Deep Learning for Early-Exercise IR Derivatives Valuation

Álvaro Leitao

Interaction with the industry

- Research carried out under a knowledge-transfer initiative with the Spanish bank BBVA.
- Budget: 57341,85€ (initial) + 11468,37€ (extension)
- BBVA team (CVA and risk management):
  - Francisco Gómez Casanova
  - Fernando de Lope Contreras
- CITIC/UDC team (M2NICA research group):
  - Carlos Vázquez Cendón
  - Álvaro Leitao Rodríguez
- Work dynamics:
  - Bi-weekly scientific meetings
  - Semiannually intermediate reports
  - Final deliverable: Final report + codes

Motivation and proposal

- Deep learning is increasingly used to accelerate scientific computing tasks.
- Computational finance faces demanding pricing and risk management problems.
- Derivative valuation remains one of the industry's most challenging tasks.
- Traditional numerical methods are effective but computationally expensive.
- We combine differential learning, sampled Monte Carlo labels, and joint learning.
- A novel interdependent ANN architecture is proposed for Bermudan swaption pricing.

Outline

1. Financial derivatives valuation
2. Specific problem formulation
3. Deep Joint learning for Bermudan swaptions
4. Numerical experiments
5. Conclusions

What Are Financial Derivatives?

- **Definition:** Financial instruments whose value depends on the price of another asset (e.g., stocks, interest rates, commodities).
- **Common Types:**
  - Options
  - Futures
  - Swaps
- **Purpose:**
  - Hedging risk
  - Speculation
  - Arbitrage
- **Example:** A farmer uses futures to lock in a price and protect against market fluctuations.

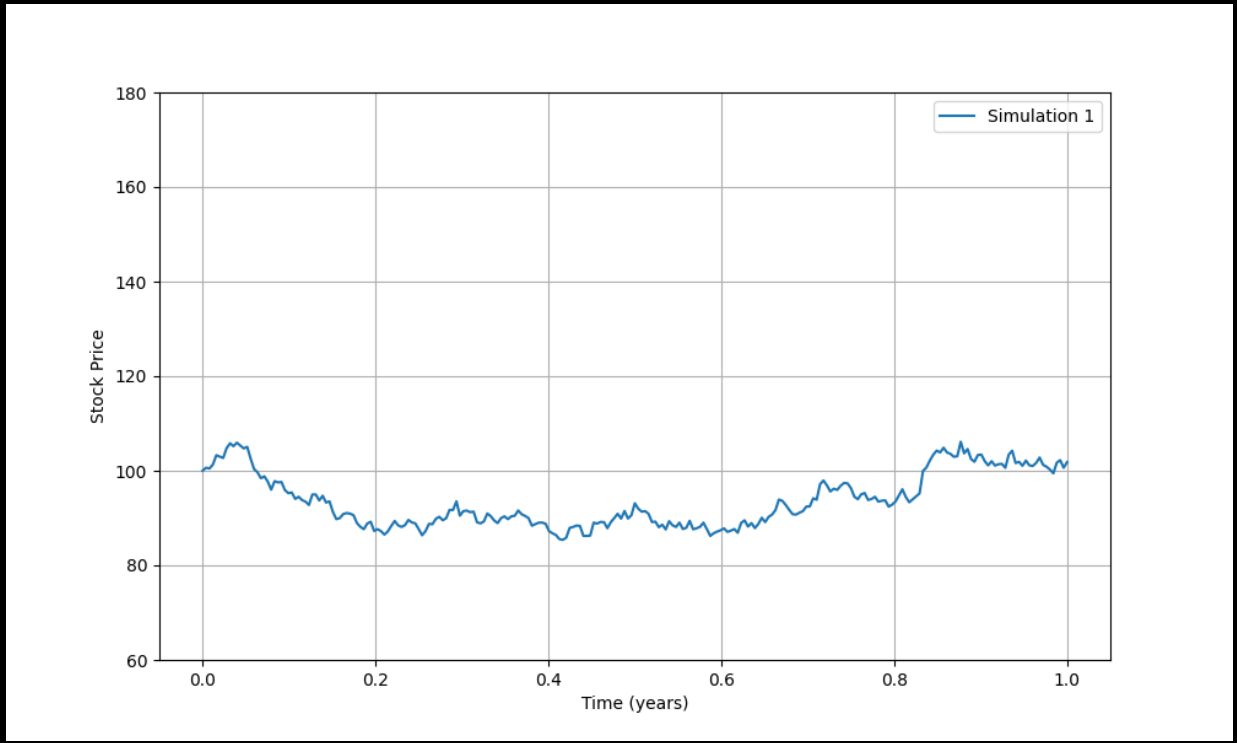
How Are Derivatives Valued?

- **Core Idea:** The value of a derivative reflects expectations about future market behavior/evolution.
- **What Influences the Value?**
  - **Time:** How long until the contract ends.
  - **Volatility:** How much the underlying asset's price moves.
  - **Interest Rates:** Affect the cost of holding or borrowing money.
- **Modeling the Future:**
  - Analysts use mathematical models to simulate possible future price paths.
  - These models help estimate the likelihood of different outcomes.
- **Main goal:** Find a **fair** price that reflects both risk and opportunity.
- **Real-World Analogy:** Like insuring a car—the price depends on risk, time, and expected events.

Modeling the Future?

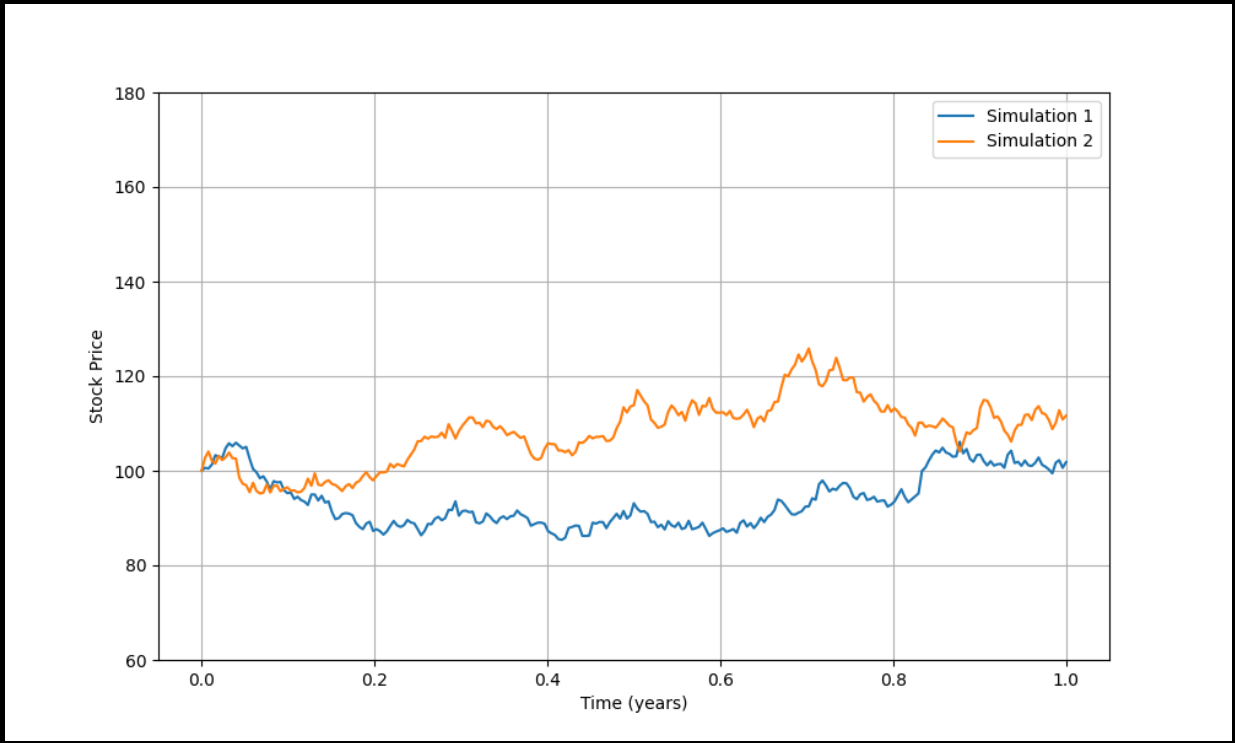
citic

Modeling the Future?



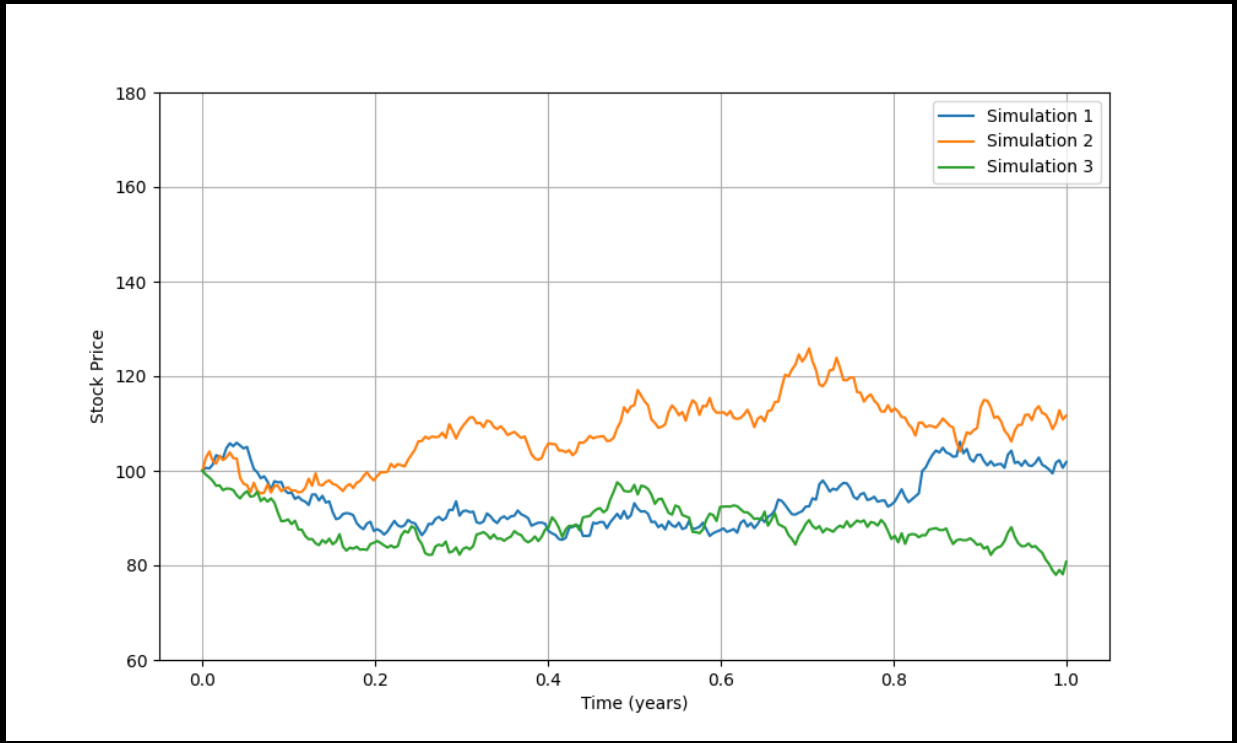
Simulation of a future scenario.

Modeling the Future?



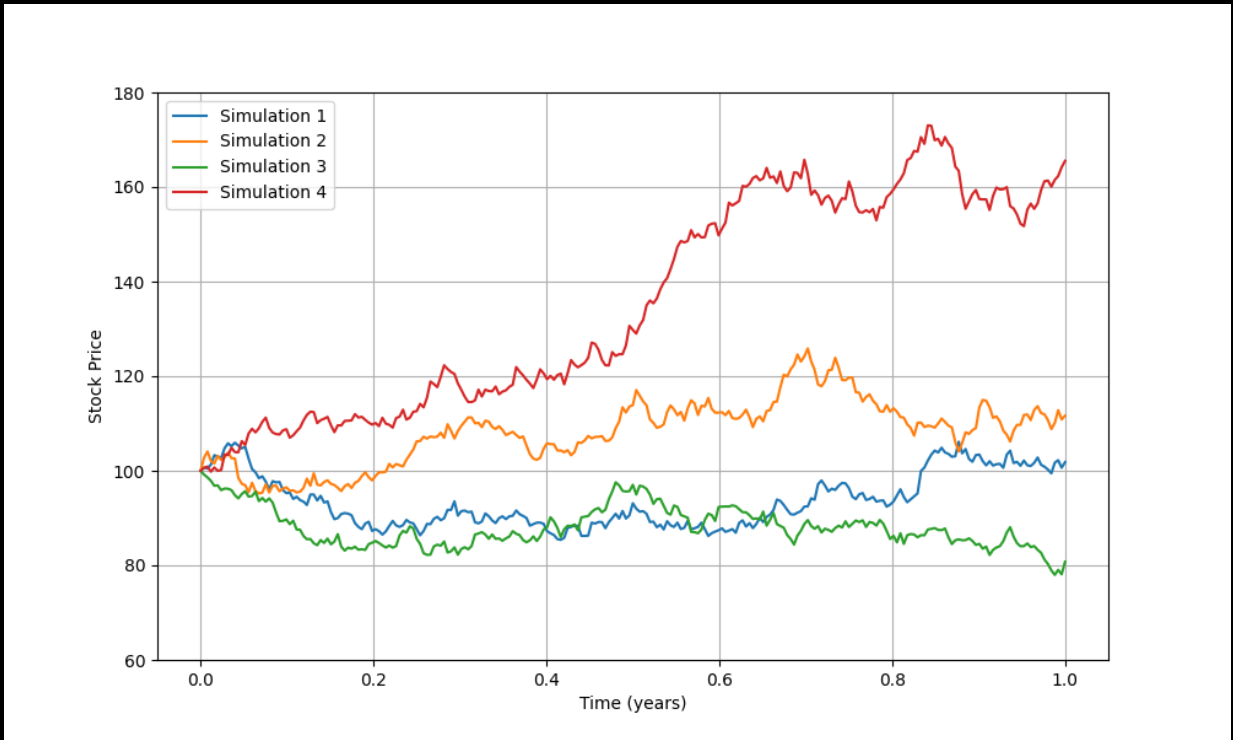
Simulation of a future scenario.

