Deep hedging with low data usage and transaction costs

Gabriel Turinici
joint work with P. Brugière

CEREMADE Université Paris Dauphine - PSL, Paris, France

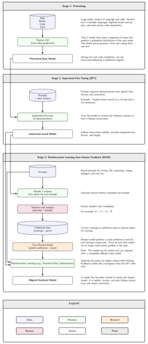
FAAI Conference, Iasi 20 oct 2025

Replication Example: Autonomous Driving

- **Situation:** Learn to drive by imitating human behavior.
- **What is cloned:** Human driving policy (steering, braking, etc.)
- **Used to clone:** Neural policy (CNN + RNN)
- **Dataset:** Logged sensor data (video, LiDAR, GPS, IMU)
- **Goal:** Generalize human-like driving to unseen situations.

Replication Example: Distilling Large Language Models

- **Situation:** Reproduce the behavior of a large model like GPT-4.
- **What is cloned:** Outputs (text responses) of the large model.
- **Used to clone:** Smaller neural model (e.g., GPT-2 or TinyLLaMA)
- **Dataset:** Synthetic queries + responses generated by the teacher model.
- **Goal:** Replicate output quality with fewer parameters and lower cost.



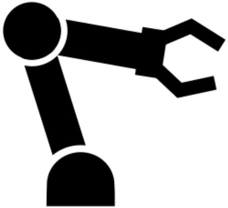
Source: wikipedia

Replication Example: Game AI

- **Situation:** Train an agent to play competitive games (e.g., StarCraft, Dota 2).
- **What is cloned:** Human expert gameplay (actions, timing, strategy).
- **Used to clone:** Neural policy trained via behavior cloning.
- **Dataset:** Logs of expert games (states + actions).
- **Goal:** Bootstrap a policy that can later be fine-tuned via reinforcement learning.

Replication Example: Robotic Manipulation

- **Situation:** Train a robot arm to manipulate or grasp objects.
- **What is cloned:** Expert physical behavior (e.g., grasping, opening doors).
- **Used to clone:** Neural policy trained on demonstrations.
- **Dataset:** Teleoperation data or kinesthetic teaching (sensor + video).
- **Goal:** Enable robots to generalize physical skills to new objects or settings.



Replication Example: Financial Deep Hedging

- **Situation:** Hedge a derivative under realistic market conditions (frictions, incomplete markets). Derivative use case : gold prices, denoted S_0 are now high. A gold miner wants to invest but will GO BANKRUPT if price a year from now drops below 3500\$/oz (today $S_0 > 4000$\$/oz.). Want a contract to pay 10M\$ if $S_T < 3500$ ($T = 1$), nothing otherwise. This is a 'digital (or binary) put'.
The seller bank cannot just sell short the gold, if gold rises again the loss for the bank is huge. Needs to sell the right amount of gold Δ_t to reach exactly the final value $G = g(S_T)$.

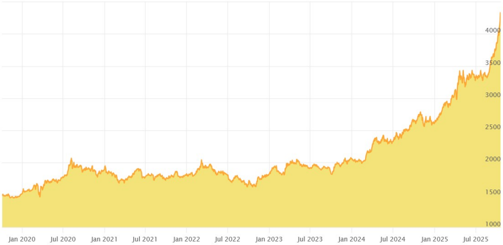


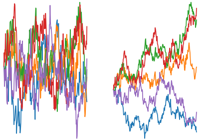
Figure: Gold prices 2000-2025

The Challenge: Hedging Options in the Real World

- **What is an Option?** A financial contract (liability) whose value G depends on a future asset price, e.g. $G = g(S_T)$. The seller (trader) must protect themselves NOW against potential loss; this protection is called **hedging** or replication; it works by buying quantity Δ_t of underlying assets S_t. Similar to insurance companies but less diversification possible.
- how to find Δ_t ? If S_t deterministic $\Delta = \partial_S g(S)$; otherwise need a model for the uncertainty process
- **Theory:** for many models like Black-Scholes (1973) $\boxed{dS_t/S_t = \mu\, dt + \sigma\, dW_t}$ under hypothesis (continuous trading, no fees) **perfect hedging** (zero risk) can be reached. The strategy Π_t containing Δ_t parts of S_t + cash (auto-financed) is such that $\Pi_T = G$ in every state of the world, not relying on the law of large numbers like insurance.

