

Deep Hedging under Transaction Costs: An Auditable NumPy Implementation and Exploratory Study

Alpha Kabinet TOURÉ¹

¹Alpha Stochastic Research

July 2026

Abstract

This computational note presents a from-scratch NumPy implementation of a neural hedging policy for a European call under discrete rebalancing and transaction costs. The implementation makes the terminal-loss convention, finite-sample Conditional Value-at-Risk estimator, transaction-cost indexing, network derivatives, and simulation schemes explicit. The accompanying code includes automated checks for empirical-CVaR boundary weights, cash translation, simulator reproducibility, and finite-difference agreement of the network and pathwise loss gradients. Archived single-run experiments show that the fitted policy traded less as proportional costs increased and produced lower CVaR point estimates than a cost-blind Black–Scholes delta at the positive cost levels examined. These observations are descriptive only. The available results do not quantify variation across training seeds, do not yet provide a matched comparison with tuned cost-aware benchmarks, and do not constitute a like-for-like reproduction of the original Deep Hedging Heston experiment. The paper is therefore intended as an auditable implementation study and as a basis for a later confirmatory evaluation, not as a claim of statistical or general performance superiority.

Keywords: deep hedging, conditional value-at-risk, transaction costs, stochastic control, Heston model, reproducibility.

JEL classification: G13, C45, C63.

Scope of evidence. The numerical tables reproduced in this manuscript are archived single-run estimates. Unless explicitly stated otherwise, they are not accompanied by repeated-training uncertainty or paired confidence intervals and should not be interpreted as confirmatory evidence.

1 Introduction

In the frictionless Black–Scholes–Merton model, continuous trading in the underlying and money-market account permits exact replication of a European option under the standard assumptions [Black and Scholes, 1973, Merton, 1973]. Implemented strategies instead trade on a finite grid and may incur transaction costs. The resulting hedging problem balances residual risk against trading activity and is generally different from continuous-time frictionless replication. Analytical and numerical approaches remain available in structured frictional models, including asymptotic approximations, stochastic control, and variational inequalities [Leland, 1985, Hoggard et al., 1994, Whalley and Wilmott, 1997].

Bühler et al. [2019] formulate hedging with generic frictions as a discrete-time stochastic-control problem and approximate admissible strategies with neural networks. Their framework

supports nonlinear risk criteria and multiple hedging instruments. The present work does not attempt to reproduce their complete architecture or empirical design. It studies a smaller setting in which a shared-weight NumPy network hedges a European call under empirical CVaR.

The aims of this paper are limited and technical:

1. state the loss, trading-cost convention, empirical-CVaR estimator, and analytic derivatives in a form that can be checked line by line;
2. compare the manual derivatives with centered finite differences and document which validation steps are complete or still outstanding;
3. preserve the archived numerical results while describing them as single-run observations rather than evidence of statistical dominance; and
4. separate the present state-only, stock-only experiments from the broader Deep Hedging framework on which they are based.

The paper is best read as an implementation and reproducibility study. It does not propose a new deep-hedging algorithm, establish optimality, or claim that a neural policy generally dominates classical hedging rules.

2 Scope Relative to the Deep Hedging Literature

Table 1 summarizes the main differences between the reported experiment in [Bühler et al. \[2019\]](#) and the present implementation. The comparison is included to prevent a qualitative resemblance from being mistaken for a full numerical reproduction.

Accordingly, the numerical section examines whether the fitted policy displays lower trading intensity and different tail-loss point estimates in the chosen simulations. It does not test equivalence to the original architecture, instrument set, or Heston comparison.

3 Related Work

The study sits at the intersection of incomplete-market hedging, coherent risk measurement, and neural stochastic control. Classical option replication under continuous frictionless trading follows [Black and Scholes \[1973\]](#) and [Merton \[1973\]](#). Once proportional transaction costs or discrete trading are introduced, perfect replication generally fails and optimal policies may be characterized by asymptotic corrections, stochastic control, or variational inequalities [[Leland, 1985](#), [Hoggard et al., 1994](#), [Whalley and Wilmott, 1997](#)]. The present work does not claim that neural networks replace these analytical methods; rather, they provide a flexible policy class when the objective, frictions, and scenario generator are specified numerically.

The risk criterion is Expected Shortfall, also called Conditional Value-at-Risk. Its optimization representation is due to [Rockafellar and Uryasev \[2000\]](#); the coherent-risk perspective is developed by [Artzner et al. \[1999\]](#) and the convex-risk framework by [Föllmer and Schied \[2002\]](#). The finite-batch distinction made in this paper is important because profiling the empirical threshold solves the batch problem exactly but does not remove sampling error relative to population CVaR.

Neural approaches to derivative pricing and hedging predate modern deep learning [[Hutchinson et al., 1994](#)]. [Bühler et al. \[2019\]](#) formulate Deep Hedging as direct optimization of a convex risk functional over neural trading strategies. Related reinforcement-learning formulations include [Kolm and Ritter \[2019\]](#), [Halperin \[2020\]](#), and [Cao et al. \[2021\]](#). The current paper differs from these works by emphasizing auditability: a compact NumPy implementation, explicit pathwise

Table 1: Scope of the present implementation relative to the original Deep Hedging experiment.

Component	Bühler et al. [2019]	Present study
Policy state	Market information and the preceding position enter the semi-recurrent policy.	Time and log-moneyness enter the GBM policy; truncated variance is added under Heston. The previous position is absent from the reported state-only results.
Time parametrization	The reported numerical study uses date-specific parameters; recurrent variants are also discussed.	One shared-weight network is evaluated at every trading date.
Network	The reported implementation uses two hidden layers, ReLU activations, and batch normalization.	Two hidden layers with 16 tanh units each and no batch normalization.
Heston instruments	Stock and an idealized variance swap.	Stock only.
Heston comparison	A model-consistent complete-market comparison using both instruments.	A fixed-volatility Black–Scholes delta with less information than the neural policy.
Primary criterion	General convex risk measures and indifference-pricing experiments.	Empirical $\text{CVaR}_{0.95}$ of terminal loss under a fixed cash convention.

derivatives, exact empirical-tail weights, and a transparent account of which parts of the original experiment are not reproduced.

