A Statistical Arbitrage Quant Terminal Pairs Trading, LSTM Diagnostics, and GPU-Accelerated Stochastic Risk Engine

Aditya Chachan[1], Aditya Surampally[2], Akash Raj[3], Arnav Saxena[4], Ishitva Singh[5], Bhaskar M G[6]
[1,2,3,4,5]*Department of EEE RV College of Engineering Bengaluru, India*
[6]*Professor, Department of EEE RV College of Engineering Bengaluru, India*

***Abstract*—This paper presents the design, theoretical foun- dations, and empirical evaluation of a Statistical Arbitrage (StatArb) Quant Terminal built in Python using Streamlit. The system integrates three core engines: an OLS Pairs Trading Engine that exploits the cointegrated price spread between HDFCBANK.NS and ICICIBANK.NS; a Simulated LSTM Di- agnostics module that visualises what a deep learning training process would look like applied to spread prediction; and a Stochastic Risk Engine that performs GPU-accelerated Monte Carlo simulation of 1,000,000 forward paths using Geometric Brownian Motion (GBM). The paper derives the theoretical basis for each component, reviews the quantitative outputs against their analytical counterparts, and identifies the key limitations of the system in its current form. The backtested strategy produces a cumulative return of 7.61% over approximately five years with a Sharpe ratio of 0.18 and a maximum drawdown of −322%, reflecting the genuine difficulty of profitable statistical arbitrage in efficient markets. The Monte Carlo risk estimates — 95% VaR of −1804% and CVaR of −2230% — are mathematically consistent with GBM theory and are calibrated to historical strategy parameters.**

***Index Terms*—Statistical Arbitrage, Pairs Trading, Cointegra- tion, OLS Regression, Z-Score, LSTM, Monte Carlo Simulation, Geometric Brownian Motion, Value at Risk, CVaR, GPU Accel- eration, Indian Equity Markets**

I. INTRODUCTION

Statistical arbitrage is a class of market-neutral quantitative strategies that seek to profit from temporary mispricings in the relative prices of related financial instruments, rather than directional bets on any individual asset. Among the most widely studied implementations is pairs trading, which exploits the cointegration between two economically linked securities whose prices, while individually non-stationary, share a stable long-run equilibrium. When their spread deviates sufficiently far from that equilibrium, the strategy opens a position betting on reversion and closes it once the mean is restored.

The pair studied in this work is HDFCBANK.NS versus ICICIBANK.NS — two of the largest private-sector banks listed on the National Stock Exchange (NSE) of India. Both institutions share near-identical exposure to RBI repo rate decisions, non-performing asset (NPA) cycles, retail lending growth trends, and Nifty Bank index weightings. This struc- tural commonality makes them strong cointegration candi- dates: their prices wander individually (non-stationary random walks), but their spread tends to revert to a mean, which is the exploitable signal.

The Quant Terminal described in this paper combines three complementary engines into a unified dashboard. The first is an OLS Pairs Trading Engine that finds and trades the spread. The second is a Simulated LSTM Diagnostics module that il- lustrates what a deep learning model trained on spread features would look like. The third is a Stochastic Risk Engine that uses GPU-accelerated GBM simulation to compute forward risk metrics including Value at Risk (VaR) and Conditional Value at Risk (CVaR).

A. Problem Statement

The fundamental flaw in most algorithmic trading

models— especially at the academic and retail level— is the attempt to predict non-stationary, random-walk data, namely individual stock prices. This approach relies on directional guessing of whether a single asset will rise or fall, which is mathematically equivalent to trying to extract a predictable signal from pure noise: such models react too slowly to sudden market moves, rely on frictionless simulations that ignore broker fees, and assess risk only by looking backward at historical drawdowns, implicitly assuming that the worst outcome still to come cannot exceed the worst outcome already observed.

