ACTIVE-SET PROJECTION LAYERS FOR HARD-CONSTRAINED NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Differentiable projection layers make a network's outputs satisfy hard constraints exactly, but the projection can dominate both training and inference cost, and with the fixed iterative budget used in practice its gradient loses accuracy as more constraints become tight. We show that, for any solver of the projection, the implicit-differentiation system is block-triangular with respect to the *active manifold* cut out by the tight constraints: the backward pass reduces to one solve on its normal and curved tangent directions, where the curvature enters. Building on this structure, we propose ASPL (Active-Set Projection Layer), which identifies the active manifold in the forward pass and differentiates through the reduced system. We prove finite-time identification and an $\mathcal{O}(1/T)$ end-to-end training rate. We evaluate ASPL on parametric quadratic programs, multi-robot motion planning, inverse kinematics on real teleoperation logs, maze trajectory planning, and constrained flow matching. Its exact gradient costs less than ambient implicit differentiation of equal accuracy, and on quadratic programs its accuracy holds as more constraints bind.

1 INTRODUCTION

Hard constraints arise across learning-based systems: joint and torque limits in robot control (Kumar et al., 2021; Nubert et al., 2020), friction cones and workspace boundaries in autonomous controllers (Ames et al., 2019; Agrawal et al., 2025), conservation laws in neural PDE solvers and generative models (Négiar et al., 2023; Utkarsh et al., 2025), state and action bounds in safe reinforcement learning (Dalal et al., 2018; Kasaura et al., 2023), and feasibility in resource allocation (Donti et al., 2021; Pan et al., 2021) and structured prediction (Wang et al., 2019; Niculae et al., 2018). In each, the network output must satisfy the constraints exactly at every context $\mathbf{x}$. Given training contexts $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathcal{X}$, we cast this as the *parametric constrained optimization* problem of learning a predictor $\mathbf{y}_{\boldsymbol{\theta}} : \mathcal{X} \to \mathbb{R}^d$ over closed feasible sets $\mathcal{C}(\mathbf{x}) \subseteq \mathbb{R}^d$ that minimizes a context-dependent objective $\varphi : \mathbb{R}^d \times \mathcal{X} \to \mathbb{R}$:

$$\min_{\boldsymbol{\theta}} \ \Phi(\boldsymbol{\theta}) \ \triangleq \ \frac{1}{N} \sum_{i=1}^{N} \varphi\big(\mathbf{y}_{\boldsymbol{\theta}}(\mathbf{x}_i),\, \mathbf{x}_i\big) \quad \text{s.t.} \quad \mathbf{y}_{\boldsymbol{\theta}}(\mathbf{x}_i) \in \mathcal{C}(\mathbf{x}_i), \ \ i = 1, \dots, N. \tag{PCO}$$

The objective φ may be a regression loss against the solution of a constrained problem or a per-context cost minimized directly, always under exact feasibility. We realize $\mathbf{y}_{\boldsymbol{\theta}}$ as a backbone $f_{\boldsymbol{\theta}} \colon \mathcal{X} \to \mathbb{R}^d$ composed with the metric projection, $\mathbf{y}_{\boldsymbol{\theta}}(\mathbf{x}) = \Pi_{\mathcal{C}(\mathbf{x})}(f_{\boldsymbol{\theta}}(\mathbf{x}))$ (Figure 1).

Prior work addresses (PCO) in four ways. Soft-penalty approaches (Raissi et al., 2019; Fioretto et al., 2021) add a violation term to the training loss, which is general but admits nonzero pointwise violation and requires per-problem weight tuning. Restricted parameterizations (Tordesillas et al., 2023; Min & Azizan, 2024) encode constraints into the architecture so feasibility holds by construction, but only for specific constraint classes (fixed convex sets or input-dependent affine constraints), not a general $\mathcal{C}(\mathbf{x})$. Correction-completion methods such as DC3 (Donti et al., 2021) predict a partial output and complete it heuristically, trading exactness for speed. Differentiable projection layers (Amos & Kolter, 2017; Agrawal et al., 2019; Grontas et al., 2026) are the fourth: feasibility is exact at every query, the constraint set may depend on $\mathbf{x}$, and gradients flow through the projection via the implicit function theorem (IFT). With many or curved constraints, the projection can dominate training and inference cost.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

