# Provable Multi-Region Affinity Enforcement and Constraint Satisfaction for Scientific Machine Learning

**Anonymous authors**
**Paper under double-blind review**

## Abstract

Neural networks have shown strong promise in scientific machine learning, but strictly enforcing boundary, interface, and safety constraints remains difficult. Soft penalties require careful tuning and do not guarantee exact satisfaction, while existing hard-constraint methods are typically specialized to particular equations or geometries. We introduce mPOLICE, a general framework that guarantees exact constraint satisfaction across multiple disjoint regions by exploiting the piecewise-linear structure of standard neural networks. By strategically configuring the network's internal activations, mPOLICE ensures that the learned function becomes exactly affine (linear) throughout each user-specified constrained zone. Once the network is locally affine, complex physical and safety constraints reduce to simple linear equations evaluated only at the corners (vertices) of each zone. Crucially, our approach handles many disjoint regions independently–a major limitation of existing single-region approaches. The method adds no inference-time overhead, requires no architectural changes, and integrates readily with standard training pipelines. We validate mPOLICE on operator learning, PDE boundary-condition enforcement, safety-critical control, and implicit 3D shape modeling.

## 1 Introduction

Scientific machine learning increasingly relies on neural networks to approximate PDE solutions, inverse maps, and operators. Representative examples include physics-informed neural networks (PINNs) (Raissi et al., 2019; Karniadakis et al., 2021), variational methods such as the Deep Ritz method and variational PINNs (E & Yu, 2018; Kharazmi et al., 2019), and neural operators such as DeepONet and the Fourier Neural Operator (Lu et al., 2021a; Li et al., 2021; Kovachki et al., 2023). In these settings, approximation accuracy alone is insufficient: predictions must also satisfy boundary conditions, interface conditions, conservation laws, and geometric constraints to remain physically meaningful. This is especially important in applications such as multi-material diffusion, fluid dynamics, climate modeling, and inverse design, where even small violations can lead to qualitatively incorrect behavior (Lu et al., 2021b; Xie et al., 2024; Beucler et al., 2021).

Yet exact enforcement of such structure remains difficult. Most practical scientific-ML pipelines still impose boundary or interface conditions through soft penalties. These methods can work well with sufficient tuning, but the required weights are problem-dependent, can create challenging optimization landscapes, and do not provide guarantees of exact satisfaction (Krishnapriyan et al., 2021). Existing hard-constraint methods provide important progress (Lu et al., 2021b; Xie et al., 2024; Beucler et al., 2021), but they are often specialized to particular PDEs, coordinate systems, or boundary-condition forms. Neural operators face a related difficulty: constraints should transfer across unseen input functions, not merely be approximately satisfied on the training set (Lu et al., 2021a; Li et al., 2021; Kovachki et al., 2023). More broadly, standard deep learning methods often rely on sampling, penalties, or post hoc corrections that may not provide provable guarantees (Kotary et al., 2021; Kotary & Fioretto, 2024; Li & Shi, 2018; Tordesillas et al., 2023).

Our approach exploits a geometric property of ReLU networks rather than relying on penalties. A piecewise-linear network partitions its input space into polyhedral activation cells, and the network restricts to a single affine map on each cell. Hard constraint satisfaction then proceeds in two stages. In the first stage, *affinity*

*enforcement*, we ensure that every constrained patch lies entirely inside a single activation cell, locking the network output to a single affine map on that patch. In the second stage, *constraint enforcement*, we impose the desired physical conditions on that affine map. Because an affine function on a convex polytope is fully determined by its vertex values, both output and first-derivative constraints reduce to small linear systems over the vertices. For instance, Dirichlet conditions become direct vertex-value equations, while Neumann and Robin conditions become linear relations among those same vertex outputs. Importantly, this requires no architectural modifications: the network is a standard feedforward ReLU model, so existing architectures, initializations, and training infrastructure can be reused without change.

We distinguish two terms used throughout the paper. *Enforcement* refers to the algorithmic step—a sign-pattern assignment, a weight projection, or a vertex solve—used to impose a desired property during training. *Constraint satisfaction* refers to the outcome: the learned function provably obeys the target equalities or inequalities on the designated region.

POLICE (Balestriero & LeCun, 2023) solves the affinity-enforcement stage for a single convex region and has already been applied to settings requiring hard guarantees in robotics and control (Bouvier et al., 2024b;a). Realistic PDE settings, however, involve multiple disjoint boundary segments, interfaces, inclusions, and obstacles. Applying POLICE with a single activation pattern across all of these regions forces the network to be affine over their combined convex hull, producing unintended coupling between distant patches (see Figure 1). In addition, prior POLICE work focused on establishing local affinity without providing a general framework for the second stage of enforcing mixed equality, inequality, and derivative constraints on the resulting affine maps.

This paper addresses both stages. For affinity enforcement, we introduce mPOLICE, a multi-region extension of POLICE that assigns a unique activation pattern to each constrained region, guaranteeing localized affine behavior on every patch independently. For constraint enforcement, we show that value, inequality, interface, and first-derivative conditions all reduce to small linear projections on vertex outputs, and we provide practical projected-training procedures to impose them. Although we motivate the framework through physical boundary and interface conditions, the same patchwise-affine mechanism applies to reinforcement-learning safety constraints, implicit-shape occupancy constraints, and operator-learning constraints. The experiments therefore demonstrate the broad applicability of our methodology beyond the motivating domain.

Our contributions are threefold. First, mPOLICE assigns unique activation patterns to multiple disjoint convex regions, provably guaranteeing independent local affine behavior and eliminating the convex-hull coupling of single-pattern methods. Second, we exploit the fact that once affinity holds, Dirichlet, Neumann, Robin, RL safety, and occupancy constraints all reduce to linear systems on vertex outputs, and we develop practical sign-assignment, sign-enforcement, and projected-training algorithms to enforce them during training with zero inference overhead. Third, we empirically validate the framework on operator learning, boundary-condition enforcement, safety-critical RL, and implicit shape approximation.

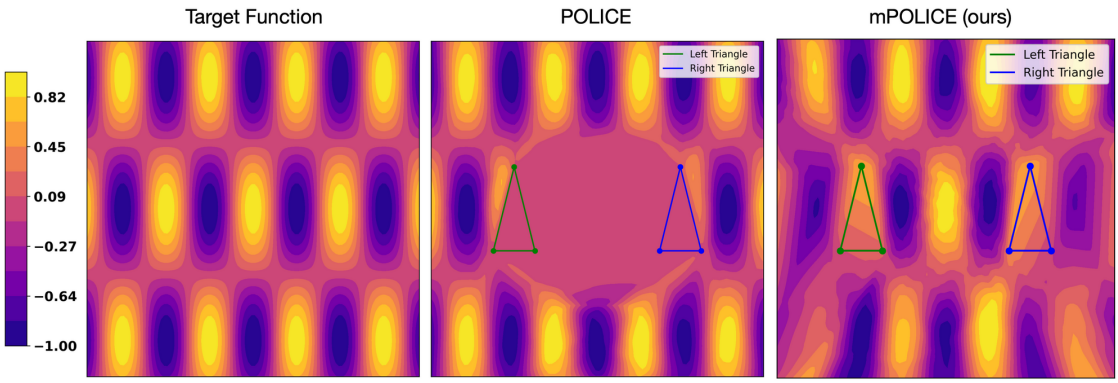


Figure 1: POLICE vs. mPOLICE. A single activation pattern makes the network affine over the hull of all regions (middle); mPOLICE assigns unique patterns, localizing affine behavior to each region (right).

