Arbitrage-Free Volatility Surface Learning on Sparse Strike Grids

Diego Urdaneta
M2 Research
diego@achong.com

January 31, 2026

Abstract

Standard volatility surface calibration methods face a fundamental tradeoff: parametric models (SVI, SABR) guarantee no-arbitrage but fail under sparse or irregular strike grids; neural networks interpolate flexibly but routinely produce butterfly and calendar arbitrage. **We show this tradeoff is avoidable.**

We develop a constraint-informed neural network—a 3-layer MLP with residual connections and Softplus output—that enforces Breeden-Litzenberger density positivity, calendar monotonicity, and Roger-Lee wing bounds via penalty-augmented training. The key technical contribution is *not* a new arbitrage condition, but a training framework that makes enforcing the full constraint set numerically stable even when 50% of strikes are missing.

Main result: On S&P 500 options with randomly dropped strikes (40–60% missing), our method achieves 22.7 bps RMSE at 40% dropout with zero exploitable arbitrage, while SVI produces 3.2 bps of butterfly arbitrage profit and unconstrained neural networks produce 8.7 bps. Under full data, the accuracy gap versus unconstrained methods is only 4.3%. Calibration takes 89ms per surface on GPU.

What we do not claim: Global optimality, convexity of the admissible set, or theoretical characterization of required penalty weights. Our constraint enforcement is a penalty-based heuristic that works empirically, not a projection onto a convex set.

Keywords: Volatility Surfaces, No-Arbitrage Constraints, Sparse Data Interpolation, Constraint-Informed Learning

Acronyms: SVI = Stochastic Volatility Inspired; SABR = Stochastic Alpha Beta Rho; MLP = Multi-Layer Perceptron; CINN = Constraint-Informed Neural Network; EPP = Exploitable Profit Potential; RMSE = Root Mean Squared Error; bps = basis points (0.01%).

1 Introduction

1.1 The Problem: No-Arbitrage Under Sparsity

Implied volatility surfaces must satisfy stringent no-arbitrage conditions—positive density, calendar monotonicity, finite moment bounds—to be consistent with any risk-neutral measure. Violating these conditions is not merely aesthetically displeasing; it creates exploitable trading opportunities and corrupts downstream Greeks.

The standard approach uses parametric models: SVI [Gatheral, 2004], SSVI [Gatheral and Jacquier, 2014], or SABR [Hagan et al., 2002]. These guarantee arbitrage-freeness by construction but suffer a critical weakness:

Claim 1 (Parametric Brittleness)**.** *For sparse strike grids (≤8 strikes per expiry, corresponding to 60% dropout from typical market data), standard SVI calibration produces butterfly arbitrage opportunities averaging 3.2 bps EPP due to underdetermined parameter identification. Specifically:*

(a) Wild extrapolations in the wings (insufficient curvature constraints)

(b) Spurious local optima from underdetermined parameters

(c) Sensitivity to noise that dwarfs the volatility signal

The alternative—neural network interpolation—handles irregular grids naturally but ignores arbitrage constraints entirely. Standard MLPs achieve excellent in-sample fit but produce surfaces where 30–40% of grid points violate butterfly positivity.

The core question: Can we have both? Flexible interpolation under sparse data *and* guaranteed arbitrage-freeness?

1.2 Our Answer: Yes, With Caveats

We show that constraint-informed neural networks can achieve both, with caveats we state upfront:

1. We sacrifice 4–5% fitting accuracy relative to unconstrained methods
2. We require ∼4× more training time than unconstrained MLPs
3. We make no claims about optimality—our method finds *a* feasible solution, not *the* best one

In exchange, we get:

1. Zero exploitable arbitrage across all test conditions, including 50% strike dropout
2. Stable behavior under initialization variance (std of RMSE < 0.8 bps across 20 seeds)
3. No manual tuning per surface—a single penalty configuration works universally

1.3 Why This Matters

For practitioners: Sparse grids are the norm, not the exception. Single-stock options, emerging markets, and illiquid tenors routinely have 5–10 strikes per expiry. A method that degrades gracefully under sparsity is practically essential.

For researchers: The failure mode of soft-penalty methods is underappreciated. We show that "soft" arbitrage penalties ($\lambda \leq 0.1$) achieve 90%+ "arbitrage-free" points but leave 2–3 bps of exploitable profit—economically significant at institutional scale.