Stochastic-volatility simulations use the Heston model [Heston, 1993] and a full-truncation Euler discretization following Lord et al. [2010]. Because the Feller condition need not hold for the selected stress parameters, the repository includes substep diagnostics and treats Heston comparisons as simulation-dependent rather than model-free evidence.

4 Discrete-Time Hedging Problem

4.1 Trading dates, payoff, and self-financing gain

Let

$$0 = t_0 < t_1 < \dots < t_n = T$$

be the trading grid, and let $S_k := S_{t_k}$ be the underlying price. The liability is a short European call

$$H(S_n) = (S_n - K)_+.$$

At t_k , for $k = 0, \dots, n-1$, the hedger holds δ_k units of the underlying over $[t_k, t_{k+1}]$. Set $\delta_{-1} = 0$. The self-financing gain from the risky asset is

$$G_T(\delta) = \sum_{k=0}^{n-1} \delta_k (S_{k+1} - S_k).$$

Interest rates are set to zero in all simulations.

Define the trade at date t_k by

$$u_k := \delta_k - \delta_{k-1}, \quad k = 0, \dots, n-1.$$

For a general formulation, set $\delta_n = 0$ and define the terminal liquidation trade $u_n = -\delta_{n-1}$. The total trading cost is

$$C_T(\delta) = \sum_{k=0}^n c_k(u_k).$$

Following the numerical convention used in the experiments reported below, terminal liquidation is free: $c_n \equiv 0$. Thus the risky position is valued through its final gain over $[t_{n-1}, T]$, and no additional fee is charged for closing it at maturity. This convention must be distinguished from a model with a positive terminal liquidation cost.

Given initial cash p_0 , terminal profit and loss is

$$\Pi_T(p_0, \delta) = p_0 - H(S_n) + G_T(\delta) - C_T(\delta), \quad (1)$$

and terminal loss is

$$L_T(p_0, \delta) := -\Pi_T(p_0, \delta) = H(S_n) - p_0 - G_T(\delta) + C_T(\delta). \quad (2)$$

The numerical tables use $p_0 = 0$. They therefore report an uncentered liability loss, not the net hedging error after receipt of an option premium.

4.2 Transaction-cost functions

For $k < n$, the two specifications are

$$c_k^{\text{lin}}(u) = \kappa S_k |u|, \quad (3)$$

$$c_k^{\text{quad}}(u) = \kappa S_k u^2. \quad (4)$$

The parameter κ is dimensionless under the chosen convention. For proportional costs, $\kappa = 0.01$ means a one-way cost equal to one percent of the traded notional, which is high for a liquid equity and should be interpreted as a stress level rather than as a universal market calibration.

4.3 Translation of the loss origin

Proposition 1 (Cash translation). *For any deterministic cash amount a and any policy class $\mathcal{D}$,*

$$\arg \min_{\delta \in \mathcal{D}} \text{CVaR}_\alpha(L_T(p_0 + a, \delta)) = \arg \min_{\delta \in \mathcal{D}} \text{CVaR}_\alpha(L_T(p_0, \delta)).$$

Moreover, absolute CVaR differences between two policies are invariant to the cash shift, whereas ratios and percentage improvements generally are not.

Proof. Equation (2) gives $L_T(p_0 + a, \delta) = L_T(p_0, \delta) - a$. CVaR is cash-equivariant, so $\text{CVaR}_\alpha(L - a) = \text{CVaR}_\alpha(L) - a$. The same constant is subtracted from the objective for every policy and from both members of an absolute comparison. A ratio changes because its denominator is shifted. $\square$

Accordingly, the primary comparisons below are stated in absolute loss units. Relative percentages are retained only as descriptive summaries under the explicit convention $p_0 = 0$.

5 Risk Criterion and Empirical Optimization

5.1 Population CVaR

For an integrable loss L and confidence level $\alpha \in (0, 1)$, the Rockafellar–Uryasev representation is

$$\text{CVaR}_\alpha(L) = \min_{w \in \mathbb{R}} \left\{ w + \frac{1}{1 - \alpha} \mathbb{E}[(L - w)_+] \right\}. \quad (5)$$

For a continuous loss distribution, a minimizer is the usual α -quantile. With atoms, the minimizer is generally a quantile set, and CVaR should not be reduced uncritically to a conditional expectation given $L \geq \text{VaR}_\alpha(L)$.

The function in Equation (5) is convex in w for a fixed policy. It is not generally convex in the neural-network parameters.

5.2 Finite-batch empirical CVaR

For a batch of B simulated losses $L_1(\theta), \dots, L_B(\theta)$, define

$$\hat{F}_B(\theta, w) = w + \frac{1}{(1 - \alpha)B} \sum_{i=1}^B (L_i(\theta) - w)_+. \quad (6)$$

Profiling out w minimizes the *empirical batch objective* exactly. It does not make the resulting random mini-batch criterion identical to the population objective, because in general

$$\mathbb{E}_{\mathcal{B}} \left[\min_w \hat{F}_{\mathcal{B}}(\theta, w) \right] \neq \min_w \mathbb{E}_{\mathcal{B}} \left[\hat{F}_{\mathcal{B}}(\theta, w) \right].$$

The profiled estimator is nevertheless a standard consistent empirical approximation as the batch size increases under the usual regularity conditions.