B. Proposed Solution

To address this, the system is engineered to treat financial markets as a digital signal processing problem rather than a price-prediction problem. Instead of attempting to predict the unpredictable trajectory of an individual stock, the Quant Terminal extracts a clean, stationary signal by pairing two struc- turally related assets (HDFCBANK.NS and ICICIBANK.NS) and trading the mathematical spread between them through cointegration-based statistical arbitrage, deliberately ignoring general market direction. This execution logic is paired with a PyTorch-driven, CUDA-accelerated Monte Carlo risk engine capable of simulating 1,000,000 alternate future realities of the strategy in under five seconds, allowing the system to calculate absolute, forward-looking tail risk (Value at Risk and Conditional Value at Risk) rather than relying solely on what has already happened historically. Realistic execution frictions— a hardcoded 0.10% transaction cost on every signal change— are deducted directly from the backtest, so that reported returns reflect net, real-world profitability rather than an idealised, frictionless simulation.

C. Motivation and Scope

The Indian equity market provides a particularly interesting environment for testing pairs trading strategies because both stocks underwent significant structural changes during the backtest window (2021–2026), most notably the HDFC-HDFC Bank merger in 2023. This creates a known structural break that challenges the assumption of a static hedge ratio— a limitation discussed in detail later. The system is designed both as a working strategy backtest and as an educational demonstration of how quantitative risk management tools are applied in practice.

D. Key Contributions

The contributions of this paper are as follows:

- A complete theoretical derivation of the OLS hedge ratio, Z-score signal, and vectorised backtest logic for pairs trading on Indian banking equities.
- A formal review of the backtested performance metrics (Sharpe ratio, cumulative return, maximum drawdown) against their theoretical interpretations.
- A description and verification of a GPU-accelerated Monte Carlo GBM risk engine, including mathematical checks of the 95%/99% VaR, CVaR, median outcome, and probability of loss against analytical GBM formulae.
- An honest assessment of the system's limitations: static OLS hedge ratio, absence of stop-loss mechanisms, and the synthetic (non-trained) nature of the LSTM diagnostics.

II. BACKGROUND AND RELATED WORK

A. Cointegration and Pairs Trading

The theoretical foundation for pairs trading rests on the concept of cointegration, introduced by Engle and Granger [1]. Two non-stationary time series $P_1(t)$ and $P_2(t)$ are said to be cointegrated if there exists a linear combination $\varepsilon(t) = P_1(t) - \beta P_2(t) - \alpha$ that is stationary (mean-reverting). The parameter β is the hedge ratio and $\varepsilon(t)$ is the spread to be traded.

The empirical application of pairs trading to financial mar- kets has been extensively studied [2], with the basic strategy documented to produce statistically significant excess returns in US equity markets over multi-decade horizons, though profitability has diminished as the strategy has become more widely known [3]. In the Indian context, the banking sector presents an especially natural domain for pairs construction owing to the regulatory and macroeconomic homogeneity among large private-sector lenders.

B. OLS as the First-Step Estimator

OLS regression is the standard first-step estimator

in the Engle-Granger cointegration framework. Under the Gauss- Markov theorem, OLS gives the minimum-variance linear unbiased estimate of β, making it an appropriate starting point. Its chief limitation is the assumption thatβ is constant over the estimation window. Dynamic alternatives such as the Kalman filter [4] allowβ to evolve over time and generally improve both signal quality and drawdown characteristics.

C. Monte Carlo Methods in Risk Management

Monte Carlo simulation under GBM is the workhorse of financial risk modelling, underpinning not only Value at Risk estimation but also the Black-Scholes option pricing frame- work [6]. GPU-accelerated implementations of Monte Carlo simulation have made it feasible to generate millions of paths in seconds, enabling real-time risk dashboard applications.

D. LSTM for Financial Time Series

Long Short-Term Memory networks [5] have attracted substantial interest for financial time series prediction owing to their ability to capture long-range temporal dependencies. Applied to spread prediction in pairs trading, an LSTM would be trained on rolling features derived from the spread history and would attempt to improve the timing of entries and exits relative to a simple threshold-based Z-score rule.