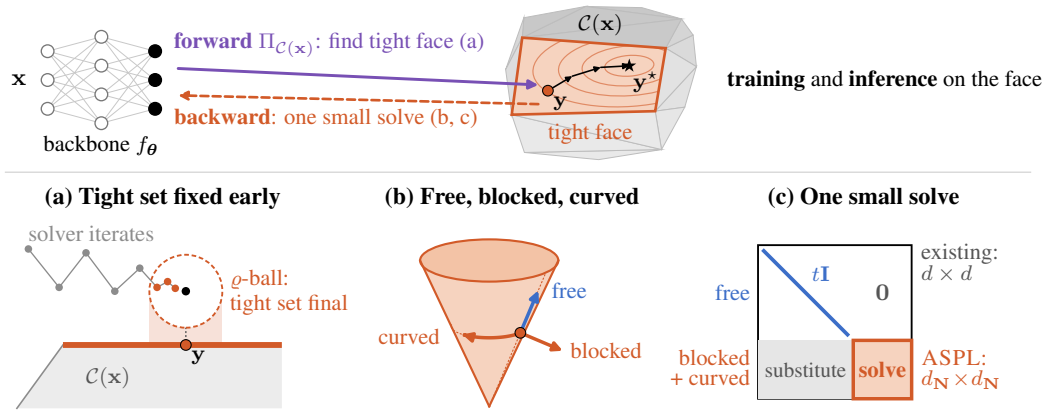

Figure 1: **ASPL overview.** *Top:* a backbone maps the context $\mathbf{x}$ to a prediction, and the projection layer returns the nearest feasible point $\mathbf{y}$, on a face where some constraints are tight (orange). The forward pass finds this face, the backward pass (dashed) reuses it with one small solve, and training moves $\mathbf{y}$ along the face toward $\mathbf{y}^\star$ (Theorem 3). *Bottom:* (a) The solver's iterates converge to a point that projects to $\mathbf{y}$; within ϱ of it, every projection lies on the tight face, so the tight set is final before convergence (Theorem 2). (b) At $\mathbf{y}$ some directions are free (blue), others blocked or curved (orange). (c) Only the blocked and curved directions need a linear solve, one $d_\mathbf{N} \times d_\mathbf{N}$ block instead of the full $d \times d$ system (Theorem 1).

These IFT-based layers solve a d-dimensional linear system on every backward pass, regardless of the projected geometry, and with the fixed Krylov budget used in practice (Grontas et al., 2026; Saad, 2003) the gradient error of ΠNet grows as more constraints bind (Section 4.1). Our key observation is that only the blocked and curved directions need a linear solve in the backward pass, so the solve scales with their number $d_\mathbf{N}$, not with d. Dropping inactive constraints from the backward pass is known for polyhedral or partly smooth problems with known support (Niculae et al., 2018; Bertrand et al., 2022). New here are a reduced form for curved and prox-regular sets under any forward solver, and identification by the layer's own solver before it converges. ASPL (*Active-Set Projection Layer*, Figure 1) identifies the *active manifold* $\mathcal{C}_\mathcal{A}(\mathbf{x})$ in the forward pass and reduces the backward pass to its normal block. Our contributions are:

(C1) Reduced IFT for the projection map (Theorem 1). We show that, under Assumptions A–D, the implicit-differentiation system of the projection onto an intersection $\mathcal{C}_1 \cap \mathcal{C}_2$ of simple sets is block lower-triangular, with a scalar block on the directions the tight constraints leave free, so the backward pass, for any solver, is one solve on the blocked and curved directions. On flat active manifolds the Jacobian is the tangent projector of $\mathcal{C}_\mathcal{A}(\mathbf{x})$. On curved ones the solve adds curvature.

(C2) Finite-time identification (Theorem 2). We prove that any solver whose block-projection arguments converge identifies $\mathcal{C}_\mathcal{A}(\mathbf{x})$ in finitely many iterations under partial smoothness and strict complementarity, after which the reduced backward is exact (Figure 1).