### 1.1 Related Work

**Scientific machine learning.** PINNs (Raissi et al., 2019; Karniadakis et al., 2021), variational methods such as the Deep Ritz method and variational PINNs (E & Yu, 2018; Kharazmi et al., 2019), and neural operators such as DeepONet and FNO (Lu et al., 2021a; Li et al., 2021; Kovachki et al., 2023) have greatly expanded the reach of neural PDE solvers. In all of these frameworks, boundary and interface conditions are most commonly imposed through soft penalties, which introduce tuning sensitivity and optimization difficulties (Krishnapriyan et al., 2021). Hard-constraint alternatives exist (Lu et al., 2021b; Xie et al., 2024; Beucler et al., 2021; Djeumou et al., 2022), but they tend to be specialized to particular PDEs, coordinate systems, or boundary-condition types. Neural operators face the additional requirement that spatial constraints must transfer to unseen input functions, not merely be satisfied on the training set (Lu et al., 2021a; Li et al., 2021; Kovachki et al., 2023).

**Constrained learning beyond PDEs.** Architectural enforcement of properties such as monotonicity, convexity, and linear output constraints has been explored in several settings (Li & Shi, 2018; Tordesillas et al., 2023; Konstantinov et al., 2024; Zhong et al., 2023), and constraint-aware learning also arises in robotics, planning, and combinatorial optimization (Kondo et al., 2024; Bouvier et al., 2024b;a; Kotary et al., 2021; Kotary & Fioretto, 2024; Giannone et al., 2023; Picard et al., 2024). A separate line of work modifies network parameters so that outputs lie in prescribed polytopes for specified input regions (Tao et al., 2023; Tao & Thakur, 2024). These target output containment, whereas mPOLICE targets a different objective: making the function computed by the network exactly affine on each designated region by controlling neuron activation patterns. HardNet-Aff (Min & Azizan, 2024) enforces hard affine constraints at evaluation points but does not establish region-wise affinity, so satisfaction is guaranteed only at queried points rather than over the entire constrained domain.

**Relation to POLICE.** POLICE (Balestriero & LeCun, 2023) introduced activation-pattern control to enforce local affinity on a single convex region. mPOLICE extends this to multiple disjoint regions with independent activation patterns and, crucially, develops the downstream constraint-enforcement layer—equality, inequality, and derivative projections on vertex outputs—that turns local affinity into provable constraint satisfaction across diverse application domains.

## 2 Methodology

We consider feedforward piecewise-linear deep networks, focusing on ReLU networks. This section formalizes the two stages of mPOLICE. *Affinity enforcement* (Sections 2.3–2.6) ensures that the network is exactly affine on each designated region by assigning and maintaining unique activation patterns. *Constraint enforcement* (Sections 2.8–2.9) then exploits that affine structure to satisfy target physical or task-level conditions via linear systems on vertex outputs.

### 2.1 Piecewise Affine Structure of ReLU Networks

A feedforward ReLU network defines a continuous piecewise-affine function. Formally, each layer $\ell$ computes

$$\boldsymbol{z}^{(\ell)} = \boldsymbol{W}^{(\ell)}\boldsymbol{x}^{(\ell)} + \boldsymbol{b}^{(\ell)}, \quad \boldsymbol{x}^{(\ell+1)} = \sigma(\boldsymbol{z}^{(\ell)}), \tag{1}$$

where $\boldsymbol{W}^{(\ell)}$ and $\boldsymbol{b}^{(\ell)}$ are the weights and biases of layer $\ell$, $\boldsymbol{x}^{(1)} = \boldsymbol{x}$ is the network input, and $\sigma(u) = \max(u, 0)$. Each ReLU neuron introduces half-space constraints splitting the input domain into two regions depending on its sign. Stacking $L$ layers yields a finite set of simultaneously satisfiable inequalities that produce a finite collection of $T$ activation cells $\{\mathcal{P}_r\}_{r=1}^{T}$ (Montufar et al., 2014). On each such cell, the activation pattern is fixed, making $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ an affine function $\boldsymbol{\Lambda}_r\boldsymbol{x} + \boldsymbol{\gamma}_r$. This piecewise-affine property is central: ensuring each region $R_i$ lies entirely within one such cell guarantees that $f_{\boldsymbol{\theta}}$ is affine on $R_i$. This property also applies to networks with linear or piecewise-linear activations (e.g., Leaky ReLU).

### 2.2 Problem Setup and Preliminaries

Consider a deep neural network $f_{\boldsymbol{\theta}} : \mathbb{R}^D \to \mathbb{R}^K$ with parameters $\boldsymbol{\theta}$, representing a learned scalar or vector field. Assume there are $N$ disjoint convex polytopal regions $\{R_i\}_{i=1}^N$ contained in $\mathbb{R}^D$. In scientific-machine-learning applications, these regions may correspond to strips adjacent to boundary segments, neighborhoods of material interfaces, inclusions, electrodes, or other localized patches where exact structure is required. Each region $R_i$ can be described by a finite set of vertices $\{\boldsymbol{v}_p^{(i)}\}_{p=1}^{P_i}$. We begin with output constraints of the form

$$\boldsymbol{E}_i f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{f}_i, \quad \forall\, \boldsymbol{x} \in R_i, \tag{2}$$

$$\boldsymbol{C}_i f_{\boldsymbol{\theta}}(\boldsymbol{x}) \leq \boldsymbol{d}_i, \quad \forall\, \boldsymbol{x} \in R_i. \tag{3}$$

These combined constraints can encode fixed values, admissible ranges, interface matching conditions on state variables, or other forms of prior physical knowledge. This immediately raises two distinct questions. First, how do we guarantee that $f_{\boldsymbol{\theta}}$ is affine on all of $R_i$ rather than merely at sampled points? Second, once such local affinity holds, how do we enforce the desired output or derivative constraints on the resulting affine map? Derivative constraints will be treated later, after the affinity step is in place.

To answer the first question, we impose *affinity* over $R_i$ by placing each region $R_i$ inside a unique activation cell $\mathcal{P}_i$. Then, $f_{\boldsymbol{\theta}}$ becomes an affine function on $R_i$:

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) \;=\; \boldsymbol{\Lambda}_i\, \boldsymbol{x} \;+\; \boldsymbol{\gamma}_i, \quad \boldsymbol{x} \in R_i, \tag{4}$$

Once this geometric condition holds, the target output constraints in Equations 2 and 3 only need to be checked on the *finite* set of vertices $\{\boldsymbol{v}_p^{(i)}\}_{p=1}^{P_i}$. For a physical patch, this replaces an infinite-dimensional condition over a continuum with a finite set of linear relations:

$$\boldsymbol{E}_i\,(\boldsymbol{\Lambda}_i\, \boldsymbol{v}_p^{(i)} + \boldsymbol{\gamma}_i) \;=\; \boldsymbol{f}_i, \quad \text{or} \quad \boldsymbol{C}_i\,(\boldsymbol{\Lambda}_i\, \boldsymbol{v}_p^{(i)} + \boldsymbol{\gamma}_i) \;\leq\; \boldsymbol{d}_i, \quad \forall\, p = 1, \ldots, P_i. \tag{5}$$

However, standard neural network training offers no guarantee that $f_{\boldsymbol{\theta}}$ will be affine on any specific $R_i$, nor that $R_i$ will naturally align with a single activation cell of the piecewise-affine decomposition induced by the network's ReLU or Leaky-ReLU activations. This lack of alignment is the primary obstacle when one wants exact behavior on many physically distinct patches simultaneously.