For an exact finite-sample implementation, sort the batch losses in descending order,

$$L_{[1]} \geq L_{[2]} \geq \dots \geq L_{[B]},$$

and let

$$q = (1 - \alpha)B, \quad r = \lfloor q \rfloor, \quad \lambda = q - r.$$

Then, for $q > 0$, empirical expected shortfall is

$$\widehat{\text{CVaR}}_{\alpha,B} = \frac{1}{q} \left(\sum_{j=1}^r L_{[j]} + \lambda L_{[r+1]} \right), \quad (7)$$

with the fractional term omitted when $\lambda = 0$. Away from ties, an exact gradient is

$$\nabla_{\theta} \widehat{\text{CVaR}}_{\alpha,B} = \frac{1}{q} \left(\sum_{j=1}^r \nabla_{\theta} L_{[j]} + \lambda \nabla_{\theta} L_{[r+1]} \right). \quad (8)$$

At ties, this expression is interpreted as a valid subgradient after a specified tie-breaking rule. A hard indicator that includes only losses strictly above a sample quantile omits the fractional boundary observation whenever $(1 - \alpha)B$ is non-integer. For $B = 4,096$ and $\alpha = 0.95$, the exact tail mass is 204.8: 204 observations receive full weight and the next observation receives weight 0.8.

5.3 Threshold profiling and training stalls

If w is updated jointly by stochastic gradient descent, its batch derivative is

$$\frac{\partial \widehat{F}_B}{\partial w} = 1 - \frac{1}{(1 - \alpha)B} \sum_{i=1}^B \mathbf{1}\{L_i > w\},$$

away from ties. When w exceeds every loss in a batch, the policy receives no gradient through the positive-part terms, while w moves only by its optimizer step. Profiling w within each batch avoids this particular zero-mask event. This is an implementation advantage, not a proof that the stochastic optimization problem has become convex or noiseless.

6 Policy Class and Analytic Gradients

6.1 State-only shared policy

The hedge at each date is generated by a shared-weight feedforward network,

$$\delta_k = g_{\theta}(x_k) = \delta_{\max} \tanh(f_{\theta}(x_k)), \quad \delta_{\max} = 1.5. \quad (9)$$

The network f_{θ} has two hidden layers with 16 tanh units per layer. Under GBM,

$$x_k = (\tau_k, \log(S_k/K)), \quad \tau_k = T - t_k,$$

and under Heston,

$$x_k = (\tau_k, \log(S_k/K), v_k^+).$$

This policy class is not recurrent. In particular, it cannot map the same market state to different actions for different current inventories. The cost terms couple the outputs during training, but the gradient is not a state variable available at inference. Thus the architecture can fit a smoother or more conservative state map, but it cannot represent a genuine inventory-dependent no-trade region of the form $\delta_k = \varphi(x_k, \delta_{k-1})$. This restriction is central when interpreting the turnover results.

6.2 Network backward pass

For one observation, the forward pass is

$$\begin{aligned} z_1 &= xW_1 + b_1, & a_1 &= \tanh z_1, \\ z_2 &= a_1W_2 + b_2, & a_2 &= \tanh z_2, \\ z_3 &= a_2W_3 + b_3, & \delta &= \delta_{\max} \tanh z_3. \end{aligned}$$

Given an upstream derivative $g_\delta = \partial\mathcal{L}/\partial\delta$,

$$\begin{aligned} g_{z_3} &= g_\delta \delta_{\max} (1 - \tanh^2 z_3), & \nabla_{b_3}\mathcal{L} &= g_{z_3}, \\ \nabla_{W_3}\mathcal{L} &= a_2^\top g_{z_3}, & g_{z_2} &= g_{a_2} \odot (1 - a_2^2), \\ g_{a_2} &= g_{z_3} W_3^\top, & \nabla_{b_2}\mathcal{L} &= g_{z_2}, \\ \nabla_{W_2}\mathcal{L} &= a_1^\top g_{z_2}, & g_{z_1} &= g_{a_1} \odot (1 - a_1^2), \\ g_{a_1} &= g_{z_2} W_2^\top, & \nabla_{b_1}\mathcal{L} &= g_{z_1}. \\ \nabla_{W_1}\mathcal{L} &= x^\top g_{z_1}, \end{aligned}$$

Because the same parameters are used at every date, the parameter gradients are summed over $k = 0, \dots, n-1$ before each optimizer update.

6.3 Indexed transaction-cost gradient

Let $u_j = \delta_j - \delta_{j-1}$ for $j < n$ and $u_n = -\delta_{n-1}$. From Equation (2),

$$\frac{\partial L_T}{\partial \delta_k} = -(S_{k+1} - S_k) + c'_k(u_k) - c'_{k+1}(u_{k+1}), \quad k = 0, \dots, n-1, \quad (10)$$

where $c'_n \equiv 0$ under the free-terminal-liquidation convention. The important index is $k+1$ in the second cost derivative. For an interior date, the subsequent transaction cost is scaled by S_{k+1} , not by S_k .

For proportional costs and $u \neq 0$,

$$c'_j(u) = \kappa S_j \operatorname{sign}(u). \quad (11)$$

At $u = 0$, the subdifferential is $\kappa S_j[-1, 1]$; the implementation uses zero as a valid subgradient. For quadratic costs,

$$c'_j(u) = 2\kappa S_j u. \quad (12)$$

6.4 Gradient checks

Table 2 reproduces the archived centered finite-difference check for a small network using step size 10^{-5} . The reported relative-error convention is

$$\frac{|g_{\text{ana}} - g_{\text{FD}}|}{\max\{1, |g_{\text{ana}}|, |g_{\text{FD}}|\}}.$$

The existing quadratic-cost pathwise check reports a maximum absolute deviation of 2.5×10^{-10} and a maximum relative deviation of 1.0×10^{-9} . The current repository additionally tests the S_{k+1} index, the fractional empirical-CVaR boundary weight, and the inventory-aware backward pass. An independent automatic-differentiation comparison of the complete profiled empirical-CVaR parameter gradient remains outstanding.

Table 2: Analytic network backward pass versus centered finite differences ($\epsilon = 10^{-5}$).

