(Deep) Learning to Trade II

Visiting Prof. Hans Buehler

hans.buehler@maths.ox.ac.uk

University of Oxford, Institute for Mathematics, 2026

V1.1

http://deep-hedging.com

Recap

- Notation
  - Everything is in discrete time → markets not complete.
  - We call s_t the **state** of the market. It represents all known information.
  - The market is observed under the *statistical* measure P.
  - Any observable random variable R_t can be written as $R(s_t)$.

- Trading environment
  - At any point t we observe **book values** $H_t^i \equiv H^i\,(s_t)$ of n_t hedging instruments.
  - The book value of our existing portfolio is $Z_t \equiv Z(s_t)$.

- Trading cost
  - Trading $a \in \mathrm{R}^n$ units of H at t will **cost** $c_t(a)$ in excess of book value H_t.
  - The function c is normalized to $c_t(0) = 0$, non-negative, and convex.
  - Trading is limited to where $c < \infty$.

Vanilla Deep Hedging [1]

- With *Statistical Hedging*, we looked at the returns $dZ_t = Z_{t+dt} - Z_t$.
- Now do the other extreme:
  - Assume T is such that all instruments in our portfolio Z and any relevant hedging instruments are expired.
  - Valuation of Z_T and H_T is trivial, and *model-independent*: it is the sum of all cashflows until T.
  - Define

$$DH_{t:T} \coloneqq H_T - H_t \in R^{n_t}$$

Vanilla Deep Hedging

- Instead of a single action a, we will now solve for a **policy** a which is a function of the state, i.e. at any time t the hedging action is $a_t \equiv a(s_t)$.
- The gains of trading a given a current position Z are

$$G(Z, a) \coloneqq Z_T + \sum_{t=0}^{T-1} a_t' D H_{t:T} - c_t(a_t)$$

Vanilla Deep Hedging

- Let U be a **monetary utility**. Then the **Vanilla Deep Hedging** problem is given as

$$U^*(Z) \coloneqq \sup_a : U\left(Z_T + \sum_{t=0}^{T-1} a_t' DH_{t:T} - c_t(a_t)\right)$$

- The main difference to our previous formulation is that a is now a *function* which takes the current state and returns the next hedge.

Vanilla Deep Hedging

Marginal Pricing – *the price of a derivative is the cost of its hedge*

- We wish to sell a derivative D to our client while we already have a position Z.
- The marginal price of selling D in the presence of Z is given as

$$p(D) \coloneqq U^*(Z) - U^*(Z - D)$$

- It represents the **minimal price** we should charge to not be worse off vs our existing position Z.
- It satisfies the invariance condition

$$U^*\big(Z - D + p(D)\big) = U^*(Z)$$

Implementation

Vanilla Deep Hedging

- To find

$$U^*(Z) \coloneqq \sup_a : U\left(Z_T + \sum_{t=0}^{T-1} a_t' DH_{t:T} - c_t(a_t)\right)$$

we are going to simulate N paths ω and solve the Monte Carlo **Periodic Policy Search** Reinforcement Learning problem:

$$U^*(Z) \coloneqq \sup_{a,y \in R} : \frac{1}{N} \sum_{\omega=1}^{N} u\left(y + Z_T + \sum_{t=0}^{T-1} a_t' DH_{t:T} - c_t(a_t)\right) - y$$

- We pre-compute the path-wise Z_T and $dH_{t;T}$ ahead of training.
- This can usually be done with existing Monte-Carlo interfaces for our products.

Vanilla Deep Hedging

- Architecture
  - Compress current market state into a recurrent hidden “history” state h_t.
  - Decode the relevant action a_t^i as a function of the history h_t and current stae.

- Networks or learned matrices are written in **bold**.

Vanilla Deep Hedging

Market State Embedding

- We trade different options in every step:
- For simplicity assume these are equity options, each identified by $w_t^i = (k_t^i, \tau_t^i, c_t^i)\ i = 1, \dots, n_t$ given in terms of
  - relative strikes k_t^i,
  - time to expiries τ_t^i, and
  - call/put indicators c_t^i.
- At each time t we observe a general market state m_t and option features $f_t^1, \ \dots, f_t^{n_t}$ such as mid-price, spread, trade volume etc.

Vanilla Deep Hedging

Market State Embedding

- The order of the options does not matter. We therefore embed the option features using a **set-invariant** architecture, e.g. Deep Sets [1]:

$$e_t := \tanh \boldsymbol{E}\left(m_t; \frac{1}{n_t}\sum_{k=1}^{n_t} \boldsymbol{F}\left(m_t; f_t^k\right)\right) \in (-1,+1)^{n_e}$$

  with networks $\boldsymbol{E}, \boldsymbol{F}$.
- Transformers are also set-invariant.
- Note that such an embedding can be computed in parallel in t.

[1] Deep Sets, Zaheer et al, 2017, https://arxiv.org/abs/1703.06114

Vanilla Deep Hedging

History State

- Once we have obtained a sequence e_t we can employ a time series model to generate our recursive "hidden" history state h, e.g. with selective SSM-like [1] models:

$$h_t := \boldsymbol{C}(e_t)h_{t-1} + \boldsymbol{B}(e_t, z_t)$$

Neural CDE [2]:

$$h_t = (1 - \boldsymbol{C}(e_t)de_t)h_{t-1}$$

- Implemented via compiled forward scan.
- Usually forced into unit space by applying tanh.

[1] Mamba: Linear-Time Sequence Modeling with Selective State Spaces, Gu et al 2024, https://arxiv.org/abs/2312.00752
[2] Neural Controlled Differential Equations for Irregular Time Series, Kidger et al 2020, https://arxiv.org/abs/2005.08926

Vanilla Deep Hedging

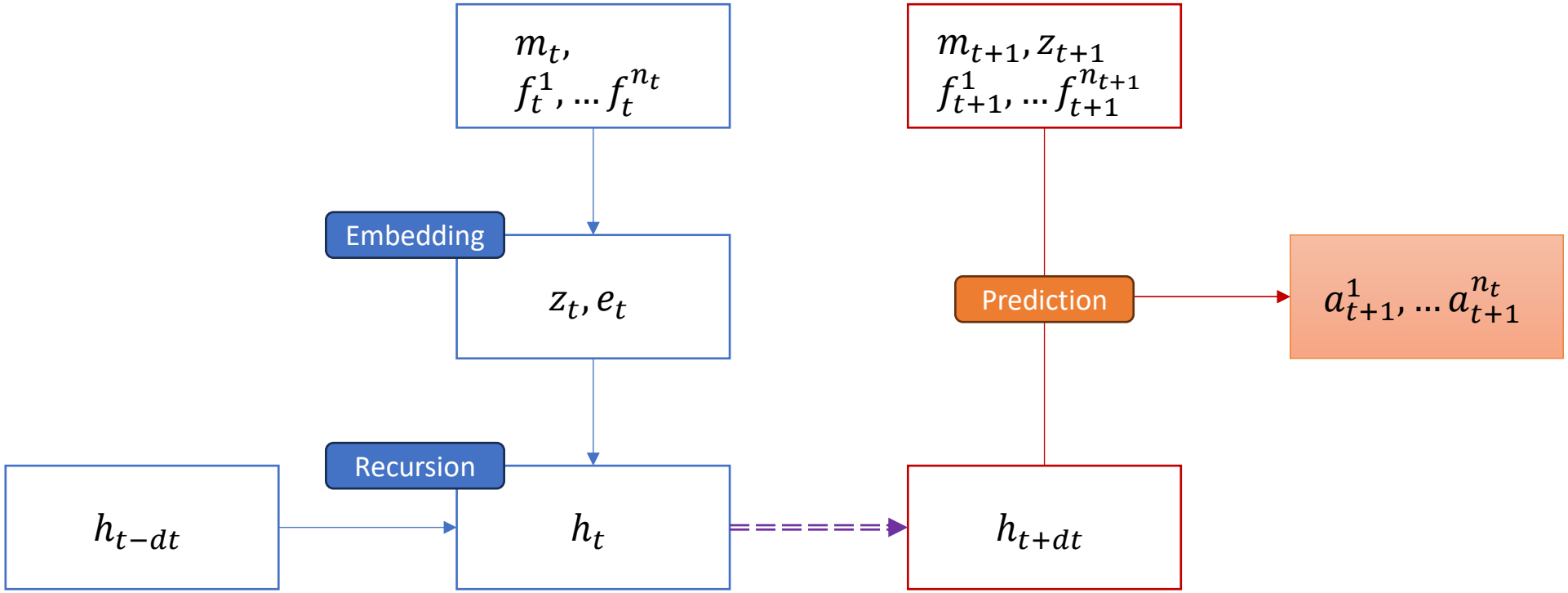
Portfolio State Encoding

- Many OTC payoffs have complicated state conditions on the past
  - Barriers depending on past KO/KI states.
  - Cliquets depend on past reset dates.

- *Should* be able to learn these.
  - However, we only look at terminal values of Z_T.
  - Identifying e.g. a monthly barrier breach requires noisy state detection, with possibly very long look back windows.
  - Simply add a "portfolio state" z_t to which captures portfolio-specific events.

- Decode into $a_t^i = \boldsymbol{A}(w_t^i; f_t^i, m_t, h_t, z_t)$.

