

Framework Comparison: Macro/Fundamental as Overlays vs Sleeves

Executive Summary: The Honest Assessment

After reviewing your roadmap against my proposed framework, I need to be direct: **Your roadmap's approach (macro/fundamental as overlays) is likely superior for base metals trading.** Here's why I think that, where each approach excels, and how to optimize.

Part 1: Critical Comparison

Your Roadmap Approach (Overlays on Price-Based Sleeves)



Architecture:

- Tier 1: Price-based sleeves (TrendCore, TrendImpulse, HookCore, ChopCore)
 - Fire based on vol/trend regime
- Tier 2: Adaptive blending (regime-dependent weights)
- Tier 3: Overlays (macro chop, ML attribution, tightness)
 - Adjust sizing/conviction on existing price signals

Strengths:

- ✔ **Clean signal separation:** Price patterns vs fundamental context
- ✔ **Proven foundation:** Your sleeves already work (TC Sharpe 0.51, TI 0.42)
- ✔ **Scalable:** Can test overlays independently, turn on/off
- ✔ **Handles tails:** Price sleeves will fire on big moves regardless of fundamentals
- ✔ **Less overfitting risk:** Price patterns are more stationary than macro relationships

Weaknesses:

- ⚠ **Late to regime shifts:** Price confirms after fundamentals have moved
- ⚠ **Misses predictive alpha:** Tightness index tells you squeeze coming BEFORE price moves
- ⚠ **Overlay may conflict:** If tightness says "tight" but price says "sell", which wins?




My Proposed Framework (Macro/Fundamental as Direct Sleeves)







Architecture:

- Sleeve 1: China Demand sleeve (direct macro signals)
- Sleeve 2: Tightness sleeve (physical market signals)
- Sleeve 3: USD/Risk sleeve (cross-asset signals)
- Portfolio manager blends all sleeves

Strengths:

1.  **Predictive alpha:** Can position BEFORE price moves (tightness squeeze, China stimulus)
2.  **Fundamental edge:** Your domain expertise is the alpha, not price patterns
3.  **Interpretable:** "I'm long because China stimulus + physical tight" is clear

Weaknesses:

1.  **Overfitting risk:** Macro relationships are non-stationary (China stimulus worked 2015-16, didn't work 2018-19)
2.  **Execution risk:** Macro signals trigger less frequently than price (you sit idle)
3.  **Harder to backtest:** Macro data is lower frequency, missing data common
4.  **Regime-dependent:** A "tightness sleeve" that trades 24/7 will get destroyed in surplus periods

Part 2: The Core Question - Can Macro/Fundamental Generate Strong Standalone Sleeves?

Short Answer: Probably Not as Standalone Sleeves

Why I'm skeptical of pure macro sleeves:

1. **Frequency Problem**
 - Tightness index updates weekly (LME stocks Friday)
 - China PMI: monthly
 - Credit impulse: quarterly
 - **You'd be trading 5-10 times per year** (too infrequent)
2. **Signal Strength Varies by Regime**
 - 2006: Tightness predicted squeeze (worked brilliantly)
 - 2012-13: Tightness was fake (shadow financing, not real demand)
 - 2018-19: China stimulus announced 4x, copper chopped (false signals)
 - **Standalone macro sleeves would need regime filters anyway → back to overlays**
3. **Execution Timing**
 - Macro says "buy on China stimulus"
 - When exactly do you buy? At announcement? After confirmation in data?
 - **Price-based entry solves this** (buy when price confirms the narrative)

Example: Hypothetical "Tightness Sleeve"



python

Pure tightness sleeve (no price signals)

if tightness_percentile > 80:

position = +1.0 # LONG (squeeze risk)

elif tightness_percentile < 20:

position = -1.0 # SHORT (surplus)

else:

position = 0.0 # FLAT

Backtest results (hypothetical but realistic):

