the caller's minimum output as protection, and decrements the published capacity.

Three problems are specific to this design and are part of its specification.

1. **Capacity must not be resurrected.** A price update must not restore capacity that fills have consumed since the publisher last looked. Each update names the cumulative execution cursor it was computed against, and the program accounts for intervening fills before accepting a new budget.
2. **Staleness must disable, not degrade.** Published state carries a validity window, after which the pool cannot be traded against until refreshed.
3. **Off-chain and on-chain arithmetic must agree exactly.** The router's simulation and the program must produce the same output, fee and rounding for the same snapshot, tested on shared fixtures including depleted capacity, a changed multiplier, and races between an update and a fill.

Publishing on every reference tick is not a strategy: updates cost fees and contend for the same accounts as fills. The cadence is a measured trade-off between staleness, by the option argument of section 6.2, and cost. Signed quotes and the published pool would draw on separate inventory budgets so that neither could overcommit the other.

> **What these chapters establish**
>
> The ZPOKE reference is Pyth's signed price for the underlying, verified and converted into raw-token units by the program, on the stated assumption that one USDC is one dollar, and it fails closed. Units are raw everywhere but the screen, and a multiplier change cannot be straddled. The serving system reserves before it promises, never treats an unknown outcome as a failure, and takes its record of fills from the chain. And the same quote function extends from a terminal, to an aggregator's request-for-quote, to a pool a router can simulate; version 1.0 runs the first of the three.

15

Security analysis

This chapter states what each party has to trust, what an adversary in each position can and cannot do, and the invariants the design is meant to uphold. It describes a design and its tests. The program has had an internal review; no independent third-party audit has been published, and this chapter is not a substitute for one.

15.1 Who trusts whom

Table 15.1 Trust assumptions, by party.

Party	Must trust	Need not trust
Taker	Their own wallet; the correctness of the settlement program; the owner key not to upgrade it against them; Solana's liveness; the token's issuer; USDC's value as a dollar	z poke with custody; z poke 's servers with the outcome of a fill; any price they did not see and sign
Liquidity provider	The program and the owner key, as a taker does; the quote signer to price within its limits honestly; Pyth's publishers and signing keys for the underlying price the pool trades against; the issuer exposure mapping and the assumption that one USDC is one dollar; the pool's inventory risk, which is not hedged	z poke with custody: shares are on chain and redeem without z poke 's signature; z poke 's relayer with the price, which it cannot choose
Maker	Its own key custody; Pyth's feed and verifier; the independent USDC price behind the parity guard; the program	Takers; the web tier with funds; the relayer, beyond its liveness
All	That the published program is the deployed one, and remains so while the owner key holds upgrade authority	Each other's honesty about amounts: the chain checks them

15.2 Adversaries

Table 15.2 What an adversary in each position can attempt, and what stops it.

Adversary	Attempt	Outcome
Malicious taker	Replay a quote, or execute it from another wallet or to another account	Receipt address collision; wallet and token accounts are inside the signed bytes

Adversary	Attempt	Outcome
Malicious taker	Present a valid signature over different bytes using cross-instruction offsets	Offsets must equal seven fixed values that keep key, signature and message in one instruction
Malicious taker	Wrap the fill in another program, or append instructions that depend on it	Must be top-level and last; prelude is allowlisted
Malicious taker	Hold quotes as free options, executing only when the market moves in their favour	Quotes issued for at most 10 seconds, never past the reference's expiry, under a 30-second, 64-slot ceiling on chain; the on-chain price band; one pending order per wallet; cost analysed in section 6.2
Malicious taker	Flood quote requests to reserve all inventory	Twelve quote requests per wallet per minute; one pending order per wallet; reservations expire with the quote
Compromised web tier	Alter a transaction after review	The taker's wallet shows and signs the amounts; the server accepts back only byte-identical messages
Compromised quote key	Sign quotes to itself at off-market prices	Bounded by the price band around the verified reference, per-trade size, per-window turnover and vault floors; voided by a pause from the emergency or administrator key. The signer's inventory bounds do not bind it
Compromised reference key	Publish an arbitrary on-chain reference	Refused on every PYTH_ONLY market, which all ten are, and the mode cannot be reversed without a program upgrade
Compromised or faulty relayer	Submit a stale, replayed, reordered or altered report	Signature, trusted signer and exact bytes are verified; feed, channel, session, age and a strictly later source time are checked on chain. It can only withhold updates, which lets references expire and stops quoting
Compromised emergency key	Pause markets	Denial of service only; it cannot resume or move anything
Compromised owner key	Upgrade the program, change limits, collect the protocol's fees	Nothing on chain stops an upgrade: the owner wallet is the upgrade authority, with no multisig or timelock. The deployed program gives it no way to withdraw pool inventory, but an upgrade could
Issuer action	Pause the token, enable a hook, change an authority or the multiplier	Detected by fingerprint at every fill; the market stops with a precise error
Faulty Pyth data	Sign a wrong price for the underlying	Only the session, publisher and confidence limits apply. A wrong price that passes them is quoted and settled against; the order, turnover and inventory limits bound the loss

Adversary	Attempt	Outcome
USDC depeg	USDC moves away from one dollar	New quotes stop outside $0.99–$1.01 or when the independent price is over 60 seconds old; prices are never adjusted, so a smaller move is borne by takers and the pool

Of the keys that sign in daily operation, the compromised quote key deserves most attention, because it is the hottest: it signs every quote and must be online. The design accepts that it may be lost and makes the loss finite. With band b, turnover cap W per fixed window, and a reaction time of T windows before the emergency key pauses the market, the most an attacker can extract is

$$L_{\max} \leq \frac{b}{10^4} \cdot W \cdot \left(\lceil T \rceil + 1\right), \tag{15.1}$$

further limited by the vault floors; the extra window is the one the attack may begin in, since fixed windows let fills either side of a boundary draw on two caps. The live markets have a one percent band and turnover caps of 500 USDC per window on NVDAx and 100 USDC on each of the other nine, so each window costs at most 5 USDC on NVDAx and 1 USDC on each of the others, 14 USDC across all ten. These are configuration choices, visible on chain, and a taker can read them. The band is measured from a reference the attacker cannot move: before verification, a stolen reference key and a stolen quote key together could have shifted the band itself; on the live markets the reference key can no longer publish at all.

15.3 Invariants

The protocol is specified by eleven invariants. Each is the subject of adversarial tests against the compiled program.

