Inverse Pricing for the Implied Volatility Surface using Physics-Informed Neural Networks

Alex Yang

December 6, 2025

Abstract

This project studies the inverse problem of recovering a smooth, continuous and arbitrage-free implied volatility (IV) surface $\sigma(K,T)$ from sparse and noisy market prices of European options. I propose a Physics-Informed Neural Network (PINN) framework in which a neural network parameterizes the IV surface, the Black–Scholes formula serves as a differentiable forward map from volatility to price, and no-arbitrage constraints are embedded as soft penalties in the loss. Using SPY options data from OptionMetrics (WRDS), I show that a two-phase "warm-up" training strategy is essential: first fit prices without constraints, then gradually introduce calendar- and butterfly-arbitrage penalties. The resulting surface achieves high pricing accuracy (RMSE around \$1.76 on the calibration day), zero calendar-arbitrage violations on a dense grid, and strong zero-shot and fine-tuned generalization to neighboring trading days.

1 Introduction and Background

The implied volatility (IV) surface $\sigma(K,T)$ is a central object in equity and index options markets. For each strike K and time-to-maturity T, the Black–Scholes formula maps a volatility input to a theoretical option price. In practice, the market quotes only a finite and noisy set of mid prices $C^{\mathrm{mkt}}(K_i,T_i)$; the IV surface itself is not directly observable. Recovering a smooth, economically meaningful $\sigma(K,T)$ from these discrete prices is therefore a classical ill-posed inverse problem.

In this project, I frame IV-surface calibration as a regularized inverse problem solved via a Physics-Informed Neural Network (PINN). A multilayer perceptron (MLP) parameterizes $\sigma_\theta(K,T)$; a differentiable Black–Scholes layer turns this into model prices $C_\theta(K,T)$; and the training loss balances data fidelity against financial "physics" in the form of no-arbitrage constraints:

$$\min_{\theta} \mathcal{L}(\theta) = L_{\mathrm{price}}(\theta) + \lambda_{\mathrm{cal}} L_{\mathrm{calendar}}(\theta) + \lambda_{\mathrm{but}} L_{\mathrm{butterfly}}(\theta). \tag{1}$$

The calendar term penalizes violations of the requirement that total variance $\sigma^2 T$ be non-decreasing in T; the butterfly term penalizes violations of convexity of the call price in K, which is equivalent to non-negativity of the risk-neutral density.

Empirically, I find that directly enforcing these constraints from epoch 0 often causes the network to collapse to an almost flat surface that trivially satisfies derivative-based penalties while ignoring the data. A key methodological contribution of this project is a simple but effective "warm-up" strategy: train first on prices only, then smoothly turn on the physics penalties with small weights.

2 Literature Review

In recent years, deep learning has emerged as a powerful tool for option pricing and, more broadly, for modeling financial derivatives. For example, Hernandez (2016) train feedforward networks to approximate option prices as functions of model parameters, demonstrating that once trained, the network can reproduce Monte Carlo or finite-difference prices at a fraction of the computational cost. In this setting, the goal is primarily numerical acceleration: the neural network replaces a slow PDE solver or simulation routine, but does not fundamentally change the structure of the pricing problem.

Building on the idea of Physics-Informed Neural Networks (PINNs), introduced by Raissi et al. (2019), several authors have applied it to option pricing problems. For instance, Dhiman and Hu (2023) train neural networks to approximate option prices while penalizing the Black–Scholes PDE residual and enforcing payoff boundary conditions. In their setup, the network takes underlying price and time as inputs and outputs the option price; the physics term in the loss drives the network toward solutions that satisfy the pricing PDE, while observed market or simulated prices serve as data anchors.

My project is inspired by this line of PINN-based option pricing but differs in emphasis and architecture. Instead of learning the option price surface directly, I parameterize the implied volatility surface $\sigma(K, T)$ with a neural network and use the Black–Scholes formula as a differentiable forward map to obtain prices. The "physics" embedded in the loss are no-arbitrage properties that are closely related to the pricing PDE but easier to express on market-relevant quantities: calendar monotonicity of total variance and convexity of call prices in strike (butterfly no-arbitrage).

3 Data and Preprocessing

3.1 Data source and selection

I use option data for the SPDR S&P 500 ETF (ticker SPY) from the OptionMetrics database via WRDS (2025). The primary calibration date is 3 January 2023; I also extract data for 4 and 5 January 2023 to evaluate temporal generalization.