III. SYSTEM ARCHITECTURE

The Quant Terminal is built in Python using Streamlit as the front-end framework. The three engines operate independently but share a common data pipeline that downloads OHLCV data for both stocks via the Yahoo Finance API, aligns the series on trading dates, and computes the spread and its derivatives. The GPU computations are performed using PyTorch with CUDA, with results transferred back to NumPy for Python- side analysis and display.

A. Data Pipeline

Historical daily closing prices for HDFCBANK.NS and ICI- CIBANK.NS are fetched for the period 2021–2026, yielding approximately 1,260 trading days. The data are cleaned for missing values and adjusted for corporate actions. The raw price series are used directly as inputs to the OLS regression; log-returns are computed separately for the GBM risk engine calibration.

IV. IPART I: OLS PAIRS TRADING ENGINE

A. Hedge Ratio Estimation

The engine fits a static OLS regression of the form:

$$HDFCBANK_t = \alpha + \beta \cdot ICICIBANK_t + \varepsilon_t \quad (1)$$

The estimated parameters over the full backtest window are:

- $\hat{\beta} = 0.322$: for every 1 unit of ICICIBANK held long, 0.322 units of HDFCBANK are held on the opposite side.
- $\hat{\alpha} = 465.09$: the long-run price level offset between the two stocks.
- ε_t: the residual series, which is the stationary spread we trade.

The spread is therefore computed as:

$$Spread_t = HDFCBANK_t - 0.322 \cdot ICICIBANK_t - 465.09 \quad (2)$$

OLS minimises the sum of squared residuals and is the the-oretically correct starting estimator under the Engle-Grange framework. Its key limitation is that$\hat{\beta}$is held constant over the entire 5-year window, which is unrealistic given structural events such as the HDFC-HDFC Bank merger in 2023.

B. Z-Score Signal Generation

Raw spread values are standardised into a Z-score using a rolling window of 30 days:

$$Z_t = \frac{Spread_t - \mu_t}{\sigma_t} \quad (3)$$

whereμt andσt are the rolling 30-day mean and standard deviation of the spread, respectively. The trading rules derived from the Z-score are summarised in Table I.

TABLE I Z-SCORE TRADING RULES

Z-Score	Meaning	Action
$Z > +2$	Spread abnormally wide	SHORT spread Z
< -2	Spread abnormally narrow	LONG spread Z
crosses 0	Spread mean-reverted	EXIT position

The rolling Z-score chart from 2021–2026 shows the Z-score oscillating between approximately−3 and +3.5, with multiple complete cycles of entry at±2, reversion through zero, and exit. This behaviour is consistent with a mean- reverting cointegrated spread. The system recorded 61 signal events over the backtest period, corresponding to approxi- mately 12 position changes per year—a trading frequency that is neither excessively

noisy nor too sparse for the 30-day lookback window.

C. Vectorised Backtest Logic

The signal generation is fully vectorised to avoid Python loops:

$$\text{Position}_t = \begin{cases} +1 & \text{if } Z_t < -Z_{entry} \quad \text{(long spread)} \\ -1 & \text{if } Z_t > +Z_{entry} \quad \text{(short spread)} \\ 0 & \text{if zero crossing} \quad \text{(exit)} \end{cases} \tag{4}$$

Positions are forward-filled (hold until the next signal) and shifted by one day to avoid look-ahead bias, so that today's signal drives tomorrow's return. Strategy returns and net returns are computed as:

$$R_t^{strategy} = \text{Position}_{t-1} \times (r_t^{HDFC} - 0.322 \cdot r_t^{ICICI}) \tag{5}$$

$$R_{net} = R_{strategy} - \text{Transaction Cost}_t \tag{6}$$

Transaction costs of 0.10% per trade are applied as a flat cost each time the position changes, which is realistic for Indian NSE markets accounting for brokerage, STT, and slippage.