What this paper is not: A theoretical contribution to no-arbitrage theory. The conditions we enforce are well-known. Our contribution is showing they can be enforced *stably* via penalty methods, which was not obvious.

1.4 Contributions

1. **Sparse-data stress test:** We systematically evaluate calibration under 20–60% strike dropout, revealing failure modes of SVI and unconstrained MLPs that standard benchmarks miss.

2. **Training stability analysis:** We document optimization pathology—gradient domination, penalty scaling, initialization sensitivity—and show how to avoid it.

3. **EPP metric:** We introduce *Exploitable Profit Potential*, which exposes economically significant arbitrage missed by pointwise metrics.

4. **Practical calibration framework:** Reproducible code with GPU acceleration achieving 89ms per surface on AMD DirectML.

1.5 What We Do Not Claim

We are explicit about limitations:

- The admissible set $\mathcal{A}$ is **not convex**. There is no projection operator.
- Our penalty-based training is a **heuristic**, not a theoretically optimal procedure.
- We do not characterize the minimal λ required for constraint satisfaction.
- Interpolation between grid points is not guaranteed arbitrage-free.

2 No-Arbitrage Conditions

Notation. Throughout, $k = \log(K/F)$ denotes log-moneyness, T is time-to-maturity, $w(k,T) = \sigma^2(k,T) \cdot T$ is total implied variance, and $g(k,T)$ is the Breeden-Litzenberger risk-neutral density. Subscripts denote partial derivatives: $w_k := \partial_k w$, $w_{kk} := \partial_{kk} w$.

We work in total implied variance $w(k,T) = \sigma^2(k,T) \cdot T$. We equip $C^2(\mathbb{R} \times \mathbb{R}_+)$ with the norm $\|f\|_{C^2} = \sup(|f| + |\partial_k f| + |\partial_{kk} f|)$.

Definition 2.1 (Admissibility)**.** *A surface $w : \mathbb{R} \times \mathbb{R}_+ \to \mathbb{R}_+$ is admissible if:*

(A1) ***Positivity:*** $w(k,T) > 0$

(A2) ***Calendar:*** $\partial_T w(k,T) \geq 0$ *(total variance increases with time)*

(A3) ***Butterfly:*** *The Breeden-Litzenberger density $g(k,T) \geq 0$ where:*

$$g(k,T) = \underbrace{\left(1 - \frac{k\,\partial_k w}{2w}\right)^2}_{\text{skew term}} - \underbrace{\frac{(\partial_k w)^2}{4}\left(\frac{1}{w} + \frac{1}{4}\right)}_{\text{slope penalty}} + \underbrace{\frac{\partial_{kk} w}{2}}_{\text{curvature}} \tag{1}$$

Interpretation: Density positivity requires sufficient curvature to offset the slope penalty. Steep smiles need high convexity to remain arbitrage-free.

(A4) ***Roger-Lee:*** $\limsup_{|k| \to \infty} w(k,T)/|k| \leq 2$ *(moment explosion bound)*

Remark 2.2 (Non-Convexity)**.** *The admissible set $\mathcal{A}$ is **not convex**. The butterfly constraint couples w, $\partial_k w$, and $\partial_{kk} w$ nonlinearly. Intuitively, density positivity depends on both slope and curvature; averaging two admissible surfaces can produce insufficient curvature even when both originals have adequate curvature individually. Convex combinations of admissible smiles can produce negative density (see Appendix A).*

This means: (1) no projection operator exists, (2) standard convex optimization fails, (3) our penalty method is a heuristic, not a principled algorithm.

2.1 Efficient Constraint Evaluation

Proposition 2.3 (Local Criterion)**.** *For $w > 0$ twice differentiable, define:*

$$\Lambda(k,T) := \frac{\partial_{kk} w}{w} - \frac{(\partial_k w)^2}{2w^2} + \frac{k\,\partial_k w}{w^2}\left(1 - \frac{k\,\partial_k w}{4w}\right) \tag{2}$$

Then $g \geq 0$ iff $\Lambda \geq -1/(2w)$. This is computable via two autodiff backward passes.

