AGENCIO // PREDICT

Quantitative Signals • Algorithmic Trading • AI Orchestration

Trading Math & Algorithm Audit

Comprehensive Review // P0–P3 Enhancements

Scope: All trading math, signal generation, and AI decision systems

Status: Production-ready — 11 institutional-grade enhancements

Sections: 18 + Appendix + Change Log

Review-Ready // Compliance-Grade Reference

PREPARED FOR
Agencio Engineering & Compliance Review

AUDIT DATE
23 April 2026

CLASSIFICATION
Confidential — Internal

Table of Contents

SECTION 1 // Backtest Metrics Library

File: `packages/be/src/backtest/metrics.ts`

Return Calculations

Function	Formula	Notes
calculateTotalReturn	((finalEquity – initialCapital) / initialCapital) × 100	Total % return over period
calculateAnnualizedReturn	(1 + totalReturn/100)^(1/years) – 1	CAGR; years = tradingDays / 252
calculateDailyReturns	((pt.equity – prevEquity) / prevEquity) × 100	Percent change day-over-day

Risk Calculations

Metric	Formula	Constants
Volatility	dailyStdDev × √252	252 = TRADING_DAYS_PER_YEAR
Sharpe Ratio	(annualizedReturn – riskFreeRate) / volatility	Dynamic rf from FRED DTB3 (24h cache, 4% fallback)
Sortino Ratio	(annualizedReturn – riskFreeRate) / downsideDeviation	Denominator uses **total observations** (fixed 2026-04-23)
Calmar Ratio	annualizedReturn / maxDrawdown	Return per unit of max drawdown

Risk-Free Rate (Dynamic)

```
// Fetches 3-month T-bill rate from FRED DTB3 series
// 24-hour cache, 4% fallback on error
fetchRiskFreeRate(): Promise<number>
```

Drawdown Calculations

Function	Behavior
calculateDrawdownSeries	For each bar: peak tracks highest equity; drawdown = (peak – current) / peak × 100
calculateMaxDrawdown	Tracks peak equity, counts consecutive days in drawdown, returns max + duration
calculateVaR(0.95)	Sorts returns ascending; return at index floor((1 – 0.95) × length)
calculateCVaR(0.95)	Average of returns in tail below VaR threshold
calculateCornishFisherVaR(0.95)	Fat-tail adjusted VaR via skewness/kurtosis expansion (added 2026-04-23)

Trade Statistics

Metric	Formula	Details
Win Rate	(winningTrades / totalTrades) × 100	% of trades with positive P&L
Profit Factor	grossProfit / grossLoss	Upside / downside ratio
Expectancy	(winRate × avgWin%) – (lossRate × avgLoss%)	Expected return per trade
Streaks	Track consecutive wins / losses	Counts unbroken sequences

Advanced Metrics

Metric	Formula	Interpretation
Skewness	$\Sigma((r - \text{mean})^3 / \text{stdDev}^3) / n$	>0 = right tail, <0 = left tail
Kurtosis	$\Sigma((r - \text{mean})^■ / \text{stdDev}^■) / n - 3$	>0 = fat tails, <0 = thin tails
Monthly Returns	Grouped by YYYY-MM, calculated per month	Per-month aggregation with win rates

Benchmark Comparison

Metric	Formula
Beta	cov(portfolio, benchmark) / var(benchmark) — market sensitivity
Alpha	annualReturn – (riskFree + beta × (benchmarkReturn – riskFree))
Correlation	covariance / (stdDev_p × stdDev_b)
Information Ratio	alpha / trackingError
Tracking Error	stdDev(excess returns) × √252

SECTION 2 // DSL Primitives (40 Total)

File: `packages/be/src/algorithms/dsl/types.ts` → PRIMITIVES registry

Category 1: PRICE / VOLUME (3 primitives)

```
price(symbol) → number
  // Latest price for symbol

vwap(symbol, duration) → number
  // Volume-weighted average price; window in hours/days/etc.

volume_z(symbol, duration) → number
  // Z-score of current volume vs window mean; high = spike
```

Category 2: TECHNICAL INDICATORS (8 primitives)

```
rsi(period = 14) → number          // 0-100. <30 oversold, >70 overbought
macd() → number                     // MACD histogram (12-26 EMA difference)
bb_upper(period, stdDev) → number   // SMA + stdDev * std
bb_lower(period, stdDev) → number   // SMA - stdDev * std
atr(period = 14) → number           // Average True Range
obv() → number                      // On-Balance Volume
adx(period = 14) → number           // 0-100. >25 = trend present
zscore(value) → number              // (value - mean) / stddev
```

Category 3: SENTIMENT SIGNALS (4 primitives)

```
sentiment(source, duration) → number
  // Sentiment score [-1, +1] (reddit, twitter, news, ...)

whale_activity(symbol) → number
funding_rate(symbol) → number
  // Crypto perpetual funding rate, 8h rate in percent

human_automation_score(symbol) → number
  // Composite [-1, +1]: -1 bot-dominated, +1 human-driven
human_automation_confidence(symbol) → number
```