### 2.3 From Single to Multiple Regions and the Convex Hull Problem

The original POLICE method (Balestriero & LeCun, 2023) was designed to solve the *affinity-enforcement* problem for a single convex region $R$: it ensures the exact affine behavior of a deep ReLU network $f_{\boldsymbol{\theta}}$ within that region. By enforcing consistent pre-activation sign patterns across all vertices of that region, the algorithm guarantees that $R$ is contained within a single activation cell of the network's piecewise-affine decomposition. At a high level, given the vertices $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_P\}$ of $R$, the algorithm identifies a binary sign pattern $\boldsymbol{s} = (s_1, \ldots, s_{N_\ell})$ for the $N_\ell$ neurons in layer $\ell$, and adjusts the parameters so that:

$$0 \leq \min_{p \in [P]} (H_{p,k}^{(\ell)}\, s_k), \quad \forall\, k \in \{1, \ldots, N_\ell\}\,, \tag{6}$$

where $\boldsymbol{H}^{(\ell)} \triangleq \boldsymbol{V}^{(\ell)}(\boldsymbol{W}^{(\ell)})^T + \mathbf{1}_P(\boldsymbol{b}^{(\ell)})^T$ is the pre-activation matrix of layer $\ell$ over the vertices of $R$. Here, $s_k \in \{-1, +1\}$ encodes on which side of the hyperplane defined by the $k$-th neuron (of layer $\ell$) the region $R$ is placed. By ensuring that all vertices share the same sign pattern, $R$ is effectively "trapped" inside a single activation cell of the network. As a result, $f_{\boldsymbol{\theta}}$ behaves as an affine function on $R$. However, this approach implicitly assumes a single region. Extending it to multiple disjoint regions $\{R_i\}_{i=1}^N$ by applying its logic independently to each leads to a problem: multiple regions may be assigned the same sign patterns. If regions share an activation pattern, the network becomes affine not just on the union of the individual regions, but on their entire convex hull—the *convex hull problem*.

This issue stems from POLICE's minimum operation in Equation 6: by seeking a non-negative minimum pre-activation across all vertices, it finds a single activation pattern, so applying it to multiple regions either

merges them into one cell or constrains only the last region processed. Appendix B gives a concrete example in which bias-only adjustments cannot resolve conflicting sign requirements.

Our work overcomes this by assigning a unique global activation sign pattern to each region $R_i$, which provably confines each $R_i$ to its own distinct activation cell (Theorem 1). This localized affine behavior prevents the spurious convex-hull coupling and preserves the geometric separation on which downstream constraint enforcement relies.

### 2.4 Problem Formulation: Multi-Region Sign Assignment and Parameter Adjustments

We now formulate the general problem of assigning unique sign patterns to multiple disjoint convex regions and adjusting the network parameters accordingly. Suppose we have a feedforward ReLU network $f_{\boldsymbol{\theta}}$ of depth $L$ with parameters $\boldsymbol{\theta}$ (the weights and biases), representing the learned quantity of interest. Let $\{R_i\}_{i=1}^{N}$ be the set of $N$ disjoint convex regions, each described by its vertices $\boldsymbol{v}_p^{(i)}$. We wish to ensure that each region $R_i$ is contained in a distinct activation cell of the piecewise-affine decomposition induced by the network.

Concretely, we introduce sign variables

$$\operatorname{sign}_n^{(i,\ell)} \in \{+1, -1\}, \tag{7}$$

where $\ell \in \{1, \ldots, L-1\}$ indexes the layer and $n$ indexes the neuron in layer $\ell$. The sign variable $\operatorname{sign}_n^{(i,\ell)}$ encodes that region $R_i$ is placed entirely in the half-space defined by

$$\operatorname{sign}_n^{(i,\ell)} \left( \mathbf{w}_n^{(\ell)\top} \boldsymbol{v}_p^{(i,\ell)} + b_n^{(\ell)} \right) \geq \delta, \quad \forall\, p \in \{1, \ldots, P_i\} \tag{8}$$

with $\boldsymbol{v}_p^{(i,\ell)}$ denoting the vertices after passing through $\ell - 1$ layers and $\delta \geq 0$ a small margin. When $\delta > 0$, each region is strictly confined to the interior of its activation cell; when $\delta = 0$, regions may touch cell boundaries but affine behavior within each region is still guaranteed. To force each $R_i$ into a *unique* activation cell, no two regions may share the same global sign pattern.

Formulating these requirements as constraints, we can define the following non-convex optimization problem:

$$\begin{aligned} \min_{\{\mathbf{w}_n^{(\ell)},\, b_n^{(\ell)},\, \operatorname{sign}_n^{(i,\ell)}\}} \quad & \Phi\big(\boldsymbol{\theta}\big) \quad \text{subject to} \\ & \operatorname{sign}_n^{(i,\ell)} \Big( \mathbf{w}_n^{(\ell)\top} \boldsymbol{v}_p^{(i,\ell)} + b_n^{(\ell)} \Big) \;\geq\; \delta, \;\; \forall\, p, i, n, \ell, \\ & \forall\, i \neq j, \;\; \exists\, n, \ell \text{ such that } \;\; \operatorname{sign}_n^{(i,\ell)} \;\neq\; \operatorname{sign}_n^{(j,\ell)}. \end{aligned}$$

where $\Phi(\boldsymbol{\theta})$ is an objective function reflecting the primary learning task, such as data mismatch plus regularization terms. The two sets of constraints can be described as *region-consistency constraints* and *uniqueness constraints*, respectively. The former enforces that for each region $R_i$, all its vertices consistently adhere to the assigned sign pattern $\operatorname{sign}_n^{(i,\ell)}$ for each neuron $(n, \ell)$, while the latter enforces that no two distinct regions share the exact same global sign pattern across all neurons and layers.

Solving this mixed-integer problem with binary variables is NP-hard. In practice, we therefore first choose $\operatorname{sign}_n^{(i,\ell)}$ heuristically and then solve simpler subproblems (e.g., quadratic or linear programs) to enforce the assigned half-space constraints by adjusting $\{\mathbf{w}_n^{(\ell)}, b_n^{(\ell)}\}$ at each layer separately.

### 2.5 Strategies for Sign Assignment

We propose two heuristic methods for determining each region's signs:

**Majority voting.** For each region $R_i$, we examine its vertex pre-activations $\{z_n^{(\ell)}(\boldsymbol{v}_p^{(i,\ell)})\}$ at layer $\ell$. We then set $\operatorname{sign}_n^{(i,\ell)} = +1$ if the majority of $\{z_n^{(\ell)}(\boldsymbol{v}_p^{(i,\ell)})\}$ are positive; otherwise, we choose $-1$. Exactly zero pre-activations are negligible in practice and are treated as positive by convention. This simple, low-cost heuristic often provides reliable region separation, especially when each neuron has a clear tendency to be either positive or negative over $R_i$.

**Mean-based pre-activation rule.** For each region $R_i$ and neuron $n$ in layer $\ell$, we compute the average of the pre-activations over the vertices of $R_i$. Specifically, let

$$m_n^{(i,\ell)} = \frac{1}{P_i} \sum_{p=1}^{P_i} z_n^{(\ell)}\big(\boldsymbol{v}_p^{(i,\ell)}\big)$$

where $z_n^{(\ell)}(\boldsymbol{v}_p^{(i,\ell)}) = \mathbf{w}_n^{(\ell)\top} \boldsymbol{v}_p^{(i,\ell)} + b_n^{(\ell)}$ is the pre-activation of neuron $n$ at layer $\ell$, with $\boldsymbol{v}_p^{(i,\ell)}$ denoting the vertex after propagation through $\ell - 1$ layers. We then set $\text{sign}_n^{(i,\ell)} = +1$ if $m_n^{(i,\ell)} > \epsilon$ and $-1$ otherwise, where $\epsilon \geq 0$ is a small margin that prevents sign ambiguity when $m_n^{(i,\ell)}$ is close to zero.

Selecting the right approach depends on how pre-activations distribute across vertices. The mean-based method works well when they cluster around distinct positive or negative values, making outliers less influential. Majority voting is more suitable if nearly all vertices share the same sign. It is robust to small sets of outliers but can become unstable if the region straddles the boundary, where a near-even split may flip the result.