1. A successful quote cannot execute twice, under any blockhash, fee payer or arrangement of instructions.
2. A caller cannot substitute a mint, token program, market, vault, recipient, fee account, trader or deployment.
3. The taker's debit never exceeds the signed amount, and the credit equals the signed amount.
4. A failed execution leaves no receipt, no transfer and no change to any counter.
5. Arithmetic cannot wrap, cannot make available inventory negative, and cannot cross unit domains implicitly.
6. Every successful execution satisfies policy, validity, token configuration, price band, inventory and turnover conditions at the time of execution, not of signing.
7. The quote authority cannot withdraw, upgrade, list assets, change permissions or raise its own limits.
8. Counters and versions are monotonic, and a turnover window cannot be reset early by timing or by a manipulated clock.

9. A change of economic configuration invalidates outstanding quotes that are incompatible with it.
10. Logs and simulations alone cannot mark an order filled; only a matching receipt on chain can.
11. Every market must present its canonical oracle account at every fill and every reference write; on a `PYTH_ONLY` market a reference can be written only from a report verified in the same transaction, and the operator-published path cannot be used or restored.

15.4 What is out of scope

A pause is effective only once the chain enforces it: a local flag stops new signatures but cannot recall a transaction already sent, and nothing can be enforced while the chain is halted. Version 1.0 places no hedges, so the pool's inventory risk is not a security property at all; it is borne by the liquidity providers (Chapter 18). The invariants above hold for the deployed binary and not against the upgrade authority, which could replace it. Verification establishes that Pyth signed a price for the underlying, not that the price was right, that the token trades at it, or that one USDC is worth one dollar; the issuer exposure mapping and the USDC parity are assumptions, not checks. A signature establishes that a price was offered and accepted, not that it was fair. And failed transactions cost their network fee, as do the relayer's publications, for which no daily spending limit is yet enforced.

16

Performance

Speed matters in market making for one reason: the staler the quote, the more the option of section 6.2 is worth to someone else. This chapter reports what was measured and is explicit about where.

16.1 Pricing

Table 16.1 Measured timings. The engine figures are HTTP round trips to the release build of the Rust engine for one firm quote with the live NVDAx pool terms, 500 calls after 50 warm-up calls, on a development machine over loopback.

Quantity	Value
Firm quote, engine round trip, median	0.34 ms
Firm quote, engine round trip, 99th percentile	1.12 ms
Underlying price stream	one update per 200 ms per market, three redundant connections
Verified on-chain publication	about every 2 s per ZPOKE market with recent quote demand
Public reference refresh, Quick and charts	every 2.5 s, one upstream request for all markets
Solana slot	≈ 400 ms

Pricing a firm quote takes well under one percent of a slot. Pricing is therefore never the bottleneck; the age of a quote is set by how often the reference is published on chain, how long the publication takes to land, and how long the taker takes to sign. For context, at 35 percent volatility the reference diffuses by about 0.9 basis points per slot and about 3 basis points over the five seconds a source may age before a quote is refused. A one-standard-deviation move over that interval is smaller than the base spread, while a curve's price may be stale by its entire fee (equation 3.5).

After the verified references went live, a 1 USDC buy quote was prepared on each of the ten markets and its transaction simulated on mainnet. All ten passed at the first attempt, with source ages of 1.55 to 2.59 seconds and 8.25 to 9.13 seconds left for the taker to review the quote. These were simulations; no swap was sent.

16.2 Settlement

A fill measured 108,037 compute units on a local validator, under eight percent of the 1.4 million a transaction may request, and takes 1,119 of 1,232 bytes. It locks a fixed set of accounts per market for writing: the risk state, the liquidity pool record, two vaults and a fee vault, plus the taker's two token accounts and the new receipt. It reads the reference and the `OracleConfig`, which a publication writes, so fills and publications in one market are ordered against each other. Fills in different markets touch disjoint accounts and execute in parallel. Fills in the same market serialise on its risk state, which is deliberate: the turnover cap is a per-market quantity. Confirmation is typically one to two seconds and finality about thirteen.

Each publication is a transaction the relayer pays for. Ten markets refreshed every two seconds around the clock would be up to 432,000 attempts a day. Demand-based scheduling, cooling of idle markets and the closed hours reduce that; the landed count and its cost have to be measured in operation.

17

Comparison

The same price path, two venues, the same capital. This chapter puts the constant-product pool of Chapter 3 and the quote of Chapter 5 side by side in a deterministic simulation whose code is public and whose parameters the reader can change on the site.

17.1 Set-up

The reference follows a geometric random walk for 1,500 slots of 400 milliseconds, ten minutes, from a fixed seed so that every reader sees the same path. The constant-product pool starts balanced at the reference with total value V and fee f. At each slot an arbitrageur who observes the reference one slot late moves the pool to the edge of the band (3.5) if and only if it is outside it. The z poke quote is recomputed each slot from a reference one slot old, with the pool at target. At each slot we record each venue's midpoint and the cost of a buy of fixed size against the true reference at that instant. One slot is an idealisation: the live reference is published about every two seconds from a source up to five seconds old (Chapter 8), so the live quote lags the underlying by more than the model's does.

This is a model, not a measurement of any named venue. It omits the curve's just-in-time liquidity and the quote's rejection risk, it holds z poke 's inventory at target, and it charges neither side for gas. Each venue's cost includes its own fee, taken from the input: the curve's, and z poke 's five basis points, exactly as the program takes it.

The quote uses the live configuration of Table 5.1: a base of 5 basis points, an impact of 400 ppm per 1,000 USDC, a session term of zero and the 5 basis point fee. It omits the confidence cost, which on the live pools adds twice the feed's relative confidence to the half-spread, 2 basis points at a confidence of 1 basis point, so every z poke cost in Table 17.1 is lower than a live quote by that amount. With the pool held at target the skew is zero, so the way the live pools apply skew does not affect these figures. What the model does not share with the live pools is scale. They accept at most 100 USDC per order on NVDAx and 20 USDC on the others, so the baseline order of 10,000 USDC, and every row of Table 17.1 above 100 USDC, are far beyond anything they quote; those rows show how the two mechanisms scale at equal capital. The last three rows are run at the live order limits.

