Abhishek R-P - Algo trading strategy tips

Tell me this, I'm an individual trader, I like to do algo trading and I want to do it in such a way that I don't lose from HFTs but I don't have a big budget like an HFT and I want to do scalping on bank nifty options so I should be as quick as possible.

What all can I do? Suggest the best possible strategy for me. How can I have the fastest of the algo trading? possible. I am willing to spend some money but possibly not like an HFT. Think really really hard about this and then come up with a solution. You can even give an example with a new strategy or something like that.

Alright—here's a plan that lets you scalp Bank Nifty options **fast**, without pretending you're an HFT. The trick is to (1) cut your latency to the bone where it matters and (2) trade edges that **survive 100–500 ms**, not 5–10 ms.

1) Win the right race (don't fight sub-10 ms HFTs)

- **Time horizon:** Target moves that persist for ~0.3–3 seconds (aftershocks, liquidity vacuums, IV overshoots). Those are outside pure HFT turf but still "scalpy".
- **Lead market:** Use **Bank Nifty futures** order flow to *signal*, but **execute in options** (near-ATM/next strike). Futures lead; options lag just enough for you to catch a few points.
- **Microstructure filters:** Trade only during *tight spreads* and *adequate depth*; cancel quickly in widening-spread/vanishing-depth regimes.
- **Costs-aware sizing:** Your edge must clear brokerage + exchange fees + STT on option sells. Build this into backtests and live checks.

2) Make your stack genuinely fast (retail-doable)

Network & host

- **Run in Mumbai** (AWS ap-south-1 or a Mumbai VPS). Don't run from Jaipur or a far cloud region.

- **Separate processes & sockets:** One persistent **market-data** WebSocket, one **orders** session. Pre-auth and keep-alive.
- **Warm path:** Prebuild order payloads; avoid JSON bloat; reuse connections; batch cancels if broker allows.

OS / runtime

- Use **Linux** on the VPS. Pin your process to a CPU core; disable CPU freq scaling; `TCP_NODELAY` on sockets.
- Language: **Go/Rust** for ultra-low jitter, or **Python** with `asyncio`/uvloop if your broker is the bottleneck (often true). Keep GC pauses out of the hot path.

Broker/API choices (principles)

- You want: **tick-by-tick futures feed** (not 200–500 ms aggregates), **low order throttle**, and **fast acks.** If your broker can't do that, you can't scalp—full stop.
- Keep **rate limits** in mind; use **server-side brackets** if offered; otherwise implement a **local OCO** (see below).

Latency budget (realistic)

- VPS→Broker: ~3–10 ms; Broker→Exchange: broker-internal; Broker→VPS ack: total **~20–80 ms** best case.
- Your code must add **<5–10 ms** from signal to order submit.

3) A strategy you can actually run: Futures-led "micro-burst" scalper

Idea: Catch 0.3–2.0s *aftershocks* when futures are swept and liquidity doesn't refill instantly. Execute in nearest ATM option (CE or PE) on the same side.

Signals (event-time, not clock-time)

1. **Sweep + imbalance:** Within ⩽100 ms, futures best ask (for longs) is hit multiple levels *and* top-of-book depth drops > X% vs 1s median; **order book imbalance** > 0.65.
2. **Spread regime:** Futures spread ⩽ 1 tick before the sweep; post-sweep spread stays tight (no immediate mean-revert tell).
3. **Microprice push:** Microprice crosses prior microstructure pivot (e.g., VWAP of last 200 ms trades + skew by imbalance).
4. **Options confirm:** Your target option's spread is ⩽ 2 ticks and depth ⩾ Y lots at best.

Entry

- Map futures move to the option via **delta**: choose ATM/±1 strike with **highest depth**.
- Place **aggressive limit**: best ask for buys (lift), but price-improve by 1 tick if fill probability is high (backtested). Use **IOC** if your broker supports; otherwise very short **TIF**.

Exit

- **Time stop:** 800–1500 ms. If it doesn't go, scratch quickly.
- **Target:** 0.05–0.20% of option price (e.g., 4–12 pts on liquid weeklies); trail by last trade mid if burst continues.
- **Hard stop:** Behind the pre-sweep micro pivot or fixed ticks (must be inside your time stop risk).