After assigning initial sign patterns $\{\text{sign}_n^{(i,\ell)}\}$ to each region $R_i$, we ensure no two regions share the same global pattern. If duplicates occur, often due to nearby constrained regions, we flip the signs of a few neurons (with near-zero pre-activations) in one layer of one region to guarantee uniqueness.

### 2.6 Enforcing Multi-Region Affinity via Sign Patterns

This subsection concerns only the geometric stage of mPOLICE: maintaining sign patterns so that each target patch stays inside its designated activation cell. Once the region-wise sign assignments are fixed, the enforcement problem reduces to a standard projection onto margin-separated half-space inequalities in parameter space. The mPOLICE-specific contribution is therefore the reduction from multi-region affinity to these signed half-space constraints; the actual projection can then be carried out by standard methods such as QP, POCS, or ADMM. These operations do not yet impose the downstream physical or task constraints; rather, they preserve the localized affine structure that makes those constraints tractable.

**Quadratic programming (QP).** Fix a layer $\ell$ and a neuron $n$ with current parameters $(\mathbf{w}_n^{(\ell)}, b_n^{(\ell)})$. For each region $R_i$ with vertices $\{\boldsymbol{v}_p^{(i,\ell)}\}$ and prescribed sign $\text{sign}_n^{(i,\ell)} \in \{\pm 1\}$, enforcing a margin $\delta \geq 0$ amounts to the half-space conditions

$$\text{sign}_n^{(i,\ell)} \left(\mathbf{w}_n^{(\ell)\top} \boldsymbol{v}_p^{(i,\ell)} + b_n^{(\ell)}\right) \;\geq\; \delta, \quad \forall p, i.$$

Seeking the smallest change from the current parameters in Euclidean norm yields, for each neuron,

$$\begin{aligned} \min_{\Delta\mathbf{w}_n^{(\ell)},\, \Delta b_n^{(\ell)}} \quad & \left\|\Delta\mathbf{w}_n^{(\ell)}\right\|_2^2 + \left(\Delta b_n^{(\ell)}\right)^2 \\ \text{s.t.} \quad & \text{sign}_n^{(i,\ell)} \left((\mathbf{w}_n^{(\ell)} + \Delta\mathbf{w}_n^{(\ell)})^\top \boldsymbol{v}_p^{(i,\ell)} + (b_n^{(\ell)} + \Delta b_n^{(\ell)})\right) \geq \delta, \;\; \forall p, i. \end{aligned}$$

This strictly convex QP is a projection of the current parameters onto the intersection of half-spaces; its solution gives the minimal perturbation that achieves the target signs with margin.

**Projection onto Convex Sets (POCS).** Because each sign constraint is just a half-space in the augmented parameter vector $\theta = [\,\mathbf{w}^\top \; b\,]^\top$, we can also use standard POCS: repeatedly project onto the violated half-spaces until the iterates stabilize. In practice, this is attractive because the update is simple, numerically stable, and often converges in a few sweeps when only a small subset of vertex constraints is active. We therefore use POCS as a lightweight projection backend rather than as a new algorithmic ingredient.

**Batched ADMM projection.** We also employ standard ADMM splitting (Gabay & Mercier, 1976) to solve the Euclidean projection layer-wise. By stacking vertex constraints, ADMM alternates between a quadratic parameter update and projection onto the inequality cone $z \geq \delta\mathbf{1}$. Since the linear system matrix is shared across neurons in a layer, its factorization can be reused, offering a scalable solver for this subproblem.

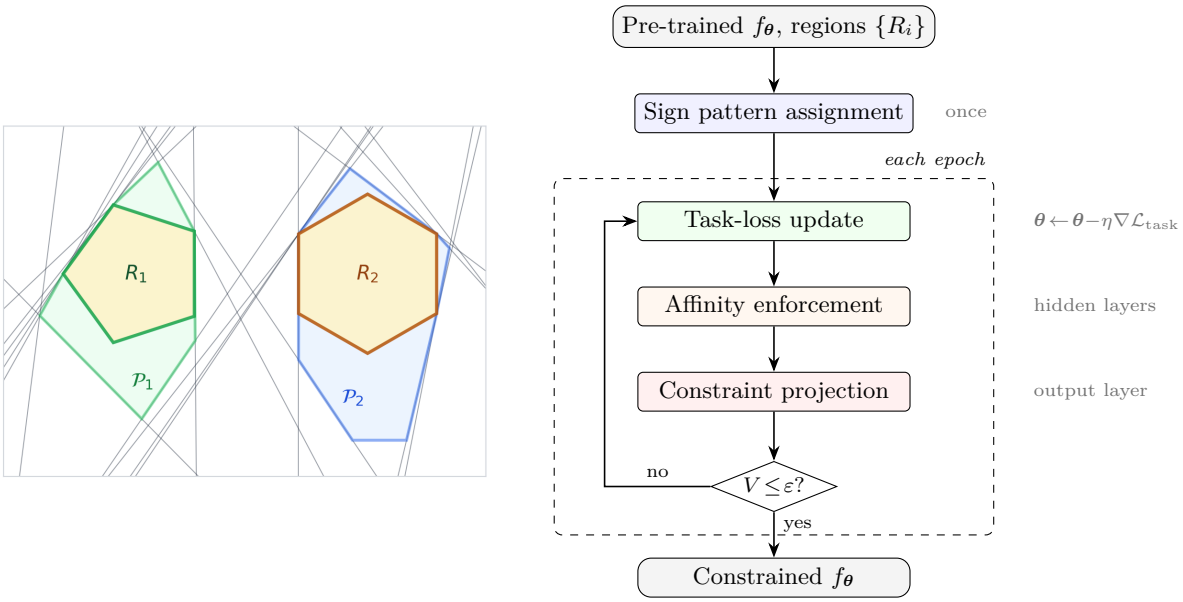


Figure 2: *Left:* Target regions (yellow) and their containing activation cells (shaded); cells are generally larger than the regions. *Right:* mPOLICE training loop: sign patterns are assigned once, then each epoch alternates task-loss updates, affinity enforcement, and constraint projection until $V \leq \varepsilon$.

### 2.7 Target Region vs. Activation Cell

Sign-pattern enforcement guarantees that each target region $R_i$ is contained in a single activation cell, but the cell can be strictly larger since its facets are determined by the network's ReLU boundaries, not by $R_i$ (Figure 2). The gap shrinks with capacity (more neurons yield finer partitions) and training (each enforcement cycle shifts ReLU boundaries toward $R_i$). Exact matching would require every facet of $R_i$ to coincide with an active ReLU boundary, but the mismatch is benign, as the affine constraint holds on $R_i$ regardless of cell size.

### 2.8 Training with Separate Affinity Enforcement and Constraint Satisfaction

After sign-pattern assignment (Section 2.5) and enforcement (Section 2.6), the network is affine on each $R_i$: $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\Lambda}_i \boldsymbol{x} + \boldsymbol{\gamma}_i$ for $\boldsymbol{x} \in R_i$. Because an affine map on a convex polytope is determined by its vertex values, let $B_i \in \mathbb{R}^{(D+1)\times(D+1)}$ stack $D{+}1$ affinely independent vertices of $R_i$ (each augmented with a trailing 1) and $Y_i$ stack the corresponding outputs:

$$\begin{bmatrix} \boldsymbol{\Lambda}_i^\top \\ \boldsymbol{\gamma}_i^\top \end{bmatrix} = B_i^{-1} Y_i. \tag{9}$$

Any constraint linear in $(\boldsymbol{\Lambda}_i, \boldsymbol{\gamma}_i)$ therefore reduces to a linear system on vertex outputs: $\tilde{\boldsymbol{E}}_i \operatorname{vec}(Y_i) = \tilde{\boldsymbol{f}}_i$ (equalities) and $\tilde{\boldsymbol{C}}_i \operatorname{vec}(Y_i) \leq \tilde{\boldsymbol{d}}_i$ (inequalities). This is the key link between the two stages.