Parameter	Maximum absolute difference	Maximum reported relative error
W_1	1.65×10^{-9}	1.39×10^{-9}
b_1	3.14×10^{-10}	5.91×10^{-11}
W_2	5.07×10^{-11}	9.35×10^{-10}
b_2	1.30×10^{-10}	3.09×10^{-11}
W_3	5.85×10^{-11}	1.89×10^{-11}
b_3	7.73×10^{-11}	1.28×10^{-11}

6.5 Optimization

Parameters are updated with Adam [Kingma and Ba, 2015], using learning rate 3×10^{-3} , $\beta_1 = 0.9$, and $\beta_2 = 0.999$. Fresh Monte Carlo batches of 4,096 paths are generated at every iteration. The reported experiments use 1,200–1,500 updates depending on the configuration. This is on-the-fly Monte Carlo stochastic optimization rather than optimization over a fixed finite training set.

For exact reproducibility, the Adam stabilizer ϵ , parameter dtype, random-number generator, seed, feature scaling, initialization distribution, iteration count by experiment, and checkpoint-selection rule must also be archived. These items cannot be inferred from the reported point estimates.

7 Simulation and Experimental Design

7.1 Geometric Brownian Motion

Under GBM with zero drift,

$$S_{k+1} = S_k \exp\left(-\frac{1}{2}\sigma^2\Delta t + \sigma\sqrt{\Delta t} Z_{k+1}\right), \quad Z_{k+1} \sim N(0, 1),$$

so the grid-point simulation is exact.

7.2 Heston full-truncation discretization

The continuous-time dynamics are

$$dS_t = S_t \sqrt{v_t} dW_t^S, \tag{13}$$

$$dv_t = \kappa_v(\theta_v - v_t) dt + \xi \sqrt{v_t} dW_t^v, \quad d\langle W^S, W^v \rangle_t = \rho dt. \tag{14}$$

Let $\tilde{v}_k$ be the Euler auxiliary variable and $v_k^+ = \max(\tilde{v}_k, 0)$. A fully specified full-truncation/log-Euler step is

$$\tilde{v}_{k+1} = \tilde{v}_k + \kappa_v(\theta_v - v_k^+) \Delta t + \xi \sqrt{v_k^+ \Delta t} Z_{k+1}^v, \tag{15}$$

$$\log S_{k+1} = \log S_k - \frac{1}{2} v_k^+ \Delta t + \sqrt{v_k^+ \Delta t} Z_{k+1}^S, \tag{16}$$

where

$$Z_{k+1}^S = \rho Z_{k+1}^v + \sqrt{1 - \rho^2} Z_{k+1}^\perp$$

and $Z^v, Z^\perp$ are independent standard normals. The auxiliary variable $\tilde{v}_{k+1}$ may be negative; non-negativity applies to the truncated variance v_{k+1}^+ used in the spot diffusion and as a network feature. This is the relevant property of full truncation, not literal non-negativity of the untruncated Euler state [Lord et al., 2010].

The Heston parameters used here violate the Feller condition:

$$2\kappa_v\theta_v = 0.16 < \xi^2 = 0.25.$$

Consequently, discretization checks with finer substeps or a higher-accuracy scheme are particularly important before treating numerical differences as model effects.

7.3 Parameters

The common parameters are

$$S_0 = K = 100, \quad T = 30/252, \quad n = 30, \quad \alpha = 0.95.$$

For GBM, $\sigma = 0.20$. For Heston,

$$v_0 = \theta_v = 0.04, \quad \kappa_v = 2.0, \quad \xi = 0.5, \quad \rho = -0.7.$$

The archived test results use 30,000 simulated paths. A confirmatory comparison would evaluate all strategies on the same held-out paths and estimate uncertainty in their differences by paired resampling.

7.4 Benchmarks and their interpretation

The main benchmark is the Black–Scholes delta

$$\Delta_{\text{BS}}(S_k, \tau_k) = \Phi\left(\frac{\log(S_k/K) + \frac{1}{2}\sigma^2\tau_k}{\sigma\sqrt{\tau_k}}\right),$$

with zero interest rate. It is rebalanced on the same grid and charged the same cost function as the neural policy.

This benchmark is standard but limited. Under discrete rebalancing, it is not known to minimize terminal CVaR, even when transaction costs are zero. Under positive costs, it is deliberately cost-blind. Therefore, a lower point estimate relative to this rule concerns only that particular comparison and does not imply global or near-global optimality. Relevant cost-aware benchmarks include a shrunk delta $\lambda\Delta_{\text{BS}}$, a no-trade band, less-frequent rebalancing, and an inventory-aware policy.

In the Heston experiment, the network observes v_k^+ , whereas the benchmark uses a fixed volatility of 20 percent. The comparison is therefore asymmetric in both information and model flexibility. It is retained only as a misspecified-benchmark exercise. A like-for-like Heston comparison requires at least a Black–Scholes delta with $\sigma_k = \sqrt{v_k^+}$, an optimized fixed volatility, a Heston delta, and network ablations with and without the variance feature.

7.5 Statistical interpretation

The experiment set reports one training realization per displayed policy and one finite test sample per comparison. No confidence intervals across training seeds are available. Accordingly:

- differences are point estimates, not significance statements;
- monotonicity in the displayed cost sweep refers to the five observed point estimates only;
- the difference between independently trained policies at the same hyperparameters cannot be labelled “within expected run-to-run variance” without repeated runs;
- confirmatory evidence requires multiple training seeds and paired bootstrap intervals on a held-out test sample.

8 Archived Exploratory Results

The tables in this section are retained from the earlier numerical record for provenance. They were produced from separate single training runs and are not presented as a multi-seed experiment. The current repository isolates these values under `results/legacy/`. No statistical significance claim is made from them.

8.1 Baseline GBM run with proportional costs

Table 3 reports one archived run at $\kappa = 0.01$. Under the $p_0 = 0$ convention, the neural policy has an estimated CVaR of 5.719, compared with 6.981 for the discretely rebalanced, cost-blind Black–Scholes delta. The absolute difference in this run is 1.262 loss units. The corresponding percentage is not emphasized because it changes under an additive cash shift.