17.2 Results

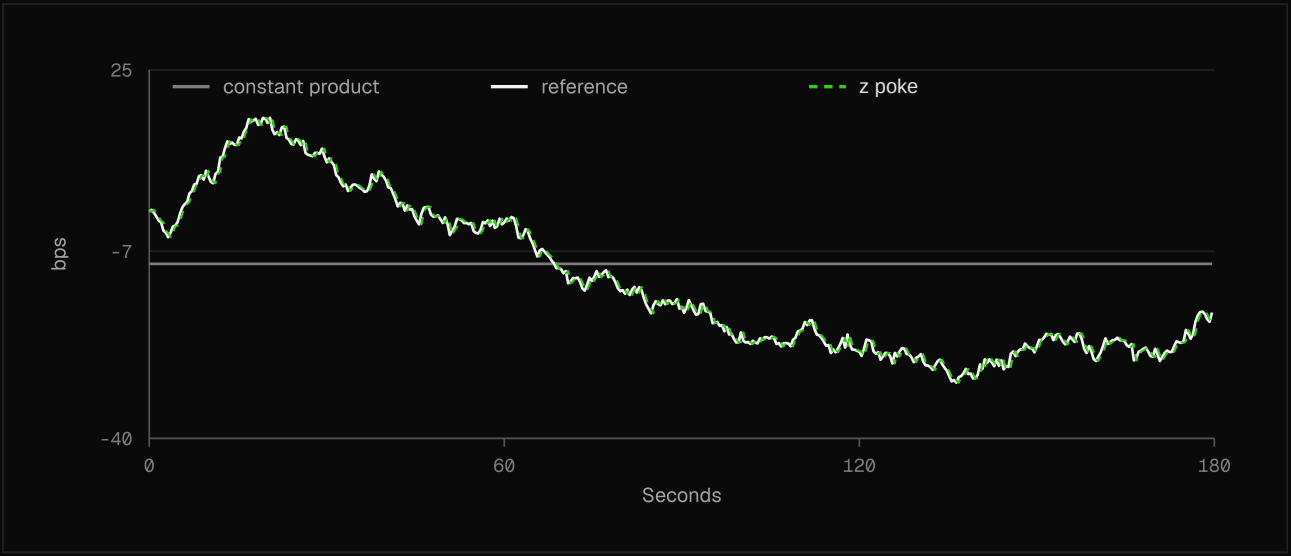


Figure 17.1 Midpoints over the last three minutes of the baseline path, in basis points from the start of the window. White: reference. Orange, dashed: z poke , one slot behind. Grey, stepped: the constant-product pool, which moves only when the reference leaves its fee band.

Table 17.1 Eleven scenarios, with the live pools' parameters and fee on the z poke side and its confidence cost set to zero. Cost is the mean over the path of a buy's price against the true reference, all fees included. Tracking is the mean absolute distance of the venue's midpoint from the reference. Stale is the share of slots in which that distance exceeded 5 bps.

Scenario	Curve cost	z poke cost	Curve tracking	z poke tracking	Curve stale	z poke stale
Baseline: 10,000 order, 2M pool, 30 bps fee, 35% vol	128.1 bps	50.0 bps	8.01	0.71	59%	0%
Curve fee 5 bps	105.5 bps	50.0 bps	3.08	0.71	13%	0%
Curve fee 100 bps	199.0 bps	50.0 bps	8.01	0.71	59%	0%
Curve pool 20M	38.1 bps	50.0 bps	8.01	0.71	59%	0%
Curve pool 0.5M	428.1 bps	50.0 bps	8.01	0.71	59%	0%
Volatility 80%	130.1 bps	50.0 bps	13.76	1.63	79%	1%
Order 1,000	38.1 bps	14.0 bps	8.01	0.71	59%	0%
Order 25,000	278.1 bps	110.1 bps	8.01	0.71	59%	0%
Order 100, the NVDAx limit	29.1 bps	10.4 bps	8.01	0.71	59%	0%
Order 20, the limit on the other pools	28.3 bps	10.1 bps	8.01	0.71	59%	0%
Order 100, curve fee 5 bps	6.5 bps	10.4 bps	3.08	0.71	13%	0%

17.3 Reading the table

Four things stand out. First, the curve's cost is dominated by impact, exactly as (3.2) says, and at large sizes so is the quote's: at 10,000 USDC the live size term of 4 basis points per 1,000 contributes 40 of its 50 basis points. Against a pool of two million dollars the quote is still about two and a half times cheaper, but a curve ten times larger fills the same order for 38 basis points, less than the quote. The live size term was set for orders of 100 USDC, not 10,000. Second, in the baseline the curve is never arbitraged at all over ten minutes, because at 35 percent volatility the reference does not leave a 30 basis point band that quickly; its midpoint is simply wherever it was left, and it is more than five basis points from the truth most of the time. Lowering the fee to five basis points makes it track better and brings in arbitrage, which is paid for by the pool.

Third, at the live order limits the quote costs about 10 basis points before the confidence cost, nearly all of it base and fee, and the comparison turns on the curve's fee. Against 30 basis points the curve charges 28 to 29, nearly three times as much. A curve with a 5 basis point fee charges 6.5, less than the quote, because at these sizes its cost is little more than its fee: 5 basis points plus about one of impact. At the baseline depth and volatility the break-even curve fee is about 9 basis points, and about 2 basis points higher for every basis point of confidence the live feed reports. What the low-fee curve does not do is track: it is still four times further from the reference, and it is corrected by arbitrageurs 191 times in ten minutes, each time at the pool's expense. Fourth, the live quote does not price the clock either. The engine supports the session term of Figure 4.1, but the live pools set it to zero. They do not quote at all at weekends or on exchange holidays, and in the pre-market, post-market and overnight sessions they charge what they charge on a Tuesday afternoon, apart from the confidence cost, whose limit is looser outside the regular session. What bounds a pool's exposure across the close is its order, turnover and inventory limits.

What this chapter establishes

With equal capital, on the same path and with the live pools' parameters and fee, the quote tracks the reference four to eleven times more closely than a constant-product pool in every scenario. Its cost advantage depends on the curve, and the figures that follow leave out the confidence cost, which adds twice the feed's relative confidence. Against a two-million-dollar curve charging 30 basis points it is about two and a half to three times cheaper at every size tested, and four times cheaper against a 100 basis point fee. A curve with a fee under about 9 basis points is cheaper at the live order sizes, and a curve with ten times the capital is cheaper at 10,000 USDC. What holds throughout is freshness, not price.

18

Economics

Each ZPOKE pool earns a fee and a spread, bears inventory risk, and belongs to its liquidity providers in proportion to their shares. There is no reward token, no liquidity mining and no promised yield. This chapter states who pays what to whom.

18.1 Revenue and cost

Table 18.1 Flows per fill.