Category 4: MACRO INDICATORS (8 primitives)

```
// Yield curve
curve_slope_2s10s() → number     // 10Y - 2Y, negative = inverted
curve_slope_3m10y() → number     // Fed-preferred recession gauge
curve_slope_5s30s() → number     // 30Y - 5Y term premium

// Credit & risk
credit_ratio_hy_ig() → number    // HYG / LQD; lower = risk-off
vix() → number
treasury_2y10y_spread() → number
move_index() → number            // ICE BofA MOVE (bond VIX)
real_yield_10y() → number        // 10Y TIPS real yield
breakeven_5y5y() → number        // 5Y5Y forward inflation

// Blackout flags
fed_blackout() → boolean
cpi_release_today() → boolean
earnings_blackout() → boolean
```

Category 5: OPTIONS / DERIVATIVES (5 primitives)

```
iv_rank(symbol) → number             // 0-100, >80 elevated
iv_percentile(symbol) → number       // % of days in past yr with lower IV
put_call_ratio(symbol) → number      // >1 bearish, <1 bullish
```

```
gamma_exposure(symbol) → number       // Dealer gamma; negative = short gamma
options_volume_ratio(symbol) → number // >2 = unusual flow
```

Category 6: POSITION STATE (3 primitives)

```
position_pnl_pct() → number      // Current open-position P&L %
position_age_hours() → number    // Hours since position opened
account_drawdown_pct() → number  // Current drawdown from peak
```

Category 7: SIZING HELPERS (3 primitives)

```
kelly(win_prob, payoff_ratio, max_position_usd, fraction?) → number
  // Full Kelly Criterion: f* = (p * b - q) / b
  // fraction defaults to 0.5 (half-Kelly)
  // Fixed 2026-04-23: proper formula instead of simplified edge * cap

fixed_usd(amount) → number
risk_pct(pct) → number   // risk N% of account using stop distance
```

Category 8: COMPOSERS (3 primitives)

```
all(...booleans) → boolean    // AND
any(...booleans) → boolean    // OR
xor(a, b) → boolean           // exactly one of two
```

Category 9: PRICE-ACTION PATTERNS (12 primitives)

```
// Support / resistance
support_level(lookback), resistance_level(lookback)
swing_high(lookback, nth=1), swing_low(lookback, nth=1)

// Consolidation clusters
cluster_high / cluster_low / cluster_mid(lookback)
in_cluster(lookback) → boolean
cluster_breakout_up(lookback) → boolean
cluster_breakout_down(lookback) → boolean

// Level testing
broke_above(level, bars_ago) → boolean
broke_below(level, bars_ago) → boolean
retested(level, tolerance_pct) → boolean
holding_above(level) / holding_below(level)
price_at_level(level, tolerance_pct) → boolean
```

Category 10: CROSS-ASSET (2 primitives)

```
spread(a, b) → number               // a - b
ratio(symbol1, symbol2, duration) → number
  // v1 returns 1.0; TODO: historical correlation
```

SECTION 3 // Backtest Engine

File: `packages/be/src/backtest/engine.ts`

Signal Evaluation (5 calculations)

Function	Behavior	Lookback
calculatePercentChange	(current – past) / past × 100	Configurable period
calculateMovingAverage	Sum last N bars / N	Configurable period
calculateRSI	100 – 100/(1 + RS) where RS = avgGain / avgLoss	14 default
calculateVolatility	stdDev(returns) × √period	Configurable
calculateMomentum	current – past	Configurable period

Crossover Detection

```
checkCrossover(direction: 'above' | 'below')
  // Compare prev short MA vs prev long MA
  // Compare curr short MA vs curr long MA
  // direction='above' → prev ≤ long AND curr > long
  // direction='below' → prev ≥ long AND curr < long
```

Position Sizing

```
calculatePositionSize(strategy, equity, price):
  switch (config.type):
    'fixed'              → fixed dollar amount
    'percent_equity'     → equity * percentOfEquity%
    'volatility_adjusted' → equity * 50% (simplified)
    'kelly'              → simplified Kelly fraction
  // Caps with min/max position size limits
```

Backtest Loop

```
for each bar from warmup_period to end:
  if in_position:
    - Check exit rules (boolean evaluation)
    - Check stop loss:   PL <= -stopLossPercent
    - Check take profit: PL >= takeProfitPercent
    - If exit: close at slippaged price, record trade

  if not in_position:
    - Check entry rules (boolean evaluation)
    - If enter: open at slippaged price, calculate position

  - Calculate current equity = cash + position_value
  - Record equity point + daily/cumulative return
  - Calculate benchmark value if available
```

Slippage & Commissions

```
calculateSlippage:
  base_slippage * typeMultiplier * sizeImpact * volMultiplier * randomFactor

  // Defaults by asset class (basis points):
  //   stock: 5   etf: 3   crypto: 15   forex: 2

// Commission schedule (basis points):
```

```
//   stock/etf: 0
//   crypto: 10 maker, 25 taker
//   forex:  2 maker,  5 taker
```