Table 3: Archived baseline GBM run with proportional costs, $\kappa = 0.01$, and 30,000 test paths. Values are single-run point estimates under $p_0 = 0$.

	Unhedged	BS delta hedge	Neural policy
Mean loss	2.830	4.976	4.376
Standard deviation of loss	4.294	0.803	0.758
CVaR _{0.95}	15.367	6.981	5.719

The analytic Black–Scholes call value for the stated parameters is approximately 2.7524. Using the archived unhedged standard deviation, the Monte Carlo standard error of the reported unhedged mean is approximately $4.294/\sqrt{30,000} = 0.0248$. The difference $2.830 - 2.7524 = 0.0776$ is about 3.1 such standard errors. This discrepancy does not by itself identify an implementation error, but it prevents the archived mean from serving as a strong simulator-validation result. A regenerated test with fixed conventions and multiple independent seeds is required.

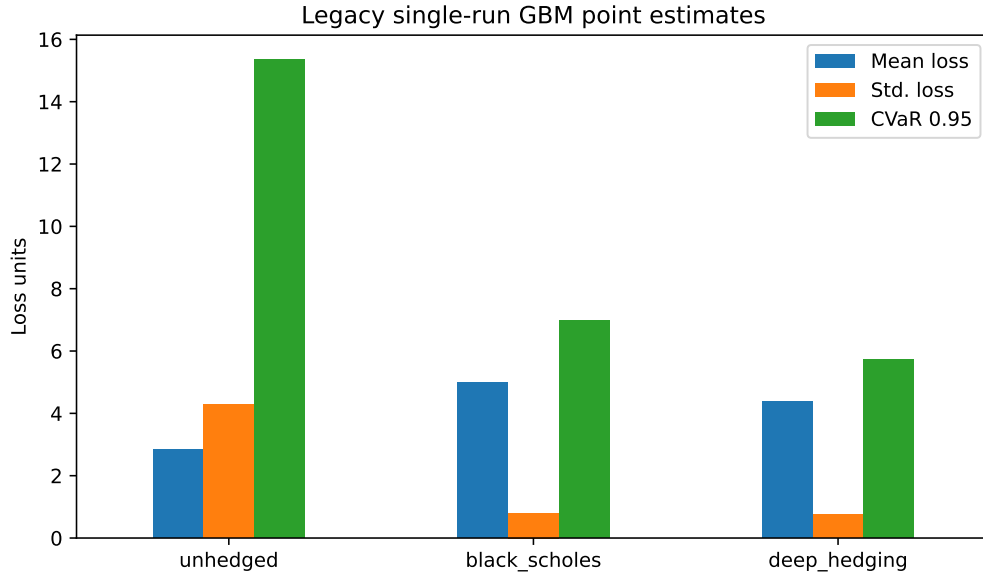


Figure 1: Archived single-run GBM summary statistics. The figure is descriptive and is not accompanied by training-seed or paired-bootstrap uncertainty.

8.2 Proportional-cost sweep

The archived sweep contains five independently fitted policies at $\kappa \in \{0, 0.005, 0.01, 0.02, 0.04\}$. Table 4 reports turnover and CVaR point estimates.

Table 4: Archived proportional-cost sweep. Each neural policy was fitted in a separate single run.

κ	Neural avg. $ \Delta\delta $	BS avg. $ \Delta\delta $	Neural CVaR	BS CVaR	BS–Neural
0.000	0.0728	0.0743	3.951	3.752	−0.199
0.005	0.0611	0.0743	5.019	5.329	0.310
0.010	0.0539	0.0743	5.946	6.981	1.035
0.020	0.0463	0.0743	7.260	10.358	3.098
0.040	0.0282	0.0743	9.769	17.203	7.434

The average absolute trade of the fitted policy decreases across the five reported cost levels, from 0.0728 to 0.0282. This is a descriptive pattern within the displayed runs. Because the policy does not observe its current inventory, the result is described only as lower trading intensity; it is not evidence of an optimal no-trade region.

The $\kappa = 0.01$ sweep value, 5.946, differs from the baseline value, 5.719, because the baseline and sweep entries were generated by separate training runs. The two values are retained for provenance. They are neither averaged nor selectively treated as the representative result, and their difference is not used to estimate run-to-run variability because only one observation from each setup is available.

At $\kappa = 0$, the neural policy has a CVaR point estimate 0.199 units above the discretized Black–Scholes rule in the archived run. At positive costs, its point estimate is below that cost-blind rule in the four displayed runs. These comparisons do not establish optimality or superiority to calibrated cost-aware benchmarks. In particular, a shrunk delta, a no-trade band, and reduced-frequency rebalancing may obtain part of the same turnover reduction without a neural policy.

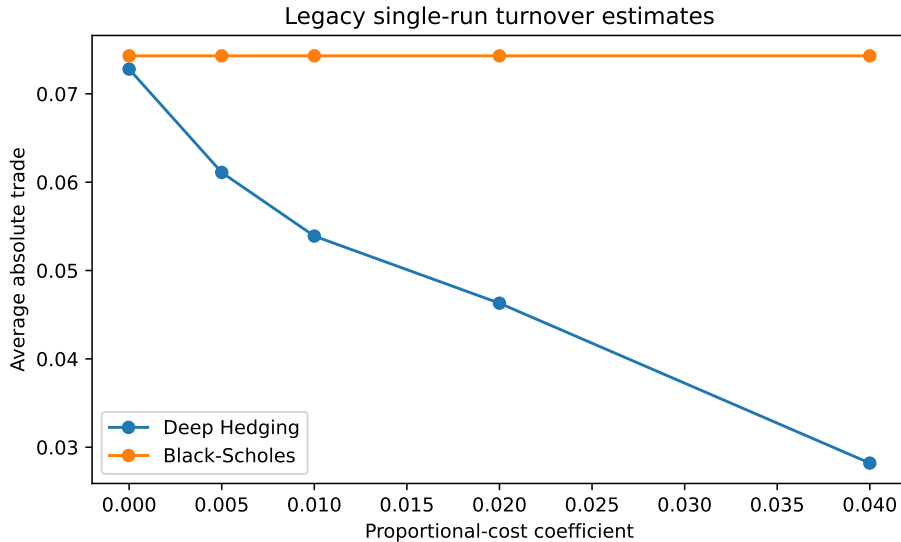


Figure 2: Archived single-run trading-intensity point estimates.