Flow	From	To	Size
Fee, providers' part	Taker	Pool vaults, for the liquidity providers	80% of 5 bps of input, rounded down, in the input token
Fee, protocol's part	Taker	Protocol fee vault	the remaining 20%; the whole fee is exact and enforced on chain
Buyback	Protocol fee revenue	$ZPOKE bought on the open market	funded from protocol fee revenue
Spread	Taker	Pool, as the distance between the executed price and the reference	base + size + confidence cost + session, around a centre moved by skew; session is zero on the live pools, and skew can move the price the restoring side pays across the reference
Network fee and rent	Taker	Solana validators; the taker's own accounts	small, in SOL
Adverse selection	Pool	Better-informed takers	measured by markout
Rebalancing	—	—	none in version 1.0: no hedges are placed
Inventory mark-to-market	—	Pool, either sign	zero in expectation, carried by the providers, bounded by the pool's capital

From equation (7.2), a market pays for itself when spread and fee exceed markout and rebalancing on average. On the live NVDAx pool, the largest order it accepts, 100 USDC with the pool at its target, pays a half-spread of 5.4 basis points plus the confidence cost, and a fee of 5: about 0.10 USDC in all before the confidence cost, and 0.12 USDC at a confidence of 1 basis point. Of the fee,

0.01 USDC goes to the protocol and the rest stays in the pool with the spread. The realistic question is how much of that the informed take back. It is answered by data, per market, and a market that does not pay can be paused.

18.2 Capital

Each live pool was opened by a founding deposit from z poke 's treasury, half bought as the token and half kept as USDC: 5,000 USDC for NVDAx and 1,000 USDC each for SPYx, QQQx, GLDx, TSLAx, AAPLx, MSFTx, METAx, COINx and CRCLx, 14,000 USDC in all. On 27 September 2026 the treasury deposited again into the nine smaller pools, 18,412 USDC in all, bringing each to about 3,000 USDC of value; the ten pools together hold about 32,000 USDC. The pool's target inventory is not fixed by that deposit: it is half the pool's available value, measured in the token at the current reference (Section 5.1), so it moves with the price and with deposits, redemptions and retained fees. The treasury holds shares for its deposit like any other provider. Markets are added as each pool is funded; the six newest, TSLAx, AAPLx, MSFTx, METAx, COINx and CRCLx, were funded together on 25 September 2026 with 6,000 USDC.

After the founding deposit anyone can deposit both assets in the vaults' current ratio and receive shares, and redeem them for both assets pro rata (Chapter 11). A share is a claim on the vaults and nothing else. It carries no reward token, no advertised rate and no guaranteed value in USDC: a provider owns its part of the inventory's price moves, of the pool's trading gains and losses, and of the 80 percent of every fee that stays in the vaults. Takers have no claim on the pool.

These pools are small. Their per-order and turnover limits are set small relative to them, and condition (5.5) holds for them as Section 5.3 shows. For comparison, Proposition 3.1 requires twenty million dollars of a constant-product pool to fill 10,000 USDC at 15 basis points; the live pools do not quote orders of that size.

18.3 $ZPOKE

z poke 's token is $ZPOKE and its link to the venue is one flow: the protocol fees z poke collects go to buying $ZPOKE back on the open market. The more the venue is used, the more there is to buy with.

The mechanism in this paper does not otherwise depend on the token. The fee is paid in the asset being traded, control is exercised by keys whose powers are enumerated in Chapter 11, and liquidity providers earn only through their pool shares, never in $ZPOKE A taker never needs to hold $ZPOKE to trade, every taker pays the same fee, and holding the token confers no claim on the pools, which belong to their share holders. Buybacks follow fee revenue, which follows volume; they are not a promise of price or of return.

19

Risks and disclosures

Stated plainly, for takers and for anyone evaluating the design.

19.1 The asset

A tokenized stock is a claim on its issuer, not a share. It confers no vote. Its value depends on the issuer and its custodian continuing to hold the underlying and honour redemptions. The issuer can pause the token, freeze an account, and move or burn any balance, and may be required to. The token can trade away from the share, most of all outside the regular session. It is not offered to residents of the United States, the United Kingdom, Canada, Australia and certain other jurisdictions, and holding it in breach of the issuer's terms exposes the holder to those powers. Every price is in USDC, which carries its own issuer and peg risk. ZPOKE converts the share's dollar price into USDC by assuming that one USDC is one dollar, and stops quoting only when an independent USDC price leaves $0.99 to $1.01; a smaller depeg, and the risk of redeeming USDC, fall on the user.

19.2 The venue

z poke is a single counterparty on ten markets. If a pool is paused, depleted or redeemed by its providers, that market is unavailable here, though the tokens remain the holder's to move. ZPOKE gives no quotes at weekends, on exchange holidays, or while the underlying's session is closed or cannot be confirmed, and a market may show no price for a moment while its reference is refreshed and confirmed, or for longer when the reference fails a check. A signed quote is not a guaranteed fill. Outside the regular session the underlying trades thinly and its price may be less certain; on the live pools, where the session term is zero, that is priced only by the base spread and the confidence cost, and bounded by the order, turnover and inventory limits. The reference is the price of the underlying share, and the token can trade elsewhere at a different price for as long as the gap persists; the pools do not hedge, rebalance or use the issuer's creation and redemption to close it. The pools may lose money, may have their limits reduced or be paused as a result, and z poke has no obligation to quote.

A liquidity provider owns a share of an unhedged inventory. Its value in USDC moves with the token, with trading gains and losses, and with the fees the pool keeps. There is no reward token, no promised rate and no guaranteed value. A redemption returns both assets in the vaults' current ratio, not the amounts deposited.

19.3 The technology

The settlement program may contain errors that testing and an internal review did not find; no independent third-party audit has been published. The program is upgradeable by a single owner key, with no multisig and no timelock, so its rules are only as fixed as that key is safe and honest. Solana may be congested or halted, during which nothing can be traded or paused. Keys can be compromised: on-chain limits bound the consequence of a lost quote key for the pools and do not eliminate it. The ZPOKE reference is verified as Pyth's signed price, which does not make it correct: a wrong price that Pyth signs and that passes the session, publisher and confidence limits would produce wrong quotes, as would an error in the issuer exposure mapping. If the relayer stops publishing, references expire and quoting stops. A taker's own key is their own responsibility, and a transfer once signed cannot be reversed.

19.4 Legal

The regulatory treatment of tokenized securities and of venues that trade them differs across jurisdictions and is changing. Nothing in this paper is an offer of securities or investment advice. z poke applies the issuer's distribution restrictions through attestation and geographic controls; it does not and cannot verify a taker's eligibility beyond that, and the taker remains responsible for their own compliance and tax.

20

Admission, growth and conclusion

A venue is a list of markets and a set of limits as much as it is a mechanism. This chapter states the rule by which a market is listed and its limits are set, what lies beyond version 1.0, and what the paper has shown.