The training combines both (Figure 2, right). Starting from a pre-trained $f_{\boldsymbol{\theta}}$ with regions $\{R_i\}$ and their vertices, sign patterns are first enforced via Section 2.6. Each epoch then minimizes $\mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{con}}$ over mini-batches, where $\mathcal{L}_{\text{con}}$ accumulates vertex-constraint penalties across all regions, followed by sign-pattern re-enforcement and violation measurement $V$. Training terminates early when $V \leq \epsilon$. If violations persist beyond a patience window, $\lambda$ is increased and patience reset. The best model by $\mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{con}}$ is restored at completion.

### 2.9 Projected Training and Output-Layer Projection

As an alternative to the penalty-based loop, we implement a projected training variant. Each epoch performs $T$ steps of task-only SGD ($\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla \mathcal{L}_{\text{task}}$), then monitors the minimum signed margin across all regions.

Table 1: Spiral benchmark with two disjoint inequality-constrained disks. All values are scaled by the factor shown in each column header. Bold marks the best mean; mean ± std over 3 seeds.

| Method | MSE Out ($\times 10^{-4}$) | MSE All ($\times 10^{-4}$) | Max Viol. ($\times 10^{-2}$) | Affine RMSE ($\times 10^{-2}$) | Train (s) | Infer (ms) |
|---|---|---|---|---|---|---|
| Unconstrained | $4.83 \pm 1.64$ | $4.68 \pm 1.58$ | $6.61 \pm 0.21$ | $2.48 \pm 0.02$ | $63.7 \pm 0.8$ | $0.24 \pm 0.02$ |
| Soft Penalty ($\lambda$=1) | $4.73 \pm 1.57$ | $4.59 \pm 1.51$ | $5.53 \pm 0.46$ | $2.42 \pm 0.05$ | $105.7 \pm 0.9$ | $0.24 \pm 0.02$ |
| Soft Penalty ($\lambda$=10) | $4.76 \pm 1.53$ | $4.63 \pm 1.48$ | $3.92 \pm 0.53$ | $2.36 \pm 0.08$ | $105.6 \pm 0.8$ | $0.23 \pm 0.02$ |
| Soft Penalty ($\lambda$=100) | $4.78 \pm 1.58$ | $4.67 \pm 1.52$ | $1.95 \pm 0.16$ | $2.34 \pm 0.06$ | $104.6 \pm 1.5$ | $0.25 \pm 0.02$ |
| HardNet-Aff | $6.76 \pm 2.91$ | $6.57 \pm 2.82$ | $0.00 \pm 0.00$ | $2.40 \pm 0.06$ | $148.1 \pm 0.9$ | $11.52 \pm 0.07$ |
| mPOLICE-POCS | $66.7 \pm 12.6$ | $64.8 \pm 12.0$ | $0.00 \pm 0.00$ | $0.00003 \pm 0.00001$ | $100.4 \pm 0.6$ | $0.23 \pm 0.02$ |

When this margin drops below a threshold $\delta_{\text{trigger}}$, parameters are re-projected onto the sign-feasible half-spaces via QP, POCS, or ADMM. A second projection on the final linear layer then enforces vertex constraints: for equalities $A\boldsymbol{\theta} = c$, the minimal-change projection is $\boldsymbol{\theta}^\star = \boldsymbol{\theta}_0 - A^\top (AA^\top)^\dagger (A\boldsymbol{\theta}_0 - c)$; inequalities use the same backends. After projection, we optionally fine-tune the output layer in the constraint tangent space using projected gradients $g_{\text{proj}} = g - A^\top (AA^\top)^\dagger (Ag)$ to recover task accuracy without violating feasibility.

## 3 Constraint Families Across Domains

Once affinity holds on $R_i$, any constraint linear in $(\boldsymbol{\Lambda}_i, \boldsymbol{\gamma}_i)$ reduces to a linear system on vertex outputs. For codimension-one boundaries, the boundary is thickened into a thin strip $R^\varepsilon$ (e.g., $[0, \varepsilon] \times [y_{\min}, y_{\max}]$) on which affinity is enforced. Dirichlet conditions reduce to fixed vertex values, Neumann conditions to linear relations on those vertices (Appendix F), and Robin conditions likewise (Section 4.3). Beyond PDEs, the same template also covers RL safety inequalities and occupancy constraints, while Section 4.2 shows the operator-learning instantiation.

## 4 Experiments

The complete source code for all experiments is included in the supplementary material; we refer the reader there for full implementation details, hyperparameters, and reproducibility instructions.

**Constraint violation.** Violation is $\|\boldsymbol{E}_i f_{\boldsymbol{\theta}} - \boldsymbol{f}_i\|$ for equalities and $\max(\boldsymbol{C}_i f_{\boldsymbol{\theta}} - \boldsymbol{d}_i, 0)$ for inequalities, evaluated at region vertices for mPOLICE and on dense samples for baselines. The Robin experiment reports the mean squared boundary-condition residual.

### 4.1 Baseline Comparison: Region-Wise Affinity vs. Pointwise Feasibility

Before turning to physics applications, we isolate what distinguishes mPOLICE from existing constraint-enforcement methods. We benchmark against soft penalties ($\lambda \in \{1, 10, 100\}$) and HardNet-Aff (Min & Azizan, 2024) on a 2D spiral regression task with two disjoint sign constraints (left disk: $f \geq 0$, right disk: $f \leq 0$). A ReLU MLP (5 hidden layers, width 32) learns the spiral field on $[-2, 2]^2$; all methods start from the same pretrained checkpoint and use the same fine-tuning budget. We use mPOLICE-POCS here and benchmark all three backends in Section 4.6.

We report four metrics: MSE Outside (complement of constrained disks), MSE Overall, Max Violation (largest sign-constraint violation), and Affine RMSE (best-fit affine residual on the disks). MSE Outside is particularly informative because the activation cell enforced by mPOLICE can extend beyond the target region (Section 2.7), so overall MSE penalizes mPOLICE for affine restrictions on the larger cell.

Table 1 reveals the feasibility-versus-affinity gap. Soft penalties achieve the lowest regression MSE but leave nonzero violations and Affine RMSE around $2.3 \times 10^{-2}$: the function is feasible at sampled points but far from affine on the region. HardNet-Aff eliminates violations but its Affine RMSE is unchanged from the unconstrained baseline ($2.40 \times 10^{-2}$), confirming that it enforces pointwise feasibility without establishing region-wise affinity. mPOLICE-POCS drives Affine RMSE to $3.04 \times 10^{-7}$, five orders of magnitude below all other methods, while matching HardNet-Aff's zero violation and achieving the fastest inference time.

mPOLICE's higher MSE is a direct and expected consequence of the stronger guarantee it provides. By locking the network to a single affine map on each activation cell—which is generally larger than the target region (Figure 2)—the model sacrifices local flexibility in and around the constrained disks. This is the fundamental trade-off between provable region-wise affinity and unconstrained regression accuracy, and it is inherent to any method that truly makes the function affine rather than merely feasible at evaluation points. Importantly, the MSE increase remains below $7 \times 10^{-3}$ on a field of order-one magnitude, and the gap narrows with network capacity as finer ReLU partitions more tightly enclose the target regions.