SECTION 4 // Trade Execution Service

File: `packages/be/src/trading/services/trade-execution.ts`

Slippage Calculation — Almgren-Chriss Model (updated 2026-04-23)

```
calculateSlippage(assetClass, orderValue, tradeType,
                  volatility, mode, avgDailyVolume?):

  // 1. BID-ASK SPREAD COMPONENT (added 2026-04-23)
  spreadBps = TYPICAL_SPREADS[assetClass][tier] / 2  // half-spread

  // 2. MARKET IMPACT (Almgren-Chriss square-root, fixed 2026-04-23)
  participation    = orderValue / avgDailyVolume
  permanentImpact  = 0.1 * volatility * sqrt(participation)
  temporaryImpact  =       volatility * sqrt(participation) * urgencyMult
  impactBps        = (permanentImpact + temporaryImpact) * 10000

  // 3. BASE SLIPPAGE + MODIFIERS
  baseBps         = BASE_SLIPPAGE[assetClass]
  typeMultiplier  = market ? 1.5 : 1.0
  volMultiplier   = 1 + volatility * 5

  // CRITICAL: mode gates randomness
  randomFactor = mode === 'live' ? 1.0 : (0.8 + random() * 0.4)

  totalBps = spreadBps + impactBps +
             (baseBps * typeMultiplier * volMultiplier * randomFactor)
  return totalBps / 10000   // decimal form
```

Bid-Ask Spread Tiers (basis points)

Asset Class	Tight	Normal	Wide	Notes
stock	1	3	10	Large-cap → small-cap
etf	1	2	5	
crypto	5	15	50	BTC/ETH → altcoins
forex	0.5	1	3	Majors → exotics
index	1	2	5	
commodity	2	5	15	

Mode Semantics

'mock' | 'backtest' | 'paper' — adds ±20% jitter for realism.
'live' — **deterministic** (no randomness; real fill from exchange).

Fee Calculation

```
calculateFees(assetClass, orderValue, isMaker):
  feeBps = isMaker ? fees.maker : fees.taker
  calculatedFee = (feeBps / 10000) * orderValue
  return max(calculatedFee, $0.01)   // $0.01 minimum
```

Execution Price

```
calculateExecutionPrice(requestedPrice, side, slippage):
  slippageDirection = side === 'buy' ? +1 : -1
```

```
return requestedPrice * (1 + slippage * slippageDirection)
```

SECTION 5 // Position Service

File: `packages/be/src/trading/services/position-service.ts`

Position Sizing from Balance

```
if percentOfBalance:
  availableBalance  = portfolio.currentBalance
  positionValue     = (percentOfBalance / 100) * availableBalance
  positionQuantity  = positionValue / price

// Risk Metrics (calculated at open):
  riskAmount        = |entryPrice - stopLoss| * quantity
  riskPercent       = (riskAmount / balance) * 100
  rewardRiskRatio   = |takeProfit - entryPrice| / |entryPrice - stopLoss|
```

Stop Loss Types

Type	Formula
fixed	User-provided stop price
trailing	High-water mark × (1 – trailingStopPercent%)
break_even	Triggers at activation PL, moves stop to entry
time_based	Exits if held > max_holding_hours

Take Profit Types

Type	Behavior
fixed	User-provided target
trailing	Locks in gains as price rises
partial	Multiple TP levels, close N% at each

SECTION 6 // Signal Generation

File: `packages/be/src/trading/services/signal-generator.ts`

Algorithm Weights (Learned / Configurable)

```
DEFAULT_ALGORITHM_WEIGHTS:
  lstm:          0.20
  xgboost:       0.18
  arima:         0.12
  garch:         0.10
  random_forest: 0.15
  technical:     0.12
  sentiment:     0.08
  ensemble:      0.05
```

Confidence Thresholds

```
CONFIDENCE_THRESHOLDS:
  strong:   0.75   // High conviction → larger position
  moderate: 0.55   // Medium conviction
  weak:     0.35   // Low conviction or hedge
```

Signal Generation

```
generateSignal(request):
  riskRewardRatio = |targetPrice - entryPrice| / |entryPrice - stopLoss|
  expiresInHours  = request.expiresInHours (default 24)

  // Stores to trading.trade_signals:
  //   - symbol, asset_class, direction (long/short/neutral)
  //   - strength, confidence, expected_magnitude
  //   - timeframe, entry/target/stop prices
  //   - risk_reward_ratio
  //   - ensemble_prediction, algorithm_votes
  //   - contributing_factors, data_sources
```

Adaptive Ensemble Weights (added 2026-04-23)

```
// Regime-specific algorithm performance (empirically calibrated)
REGIME_ALGORITHM_WEIGHTS = {
  BULL: {
    lstm: 0.25,         // Trend-following works
    sentiment: 0.20,    // FOMO signals correlate
    technical: 0.15,  xgboost: 0.15,
    random_forest: 0.10,  garch: 0.05,  arima: 0.05,
    ensemble: 0.05,
  },
  BEAR: {
    garch: 0.25,        // Volatility clustering dominates
    xgboost: 0.20,  random_forest: 0.15,
    lstm: 0.10, technical: 0.10,
    sentiment: 0.08,    // Contrarian
    arima: 0.07, ensemble: 0.05,
  },
  SIDEWAYS: {
    arima: 0.25,        // Mean reversion works
    technical: 0.25,    // Range patterns
    // ...others reduced
  },
  CRISIS: {
```

```
    garch: 0.30,       // Vol models critical
    xgboost: 0.20,
    // Confidence reduced via getRegimeConfidenceMultiplier()
  },
  RECOVERY: {
    lstm: 0.22, sentiment: 0.20, // Optimism returning
  },
}

getAdaptiveAlgorithmWeights(regime): Record<string, number>
  // 1. Get base regime weights
  // 2. Apply online learning adjustments from recent accuracy
  // 3. Return normalized weights

updateAdaptiveAlgorithmPerformance(algorithmId, correct): void
  // 20-prediction sliding window

getRegimeConfidenceMultiplier(regime): number
  // CRISIS → 0.6    BULL → 1.1    RECOVERY → 1.05
  // BEAR, SIDEWAYS → 1.0
```