Data flow in a classical approach

Historical data



Model list

Model	**Equation**	**Parameters**
Black-Scholes (B&S)	$dS_t = \mu S_t\,dt + \sigma S_t\,dW_t$	S_t, σ, r, T, K
Merton Jump-Diffusion	$dS_t = (\mu - \lambda k) S_t\,dt + \sigma S_t\,dW_t + S_{t-}(J-1)\,dN_t$	$S_t, \sigma, \lambda, J, r, T, K$
SABR	$\begin{cases} dF_t = \sigma_t F_t^{\beta}\,dW_t^1 \\ d\sigma_t = \nu \sigma_t\,dW_t^2 \end{cases}$	$F_t, \sigma_t, \beta, \nu, \rho$
Heston	$\begin{cases} dS_t = \mu S_t\,dt + \sqrt{v_t} S_t\,dW_t^S \\ dv_t = \kappa(\theta - v_t)\,dt + \xi\sqrt{v_t}\,dW_t^v \end{cases}$	$S_t, v_t, \kappa, \theta, \xi, \rho$
Vasicek	$dr_t = a(b - r_t)\,dt + \sigma\,dW_t$	r_t, a, b, σ
CIR	$dr_t = a(b - r_t)\,dt + \sigma\sqrt{r_t}\,dW_t$	r_t, a, b, σ

model choice

calibration

calibration

model + parameters

analytic formula

Hedging formula

Practical limitations of Classic Models

- **Reality Check:** In practice, perfect hedging is impossible because: trading occurs only in **discrete time** (e.g., once a day, once an hour), moreover every trade incurs **transaction costs** (α), reducing profits.
- **The Goal:** We need a flexible strategy that minimizes the uncertainty (risk) associated with the final cost of hedging, especially when transaction costs are high.
- **Discretization Risk:** When trading is discrete, the instantaneous "sensitivity" Δ_t and $\Gamma_t = \partial_S \Delta_t$ calculated by models like Black-Scholes (B&S) becomes inaccurate, leading to hedge errors.
- **The Infinite Cost Problem:** If B&S is used for frequent hedging ($\Delta t \to 0$) in the presence of transaction costs ($\alpha > 0$), the cumulative costs approach **infinity**.
- **The Leland Adjustment:** Leland (1985) introduced a mathematical fix by adjusting the asset's volatility (ν^*) based on the frequency and cost (α) to keep costs finite, but this is still a model-dependent solution while in practice model parameters are unknown.

Deep Hedging: Learning the Optimal Action

- **The Deep Hedging Advantage:** Deep Learning offers a **model-free** alternative, learning the optimal action directly from the data, adapting inherently to discrete time and costs.

- Deep Hedging (cf. H Buehler, L Gonon, J Teichmann, B Wood 2019) uses a **Neural Network (NN)** to determine the optimal number of shares Δ_t^{DH} to hold at each trading point.

- 'classical' Deep Hedging : mostly a reinforcement learning approach to maximize average final gain $\mathbb{E}[gain]$ of the option seller or $\mathbb{E}[utility(gain)]$); works well in several cases but may be slow to converge in some situations, needs many training trajectories $\sim 10^6$(!!!), but data 100Y old is not relevant !!!

Deep Hedging: our approach

• tailored optimization criterion: Instead of aiming for a theoretical perfect price (like in B&S), **our NN** is trained to minimize the **Standard Deviation** (variability or risk) of the final profit or loss (Z_T) incurred by the trader. It uses statistical metrics on the distribution of the gains not only the average.