### 4.2 Neural Operator with Internal Dirichlet Constraints

We apply mPOLICE to a DeepONet (Lu et al., 2021a) that learns the solution operator $\mathcal{G} : f \mapsto u$ for the Poisson equation $-\Delta u = f$ on $[0,1]^2$, with two disjoint internal Dirichlet pads $R_1, R_2$ on which $u = 0$. The DeepONet factorizes the operator as $u_\theta(f, y) = \mathbf{b}(f)^\top \mathbf{t}(y)$, where $\mathbf{b}$ is a branch net mapping forcing coefficients to basis weights and $\mathbf{t}$ is a trunk net mapping coordinates to basis functions.

mPOLICE is applied only to the trunk, using ADMM for sign-pattern enforcement. For each pad $R_i$, we enforce affinity on the trunk and project its output layer so that $\mathbf{t}(v) = \mathbf{0}$ at every vertex $v$ of $R_i$. The affine structure then guarantees $\mathbf{t}(y) \equiv \mathbf{0}$ throughout $R_i$, and therefore

$$u_\theta(f, y) = \mathbf{b}(f)^\top \mathbf{0} = 0 \quad \forall f,\ \forall y \in R_1 \cup R_2. \tag{10}$$

The constraint holds for every branch input, including unseen forcings, without modifying the branch network. Soft penalties do not provide this guarantee, as they only approximately suppress violations for forcings seen during training. The main training hyperparameters are listed in Appendix E.

Table 2 and Figure 3 summarize the results. Soft penalty baselines, which add a weighted constraint-violation term $\lambda \mathcal{L}_{\text{con}}$ to the task loss, are swept over $\lambda \in \{10, 100, 1000\}$. They reduce violations by 1–2 orders of magnitude relative to the unconstrained model but degrade accuracy (4.5–8.1%), forming a Pareto frontier. mPOLICE achieves a maximum violation of $6.18 \times 10^{-12}$, over **8 orders of magnitude** below the best soft variant, while maintaining 9.2% relative error. Because the trunk is affine on each pad and the vertex constraints are enforced exactly, the condition holds throughout $R_1 \cup R_2$ for every possible forcing $f$, up to numerical precision, rather than only at sampled training points.

Table 2: DeepONet results for the Poisson operator with two internal Dirichlet pads ($u = 0$).

| **Method** | **Rel-$L^2$ (mean $\pm$ std)** | **Max Violation (mean)** | **Mean Violation (mean)** |
|---|---|---|---|
| Base (no constraint) | $\mathbf{0.0203 \pm 0.0098}$ | $5.75 \times 10^{-3}$ | $5.90 \times 10^{-4}$ |
| Soft Penalty ($\lambda$=10) | $0.0460 \pm 0.0161$ | $8.82 \times 10^{-4}$ | $5.74 \times 10^{-5}$ |
| Soft Penalty ($\lambda$=100) | $0.0452 \pm 0.0150$ | $1.91 \times 10^{-4}$ | $1.44 \times 10^{-5}$ |
| Soft Penalty ($\lambda$=1000) | $0.0808 \pm 0.0281$ | $6.81 \times 10^{-5}$ | $1.56 \times 10^{-5}$ |
| mPOLICE | $0.0916 \pm 0.0300$ | $\mathbf{6.18 \times 10^{-12}}$ | $\mathbf{1.71 \times 10^{-12}}$ |

The Pareto plot (Figure 3, bottom left) illustrates the key distinction: soft penalties trade accuracy for reduced violations along a frontier but do not reach zero, whereas mPOLICE's constraint satisfaction is structural and independent of tuning. The 9.2% relative error reflects the reduced trunk capacity after freezing hidden layers, a cost inherent to exact enforcement. This trunk-only strategy, constraining only the coordinate-dependent component, cleanly separates operator learning (branch) from constraint enforcement (trunk) and applies to any architecture with a coordinate decoder.

### 4.3 Robin Boundary Condition

We show mPOLICE on a convective heat-transfer problem: steady-state conduction on $[0,1]^2$ with a localized heat source, Dirichlet boundaries on three sides, and a Robin (Newton cooling) condition $h\,T + k\,\partial_n T = h\,T_{\text{amb}}$ on a cooling-wall segment at $x$=0, $y \in [0.3, 0.7]$. This mixes value and normal-derivative terms, naturally testing the boundary-strip mechanism of Section 3. mPOLICE thickens the wall into a thin strip, enforces affinity via ADMM, and imposes the Robin law exactly through output-layer projection (details in Appendix I).

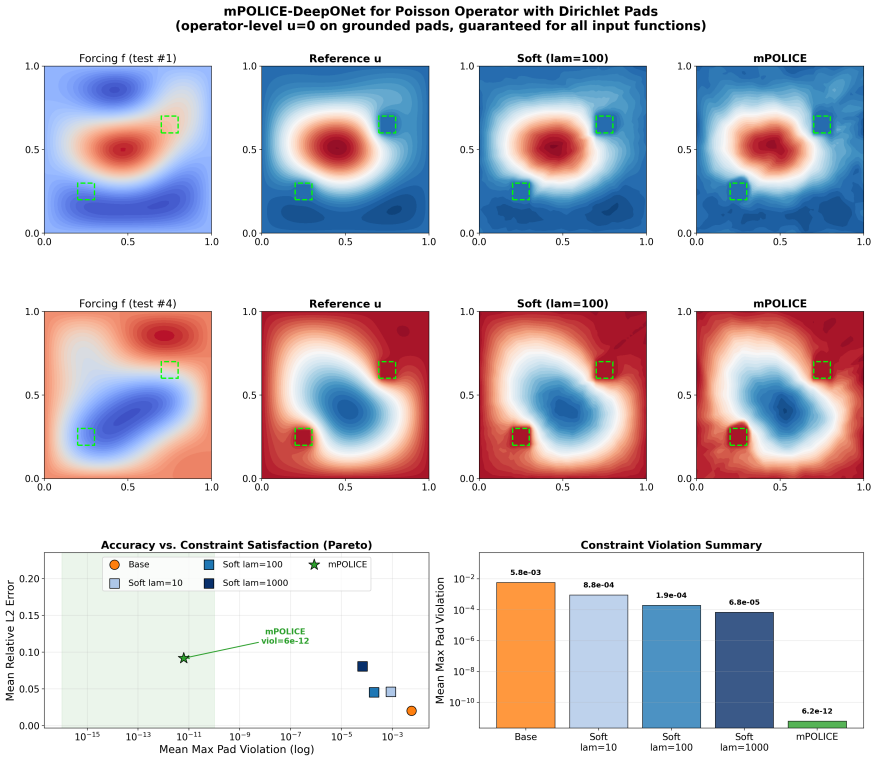


Figure 3: DeepONet with mPOLICE. Predicted solutions for two test fields *(top)*, Pareto trade-off between accuracy and constraint violation *(bottom left)*, and violation summary on a log scale *(bottom right)*.

Figure 4 shows mPOLICE drives Robin MSE to numerical precision ($\sim 10^{-30}$), over 24 orders of magnitude below the best soft variant ($7.6 \times 10^{-6}$ at $\lambda$=100), while interior accuracy ($1.5 \times 10^{-5}$) remains competitive. Increasing $\lambda$ beyond 100 degrades interior fit without improving satisfaction, showing penalty-method tuning sensitivity. Because the network is affine on the boundary strip, satisfaction holds at every wall point up to numerical precision, as the residual plot shows, whereas soft penalties leave spatially varying violations.

### 4.4 Additional Experiments

Several additional experiments are reported in the appendix: a Neumann boundary-strip example (Appendix F), implicit 3D occupancy-field learning with geometric hole constraints (Appendix H), a fluid-dynamics experiment enforcing zero-velocity obstacles (Appendix J), and illustrative classification and regression examples (Appendix M). Non-convex region handling is discussed in Appendix C. Of particular note is the impermeable-wall diffusion experiment (Appendix G), which illustrates how even a small boundary violation can be catastrophic: a soft-penalty model with a face residual of only $2.76 \times 10^{-3}$ allows enough flux leakage to destroy the downstream temperature shadow entirely (mean 0.494 vs. reference 0.078), while mPOLICE's exact zero-flux enforcement preserves it (mean 0.091).