Modeling the Future?



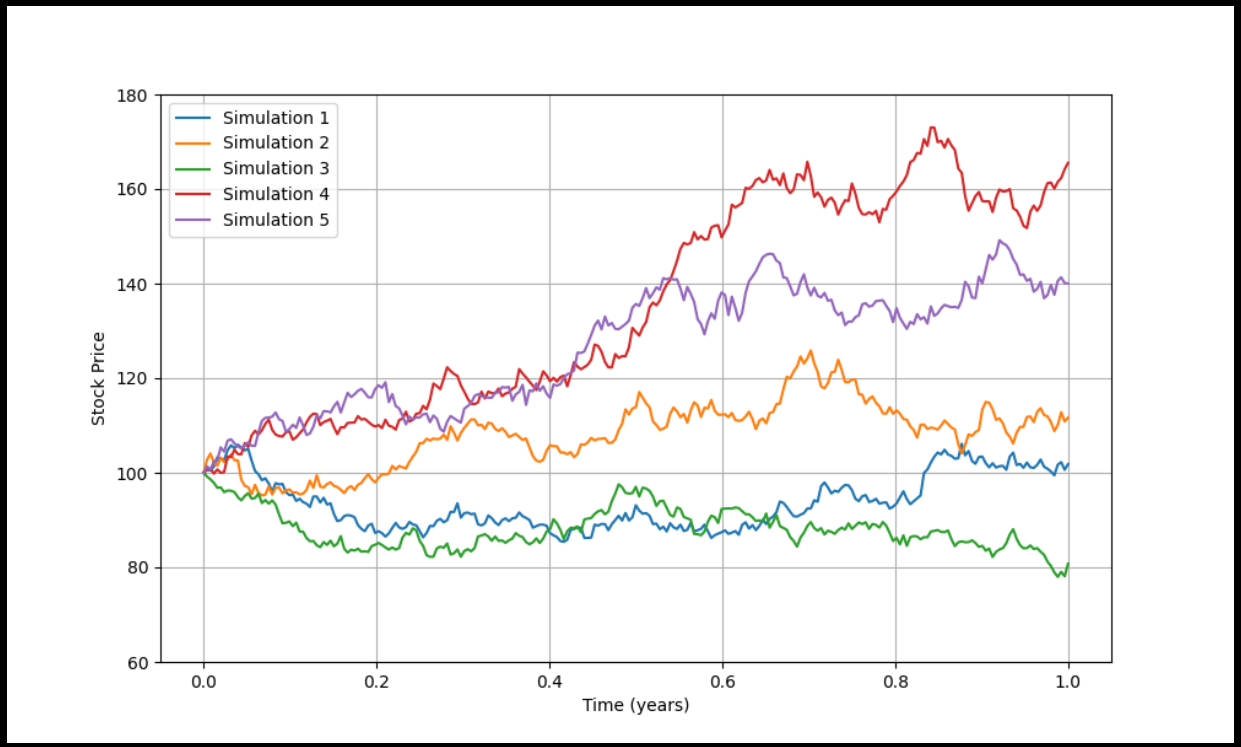
Simulation of a future scenario.

Modeling the Future?



Simulation of a future scenario.

Modeling the Future?



Simulation of a future scenario.

Modeling Financial Assets

- Financial assets are often modeled using **Stochastic Differential Equations (SDEs)**.
- A generic SDE for asset price S_t is:

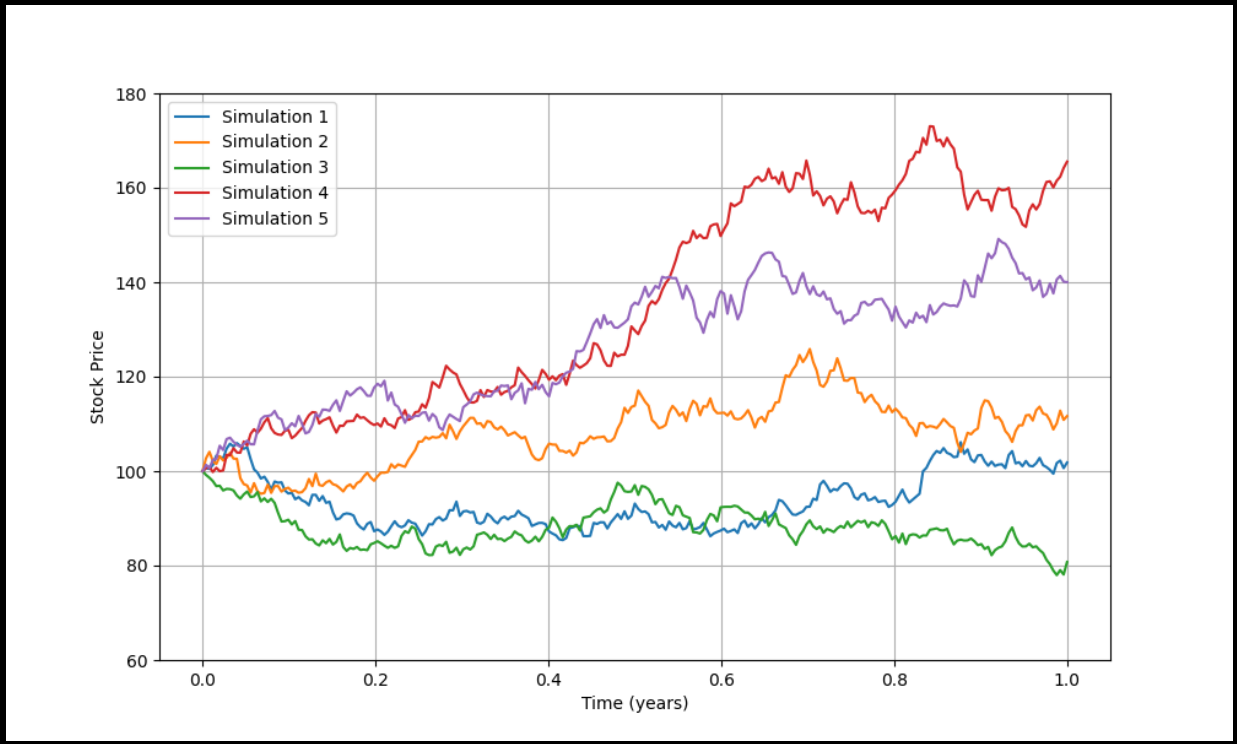
$$dS_t = \mu(S_t, t)dt + \sigma(S_t, t)dW_t,$$

  where:
  - $\mu(\cdot)$: expected return (drift)
  - $\sigma(\cdot)$: volatility (random fluctuations)
  - W_t: Brownian motion (random noise)
- This captures both predictable trends and unpredictable market movements.
- We also need a starting point, S_0, typically observed in the market "today".
- A SDE encapsulates a statistial distribution that drives the evlolution

Most basic derivatives: European Options

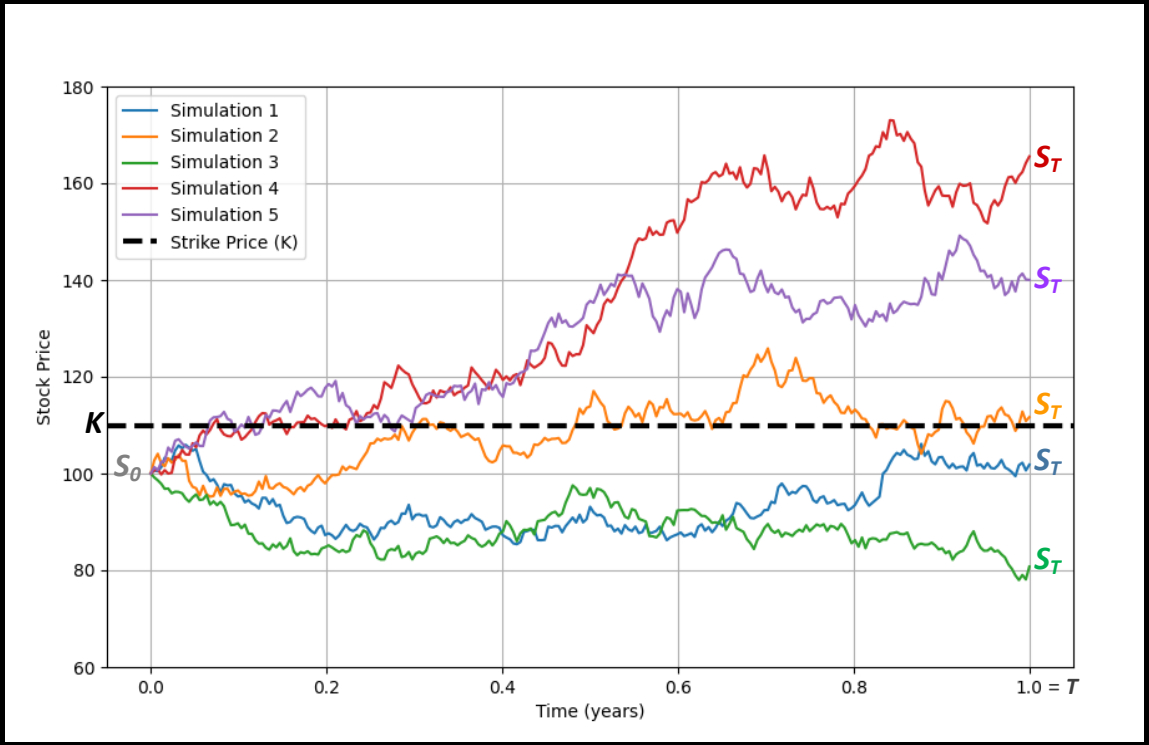
- A **European option** gives the holder the right (but not the obligation) to buy or sell an asset at a fixed price, called strike, on a specific future date.
- Two main types:
  - **Call option**: Right to buy
  - **Put option**: Right to sell
- Key features:
  - Can only be exercised at maturity...
  - ... so it only depends on the asset price at expiration
- Used for hedging, speculation, and risk management.

How to price a European call option?



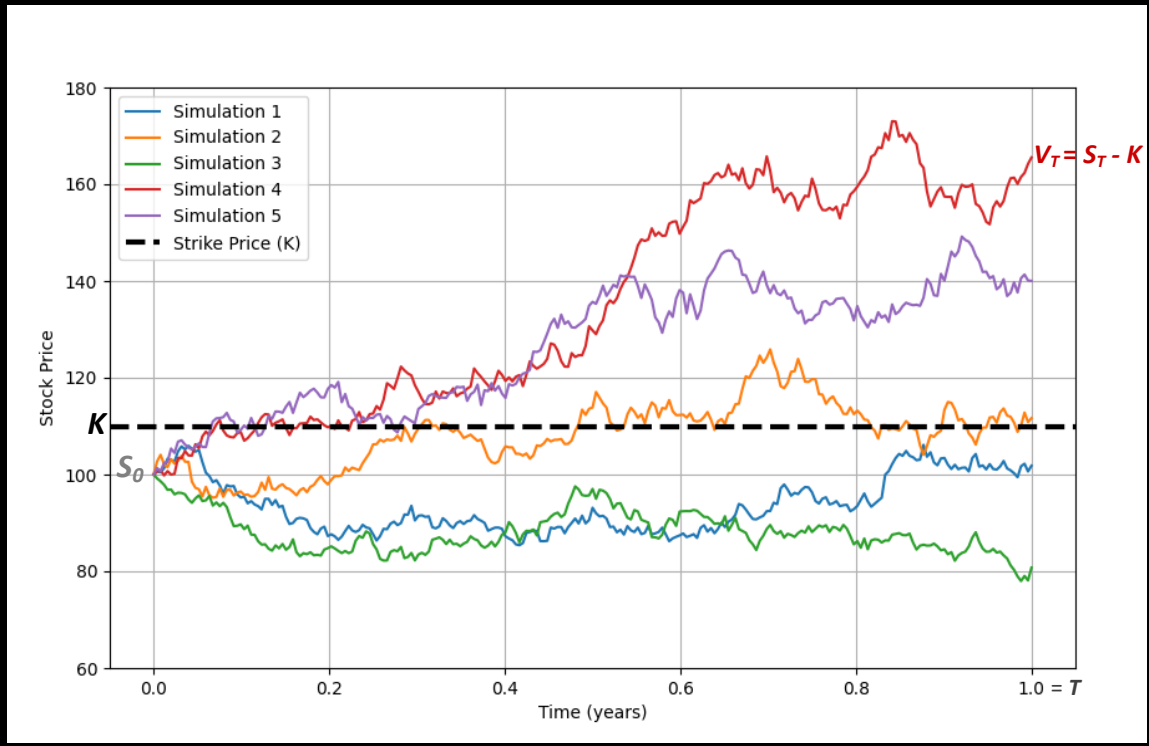
Pricing European options.