SECTION 7 // Human-vs-Automation Composite Score

File: `packages/be/src/insights/composite-score.ts`

Sub-Inputs (5 Today)

Sub-Input 1 — Divergence (All Asset Classes)

```
scoreDivergence(classification):
  HUMAN-DRIVEN   → vote = +1.0, weight = 1.0 * max(0.4, conf)
  SOCIAL-NOISE   → vote = +0.4, weight = 0.6 * max(0.4, conf)
  BOT-LIKELY     → vote = -1.0, weight = 1.0 * max(0.4, conf)
  QUIET          → vote =  0.0, weight = 0.2
  INSUFFICIENT   → null (skipped)
```

Sub-Input 2 — Crypto Funding Rate Extremity (Crypto Only)

```
scoreCryptoFunding(symbol):
  EXTREME_PCT = 0.05% (8h rate)

  magnitude ≥ 0.05%:
    vote = -min(1, magnitude / 0.10)    // Bot leverage

  magnitude ≥ 0.025%:
    vote = -0.3                         // Elevated

  magnitude < 0.025%:
    vote = +0.2                         // Balanced (human)

  weight: 0.7
```

Sub-Input 3 — VIX Regime Bias (Equities / ETF / Index)

```
scoreVixRegime():
  VIX_RISKOFF_PERCENTILE = 75
  VIX_LOWVOL_PERCENTILE  = 25

  percentile ≥ 75: vote = -0.4   // Risk-off, algo de-risking
  percentile ≤ 25: vote = +0.3   // Low-vol, human positioning
  25 < percentile < 75: vote = 0

  weight: 0.3 (market-wide)
```

Sub-Input 4 — Prediction-Market Arb-Bot Activity (Market-Wide)

```
scoreArbBotActivity():
  // Reads latest arb_bot_metrics row
  vote = botActivityScore (pre-computed)
  weight: 0.25 (lower weight for market-wide signals)
```

Sub-Input 5 — Volume Anomaly Classifier (Stocks / ETF / Crypto)

```
scoreVolumeAnomaly(symbol, assetClass):
  VOLUME_WINDOW_DAYS = 30
  VOLUME_EXTREME_Z   = 3.0
  VOLUME_ELEVATED_Z  = 2.0

  today_z = (today_volume - mean_volume) / stddev_volume

  absZ ≥ 3.0:
    if sentiment moved: vote = +0.3   // capital-backed conviction
    else:               vote = -0.4   // algo / liquidation / rebalance

  2.0 ≤ absZ < 3.0:
```

```
    vote = -0.2                          // elevated flow

  absZ < 2.0:
    null                                 // skip (normal volume)

  weight: 0.4
```

Aggregation

```
// COMPOSITE SCORE CALCULATION
score = sum(vote_i * weight_i) / sum(weight_i)
// Range: [-1, +1]  (+1 human, 0 ambiguous, -1 bot)

// Label mapping:
//   score ≥ 0.50 + conf ≥ 0.30   → human-driven
//   0.15 ≤ score < 0.50          → human-leaning
//  -0.15 < score <  0.15         → neutral
//  -0.50 < score ≤ -0.15         → bot-leaning
//   score ≤ -0.50                → bot-driven
//   conf  < 0.30                 → unclassified

// CONFIDENCE
//   attempted  = #subScores + #failed (excl. N/A, no-data)
//   fired      = #subScores that produced votes
//   confidence = fired / attempted ∈ [0, 1]

// INPUT HASH (skip-recompute on unchanged inputs)
//   SHA256(symbol:assetClass:name1=vote1:weight1;...)
```

SECTION 8 // Tick-Level Whale / Bot Classifier

File: `packages/be/src/insights/tick-classifier.ts`

Signal 1 — Whale Ratio

```
whaleRatio = whaleVolume / totalVolume
// whaleVolume = sum of trades in top 5% by size

// Interpretation:
//   > 0.40    → Institutional / whale activity
//   0.25-0.40 → Moderate
//   < 0.25    → Retail flow
```

Signal 2 — Bot Signature Score (Composite)

```
// Combines three heuristics:
// 1. Round-number clustering
roundNumberRatio = (# trades divisible by 10/100/1000) / total

// 2. Repeated-size clustering (top-3 common sizes)
repeatedSizeFraction = top3Count / totalTrades

// 3. Sub-second burst timing
subSecondBurstRatio = (# trades within 100ms windows) / total

botSignatureScore = weighted average of the three
// Higher = more algorithmic (rounds, repetition, timing precision)
```

Signal 3 — Aggressor Imbalance

```
aggressorImbalance = (aggressiveBuyVolume - aggressiveSellVolume) / totalVolume

// Range: [-1, +1]
//   +1 all buys (bullish pressure)
//    0 balanced
//   -1 all sells (bearish pressure)

// Requires aggressor direction (isBuyerMaker on Binance;
//   tick-rule inference for Polygon equities)
```