(C3) End-to-end convergence (Theorem 3). We prove that, with exact backward solves and a locally contracting forward solver, ASPL reaches an $\mathcal{O}(1/T)$ stationarity rate on (PCO) using $\mathcal{O}(\log T)$ forward steps per training iteration.

(C4) Experiments across applications (Section 4). We evaluate ASPL on parametric QPs, multi-robot motion planning, inverse kinematics on real teleoperation logs, maze trajectory planning, and constrained flow matching. Across these tasks, ASPL matches or outperforms existing projection layers and solvers in accuracy and cost.

2 Proposed Framework

A differentiable projection layer has two passes. The *forward pass* maps the backbone output to the nearest feasible point, usually with an iterative solver such as operator splitting (Grontas et al., 2026), Dykstra's algorithm (Boyle & Dykstra, 1986), or an interior-point method (Amos & Kolter, 2017; Agrawal et al., 2019). The *backward pass* sends the loss gradient back through this projection to the network, usually by implicitly differentiating the solver's optimality or fixed-point conditions,

at the cost of a $d \times d$ linear solve per sample. ASPL uses the active set found by any solver in both passes, restricting the forward solve to the active face and the backward solve to its normal block.

With $\mathbf{y}_{\text{raw}} \triangleq f_{\boldsymbol{\theta}}(\mathbf{x})$, $\mathbf{y}_{\boldsymbol{\theta}}(\mathbf{x}) = \Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}}) \in \mathcal{C}(\mathbf{x})$ by construction, so (PCO) is unconstrained in $\boldsymbol{\theta}$. For $\ell \triangleq \varphi(\cdot, \mathbf{x})$, the chain rule gives

$$\nabla_{\boldsymbol{\theta}} \ell \;=\; \frac{\partial \ell}{\partial \mathbf{y}} \;\cdot\; \frac{\partial \Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}})}{\partial \mathbf{y}_{\text{raw}}} \;\cdot\; \frac{\partial f_{\boldsymbol{\theta}}(\mathbf{x})}{\partial \boldsymbol{\theta}}. \tag{1}$$

The outer factors are cheap (an analytic loss and an autodiff backbone). The middle factor is the backward pass. Existing layers obtain it from the solver by implicit differentiation or unrolling (Amos & Kolter, 2017; Agrawal et al., 2019; Donti et al., 2021; Grontas et al., 2026). ASPL obtains it from the projection map itself.

2.1 The projection layer and its active manifold

The forward pass computes

$$\overline{\mathbf{y}} \;\triangleq\; \Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}}) \;=\; \operatorname*{arg\,min}_{\mathbf{z} \in \mathcal{C}(\mathbf{x})} \tfrac{1}{2}\|\mathbf{z} - \mathbf{y}_{\text{raw}}\|^2, \tag{2}$$

the best-approximation problem for $\mathcal{C}(\mathbf{x})$ (Bauschke & Combettes, 2017). The backward pass below depends only on the map $\Pi_{\mathcal{C}(\mathbf{x})}$, not on the procedure that evaluates it, so any solver returning $\overline{\mathbf{y}}$ is admissible (Proposition 1).

Two-set representation. Practical constraint sets are intersections of simple pieces. We write $\mathcal{C}(\mathbf{x}) = \mathcal{C}_1(\mathbf{x}) \cap \mathcal{C}_2(\mathbf{x})$ with closed-form block projections $\Pi_{\mathcal{C}_i}$ (affine $\cap$ box, affine $\cap$ Lorentz cone, affine $\cap$ product of obstacle complements); three or more pieces reduce to two by the product-space construction of Pierra (1984). For example, the QP feasible set $\{\mathbf{y} : \mathbf{A}\mathbf{y} = \mathbf{x},\ \mathbf{C}\mathbf{y} \le \mathbf{u}\}$ of Section 4.1 becomes, with slacks $\boldsymbol{\sigma} = \mathbf{u} - \mathbf{C}\mathbf{y}$, the intersection of the affine set $\mathcal{C}_1 = \{(\mathbf{y}, \boldsymbol{\sigma}) : \mathbf{A}\mathbf{y} = \mathbf{x},\ \mathbf{C}\mathbf{y} + \boldsymbol{\sigma} = \mathbf{u}\}$, projected by one linear solve, and the orthant $\mathcal{C}_2 = \mathbb{R}^d \times \mathbb{R}^p_{\ge 0}$, projected by clipping. A second-order-cone constraint takes a Lorentz cone as $\mathcal{C}_2$ instead. At the solution the residual splits into block normals,