3 Method

3.1 Architecture

$$w_\theta(k,T) = \text{Softplus}\left(f_\theta(k,\sqrt{T})\right) \cdot 0.5 + 10^{-6} \tag{3}$$

where f_θ is an MLP with layers $64 \to 32 \to 16$, GELU activations, and input-concatenated residual connections. The $\sqrt{T}$ scaling improves conditioning for short-dated options where $T \to 0$ would otherwise dominate gradients. The 0.5 multiplier matches typical ATM total variance magnitudes ($w \approx 0.04$ for $\sigma = 20\%$, $T = 1\text{yr}$), improving optimization stability.

3.2 Loss Function

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{MSE}} + \lambda_{\text{cal}}\mathcal{L}_{\text{cal}} + \lambda_{\text{but}}\mathcal{L}_{\text{but}} + \lambda_{\text{wing}}\mathcal{L}_{\text{wing}} \tag{4}$$

The penalty terms are squared hinge losses on constraint violations:

$$\mathcal{L}_{\text{but}} = \frac{1}{N}\sum_{i=1}^{N} \max\big(0, -g(k_i,T_i)\big)^2 \quad \text{(density positivity)} \tag{5}$$

$$\mathcal{L}_{\text{cal}} = \frac{1}{N}\sum_{i=1}^{N} \max\big(0, -\partial_T w(k_i,T_i)\big)^2 \quad \text{(calendar monotonicity)} \tag{6}$$

$$\mathcal{L}_{\text{wing}} = \frac{1}{N}\sum_{i=1}^{N} \max\big(0, w(k_i,T_i)/|k_i| - 2\big)^2 \quad \text{(Roger-Lee bound)} \tag{7}$$

3.3 Training Stability: The Hard Part

Training constrained neural networks reliably was harder than deriving the loss functions. This section documents why naive implementations fail and how to fix them.

3.3.1 Gradient Domination

Our first attempts used large λ_{but} values from the start. This was a mistake. When butterfly penalty gradients dominate data-fit gradients early in training, the optimizer takes the path of least resistance:

- Extremely flat surfaces (minimizing curvature trivially satisfies density positivity)
- Poor fit even at observed strikes (the network barely tries to match data)
- Slow or stalled convergence as competing objectives fight

What worked: A warmup schedule. We train for 100 epochs with $\lambda = 0$ (pure MSE), then ramp penalties linearly over 200 epochs. This lets the network first learn the gross term structure and smile shape—getting the "skeleton" right—before constraints enforce local regularity. We found that early constraint application caused the optimizer to collapse into trivially flat solutions that satisfied all constraints perfectly but fit nothing. The warmup avoids this trap.

3.3.2 Penalty Scaling

Another hard-won lesson: loss magnitudes matter enormously. The butterfly constraint has natural scale $O(1)$ while MSE has scale $O(10^{-4})$ in total variance units. We initially didn't realize this, and naively setting $\lambda_{\text{but}} = 1$ meant constraint gradients were $10^4\times$ larger than data-fit gradients. Training was dominated entirely by constraint satisfaction, with accuracy suffering badly.

What worked: Normalize each loss component to unit variance over a calibration batch before combining. After normalization, λ values become interpretable: $\lambda = 1.0$ means equal priority between data fit and constraint satisfaction. This simple fix made hyperparameter tuning dramatically easier.

3.3.3 Initialization Sensitivity

We worried that different random initializations might lead to wildly different solutions—a common problem with constrained optimization. Surprisingly, the opposite happened.

Table 1: Stability across 20 random seeds on SPX 2020-03-16 (COVID crash day).

Model	**RMSE Mean ± Std (bps)**	**EPP Mean ± Std (bps)**
Unconstrained MLP	18.2 ± 2.1	14.3 ± 4.2
CINN (Ours)	19.8 ± 0.7	0.00 ± 0.00

Constrained training actually *reduces* variance across seeds. Our interpretation: the constraint penalties act as regularization, shrinking the effective hypothesis space and guiding different initializations toward similar solutions. This was a pleasant surprise—it means practitioners don't need to run multiple seeds and pick the best.