D. Performance Metrics

1) Sharpe Ratio: 0.18: The Sharpe ratio is computed as:

$$\text{Sharpe} = \frac{\bar{r}_{daily}}{\sigma_{daily}} \times \sqrt{252} \tag{7}$$

A value of 0.18 is very low—a Sharpe of 1.0 is considered

the minimum threshold for a viable strategy, and values above 2.0 are considered excellent. This result suggests the strategy is barely generating risk-adjusted excess return and has high return volatility relative to its mean. This outcome is realistic for a simple static OLS pairs strategy operating under transaction costs; more sophisticated implementations using dynamic hedge ratios and improved entry/exit timing would be expected to produce higher Sharpe ratios.

2) Cumulative Return: 7.61%:

$$R_{cumulative} = \prod_t (1 + r_t) - 1 \tag{8}$$

A cumulative return of 7.61% over approximately five years implies roughly 1.5% per year—comparable to a savings account. As a market-neutral strategy, this should be evaluated against its absolute return rather than equity market bench- marks. That said, 7.61% total over 5 years with a Sharpe of 0.18 barely justifies the strategy's complexity and risk. The 0.10% transaction cost per trade is meaningfully eroding returns

3) Maximum Drawdown:−32.20%:

$$\text{MDD} = \min_t \frac{\text{Equity}_t}{\max_{s \le t} \text{Equity}_s} - 1 \tag{9}$$

A drawdown of−32.2% is dangerously severe for a suppos-edly market-neutral strategy. A Calmar ratio (annual return / max drawdown) of approximately 1.5%/32.2% ≈ 0.047 is extremely poor. This drawdown most likely arose from either prolonged spread divergence, structural break in the hedge ratio around the HDFC-HDFC Bank merger in 2023, or the absence of any stop-loss mechanism in the backtest.

4) Hedge Ratio$\hat{\beta}$: 0.322: For stocks priced at approxi-mately—HDFCBANK at∼INR 1,600 and ICICIBANK at∼INR 1,000—$\hat{\beta}$= 0.322 is plausible given the price ratio.

However, a static$\hat{\beta}$estimated over five years will drift—the actual relationship in 2026 may differ substantially from 2021, particularly following the 2023 corporate restructuring.

V. PART II: SIMULATED LSTM DIAGNOSTICS

A. Architecture Overview

The LSTM Diagnostics module visualises what an AI model's training process would look like if applied to spread prediction. It is explicitly disclosed in the user interface as a synthetic training-cycle visualisation with no live model inference performed. The simulated architecture comprises five layers:

- LSTM-1 (64 units): First recurrent layer extracting raw temporal patterns.
- LSTM-2 (32 units): Second recurrent layer for abstrac-tion and compression.
- Attention Layer: Learns to weight which time steps matter most.
- Dense-1 (16 units): Fully connected layer combining features.
- Output Gate: Final signal prediction.

B. Input Features

Eight engineered features are used as hypothetical inputs to the LSTM: (1) Spread Momentum (5d), (2) Spread Momentum (21d), (3) Volatility Vector Y (HDFCBANK realised volatil- ity), (4) Volatility Vector

X (ICICIBANK realised volatility), (5) Cross-Asset Correlation (rolling), (6) Z-Score Level, (7) Volume Imbalance, and (8) Term Spread Drift.

In the feature weight heatmap, the Volatility Vector (X) has a normalised weight of 0.176 in the Attention layer, indicat- ing that ICICIBANK's volatility is moderately important in determining which time steps receive the most focus.

TABLE II SIMULATED LSTM TRAINING METRICS

Metric	Value	Assessment
Final Train Loss (MSE)	0.05265	Low— model fits training data
Final Val Loss (MSE)	0.07582	Acceptable
Generalisation Gap	0.02318	Val− Train; mild overfitting
Learning Rate	2.50×10^{-4}	Stable, cosine-decayed

The MSE loss is defined as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y - y_i)^2 \qquad (10)$$

The training loss being consistently lower than validation loss is normal and expected, as the model has seen the training data. A generalisation gap of 0.023 is relatively small, suggesting only mild overfitting. The learning rate schedul follows step decay: lr = 0.001×0.5⌊epoch/18⌋, halving every 18 epochs.