### 4.5 Reinforcement Learning

In this section, we apply mPOLICE to a reinforcement-learning task that requires safety-critical policy constraints. An agent trained with the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm (Fujimoto et al., 2018) navigates a 2D environment toward a target while avoiding two static obstacles. This extends prior work (Bouvier et al., 2024b), which used POLICE for a single constrained buffer region, to the multi-region setting considered here. As in the rest of the paper, mPOLICE first assigns a distinct activation pattern to each buffer zone and enforces it via ADMM, making the policy locally affine there. It then enforces the safety inequalities on the resulting affine maps. Concretely, within each buffer zone $R_i$,

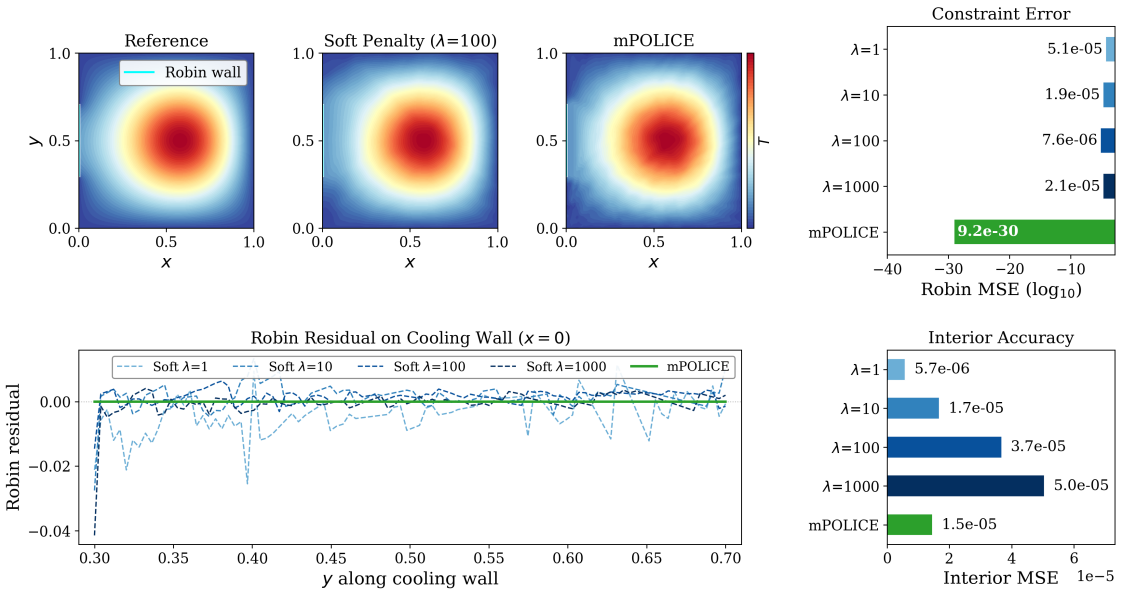


Figure 4: Robin example (convective heat transfer). Top row: reference, soft-penalty (λ=100), and mPOLICE solutions with constraint error; bottom row: Robin residual along the cooling wall and interior MSE.

the policy satisfies $\pi_{\boldsymbol{\theta}}(\boldsymbol{s}) = \boldsymbol{\Lambda}_i \boldsymbol{s} + \boldsymbol{\gamma}_i$ for $\boldsymbol{s} \in R_i$. A safety condition such as $\mathbf{e}_2^\top \pi_{\boldsymbol{\theta}}(\boldsymbol{s}) \leq 0$, which selects the vertical action component, can therefore be verified by checking only the vertices of $R_i$. Appendix D details the RL setup and specific constraints.

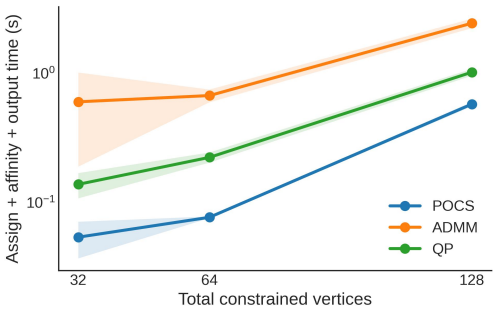


Figure 5: mPOLICE overhead vs. total constrained vertex count. This cost is training-time only.

Figure 6 (right) shows training dynamics. Enforcement begins around 16,000 timesteps, causing brief spikes in losses as affinity is restored and satisfaction of safe-action constraints is enforced—typical transients in constrained optimization. Despite this, reward steadily rises as the policy becomes safer and more successful. The left panel shows the learned policy and affine partitions, visualized via (Humayun et al., 2023; 2022). Blue arrows show actions; black lines mark activation-cell boundaries. The inset shows each red buffer region lies within one activation cell, indicating that mPOLICE enforces affine behavior there.

### 4.6 Computational Performance

We benchmark a single mPOLICE invocation on a frozen $4 \times 64$ ReLU MLP after spiral warm-up, fixing 32 vertices per region and varying region count (3 seeds). Each invocation performs (i) sign-pattern assignment, (ii) hidden-layer affinity enforcement, and (iii) output projection. Table 3 reports a mid-scale setup with 4 regions (128 vertices). Affinity enforcement dominates cost; output projection is nearly constant at $\sim 4 \times 10^{-2}$s.

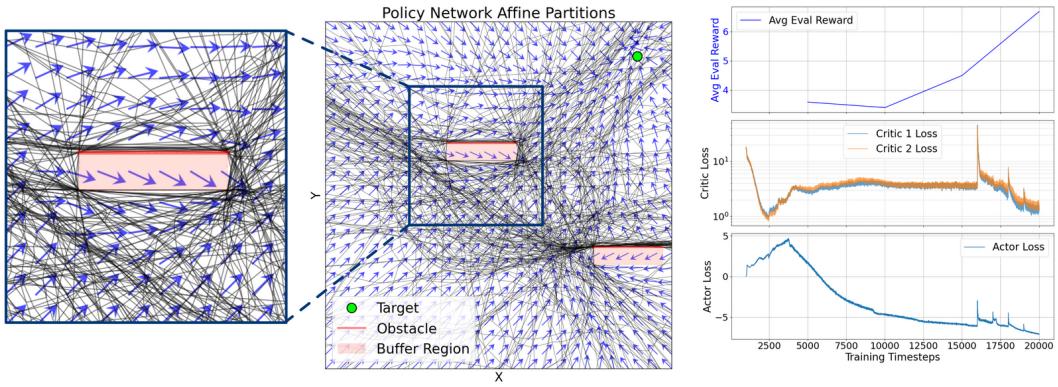


Figure 6: RL results. The learned policy (blue arrows) with activation-cell boundaries is shown on the left; insets show each buffer region lies in a single cell. Training curves are on the right.

Table 3: Per-call mPOLICE runtime on a $4 \times 64$ ReLU MLP, 4 regions (128 vertices). Mean $\pm$ std, 3 seeds.

| Backend | Sign Assignment (s) | Affinity Enforcement (s) | Output Projection (s) | Total Time (s) | Forward Time (ms) |
|---|---|---|---|---|---|
| POCS | $0.183 \pm 0.001$ | $0.353 \pm 0.024$ | $0.041 \pm 0.000$ | $0.577 \pm 0.023$ | $1.72 \pm 0.01$ |
| ADMM | $0.183 \pm 0.001$ | $2.223 \pm 0.207$ | $0.041 \pm 0.000$ | $2.447 \pm 0.206$ | $1.69 \pm 0.02$ |
| QP | $0.183 \pm 0.001$ | $0.797 \pm 0.060$ | $0.040 \pm 0.000$ | $1.020 \pm 0.061$ | $1.68 \pm 0.02$ |

POCS is fastest, with QP about 1.8× and ADMM about 4.0× slower. Figure 5 shows runtime grows with vertices but stays under one second for POCS and two seconds for QP/ADMM. Details in Appendix L.