20.1 How a market is admitted

Version 1.0 quotes ten markets, NVDAx, SPYx, QQQx, GLDx, TSLAx, AAPLx, MSFTx, METAx, COINx and CRCLx. Each meets the four tests of Table 20.1, and markets are added as each pool is funded. The other 31 of the 41 listed tokens trade through Quick only. A market that stops passing the first two tests stops being quoted at once. None of the tests is a judgment about the company behind the ticker.

Table 20.1 Admission tests, applied to each market before its pool opens. The first two are re-applied at every quote and every fill.

Test	Requirement	Source
Reference	A Pyth Pro feed for the underlying, pinned by identifier, with a verified issuer exposure mapping, and a fresh verified report in every session the market is quoted in, under the rules of Chapter 8	Chapter 8
Token	The mint's extension set is on the allowlist and its fingerprint is registered	Chapter 12
Round trip	The pool's target value and largest order satisfy condition (5.5), so that skew cannot be harvested	Proposition 5.3
Funding	A founding deposit from the treasury opens the pool, with per-order and turnover limits small relative to it	Section 18.2

Limits follow evidence in the same way. A market opens with a low per-order size and a low turnover cap per fixed window, both enforced by the program: 100 and 500 USDC on NVDAx, 20 and 100 USDC on the others. They are raised when the market's fills have settled with receipts and vault balances that agree, and its spread and fee have exceeded markout over a sustained period, in the sense of equation (7.2). They are lowered, by the same keys and under the same hard bounds of Chapter 11, when either stops being true.

20.2 Beyond 1.0

The following extend version 1.0 without changing the quote function. None of them is deployed.

1. **More markets.** ZPOKE on more of the 31 listed tokens that are Quick only today, each admitted by Table 20.1 and opened when its pool is funded.
2. **Hedging and rebalancing.** Version 1.0 carries its inventory, and skew is the only force that restores it. A hedge that lays fills off in the underlying during the regular session, and active rebalancing when inventory leaves a band (Section 7.1), would make the size term pay for a real cost (Section 6.3) and let pools take larger orders.
3. **The session term.** The engine already applies the schedule of Figure 4.1 to any pool whose session parameter is set. The live pools set it to zero; turning it on is a configuration change, re-estimated per name by equation (4.3).
4. **Aggregator request-for-quote.** Answering an aggregator's quote requests (Section 14.2) would bring routed flow into the pools, which by equation (7.1) also restores inventory faster.
5. **The published-state pool.** Section 14.3 specifies a pool whose quote parameters live on chain, so that any router can simulate it. It evaluates the same integer function as Chapter 5 and draws on a separate inventory budget. It would need its own review before deployment.
6. **Control.** In version 1.0 one owner key is the program's upgrade authority, its administrator and its treasury. A multisig, a timelock on upgrades, an immutable program and an independent audit are each possible; none is part of 1.0.
7. **Longer sessions.** As venues for the underlying shares extend their hours, the session calendar of Chapter 4 changes and the session term falls with it. Nothing else in the quote needs to know. A market that can be hedged around the clock is priced as one.
8. **Further issuers and asset classes.** The quote depends on an asset only through its reference, its calendar and its token. Tokenized funds, treasuries and equities from other issuers could be admitted by the tests of Table 20.1, each with the session structure of its own underlying market.

20.3 Conclusion

Tokenized equities inherited their market structure from assets that never close and can always be hedged. This paper has argued that the inheritance does not fit. A passive curve charges the taker for capital the market has not supplied, pays arbitrageurs to tell it the price, and is blind to the one fact that most determines the maker's risk, which is whether New York is open.

z poke replaces the curve with a quote: four published terms in integer arithmetic, ordered, monotone in size, bounded in skew and closed to round trips. It replaces trust in the maker's word, fill by fill, with a program that enforces exact amounts or nothing, exactly once, inside a band around a reference the program itself verifies from Pyth's signed price for the underlying, under limits the maker's quote key cannot lift. The program is upgradeable by z poke 's owner key, and that trust remains, as does trust in the quote signer within those limits, in Pyth, in the issuer exposure mapping and in the assumption that one USDC is one dollar. And it shows its working. The taker sees every term of the price before signing, and can verify every byte of what they signed afterwards.

With equal capital on a common path and the live pools' parameters and fee, that design tracks its reference four to eleven times more closely than a constant-product pool, and before its confidence cost costs the taker less than half as much as a curve charging 30 basis points; a curve whose fee is close to z poke 's own is cheaper for small orders. Version 1.0 runs it at small scale, on ten markets, only while a US session is open, with the session term at zero and no hedging. The remaining gap between a share and its token is the clock. The design can price the clock; the live pools do not yet. They stop quoting when the underlying is closed and bound the open sessions with limits.

A

Notation

Table A.1 Symbols used in this paper.

Symbol	Meaning	Unit
p	Reference price of one raw token	micro-dollars
S	Price of one underlying share, from its Pyth feed	dollars
C	Confidence of that price, as stated in the feed	dollars
E	Issuer exposure: shares per displayed token unit, 1 for the listed xStocks	–
m	Active multiplier: displayed units per raw token, which are shares when E = 1	–
$I,\ I^*$	Available pool inventory and its target, half the available value	token atoms
Y	Available USDC in the pool	USDC atoms
x	Fractional deviation of inventory from target	–
w	Token's share of the pool's available value	–
b	Base half-spread	ppm
κ	Impact coefficient per 1,000 USDC of notional	ppm
s	Session premium	ppm
c	Confidence cost, twice C over S	ppm
$k,\ k_{\max}$	Skew, which moves the centre of the quote, and its bound	ppm
z	Size term	ppm
h	Half-spread around the skewed centre	ppm
$P_{\text{ask}},\ P_{\text{bid}}$	Quoted prices	micro-dollars
N	Order notional	micro-dollars, or USDC where stated
$a,\ a',\ \varphi$	Input, net input, fee	atoms of the input asset
f	Fee rate	basis points
$V,\ L$	Total value and liquidity parameter of a constant-product pool	dollars
$\sigma,\ \theta$	Annual volatility; share of variance arriving while the market is shut	–
τ	Lifetime of a firm quote	seconds
λ	Price of risk per unit standard deviation	–

Symbol	Meaning	Unit
$\varepsilon,\ \Lambda,\ \rho$	Flow elasticity, arrival rate, inventory reversion rate	–

One basis point is 10^{-4}; one ppm is 10^{-6}. A token atom is 10^{-8} of a raw stock token or 10^{-6} of a USDC. Floors and ceilings are as written; every other division in the engine truncates toward zero.