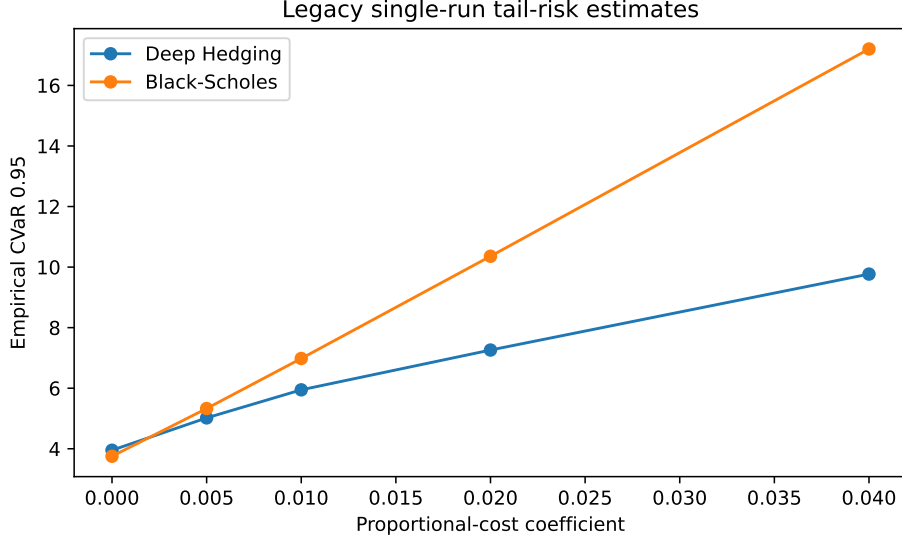


Figure 3: Archived single-run empirical-CVaR point estimates. The absence of uncertainty bands reflects the absence of repeated training runs and retained pathwise bootstrap records.

8.3 Quadratic transaction costs

For one archived quadratic-cost run with $\kappa = 0.05$, the neural policy has a CVaR point estimate of 5.769, compared with 7.633 for the discretized Black–Scholes delta. The absolute difference is 1.864 loss units under the same $p_0 = 0$ convention.

The archived proportional and quadratic calibrations are not matched. A comparison between cost functions would require a common calibration rule, for example matching expected total cost or a specified cost quantile under a fixed reference strategy. The present values should therefore not be used to rank the two cost specifications.

8.4 Heston comparison with an asymmetric benchmark

In the archived Heston run, the neural policy observes instantaneous truncated variance and has $\text{CVaR}_{0.95} = 6.006$. A Black–Scholes delta using a fixed 20-percent volatility has $\text{CVaR}_{0.95} = 7.661$. The difference of 1.655 is a single-run comparison against this specific fixed-volatility rule.

The experiment does not measure robustness to model misspecification. The network is trained and tested under the same Heston generator, while only the benchmark is misspecified and receives less information. The result therefore does not isolate the value of the variance feature and is not a like-for-like reproduction of the stock-plus-variance-swap experiment in [Bühler et al. \[2019\]](#).

A balanced Heston evaluation would include the same neural architecture with and without v_k^+ , an instantaneous-volatility Black–Scholes delta, an optimized fixed-volatility delta, a Heston delta or other model-consistent benchmark, and tests in which simulator parameters change between training and evaluation.

9 Validation Status and Confirmatory Plan

This section distinguishes checks that are implemented in the accompanying repository from analyses that remain planned. The distinction is important because an available script or benchmark is not the same as a reported confirmatory result.

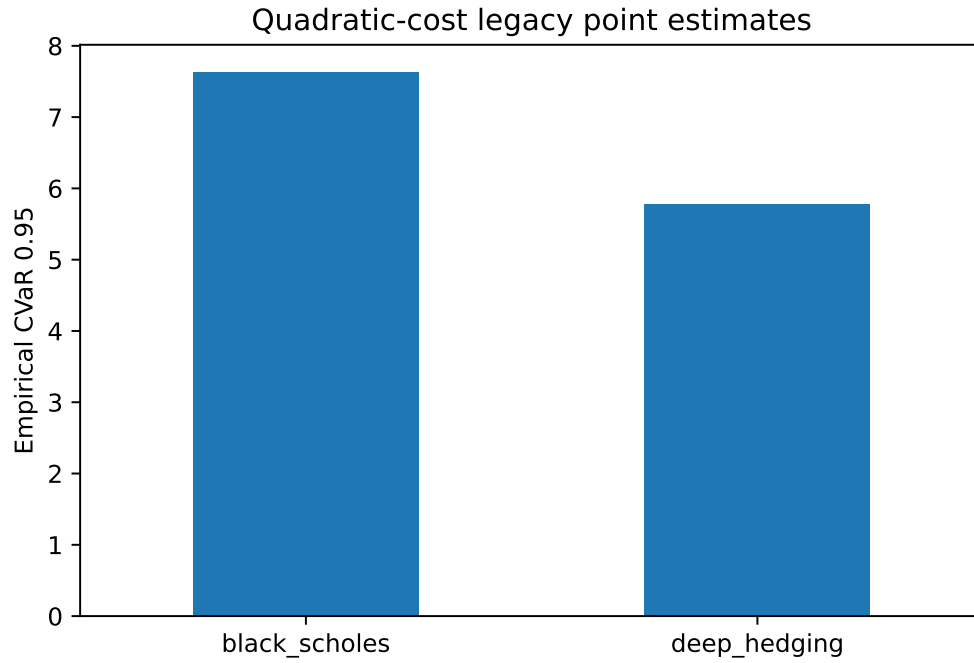


Figure 4: Archived quadratic-cost CVaR point estimates. A confirmatory table would also report total cost, turnover, VaR, and paired uncertainty intervals.