For each date, I obtain:

- European call options across a wide range of strikes K and maturities T,
- bid and ask quotes, from which I compute mid prices,
- underlying SPY close price S,
- zero-coupon yield curves, from which I interpolate the relevant risk-free rate $r(T)$,
- an estimate of the continuous dividend yield $q(T)$ for SPY.

3.2 Filtering and cleaning

Raw option chains contain many illiquid or economically uninformative contracts. I therefore apply the following filters:

- **Price filter**: discard options with mid price below \$0.01.
- **Maturity filter**: keep options with time-to-maturity between 7 days and 2 years. Very short-dated options are highly sensitive to microstructure noise; very long-dated options are often illiquid.
- **Moneyness filter**: restrict to contracts with moneyness K/S between 0.7 and 1.3. Deep ITM/OTM options tend to be noisy and exert disproportionate leverage on the calibration.

For each option, I compute the mid price as the simple average of bid and ask:

$$C^{\text{mkt}} = \frac{\text{bid} + \text{ask}}{2}.$$

3.3 Feature engineering

For each remaining contract, I construct the following features:

- **Moneyness & Log-moneyness**: $m = K/S$, $x = \log(K/S)$,
- **Time-to-maturity**: T in years, computed from calendar dates,
- **Risk-free rate**: $r = r(T)$ from the interpolated yield curve,
- **Dividend yield**: $q = q(T)$.

I standardize inputs (x, T) to zero mean and unit variance before feeding them into the neural network; this helps optimization.

3.4 Reference implied volatility

For diagnostics and visualization, I compute a reference implied volatility $\hat{\sigma}_i$ for each contract by numerically inverting the Black–Scholes call price:

$$C^{\mathrm{BS}}(S, K, T, \hat{\sigma}_i, r, q) = C_i^{\mathrm{mkt}}.$$

I use a robust root-finding method (Brent's method) initialized with a reasonable volatility bracket. These $\hat{\sigma}_i$ are *not* used as training targets, but provide a useful benchmark scatter plot and starting point for understanding the structure of the market IV surface.

4 Methodology

4.1 Neural network parameterization of the IV surface

I parameterize the implied volatility surface as a neural network

$$\sigma_\theta : (x, T) \mapsto \sigma_\theta(x, T) > 0,$$

where $x = \log(K/S)$ is log-moneyness. The positive range is enforced by using a Softplus activation at the output.

The architecture, implemented as `IVSurfaceMLP`, is:

- **Input layer**: 2 units (log-moneyness x and time-to-maturity T), after standardization.
- **Hidden layers**: three fully connected layers with widths $[128, 128, 64]$ and SiLU activations.
- **Output layer**: 1 unit with Softplus activation to enforce $\sigma_\theta(x, T) > 0$.
- **Dropout**: rate 0.05 on hidden layers for mild regularization.

This MLP is flexible enough to approximate realistic smile and term-structure shapes, while remaining small enough to train efficiently.

4.2 Black–Scholes forward map

Given the network output $\sigma_\theta(x, T)$, I compute model call prices via the Black–Scholes formula with continuous dividend yield:

$$\begin{aligned} C_\theta(S, K, T, r, q) &= C^{\mathrm{BS}}\big(S, K, T, \sigma_\theta(x, T), r, q\big) && (2) \\ &= S e^{-qT}\Phi(d_1) - K e^{-rT}\Phi(d_2), && (3) \end{aligned}$$

where

$$d_1 = \frac{\log(S/K) + (r - q + \frac{1}{2}\sigma_\theta^2)T}{\sigma_\theta\sqrt{T}}, \qquad d_2 = d_1 - \sigma_\theta\sqrt{T},$$

and Φ denotes the standard normal cumulative distribution function. This mapping is implemented in a differentiable way using automatic differentiation.

4.3 Physics-informed loss: data and no-arbitrage terms

The total loss function (1) consists of three terms.