The loss curves are generated synthetically as:

train loss(e) = 0.85·e−e/11 + 0.042 +η (11)

val loss(e) = 0.92·e−e/13.5+ 0.055 +η (12)

where η represents additive noise. This exponential decay shape is textbook-correct for real LSTM training curves in the early epochs before plateauing.

VI. PART III: STOCHASTIC RISK ENGINE

A. Geometric Brownian Motion Theory

The core risk model uses Geometric Brownian Motion (GBM), the same stochastic process that underlies Black- Scholes option pricing [6]. GBM assumes that equity returns follow the stochastic differential equation:

$$\frac{dS}{S} = \mu\, dt + \sigma\, dW_t \qquad (13)$$

where μ is the drift, σ is the volatility, and dWt is the increment of a standard Wiener process. The GBM parameters are calibrated from historical strategy returns:

• μ= 0.0093%/day (from strat_ret.mean())

• σ= 0.8202%/day (from strat_ret.std()) The discrete-time simulation draws daily returns as:

rt∼N (μ,σ), Equityt = Equityt−1×(1 + rt) (14) Across T = 252 trading days (one year) and N = 1,000,000 independent paths, this generates the complete forward distri- bution of strategy equity outcomes. With N = 1M paths, the Law of Large Numbers ensures that Monte Carlo estimates of any percentile converge to the true analytical value with approximately 0.01% error.

B. GPU Acceleration

The entire computation is performed in GPU VRAM in float32:

shocks = N (μ,σ)252×1M (CUDA) (15)

paths = $_t$(1 + shockst) (torch.cumprod) (16)

This approach generates 252,000,000 random numbers in a single CUDA kernel call and computes all cumulative products simultaneously. The compute time on an RTX 5070 Laptop GPU is 3.09 seconds for 252M float32 operations— demonstrating the practical feasibility of real-time risk dashboard applications.

C. Risk Metrics

1) 95% Value at Risk (1Y):−18.04%: VaR at 95% confi- dence is the 5th percentile of the terminal return distribution: in the worst 5% of scenarios, the minimum loss is−18.04%. The analytical GBM 5th percentile is

$$VaR_{95\%} = \exp\left(\mu T - z_{0.05}\sigma\sqrt{T}\right) - 1 \qquad (17)$$

$$= \exp(0.000093\times 252 + (-1.645)\times 0.008202\times\sqrt{252}) - 1$$

$$= \exp(0.02344 - 0.21420) - 1 = \exp(-0.19076) - 1 \approx -17.4\%$$

The simulation gives −18.04%. The small difference from−17.4% is expected owing to convexity bias from discrete arithmetic returns (not log-returns) and ∼0.1–0.2% sampling error at extreme percentiles. The result is mathematically

C. Training Metrics

The simulated training results are summarised in Table II.

2) 99% Value at Risk (1Y):−24.99%:

$$VaR_{99\%} = \exp\left(\mu T - z_{0.01}\sigma\sqrt{T}\right) - 1 \approx -24.4\% \qquad (18)$$

The simulation gives −24.99%, again within the expected convexity and discretisation bias range. Correct and consistent. 3) 95% CVaR / Expected

Shortfall:−22.30%: CVaR (Con- ditional Value at Risk), also known as Expected Shortfall, is the expected loss given that the scenario is already in the worst 5%:CVaR95% = E[Return | Return < VaR95%] (19)

By construction, CVaR must be more negative than VaR. The result−22.30% <−18.04% satisfies this requirement. The spread between VaR and CVaR of 4.26 percentage points indicates the tail beyond the 5th percentile is not extremely fat, consistent with the approximately normal GBM distribution.