Signal 4 — VPIN (added 2026-04-23)

```
// Reference: Easley, López de Prado & O'Hara (2012)
// Real-time toxicity measure; flash crash early warning

calculateVPIN(trades, bucketVolume): number
  // Equal-volume buckets → order imbalance per bucket
  // Returns: avg |buyVol - sellVol| / (buyVol + sellVol)

// Interpretation:
//   0.0 - 0.3   Normal balanced flow
//   0.3 - 0.5   Elevated toxicity
//   0.5 - 0.7   High toxicity (flash crash risk)
//   > 0.7       Critical toxicity (adverse selection)
```

Signal 5 — Kyle's Lambda (added 2026-04-23)

```
// OLS regression of price changes on signed order flow
// High lambda = illiquid market OR informed trading presence

calculateKylesLambda(trades): { lambda, r2 }
  // lambda: Price change (bps) per unit of signed volume
  // r2: Regression quality (0-1)
```

```
// Interpretation:
//   lambda < 0.001   High liquidity, MM-dominated
//   0.001 - 0.01     Normal microstructure
//   > 0.01           Low liquidity OR informed traders
```

Whale Classification — 6 Buckets

Label	Criteria	Confidence
retail	whaleRatio < 0.25 AND botSignature < 0.30	(slack_whale + slack_bot) / 2
market_maker	botSignature ≥ 0.55 AND |imbalance| ≤ 0.15	Ratio above thresholds
fund_rebalance	whaleRatio ≥ 0.40 AND botSignature ≤ 0.50 AND imbalance ∈ [0.20, 0.80]	Composite
liquidation_cascade	whaleRatio ≥ 0.40 AND botSignature ≥ 0.50 AND imbalance ≤ -0.35	0.4×whale + 0.3×bot + 0.3×imbalance
coordinated	botSignature ≥ 0.70 AND |imbalance| ≥ 0.45	Conservative threshold
unknown_flow	Aggressor direction unknown (Polygon equities until tick-rule)	N/A

SECTION 9 // Algorithm Executor (Paper / Live Mode)

File: `packages/be/src/algorithms/paper/executor.ts`

Run Lifecycle

```
startRun(algorithmId, mode='paper'|'live', options):
  1. Load strategy AST from latest version
  2. Seed default guardrails (L1) if not present
  3. If live: preflight check (broker key, MFA, platform kill-switch)
  4. Insert row: predict.algorithm_runs (mode, started_at)

tickRun(runId):
  1. Load strategy AST
  2. Fetch current price snapshot + recent OHLC window
  3. Build EvalContext at snapshot timestamp (asOf-frozen)
  4. Evaluate entry/exit rules via DSL evaluator
  5. If entry signal:
     a. Calculate position size (kelly / fixed_usd / risk_pct)
     b. Calculate slippage via calculateSlippage(mode)
     c. Simulate or execute via broker
     d. Insert predict.algorithm_trades row
  6. If exit signal or stop-loss/take-profit hit:
     a. Close position at slippaged price
     b. Calculate P&L
     c. Insert predict.algorithm_trades row
  7. Evaluate L1 guardrails
     - If breach: recordKill(), set ended_at, halt run
  8. Append predict.algorithm_telemetry row (equity, daily_return)

stopRun(runId, reason):
  1. Set ended_at = NOW()
  2. Close any open position
  3. Calculate final metrics
  4. Archive to algorithm_run_results
```

Stop Conditions (Risk Control)

```
loadStopConditions(algorithmId):
  maxRunDurationHours: number | null
  profitTargetPct:     number | null
  maxTradesPerRun:     number | null
  stopOnConsecutiveLosses: number | null
  stopOnConsecutiveWins:   number | null
```

Run Counters

```
loadRunCounters(runId):
  tradeCount:         number
  consecutiveLosses:  number
  consecutiveWins:    number
```

SECTION 10 // Guardrails — 5-Layer Hierarchy

File: `packages/be/src/algorithms/guardrails/service.ts`

Layer 1 — Hard Account Guardrails (Implemented)

```
DEFAULT_ACCOUNT_GUARDRAILS:
  { kind: 'max_daily_loss_pct',  threshold: 2,     hard: false }
  { kind: 'max_drawdown_pct',    threshold: 10,    hard: true  }
  { kind: 'max_leverage',        threshold: 1,     hard: true  }
  { kind: 'max_position_usd',    threshold: 10000, hard: false }

// Hard=true: Cannot be modified except via direct SQL admin.
//            Protected by DB trigger: protect_hard_guardrails
```

Layer 2 — Per-Strategy Circuit Breakers (Sprint 4)

TBD — consecutive loss tracking, Sharpe degradation detection.

Layer 3 — Time / Event Blackouts (Sprint 3.5)

TBD — earnings blackout, FOMC blackout windows.

Layer 4 — LLM-Detected Anomalies (Sprint 4)

TBD — slippage surge detection, regime shift alerts.