4.3.1 Price data fidelity

The primary objective is to fit observed mid prices. Over the training set $\mathcal{D}$ of contracts indexed by i, I use a mean squared error:

$$L_{\text{price}}(\theta) = \frac{1}{|\mathcal{D}|}\sum_{i\in\mathcal{D}}\Big(C_\theta(S_i, K_i, T_i, r_i, q_i) - C_i^{\text{mkt}}\Big)^2. \tag{4}$$

4.3.2 Calendar-arbitrage penalty

In an arbitrage-free setting, total variance $w(K,T) = \sigma^2(K,T)\,T$ should be non-decreasing in T for each strike: $\partial_T w(K,T) \geq 0$. To encourage this, I define a penalty over a grid of collocation points (K_j, T_j):

$$L_{\text{calendar}}(\theta) = \frac{1}{|\mathcal{G}_{\text{cal}}|}\sum_{(K,T)\in\mathcal{G}_{\text{cal}}}\Big(\min\big\{0,\ \partial_T w_\theta(K,T)\big\}\Big)^2, \tag{5}$$

where $w_\theta(K,T) = \sigma_\theta^2(x,T)\,T$ and $\partial_T w_\theta$ is computed via automatic differentiation with respect to T. Negative values indicate a violation and are penalized quadratically.

4.3.3 Butterfly-arbitrage penalty

Absence of butterfly arbitrage requires that the call price be convex in strike, i.e.,

$$\frac{\partial^2 C}{\partial K^2}(K,T) \geq 0,$$

which is equivalent to non-negativity of the risk-neutral density. I therefore define

$$L_{\text{butterfly}}(\theta) = \frac{1}{|\mathcal{G}_{\text{but}}|}\sum_{(K,T)\in\mathcal{G}_{\text{but}}}\Big(\min\big\{0,\ \partial_K^2 C_\theta(S,K,T,r,q)\big\}\Big)^2, \tag{6}$$

where $\partial_K^2 C_\theta$ is obtained via second-order automatic differentiation of the Black–Scholes formula with respect to K. In practice, I evaluate this penalty on a dense grid of strikes and maturities that covers the support of the data.

4.4 Two-phase warm-up training strategy

Initial experiments revealed that if the physics penalties are activated from the first epoch with non-trivial weights, the optimizer often finds a nearly flat $\sigma_\theta(K,T)$ that satisfies the derivative-based constraints at the expense of large pricing errors. To avoid this "mode collapse," I adopt a two-phase schedule:

Phase A (Warm-up). For the first 30 epochs, I set $\lambda_{\text{cal}} = \lambda_{\text{but}} = 0$ and train solely on the price loss L_{price}. This allows the network to learn the coarse shape of the implied-volatility smiles and term structures directly from data, unconstrained by the physics penalties.

Phase B (Refinement). Starting from epoch 31, I turn on the physics penalties with small weights:

$$\lambda_{\text{cal}} = \lambda_{\text{but}} = 10^{-4}.$$

Training continues up to epoch 150. In this phase, the optimizer nudges the data-fit solution towards one that also satisfies calendar and butterfly constraints, with only a modest increase in pricing error.

I train using mini-batch stochastic gradient descent with the Adam optimizer. Hyperparameters such as learning rate and batch size are tuned empirically to achieve stable convergence.

5 Experiments and Results

5.1 In-sample calibration on January 3, 2023

I first calibrate the model on SPY options from 3 January 2023 and evaluate performance on a held-out validation subset from the same day. Table 1 summarizes the main quantitative results.

Metric (Validation Set)	**Phase A (Data Only)**	**Phase B (Final PINN)**
Price Loss (MSE)	≈ 2.09	3.44
Constraint Loss	N/A (no constraints)	1.91
Pricing RMSE	–	\$1.76
MAE	–	\$1.23
Calendar-arbitrage violation	–	0.01% (on test grid)

Table 1: Performance before and after introducing physics-informed constraints.

Several observations emerge:

- The unconstrained Phase A model achieves a lower price MSE but exhibits numerous arbitrage violations, especially calendar violations at short maturities and convexity violations in the wings.
- After turning on the physics penalties, Phase B slightly increases the price MSE but drives the calendar-arbitrage violation rate on the evaluation grid down to zero and significantly reduces butterfly violations.
- The resulting pricing accuracy (RMSE around \$1.76, MAE around \$1.23) is acceptable for practical use, especially considering that many outliers are in illiquid, far-from-the-money contracts.

Qualitatively, the fitted IV smiles at fixed maturities are smooth and convex, passing through the center of the noisy market scatter cloud, with no visible oscillations. The 3D surface $\sigma_\theta(K, T)$ appears smooth across both strike and maturity, without spikes or ridges.