$$\mathbf{y}_{\text{raw}} - \overline{\mathbf{y}} \;=\; \mathbf{n}_1 + \mathbf{n}_2, \qquad \overline{\mathbf{y}} = \Pi_{\mathcal{C}_1(\mathbf{x})}(\overline{\mathbf{y}} + \mathbf{n}_1), \qquad \overline{\mathbf{y}} = \Pi_{\mathcal{C}_2(\mathbf{x})}(\overline{\mathbf{y}} + t\,\mathbf{n}_2), \tag{3}$$

for some $t \in (0, 1]$. Each $\mathbf{n}_i$ is the part of the residual normal to block i, and every $t \in (0, 1]$ works when $\mathcal{C}_2$ is convex. The Jacobian below does not depend on the choice. The differentials of the two block projections at the points where (3) evaluates them,

$$\mathbf{P} \;\triangleq\; D\Pi_{\mathcal{C}_1(\mathbf{x})}(\overline{\mathbf{y}} + \mathbf{n}_1), \qquad \mathbf{Q} \;\triangleq\; D\Pi_{\mathcal{C}_2(\mathbf{x})}(\overline{\mathbf{y}} + t\,\mathbf{n}_2), \tag{4}$$

are the only objects the backward pass needs.

Any exact method for (2) may be used, provided it returns $\overline{\mathbf{y}}$ together with the split (3). For the training analysis we view the solver as a fixed-point iteration $\mathbf{s}^{(k+1)} = \mathcal{T}(\mathbf{s}^{(k)}, \mathbf{y}_{\text{raw}}; \mathbf{x})$ whose fixed point determines $\overline{\mathbf{y}}$ and the split. Appendix A2.2 gives the solvers used in our experiments.

Definition 1 (Active manifold)**.** *Let $\mathcal{C} \subseteq \mathbb{R}^d$ be closed and prox-regular at $\mathbf{y} \in \mathcal{C}$. A set $\mathcal{M} \ni \mathbf{y}$ is the* active manifold *of $\mathcal{C}$ at $\mathbf{y}$ if, near $\mathbf{y}$, $\mathcal{M} \subseteq \mathcal{C}$ is a C^2 manifold with normal space $N_{\mathcal{M}}(\mathbf{y}) = \operatorname{span} N_{\mathcal{C}}(\mathbf{y})$ and $N_{\mathcal{C}}$ is continuous along $\mathcal{M}$ at $\mathbf{y}$. That is, $\mathcal{C}$ is partly smooth at $\mathbf{y}$ relative to $\mathcal{M}$ (Lewis, 2002), and $\mathcal{M}$ is locally unique (Hare & Lewis, 2004). We write $\mathcal{C}_{\mathcal{A}}(\mathbf{x})$ for the active manifold of $\mathcal{C}(\mathbf{x})$ at $\overline{\mathbf{y}}$, $T_{\mathcal{C}_{\mathcal{A}}}$ for its tangent space, and $\mathcal{C}_{\mathcal{A},1}, \mathcal{C}_{\mathcal{A},2}$ for the active manifolds of the two blocks.*

The active set is the special case of a set cut out by constraint functions. Let $\mathcal{C}(\mathbf{x})$ be given by constraints $c_i(\mathbf{y}; \mathbf{x}) \le 0$ or $c_i(\mathbf{y}; \mathbf{x}) = 0$, $i = 1, \dots, m$, with C^2 functions c_i. The *active set* $\mathcal{A}(\mathbf{y}) \triangleq \{i : c_i(\mathbf{y}; \mathbf{x}) = 0\}$ contains every equality. If the gradients $\nabla c_i(\mathbf{y})$, $i \in \mathcal{A}(\mathbf{y})$, are linearly independent, the active manifold near $\mathbf{y}$ is $\{\mathbf{z} : c_i(\mathbf{z}; \mathbf{x}) = 0,\ i \in \mathcal{A}(\mathbf{y})\}$, of codimension $|\mathcal{A}(\mathbf{y})|$. A polyhedron needs no independence condition: its active manifold is the face with $\mathbf{y}$ in its relative interior. The definition needs no constraint functions: on a Lorentz cone the active manifold is $\{\mathbf{0}\}$ at the apex and the boundary without the apex at other boundary points.