How to price a European call option?



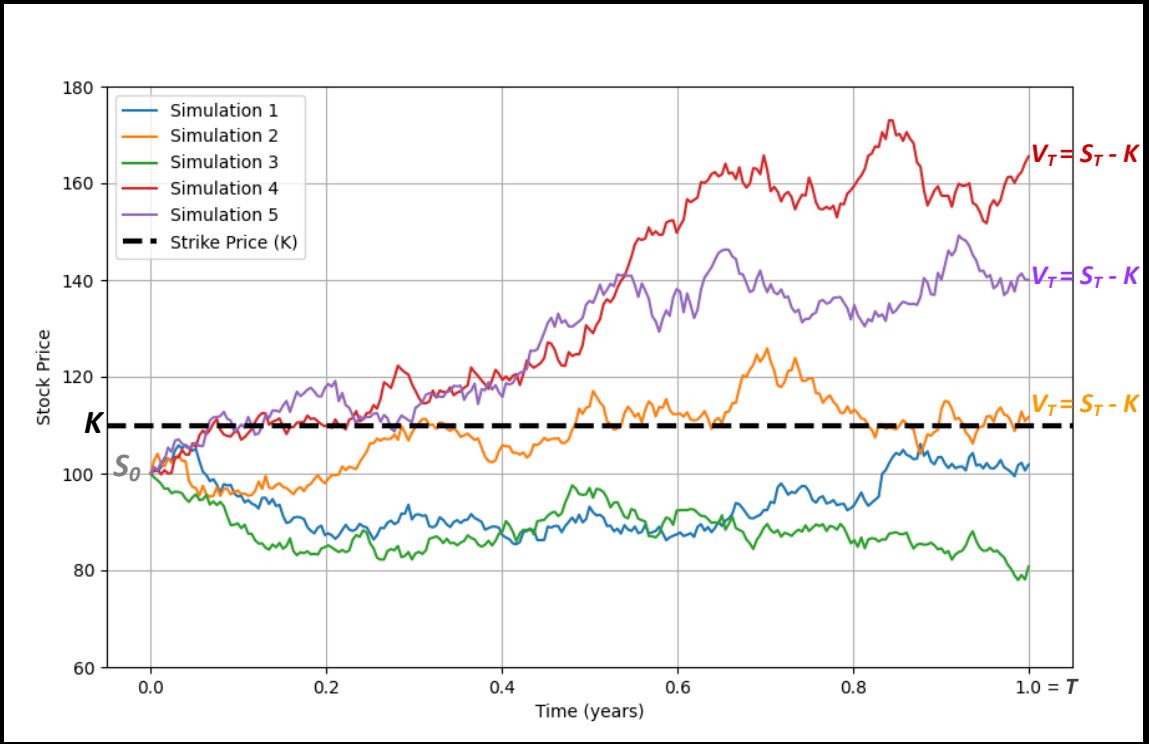
Pricing European options.

How to price a European call option?



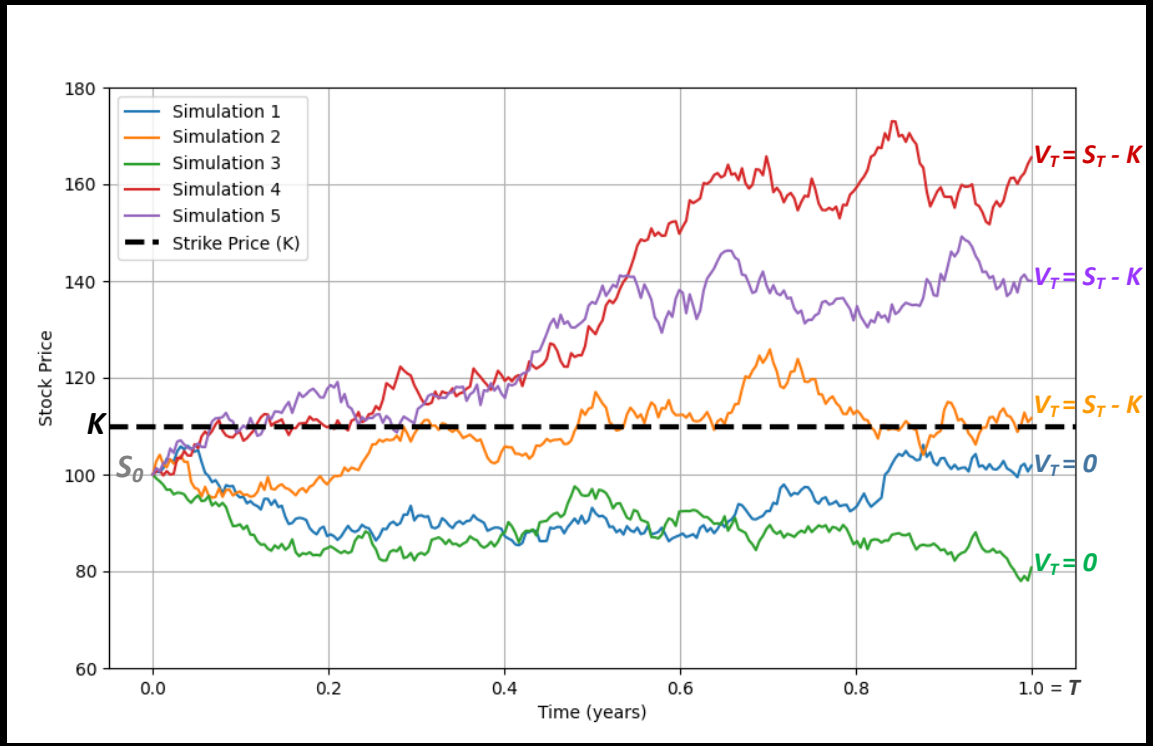
Pricing European options.

How to price a European call option?



Pricing European options.

How to price a European call option?



Pricing European options.

Valuing European Options with Monte Carlo integration

- Apply the **Monte Carlo estimator** over the simulated possible future asset price scenarios.
- For each scenario:
  - Compute the payoff at maturity, T:

$$\text{Call: } V_T := \max(S_T - K, 0) \quad \text{Put: } V_T := \max(K - S_T, 0)$$

  - Discount the payoff to present value.
- The option price is average the discounted payoffs across all paths:

$$V_0^N = e^{-rT} \left(\frac{1}{N} \sum_{i=1}^{N} V_T^{(i)} \right)$$

Valuing European Options with Monte Carlo integration

- Apply the **Monte Carlo estimator** over the simulated possible future asset price scenarios.
- For each scenario:
  - Compute the payoff at maturity, T:

$$\text{Call: } V_T := \max(S_T - K, 0) \quad \text{Put: } V_T := \max(K - S_T, 0)$$

  - Discount the payoff to present value.
- The option price is average the discounted payoffs across all paths:

$$V_0^N = e^{-rT}\left(\frac{1}{N}\sum_{i=1}^{N} V_T^{(i)}\right) \approx e^{-rT}\mathbb{E}[V_T|S_0]$$

Valuing European Options with Monte Carlo integration

- Apply the **Monte Carlo estimator** over the simulated possible future asset price scenarios.
- For each scenario:
  - Compute the payoff at maturity, T:

$$\text{Call: } V_T := \max(S_T - K, 0) \quad \text{Put: } V_T := \max(K - S_T, 0)$$

  - Discount the payoff to present value.
- The option price is average the discounted payoffs across all paths:

$$V_0^N = e^{-rT}\left(\frac{1}{N}\sum_{i=1}^{N} V_T^{(i)}\right) \approx e^{-rT}\mathbb{E}[V_T|S_0] =: V_0$$

Valuing European Options with Monte Carlo integration

- Apply the **Monte Carlo estimator** over the simulated possible future asset price scenarios.
- For each scenario:
  - Compute the payoff at maturity, T:

$$\text{Call: } V_T := \max(S_T - K, 0) \quad \text{Put: } V_T := \max(K - S_T, 0)$$

  - Discount the payoff to present value.
- The option price is average the discounted payoffs across all paths:

$$V_0^N = e^{-rT}\left(\frac{1}{N}\sum_{i=1}^{N} V_T^{(i)}\right) \approx e^{-rT}\mathbb{E}[V_T|S_0] =: V_0$$

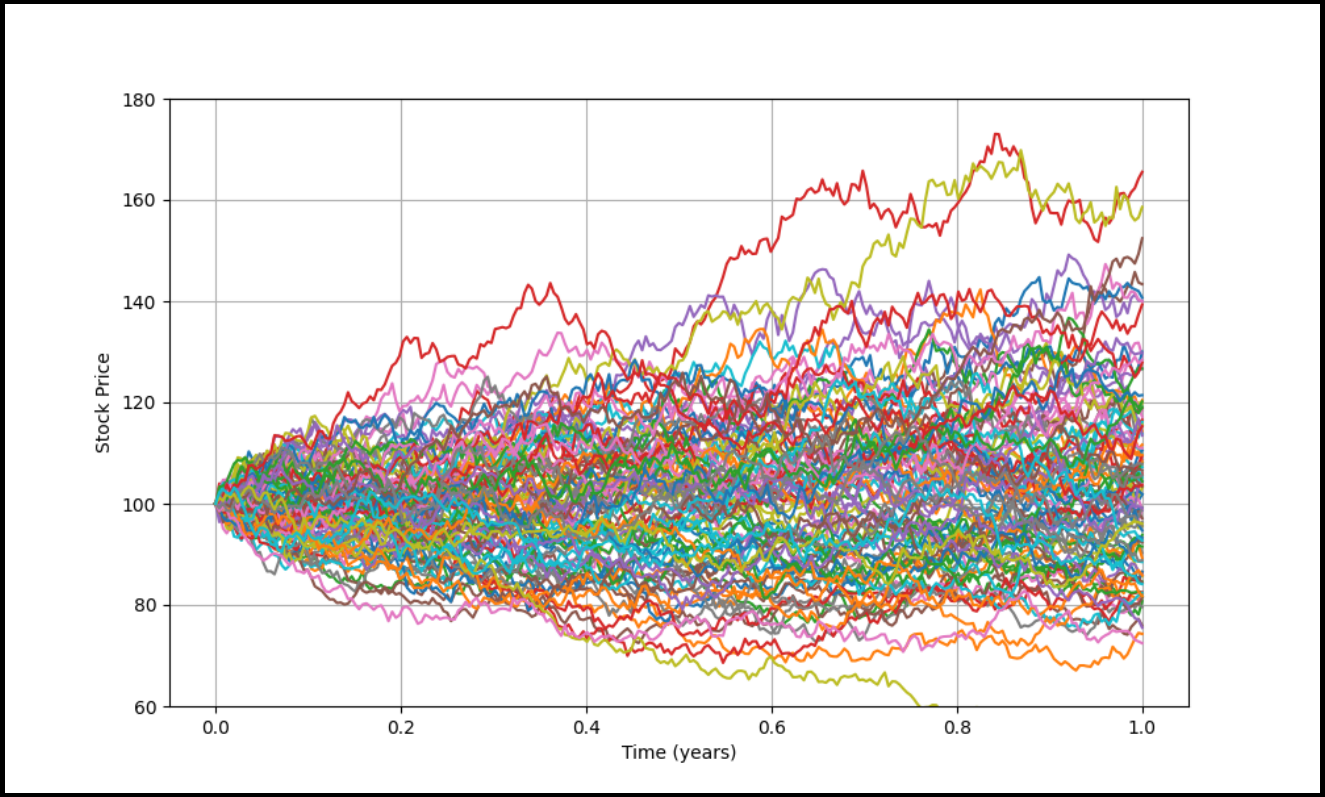
- **Main advantage**: Flexible and powerful for complex derivatives.

Problems of Monte Carlo-based valuation?

citic

- Convergence of order $\frac{1}{\sqrt{N}}$, so...

Problems of Monte Carlo-based valuation?