5.2 Zero-shot generalization to January 4–5, 2023

To test whether the model learned structural features of the volatility surface rather than memorizing the Jan 3 cross-section, I evaluate the trained Phase B model directly on SPY option data from 4 and 5 January 2023, without any retraining. Using the same preprocessing pipeline, I compute the pricing RMSE on these new days: **Jan 4, 2023**: RMSE ≈ 1.50, **Jan 5, 2023**: RMSE ≈ 0.93.

These zero-shot errors are comparable to or better than the in-sample RMSE, reflecting that the market conditions on these neighboring days are similar and that the model has captured the underlying term-structure and moneyness dependence robustly.

5.3 Transfer learning via short fine-tuning

In a production setting, one would recalibrate the model daily. To test whether the learned IV surface can be efficiently updated, I use the Jan 3 Phase B parameters as initialization and fine-tune on Jan 4 and Jan 5 data for only 10 additional epochs.

The resulting RMSEs are: **Jan 4, 2023**: RMSE ≈ 1.47 (about 2% improvement over zero-shot), **Jan 5, 2023**: RMSE ≈ 0.87 (about 7% improvement).

6 Discussion and Conclusions

This project demonstrates that Physics-Informed Neural Networks provide a practical and effective framework for IV-surface inversion when combined with a suitable training strategy.

6.1 Key findings

First, the experiments confirm the feasibility of PINN-based IV inversion: a relatively shallow MLP parameterizing $\sigma(K, T)$, coupled with the Black–Scholes formula as a differentiable layer, is sufficient to recover a smooth implied-volatility surface with good pricing accuracy on realistic SPY option data. Second, the role of no-arbitrage penalties is crucial. When trained without any physics terms, the network can fit noisy quotes but produces economically invalid surfaces that exhibit calendar and butterfly arbitrage. Introducing calendar and convexity penalties eliminates these violations on a dense grid while preserving most of the pricing accuracy. Third, the training dynamics matter: if constraints are active from epoch 0, the optimizer tends to converge to a trivial, nearly flat solution, whereas a two-phase schedule that first fits the data and then gradually enforces constraints yields significantly better solutions. Finally, the calibrated model shows robust temporal generalization to neighboring trading days, both in zero-shot mode and after short fine-tuning, indicating that it has captured structural features of the SPY volatility surface rather than merely memorizing a single-day cross-section.

6.2 Limitations and future work

The present PINN focuses on static no-arbitrage constraints, namely calendar monotonicity and butterfly convexity. A more ambitious design would incorporate the Dupire or Black–Scholes PDE directly as a residual term—possibly through an additional price network—to enforce a richer set of dynamic constraints, at the cost of higher model and training complexity. The chosen MLP architecture is deliberately simple and may not fully capture extreme-wing behavior or very long maturities; augmenting it with explicit asymptotic priors or hybrid parametric–neural components could improve extrapolation in these regions. Finally, the experiments are restricted to a single underlying and daily snapshots of the surface. Extending the approach to multiple assets or to intraday time slices would provide a stronger test of scalability and robustness, and would help assess how well the framework handles more complex, higher-dimensional market dynamics.

Despite these limitations, the results suggest that PINNs are a promising tool for robust, arbitrage-aware volatility-surface calibration. The combination of neural-network flexibility, automatic differentiation, and financially motivated constraints yields surfaces that are both accurate and economically interpretable, and supports efficient re-calibration as market conditions evolve.

References

Y. Aït-Sahalia and A. W. Lo. Nonparametric Estimation of State-Price Densities Implicit in Financial Asset Prices. *Journal of Finance*, 53(2):499–547, 1998.

R. Cont and P. Tankov. *Financial Modelling with Jump Processes.* Chapman and Hall/CRC, 2004.

B. Dupire. Pricing with a smile. *Risk*, 7(1):18–20, 1994.

J. Gatheral and A. Jacquier. Arbitrage-free SVI volatility surfaces. *Quantitative Finance*, 14(1):59–71, 2014.

A. Hernandez. Model calibration with neural networks. SSRN working paper, 2016.

M. P. Laurini. Imposing no-arbitrage conditions in implied volatility surfaces using constrained smoothing splines. Insper Working Paper wpe_89, 2007.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

A. Dhiman and Y. Hu. Physics Informed Neural Network for Option Pricing. arXiv preprint arXiv:2312.06711, 2023.

Wharton Research Data Services. *WRDS Research Platform.* University of Pennsylvania. Accessed for OptionMetrics IvyDB US, 2025.