Risk filters

- **No-trade windows:** Right at open first 1–2 min (if fills are chaotic) and immediately around macro spikes you can't price.
- **Spread guard:** If option spread widens beyond your threshold mid-trade → immediate exit.
- **Inventory:** Cap concurrent positions (1–2). You're not a market maker.

4) Second edge: IV-shock mean reversion (very short hold)

Setup: Detect **IV spike** in the ATM option (rapid rise in mid minus model IV from futures) **without** proportional futures follow-through.

- **Signal:** Δ(IV) over 200–400 ms > Z σ, futures microprice flat or reverting.
- **Trade:** Fade the IV pop (sell that option; hedge tiny delta with a 1-lot futures micro position if your broker allows quick futures hedges), **cover in ⩽2s** or when IV normalizes.
- **Note:** This requires extremely strict risk controls because you're short option—size tiny, auto-exit if futures starts trending.

5) Execution playbook (where most edges die)

- **Local OCO engine:** If broker doesn't support bracket orders with ms-fast modification, maintain a process that:
  - Sends entry,
  - Immediately posts **client-side stop & target**, and

    - Cancels siblings on first fill/exit event with **atomicity** (sequence numbers).
- **Queue awareness:** If your limit sits ⩾ 150 ms without queue advancement, cancel/replace one tick through or scratch.
- **Partial fills:** If partial < 40% after 250 ms, scratch—don't marry the order.
- **Kill switch:** If no ack within X ms or 3 rejects in 1s, **halt strategy**.

6) Build/measure like a pro (with your 6-month data)

- **Event-driven backtest** on **futures ticks + top-of-book** joined to option trades/quotes. Simulate:
    - Queue position & fill probability,
    - Real fees (brokerage, exchange, GST, stamp, STT on sells),
    - Slippage curves by spread regime and time-of-day.
- **Live shadow mode:** First run **paper/1-share equivalence** capturing: signal-to-send latency, send-to-ack, ack-to-fill, fill % by price improvement. Only go real once these are stable.
- **Daily limits:** Max loss (₹), max trades, max consecutive losers. Auto-stop.

7) Minimal tech blueprint (actionable)

- **Infra:** 1× Mumbai VPS (2 vCPU/4–8 GB), Linux; dedicated user; NTP sync.
- **Runtime:** Go/Rust or Python `asyncio` + uvloop; two processes (market, orders) + in-memory ring buffer; structured logging.
- **Math:** Precompute greeks for a grid of strikes; update with futures changes; keep a tiny, fast Black-76/BSM for ATM strikes only.
- **Storage:** Append-only binary logs per session for replay; rotate daily.
- **Monitoring:** Latency histograms (p50/p90/p99), reject codes, fill ratios, P&L net of fees, drawdown; pager on anomalies.

8) Expectations & position sizing

- Target **tiny, repeatable edges**: +2 to +10 pts per trade, 20–80 trades/day, 45–60% win rate, **tight losers**. Net edge comes from *fast scratches* and **not trading bad regimes**.
- Size **after** you prove net-of-fees expectancy. It's better to run small and scale frequency than to size up and eat slippage.

If you want, I can help you:

1. turn this into a **checklist + code skeleton** (signal engine, order router, OCO), and
2. design a **latency benchmark harness** tailored to your broker so we know exactly where the ms are going.

You want: tick-by-tick futures feed (not 200–500 ms aggregates), low order throttle, and fast acks. If your broker can't do that, you can't scalp—full stop. -- Can you check if upstox tick by tick data (https://upstox.com/trading-api/, https://upstox.com/tick-by-tick-trading/)is good enough in this? And also I would like to capture the tick data from Dhan tick by tick and place order on Shoonya Finvasis or flattrade.

Short answer: **Upstox is the closest "retail-fast" stack if you want one broker for both ticks + orders**, but you still need to measure their real-time feed quality yourself. **Dhan is fine for capturing real-time feeds**, and **Shoonya/Flattrade are solid for low-latency execution**, but using one broker's data to trade on another can run into exchange/broker T&Cs—so tread carefully (details below).