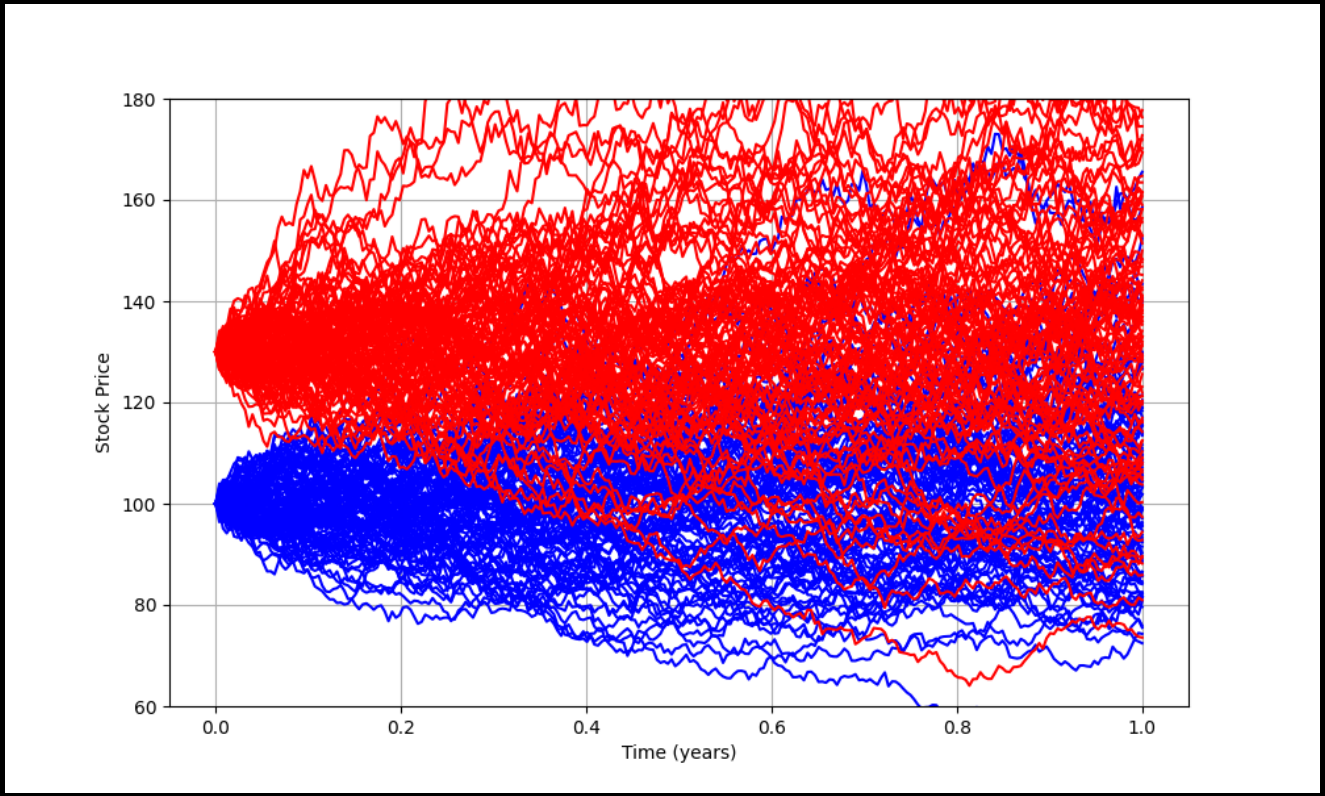
Problems of Monte Carlo-based valuation?

- Convergence of order $\frac{1}{\sqrt{N}}$, so...
- High computational cost to achieve high precision

Problems of Monte Carlo-based valuation?

- Convergence of order $\frac{1}{\sqrt{N}}$, so...
- High computational cost to achieve high precision
- Re-run if the “conditions” change ...

Problems of Monte Carlo-based valuation?



Problems of Monte Carlo-based valuation?

- Convergence of order $\frac{1}{\sqrt{N}}$, so...
- High computational cost to achieve high precision
- Re-run if the “conditions” change ...
- **Solution**: Incorporate Machine/Deep Learning techniques

Mathematical problem formulation

- Linear Gauss Markov (LGM) model:

$$dx_t = \alpha(t)dW_t, \quad x_0 = 0,$$

with $\zeta(t) := \int_0^t \alpha^2(\tau)d\tau$.

- The *numeraire* under LGM reads

$$N(t, x_t) = \frac{1}{D(t)} \exp\left(H(t)x_t + \frac{1}{2}H^2(t)\zeta(t)\right),$$

with $D(t)$ the discount factor of time t (observed in the market)

- $H(t)$ is a curve with a similar interpretation as the mean reversion in the Hull-White model:

$$H(t) = \frac{1 - \exp(-\kappa t)}{\kappa},$$

such as κ corresponds to the Hull-White mean reversion.

LGM: Analytic pricing formulas

- Zero coupon bond at t with maturity T:

$$Z(t, x_t; T) = \frac{D(T)}{D(t)} \exp\left(-(H(T) - H(t))x_t - \frac{1}{2}(H^2(T) - H^2(t))\zeta(t)\right)$$

- Interest Rate Swap (IRS) with payment tenor T_i, $i = 1, \ldots, M$:

$$V_S(t, x_t) = \phi\left(Z(t, x_t; T) - Z(t, x_t; T_M) - K\sum_{i=1}^{M} \Delta T_i Z(t, x_t; T_i)\right)$$

- European Swaption (on the previous IRS):

$$\begin{aligned} V_E(t, x_t) = {} & \phi Z(t, x_t, T)\mathcal{N}\left(-\phi\frac{y_T^*}{\sqrt{\zeta(T) - \zeta(t)}}\right) \\ & - \phi Z(t, x_t, T_M)\mathcal{N}\left(-\phi\frac{y_T^* + (H(T_M) - H(T))(\zeta(T) - \zeta(t))}{\sqrt{\zeta(T) - \zeta(t)}}\right) \\ & - \phi K\sum_{i=1}^{M} \Delta T_i Z(t, x_t, T_i)\mathcal{N}\left(-\phi\frac{y_T^* + (H(T_i) - H(T))(\zeta(T) - \zeta(t))}{\sqrt{\zeta(T) - \zeta(t)}}\right) \end{aligned}$$

LGM: Bermudan swaptions

- No closed-form solution.
- Valuation of a related product, i.e., Cancellable IRS (cIRS):

$$V_C^p = V_S^p - V_B^p,$$
$$V_C^r = V_S^r - V_B^r,$$

- The price of the Cancellable IRS is

$$\frac{V_C^p(t,x_t)}{N(t,x_t)} = \sup_{\tau\in\{T_i/T_i>t\}} \mathbb{E}\left[\max\left(\frac{V_S^p(\tau,x_\tau)}{N(\tau,x_\tau)},0\right)\right],$$
$$\frac{V_C^r(t,x_t)}{N(t,x_t)} = \sup_{\tau\in\{T_i/T_i>t\}} \mathbb{E}\left[\max\left(\frac{V_S^r(\tau,x_\tau)}{N(\tau,x_\tau)},0\right)\right].$$

- This formulation enables the use of dynamic programming and backward induction to determine the optimal cancellation policy and, then, solve the problem.

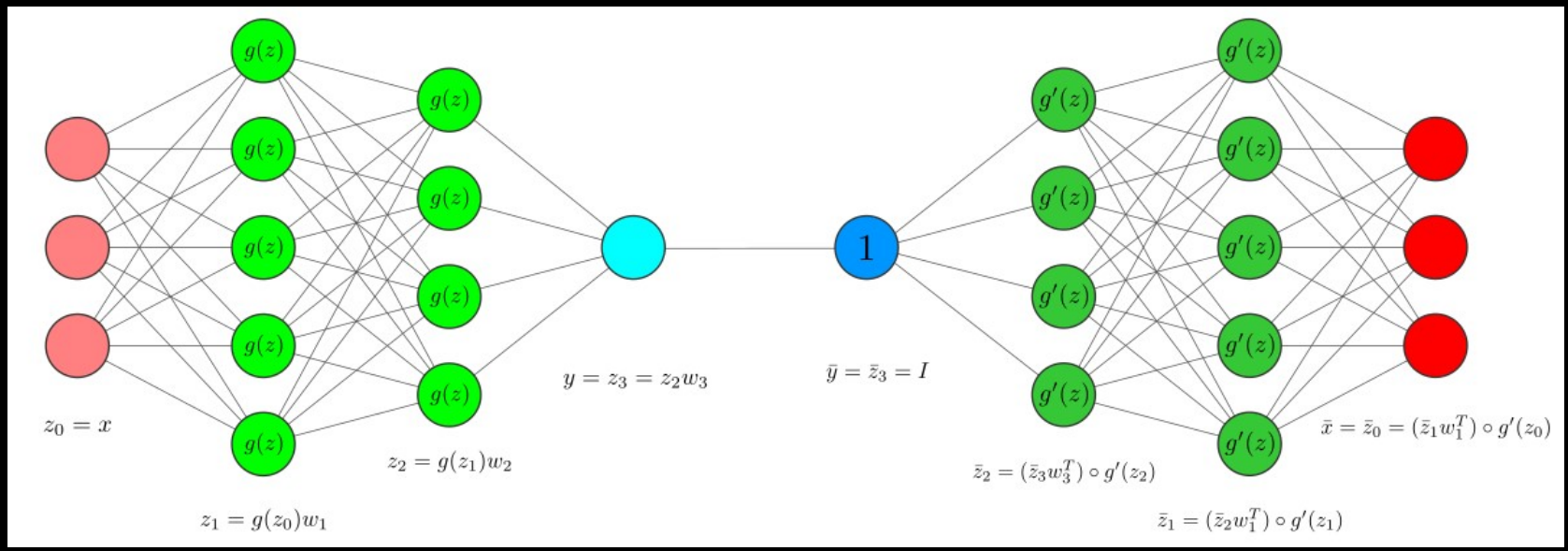
Deep learning approach for Cancellable IRS

citic

- We smartly combine three (individually) successful ANN components:
  - Differential Machine Learning $\rightarrow$ DANN
  - Training with *sampled* labels
  - Joint learning
- Cancellation policy: sequence of interconnected DANNs.
- Each DANN, associated with a cancellation opportunity, is used (once trained) to compute the labels of the next DANN.
- Then, the labels for the DANN of each cancellation opportunity depends on the estimations of all the “previous” DANNs.
- This specific design gets inspiration from the classical methods based on dynamic programing and regression.
- An additional DANN approximates the final price given a newly generated samples, using joint learning feature.

Differential Machine Learning (DANN)

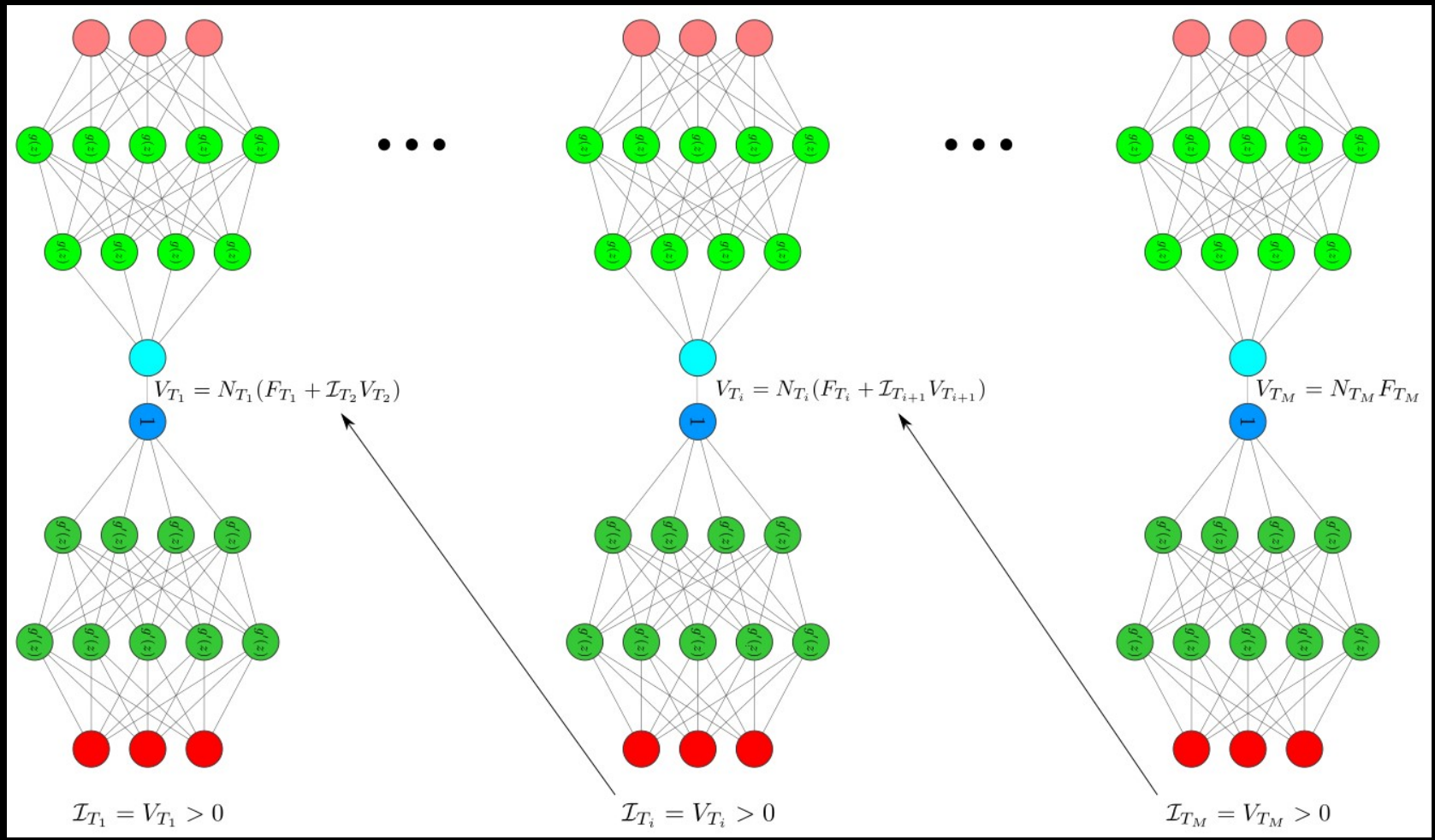
- Enlarge the network to consider (benefit from) the differentials of the output w.r.t the inputs.
- It requires the availability of those differentials.
- The loss function needs to incorporate both components.