4) Median Outcome: 1.51%: The GBM median is: Median = exp(μT)−1 = exp(0.000093×252)−1≈2.37% (20)

The simulation gives 1.51% rather than 2.37% due to the volatility drag effect (Jensen's inequality / Itˆo's lemma):

$$\text{Annualised drag} = \frac{\sigma^2}{2} \times 252 = \frac{(0.008202)^2}{2} \times 252 \approx 0.85\%/\text{year} \quad (21)$$

Adjusted median≈ 2.37%− 0.85% = 1.52%, matching the simulation. Correct.

5) Probability of Loss: 45.4%: The analytical probability of loss under log-normal GBM is:

$$P(\text{loss}) = \Phi\left(\frac{-\mu T}{\sigma \sqrt{T}}\right) = \Phi(-0.18) = 1 - \Phi(0.18) \approx 42.9\% \quad (22)$$

The simulation gives 45.4%, which is2–3percentage points above the analytical estimate. This discrepancy arises because discrete arithmetic GBM slightly understates the upward drift relative to continuous GBM, and is within normal Monte Carlo discretisation error for this parameter set.

VII. RESULTS

This section presents the system in its deployed, end-to-end operating state. Screenshots of the live Quant Terminal dashboard are provided for each of the three engines (AI Core, Execution, and Risk Engine), followed by a structural comparison between the engineered approach taken in this work and conventional academic implementations.

A. Execution Engine: Mean-Reversion Signal Surface

Fig. 1 shows the live "leash" monitor for the pair. The blue line tracks the rolling 30-day Z-score, i.e. how stretched the HDFCBANK/ICICIBANK relationship is at any given point in time relative to its recent mean. The red and green markers indicate exactly where the spread crosses the±2 entry thresholds, which is where the system automatically triggers a SHORT or LONG signal on the spread. At the time of capture, the dashboard reports a Sharpe ratio of 0.22, a cumulative return of 10.65%, a maximum drawdown of−32.21%, a 95% VaR (1Y) of −17.69%, a hedge ratio βˆ of 0.321, and 61 total signal events, consistent in magnitude with the theoretical values derived in Part I.

Fig. 1. Execution Engine dashboard: rolling 30-day Z-score ("Mean- Reversion Signal Surface") for HDFCBANK.NS vs. ICICIBANK.NS from 2021–2026, with± 2 entry thresholds, current Z-score, current position, OLS intercept α, and spread mean displayed alongside top-line strategy KPIs

B. AI Core: Simulated LSTM Diagnostics Fig. 2 shows the telemetry panel for the simulated neural network. The two curves are the Training Loss and Validation Loss, which converge smoothly over 50 epochs towards the shaded "convergence zone." This convergence pattern is the vi- sual signature expected of a model that is learning underlying structure in the data rather than simply memorising the training set, since the validation curve tracks the training curve closely instead of diverging upward. The accompanying training log lists the per-epoch train loss, validation loss, and learning rate for the final eight epochs, ending at a train loss of 0.05265, a validation loss of 0.07582, and a generalisation gap of 0.02318–consistent with the values derived analytically in Part II. As noted throughout the paper, this module is explicitly disclosed in the interface as a synthetic, non-trained visualisation rather than live model inference.

Fig. 2. AI Core dashboard: simulated LSTM loss convergence over 50 training epochs, with the last-eight-epoch training log (train loss, val loss, learning rate) and generalisation gap shown alongside the top-line strategy KPIs.

C. Risk Engine: GPU Monte Carlo Simulation

Fig. 3 shows the output of the GPU-accelerated Monte Carlo engine, which simulates 1,000,000 possible one-year-forward futures for the strategy's equity curve under GBM. The blue "fan"chart plots a representative subsample of 1,000 of these simulated paths in equity-multiple terms, bounded by the 5th-percentile (VaR) and 95th-percentile envelopes and centred on the median path. The histogram quantifies the full terminal return distribution across all 252-day horizons, with the 95% VaR and median outcome marked directly on the distribution. Together, the fan chart and histogram allow the exact Value at Risk to be read off visually as well as numerically: the worst- case 95% scenario is mathematically contained, with a 95% VaR (1Y) of−17.69%, a 99% VaR (1Y) of−24.71%, a 95% CVaR (Expected Shortfall) of−21.99%, a median outcome of 2.09%, and a probability of loss of 43.8% at the time of capture.