Learning to Trade: Action



Vanilla Deep Hedging

Periodic Policy Search

- Solve

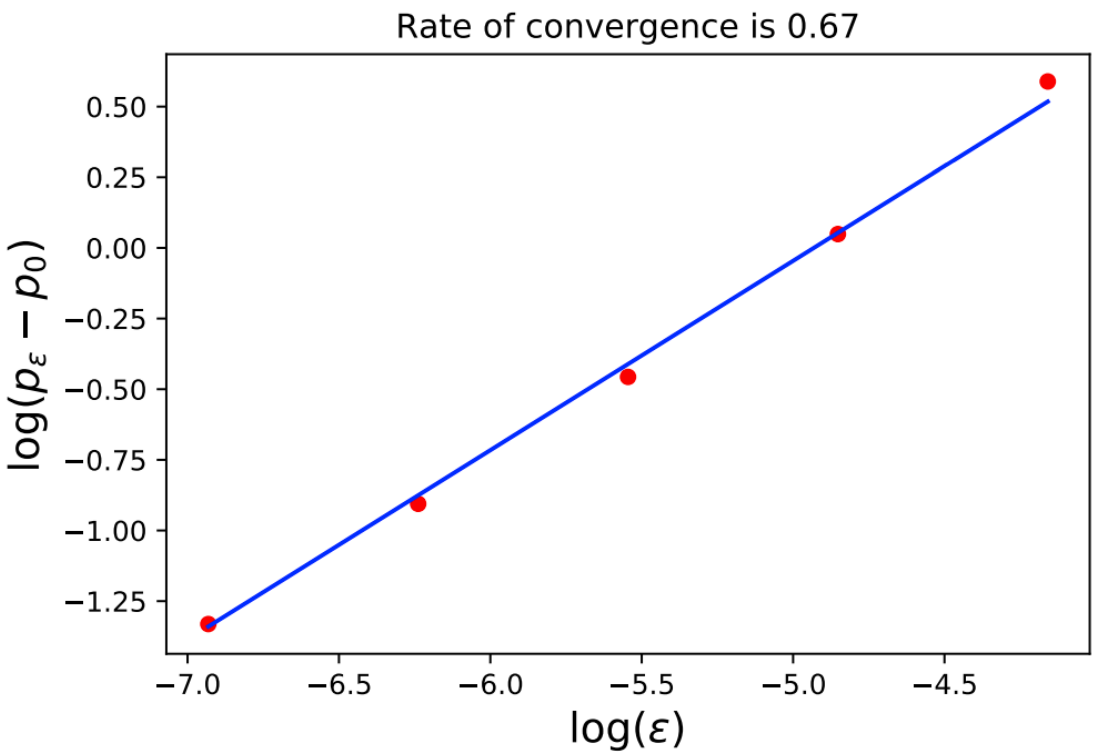
$$U^*(Z) \equiv \max_{\theta, y \in R} : \frac{1}{N} \sum_{\omega=1}^{N} u\left(y + Z_T + \sum_{t=0}^{T-1} a_t' DH_{t:T} - c_t(a_t) \right) - y$$

  where θ represents the weights of all networks involved.

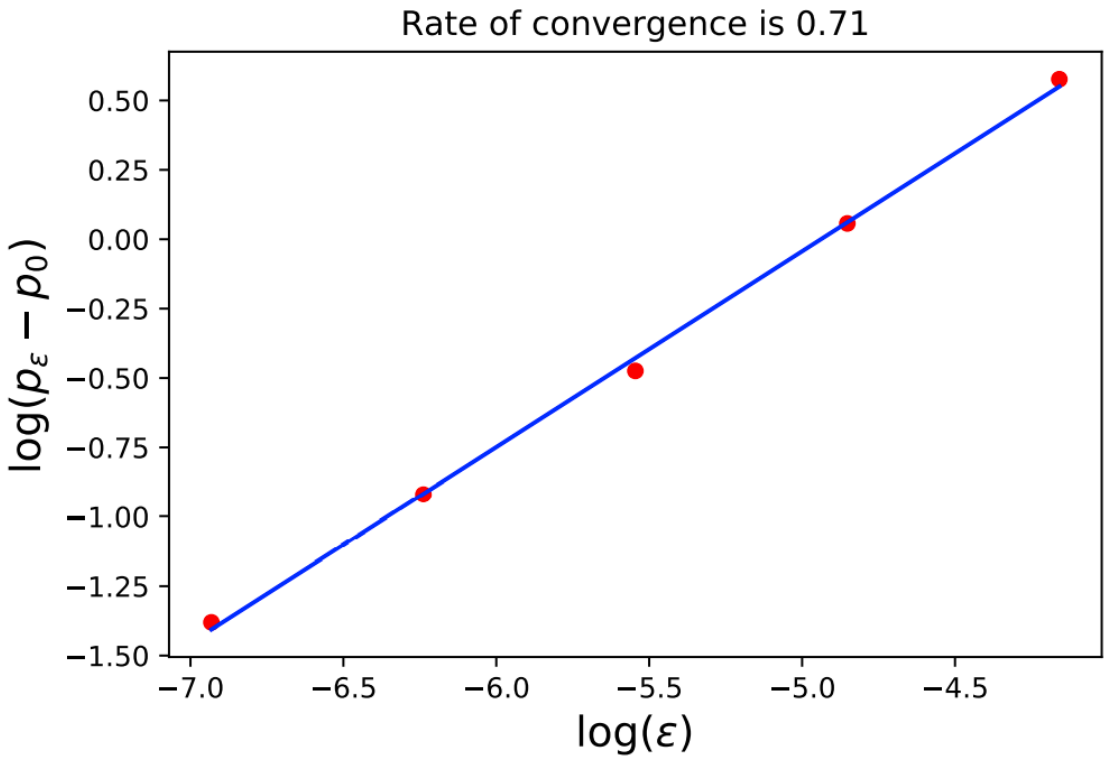
- *Research opportunity on the optimal embedding and time series approach.*

Vanilla Deep Hedging

- Test vs cases where we know or guess the theoretical answer [1]



Black-Scholes model price asymptotics.

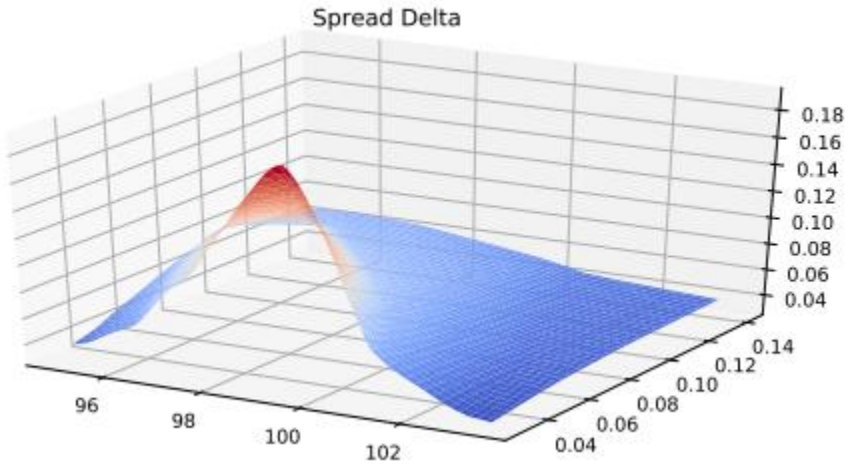


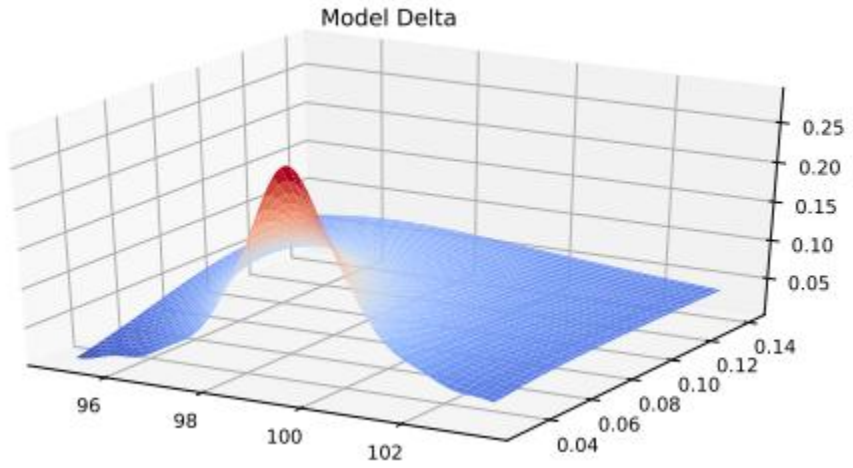
Heston model price asymptotics

[1] Deep Hedging, Buehler et all 2018, https://arxiv.org/pdf/1802.03042.pdf

Vanilla Deep Hedging

- Delta of a call spread in Black & Scholes [1]