• No complicated reinforcement learning, just plain function optimization
$\Delta = NN(time\ left, price/strike, previous\ \Delta)$

• Model-Free: The NN learns the best strategy directly from simulated asset paths generated under the **real-world probability ($\mathbb{P}$)**, not under complex, often fictional, risk-neutral measures ($\mathbb{Q}$). This eliminates the need of a model !

Neural Network Machinery

- **Architecture:** standard feed-forward neural network (ReLU activated linear layers).
- **Design Philosophy:** The architecture is intentionally **simple and lightweight** to demonstrate that powerful performance doesn't require massive computational resources.
- **Output:** The final layer outputs the desired **hedge position** (Δ_t), which is the number of assets the trader should hold at time t.

Inputs (3 features)

Moneyness (K/X_t)   Time Left ($T-t$)   Pre-existing Position ($\Delta_{t-\Delta t}$)

$\longrightarrow$ Linear ($3 \to 64$) $\longrightarrow$ ReLU $\longrightarrow$ Linear ($64 \to 32$) $\longrightarrow$ ReLU $\longrightarrow$ Linear ($32 \to 1$) $\longrightarrow$

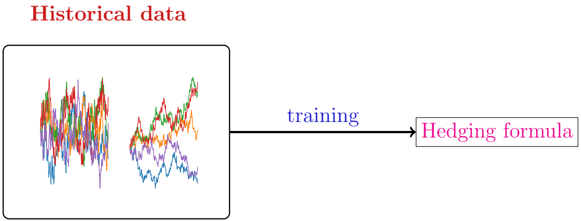
Output (1 feature)

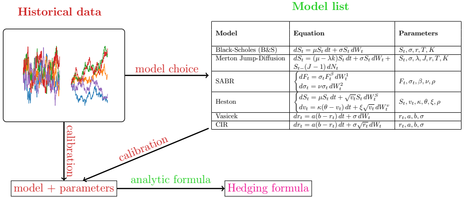
New Hedge Position (Δ_t)

Training

• **Training procedure:** a batch of trajectories are selected at each step. NN computes the delta and final value for each one and is updated to minimize the loss at final time. Same NN model is used at all times in a training step.

• **Unusual setting:** several NN calls per trajectory, the loss is not in average form but is a statistical metric of the batch losses





Model	Equation	Parameters
Black-Scholes (B&S)	$dS_t = \mu S_t\,dt + \sigma S_t\,dW_t$	S_t, σ, r, T, K
Merton Jump-Diffusion	$dS_t = (\mu - \lambda k) S_t\,dt + \sigma S_t\,dW_t + S_{t-}(J-1)\,dN_t$	$S_t, \sigma, \lambda, J, r, T, K$
SABR	$\begin{cases} dF_t = \sigma_t F_t^\beta\,dW_t^1 \\ d\sigma_t = \nu\sigma_t\,dW_t^2 \end{cases}$	$F_t, \sigma_t, \beta, \nu, \rho$
Heston	$\begin{cases} dS_t = \mu S_t\,dt + \sqrt{v_t} S_t\,dW_t^S \\ dv_t = \kappa(\theta - v_t)\,dt + \xi\sqrt{v_t}\,dW_t^v \end{cases}$	$S_t, v_t, \kappa, \theta, \xi, \rho$
Vasicek	$dr_t = a(b - r_t)\,dt + \sigma\,dW_t$	r_t, a, b, σ
CIR	$dr_t = a(b - r_t)\,dt + \sigma\sqrt{r_t}\,dW_t$	r_t, a, b, σ