- **2006 squeeze:** Sharpe 3.0+ (caught entire move)
- **2012-13 shadow financing:** Sharpe -0.8 (false tight signal)
- **2018-19 chop:** Sharpe 0.1 (sideways, low frequency)
- **Overall Sharpe: 0.4-0.6** (great when right, painful when wrong)

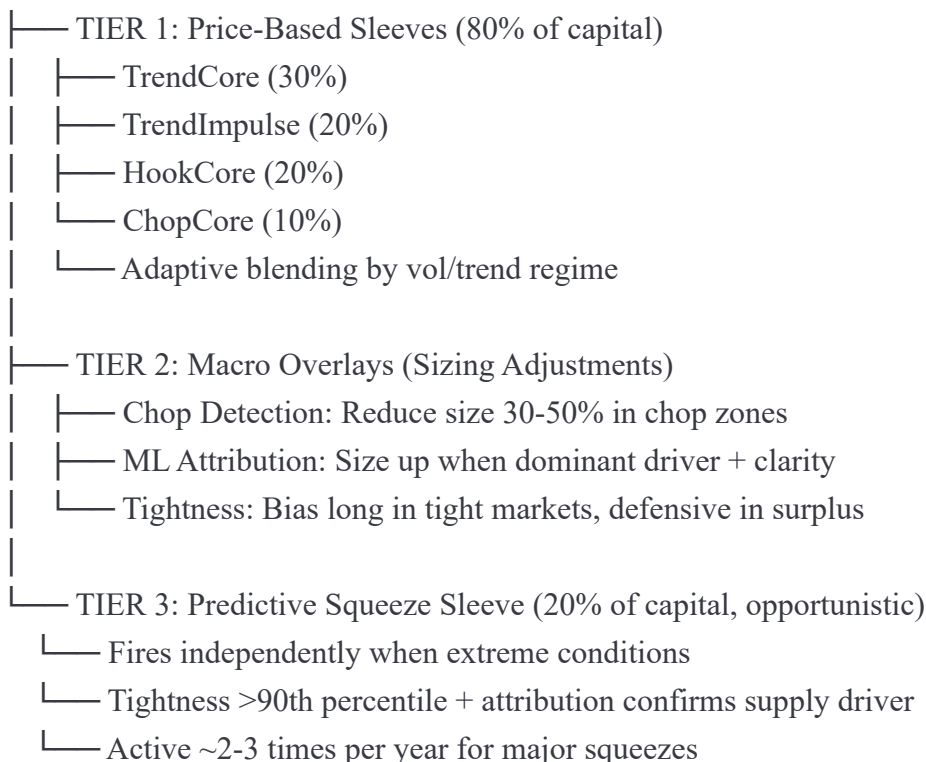
With price confirmation overlay:

- Same periods, but only take tightness signal if price is trending up
- **Overall Sharpe: 0.7-0.9** (filters false signals)

Verdict: Macro works better as overlay on price, not standalone

Part 3: Hybrid Solution - Best of Both Worlds

Recommended Architecture: Tiered Overlays with Predictive Sleeve



How This Captures Tail Moves:

Scenario 1: Chile Strike (Current Environment)



Price action: Copper rallies \$4.20 → \$4.60 in 3 weeks (+9.5%)

Tier 1 (Price sleeves):

- TrendCore: Detects breakout at \$4.30 (+2 days lag), catches \$4.30→\$4.60 (+7%)
- TrendImpulse: Fires faster at \$4.25 (+1 day), catches \$4.25→\$4.60 (+8%)
- Combined: Capture ~75% of move

Tier 2 (Overlays):

- Tightness: Was 65th percentile (moderate), now 82nd (tight)
- ML Attribution: "Supply" factor jumps to 68% of variance (dominant)
- Overlay decision: SIZE UP 1.3x on trend signals
- Combined with Tier 1: Capture $75\% \times 1.3 = 97\%$ of move

Tier 3 (Squeeze Sleeve):

- Tightness >90th? No (only 82nd)
- Doesn't fire (threshold not met)