[1] Deep Hedging, Buehler et all 2018, https://arxiv.org/pdf/1802.03042.pdf

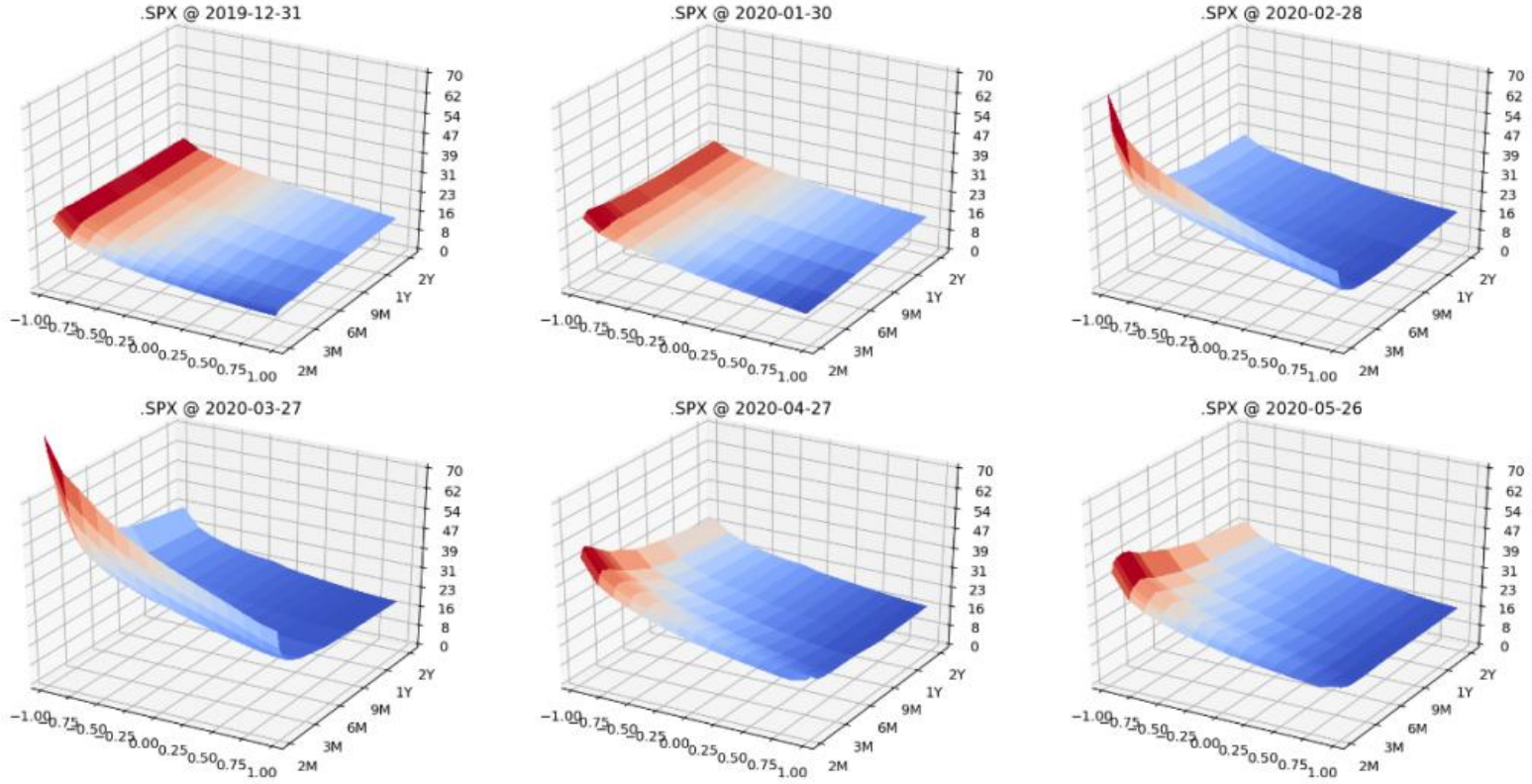
Market Simulation

- Ideally simulate against purely historic data.
- Lack of data: for example, S&P 500 index spot and option prices.
  - 10Y of data ~ 2500 data points
  - Typically index option surface has 100's of options

→Market Simulator

- Theoretical open question: does training against a simulator, which itself is trained only on a fixed, small data set outperform direct training on the fixed, small data set?
  - Intuitively: yes → just like classic quant finance models actually work quite well.
  - Statistically: why would that work …?

Market Simulation



- Our approach relies on training under full market data as opposed to classic “interpolation” derivatives models with ~2 factors.

Market Simulation

- Observe in every historical sample different tradable options w_t^i in with features f_t^i for $i = 1, \dots, N_t$ and a general market state m_t.
  - Embed "the market" into e_t using a set-invariant embedding.
  - Implement a hidden **stochastic** state process h_t.
  - Decode into arbitrage-free option prices using DLV/SANOS.

- Main difference: for market generators we are after *generative* models for h_t which generate distributions, not predictions.

Market Simulation

- Basic idea: use noise dW_t and write

$$h_t = \boldsymbol{N}(h_{t-dt}, m_t; dW_t)$$

- SDE approximation which looks like Ornstein-Uhlenbeck:

$$h_t = (1 - \boldsymbol{K}(m_t)dt)h_{t-1} + \boldsymbol{\Sigma}'\boldsymbol{\Sigma}(h_{t-1}, m_t)dW_t$$

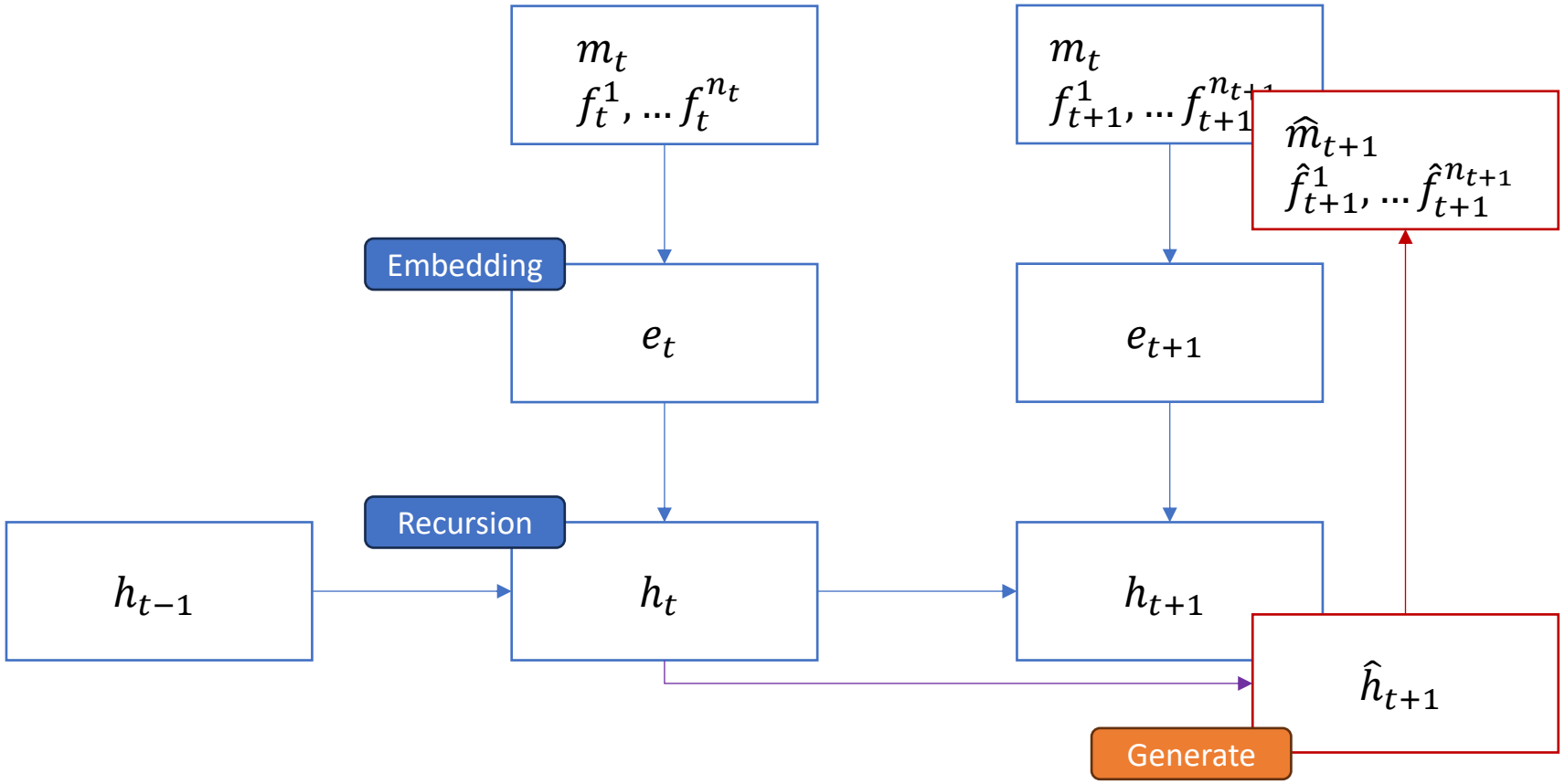
- How to assess quality of fit, since conditional properties are important → Conditional Wasserstein distance; Signatures; etc
- *No standardized architecture in the literature at this point – lots of ideas.*

Market Simulation

- There are several machine learning methods, all discussed in various papers
  - PCA [1]
  - Autoencoders [2], Variational Autoencoders [2]
  - Multi-asset versions [3]
  - SDE methods [4]
  - Neural SDEs [5]
- *Plenty of opportunity for research.*

[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844
[2] Deep Hedging: Learning to Simulate Equity Option Markets, Wiese et al 2020 https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3470756
[3] Multi-Asset Spot and Option Market Simulation, Wiese et al 2021 https://arxiv.org/abs/2112.06823
[4] Arbitrage-free neural-SDE market models, Cohen et al, 2021, https://arxiv.org/abs/2105.11053
[5] Neural SDEs as Infinite-Dimensional GANs, Kidger et al, 2021, https://arxiv.org/abs/2102.03657

Learning to Trade: Market Simulation



Static Arbitrage

Static Arbitrage

- We map the simulated h_t to option prices via a map

$$w_t^i, h_t \longmapsto H_t^i$$

- This map must prevent *static arbitrage* opportunities i.e linear combinations $a'(H_T - H_t) - c_t(a) \geq 0$ with a non-zero probability of generating a positive return.
- If this happens we can make money with **no risk** → Deep Hedging will try to take a maximum position, regardless of risk-aversion → numerics will explode.