Is Upstox "tick-by-tick" and fast enough?

- **They market TBT (tick-by-tick) to retail** and say it's **free**, with infra colocated and much higher bandwidth than snapshot feeds. Good sign. Upstox - Online Stock and Share Trading
- **API docs** show a **WebSocket market feed** with **D5 depth** and other fields; the v3 docs note an initial **snapshot** on connection (normal). Upstox - Online Stock and Share Trading +1
- **Order speed & limits:** Upstox advertises a **dedicated low-latency order endpoint** with **<45 ms** execution (their claim) and **rate limits of 50 order requests/sec** (500/min). That's plenty for a sub-second scalper if your code is tight and hosted in Mumbai. Upstox - Online Stock and Share Trading +1
- **Reality check:** Dev-forum posts show some users observing **~1–1.5 s WS latency** at times. Treat this as anecdotal but it underscores why you must benchmark your own path. Upstox Community

- **Access nuance:** In late-2024, TBT via API was still "forwarded to the team"; recent threads (Jul 2025) discuss **TBT over WebSocket for "Plus"** users. Net: **TBT looks available but may be gated by plan/eligibility.** Confirm with Upstox support and test. Upstox Community +1

My take: If you can get **TBT over WS** + the **dedicated order endpoint**, Upstox alone can meet your "fast retail scalper" needs—just colocate your bot in **Mumbai** and measure p50/p99 latencies.

Dhan (data) → Shoonya / Flattrade (execution)

Technically feasible? Yes.
Compliance-wise? Risky without explicit permission.

- **Dhan's feed:** Real-time **WebSocket market feed** (v2) and even **20-level depth WS**; data APIs require a **paid subscription** (₹499/mo). DhanHQ +1 Dhan
- **Shoonya (Finvasia): Noren OMS** with REST+WebSocket; widely used for low-latency retail; official docs + Python/.NET SDKs. shoonya.com +1 GitHub
- **Flattrade: pi API** with WS market feed + Python SDK; free but **rate limiting** may apply. pi.flattrade.in +1 GitHub
- **But: NSE market-data policy prohibits redistribution / unlicensed use**. Using **Broker-A's data** to power trading at **Broker-B** can fall foul of agreements unless you have explicit rights. At minimum, it's a gray area—get written OK or use a licensed vendor feed. NSE India NSE India Search Archives

Safer alternatives for cross-broker setups

- Use a **licensed data vendor** (e.g., NSE-approved) for real-time ticks and route orders wherever you want. Or…
- Keep **data + orders with the same broker** (simplest compliance posture), and build **hot-spare** execution adapters for Shoonya/Flattrade you can switch to if needed.

What I'd do in your shoes (pragmatic plan)

Option A (clean & fast, one broker): Upstox-only

1. Get **TBT WebSocket access** (confirm plan), and use the **dedicated order endpoint.** Upstox - Online Stock and Share Trading Upstox Community

2. Host in **ap-south-1 / Mumbai**; pin CPU; `TCP_NODELAY`; prebuild orders.
3. Run a **latency harness**: measure **tick→decision**, **send→ack**, **ack→fill** for BANKNIFTY futures + ATM options across the session; alert if WS p99 > 150 ms.
4. If your p99 stays sub-150 ms and acks are snappy, your **aftershock scalper** and **IV-shock fade** will work.