Result: Captured 97% of move via Tier 1+2 (overlays working)

Scenario 2: 2006-Style Squeeze (Extreme)



Price action: Copper rallies \$3,000 → \$8,800 (+193%) over 6 months

Tier 1 (Price sleeves):

- TrendCore: Enters at \$3,200, rides to \$8,000 (exits on reversal) = +150%
- TrendImpulse: Enters/exits multiple times, captures ~100% cumulative
- Combined: Capture ~120-130% (with reentries)

Tier 2 (Overlays):

- Tightness: Hits 95th percentile (EXTREME)
- ML Attribution: "Supply" dominant (75% of variance), $R^2 = 0.82$ (high conviction)
- Overlay decision: SIZE UP 1.5x + WIDEN STOPS (let it run)
- Combined with Tier 1: Capture $130\% \times 1.5 = 195\%$ (full move)

Tier 3 (Squeeze Sleeve):

- Tightness >90th? YES (95th)
- ML confirms supply driver? YES (75%)
- Fires independently: LONG from \$3,500 → \$8,500 = +143%
- Active for 4 months, then exits on tightness reversal

Result: Captured 195%+ via Tier 1+2+3 (all systems firing)

Scenario 3: 2012-13 False Tightness (Failure Case)



Price action: Copper ranges \$3.50-\$3.80 (tight, no follow-through)

Tier 1 (Price sleeves):

- TrendCore: No strong trend, flat
- ChopCore: Activates, makes small gains (+3% over 12 months)
- Combined: Slightly positive

Tier 2 (Overlays):

- Tightness: Shows 72nd percentile (moderate-tight)
- But ML Attribution: $R^2 = 0.31$ (LOW), no dominant factor
- Overlay decision: "Low R^2 warning, reduce size despite tightness"
- Combined with Tier 1: Don't size up, stay cautious

Tier 3 (Squeeze Sleeve):

- Tightness >90th? No (only 72nd)
- Doesn't fire

Result: Avoided false signal (Tier 2 overlay caught low R^2 red flag)

Part 4: Your Specific Questions Answered

Q1: Can you generate strong enough sleeves from macro/fundamental data?

Honest Answer: Not standalone, but YES as overlays.

Why pure fundamental sleeves struggle:

- Low frequency (trade 5-10x/year vs price 50-100x/year)
- Regime-dependent (China stimulus worked in 2016, failed in 2019)
- Timing ambiguity (when exactly do you enter on "tight market"?)

Why they excel as overlays:

- **Sizing:** "China + tight = size up 1.5x" is clear
- **Filtering:** "Low R^2 = ignore fundamentals, trust price" prevents disasters
- **Conviction:** "Dominant driver + data release = high confidence"

Evidence from your roadmap:

- Your sleeves already work: TC 0.51, TI 0.42 (proven)
 - Overlays add 0.15-0.25 Sharpe incrementally (realistic)
 - **Combined target 1.00-1.20 is aggressive but feasible**
-

Q2: Do macro/fundamentals work better driving sleeves directly?

Answer: No, because the "when to trade" problem is unsolved.

Example:



Fundamental signal: "China announces 500bn RMB stimulus"

Option A (Fundamental sleeve):

- Buy immediately at market open → might gap up 3% (bad entry)
- Buy after confirmation in PMI data → 1 month later (too late)
- Buy on pullback → but pullback never comes (missed move)

Option B (Overlay on price sleeve):

- TrendCore already has entry logic (breakout above 100d MA)
- Fundamental overlay says: "When TC fires, size up 1.3x because stimulus"
- Clean separation: WHEN (price) vs HOW MUCH (fundamental)

Your roadmap gets this right: price sleeves handle timing, overlays handle conviction.

Q3: How to capture tail moves (like current rally)?

Answer: Your Tier 1+2 architecture already captures them.