Static Arbitrage

- Static Arbitrage can happen due to data: e.g. some downside puts on S&P never paid out, e.g. the steepest one-day drop in the S&P 500 was Oct 19, 1987 with -20.1%.
- That does not mean selling 0DTE 21% puts is “free money”.
- Need to remove instruments which individually have static arbitrage. Simple pre-data cleaning exercise.

Static Arbitrage

- Classic European option static arbitrage is linked to trading option combinations such that the P&L of any payoff is non-negative and can be positive.

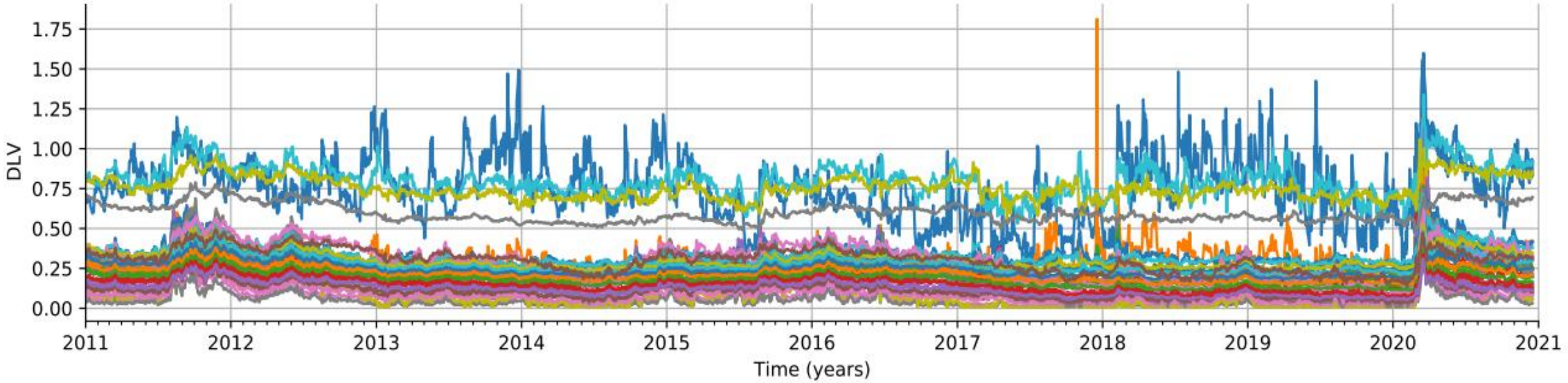
- **Discrete Local Volatility** "DLV" [1] is a numerically robust form of local volatility.
  - Bijection between strictly arbitrage-free call prices C and DLV's σ.
  - Positivity of σ is sufficient to define a strictly arbitrage-free option surface.
  - Our recent paper [2] **SANOS** adds smooth interpolation to essentially the same method.

→ Decode h_t into statically arbitrage-free prices H_t using DLV/SANOS.

[1] Discrete Local Volatility for Large Time Steps (Extended Version), Buehler et al 2015, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2642630
[2] SANOS – Smooth Arbitrage-Free Non parametric Option Surfaces, Buehler et al 2026, https://arxiv.org/abs/2601.11209

Static Arbitrage

Simulated DLV's



[1] graph from Multi-Asset Spot and Option Market Simulation, Wiese et al 2021, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3980817

Statistical Arbitrage

Statistical Arbitrage

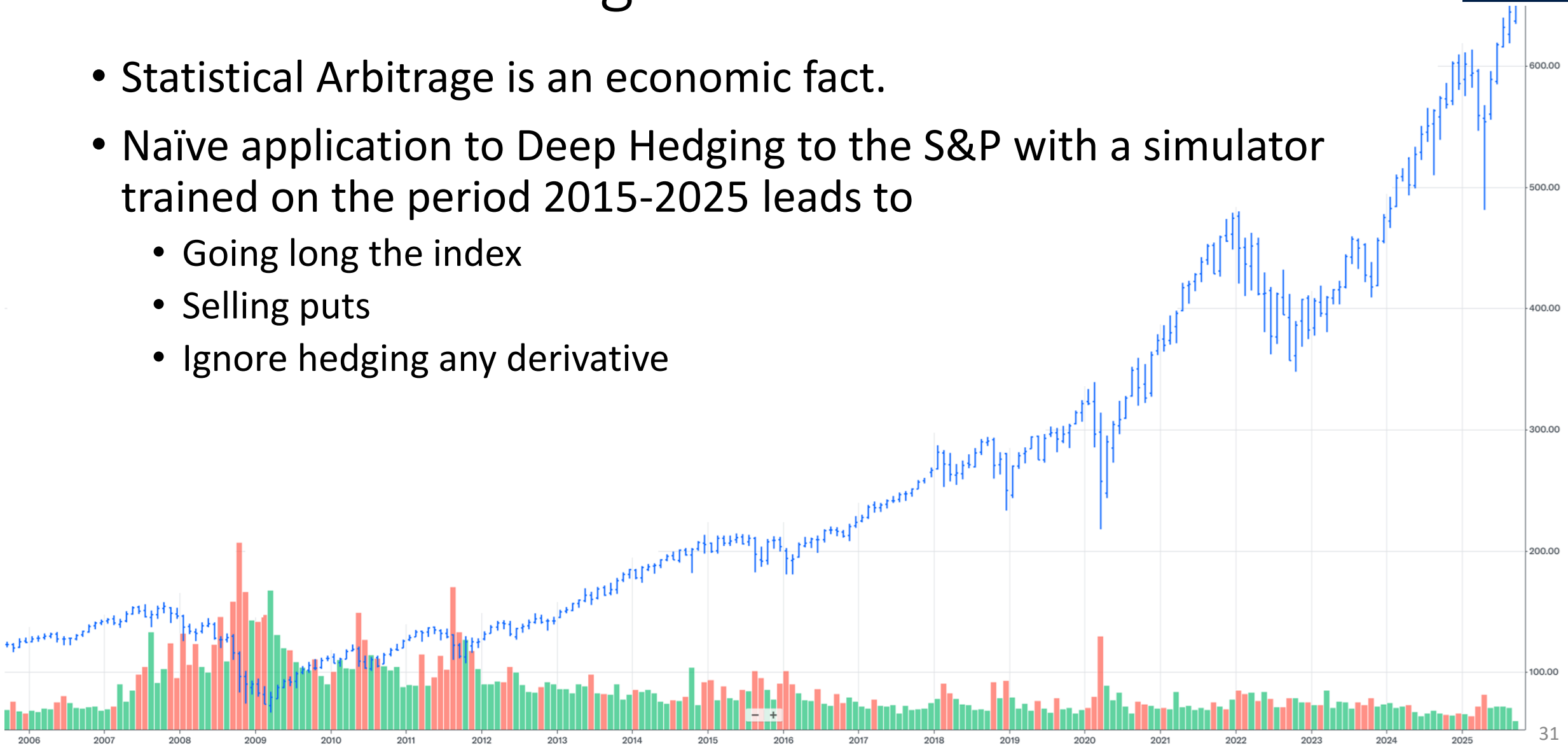
- We have earlier defined

$$U^*(Z) \coloneqq \sup_a : U\left(Z_T + \sum_{t=0}^{T-1} a_t' D H_{t:T} - c_t(a_t)\right)$$

- We have no static arbitrage.
- We say that the market exhibits **statistical arbitrage** if $U^*(0) > 0$.
- Happens naturally.

Statistical Arbitrage

- Statistical Arbitrage is an economic fact.
- Naïve application to Deep Hedging to the S&P with a simulator trained on the period 2015-2025 leads to
  - Going long the index
  - Selling puts
  - Ignore hedging any derivative



Statistical Arbitrage

- Statistical Arbitrage is an economic fact
  - Selling insurance commands a risk premium
    - Selling puts
    - Selling long-term rates vs. short term rates
    - Selling credit protection
    - …
  - Arbitrageurs exploit opportunities to make the market more efficient
    - Hedge funds' economic role is to find such opportunities and drive markets towards efficiency
    - Some use forms of crowd sourcing to add signals
    - Entire (well paid) industry

Statistical Arbitrage

- We noted before that if an arbitrage opportunity is strictly non-negative, then Deep Hedging will take a maximal position in this opportunity.
- In the case of *statistical arbitrage* it will take a position which exhaust its risk capacity. That means if $u(x) = x$ and the monetary utility is "the expectation" then Deep Hedging will again try take maximal positions, regardless of the risk involved.
- Deep Hedging with "the expectation" as objective is very unstable.