Option B (data + orders split, but compliant): Vendor data + Shoonya or Flattrade

1. Subscribe to an **NSE-approved tick feed** (not broker-bound).
2. Execute via **Shoonya** (Noren OMS) or **Flattrade pi**; measure order acks and rejects; respect their per-second limits. shoonya.com pi.flattrade.in

---

Go/No-Go checklist (run this 1 day before you commit money)

1. **Market-data quality**
   - Upstox/Dhan WS **tick cadence**: sustained ticks during active periods (09:20–10:00, 14:45–15:25) with **no >250 ms gaps** except when market is truly idle.
   - **Depth freshness**: D5 or 20-depth updates land within **⩽100 ms** of trades during bursts. Upstox - Online Stock and Share Trading DhanHQ
2. **Order channel**
   - **Upstox:** verify **<45 ms** claim from *send→exchange-ack* on small IOC limits (record p50/p90/p99). **Rate-limit headroom** (50/s) must cover entry + stop/target + cancels. Upstox - Online Stock and Share Trading +1
   - **Shoonya/Flattrade:** record **send→ack** and **reject codes** under stress (10–20 ops/s bursts). shoonya.com pi.flattrade.in
3. **Compliance**
   - If you still want **Dhan-data → Shoonya/Flattrade-orders**, get **written confirmation** or switch to a licensed data feed. **NSE policy** is strict about redistribution/usage. NSE India

---

A concrete micro-scalp you can try (with Upstox or vendor+Shoonya)

- **Signal (event-time):** BANKNIFTY FUT best-ask gets swept ⩾3 levels in ⩽120 ms, **OB imbalance > 0.65**, spread stays tight; enter **ATM CE** with an **aggressive limit IOC**.

- **Exit: Time stop 900 ms**, **target** +6 to +10 pts, **hard stop** 4–6 pts or spread-widen event.
- **Throughput math:** With Upstox **50 req/s**, you can do: entry + (target & stop) + up to 3 cancels/replace safely per trade, even with 1–2 concurrent positions. Upstox - Online Stock and Share Trading

---

Bottom line

- **Best single-broker bet:** Upstox (if you get TBT WS + dedicated order endpoint). Measure and proceed. Upstox - Online Stock and Share Trading +1
- **Split-stack:** Dhan feed + Shoonya/Flattrade orders is **technically fine but check T&Cs**; NSE generally forbids using one party's live data beyond the licensed scope. Prefer **licensed vendor data** if you want that architecture. DhanHQ +1 NSE India

If you want, I can give you a **minimal latency harness** (CLI script) that timestamps **tick→order→ack** for whichever combo you choose—then we'll tune until your **p99** is where it needs to be.

Sources

Okay, so yeah, don't worry about those permissions, I have them. And I want to do like upstocks plus flat trade or upstocks plus Shunye because otherwise the brokerage will kill it as I have a small trading account and I'll be placing like small lots like 1, 2, 3, 4, 5 lots kind of a thing. So with this combination gave me the fastest approach and how fast can I assume it to be? From the tick emitted from NSE server to reaching me to me calculating on the basis of my formula. Let's assume the formula is like above EMA5 and above EMA20 on a 10 second horizon. Then placing that order. Yes, this cycle, size of time of this cycle.

Got it—let's assume you're greenlit on permissions and you want **Upstox (TBT data) → compute → Shoonya/Flattrade (execution)** for minimal brokerage. Here's the **fastest realistic path** and the **latency you can plan for**.

Fastest retail-grade setup (pragmatic)

- **Host**: VPS in **Mumbai** (ap-south-1 or local DC). Pin CPU, `TCP_NODELAY`, NTP sync.
- **Data**: **Upstox TBT WebSocket** (depth+ticks). Their stack markets retail TBT and persistent WS without WS rate limits. Upstox - Online Stock and Share Trading Upstox Community
- **Orders**:
  - **Shoonya (Finvasia)** via **Noren OMS** (low-latency focus). shoonya.com
  - or **Flattrade pi** (WS + REST, **40 req/s** limit). pi.flattrade.in
- **Upstox order alt**: Upstox orders allow **50 req/s**—useful as a benchmark/comparison. Upstox - Online Stock and Share Trading
- **Process split**: one WS process (ticks), one order-router process (Shoonya/Flattrade). Pre-build order payloads; IOC where possible.

Tick→Decision→Order: realistic latency budget

Below are **best / typical / p99** ranges if you're in Mumbai and everything's healthy.