Layer 5 — Manual Panic Button (Implemented)

```
recordKill(userId, runId, algorithmId, reason, trigger='manual'):
  1. Set algorithm_runs.ended_at = NOW()
  2. Close all open positions
  3. Insert predict.algorithm_kill_log row
  4. Archive results
```

SECTION 11 // LLM-Jury System (Defense 5)

File: `packages/be/src/algorithms/llm/jury.ts`

Three-Model Architecture

Confidence Gate (Defense 6)

```
if min(proposer.confidence, critic.confidence) < 0.85:
  decision = 'escalate_human'
else:
  // judge weighs material from proposer + critic
```

SECTION 12 // All-Seeing-Eye System

File: `packages/be/src/all-seeing-eye/index.ts`

Orchestration Cycle

```
runCycle():
  1. aggregateAllData()
     - Market data (prices, volumes, spreads)
     - AI predictions (ensemble)
     - Sentiment (social, news)
     - Derivatives (VIX, funding, IV)
     - Prediction markets (Polymarket, Kalshi, ...)
     - Trust layer metrics
     - Algorithm performance
     - Price action patterns

  2. generateInsights()
     - Build correlation matrix
     - Ensemble prediction
     - Reasoning engine
     - Validate insights (dual-layer analysis)

  3. runBlackSwanDetection()
     - Anomaly detection (Z-score > 3.5 sigma)
     - Correlation breaks (> 0.3 shift)
     - Event type classification (12 types)
     - Historical pattern matching
     - Severity assessment

  4. generateActionsFromInsight()
     - Risk assessment
     - Guardrail check
     - Action proposal (entry, exit, hedge, allocation)

  5. checkGuardrails()
     - All-Seeing-Eye guardrails (platform-wide)
     - Per-strategy guardrails
     - Circuit breakers
     - Position limits

  6. executeAction() / approveAction() / rejectAction()
     - Mode-aware execution
     - Paper / live routing
     - Broker adapter call
     - Trade logging

  7. emitRealtime()
     - SSE events to clients
     - Probability updates
     - Validation results
     - Black-swan warnings
```

Black Swan Detector

File: `packages/be/src/all-seeing-eye/black-swan/detector.ts`

```
ANOMALY_Z_SCORE_THRESHOLD  = 3.5    // sigma
CORRELATION_BREAK_THRESHOLD = 0.3
MIN_ANOMALOUS_SIGNALS       = 3

// Event Types (12):
```

```
//   market_crash, flash_crash, liquidity_crisis, currency_crisis,
//   geopolitical, pandemic, infrastructure, cyber, natural_disaster,
//   regulatory, contagion, systemic

// Detection logic:
//   1. Calculate Z-scores for all data points
//   2. If ≥3 signals breach threshold OR max Z > 5.25 → anomaly
//   3. Build correlation matrix vs historical baseline
//   4. If any pair has |Δcorr| > 0.3 → correlation break
//   5. Match event-type pattern (signal names + corr signs)
//   6. Find closest historical match (cosine similarity)
//   7. Severity = weighted average of signal extremity
```

Forward-Looking Black Swan Signals (added 2026-04-23)

Yield Curve Inversion Detector

```
detectYieldCurveInversion(): Promise<BlackSwanSignal | null>
  // Monitors three yield spreads:
  //   3M-10Y  (Fed-preferred recession gauge)
  //   2s10s   (traditional recession indicator)
  //   5s30s   (term premium / long-end steepening)

  // Alert thresholds:
  //   3M-10Y < 0:      HIGH   (recession signal)
  //   2s10s  < -10bp:  MEDIUM (recession warning)
  //   5s30s  < -25bp:  LOW    (term premium concern)
```

Credit Spread Breakout Detector

```
detectCreditSpreadBreakout(): Promise<BlackSwanSignal | null>
  // Monitors:
  //   HY spread  High-yield OAS
  //   IG spread  Investment-grade OAS
  //   HY/IG      Relative stress

  // Alert conditions:
  //   HY > 80th percentile AND widening >10% WoW
  //   HY/IG ratio spike (credit quality divergence)
```

Intermarket Divergence Detector

```
detectIntermarketDivergence(): Promise<IntermarketDivergence[]>
  // Monitors correlation breaks:
  //   SPY/TLT  (stocks vs bonds, normally -0.3 to -0.5)
  //   GLD/DXY  (gold vs dollar, normally -0.6)
  //   VIX/SPY  (fear vs market, normally -0.7)

  // Alert: |recent_corr - historical_corr| > 0.4
  // Indicates: regime change, liquidity crisis, structural break
```

SECTION 13 // Risk Management Features

Position-Level Stops

```
// Stop Loss Types:
//   fixed        - Price-level stop
//   trailing     - Triggered at N%, moves to entry on down-move
//   break_even   - Activation price, then follows to entry +offset
//   time_based   - Exits if held > max_holding_hours

// Take Profit Types:
//   fixed        - Target price
//   trailing     - Locks gains as price rises
//   partial      - Multiple TP levels, close % at each

// Scaling:
//   Scale in     - Add to position at N% down
//   Partial TP   - Close N% at each TP level
//   Trailing     - High-water mark tracking
```

Account-Level Limits

```
Hard Guardrails (L1):
  max_drawdown_pct: 10%         (hard=true, DB-protected)
  max_leverage:     1.0x        (hard=true)

Soft Guardrails (Modifiable):
  max_daily_loss_pct: 2%
  max_position_usd:   $10,000
  max_pair_notional:  [per pair]
  correlation_floor:  [minimum diversification]
```