B

Markets

This paper's original forty-one xStocks markets each use a Token-2022 stock token with eight decimals against USDC with six. Mint addresses are taken from the issuer's registry and re-verified on chain every five minutes; a token is identified by this address and never by its symbol. All 41 are listed in Quick. Ten also have an ZPOKE pool on z poke 's settlement program, currently paused after withdrawal. The expanded issuer catalog, including Backpack Securities, is listed in the current market directory.

Table B.1 Listed markets, routes and mint address.

Token	Tracks	Class	Routes	Mint
NVDAx	NVIDIA	Equity	Quick, ZPOKE (paused)	Xsc9qvGR1efVDFGLrVsmkzv3qi45LTBjeUKSPmx9qEh
SPYx	S&P 500 ETF	ETF	Quick, ZPOKE (paused)	XsoCS1TfEyfFhfvj8EtZ528L3CaKBDBRqRapnBbDF2W
TSLAx	Tesla	Equity	Quick, ZPOKE (paused)	XsDoVfqeBukxuZHWhdvWHBhgEHjGNst4MLodqsJHzoB
AAPLx	Apple	Equity	Quick, ZPOKE (paused)	XsbEhLAtcf6HdfpFZ5xEMdqW8nfAvcsP5bdudRLJzJp
MSFTx	Microsoft	Equity	Quick, ZPOKE (paused)	XspzcW1PRtgf6Wj92HCiZdjzKCyFekVD8P5Ueh3dRMX
GOOGLx	Alphabet	Equity	Quick	XsCPL9dNWBMvFtTmwcCA5v3xWPSMEBCszbQdiLLq6aN
AMZNx	Amazon	Equity	Quick	Xs3eBt7uRfJX8QUs4suhyU8p2M6DoUDrJyWBa8LLZsg
METAx	Meta Platforms	Equity	Quick, ZPOKE (paused)	Xsa62P5mvPszXL1krVUnU5ar38bBSVcWAB6fmPCo5Zu
AVGOx	Broadcom	Equity	Quick	XsgSaSvNSqLTtFuyWPBhK9196Xb9Bbdyjj4fH3cPJGo
AMDx	AMD	Equity	Quick	XsXcJ6GZ9kVnjqGsjBnktRcuwMBmvKWh8S93RefZ1rF
INTCx	Intel	Equity	Quick	XshPgPdXFRWB8tP1j82rebb2Q9rPgGX37RuqzohmArM
ORCLx	Oracle	Equity	Quick	XsjFwUPiLofddX5cWFHW35GCbXcSu1BCUGfxoQAQjeL
CRMx	Salesforce	Equity	Quick	XsczbcQ3zfcgAEt9qHQES8pxKAVG5rujPSHQEXi4kaN
CSCOx	Cisco	Equity	Quick	Xsr3pdLQyXvDJBFgpR5nexCEZwXvigb8wbPYp4YoNFf
IBMx	IBM	Equity	Quick	XspwhyYPdWVM8XBHZnpS9hgyag9MKjLRyE3tVfmCbSr

Token	Tracks	Class	Routes	Mint
NFLXx	Netflix	Equity	Quick	XsEH7wWfJJu2ZT3UCFeVfALnVA6CP5ur7Ee11KmzVpL
PLTRx	Palantir	Equity	Quick	XsoBhf2ufR8fTyNSjqfU71DYGaE6Z3SUGAidpzriAA4
COINx	Coinbase	Equity	Quick, ZPOKE (paused)	Xs7ZdzSHLU9ftNJsii5fCeJhoRWSC32SQGzGQtePxNu
HOODx	Robinhood	Equity	Quick	XsvNBAYkrDRNhA7wPHQfX3ZUXZyZLdnCQDfHZ56bzpg
MSTRx	Strategy	Equity	Quick	XsP7xzNPvEHS1m6qfanPUGjNmdnmsLKEoNAnHjdxxyZ
CRCLx	Circle	Equity	Quick, ZPOKE (paused)	XsueG8BtpquVJX9LVLLEGuViXUungE6WmK5YZ3p3bd1
GMEx	GameStop	Equity	Quick	Xsf9mBktVB9BSU5kf4nHxPq5hCBJ2j2ui3ecFGxPRGc
JPMx	JPMorgan Chase	Equity	Quick	XsMAqkcKsUewDrzVkait4e5u4y8REgtyS7jWgCpLV2C
GSx	Goldman Sachs	Equity	Quick	XsgaUyp4jd1fNBCxgtTKkW64xnnhQcvgaxzsbAq5ZD1
BACx	Bank of America	Equity	Quick	XswsQk4duEQmCbGzfqUUWYmi7pV7xpJ9eEmLHXCaEQP
Vx	Visa	Equity	Quick	XsqgsbXwWogGJsNcVZ3TyVouy2MbTkfCFhCGGGcQZ2p
MAx	Mastercard	Equity	Quick	XsApJFV9MAktqnAc6jqzsHVujxkGm9xcSUffaBoYLKC
BRK.Bx	Berkshire Hathaway	Equity	Quick	Xs6B6zawENwAbWVi7w92rjazLuAr5Az59qgWKcNb45x
WMTx	Walmart	Equity	Quick	Xs151QeqTCiuKtinzfRATnUESM2xTU6V9Wy8Vy538ci
KOx	Coca-Cola	Equity	Quick	XsaBXg8dU5cPM6ehmVctMkVqoiRG2ZjMo1cyBJ3AykQ
MCDx	McDonald's	Equity	Quick	XsqE9cRRpzxcGKDXj1BJ7Xmg4GRhZoyY1KpmGSxAWT2
PGx	Procter & Gamble	Equity	Quick	XsYdjDjNUygZ7yGKfQaB6TxLh2gC6RRjzLtLAGJrhzV
XOMx	Exxon Mobil	Equity	Quick	XsaHND8sHyfMfsWPj6kSdd5VwvCayZvjYgKmmcNL5qh
CVXx	Chevron	Equity	Quick	XsNNMt7WTNA2sV3jrb1NNfNgapxRF5i4i6GcnTRRHts
LLYx	Eli Lilly	Equity	Quick	Xsnuv4omNoHozR6EEW5mXkw8Nrny5rB3jVfLqi6gKMH
UNHx	UnitedHealth	Equity	Quick	XszvaiXGPwvk2nwb3o9C1CX4K6zH8sez11E6uyup6fe
JNJx	Johnson & Johnson	Equity	Quick	XsGVi5eo1Dh2zUpic4qACcjuWGjNv8GCt3dm5XcX6Dn
QQQx	Nasdaq-100 ETF	ETF	Quick, ZPOKE (paused)	Xs8S1uUs1zvS2p7iwtsG3b6fkhpvmwz4GYU3gWAmWHZ
TQQQx	3x Nasdaq-100 ETF	ETF	Quick	XsjQP3iMAaQ3kQScQKthQpx9ALRbjKAjQtHg6TFomoc

Token	Tracks	Class	Routes	Mint
VTIx	Total Market ETF	ETF	Quick	XsssYEQjzxBCFgvYFFNuhJFBeHNdLWYeUSP8F45cDr9
GLDx	Gold ETF	ETF	Quick, ZPOKE (paused)	Xsv9hRk1z5ystj9MhnA7Lq4vjSsLwzL2nxrwmwtD3re

USDC mint: EPjFWdd5AufqSSqeM2qN1xzybapC8G4wEGGkZwyTDt1v.