## 5 Limitations

Limitations of the proposed framework should be noted. The sign-enforcement projection can be computationally intensive for larger networks or many vertices, and our implementation is not optimized for throughput. The method is restricted to piecewise-linear architectures (e.g., ReLU networks); extending it to smooth activations or Transformers remains open. Convolutional layers admit the same half-space formulation, but weight sharing couples constraints across spatial locations and gating constraints grow with channel $\times$ height $\times$ width, so practical CNN enforcement would require low-dimensional regions or cutting-plane screening. Geometrically, the enforced activation cell can be strictly larger than the region (Section 2.7), and because ReLU networks are piecewise affine and therefore not twice differentiable, PDE losses involving second-order derivatives cannot be used directly. Additionally, the weight perturbations from sign enforcement can introduce localized gradient artifacts $\|\nabla f - \nabla f_{\text{ref}}\|$ even when the function error $\|f - f_{\text{ref}}\|$ remains small.

## 6 Conclusion

We introduced mPOLICE, a framework that decouples two roles in hard-constrained learning: geometric affinity enforcement via unique sign-pattern assignment on multiple disjoint regions, and algebraic constraint enforcement via linear systems on vertex outputs. This perspective unifies boundary, interface, RL safety, occupancy, and operator-learning constraints under one patchwise-affine template, with zero inference overhead. By grounding constraint satisfaction in piecewise-affine geometry rather than penalty tuning, mPOLICE offers a practical path toward trustworthy models in safety-critical and physics-grounded applications.

## References

Mohammadmehdi Ataei and Hesam Salehipour. XLB: A differentiable massively parallel lattice Boltzmann library in python. *Computer Physics Communications*, 300:109187, 2024.

Randall Balestriero and Yann LeCun. Police: Provably optimal linear constraint enforcement for deep neural networks. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, June 2023. doi: 10.1109/ICASSP49357.2023.police.

Tom Beucler, Michael Pritchard, Stephan Rasp, Jordan Ott, Pierre Baldi, and Pierre Gentine. Enforcing analytic constraints in neural networks emulating physical systems. *Phys. Rev. Lett.*, 126:098302, Mar 2021. doi: 10.1103/PhysRevLett.126.098302. URL https://link.aps.org/doi/10.1103/PhysRevLett.126.098302.

Jean-Baptiste Bouvier, Kartik Nagpal, and Negar Mehr. Learning to provably satisfy high relative degree constraints for black-box systems. *arXiv preprint arXiv:2407.20456*, 2024a.

Jean-Baptiste Bouvier, Kartik Nagpal, and Negar Mehr. POLICEd RL: Learning closed-loop robot control policies with provable satisfaction of hard constraints. *arXiv preprint arXiv:2403.13297*, 2024b.

Franck Djeumou, Cyrus Neary, Eric Goubault, Sylvie Putot, and Ufuk Topcu. Neural networks with physics-informed architectures and constraints for dynamical systems modeling. In Roya Firoozi, Negar Mehr, Esen Yel, Rika Antonova, Jeannette Bohg, Mac Schwager, and Mykel Kochenderfer (eds.), *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, volume 168 of *Proceedings of Machine Learning Research*, pp. 263–277. PMLR, 23–24 Jun 2022. URL https://proceedings.mlr.press/v168/djeumou22a.html.

Weinan E and Bing Yu. The deep Ritz method: A deep learning-based numerical method for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Daniel Gabay and Bertrand Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & mathematics with applications*, 2(1):17–40, 1976.

Giorgio Giannone, Akash Srivastava, Ole Winther, and Faez Ahmed. Aligning optimization trajectories with diffusion models for constrained design generation. *Advances in Neural Information Processing Systems*, 36:51830–51861, 2023.

Ahmed Imtiaz Humayun, Randall Balestriero, and Richard Baraniuk. Exact visualization of deep neural network geometry and decision boundary. In *NeurIPS 2022 Workshop on Symmetry and Geometry in Neural Representations*, 2022.

Ahmed Imtiaz Humayun, Randall Balestriero, Guha Balakrishnan, and Richard Baraniuk. Splinecam: Exact visualization and characterization of deep network geometry and decision boundaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Ehsan Kharazmi, Zhongqiang Zhang, and George Em Karniadakis. Variational physics-informed neural networks for solving partial differential equations. In *arXiv preprint arXiv:1912.00873*, 2019.

Kota Kondo, Andrea Tagliabue, Xiaoyi Cai, Claudius Tewari, Olivia Garcia, Marcos Espitia-Alvarez, and Jonathan P How. CGD: Constraint-guided diffusion policies for UAV trajectory planning. *arXiv preprint arXiv:2405.01758*, 2024.

Andrei Konstantinov, Lev Utkin, and Vladimir Muliukha. Imposing star-shaped hard constraints on the neural network output. *Mathematics*, 12(23):3788, 2024.

James Kotary and Ferdinando Fioretto. Learning constrained optimization with deep augmented lagrangian methods. *arXiv preprint arXiv:2403.03454*, 2024.

James Kotary, Ferdinando Fioretto, Pascal Van Hentenryck, and Bryan Wilder. End-to-end constrained optimization learning: A survey. *arXiv preprint arXiv:2103.16378*, 2021.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhatt, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 26548–26560. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/df438e5206f31600e6ae4af72f2725f1-Paper.pdf.

Chong Li and C. J. Richard Shi. Constrained optimization based low-rank approximation of deep neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhatt, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2021.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021a.

Lu Lu, Raphaël Pestourie, Wenjie Yao, Zhicheng Wang, Francesc Verdugo, and Steven G. Johnson. Physics-informed neural networks with hard constraints for inverse design. *SIAM Journal on Scientific Computing*, 43(6):B1105–B1132, 2021b. doi: 10.1137/21M1397908.

Youngjae Min and Navid Azizan. Hardnet: Hard-constrained neural networks with universal approximation guarantees. *arXiv preprint arXiv:2410.10807*, 2024.

Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. *Advances in neural information processing systems*, 27, 2014.

Cyril Picard, Lyle Regenwetter, Amin Heyrani Nobari, Akash Srivastava, and Faez Ahmed. Generative optimization: A perspective on AI-enhanced problem solving in engineering. *arXiv preprint arXiv:2412.13281*, 2024.

Maziar Raissi, Paris Perdikaris, and George 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.

Zhe Tao and Aditya V Thakur. Provable editing of deep neural networks using parametric linear relaxation. *Advances in Neural Information Processing Systems*, 37:113846–113883, 2024.

Zhe Tao, Stephanie Nawas, Jacqueline Mitchell, and Aditya V Thakur. Architecture-preserving provable repair of deep neural networks. *Proceedings of the ACM on Programming Languages*, 7(PLDI):443–467, 2023.

Jesus Tordesillas, Jonathan P How, and Marco Hutter. Rayen: Imposition of hard convex constraints on neural networks. *arXiv preprint arXiv:2307.08336*, 2023.

Yuchen Xie, Honghang Chi, Yahui Wang, and Yu Ma. Physics-specialized neural network with hard constraints for solving multi-material diffusion problems. *Computer Methods in Applied Mechanics and Engineering*, 430:117223, 2024. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2024.117223. URL https://www.sciencedirect.com/science/article/pii/S0045782524004791.

Fangcheng Zhong, Kyle Fogarty, Param Hanji, Tianhao Wu, Alejandro Sztrajman, Andrew Spielberg, Andrea Tagliasacchi, Petra Bosilj, and Cengiz Oztireli. Neural fields with hard constraints of arbitrary differential order. *arXiv preprint arXiv:2306.08943*, 2023.