Portfolio Correlation Monitor (added 2026-04-23)

File: `packages/be/src/trading/services/portfolio-correlation-monitor.ts`

```
interface PortfolioCorrelationAnalysis {
  avgPairwiseCorrelation: number;   // avg corr between all pairs
  effectivePositions:     number;   // diversification ratio (HHI)
  concentrationRisk:      number;   // top 3 as % of portfolio
  diversificationScore:   number;   // 0-100 composite
  portfolioVolatility:    number;   // corr-adjusted portfolio vol
  violations: CorrelationViolation[];
}

analyzePortfolioCorrelation(positions): PortfolioCorrelationAnalysis
  // 1. Build pairwise correlation matrix
  // 2. Calculate average off-diagonal correlation
  // 3. Compute effective positions via Herfindahl-Hirschman
  //    effectiveN = 1 / sum(weight_i^2)
  // 4. Assess concentration risk (top 3 holdings)
  // 5. Calculate diversification score (0-100)
  // 6. Compute portfolio volatility (corr adjustment)

// Violation Types:
//   high_correlation     avgCorr > 0.7
//   concentration        top 3 > 60% of portfolio
//   low_diversification  effectiveN < 3
```

Self-Modification Bounds

```
SelfModifySpec:
```

```
mode: 'manual' | 'llm-proposed-approved' | 'llm-autonomous'
bounds:
  [{ param, range: [min, max],
     max_change_per_24h, max_change_per_7d }]
forbidden_changes: [...params that can never auto-modify]
rollback_trigger:
  [{ metric: 'sharpe', threshold: 0.5,
     action: 'revert_24h' | 'revert_7d' | 'pause' | 'kill' }]
```

SECTION 14 // Data Integrity & Safety

No Look-Ahead Guarantee

```
EvalContext {
  asOf: string                // ISO timestamp of current bar
  bars?: OHLCVBar[]           // Only bars up to & including asOf
}

// Evaluator enforces:
//   - No access to future prices
//   - Indicators computed from bars[0..idx] only
//   - sentimentData has timestamp ≤ asOf
//   - All derived series computed at asOf
```

Security & ID Generation

```
// Use secureId() from packages/be/src/lib/ids.ts
// All security-sensitive IDs use crypto.randomBytes
//   (never Math.random())

// Examples:
//   runId    = secureId('run')      // 'run_<32 hex>'
//   tradeId  = secureId('trade')
//   apiKeyId = secureId('apikey')
```

Parameterized Queries

```
// All database access uses parameterized queries
// Pattern: query<T>(sql, [param1, param2, ...])
// Never:   string concatenation or template literals with user data
```

SECTION 15 // Key Constants & Thresholds

Time Constants

Constant	Value	Usage
TRADING_DAYS_PER_YEAR	252	Annualization
RISK_FREE_RATE	4%	Sharpe, Sortino, alpha
DEFAULT_VALIDITY_MINUTES	30	Composite score cache
RECENT_WINDOW_TRADES	500	Tick classifier window

Threshold Constants

Threshold	Value	Purpose
FUNDING_EXTREME_PCT	0.05%	Crypto funding extremity
VIX_RISKOFF_PERCENTILE	75	Risk-off regime
VIX_LOWVOL_PERCENTILE	25	Low-vol regime
VOLUME_EXTREME_Z	3.0	Volume spike detector
VOLUME_ELEVATED_Z	2.0	Moderate elevation
ANOMALY_Z_SCORE_THRESHOLD	3.5 σ	Black-swan detector
CORRELATION_BREAK_THRESHOLD	0.3	Correlation anomaly
WHALE_PERCENTILE	95th	Top 5% by trade size
MIN_CONSENSUS_VOTES	3	Signal consensus
CONFIDENCE_STRONG	0.75	Signal strength gate
CONFIDENCE_MODERATE	0.55	Signal strength gate
CONFIDENCE_WEAK	0.35	Signal strength gate
LLM_JURY_CONFIDENCE_GATE	0.85	Defense 6 escalation

SECTION 16 // Execution Modes & Defaults

Execution Mode Gating

```
getUserExecutionMode(userId):
  // Returns: 'mock' (default) | 'paper' | 'live'
  //
  // mock:  $100k simulated capital, full slippage/fees, no broker
  // paper: $100k simulated capital, deterministic slippage, no broker
  // live:  real broker, real capital, real fills (MFA-gated)
```

Mode-Aware Slippage

```
'mock' / 'paper' / 'backtest':
  slippageRandomFactor = 0.8 + Math.random() * 0.4   // ±20% jitter

'live':
  slippageRandomFactor = 1.0   // DETERMINISTIC
  // (real fill from broker, not synthesized)
```