Table B.2 The ten ZPOKE markets, their initial funding in USDC, 14,000 in all, their per-market limits in USDC notional, and the Pyth Pro feed of the underlying each is priced from, with its numeric identifier. GLD is the fund, not the spot price of gold.

Market	Initial funding	Largest order	Per fixed 60 s window	Underlying feed	Feed id
NVDAx	5,000	100	500	Equity.US.NVDA/USD	1314
SPYx	1,000	20	100	Equity.US.SPY/USD	1398
QQQx	1,000	20	100	Equity.US.QQQ/USD	1363
GLDx	1,000	20	100	Equity.US.GLD/USD	1157
TSLAx	1,000	20	100	Equity.US.TSLA/USD	1435
AAPLx	1,000	20	100	Equity.US.AAPL/USD	922
MSFTx	1,000	20	100	Equity.US.MSFT/USD	1292
METAx	1,000	20	100	Equity.US.META/USD	1272
COINx	1,000	20	100	Equity.US.COIN/USD	1042
CRCLx	1,000	20	100	Equity.US.CRCL/USD	1683

Table B.3 ZPOKE program, keys and accounts on mainnet. The program is upgradeable; the owner wallet is its upgrade authority and is also the administrator, the treasury and the maker identity of every market. Each market's oracle account is its OracleConfig, in PYTH_ONLY mode since the upgrade at slot 450,310,590.

Account	Address
Settlement program	tDDsGm8ga9hETFRCut3mb7L9kkBiS4xrBEsXmEpgB7c
Owner wallet	FyRz7wPjrFqj6RAwk1EcrJfhFaEpnChXAm21MrdGKyw6
Quote authority	E9jZWuYZzXwaGrQnX245jG1Ay54kqKRM9FJuuDtx3u6y
Pyth Pro verifier program	pytd2yyk641x7ak7mkaasSJVXh6YYZnC7wTmtgAyxPt
Pyth Pro verifier storage	3rdJbqfnagQ4yx9HXJViD4zc4xpiSqmFsKpPuSCQVyQL
NVDAx market	HSLNM4FEvLeE3XMbP5m2B6Zthdf8uC29FGbajLkGk7mk
NVDAx liquidity pool	8mJYU2UKEBhNm6ZrYkeHvxBW2W1TKSG2dEvGxWkjEBsc

Account	Address
NVDAx oracle	7FyqvSc9i8byqd8XvvdQYdmcP32HsR8NyWs22xc4XSZ6
SPYx market	9Evw9q53MYkMwHSDDdW1eDQfs2GhzCuptVAintPBLnv9
SPYx liquidity pool	Bog44oC7TTMzJD2ciDxd8vqJAhc9oGVsgDnicQB87mRa
SPYx oracle	3rVd6VYrwreWRiG4dmWrQt36xuQ9UtUGLQCK61ra28fE
QQQx market	68UdxwfbkBhDjcs9X3yTTVmSCs72gx71NYBYFRFnRXGq
QQQx liquidity pool	Bdb5WBPdFjfwouHdymZZFEV5xjNphuQnwJxY4uCpd74
QQQx oracle	CPGMXnm8qCnUNCEz1qh4DgJMQLBmwwQqC3TRTeYsYj6r
GLDx market	14D1csLkWDfvUdSo9U6ZnMXCCgjwGdFNuvou1DTXWWXp
GLDx liquidity pool	Enq6TK7xLGE6XZJCQKaLjD5jiAZHt54Kxv9WeC4PsQnq
GLDx oracle	EaxSRbNVqg8FizEA8xtBTFEFsRvVnxt12PdgB8wmXFw7
TSLAx market	33TCkGzGNxJqb6L7p76yy4LVs1m51UbsFaZjqiUBG4cC
TSLAx liquidity pool	52kbqqTNftDnkjMCesp7F2CFqywgibr67R9p65MvGzaV
TSLAx oracle	2U82NKtANesFs6U8PujXskvLJdSv4V5DnaQBe65j83Fp
AAPLx market	CAB9J7ZDRCxDWDrn8R9KVhRmEHewm9gToPPZg5iDfrya
AAPLx liquidity pool	RXjTKLJiYuQveSiSsH63nXhUQpLoy8tY4FEHws7Abdm
AAPLx oracle	BFz7fxNzAixp4f252osy6kiF7fsPmASRitHXgG6y5H5L
MSFTx market	BfDMgzLU1nsgaUzmGPLdJePx5UgD8fJ7PoxLFkpiPRw4
MSFTx liquidity pool	8eHS8ZSdHUvXAJo2681DAXc823UKRHXSctSCrjptPKtT
MSFTx oracle	2gVrbBaHUbMtyfuKiwpFQEBbEREtoNcvMaoavDESQnZF
METAx market	FXCyuxGefk58enYXCcPduRfWSU1LBj4hR2T6azSExMDe
METAx liquidity pool	gdQApwzhkD7EBHeRjMCx8WeDND8i3qX64BUBN8gp9VH
METAx oracle	81pU3Xpb4BqCRrXxjcV28HJUEmyGMnNMrK48JYUfZc1u
COINx market	7KqSwQPjBj9GkDyZhZLT2g4kH8KNQ1goWVyMdUdS3mP4
COINx liquidity pool	FAqREUd1HZyvsQGWuRtoQ5cQqmBhHy1euojV4UXT9Da8
COINx oracle	7pQDSfzpGKruvYA9Z6MdVWxDKSvQYWrZ1a2wf9qJ3WwH
CRCLx market	3K9C97nmppdKXsrnpdqnSpfB9TsvVpjxArNZXmzjYnEP
CRCLx liquidity pool	CBHCM1LYq78z46CJctT9RhJLaCFJuJp8parkwbKkUywC
CRCLx oracle	5bHHjgHPAB45bVWSZLU7NVuc7iKTBzngLjkbHtW1RhLm

C

Program errors

Table C.1 Errors returned by the settlement program, numbered from 6000 in this order.