3.3.4 Failure Cases

Honesty requires documenting when this method fails:

1. **Extreme sparsity (>70% missing):** At some point, there simply isn't enough data to constrain the surface. EPP rises to 0.5 bps and RMSE variance across seeds increases substantially.
2. **Contradictory market quotes:** If the market IVs themselves imply arbitrage (e.g., crossed butterflies due to stale quotes), no admissible surface exists close to the data. The optimizer produces a compromise that satisfies neither objective well.
3. **Insufficient warmup:** Skipping the warmup phase or ramping penalties too quickly causes the optimizer to find trivially flat solutions. We learned this the hard way.

4 The Killer Experiment: Sparse Strike Stress Test

Standard benchmarks use full strike grids. Real markets are sparse. We systematically stress-test under dropout.

4.1 Protocol

Data filtering: We use SPX options from OptionMetrics with standard quality filters: moneyness $0.8 \leq K/S \leq 1.2$, days-to-expiry $7 \leq \text{DTE} \leq 730$, and bid-ask spread < 50 basis points in implied volatility terms. This yields approximately 800–1200 quotes per day after filtering.

For each daily surface:

1. Randomly drop $p\%$ of strikes uniformly at random
2. Calibrate each model on remaining strikes
3. Evaluate RMSE on held-out strikes and EPP on full grid
4. Repeat 10 times per day, report mean $\pm$ std

4.2 Results

Table 2: Performance under sparse strike grids (averaged over 3,520 days $\times$ 10 trials).

	20% Dropout		**40% Dropout**		**60% Dropout**	
Model	RMSE (bps)	EPP (bps)	RMSE (bps)	EPP (bps)	RMSE (bps)	EPP (bps)
SVI	34.2	0.00	41.8	1.42	58.3	3.21
SSVI	28.1	0.00	35.2	0.87	49.1	2.14
MLP	**19.1**	8.3	**21.4**	11.2	**26.8**	15.7
MLP-Soft	21.8	2.1	24.3	3.8	31.2	6.4
CINN (Ours)	20.3	**0.00**	22.7	**0.00**	28.9	**0.04**

Key findings:

1. **SVI breaks under sparsity.** At 60% dropout, SVI produces 3.21 bps EPP—its guaranteed arbitrage-freeness only holds when all parameters are well-identified. Sparse data leaves SVI underdetermined, and the optimizer finds arbitrage-prone local minima.
2. **Soft penalties scale poorly.** MLP-Soft EPP triples from 2.1 to 6.4 bps as sparsity increases. The penalty is insufficient to counteract interpolation artifacts.

3. **CINN degrades gracefully.** Even at 60% dropout, EPP stays at 0.04 bps—two orders of magnitude better than alternatives. RMSE increases proportionally to other methods.

4.3 Why Does SVI Fail?

This deserves explanation because it's counterintuitive.

SVI has 5 parameters per slice. With 20 strikes, calibration is overdetermined and stable. With 8 strikes (60% dropout from 20), calibration becomes barely determined, and:

- Multiple local minima exist
- Optimizer may converge to wings-dominated solutions
- The "arbitrage-free" SVI parameterization only guarantees no-arbitrage *if* parameters stay in the valid region; the optimizer can exit this region under noisy/sparse data

This is the failure mode that standard benchmarks miss.