A) Tick arrival (NSE → Upstox → you)

- **Exchange→Upstox internal**: sub-ms to a few ms (infrastructure/colo domain).
- **Upstox → your VPS over WS (Mumbai)**: **~10–30 ms typical**, **~50–120 ms p99** depending on network jitter and broker dispatch. (Upstox themselves publish intra-Mumbai network latencies in this ballpark; you must measure your feed.)
  Upstox - Online Stock and Share Trading
- **App overhead (parse + EMA update)**: **<0.1 ms** if you maintain rolling EMA state.

B) Decision (your EMA rule)

- For **EMA5/EMA20 on a 10-second horizon**, compute is negligible. The only "latency" is whether you wait for bar-close; if you update EMAs tick-by-tick (recommended), decision latency is **~0.1–1 ms**.

C) Order path (your VPS → broker OMS → exchange)

- **To Shoonya/Flattrade gateway (Mumbai): ~3–10 ms typical**.
- **Broker OMS + risk + to-exchange + ack back**: **~20–80 ms typical**, **~120–200 ms p99** at busy moments (rejects, throttling, OMS queue). (Shoonya/Noren marketed as low-latency; Flattrade publishes rate limits, not lat—so measure.)
  shoonya.com pi.flattrade.in

Put together (Tick→Order SENT)

- **Best**: 15–25 ms
- **Typical: 30–70 ms**
- **p99**: 100–220 ms

Tick→Exchange ACK back to you

- **Best**: 35–60 ms
- **Typical: 60–140 ms**
- **p99**: 150–300 ms

> These are **retail-fast** numbers—good enough for sub-second scalps that survive ~0.2–1.0 s. They won't beat true colo HFT, but they're fast enough to monetize **aftershocks** and **IV pops**.

Why your 10-second EMA rule is fine (and how to fire sooner)

- Maintain **incremental EMAs** on each tick; don't wait for the bar to close.
- Trigger when **both EMAs are crossed on the rolling 10-sec window**, optionally require the futures **microprice** to confirm (tight spread + imbalance direction).
- This keeps compute latency $\ll$ 1 ms; your edge depends on **feed jitter + order ack**, not math.

What to measure on day 1

1. **WS feed jitter** from Upstox: inter-tick gaps, burst handling (09:20–10:00, 14:45–15:25). You want **few gaps >150 ms**. Upstox TBT for retail exists; validate freshness on your line. Upstox - Online Stock and Share Trading
2. **Order RTT** with Shoonya/Flattrade: send→ack histograms (p50/p90/p99), and reject codes under **10–20 ops/s** bursts. (Flattrade limit: **40 req/s**; Upstox: **50 req/s** for comparison.) pi.flattrade.in Upstox - Online Stock and Share Trading
3. **End-to-end** (tick ts vs ack ts): log monotonic timestamps at **tick receive**, **decision**, **send**, **ack**.

Concrete wiring tips (that actually shave ms)

- **Persistent WS** (no reconnect churn), small GC heaps, pinned CPU core.
- **Binary/compact JSON** parsing on the hot path; pre-allocated objects.
- **IOC or very short TIF**; if not filled in ~150–250 ms, **scratch/price-improve by 1 tick**.
- **Client-side OCO** if broker brackets are slow (entry → instant post target+stop; cancel siblings on fill).
- **Mumbai VPS** with low jitter uplink (avoid noisy neighbors, avoid NAT if possible).

Sanity check against the market

- NSE keeps expanding **colo capacity**—institutions will always be faster at the nanosecond layer. Your goal is to catch **moves that persist 0.3–2.0 s**, where retail-fast still wins. Reuters

---

TL;DR numbers to plan around (Upstox TBT data → Shoonya/Flattrade orders)

- **Tick→Order SENT**: **30–70 ms typical** (best ~20 ms; p99 up to ~220 ms).
- **Tick→ACK back**: **60–140 ms typical** (best ~40–60 ms; p99 up to ~300 ms).
- **Compute (EMA5/20, 10s)**: ≪ 1 ms—irrelevant.

If you want, I'll give you a tiny **latency harness** template next (logs **tick_ts, decide_ts, send_ts, ack_ts**), so you can verify these numbers on your exact lines and choose between Shoonya vs Flattrade empirically.