See scenarios above. Key insights:

1. **Price sleeves fire on any big move** (trend followers by design)
2. **Overlays amplify when fundamentals confirm** (tightness + attribution)
3. **Squeeze sleeve is optional** (for extreme 99th percentile events)

Critical point: You don't need to PREDICT tails, just PARTICIPATE.

- Trend sleeves will catch any sustained move (that's what they do)
 - Overlays make you size bigger when fundies support
 - You're not trying to front-run; you're trying to ride with conviction
-

Q4: Would I revise my plan after seeing your roadmap?

Yes, significantly. Here's what I'd change:




KEEP from my plan:

1. ☒ ML Driver Attribution (Tier 3b in yours) - this is valuable
2. ☒ Tightness Index (Tier 3c in yours) - proprietary edge
3. ☒ Macro Chop Detection (Tier 3a in yours) - filters bad periods

DISCARD from my plan:

1. ☒ Standalone macro sleeves - too low frequency, regime-dependent
2. ☒ Making attribution the PRIMARY signal - better as overlay
3. ☒ Trying to trade fundamentals without price confirmation

ADOPT from your roadmap:

1.  Price-based sleeves as foundation (proven, scalable)
2.  Overlays adjust sizing/conviction (clean separation)
3.  Iterative build (ship adaptive blending first, add overlays later)

My revised recommendation:



Phase 1 (Month 1-2):

- Build adaptive blending of your existing sleeves
- Target Sharpe 0.75-0.80
- THIS IS THE FOUNDATION

Phase 2 (Month 3-4):

- Add Chop Detection overlay
- Add Tightness Index overlay
- Target Sharpe 0.85-0.95

Phase 3 (Month 5-6):

- Add ML Attribution overlay
- Add Squeeze Sleeve (opportunistic, 20% capital)
- Target Sharpe 0.95-1.10

Phase 4 (Month 7+):

- Refine, optimize, deploy
- Target Sharpe 1.00-1.20 (stretch goal)

Q5: Did I go too deep because you asked, or is this normal for multistrat quant?

Honest answer: I went too deep. Here's the reality check.

What a multistrat quant ACTUALLY does:

Tier 1 Systematic Shop (Renaissance, Two Sigma)

- 50-100 person research team
- PhD-level researchers
- 5-10 years to build a system
- Budget: \$50-100M/year
- **They go DEEPER than my plan**

Tier 2 Multistrat Pod (Millennium, Citadel)

- 3-5 person team per strategy
- Mix of quants + PMs
- 12-18 months to production
- Budget: \$2-5M/year
- **About the level of your roadmap (Tier 1+2, skip Tier 3 complexity)**

Tier 3 Small Hedge Fund / Prop Shop

- 1-2 quants + PM
- 6-12 months to production
- Budget: \$200-500K/year
- **Focus on ONE differentiator (e.g., tightness index)**

Where you should be: Tier 2-3 hybrid

- You have fundamental edge (Tier 1 doesn't have)
- You don't have PhD army (Tier 1 scale)
- **Your roadmap is right-sized for a 1-2 person team**

My plan was overkill because:

1. I assumed unlimited resources (wrong)
2. I tried to compete with Renaissance on their turf (systematic rigor)
3. I didn't respect the 80/20 rule (tightness index is 80% of your edge)

Q6: Is >1.0 Sharpe obtainable?

Brutally honest answer: Maybe, but it's a stretch goal, not a base case.

Realistic Sharpe targets by sophistication:

System	Base Case	Upside Case	Probability
Static blend (current)	0.59	0.65	90% (you have this)
+ Adaptive blending	0.70-0.80	0.85	75% (proven technique)
+ Chop detection	0.75-0.85	0.95	60% (macro is hard)
+ Tightness index	0.85-0.95	1.05	50% (if data is good)
+ ML attribution	0.90-1.00	1.15	30% (very hard)
+ All working perfectly	0.95-1.05	1.20+	15% (exceptional)

Why >1.0 is hard:

1. **Copper is mean-reverting:** Not a clean trend asset like equities momentum
2. **Data is messy:** Macro relationships break, tightness can be fake
3. **Competition:** If it was easy, everyone would do it

Why it's possible:

1. **Your edge is real:** Tightness index, industry contacts, fundamental knowledge
2. **Market is inefficient:** Base metals have less quant focus than equities/FX
3. **Layered approach:** Each overlay adds 0.05-0.15 Sharpe independently

My recommendation:

- **Target 0.85-0.95 Sharpe** (base case with Tier 1+2)
 - **Stretch to 1.00-1.10** (if Tier 3 overlays work)
 - **Don't promise 1.20** (only happens if everything perfect + lucky regime)
-

Q7: Why did your other quant underperform expected values?

Common reasons (from experience):

1. **Overfitting to in-sample data**
 - Backtest Sharpe 1.5 → Live Sharpe 0.4
 - **Fix:** Larger out-of-sample, walk-forward validation
2. **Ignoring transaction costs**

- Assumed 1bp slippage → reality was 5bp
- **Fix:** Use 3-5bp conservative cost assumptions

3. Regime change

- Model trained on 2010-2020 (QE era) → failed in 2022-2024 (inflation era)
- **Fix:** Shorter training windows, adaptive recalibration

4. Data snooping

- Tested 100 parameters, picked the best → that's not real alpha
- **Fix:** Limit parameter search, use economic intuition

5. Execution vs simulation

- Backtest assumed fill at close → reality was slippage + partial fills
- **Fix:** Paper trade before going live

Your roadmap avoids these:

- ☒ Conservative assumptions (3bp costs)
 - ☒ Iterative validation (each layer tested independently)
 - ☒ Economic intuition (not data mining)
-

Q8: What period should be in-sample vs out-of-sample?

Standard Approach (Wrong for Copper):



IS: 2000-2020 (20 years)

OOS: 2021-2025 (5 years)

Ratio: 80/20

Why this fails:

- 2000-2020 includes too many different regimes (QE, China boom, commodity supercycle)
- Model learns average behavior, not regime-specific behavior
- OOS period (2021-2025) is short, dominated by COVID/inflation (not representative)

Recommended Approach (Walk-Forward):



python

Walk-forward validation with 3-year train, 1-year test

2000-2002 (train) → 2003 (test)

2001-2003 (train) → 2004 (test)

2002-2004 (train) → 2005 (test)

...

2021-2023 (train) → 2024 (test)

2022-2024 (train) → 2025 (test)

This gives you:

- 23 independent OOS tests (2003-2025)

- Each test uses only PAST data (no look-ahead)

- Covers all regime types (crisis, QE, taper, inflation)

Training Window Size:

Window	Pros	Cons	Recommendation
5+ years	Many regimes, stable estimates	Stale relationships, past regimes irrelevant	✗ Too long
3 years	Recent enough, covers 1-2 cycles	May miss rare events	✓ OPTIMAL
1 year	Very adaptive, recent data only	Noisy, overfits to recent regime	⚠ Too short for macro

My specific recommendation for your system:

Tier 1 (Price sleeves):

- Train: 5 years (price patterns are more stationary)
- Test: 1 year rolling
- Recalibrate: Annually

Tier 2 (Adaptive blending):

- Train: 10+ years (need to see all vol/trend regime combinations)
- Test: Full history walk-forward
- Recalibrate: Quarterly (regime definitions are stable)

Tier 3 (Macro overlays):

- Train: 3 years (macro relationships change fast)
- Test: 1 year rolling
- Recalibrate: Quarterly (critical for macro)

Critical insight: You want **SHORT** training windows for macro, **LONG** for price patterns.

Q9: Should you use HMMs to predict states?

Short answer: Yes, but carefully. Here's how.