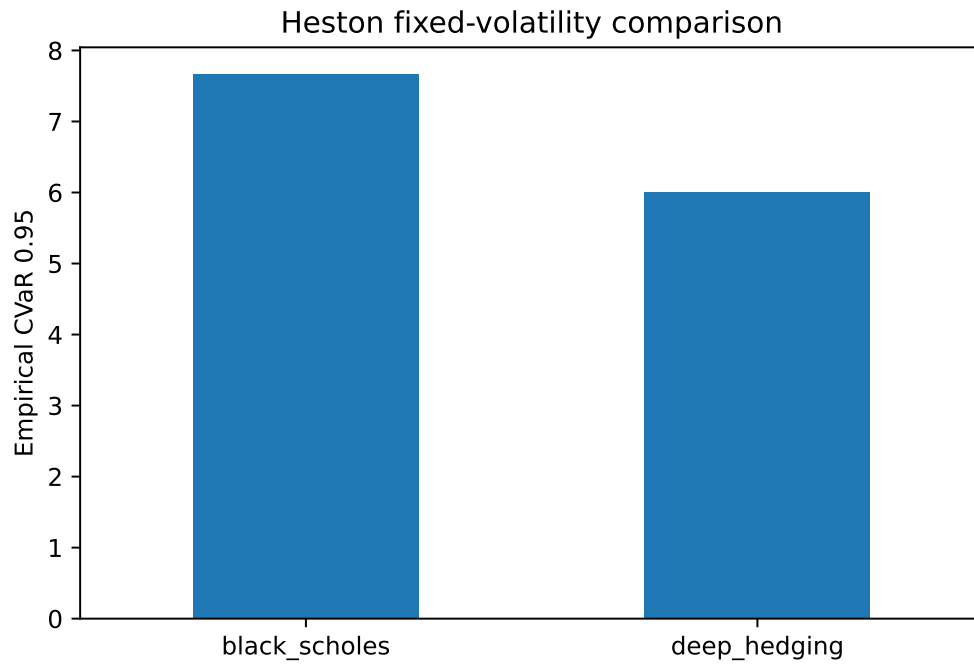


Figure 5: Archived Heston CVaR point estimates for the state-only policy and a fixed-volatility Black–Scholes delta. The comparison is asymmetric in both information and model specification.

9.1 Completed automated checks

In the reviewed release candidate, the automated test suite contains 14 tests covering empirical-CVaR boundary weights, cash translation, deterministic GBM simulation, a GBM martingale-mean check, Heston output constraints, basic Heston pricing bounds, transaction-cost indexing, network derivatives, and the inventory-aware backward pass. All 14 tests passed in the local validation run.

The standalone centered finite-difference validator reported maximum absolute errors of 3.208×10^{-10} for the feedforward-network backward pass and 5.410×10^{-10} for the quadratic pathwise loss gradient. These checks support the corresponding local derivative implementations. They do not yet constitute an independent automatic-differentiation verification of the full profiled empirical-CVaR training objective.

9.2 Implemented but not yet reported as confirmatory evidence

The repository implements an inventory-aware policy, a shrunk Black–Scholes delta, reduced-frequency rebalancing, a no-trade-band rule, validation-only benchmark calibration, multi-seed experiment plans, and paired-bootstrap utilities. The archived numerical tables in Section 8 predate that complete protocol. Consequently, the existence of these components is not presented as evidence that the associated comparisons have already been completed.

9.3 Required confirmatory design

A later confirmatory version would:

1. freeze the source commit, environment, configuration, random seeds, dtype, optimizer, and checkpoint-selection rule before final testing;
2. use multiple independent training seeds for every policy and cost level;
3. calibrate classical cost-aware benchmarks on validation paths only;
4. evaluate all policies on identical held-out test paths;
5. report paired pathwise bootstrap intervals together with variation across training seeds;
6. compare state-only and inventory-aware policies under otherwise matched settings;
7. equalize information sets in the Heston comparison and test parameter shifts between training and evaluation; and
8. regenerate every table and figure from machine-readable result files.

Until those steps are executed, the appropriate interpretation is exploratory and focused on implementation.

10 Limitations

The numerical evidence is limited by design and by the available archive. The reported policy values come from single training runs, so the manuscript does not estimate sensitivity to initialization, Monte Carlo batches, or checkpoint selection. The principal benchmark in the archived tables is cost-blind, while the policy is trained with costs in its objective. This asymmetry can make the observed differences larger than they would be against tuned cost-aware rules.

The state-only policy omits current inventory. Under transaction costs, two paths with the same market variables but different positions can have different economically relevant actions. The

fitted map can trade more conservatively, but it cannot represent a genuinely inventory-dependent no-trade region.

The positive proportional-cost levels include stress values that are high for many liquid equity applications. They are useful for exposing qualitative responses to costs, but they should not be interpreted as a universal market calibration.

The Heston comparison is asymmetric: the neural policy observes truncated instantaneous variance, the benchmark uses a fixed volatility, and training and test paths come from the same simulator. The reported difference therefore measures neither out-of-distribution robustness nor a model-consistent Heston hedging advantage.

Finally, all conclusions are conditional on the simulated GBM and Heston settings. The study omits jumps, market-impact feedback, latency, discrete contract sizes, funding asymmetries, time-varying liquidity, parameter uncertainty, and historical regime changes. These omissions are material for real deployment. The repository and manuscript are intended for research and reproducibility, not live trading.