Differential Machine Learning with ANNs.

Backward DANNs

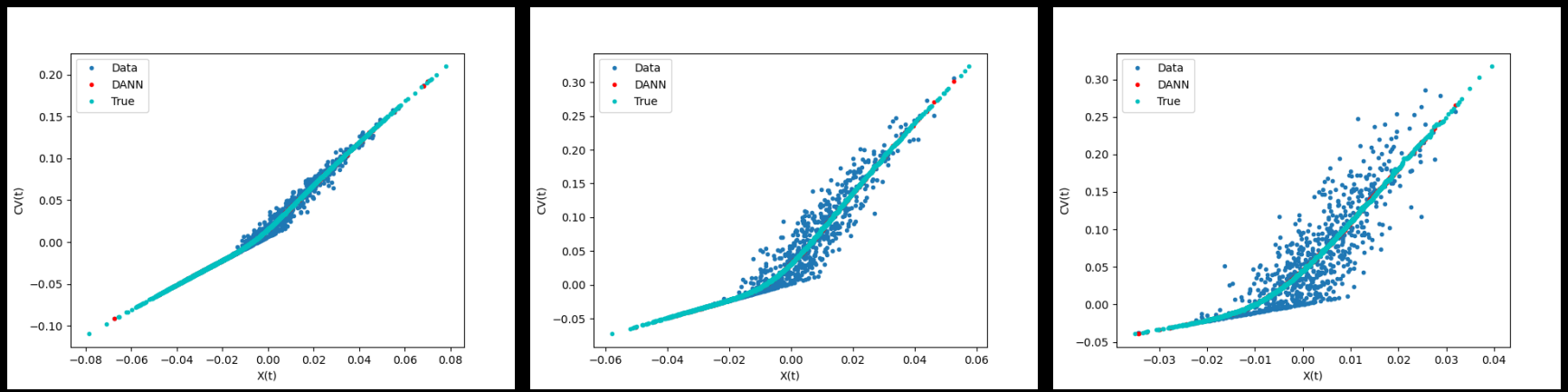
- Recursive DANN structured design.



Sequence of DANNs encapsulating the exercise policy.

Sampled (labels) payoffs

- Training the DANNs with highly noisy labels.
- The differential labels (also noisy) are obtained by AAD.



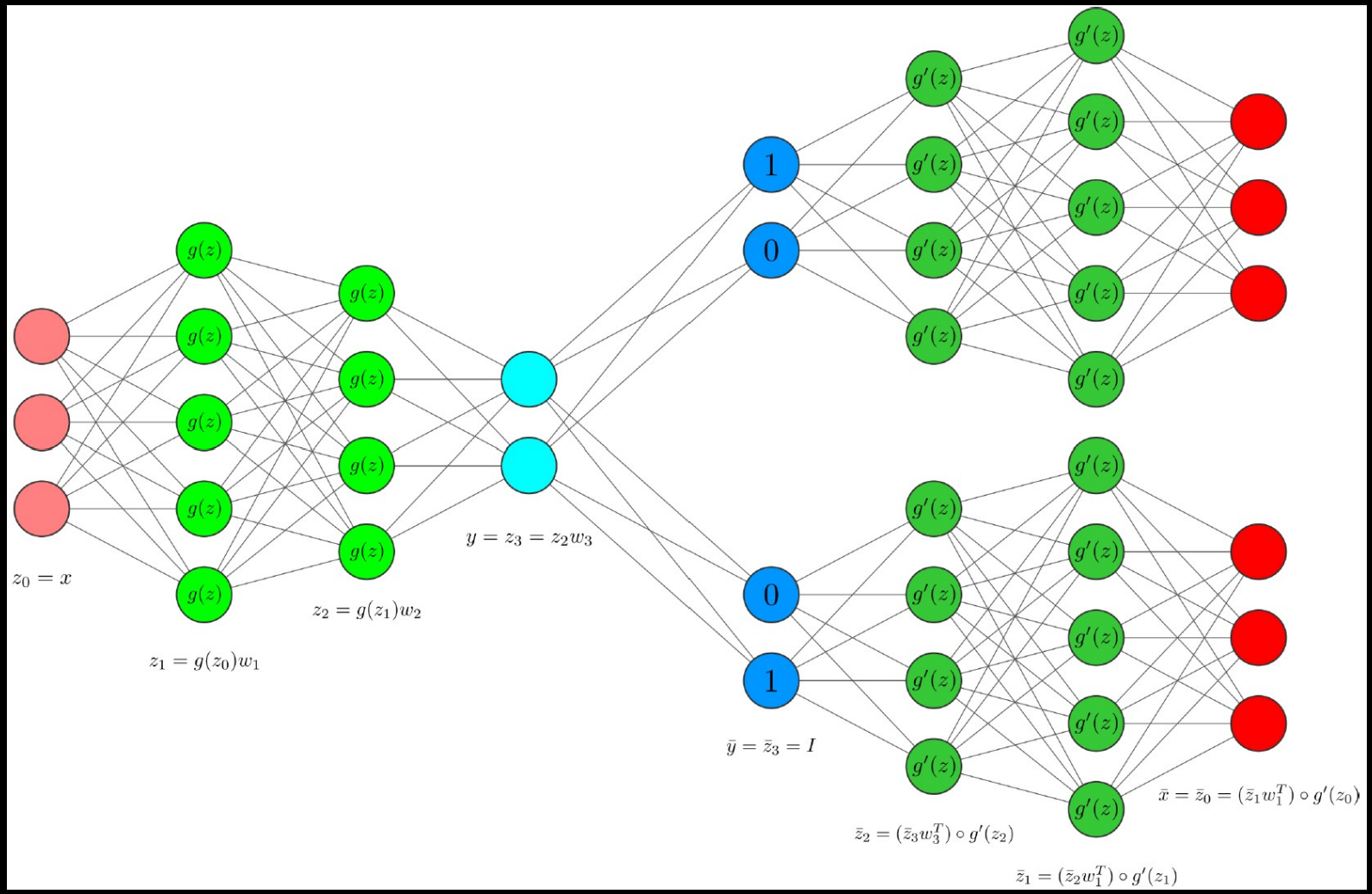
(a) $t = 7$ (b) $t = 4$ (c) $t = 1$

Sampled payoffs and Backward DANN approximation

Joint learning

- The cIRS price is estimated jointly with related financial products.
- Additional outputs include coterminal European swaptions.
- Bermudan derivatives can be viewed as combinations of European counterparts whose maturities coincide with each of the cancellation times (coterminal swaptions).
- Coterminal swaption prices are available in closed form under the LGM model.
- These exact prices provide noise-free labels, unlike sampled cIRS payoffs.
- Joint learning leverages this additional information to improve accuracy at similar computational cost.

Joint learning (with DANNs)



DANN structure considering multiple outputs, i.e., integrating the joint learning approach.

Training set generation (I)

- Training data quality depends on two key factors: the input domain and the sampling strategy.
- The input domain is typically defined according to market conditions and financial expertise.
- Sampling is critical: targeted sampling of relevant or high-error regions can outperform uniform designs under a fixed computational budget.
- Financial constraints and parameter relationships help exclude unrealistic market scenarios.
- For each generated input, a single Monte Carlo path is simulated to evolve the LGM model.

Training set generation (II)

- **Mean reversion**, $\kappa = \mathcal{U}(l_\kappa, u_\kappa)$.
- **Discount factors**: interest rate curves $\rightarrow$ discount curves

$$R(t) = \beta_0 G_0(t) + \beta_1 G_1(t) + \beta_2 G_2(t),$$

where

$$\beta_0 = \mathcal{U}(l_0, u_0), \quad \beta_1 = \mathcal{U}(l_1, u_1), \quad \beta_2 = \mathcal{U}(l_2, u_2), \quad \tau = \mathcal{U}(l_\tau, u_\tau).$$

The discount curve is constructed as $D(t) = \exp\left(-\int_0^t R(s)ds\right)$.

- **Fixed rate**. The fixed rate K, is perturbed from the ATM level:

$$K = ATM + \Delta K,$$

where

$$ATM = \frac{1 - D(T_M)}{\sum_{i=1}^{M} \Delta T_i D(T_i)},$$

with $\Delta K \in \mathcal{U}(l_K, u_K)$.

Training set generation (III)

- **LGM volatility**, α: assumed to be a piecewise constant function of the time with a dependency on κ (as observed in the market).
- How? First, Rebonato's parameterization

$$h(t) = (a + bt)\exp(-ct) + d$$

  Then, we choose implied volatilities as $\Sigma_j = h\left(\frac{t_{j-1}+t_j}{2}\right)$

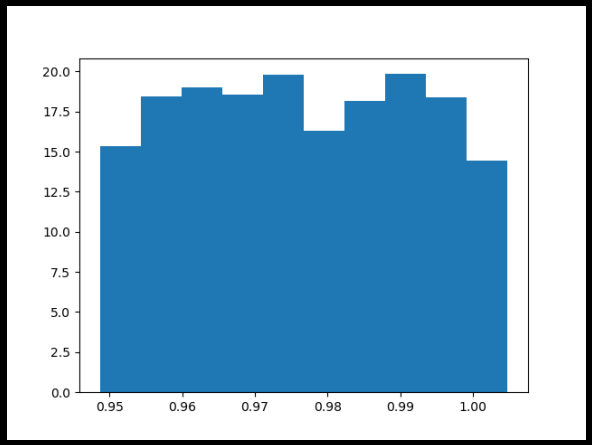
- As $\Sigma_j^2 \Delta t_j = \alpha_j^2 \int_{t_{j-1}}^{t_j} \exp\left(-2\kappa(t_j - s)\right) ds$, then $\alpha_j^2 = 2\kappa \frac{\Sigma_j^2 \Delta t_j}{1-\exp(-\kappa \Delta t_j)}$
- Finally,

$$\alpha(t) = \begin{cases} \alpha_1, & t \in (t_0, t_1], \\ \alpha_2, & t \in (t_1, t_2], \\ \vdots & \vdots \\ \alpha_J, & t \in (t_{J-1}, t_J]. \end{cases}$$

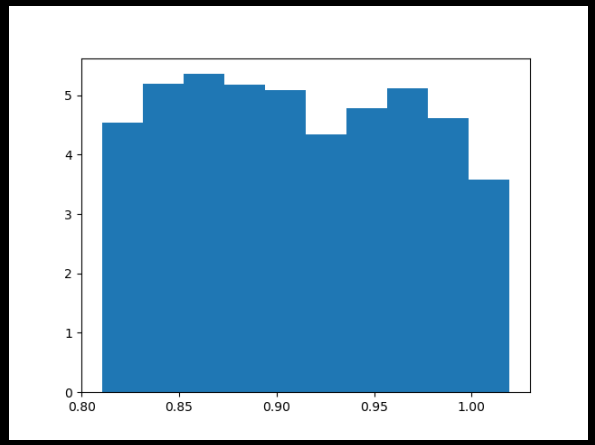
- The values of the Rebonato's model parameters are sampled by

$$a = \mathcal{U}(l_a, u_a), \quad b = \mathcal{U}(l_b, u_b), \quad c = \mathcal{U}(l_c, u_c), \quad d = \mathcal{U}(l_d, u_d).$$

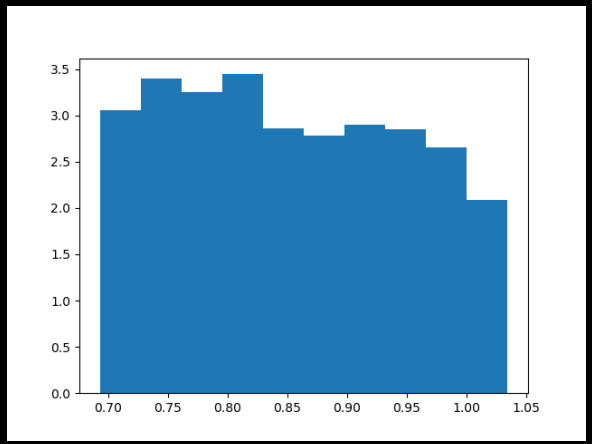
Training set generation (IV)



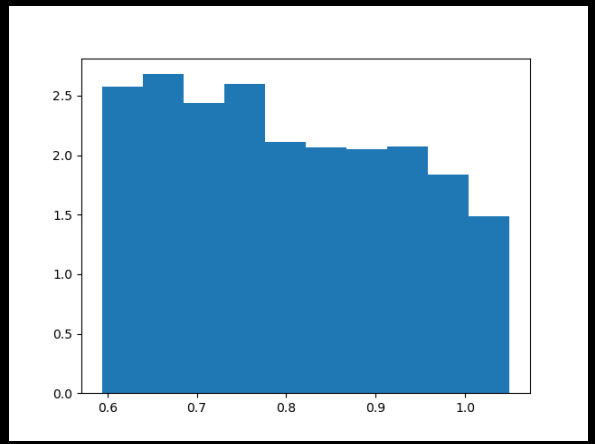
(a) $t = 1$.