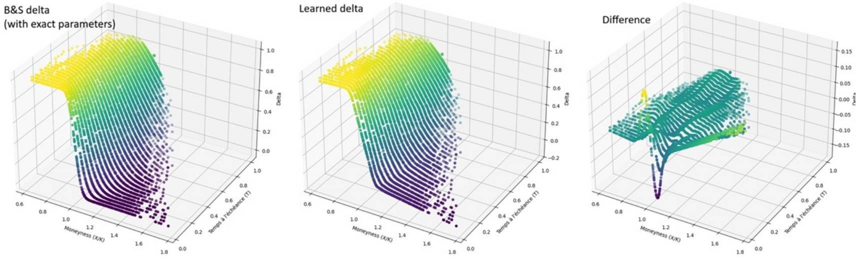
Figure: **left:** Data flow in our model : the NN train directly on historical data without any stochastic model; **right:** recall of classical model

Empirical results

• **Minimal Data Requirement:** Unlike previous deep hedging studies that required 10^5 or 10^6 asset paths, this research shows that satisfactory performance can be achieved using **as few as 256 simulated trajectories** for training.

• **Overlapping Sequences:** The model's robustness was tested using an even smaller dataset: 256 overlapping sequences of length 30 derived from a single sequence of just 285 observation points.

Example of result: left B& S delta **with knowledge of model**. center: our result, right: difference



Conclusion and Future Directions

- **Success in Simplified Markets:** The neural network successfully finds an optimal, risk-minimizing hedge strategy in the Geometric Brownian Motion environment, superior to both Black & Scholes and Leland models, despite minimal training data.
- **Implication for Real Markets:** This ability to calibrate on limited data paves the way for **practical use on real-time financial series**
- **Practical Implementation:** The ability to train effectively using only a few asset paths, or even derived overlapping sequences, suggests that Deep Hedging can be applied using **raw market data alone** and when large volumes of historical data for a specific option/maturity may not be available.
- **Current Limitation:** When applied to real S&P 500 daily data (which features changing volatility regimes), the simple NN failed, producing unrealistic deltas.
- **Future Work:** To address real-market complexity, future neural network development should incorporate **additional state variables** (inputs) such as market implied volatility or realized volatility over a historical window, enabling the model to adapt to volatility regime changes.

References I

F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81: 637–654, 1973.

P. Brugiere and G. Turinici. Deep learning of value at risk through generative neural network models: The case of the variational auto encoder. *Methods X*, 10: 102192, 2023.

P. Brugiere and G. Turinici. Model-Free Deep Hedging with Transaction Costs and Light Data Requirements *ArXiv*, 2505.22836, 2025 `https://arxiv.org/abs/2505.22836`.

H. Buehler, B. Mohan, B. Wood, and J. Teichmann. Deep hedging: from theory to practice, 2018. `https://arxiv.org/abs/1802.03042`.

H. Buehler, L. Gonon, J. Teichmann, and B. Wood. Deep hedging. *Quantitative Finance*, 19 (8): 1271–1291, 2019.

J. Cao, J. Chen, J. Hull, and Z. Poulos. Deep hedging of derivatives using reinforcement learning, 2019. `https://arxiv.org/abs/2103.16409`.

H. Föllmer and P. Leukert. Efficient hedging: Cost versus shortfall risk. *Finance and Stochastics*, 4: 117–146, 2000.

M. Hirano, K. Minami, and K. Imajo. Adversarial deep hedging: Learning to hedge without price process modeling, 2023. `https://arxiv.org/pdf/2307.13217`.

P.N. Kolm and G. Ritter. Dynamic replication and hedging: A reinforcement learning approach. *The Journal of Financial Data Science*, 1: 159–171, 2019.

H.E. Leland. Option pricing and replication with transactions costs. *The Journal of Finance*, 40 (5), 1985.

O. Mikkilä and J. Kanniainen. Empirical deep hedging. *Quantitative Finance*, 23 (1): 111–122, 2022. doi: 10.1080/14697688.2022.2136037.

H.M. Soner, S.E. Shreve, and J. Cvitanic. There is no nontrivial hedging portfolio for option pricing with transaction costs. *The Annals of Applied Probability*, 5 (2): 327–355, 1995.