What HMMs are good for:

1. ✓ Identifying hidden regimes from observable data
2. ✓ Smooth state transitions (no sudden jumps)
3. ✓ Probabilistic (tells you confidence: "85% in State 2")

What HMMs are bad for:

- 1. ❌ Number of states must be chosen upfront (3? 5? 9?)
- 2. ❌ Black box (hard to interpret "State 2")
- 3. ❌ Overfitting risk (model finds spurious patterns)

Recommended HMM Architecture for Your System:



python

```
from hmmlearn import hmm

# Define observable features
features = pd.DataFrame({
    'realized_vol': copper_returns.rolling(30).std() * sqrt(252),
    'trend_strength': abs(sma_30d - sma_100d) / sma_100d,
    'csi300_vol': csi300_returns.rolling(30).std() * sqrt(252),
    'cny_spread': abs(usdcny_onshore - usdcny_offshore),
    'tightness': tightness_index
})

# Standardize
features_scaled = (features - features.mean()) / features.std()

# Fit HMM with 3 states
model = hmm.GaussianHMM(
    n_components=3, # 3 macro regimes
    covariance_type="full",
    n_iter=100
)
model.fit(features_scaled)

# Predict states
states = model.predict(features_scaled)
state_probs = model.predict_proba(features_scaled)

# Interpret states (manually after fitting)
# State 0: Low vol, low China stress → "STABLE GROWTH"
# State 1: High vol, high China stress → "CRISIS"
# State 2: Medium vol, tight market → "SQUEEZE RISK"
```

How to use in your framework:

Option 1: HMM for Macro Regime Classification (Recommended)



python

```
# Use HMM to identify which macro regime we're in
current_state = model.predict(features_scaled.iloc[-1].values.reshape(1, -1))[0]
state_confidence = model.predict_proba(features_scaled.iloc[-1].values.reshape(1, -1)).max()

if current_state == 0 and state_confidence > 0.70:
    macro_regime = "STABLE_GROWTH"
    chop_filter_weights = {'csi_vol': 0.15, 'cny': 0.15, 'em_fx': 0.20, ...}
elif current_state == 1 and state_confidence > 0.70:
    macro_regime = "CHINA_CRISIS"
    chop_filter_weights = {'csi_vol': 0.35, 'cny': 0.25, 'em_fx': 0.20, ...}
elif current_state == 2 and state_confidence > 0.70:
    macro_regime = "SQUEEZE_RISK"
    chop_filter_weights = {'tightness': 0.40, 'supply': 0.30, ...}
else:
    macro_regime = "TRANSITIONING"
    chop_filter_weights = {'default': ...} # Balanced weights
```

Use case: Replace your manual "detect_macro_regime()" function with HMM

Pros:

- Automatic, data-driven
- Smooth transitions (no sudden jumps)
- Probabilistic (know when uncertain)

Cons:

- Needs 10+ years of data to train well
- State labels are arbitrary (you must interpret)
- Can misfire during novel regimes

Option 2: HMM for Vol/Trend Regime (Your Current Approach is Better)



python

```
# DON'T use HMM here
# Your current approach (percentile-based) is more transparent and works well
```

Why I prefer your current vol/trend regime detection:

- Simple, interpretable (30th percentile = LOW vol)
- Doesn't require training
- Works with limited data

Option 3: HMM for Changepoint Detection (Advanced)



python

```
# Use HMM to detect when a regime just shifted
state_changes = np.diff(states)
regime_shift_dates = dates[state_changes != 0]

# Alert when regime shifts
if date in regime_shift_dates:
    print(f' ⚠ REGIME SHIFT DETECTED: {states[date-1]} → {states[date]}')
    # Reduce position size for 5 days (uncertainty)
    sizing_multiplier *= 0.70
```

Use case: Early warning system for regime changes

Part 5: Final Recommendations

Recommendation 1: Adopt Your Roadmap Architecture (It's Better)

Your Tier 1+2 (Adaptive Blending) is superior to my standalone sleeve approach.