(b) $t = 4$.



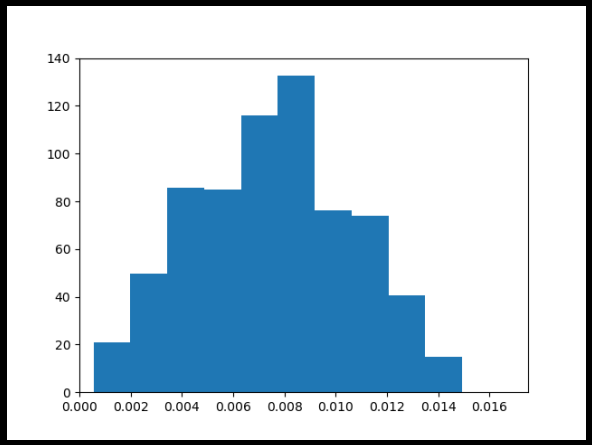
(c) $t = 7$.



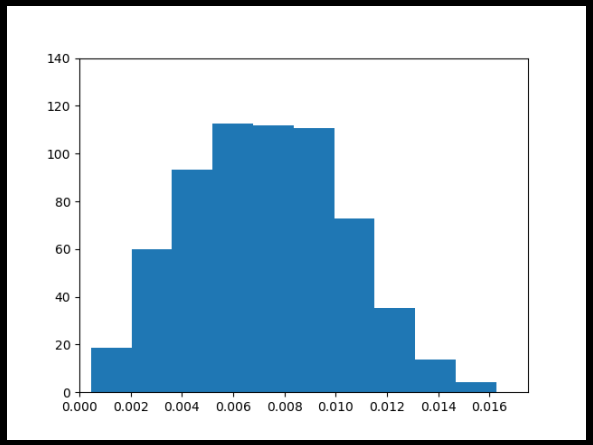
(d) $t = 10$.

Histograms of the discount factors at different time instants.

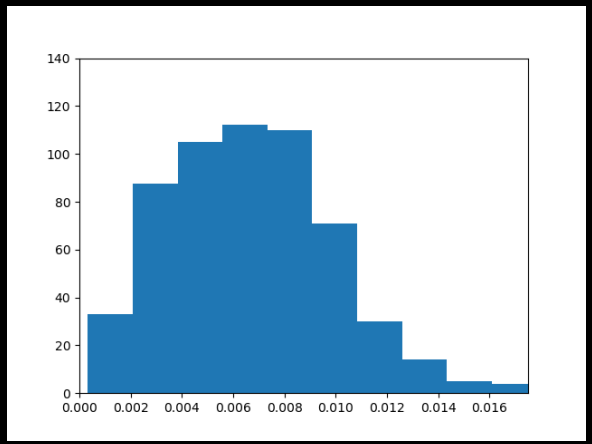
Training set generation (and V)



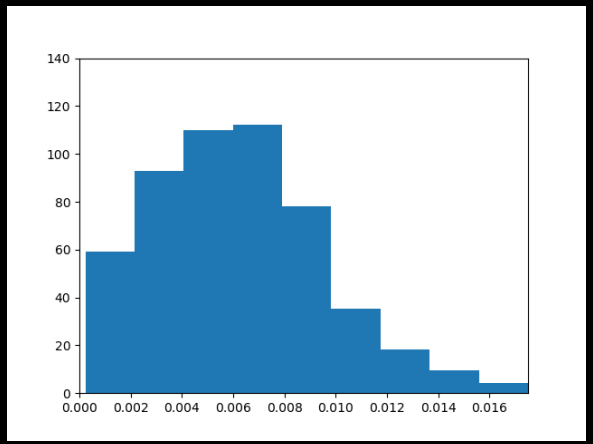
(a) α_1.



(b) α_2.



(c) α_3.



(d) α_4.

Histograms of the volatilites.

Numerical experiments

- Incremental testing in terms of the inputs' domain.
- The results are presented in the form of *differences*' histograms, including an *interquantile confidence interval*.
- The ground truth values (prices of cIRS) of the validation set are computed by a highly converged Monte Carlo pricer (acc: $\sim$ 1 bp)
- ANNs' hyperparameters configuration:

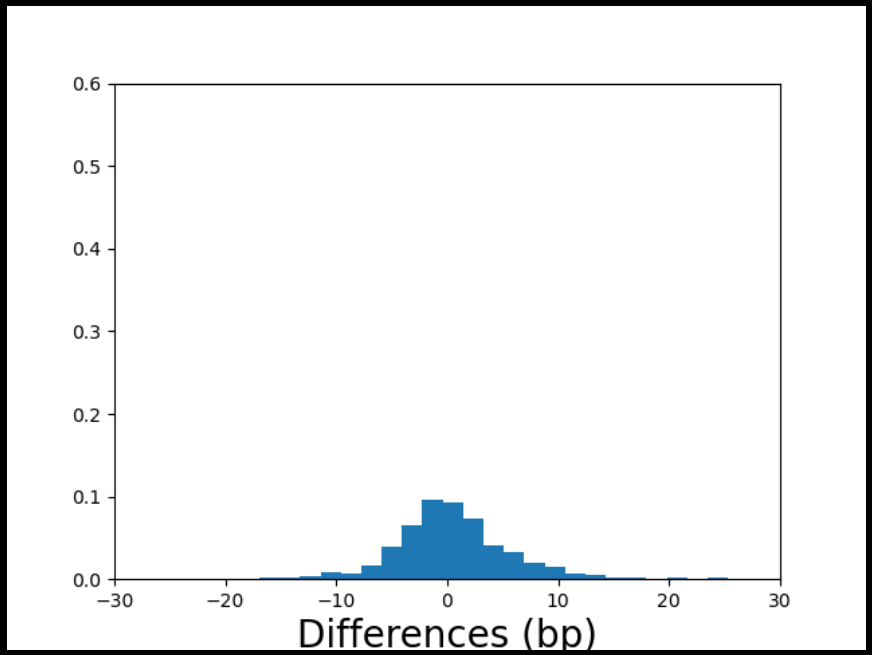
Hyperparameter	Value
Layers	4
Neurons	32
Epochs	128
Batch size	4096

Test base cases

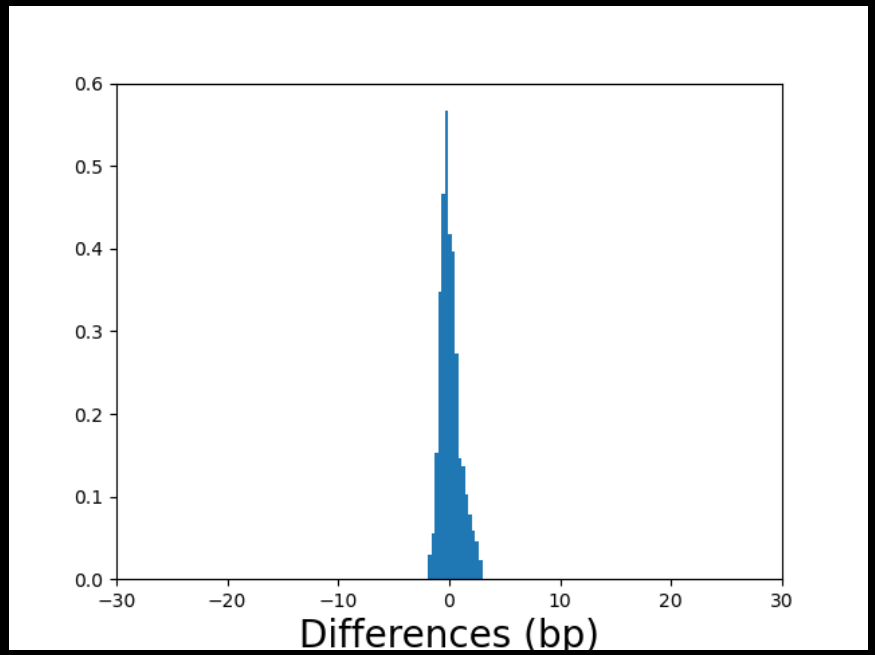
Test Case I	Test Case II
$l_\kappa = -0.05, \quad u_\kappa = 0.1$	$l_\kappa = -0.05, \quad u_\kappa = 0.1$
$l_a = -10^{-5}, \quad u_a = 10^{-5}$	$l_a = 10^{-5}, \quad u_a = 0.0075$
$l_b = -10^{-5}, \quad u_b = 10^{-5}$	$l_b = 0, \quad u_b = 0.0005$
$l_c = -10^{-5}, \quad u_c = 10^{-5}$	$l_c = 0, \quad u_c = 0.25$
$l_d = 0.0075 - 10^{-5}, \quad u_d = 0.0075 + 10^{-5}$	$l_d = 10^{-5}, \quad u_d = 0.0075$
$l_o = 0.02 - 10^{-5}, \quad u_o = 0.02 + 10^{-5}$	$l_o = 0.02 - 10^{-5}, \quad u_o = 0.02 + 10^{-5}$
$l_1 = -10^{-5}, \quad u_1 = 10^{-5}$	$l_1 = -10^{-5}, \quad u_1 = 10^{-5}$
$l_2 = -10^{-5}, \quad u_2 = 10^{-5}$	$l_2 = -10^{-5}, \quad u_2 = 10^{-5}$
$l_\tau = 1 - 10^{-5}, \quad u_\tau = 1 + 10^{-5}$	$l_\tau = 1 - 10^{-5}, \quad u_\tau = 1 + 10^{-5}$
$l_K = -10^{-5}, \quad u_K = 10^{-5}$	$l_K = -10^{-5}, \quad u_K = 10^{-5}$

Test Case III	Test Case IV
$l_\kappa = -0.05, \quad u_\kappa = 0.1$	$l_\kappa = -0.05, \quad u_\kappa = 0.1$
$l_a = 10^{-5}, \quad u_a = 0.0075$	$l_a = 10^{-5}, \quad u_a = 0.0075$
$l_b = 0, \quad u_b = 0.0005$	$l_b = 0, \quad u_b = 0.0005$
$l_c = 0, \quad u_c = 0.25$	$l_c = 0, \quad u_c = 0.25$
$l_d = 10^{-5}, \quad u_d = 0.0075$	$l_d = 10^{-5}, \quad u_d = 0.0075$
$l_o = -0.005, \quad u_o = 0.05$	$l_o = -0.005, \quad u_o = 0.05$
$l_1 = 0, \quad u_1 = 0.001$	$l_1 = 0, \quad u_1 = 0.001$
$l_2 = 0, \quad u_2 = 0.01$	$l_2 = 0, \quad u_2 = 0.01$
$l_\tau = 0.01, \quad u_\tau = 2$	$l_\tau = 0.01, \quad u_\tau = 2$
$l_K = -10^{-5}, \quad u_K = 10^{-5}$	$l_K = -0.01, \quad u_K = 0.01$

Impact of joint learning (I)



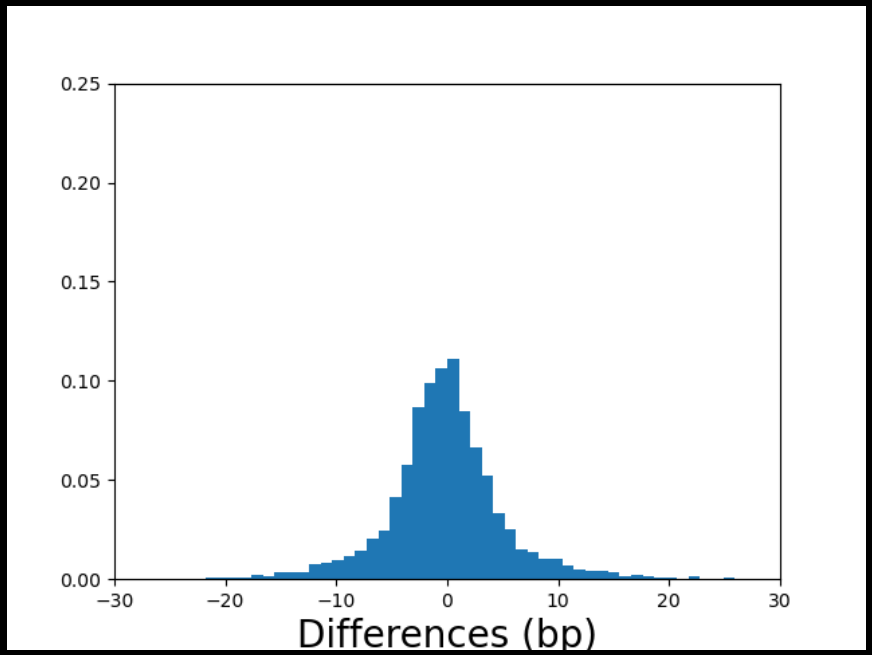
(a) $(Q_{10}Q_{90}) = (-5.7, 7.0)$.