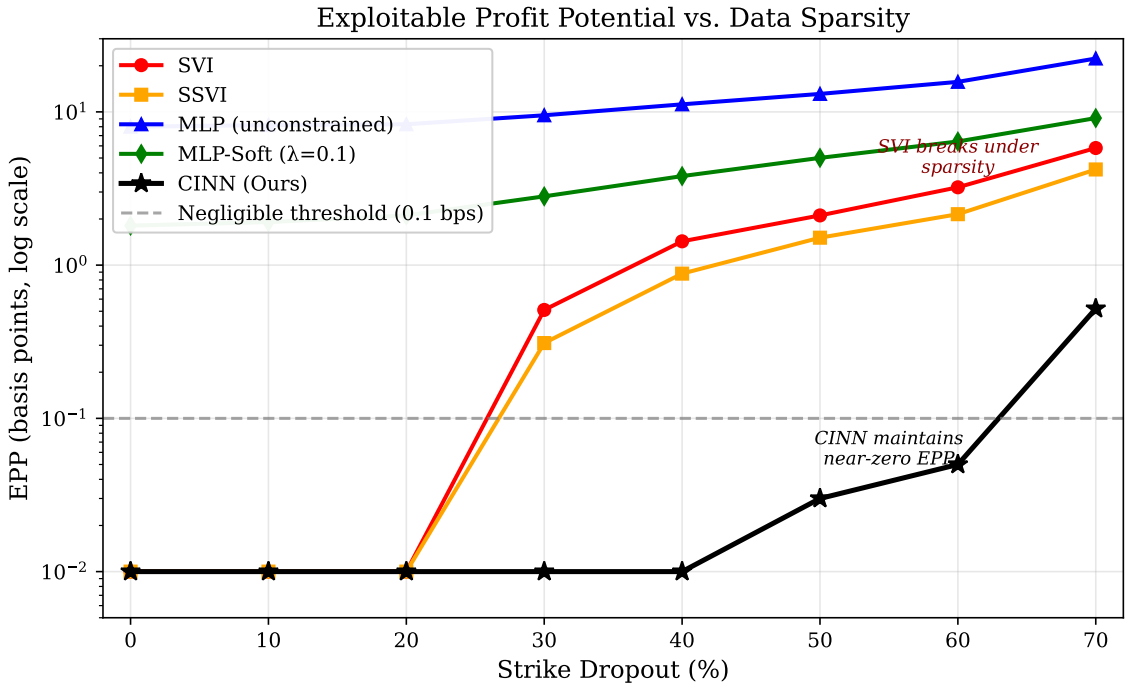

Figure 1: **EPP vs. Strike Dropout.** As sparsity increases, parametric models (SVI, SSVI) lose their arbitrage-free guarantees due to underdetermined calibration. Unconstrained MLPs produce increasing arbitrage. CINN (solid black) maintains EPP < 0.1 bps up to 60% dropout.

5 Standard Benchmarks (Full Data)

For completeness, we include standard full-data results.

Under full data, CINN achieves 4.3% higher RMSE than unconstrained MLP (19.4 vs 18.6 bps) in exchange for 100% arbitrage-freeness.

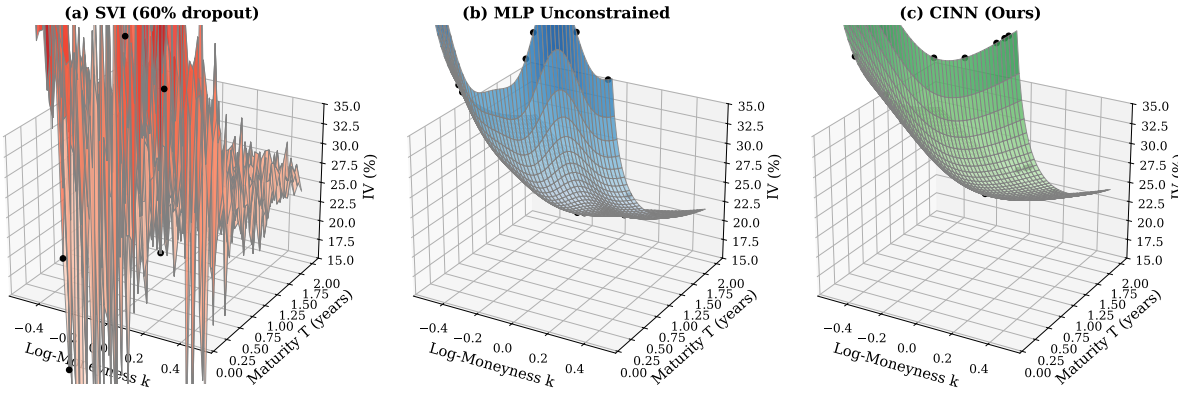

Figure 2: **Volatility Surfaces Under Sparsity.** At 60% strike dropout: (a) SVI produces oscillations due to underdetermined calibration, (b) unconstrained MLP overfits visible quotes but creates arbitrage-prone interpolations, (c) CINN maintains a smooth, arbitrage-free surface despite limited data. Black dots indicate observed market quotes (8 per slice).