Fig. 3. Risk Engine dashboard: GPU Monte Carlo fan chart of 1,000 (of 1,000,000 computed) simulated equity paths over a 252-day horizon, with 5th/95th percentile and median overlays, alongside the tail-risk metrics table and terminal return distribution histogram.

D. Comparison with Conventional Academic Models

Tables III and IV summarise, feature by feature, how the engineered approach taken in the Quant Terminal differs from the conventional academic and retail baseline it is designed to improve upon. The central distinction is the shift from backward-looking, frictionless, directional prediction toward forward-looking, friction-aware, market-neutral signal extraction.

VIII. INTERNAL CONSISTENCY REVIEW

Table V summarises the key internal consistency checks across the system. All mathematical relationships between risk metrics hold as required.

A. Issues and Concerns

1) The Sharpe–VaR Tension: With a daily Sharpe of 0.18 and$\sqrt{\sigma}$ = 0.82%/day, the daily drift is μ = 0.18×0.82%/ 252≈0.0093%/day, exactly matching the displayed μ. The numbers are internally consistent but reveal a strategy with a razor-thin edge: the drift is 88× smaller than the daily volatility.

TABLE III COMPARISON WITH CONVENTIONAL ACADEMIC MODELS (I)

Feature	Conventional Models	Quant Terminal
Core Philosophy	Directional guessing of single-stock movement	Market neutrality via statistical arbitrage on the HDFC/ICICI spread
Mathematical Basis	Lagging indicators (SMA, RSI)	OLS regression for β and rolling Z-score for entries
Execution Realism	Frictionless, ignores broker fees	Hardcoded 0.10% transaction cost per signal change
Risk Management	Historical drawdown only	Forward-looking GBM simulation of 1,000,000 paths

TABLE IV COMPARISON WITH CONVENTIONAL ACADEMIC MODELS (II)

Feature	Conventional Models	Quant Terminal
Risk Metrics	Standard deviation of daily volatility only	Tail-risk analysis via 95% VaR and CVaR
Computational Engine	CPU-bound NumPy/Pandas loops	GPU-accelerated PyTorch CUDA tensor vectorisation
AI/ML Integration	Black-box next-day price prediction	Diagnostic neural telemetry (loss convergence, feature weights)
User Interface	Static charts rendered after completion	Progressive chunked streaming of simulated paths to a live UI

2) Max Drawdown vs VaR: The historical maximum draw- down(−32.2%) is worse than the forward VaR(−18.04%). While the VaR is a one-year forward estimate and the draw- down is cumulative over five years, this discrepancy suggests the fitted GBM parameters may not fully capture tail risk seen historically. Structural breaks such as the HDFC-HDFC Bank merger are not captured by simple GBM.

3) Static OLS Suboptimality: A staticβˆ= 0.322 estimated
over five years for these two stocks is a known limitation. The hedge ratio almost certainly shifted during the 2023 HDFC merger. A Kalman filter would dynamically trackβand likely improve both Sharpe and drawdown characteristics.
4) Absence of Stop-Loss: The backtest has no stop-loss or position size limit. A position is held indefinitely until the Z-score mean-reverts through zero. During a regime where cointegration breaks down temporarily, this can produce very large drawdowns—which is the most likely explanation for the−32.2% observed.