Statistical Arbitrage

- In classic portfolio optimization we only have “linear” assets. To remove the drift, we simply divide each return by the mean return over the sample period: if X is some asset then:

$$d\tilde{X}_t^i := \frac{dX_t^i}{\frac{1}{m}(X_m - X_0)}$$

- For markets with complex assets removing the drift distorts the co-dependence of the instruments, e.g. stock and options thereon.
- Instead of changing the paths we aim now to *reweight* the observed paths such that the drift disappears – that means constructing a new equivalent measure $Q \ll P$.

Statistical Arbitrage

- Assume zero cost for illustration. Recall P is our statistical measure and that our OCE monetary utility is given in terms of a utility u as:

$$U^*(X) \coloneqq \sup_{a,y \in R} \underbrace{\mathrm{E}^P[u(y + \sum a_t' DH_{t:T}) - y]}_{=:W(y,a)}$$

- Solve now for an optimal static arbitrage policy a^0 and y^0 such that

$$0 < U^*(0) = W(y^0, a^0)$$

Statistical Arbitrage

- In an extremal point we get using the unit vector $e_t \in R^{n_t}$

$$y^0: \quad 0 =^! \partial_y W(y^0, a^0) \Longrightarrow \qquad \mathrm{E}^P\left[u'\left(y^0 + \sum a_t^{0\prime} DH_{t:T}\right)\right] = 1 \qquad (*)$$

$$a^0: \quad 0 =^! \partial_\epsilon W\left(y^0, a^0 + \epsilon e_t^i\right)\Big|_{\epsilon=0} \Longrightarrow \qquad \mathrm{E}^P\left[u'\left(y^0 + \sum a_t^{0\prime} DH_{t:T}\right) DH_{t:T}^i\right] = 0 \qquad (**)$$

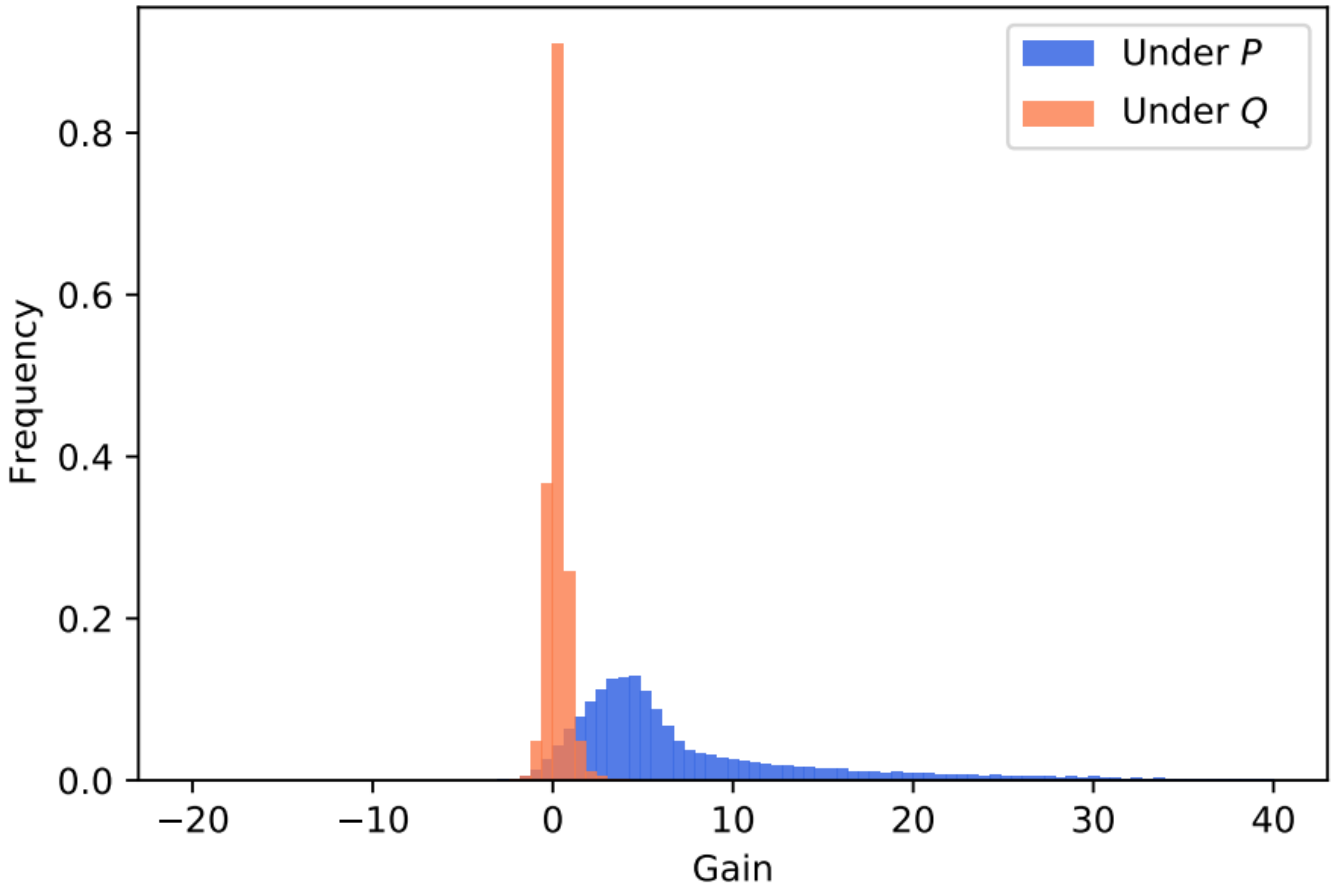
- Therefore, [1] Q defined as

$$dQ := u'\left(y^0 + \sum a_t^{0\prime} DH_{t:T}\right) dP$$

is an absolutely continuous probability measure thanks to $(*)$ and a martingale measure because of $(**)$.

[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844

Statistical Arbitrage



Numerical results of reducing the drift

[1] Learning Risk-Neutral Implied Volatility Dynamics, Buehler et al 2021, https://arxiv.org/pdf/2103.11948.pdf

Statistical Arbitrage

- The measure Q is one of many equivalent martingale measures.
- Our construction minimizes the $\tilde{u}$-divergence

$$D_f(Q|P) = \mathrm{E}^P[\tilde{u}(dQ/dP)]$$

where $\tilde{u}(y) := \sup_{x \in R}\{yx - f(x)\}$ is the Legendre-Fenchel transform of $f(x) = -u(-x)$. See [1] for a proof.

- If u is the exponential utility, then Q is the **minimum entropy martingale measure** by [2] which minimizes the entropy of Q with respect to P among all equivalent martingale measures i.e.

$$Q \mapsto E_P\left[\frac{dQ}{dP}\log\frac{dQ}{dP}\right]$$

[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844

[2] The Minimal Entropy Martingale Measure and the Valuation Problem in Incomplete Markets, Fritteli, 2000 Mathematical Finance, 0: 39-52

Statistical Arbitrage

- In the case of the entropy, without transaction cost we have the following intuitive result [1]:
- For a portfolio Z the optimal strategy a^* under U_P is given as

$$a^* := a^0 + \tilde{a}$$

- Where:
  - a^0 is the optimal "prop trading" strategy for the empty portfolio under P.
  - $\tilde{a}$ is the optimal "pure hedging" strategy under the risk-neutral Q.