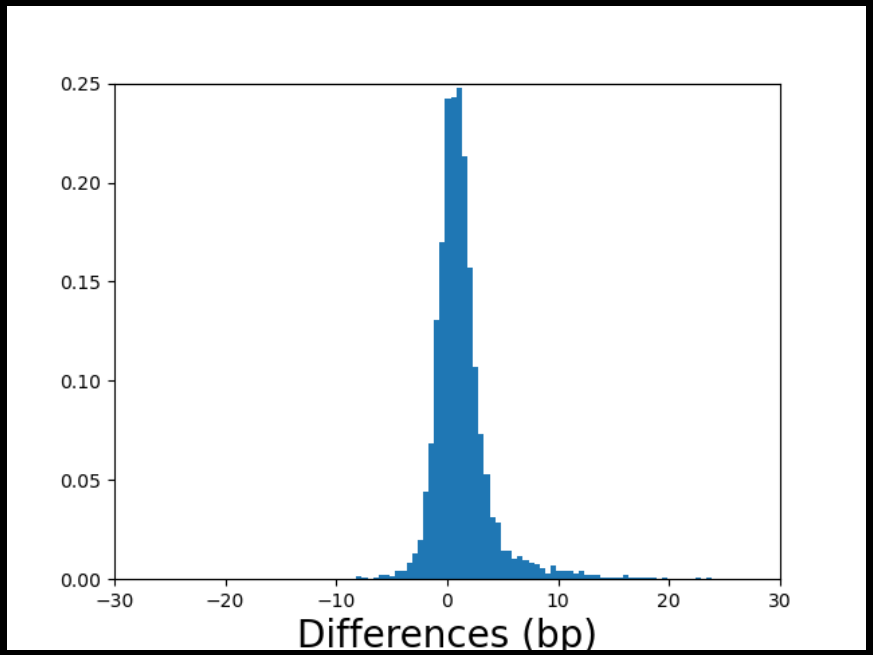
(b) $(Q_{10}, Q_{90}) = (-0.8, 1.4)$.

Pricing differences in basis points of Test Case I: Plain DANN (left) and DANN with joint learning (right).

Impact of joint learning (II)



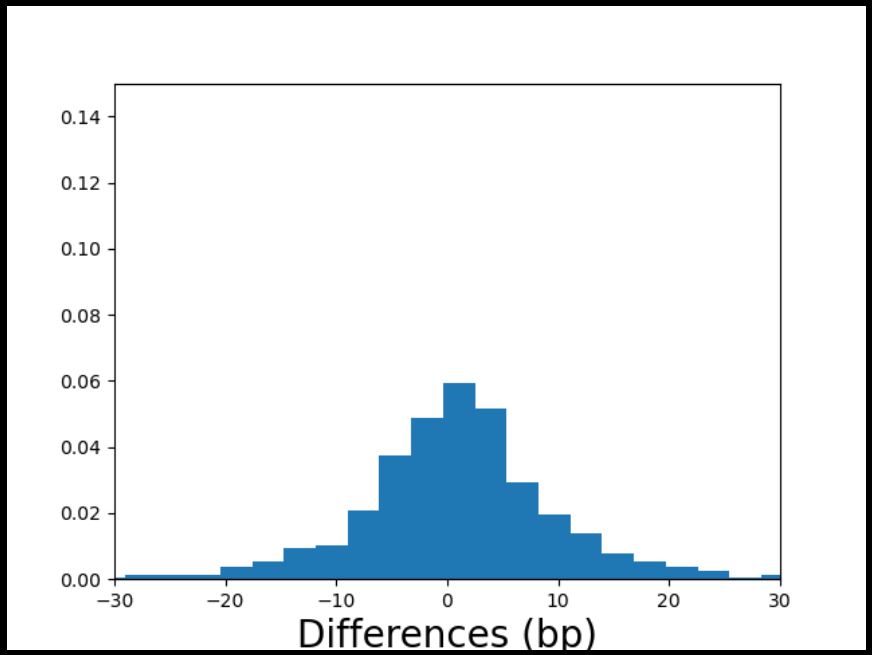
(a) $(Q_{10}, Q_{90}) = (-6.0, 5.7)$.



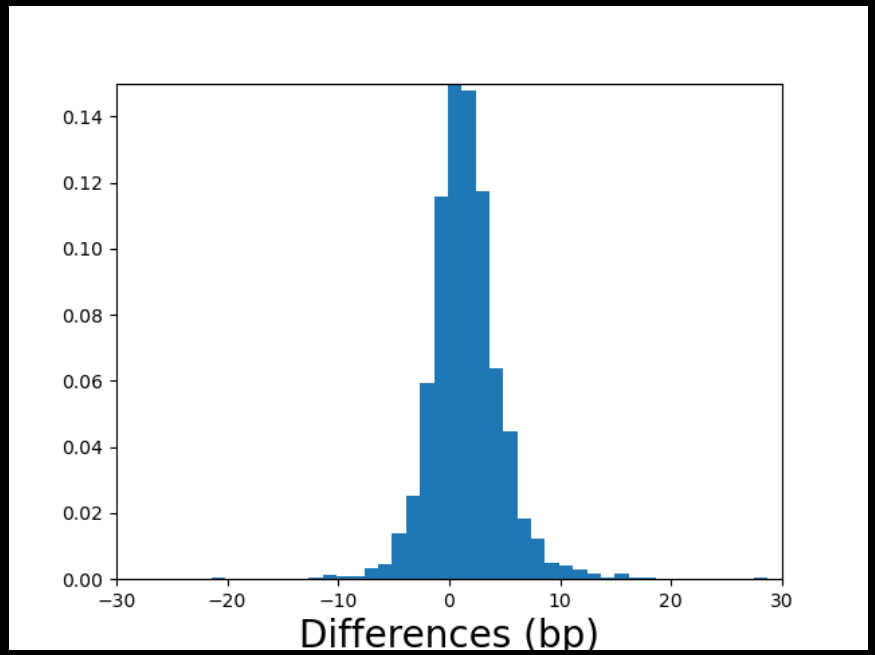
(b) $(Q_{10}, Q_{90}) = (-1.0, 3.7)$.

Pricing differences in basis points of Test Case II: Plain DANN (left) and DANN with joint learning (right).

Impact of joint learning (III)



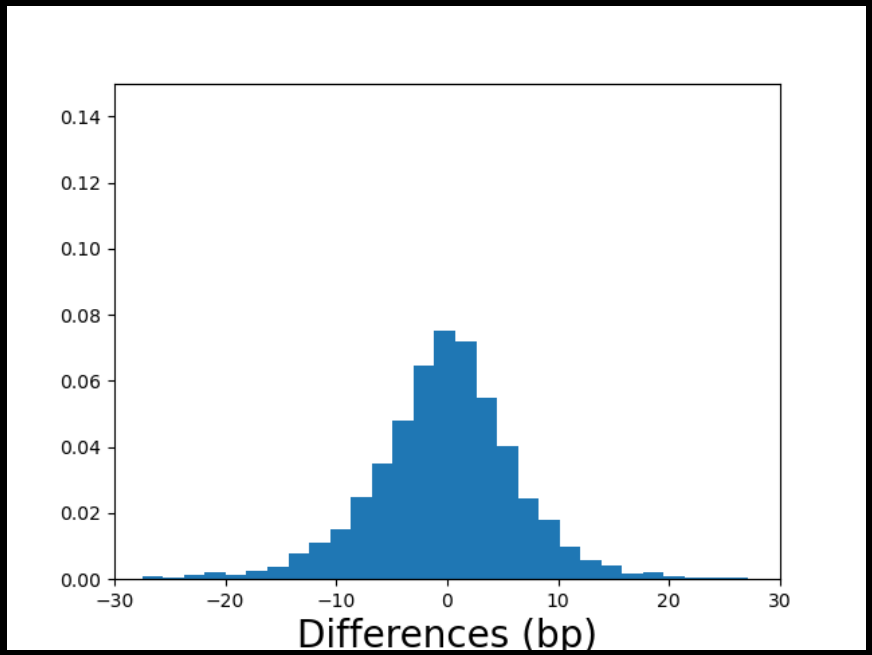
(a) $(Q_{10}, Q_{90}) = (-10.0, 12.4)$.



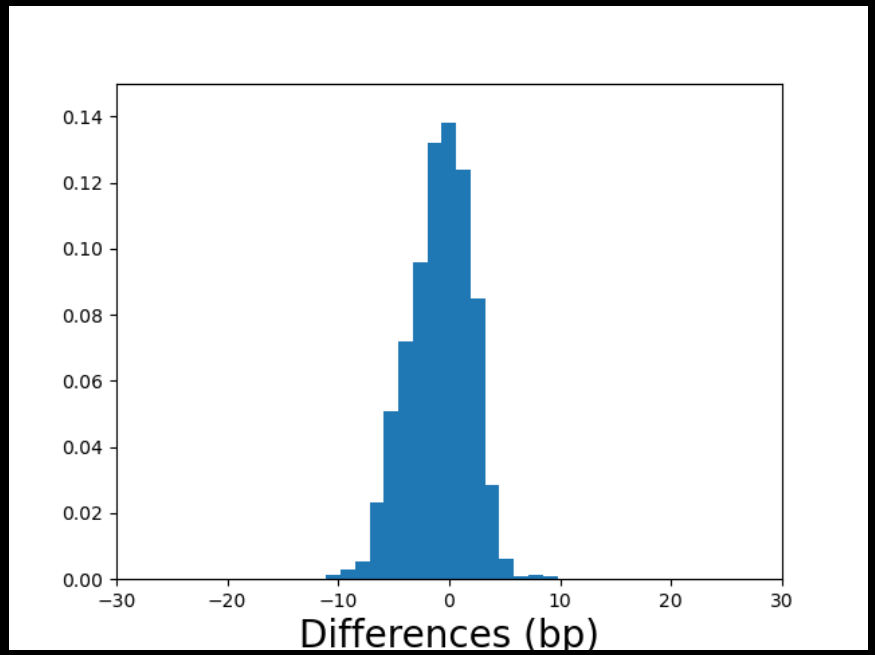
(b) $(Q_{10}, Q_{90}) = (-1.9, 5.1)$.

Pricing differences in basis points of Test Case III: Plain DANN (left) and DANN with joint learning (right).

Impact of joint learning (IV)



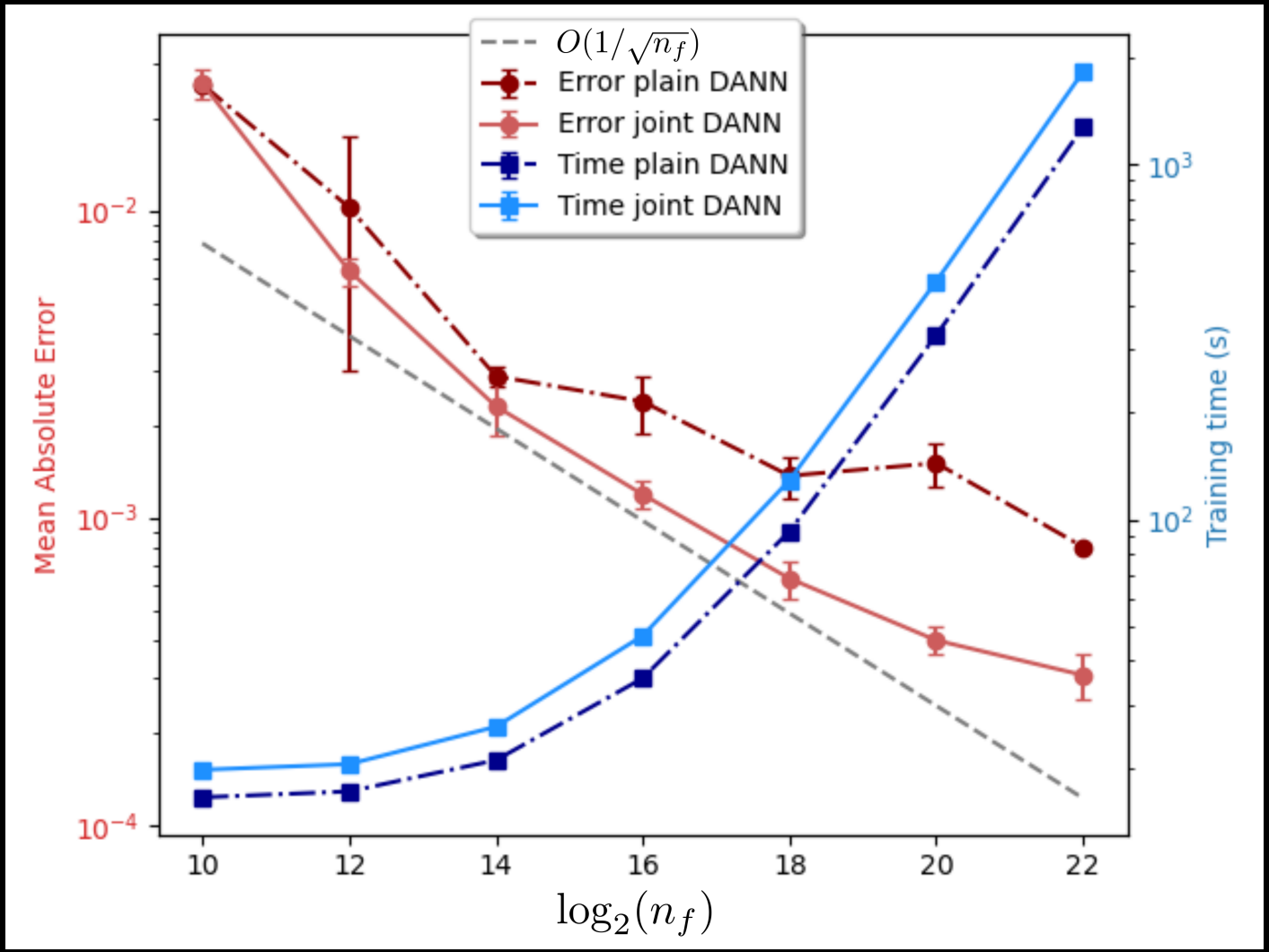
(a) $(Q_{10}, Q_{90}) = (-8.1, 7.4)$.