Let $\mathbf{F}$ be the orthogonal projector onto the free directions $\mathcal{F} \triangleq \{\mathbf{v} : \mathbf{Q}\mathbf{v} = \mathbf{v}\}$, along which $\mathcal{C}_{\mathcal{A},2}$ is flat at $\overline{\mathbf{y}}$ (Figure 1b), and $\mathbf{N} \triangleq \mathbf{I} - \mathbf{F}$ with $d_{\mathbf{N}} \triangleq \operatorname{rank} \mathbf{N}$. We write $\mathbf{X}_{\mathbf{UV}} \triangleq \mathbf{UXV}$ and $\mathbf{u}_{\mathbf{U}} \triangleq \mathbf{Uu}$. On a polyhedral face $\mathbf{T} \triangleq \mathbf{F}$ and $\mathbf{N}$ are the tangent and normal projectors and $d_{\mathbf{N}}$ is the number of blocked coordinates. On a curved manifold $d_{\mathbf{N}}$ also counts the curved tangent directions.

2.2 Curvature-aware IFT for the projection map

The chain rule (1) routes $\nabla_{\boldsymbol{\theta}} \ell$ through $D\Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}})$. ASPL obtains it by differentiating the three identities (3), which every exact solver satisfies, not the fixed-point map of one iteration. Our results use four standing conditions at the projection $\overline{\mathbf{y}}$.

Assumption A (Prox-regular feasible set)**.** *For every context $\mathbf{x}$, $\mathcal{C}(\mathbf{x})$ is closed and prox-regular at $\overline{\mathbf{y}}$ with reach $R(\mathbf{x}) > 0$ (Poliquin et al., 2000), and* $\operatorname{dist}(\mathbf{y}_{\text{raw}}, \mathcal{C}(\mathbf{x})) < R(\mathbf{x})$.

Assumption B (Partial smoothness and prox-regularity)**.** *For every context $\mathbf{x}$, the blocks $\mathcal{C}_1(\mathbf{x}), \mathcal{C}_2(\mathbf{x})$ are prox-regular at $\overline{\mathbf{y}}$, and each is partly smooth there relative to its active manifold $\mathcal{C}_{\mathcal{A},i}(\mathbf{x})$ in the sense of Lewis (2002). If $\mathcal{C}_1(\mathbf{x})$ is nonconvex, the block normal $\mathbf{n}_1$ of (3) has norm below its prox-regularity radius at $\overline{\mathbf{y}}$.*

Assumption C (Strict complementarity)**.** *(i)* For one context. *At the given $(\mathbf{x}, \mathbf{y}_{\text{raw}})$ the residual lies in the relative interior of the normal cone, $\mathbf{y}_{\text{raw}} - \overline{\mathbf{y}} \in \operatorname{relint} N_{\mathcal{C}(\mathbf{x})}(\overline{\mathbf{y}})$, with margin $\nu > 0$, and the block normals are linearly independent,* $\operatorname{span} N_{\mathcal{C}_1}(\overline{\mathbf{y}}) \cap \operatorname{span} N_{\mathcal{C}_2}(\overline{\mathbf{y}}) = \{\mathbf{0}\}$. *(ii)* Across contexts. *Condition (i) holds with a common margin ν at every training context $\mathbf{x}_i$ and every $\boldsymbol{\theta}$ in a bounded open set $\mathcal{N}_\circ$.*