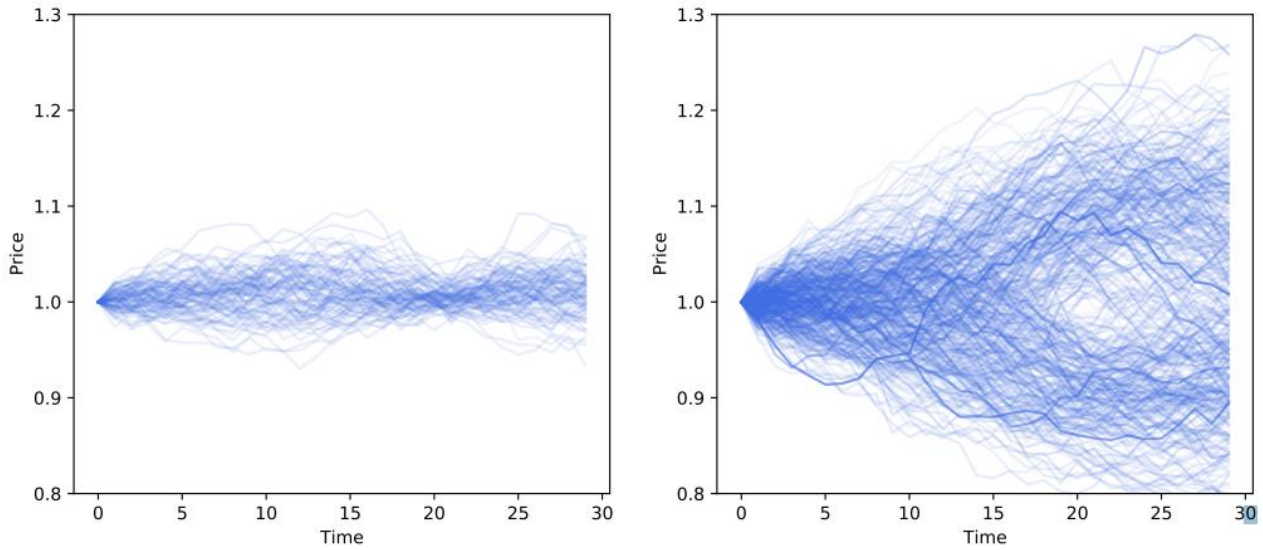
[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844

Statistical Arbitrage

- Fun fact: in discrete time, we can change also the *volatility* of a process by changing measure.
  - Experiment: market with 15% annual realized volatility. Option traded with 20% volatility. Statistical arbitrage is selling the option and delta hedge.
  - What happens when we change our measure:
    The measure will put more weight on paths with lower realized (discrete time) variance per path

Statistical Arbitrage

- Experiment [1]: market with 15% annual realized volatility. Option traded with 20% volatility. Statistical arbitrage is selling the option and delta hedge.
- The measure will put more weight on paths with lower realized (discrete time) variance per path



Left: paths given the highest 0.1% of probabilities under Q; right: lowest 0.1%

[1] Learning Risk-Neutral Implied Volatility Dynamics, Buehler et al 2021, https://arxiv.org/pdf/2103.11948.pdf

Statistical Arbitrage

Generalization to cost and arbitrary u

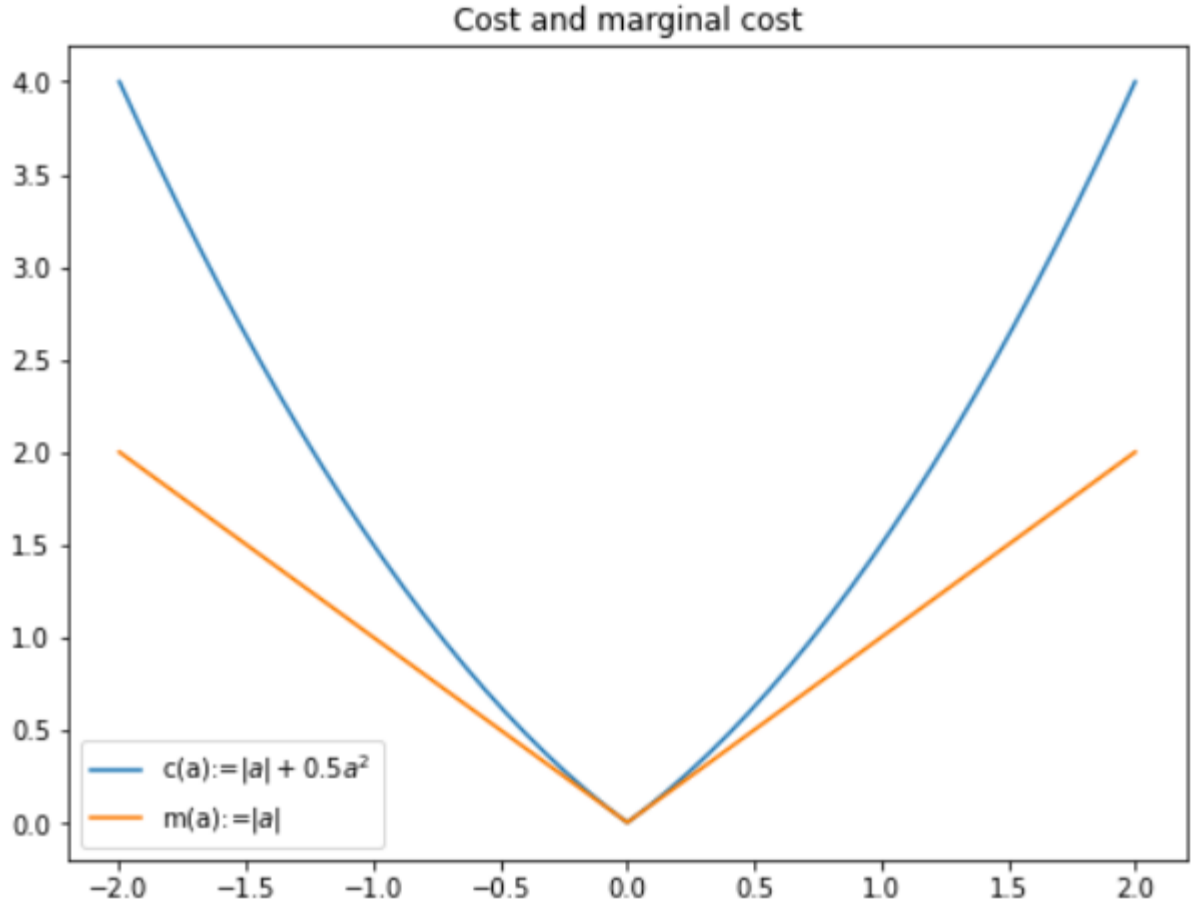
- Define "marginal cost" m_t as linearization of c in zero as follows [1]:
  - Denote by $\nabla^{\pm} c_t(0)$ the gradients in zero in positive and negative direction – practically that will be half the bid/ask spread γ_t, or infinity.
  - Let $a^{\pm} := \max\{\pm a, 0\}$ and define the linearized cost

$$m_t(a) := a^{+} \cdot \nabla^{+} c_t(0) - a^{-} \cdot \nabla^{-}(0)$$

  - Essentially, if the asset is tradable in the respective direction $\nabla^{+} c_t(0)^i$ is the spread between ask price and book value, and $\nabla^{-} c_t(0)^i$ is the spread between book value and bid price. If the asset can be bought/sold the respective value is infinite.

[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844

Statistical Arbitrage



Statistical Arbitrage

- Apply similar idea to before [1]: find a^0, y^0 as solution to

$$\sup_{a,y \in R} : \mathrm{E}^P \left[u\left(y^0 + \sum a_t' DH_{t:t} - m_t(a_t)\right) - y^0 \right]$$

- Define the measure Q similar to before via

$$dQ := u'\left(y^0 + \sum a_t^{0\prime}\ DH_{t:t} - m_t(a_t^0)\right)\ dP$$

[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844

Statistical Arbitrage

- Under the new measure the expected returns of all instruments are within marginal bid/ask spread in the following sense:
  - Let $\gamma_t^{i\pm} := \nabla^{\pm}(e^i)$ be the marginal cost of buying/selling a small quantity of the ith asset. Then [1] the measure Q is a **near-martingale measure** in the sense that

$$H_t^i - \gamma_t^{i-} \leq \mathrm{E}^Q\left[H_T^i | s_t\right] \leq H_t^i - \gamma_t^{i+}$$

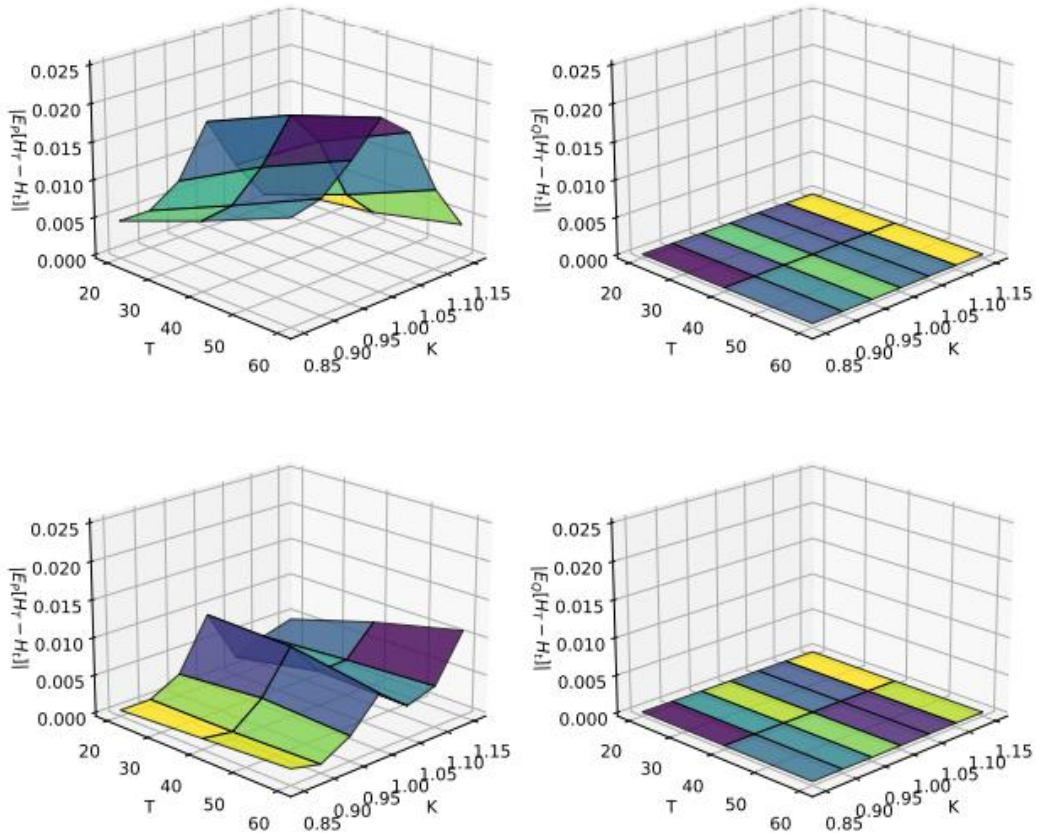
  - There is no statistical arbitrage under Q with full or marginal transaction cost:

$$0 = \max_a : U_Q\left(\sum a_t' D H_{t:t} - m_t(a_t)\right)$$

  - The measure Q minimizes the $\tilde{u}$-divergence to P among all near-martingale measures [1].