(b) $(Q_{10}, Q_{90}) = (-4.8, 2.5)$.

Pricing differences in basis points of Test Case IV: Plain DANN (left) and DANN with joint learning (right).

Impact of joint learning (V)



Impact of joint learning (VI)

- Error distributions are centered around zero, indicating unbiased DANN predictions.
- Discount-factor and volatility inputs increase the complexity of the approximation task.
- Including the strike spread (Test Case IV) slightly improves accuracy, mainly due to far-from-ATM options, with mild skewness appearing in the error distributions.
- Joint learning consistently provides the best results, reducing both mean error and dispersion by more than 50%.

Conclusions

- A deep learning framework for Bermudan swaption valuation is proposed.
- The approach extends standard ANNs with advanced learning enhancements.
- Key components include sampled-payoff labels and differential learning.
- A novel joint learning strategy is introduced for quantitative finance.
- Related financial products are learned simultaneously to improve accuracy at similar computational cost.
- Numerical experiments demonstrate significant gains from the joint learning approach.
- The incorporation of the industry view was critical in training set generation and joint learning.
- The follow-up feedback received from the industrial partner was very positive (we keep collaborating).

References

[1] Patrick S. Hagan. *Evaluating and hedging exotic swap instruments via LGM*. Available at ResearchGate. 2002.

[2] Brian Huge and Antoine Savine. *Differential machine learning*. Available at ArXiv: 2005.02347. 2020.

[3] Francisco Gómez-Casanova et al. “Deep joint learning valuation of Bermudan swaptions”. In: *International Journal of Computer Mathematics* 102.7 (2025), pp. 913–942.

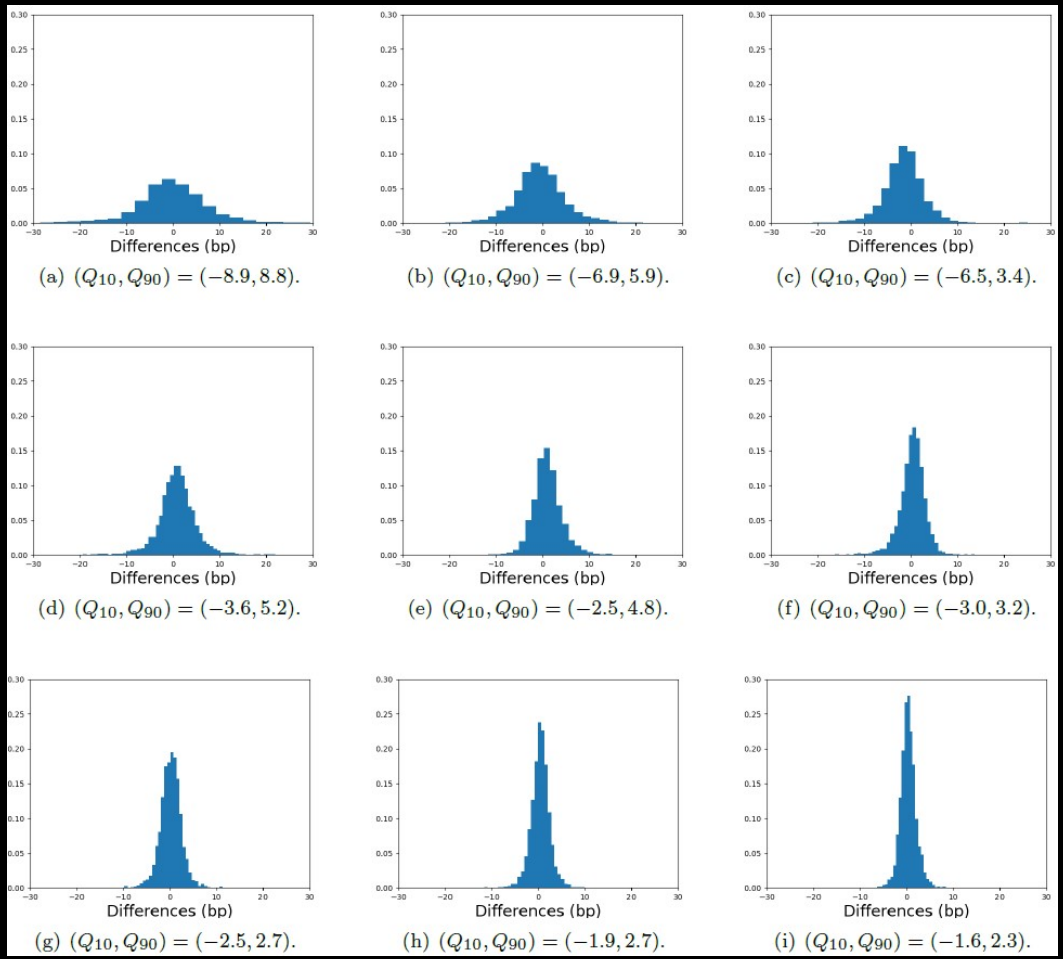
Acknowledgements & Questions

Moltes gràcies!

More:
alvaro.leitao@udc.gal
alvaroleitao.github.io

Thanks to the funding received from the MCIU of Spain through the Ramón y Cajal 2022 and PID2022-141058OB-I00 grants, and from the Xunta de Galicia through the programs ED451C 2022/047 and ED431F 2025/032, as well as the support from CITIC, as a centre accredited for excellence within the Galician University System and a member of the CIGUS Network, receiving subsidies from the Xunta de Galicia, additionally, co-financed by the EU through the FEDER Galicia 2021-27 operational program ED451G 2023/01.

Impact of number of samples and MC paths/sample (I)



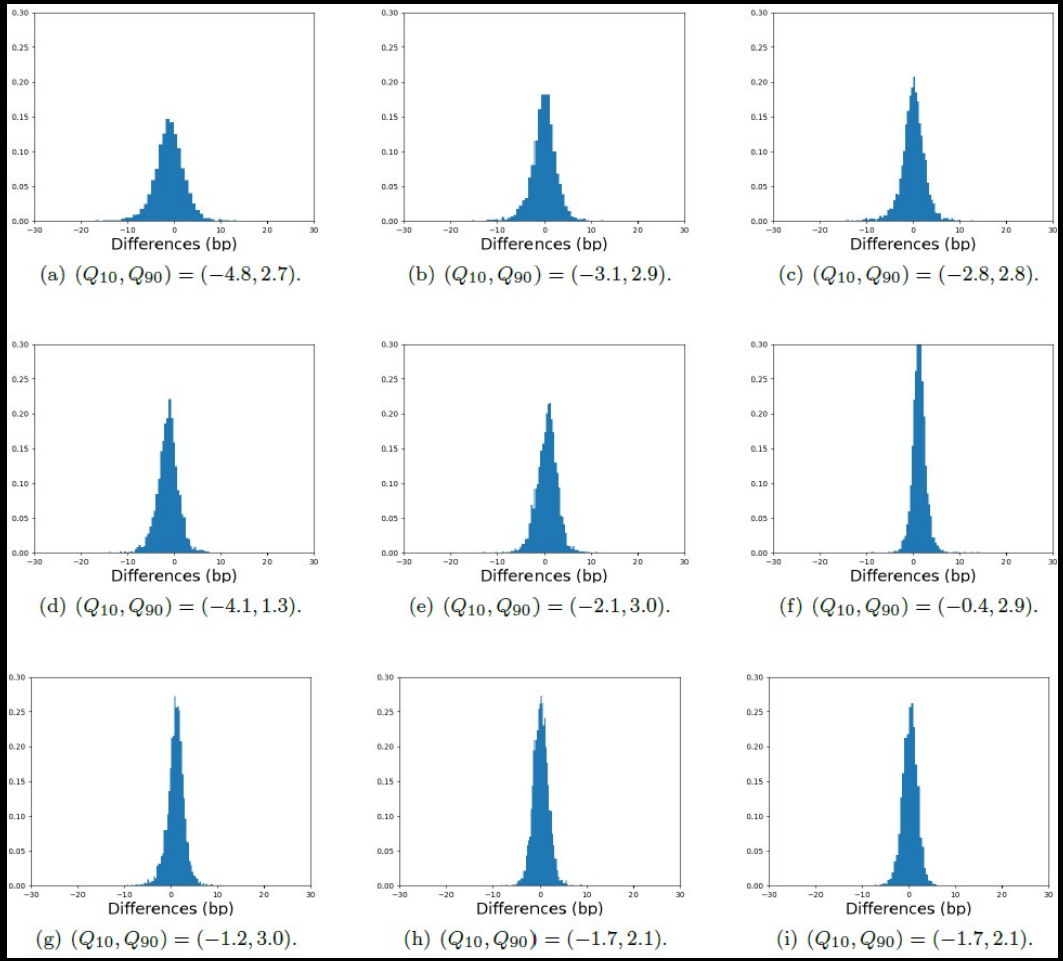
(a) $(Q_{10}, Q_{90}) = (-8.9, 8.8)$. (b) $(Q_{10}, Q_{90}) = (-6.9, 5.9)$. (c) $(Q_{10}, Q_{90}) = (-6.5, 3.4)$.

(d) $(Q_{10}, Q_{90}) = (-3.6, 5.2)$. (e) $(Q_{10}, Q_{90}) = (-2.5, 4.8)$. (f) $(Q_{10}, Q_{90}) = (-3.0, 3.2)$.

(g) $(Q_{10}, Q_{90}) = (-2.5, 2.7)$. (h) $(Q_{10}, Q_{90}) = (-1.9, 2.7)$. (i) $(Q_{10}, Q_{90}) = (-1.6, 2.3)$.

Plain DANN: Columns: $n_f = 2^{22}$ (left), $n_f = 2^{23}$ (central), $n_f = 2^{24}$ (right); Rows: $n_{MC} = 1$ (top), $n_{MC} = 4$ (middle), $n_{MC} = 16$ (bottom).

Impact of number of samples and MC paths/sample (II)



(a) $(Q_{10}, Q_{90}) = (-4.8, 2.7)$. (b) $(Q_{10}, Q_{90}) = (-3.1, 2.9)$. (c) $(Q_{10}, Q_{90}) = (-2.8, 2.8)$.

(d) $(Q_{10}, Q_{90}) = (-4.1, 1.3)$. (e) $(Q_{10}, Q_{90}) = (-2.1, 3.0)$. (f) $(Q_{10}, Q_{90}) = (-0.4, 2.9)$.

(g) $(Q_{10}, Q_{90}) = (-1.2, 3.0)$. (h) $(Q_{10}, Q_{90}) = (-1.7, 2.1)$. (i) $(Q_{10}, Q_{90}) = (-1.7, 2.1)$.

Joint DANN: Columns: $n_f = 2^{22}$ (left), $n_f = 2^{23}$ (central), $n_f = 2^{24}$ (right); Rows: $n_{MC} = 1$ (top), $n_{MC} = 4$ (middle), $n_{MC} = 16$ (bottom).

Impact of number of samples and MC paths/sample (and III)

- As expected, systematically increase the number of samples provided to the DANN improves the predictions, although the reduction of the interquantile intervals, i.e., in the deviations' variance, is rather limited.
- In contrast to the previous point, we again observe that the DANN trained relying on the joint learning approach provides more accurate estimations, significantly reducing the variance.
- The effect of including more Monte Carlo paths per sample presents the expected behavior, i.e., when the number of paths is multiplied by four the error is approximately halved (according to the theoretical convergence rate of Monte Carlo methods, $n^{-1/2}$).
- When most of the differences fall below ± 3 basis points, a certain level of saturation is observed meaning that considering either more samples or more Monte Carlo paths per sample no longer reduces the deviations in the predictions (or the reduction results to be negligible).