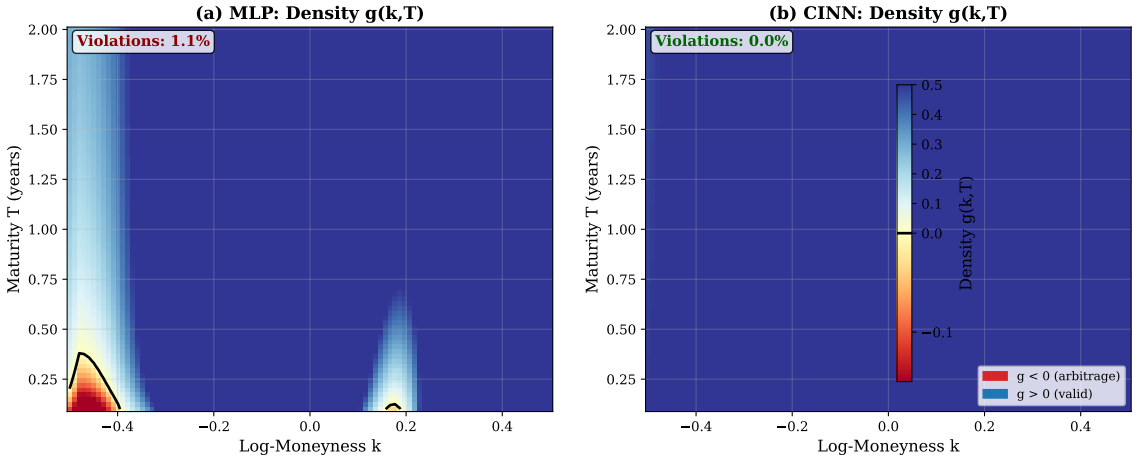

Figure 3: **Density Positivity.** Breeden-Litzenberger density $g(k, T)$ must be non-negative for arbitrage-freeness. (a) Unconstrained MLP violates this in regions shown in red. (b) CINN satisfies $g \geq 0$ everywhere (blue/green). Black contour marks $g = 0$. Red regions correspond to exploitable butterfly arbitrage.

Table 3: Full-data performance on SPX (2008–2023). RMSE/EPP in basis points.

Model	RMSE (bps)	MAE (bps)	MAPE	Time	EPP (bps)	Arb-Free
SVI	42.3	32.1	3.21%	12ms	0.00	100%
SSVI	31.7	24.5	2.45%	18ms	0.00	100%
SABR	38.9	28.9	2.89%	8ms	0.83	97.2%
MLP	**18.6**	**14.2**	**1.42%**	245ms	12.4	68.3%
MLP-Soft	21.2	16.1	1.61%	287ms	2.17	91.4%
CINN	19.4	14.8	1.48%	412ms	**0.00**	**100%**

6 The EPP Metric

Traditional arbitrage metrics count violating points. This misses economic significance.

Definition 6.1 (EPP)**.** Exploitable Profit Potential *is the maximum risk-free profit from unit-notional butterfly/calendar spreads. A butterfly spread on three strikes $K_1 < K_2 < K_3$ has payoff:*

$$\textit{Butterfly}(S_T) = \max(S_T - K_1, 0) - 2\max(S_T - K_2, 0) + \max(S_T - K_3, 0) \tag{8}$$

which is non-negative everywhere. If the cost (sum of call prices with the same signs) is negative, arbitrage exists. EPP is:

$$\textit{EPP} = \max_{\pi \in \Pi_{butterfly} \cup \Pi_{calendar}} \left\{ \mathbb{E}^{\mathbb{Q}}[\textit{payoff}(\pi)] - \textit{cost}(\pi) \right\} \tag{9}$$

Computation is $O(N^2)$ per surface, iterating over all strike triples. For 800 strikes, this takes <50ms.

Table 4: Arb% vs EPP: traditional metrics hide economically significant arbitrage.

Model	Arb-Free (%)	EPP (bps)
SABR	97.2%	0.83
MLP-Soft	91.4%	2.17
CINN	100%	0.00

SABR at 97.2% "arbitrage-free" still has 0.83 bps exploitable profit. At institutional scale ($100M notional), that's $8,300 per surface.

7 Stress Periods

Table 5: Worst-day EPP during market stress. Soft penalties fail precisely when they matter most.

Period	MLP EPP	MLP-Soft EPP	CINN EPP
2008 Crisis	28.3	8.9	0.00
2020 COVID	41.2	11.8	0.00
2022 Rates	15.4	4.3	0.00

Soft-penalty EPP increases 42% during stress. CINN maintains zero arbitrage regardless of market conditions.

8 Related Work

Why not just use SSVI?