[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844

Statistical Arbitrage



Full market simulation results [1]: left are expected returns under P, right under Q under transaction cost

[1] Deep hedging: learning to remove the drift, Buehler et al 2022 https://www.risk.net/cutting-edge/banking/7932226/deep-hedging-learning-to-remove-the-drift and https://arxiv.org/abs/2111.07844

Stochastic Implied Volatility

- We built a market simulator for (a form of) implied volatilities
- We then removed the drift … such that the price processes become martingales
- We have created with machine learning a *stochastic implied volatility model* … a task not achieved through years of classic quantitative finance research !
- Back to risk-neutral evaluation with expectations and greeks to compute hedging ratio… *performance analysis not quite done yet – needs volunteers for some research work.*

Statistical Arbitrage

- Good
  - Nice theoretical framework
  - Model-free approach that works for any market model, any currency etc
  - Same numerical complexity as Deep Hedging

- But
  - Removing the drift is akin to giving up coming up on a better estimate on where the market goes
  - In some sense *too pessimistic* … should we not pick up some drift if we can be certain about it?

Model Uncertainty

Area of active research in all aspects of quantitative finance.

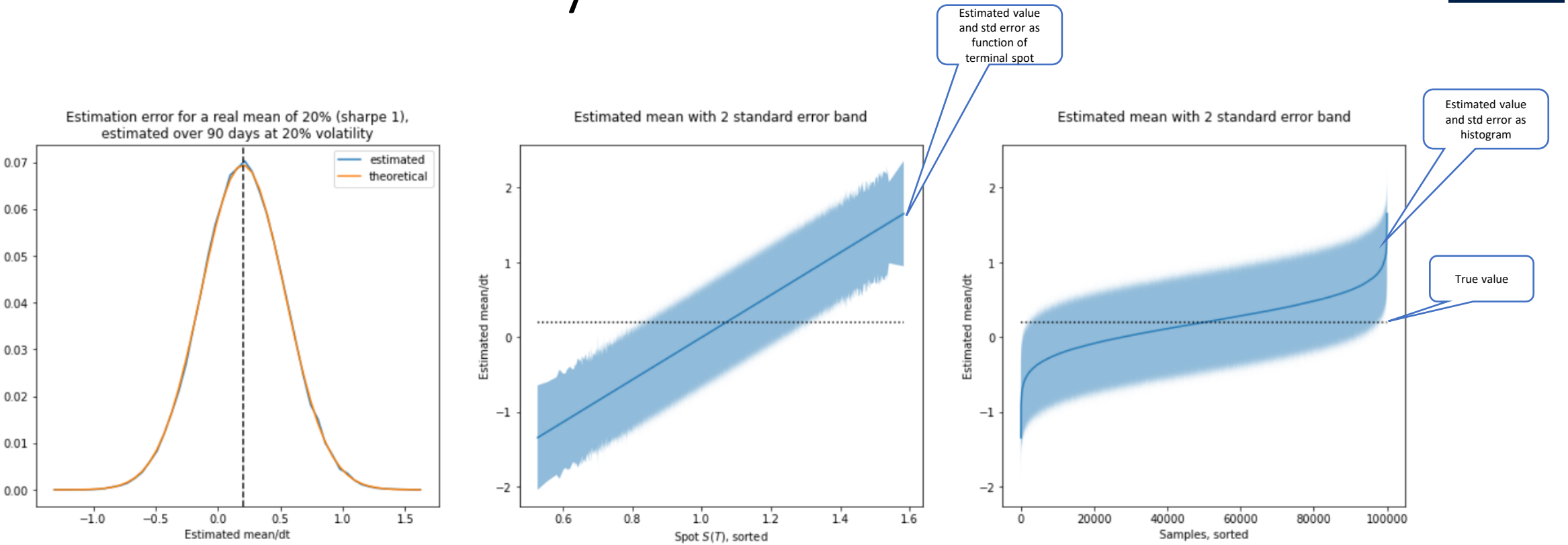
Model Uncertainty

- Assume we observe a market generated by a normal asset price X
  - Annualized volatility 20%
  - Annualized drift 20% i.e. sharpe is 1
- In practise, we observe *one path.* On that path we estimate the mean of the process.

$$\text{mean}(dX) = \frac{1}{m}\sum_{t=1}^{m} dX_t = \frac{1}{m}(X_m - X_0)$$

  - The estimator itself is a random variable
  - How does this look like statistically?

Model Uncertainty



The naïve mean estimator is *uncertain* as it depends on the path we are on.
Effect is called *Knightian Uncertainty*

Model Uncertainty

- Why does the uncertainty of the mean matter?

$$U(X) = \mathrm{E}^{Q}[X] + \underbrace{U(X - \mathrm{E}^{Q}[X])}_{Risk\ term}$$

- Finding an optimal hedging strategy is very sensitive to the mean.
- Machine *thinks* it can make money … but it doesn't work because of the estimation error.
- If we have removed the drift → problem solved.
- What if we do not want to do that?

Model Uncertainty

- Knightian Uncertainty: classic problem in systematic investment strategies – plenty of heuristics to address the issue even for classic "Markoviz" portfolio construction [1]

- Let us state the dependency on "Q" in Deep Hedging explicitly: Our OCE Monetary Utility given a utility function u is:

$$U_{\mathrm{Q}}(X) \coloneqq \sup_{y \in R} \mathrm{E}^{Q}[u(y + X) + y]$$

Hence:

$$U_Q^*(Z) \coloneqq \sup_a U_{\mathrm{Q}}\left(Z_T - \sum_t a_t' D H_{t:T} - c_t(a)\right) \coloneqq \sup_{a, y \in R} \mathrm{E}^{Q}\left[u\left(y + Z_T - \sum_t a_t' D H_{t:T} - c_t(a)\right) + y\right]$$

[1] good summary: Marcos Lopez de Prado: Advances in Financial Machine Learning, Wiley, 2018

Model Uncertainty

- Classic Robust statistics: basic idea to scramble the distribution and take a sort of “inf”:

$$\inf_{Q \sim P} : U_Q(Z)$$

- Too conservative.
- We can do better [1] and also [2].

[1] Uncertainty-Aware Strategies: A Model-Agnostic Framework for Robust Financial Optimization through Subsampling, Buehler et al, 2025, https://arxiv.org/abs/2506.07299
[2] Distributional Adversarial Attacks and Training in Deep Hedging, He 2025, https://arxiv.org/abs/2508.14757

Model Uncertainty

Convex Uncertainty Measures

- Assume there is a distribution μ over the set of absolutely continuous measures $Q \ll P$.
  - Note that if we have N paths, then each $Q \ll P$ is simply a vector $Q \in [0,1]^N$ of probabilities which add up to 1.
  - That means μ is a measure over a subset of R^N.

- An example is a normalized version of the likelihood of observed samples:

$$\mu(Q) = \text{likelihood}(\omega_1, \dots, \omega_N)/|\dots|$$

- If we again assume we consider M candidate measures, then we can pre-compute $\mu(Q)$ ahead of our optimzaiton.

Model Uncertainty

Uncertainty Measures

- Define the **uncertainty-adjusted return measure** in terms of a utility w for a function $f: Q \mapsto f(Q)$ as "OCE" style functional:

$$W(f) \coloneqq \sup_{e \in R} \int_Q \{w(e + f(Q)) - e\} \mu(dQ)$$

- W is a measure of return adjusted by *uncertainty*.
- Discretized:

$$W(f) = \sup_{e \in R} \sum_{k=1}^{M} \mu(Q_k) \{w(e + f(Q_k) - e\}$$

Model Uncertainty

Uncertainty-Aware Deep Hedging

- We now apply W to Deep Hedging:

$$UW^*_{Q,\mu}(Z) \coloneqq \sup_a : W\left[U_Q\left(Z_T - \sum_t a'_t DH_{t:T} - c_t(a)\right)\right]$$

- Discretized, and written out

$$UW^*_{Q,\mu}(Z) = \sup_{a,e\in R} \sum_{k=1}^{M} \mu(Q_k)\left\{w\left(e + \sup_{y^k\in R} \sum_{\omega=1}^{N} Q_k(\omega)\, u\left(y^k + Z_T(\omega) + \sum_t a'_t DH_{t:T}(\omega) - c_t(a_t)(\omega)\right) - y^k\right) - e\right\}$$

- This gives the **Uncertainty-Aware Deep Hedging** problem:

$$UW^*_{Q,\mu}(Z) = \sup_{a,y^k\in R,e\in R} : \sum_{k=1}^{M} \mu(Q_k)\left\{w\left(e + \sum_{\omega=1}^{N} Q_k(\omega)\, u\left(y^k + Z_T(\omega) + \sum_t a'_t DH_{t:T}(\omega) - c_t(a_t)(\omega)\right) - y^k\right) - e\right\}$$

Model Uncertainty

Convex Uncertainty Measures

- …?