Sources

Okay, what are some possibly good strategies for this bank nifty options scalping in this particular case considering these numbers and everything? What should I explore on my backtesting? I have like kind of 3-4 years of data. It's 1 second snapshot data.

Great—you've got ~3–4 years of **1-second snapshots**. That's enough to design scalps that survive ~1–5 seconds. Below are **strategy archetypes that backtest well on 1s data**, with concrete, testable rules and how to bridge to live TBT execution later.

Ground rules for 1-second data

- **Trade horizon:** 1–15 seconds. Signals must be defined on **consecutive 1s bars** (no tick-only logic).
- **Lead market:** Use **Bank Nifty Futures** for the signal, **execute in ATM/±1 strike options** (map via delta).
- **Execution model in backtest:** Entry price = option **mid ± half-spread haircut**; exit = mid ± half-spread + **slippage add-on** during volatile seconds. Sensitivity-test spreads by time-of-day.

- **Fees:** Bake in brokerage + taxes + STT (especially for shorts). Your edge must clear these.
- **Walk-forward:** 6–12 month rolling windows; **anchored** out-of-sample evaluation.

Strategy set (purpose-built for 1-second bars)

A) Burst continuation (1–5s “aftershock”)

Hypothesis: When futures prints a short **range expansion** burst, the move persists for a couple seconds more.

Signal (on futures 1s bars):

- Two consecutive 1s returns same direction, each > **$k \cdot \sigma_1$s** (σ over last 120–300s).
- Acceleration: $|r(t)| > |r(t-1)|$ and **range(t) > p-th percentile** of last 30 min.
- Optional: 1s RSI(5) > 70 (long) / < 30 (short) as regime tag.

Entry: Buy **ATM CE** if up-burst (or next OTM if spread tighter); for down-burst buy **ATM PE**.
Exit: Time stop 2–5s, **target** +6–12 pts (scalable), **fail-fast stop** 3–6 pts or opposite 1s close.
Filters: Skip if option 1s spread > threshold (e.g., >0.25% of price) or lunchtime lull (12:15–13:15).
Tune: $k \in [1.0, 1.8]$; target/stop ratio ~1.2–1.5; hold 1–7s.

B) Micro pullback → re-impulse (the “BPB”)

Hypothesis: After a strong 1s impulse, a **single 1s counter bar** that **doesn’t** take back half the move is a springboard.

Signal: $r(t-1) > k \cdot \sigma$; r(t) opposite sign but $|r(t)| <$ **$0.5 \cdot |r(t-1)|$**; then **entry** on the first tick of r(t+1) continuing original direction (use bar open proxy).
Entry/Exit: Same as (A) but smaller targets (4–8 pts).
Filters: Only when **spread percentiles** are in the tighter half of day; skip first 60–90s of market.

C) ORB micro-scalps (opening range breakout, 1s)

Hypothesis: Breaks of a **3–5 min OR** with immediate 1s follow-through pay quickly.

Setup: Define **OR high/low** from 09:15–09:18 (or 09:20).
Signal: First 1s close that leaves OR by > one **min-tick and** follow-up 1s bar in same direction.
Entry: Buy ATM in breakout direction.
Exit: Target 8–15 pts or **time stop** 5–8s; scratch if back inside OR.
Filters: Avoid day type: gap-and-fade days—add a regime tag using pre-open gap size.

D) VWAP slapback (very short mean reversion)

Hypothesis: 1s deviations > $z \cdot \sigma$ from **rolling VWAP** (3–10 min) snap back partially within ~5–15s—works best mid-day.

Signal: $|\text{Price} - \text{VWAP}| / \sigma_{1}s > z$ and the next 1s bar **stalls** (smaller body than previous).
Trade: Fade via **ATM option** (CE if below VWAP and stalling down-move; PE if above).
Exit: To VWAP or **time stop** 10–15s; stop beyond last 1s extreme.
Tune: $z \in [1.0, 2.0]$; works only when spreads are tight—add spread filter.