Error	Raised when
`Unauthorized`	A control instruction is signed by the wrong key, the key in the signature instruction is not the market's quote authority, or a pool's founding deposit is not made by the treasury
`Paused`	The protocol or the market is paused
`MustBePaused`	A configuration change was attempted on a live market
`AccountMismatch`	An account supplied is not the one the quote or the market names
`AccountAlias`	Two of the instruction's accounts are the same
`QuoteEncoding`	The message is not 509 bytes or lacks the version tag
`SignatureLayout`	The signature instruction is missing, misplaced or has unexpected offsets
`TransactionTemplate`	The transaction contains anything outside the permitted shape, or the program was invoked by another program
`DomainMismatch`	Deployment, program, protocol, market or maker in the quote does not match
`EpochMismatch`	One of the four epochs has advanced since signing
`QuoteExpired`	Outside the slot or clock window, or the window is longer than the market allows
`MintConfiguration`	The token mint is paused, has an unsupported extension, or its fingerprint changed
`TokenConfiguration`	A token account is frozen, delegated, closable, or carries an unsupported extension
`NormalizationChanged`	The mint's active multiplier differs from the one registered
`ReferenceUnavailable`	No on-chain reference, or it is stale by slot or by clock
`PriceBand`	The price net of fee lies outside the band around the reference
`Amount`	A zero amount, a fee not exactly equal to the configured rate, or arithmetic overflow
`RiskLimit`	Per-trade size, vault floor or turnover in the current window exceeded
`BalanceDelta`	A token balance did not change by exactly the authorised amount
`ParticipantPolicy`	The market requires a permission the taker lacks, or it has expired

Error	Raised when
`Configuration`	Limits outside the program's hard bounds, or non-distinct authorities
`WithdrawalDelay`	Retained in the numbering; it guarded a treasury withdrawal that the deployed program disables
`LiquiditySharesRequired`	A fill on a pool with no shares outstanding, a redemption of more shares than the position holds, or a treasury withdrawal
`LiquiditySlippage`	A deposit or redemption would issue fewer shares or return less than the signed minimums
`LiquidityExpired`	A deposit or redemption past its deadline, or with a deadline more than 300 seconds away
`OracleConfiguration`	The market's oracle account is missing, not canonical, or holds an invalid policy
`OracleMode`	An operator-published reference on a PYTH_ONLY market, a verified publication on a market not in that mode, or an attempt to change the mode again
`OraclePayload`	The signed Pyth report is not exactly one feed with the six required properties in the expected format
`OraclePolicy`	The report's feed, channel, session, exponent, publisher count, confidence or source age fails the market's policy, at publication or at a fill
`OracleProvenance`	The source time is not newer, the sequence or validity is wrong, or the reference does not match the provenance recorded in the oracle account

D

Formats

D.1 The signature instruction

Table D.1 Data of the Ed25519 verification instruction, 621 bytes.

Offset	Bytes	Content
0	1	number of signatures, must be 1
1	1	padding, must be 0
2	14	seven little-endian u16 values: 48, 65535, 16, 65535, 112, 509, 65535
16	32	public key: must equal the market's quote authority
48	64	signature
112	509	the quote of Table 10.1

D.2 The engine contract

The pricing engine accepts JSON with amounts as decimal strings of at most twenty digits, without sign, point or exponent, and rejects unknown fields. A firm request carries the side, input atoms, multiplier in ppm, token decimals, fee in basis points, and the pool: reference in micro-dollars, inventory, target, the four parameters in ppm, and a pricing policy. The live pools use the policy `PYTH_PRIMARY`, under which skew moves the centre of both prices, a required spread that exceeds the cap is refused rather than clipped, and the multiplier must be exactly 1,000,000 ppm because the reference is already per raw token; the confidence cost is passed inside the session parameter. The response carries output atoms, fee atoms, the execution price in micro-dollars, and the five spread figures. An indicative request carries up to 128 pools and 8 sizes and returns bid and ask, in micro-dollars and in ppm from the reference, for each. Failures are `INVALID_INPUT`, `ARITHMETIC_OVERFLOW` and `AMOUNT_TOO_SMALL`. There is no partial result.

D.3 The fill event

`FillV1`: version, quote id, quote hash, market, trader, input mint, output mint, amount in, amount out, fee, the four epochs, and the receipt address. It is the only event the program emits.

D.4 The signed Pyth report

Table D.2 The Pyth Pro report in Solana format that publish_pyth_reference accepts, 158 bytes. Integers are little-endian. The six properties may appear in any order, each exactly once.

Offset	Bytes	Content
0	4	format tag of Pyth's Solana encoding
4	64	Ed25519 signature, checked by the native instruction
68	32	signer's public key, which the Pyth verifier must trust and find unexpired
100	2	length of the signed payload, 56
102	4	payload tag
106	8	envelope time, microseconds
114	1	channel, 3 for the fixed 200 ms rate
115	1	number of feeds, must be 1
116	4	feed identifier, as in Table B.2
120	1	number of properties, must be 6
121	37	price (i64), publisher count (u16), exponent (i16), confidence (i64), market session (i16) and source time (a presence byte, then u64 microseconds), each after a one-byte tag

E

References

1. Avellaneda, M. and Stoikov, S. (2008). High-frequency trading in a limit order book. *Quantitative Finance* 8(3), 217–224.
2. Glosten, L. and Milgrom, P. (1985). Bid, ask and transaction prices in a specialist market with heterogeneously informed traders. *Journal of Financial Economics* 14(1), 71–100.
3. Kyle, A. (1985). Continuous auctions and insider trading. *Econometrica* 53(6), 1315–1335.
4. Milionis, J., Moallemi, C., Roughgarden, T. and Zhang, A. (2022). Automated market making and loss-versus-rebalancing. *arXiv:2208.06046*.
5. Adams, H., Zinsmeister, N., Salem, M., Keefer, R. and Robinson, D. (2021). Uniswap v3 Core.
6. Angeris, G. and Chitra, T. (2020). Improved price oracles: constant function market makers. *Proceedings of AFT '20*.
7. Solana Foundation. Token-2022 program and token extensions; Scaled UI Amount integration guide; Ed25519 program.
8. Backed Finance. xStocks product and legal overview; developer documentation on multipliers.
9. Pyth Network. Pyth Pro documentation: subscribing to prices, payload reference, understanding price data, market hours, and integration as a consumer on SVM.
10. Bernstein, D. J., Duif, N., Lange, T., Schwabe, P. and Yang, B.-Y. (2012). High-speed high-security signatures. *Journal of Cryptographic Engineering* 2(2), 77–89.