11 Conclusion

This paper documents a compact NumPy implementation of neural option hedging under transaction costs. Its main value is transparency: the loss convention, empirical-CVaR boundary weighting, transaction-cost indices, network backward pass, and simulation assumptions are written explicitly and connected to executable tests.

The archived experiments support only a narrow descriptive observation. In the reported proportional-cost sweep, the fitted policy traded less as the cost coefficient increased. Its CVaR point estimate was below that of a cost-blind Black–Scholes delta in the four positive-cost runs and above it in the zero-cost run. Separate $\kappa = 0.01$ runs produced different neural-policy values, which illustrates why repeated training is necessary before making an inferential comparison.

No claim is made that the policy is optimal, statistically superior, robust to model misspecification, or a full reproduction of [Bühler et al. \[2019\]](#). A confirmatory study would require multi-seed training, validation-calibrated cost-aware benchmarks, paired uncertainty estimates, inventory-aware ablations, and balanced Heston comparisons. In its present form, the work is an auditable implementation study and a reproducible starting point for those experiments.

12 Author Contributions

Using the CRediT taxonomy, Alpha Kabinet Touré contributed conceptualization, methodology, software, validation, formal analysis, investigation, data curation, visualization, writing—original draft, and writing—review and editing. The machine-readable contribution record is maintained in `authorship/contributions.csv`.

Data and Code Availability

The accompanying repository contains the editable manuscript, configuration files, NumPy implementation, automated tests, and scripts for running experiments and generating outputs. No proprietary market data are distributed. The numerical values reproduced in [Section 8](#) are isolated under `results/legacy/` and are identified as archived point estimates rather than results regenerated by the current confirmatory pipeline.

Funding

No external funding was reported for this study.

Conflict of Interest

The author declares no conflict of interest.

References

- Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, 1999.
- Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.
- Hans Bühler, Lukas Gonon, Josef Teichmann, and Ben Wood. Deep hedging. *Quantitative Finance*, 19(8):1271–1291, 2019. doi: 10.1080/14697688.2019.1571683.
- Jay Cao, Jacky Chen, John Hull, and Zissis Poulos. Deep hedging of derivatives using reinforcement learning. *Journal of Financial Data Science*, 3(1):10–27, 2021.
- Hans Föllmer and Alexander Schied. Convex measures of risk and trading constraints. *Finance and Stochastics*, 6(4):429–447, 2002.
- Igor Halperin. Qlbs: Q-learner in the black–scholes(–merton) worlds. *Journal of Derivatives*, 28(1):99–122, 2020.
- Steven L. Heston. A closed-form solution for options with stochastic volatility. *Review of Financial Studies*, 6(2):327–343, 1993.
- Tim Hoggard, A. E. Whalley, and Paul Wilmott. Hedging option portfolios in the presence of transaction costs. *Advances in Futures and Options Research*, 7:21–35, 1994.
- James M. Hutchinson, Andrew W. Lo, and Tomaso Poggio. A nonparametric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, 49(3):851–889, 1994.
- Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.
- Petter N. Kolm and Gordon Ritter. Dynamic replication and hedging: A reinforcement learning approach. *Journal of Financial Data Science*, 1(1):159–171, 2019.
- Hayne E. Leland. Option pricing and replication with transactions costs. *Journal of Finance*, 40(5):1283–1301, 1985.
- Roger Lord, Remmert Koekkoek, and Dick van Dijk. A comparison of biased simulation schemes for stochastic volatility models. *Quantitative Finance*, 10(2):177–194, 2010.
- Robert C. Merton. Theory of rational option pricing. *Bell Journal of Economics and Management Science*, 4(1):141–183, 1973.
- R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2(3):21–41, 2000.
- A. E. Whalley and Paul Wilmott. An asymptotic analysis of an optimal hedging model for option pricing with transaction costs. *Mathematical Finance*, 7(3):307–324, 1997.

A Boundary Cases for the Position Gradient

Under free terminal liquidation cost, Equation (10) becomes

$$\begin{aligned}\frac{\partial L_T}{\partial \delta_0} &= -\Delta S_0 + c'_0(\delta_0) - c'_1(\delta_1 - \delta_0), \\ \frac{\partial L_T}{\partial \delta_k} &= -\Delta S_k + c'_k(\delta_k - \delta_{k-1}) - c'_{k+1}(\delta_{k+1} - \delta_k), \quad 0 < k < n-1, \\ \frac{\partial L_T}{\partial \delta_{n-1}} &= -\Delta S_{n-1} + c'_{n-1}(\delta_{n-1} - \delta_{n-2}).\end{aligned}$$

If terminal liquidation is charged, the final line additionally contains $-c'_n(-\delta_{n-1})$.

B Completed and Outstanding Gradient Checks

The current repository includes centered finite-difference checks for the feedforward-network backward pass, the quadratic pathwise loss gradient, and the inventory-aware backpropagation-through-time implementation. It also tests the S_{k+1} index in the subsequent transaction-cost term and the fractional empirical-CVaR boundary weight.

One stronger validation remains outstanding: a deterministic end-to-end comparison of the complete profiled empirical-CVaR parameter gradient with an independent automatic-differentiation implementation. Such a test should use a small fixed path array and include:

1. quadratic costs with terminal cost both zero and nonzero;
2. proportional costs with trades bounded away from zero;
3. an interior date where both c_k and c_{k+1} contribute;
4. a batch size for which $(1 - \alpha)B$ is non-integer; and
5. shared parameters evaluated at every trading date.

C Minimum Reproducibility Record

For each reported table row, the archived record should contain:

- source-code commit and environment lock file;
- configuration file and random seeds;
- simulator equations and substep count;
- network architecture, initialization, dtype, and input scaling;
- Adam hyperparameters including ε ;
- exact batch-CVaR routine and tie convention;
- training, validation, and test sizes;
- checkpoint-selection rule;
- raw per-path terminal losses, gains, costs, positions, and trades; and
- the script used to generate the final table and figure.