E) IV lag/overshoot pairs (price–IV dislocation)

Hypothesis: Options' **implied vol** briefly **overshoots** or **lags** the futures move on 1s scale.

Compute: Approx **Black-76** IV for **ATM option** each second using futures price, time-to-expiry, $r \approx 0$.
Signals:

- **Lag long:** Futures $+1\sigma$ burst, option price up but **IV flat/down** over same 1–2s → **buy** the option; target small (4–8 pts).
- **Overshoot fade (tiny size):** Futures flat, option IV jumps > $z\sigma$ → **sell** that option, **strict 3–6s cover** or if futures starts moving.
  Notes: For shorts, size tiny and stress-test borrow/STT effects; many will prefer **buy-only** version.

F) Strike magnet micro-gamma (expiry day)

Hypothesis: On weekly expiry, **ATM strikes** act like magnets; quick fades/continuations around the strike print.

Signal: Futures oscillates inside **±0.1–0.2%** of strike; when a 1s break away fails and returns inside band within 2s → fade to the strike (buy opposite option).
Exit: Hit +5–10 pts or **time stop** 5–10s.
Filters: Only 13:30–15:00; exclude news spikes.

G) Liquidity vacuum proxy (no true depth? emulate)

Hypothesis: When a 1s bar has **large range** but **low volume** (vs 5-min baseline), next 1–3s often continue in same direction (thin book getting swept).

Proxy: `range(t) > p95(range_30m)` **and** `vol(t) < p30(vol_30m)`.
Trade: Continue in direction with small target (4–8 pts), **tight time stop**.

What to measure & tune in backtests

Common parameters to grid-search

- `k` (σ multipliers): 1.0–2.0
- Targets: 4, 6, 8, 10, 12, 15 pts
- Time stops: 2, 3, 5, 8, 12, 15 s
- Spread cutoff: 70th, 80th, 90th percentile of day’s spread (if you have bid/ask; else use option range/price proxy)
- Day/Time filters: exclude 09:15–09:16; lunch; include last 30–45 min as separate regime

Mapping futures → option

- Use **ATM** or **±1 strike** with **highest 1s traded volume** and **lowest spread**.
- If you compute greeks, pick strike with **max gamma/vega per unit spread**.
- For each entry, record chosen strike, spread, and assumed fill to estimate **fill probability**.

Slippage model (critical with 1s data)

- Base assumption: **entry at mid + 0.5×spread**, exit at **mid + 0.5×spread**; **add 1–2 ticks** extra slippage when your signal coincides with p90 1s range bars.
- Run a **stress test**: double spreads and add +1 tick slippage to see if edge survives.

Walk-forward protocol

- Split years into rolling windows: e.g., **train 6 months → test next 3 months**, slide by 1 month.

- Track **expectancy per trade**, **trade frequency**, **hit rate**, **max consecutive losses**, **net after fees**.

Risk & trade management to simulate

- **Client-side OCO**: on entry, you immediately have target+stop; cancel sibling on first fill.
- **Scratch rule**: if price returns to entry within **2s**, exit flat (this saves the day).
- **Daily circuit breakers**: max trades, max loss, stop after N consecutive losers.

Bridging 1s research → live TBT

- Keep **the same rules**, but on live TBT you'll trigger **earlier within the 1s bar** → slightly better fills.
- Your live latency budget (from earlier): **tick→order sent 30–70 ms typical**; design **time stops** and **targets** that still work if entry is late by ~100–200 ms.

Suggested starting short-list (ranked)

1. **Burst continuation (A)** with conservative targets (6–10 pts) + 3–5s time stop.
2. **BPB re-impulse (B)** for a second, smaller edge on the same days.
3. **IV lag long (E, buy-only)** once your IV calc is stable.
4. **ORB micro-scalps (C)** restricted to clean-trend days.

Prove each **net of fees** with walk-forward, keep the top two, and **ensemble** them: take a trade only when **two signals agree** (e.g., A + E).

If you want, tell me exactly which fields your 1s snapshots have (futures LTP/volume? options LTP/bid/ask/volume?), and I'll turn one of these into a **precise, parameterized backtest spec** you can code right away.