IX. DISCUSSION AND IMPLICATIONS

A. On Strategy Performance

The overall assessment is that the mathematics, theory, and implementation of the Quant Terminal are sound and internally consistent. The strategy's weak performance is not a bug; it reflects the genuine difficulty of profitable statistical

TABLE V INTERNAL CONSISTENCY CHECKS

Check	Required	Result
VaR < CVaR < 0 (ordering)	−18.04%> −22.30%	√Correct
Median≫ VaR	1.51%≫ −18.04%	√Correct
Prob(loss) < 50% giveny > 0	45.4%	√Correct
GPU compute time plausible	3.09s for 252M ops	√Correct
Train loss < Val loss throughout	By construction	√Correct
Generalisation gap	0.07582− 0.05265	√Matches
Z-score chart respects ±2 thresholds	Entries marked	√Correct
Spread mean≈ 0	OLS residual mean =0	√Correct

arbitrage in efficient markets. Simple OLS pairs trading on HDFCBANK/ICICIBANK with static parameters produces very modest risk-adjusted returns (Sharpe 0.18) with alarming drawdown risk(−32.2%). This is a known result in the pairs trading literature: strategies that worked in the 1990s have largely been arbitraged away as the approach has become mainstream [3].

B. Paths to Improvement

Three concrete enhancements would materially improve the strategy:
1) Dynamic hedge ratio: Replace static OLS with a Kalman filter to trackβin real time, adapting to structural breaks.
2) Stop-loss mechanism: Implement a maximum loss threshold per trade (e.g., Z > 3.5 as a stop) to prevent runaway drawdowns when cointegration temporarily breaks down.
3) Real LSTM integration: Replace the synthetic LSTM diagnostics with an actual trained model that predicts Z-score direction, potentially improving entry/exit timing above the threshold-only rule.

C. On the Risk Engine

The Monte Carlo risk estimates are mathematically correct and properly calibrated to historical parameters. The 3.09- second GPU compute time for 1,000,000 paths demonstrates that real-time risk dashboard applications are computationally feasible on modern consumer-grade hardware. The dashboard represents a professional-grade implementation of the theory.

X. CONCLUSION

This paper has presented and reviewed a Statistical Arbitrage Quant Terminal comprising an OLS pairs trading engine, a simulated LSTM diagnostics module, and a GPU- accelerated Monte Carlo stochastic risk engine, applied to the HDFCBANK.NS/ICICIBANK.NS pair on the Indian NSE. The theoretical foundations of each component have been derived, and the quantitative outputs have been verified against their analytical counterparts.
The backtested strategy produces a five-year cumulative return of 7.61% (Sharpe 0.18, max drawdown −32.2%), reflecting the inherent challenge of generating consistent alpha from a simple, static cointegration framework. The Monte
Carlo risk engine produces a 95% VaR of−18.04% and CvaR of−22.30%, both mathematically consistent with GBM theory and calibrated to the historical strategy parameters.
The principal limitations—a static OLS hedge ratio, no stop-loss, and synthetic rather than real LSTM training—are clearly identified. These limitations point

to natural extensions:
Kalman filter-based dynamic hedging, hard stop mechanisms, and end-to-end machine learning integration for signal gener- ation. The system in its current form serves as a theoretically rigorous and practically illuminating demonstration of how quantitative finance methods are applied to real Indian equity market data.

REFERENCES

[1] R. F. Engle and C. W. J. Granger, "Co-integration and error correction: Representation, estimation, and testing," Econometrica, vol. 55, no. 2, pp. 251–276, Mar. 1987.
[2] E. Gatev,W.N.Goetzmann,and K G.Rouwenhorst,"Pairstrading: Performance of a relative-value arbitrage rule," Review of Financial Studies, vol. 19, no. 3, pp. 797–827, 2006.
[3] B. Do and R. Faff, "Does simple pairs trading still work?" Financial Analysts Journal, vol. 66, no. 4, pp. 83–95, 2010.
[4] J. D. Hamilton, Time Series Analysis. Princeton: Princeton University Press, 1994.
[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[6] J. C. Hull, Options, Futures, and Other Derivatives, 10th ed. New York:Pearson, 2017.
[7] M. Avellaneda and J.-H. Lee, "Statistical arbitrage in the US equities market," Quantitative Finance, vol. 10, no. 7, pp. 761–782, 2010.
[8] G. Vidyamurthy, Pairs Trading: Quantitative Methods and Analysis. Hoboken: Wiley, 2004.