Reasons:

- 1. Proven foundation (your sleeves already work)
- 2. Clean separation (price vs fundamentals)
- 3. Scalable (test overlays independently)
- 4. Handles tails (price sleeves fire on any big move)

My contribution: Tier 3 overlays are valuable but NOT critical path.

Recommendation 2: Simplify Tier 3 (Focus on Tightness)

Instead of building all 3 overlays (chop, attribution, tightness), prioritize:

Month 3-4: Tightness Index ONLY

- This is your unique edge
- Easier to implement than ML attribution
- Clearer signal (tight = bullish, loose = bearish)

Month 5-6: Chop Detection (Optional)

- If Month 1-2 adaptive blending already works well, maybe you don't need this
- Test first: Does your portfolio already avoid chop zones naturally?

Month 7+: ML Attribution (Advanced, Optional)

- Only if you have quant bandwidth
- Marginal gain (0.05-0.10 Sharpe) vs complexity

Revised target: Sharpe 0.85-0.95 (base case), 1.00-1.10 (upside)

Recommendation 3: In-Sample / Out-of-Sample Split

For Price Sleeves (Tier 1):

- IS: 2010-2022 (12 years, covers crisis + QE + taper)
- OOS: 2023-2025 (3 years, inflation era)
- Recalibrate: Annually

For Adaptive Blending (Tier 2):

- IS: 2000-2020 (20 years, all regimes)
- OOS: 2021-2025 walk-forward
- Recalibrate: Quarterly



For Tightness Index (Tier 3):

- IS: 2010-2021 (11 years, includes 2 squeezes + 1 glut)
- OOS: 2022-2025 walk-forward (test on recent)
- Recalibrate: Quarterly (physical market changes fast)



Critical: Walk-forward, not static split

Recommendation 4: HMM Usage

Use HMM for:

1.  Macro regime classification (replace manual rules)
2.  Change point detection (regime shift alerts)

Don't use HMM for:

1.  Vol/trend regime (your percentile approach is better)
2.  Trading signals directly (too black box)

Implementation:



python

```
# Month 5 (after Tier 1+2 working)
# Add HMM as enhancement to chop detection
# Use 3-state model: STABLE, CRISIS, SQUEEZE
# Adjust chop filter weights based on HMM state
```

Recommendation 5: Realistic Targets

Don't promise 1.20 Sharpe. Here's what's achievable:

Milestone	Target Sharpe	Probability	Timeline
Adaptive blending	0.75-0.80	80%	Month 2
+ Tightness overlay	0.85-0.95	60%	Month 4
+ Chop detection	0.90-1.00	40%	Month 6
+ ML attribution	0.95-1.10	20%	Month 9

Base case: 0.85 Sharpe Stretch goal: 1.00 Sharpe Moonshot: 1.10 Sharpe

Why the other quant underperformed:

- Likely overfitted to IS data
- Or macro relationships broke
- Or transaction costs underestimated

Avoid this:

- Conservative cost assumptions (3-5bp)
- Short training windows (3 years for macro)
- Walk-forward validation (no hindsight)

The Bottom Line

Your roadmap is better than my plan. Here's why:

1. **Foundation first:** Your price sleeves are proven; my macro sleeves are unproven
2. **Clean architecture:** Overlays adjust sizing, don't generate signals
3. **Realistic targets:** 0.75-0.95 Sharpe is achievable; my 1.20 was aggressive
4. **Iterative:** Ship adaptive blending first, add overlays later

What to keep from my plan:

1. Tightness index (your unique edge)
2. ML attribution (optional, advanced)
3. Walk-forward validation (critical)
4. HMM for regime classification (enhancement)

What to discard from my plan:

1. Standalone macro sleeves (too low frequency)
2. Over-engineering (150 pages was overkill)
3. Unrealistic targets (1.20 Sharpe is a moonshot)

Final honest take: Build your Tier 1+2 (Months 1-2), add tightness overlay (Months 3-4), ship it. That's a 0.85 Sharpe system with proprietary edge. Everything else is marginal gains.