Model Uncertainty

Bootstrapped Deep Hedging

- If $w(x) = x$ then $W(f) = \int f(Q)d\mu(Q)$ which gives a "bootstrapped" Deep Hedging

$$UW^*_{Q,\mu}(Z) = \sup_{a,y^1,\dots y^M \in R} : \sum_{k=1}^{M} \mu(Q_k) \sum_{\omega=1}^{N} Q_k(\omega)\, u\left(y^k + Z_T(\omega) + \sum_t a'_t DH_{t:T}(\omega) - c_t(a_t)(\omega) \right) - y^k$$

Model Uncertainty

Entropy

- For the exponential utility $u(x) = (1 - \mathrm{e}^{-\lambda x})/\lambda$ with equal risk and uncertainty aversion $\lambda > 0$, and $w = u$ we get

$$U_Q(X) = -\frac{1}{\lambda}\log \mathrm{E}^Q\left[e^{-\lambda X}\right] \quad \text{and} \quad W(f) = \frac{1}{\lambda}\log \int e^{-\lambda f(Q)}\mu(dQ)$$

- We obtain:

$$UW^*_{Q,\mu}(Z) = \sup_a : \frac{1}{\lambda}\log \sum_{\omega=1}^{N}\left(\sum_{k=1}^{M} Q_k(\omega)\,\mu(Q_k)\right) e^{-\lambda\left(Z_T(\omega)+\sum_t a_t' DH_{t:T}(\omega)-c_t(a_t)(\omega)\right)}$$

- We can translate this back to the more robust

$$UW^*_{Q,\mu}(Z) = \sup_{a,y\in R} : \sum_{\omega=1}^{N}\left(\sum_{k=1}^{M} Q_k(\omega)\,\mu(Q_k)\right) u\left(y + Z_T(\omega) + \sum_t a_t' DH_{t:T}(\omega) - c_t(a_t)(\omega)\right) - y$$

- We can therefore solve the **uncertain deep hedging problem** for the same entropy using our existing framework.

Model Uncertainty

CVaR

- The CVaR monetary utility with $u(x) = (1+r)\min\{x, 0\}$ computes the expectation over the $\alpha = 1/(1+\lambda)$ bottom fraction of our measures.
- Hence, solving the (original) Deep Hedging with, say, $\alpha = 90\%$ finds the best hedge ignoring the top 10% of performance.
  The top 10% here is computed the statistical weight $1/N$ per path.

- Assume now also that $w(x) = (1+r)\min\{x, 0\}$.
- The uncertainty-adjusted Deep Hedging problem in this case finds the best hedge across the bottom 90% of paths, again with a conservative bottom 90% of uncertainty.

- This was discussed in [11]
- Plenty of research questions of optimal choices.

[1] Uncertainty-Aware Strategies: A Model-Agnostic Framework for Robust Financial Optimization through Subsampling, Buehler et al, 2025, https://arxiv.org/abs/2506.07299

Deep Hedging & Market Simulation

- Used in JP Morgan for Cliquet Options
  https://www.risk.net/derivatives/equity-derivatives/7921526/jp-morgan-testing-deep-hedging-of-exotics

- Good
  - Promising, versatile, intuitive machinery
  - Model-free in the sense that the machine is easily transferrable
  - Takes into account market frictions
  - Results deviate from classic models
  - Recent work on robust “bootstapped” hedging [1] and [2]

- But
  - Very heavy numerically
  - Results deviate from classic models and need explanation
  - Vanilla Deep Hedging requires re-training every day and every time Z changes

[1] Uncertainty-Aware Strategies: A Model-Agnostic Framework for Robust Financial Optimization through Subsampling, Buehler et al, 2025, https://arxiv.org/abs/2506.07299
[2] Distributional Adversarial Attacks and Training in Deep Hedging, He 2025, https://arxiv.org/abs/2508.14757

Deep Bellman Hedging

(on-going research)

Reset notation – but keep the spirit

Deep Bellman Hedging

- The one-day deterministic discount factor is $\beta_t \in (0, \beta^*]$ with $\beta^* < 1$.

- Assume that we are given a portfolio $Z^{(t)}$ with book-value $Z_t^{(t)}$ today. Here book value excludes past cash flows.
  - Tomorrow's book value of today's portfolio is $Z_{t+1}^{(t)}$.
  - Hence, cashflows and changes in discounted book value to tomorrow are reflected in

$$dZ_t^{(t)} := \beta_t Z_{t+1}^{(t)} - Z_t^{(t)}$$

- We can trade today in hedging instruments $H^{(t)}$ with book values $H_t^{(t)}$ for cost $c_t(a) \geq 0$. As before

$$dH_t^{(t)} := \beta_t H_{t+1}^{(t)} - H_t^{(t)}$$

Deep Bellman Hedging

- If we trade a units of $H^{(t)}$ then tomorrow's portfolio is formally defined as

$$Z^{(t+1)} := Z^{(t)} \oplus a' H^{(t)}$$

- Our one-day **reward** of following action a is then the (β_t-discounted) book P&L

$$dZ_t^{(t)} + a' dH_t^{(t)} - c_t(a)$$

- This gives rise to the Bellman equation c.f. [1]:

$$V^*\left(Z^{(t)}, s_t\right) = \sup_{\mathrm{a}} U_t \left[\beta_t V^*\left(Z^{(t)} \oplus a' H^{(t)}, s_{t+1}\right) + dZ_t^{(t)} + a' dH_t^{(t)} - c_t(a)\right]$$

[1] Deep Bellman Hedging, Buehler et al, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4151026

Deep Bellman Hedging

- Theorem [1]: define the Bellman Operator T for candidate value functions V as the *statistical hedging problem* we discussed before:

$$TV\left(Z^{(t)}, s_t\right) = \sup_{\mathrm{a}} U_t\left[\beta_t V\left(Z^{(t)} \oplus a' H^{(t)}, s_{t+1}\right) + dZ_t^{(t)} + a' dH_t^{(t)} - c_t(a)\right]$$

  - Assume that there is no static or statistical arbitrage such that $T0 < \infty$.
  - Then the sequence $V^{n+1} := TV^n$ starting in $V^0 := 0$ converges to a finite optimal solution V^* for any monetary utility U.
    The solution V^* then satisfies the Belman equation.

- Note that V represents the correction on top of the book value.
  - If our book values already fully explain all P&L, then $V^* = 0$. See [1] for a discussion on this point and why this setup helps convergence.

[1] Deep Bellman Hedging, Buehler et al, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4151026

Deep Bellman Hedging

- Bellman relationship for an optimal value V^*.

$$V^*\left(Z_t^{(t)}, s_t\right) \coloneqq \sup_{\mathrm{a}} U_t\left[\beta_t V^*\left(Z^{(t)} \oplus a' H^{(t)}, s_{t+1}\right) + dZ_t^{(t)} + a' dH_t^{(t)} - c_t(a)\right]$$

- How do we represent our portfolio → use what trading has been using for years: the greeks and risk scenarios available to them as well as any cashflows.
  - Typically, we have these values historically at individual hedging and OTC instrument level.
  - We can therefore sample over historic periods and solve for V^*.

[1] Deep Bellman Hedging https://arxiv.org/abs/2207.00932

Practical Implementation

- Numerical solution via Actor-Critic:

$$TV\left(\mathrm{Z}_t^{(t)}, s_t\right)$$
$$:= \sup_{\mathrm{a,y}} E_t\left[u\left(y + \beta_t V\left(Z^{(t)} \oplus a' H^{(t)}, s_{t+1}\right) + dZ_t^{(t)} + a' dH_t^{(t)} - c_t(a)\right) - y\right]$$

- **Actor:** Start in $V^0 := 0$; given V^{n-1} maximize a^n and y^n which also yields samples of TV^n.
- Note that both a and y are functions of the current state.

Practical Implementation

- **Critic:** given samples of TV^{n-1} solve for a neural network V^n to satisfy

$$V^n(w, s_t) \equiv TV^{n-1}(w, s_t)$$

- This interpolation program may also be solved by simpler, classic methods such as kernel interpolators.

Deep Bellman Hedging

- Implementation with fixed T: worked [1]
- Actual actor/critic: unstable with results so far for only simple cases (such as portfolios of vanillas)
- [2] was/is the first of its kind; much more work to be done

[1] Deep Hedging: Continuous Reinforcement Learning for Hedging of General Portfolios across Multiple Risk Aversions, Murray et al, https://arxiv.org/abs/2207.07467
[2] Deep Bellman Hedging, Buehler et al, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4151026

Please ask questions