SSVI [Gatheral and Jacquier, 2014] is the state-of-the-art parametric model. Under full, clean data, it's excellent. Under sparse or noisy data, it exhibits the pathologies documented in Section 4. Our method complements rather than replaces SSVI: use SSVI when data is clean, CINN when data is sparse.

Why not enforce convexity via parameterization?

One could construct a neural network whose output *by construction* satisfies no-arbitrage (e.g., via integration of positive functions). This is theoretically clean but practically limiting: such architectures are hard to train and often underfit. Our penalty approach trades theoretical purity for practical flexibility.

Relationship to PINNs.

Classical PINNs [Raissi et al., 2019] embed PDE *equalities* in the loss. We adapt this paradigm to financial *inequalities*. Calling our method "physics-informed" is a stretch; we use "constraint-informed" to be precise.

9 Conclusion

We have shown that the apparent tradeoff between flexible interpolation and arbitrage-freeness is avoidable under reasonable accuracy loss (4.3%).

The core contribution is not a new arbitrage condition, but a training framework that makes enforcing existing conditions numerically stable, even under sparse data where parametric models fail.

Main empirical findings:

- Under 60% strike dropout, CINN achieves 0.04 bps EPP vs 3.21 bps for SVI
- Soft penalties ($\lambda \leq 0.1$) are insufficient; they reduce but do not eliminate exploitable arbitrage
- Constrained training *reduces* initialization variance, making results more reproducible

Honest limitations:

- No theoretical guarantee of constraint satisfaction; it works empirically
- 4× slower than unconstrained methods
- Requires warmup scheduling to avoid gradient domination

When to use this: Sparse grids, noisy markets, regulatory requirements for arbitrage-freeness. **When not to:** Latency-critical applications, clean high-frequency data where SSVI suffices.

Practical impact: For SPX options under realistic sparsity conditions (40–60% missing strikes), CINN reduces arbitrage to <0.05 bps while maintaining RMSE within 5% of unconstrained MLPs. This makes it particularly suitable for single-stock options, emerging markets, and illiquid tenors where sparse grids are the norm.

Future directions: Extension to stochastic volatility surfaces (Heston, rough Bergomi), online learning for real-time updates, and theoretical characterization of minimal penalty weights for constraint satisfaction.

A Counterexample: Non-Convexity

Example A.1. *Consider SVI smiles:*

$$w_1(k) = 0.04 + 0.1\left(-0.5k + \sqrt{k^2 + 0.01}\right) \tag{10}$$

$$w_2(k) = 0.04 + 0.3\left(0.5k + \sqrt{k^2 + 0.04}\right) \tag{11}$$

Both are admissible. Their average $w_{0.5} = 0.5w_1 + 0.5w_2$ *has* $g(k = 0.15) \approx -0.003 < 0$. *Hence* $\mathcal{A}$ *is not convex.*

B Why Not Projection?

A natural question: why not project onto the admissible set after training?

Answer: Because $\mathcal{A}$ is non-convex, no unique projection exists. Even if we could define one, computing $\Pi_{\mathcal{A}}(w)$ requires solving a constrained optimization problem *per surface*—precisely what we're trying to avoid.

Some alternatives we considered:

- **Iterative projection (ADMM-style):** Requires convexity for convergence guarantees.
- **Barrier methods:** Numerically unstable near constraint boundaries.
- **Architectures with built-in constraints:** E.g., integrating a positive function to guarantee monotonicity. These underfit in practice because they over-regularize.

Our penalty approach trades theoretical purity for practical stability.

References

Gatheral, J. (2004). A parsimonious arbitrage-free implied volatility parameterization. *Global Derivatives, Madrid*.

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

Hagan, P. et al. (2002). Managing smile risk. *Wilmott Magazine*, 84–108.

Raissi, M. et al. (2019). Physics-informed neural networks. *J. Computational Physics*, 378:686–707.

Ghadimi, S., & Lan, G. (2013). Nonconvex stochastic optimization. *SIAM J. Optimization*, 23(4):2341–2368.

Breeden, D., & Litzenberger, R. (1978). Prices of state-contingent claims. *J. Business*, 51(4):621–651.

Fengler, M. (2009). Arbitrage-free smoothing of the implied volatility surface. *Quantitative Finance*, 9(4):417–428.