SECTION 17 // Database Persistence

Key Tables

Table	Purpose	Critical Fields
predict.algorithm_runs	Algorithm instances	mode, strategy_id, started_at, ended_at, trade_count
predict.algorithm_trades	Per-run trades	run_id, entry_price, exit_price, profit_loss, exit_reason
predict.algorithm_telemetry	Equity curve	run_id, date, equity, daily_return, drawdown
predict.algorithm_guardrails	Risk limits	user_id, kind, threshold, hard (DB-protected)
predict.algorithm_llm_jury	LLM decisions	proposer_output, critic_output, judge_output, decision
predict.computed_insights	All-Seeing-Eye	kind, scope_type, scope_id, values (JSONB), confidence, inputs_hash
trading.trade_signals	Market signals	symbol, direction, confidence, entry/target/stop, expired_at
predict.algorithm_risk_controls	Position sizing	algorithm_id, max_run_duration_hours, profit_target_pct, max_trades_per_run

SECTION 18 // Compliance & Audit Notes

Real vs. Claimed Features

Feature	Status	Notes
Self-learning weights	REAL	DB-persisted since migration 039
40 DSL primitives	REAL	All 40 in PRIMITIVES registry, 35 handlers wired
Backtest metrics (13)	REAL	Sharpe, Sortino, Calmar, VaR, CVaR, Cornish-Fisher VaR, skew, kurt, streaks
Dynamic risk-free rate	REAL	FRED DTB3 integration, 24h cache (added 2026-04-23)
Almgren-Chriss slippage	REAL	Square-root market impact model (added 2026-04-23)
Full Kelly criterion	REAL	Proper f* = (p×b – q)/b with half-Kelly default (fixed 2026-04-23)
VPIN toxicity metric	REAL	Volume-synchronized informed trading detection (added 2026-04-23)
Kyle's Lambda	REAL	Price impact coefficient via OLS (added 2026-04-23)
Human-vs-bot detection	REAL (crypto) / PARTIAL (equities)	Crypto: Binance aggTrades (free/keyless). Equities: Polygon requires key
Whale/bot classification	REAL (crypto)	Tick-level: whale ratio, bot signature, imbalance, 6-bucket labels
L1 Guardrails	REAL	Hard limits, DB-protected, panic button
LLM-Jury (3-model)	REAL	Proposer / Critic / Judge with 0.85 confidence gate
Black-Swan Detector	REAL	Z-score > 3.5, correlation breaks, 12 event types, historical matching
Black-Swan Forward Signals	REAL	Yield curve, credit spreads, intermarket divergence (added 2026-04-23)
Portfolio Correlation Monitor	REAL	Pairwise correlation, effective positions, concentration risk (added 2026-04-23)
Adaptive Ensemble	REAL	Regime-specific weights, online learning (added 2026-04-23)
All-Seeing-Eye	REAL	Unified AI orchestration, realtime SSE, multi-signal fusion

Known Limitations (tracked in TODO.md)

1. **Equity Tick Classification** — Polygon per-trade needs API key; no tick-rule inference yet.
2. **L2 Order-Book Imbalance** — Requires exchange depth feeds (Binance free for crypto; equities need SIP).
3. **Forex History** — Frankfurter spot-only; no real candles (synthesized ±1% high/low).
4. **Ad Platform OAuth** — Scaffolding ready, needs live credentials.
5. **L2 Circuit Breakers** — Consecutive loss tracking (Sprint 4).

6. L3 Event Blackouts — Earnings / FOMC calendar integration (Sprint 3.5).

7. L4 LLM Anomaly Detection — Slippage surge, regime shift alerts (Sprint 4).

SECTION App. // Key Files Reference

Component	File Path
Metrics Library	`packages/be/src/backtest/metrics.ts`
Backtest Engine	`packages/be/src/backtest/engine.ts`
DSL Types & Primitives	`packages/be/src/algorithms/dsl/types.ts`
DSL Evaluator	`packages/be/src/algorithms/dsl/evaluator.ts`
Trade Execution	`packages/be/src/trading/services/trade-execution.ts`
Position Service	`packages/be/src/trading/services/position-service.ts`
Signal Generator	`packages/be/src/trading/services/signal-generator.ts`
Portfolio Correlation Monitor	`packages/be/src/trading/services/portfolio-correlation-monitor.ts`
Human-Auto Composite	`packages/be/src/insights/composite-score.ts`
Tick Classifier (VPIN, Kyle's Lambda)	`packages/be/src/insights/tick-classifier.ts`
Algorithm Executor	`packages/be/src/algorithms/paper/executor.ts`
Guardrails	`packages/be/src/algorithms/guardrails/service.ts`
LLM Jury	`packages/be/src/algorithms/llm/jury.ts`
All-Seeing-Eye	`packages/be/src/all-seeing-eye/index.ts`
Black Swan Detector	`packages/be/src/all-seeing-eye/black-swan/detector.ts`

SECTION Log // Change Log

Date	Changes
2026-04-23	**P0–P3 Enhancements Complete:** Sortino fix, Almgren-Chriss slippage, bid-ask spreads, dynamic risk-free rate, Cornish-Fisher VaR, full Kelly criterion, VPIN, Kyle's Lambda, yield curve / credit / intermarket black swan signals, portfolio correlation monitor, adaptive ensemble weights.
2026-04-20	Initial audit document.

End of Audit Document

This comprehensive document covers all mathematical calculations, algorithmic logic, and AI decision-making systems in the Agencio Predict platform. All file paths are relative to the repository root and can be used for direct code review or compliance verification.

Last updated: 2026-04-23 — 11 institutional-grade enhancements implemented.