Assumption D (Reduced nondegeneracy)**.** *(i)* For one context. *With $\mathbf{K}_t \triangleq \mathbf{P} + \mathbf{Q}(t\mathbf{I} - \mathbf{P})$, the block $\mathbf{K}_{\mathbf{NN}} \triangleq \mathbf{N}\mathbf{K}_t\mathbf{N}$ is nonsingular on* $\operatorname{range} \mathbf{N}$. *(ii)* Across contexts. *Its condition number is bounded uniformly over the training contexts and $\boldsymbol{\theta} \in \mathcal{N}_\circ$.*

Theorem 1 (Curvature-aware reduced IFT operator)**.** *Suppose Assumptions A–D hold. Then $\Pi_{\mathcal{C}(\mathbf{x})}$ is differentiable at $\mathbf{y}_{\text{raw}}$ and*

$$D\Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}}) \;=\; t\,\mathbf{P}\,\mathbf{K}_t^{-1}\mathbf{Q}, \qquad \mathbf{K}_t \;=\; \mathbf{P} + \mathbf{Q}(t\mathbf{I} - \mathbf{P}). \tag{5}$$

The matrix $\mathbf{K}_t$ is block lower-triangular in $\mathbb{R}^d = \mathcal{F} \oplus \mathcal{F}^\perp$ with scalar leading block $t\mathbf{I}$ and $\mathbf{K}_{\mathbf{NN}} = t\,\mathbf{Q}_{\mathbf{NN}} + (\mathbf{I} - \mathbf{Q}_{\mathbf{NN}})\mathbf{P}_{\mathbf{NN}}$. Consequently, for a loss cotangent $\mathbf{c} = \partial\ell/\partial\mathbf{y}|_{\overline{\mathbf{y}}}$, the gradient with respect to the raw output is $\partial\ell/\partial\mathbf{y}_{\text{raw}} = t\,\mathbf{Q}\mathbf{v}$, where $\mathbf{v}$ solves $\mathbf{K}_t^\top \mathbf{v} = \mathbf{P}\mathbf{c}$ in two stages,

$$\mathbf{K}_{\mathbf{NN}}^\top \mathbf{v}_{\mathbf{N}} \;=\; (\mathbf{P}\mathbf{c})_{\mathbf{N}}, \qquad \mathbf{v}_{\mathbf{F}} \;=\; \tfrac{1}{t}\big[(\mathbf{P}\mathbf{c})_{\mathbf{F}} - \mathbf{K}_{\mathbf{NF}}^\top \mathbf{v}_{\mathbf{N}}\big], \tag{6}$$

at the cost of one $d_{\mathbf{N}} \times d_{\mathbf{N}}$ solve.

Once the tight constraints are known, the free directions need no solve, only the substitution in (6). Applying $\mathbf{P}$ reuses the factorization the forward pass already holds. Only the $d_{\mathbf{N}}$ blocked and curved directions need a linear solve, and the curvature of the set enters only there (Figure 1c). On flat manifolds the Jacobian is the orthogonal projector onto the tangent space of $\mathcal{C}_{\mathcal{A}}$ (Appendix A2.4).

The reduction pays off when $d_{\mathbf{N}} \ll d$, and not on a Lorentz cone of dimension k, where $k - 1$ of its k coordinates enter the dense block. Measured speedups are below the flop ratio $(d/d_{\mathbf{N}})^3$ and vanish as $d_{\mathbf{N}} \to d$ (Section 4).

Proposition 1 (Independence of the solver)**.** *Under Assumptions A–D, let the solver be a C^1 fixed-point map $\mathcal{T}$ whose fixed point returns $\overline{\mathbf{y}}$ through a C^1 read-out for all $\mathbf{y}_{\text{raw}}$ nearby. If $\mathbf{I} - D_{\mathbf{s}}\mathcal{T}$ is invertible there, implicit differentiation through $\mathcal{T}$ returns (5), so operator splitting, Dykstra's algorithm, and an active-set QP give the same gradient.*

2.3 Active-manifold training

Algorithm 1 summarizes training. The two passes need the face at different times. The backward needs only the face read off the current solver state: with $\mathbf{P}$ and $\mathbf{Q}$ taken at that state, the reduced solve of Theorem 1 equals implicit differentiation of the solver map in $\mathbb{R}^d$, even before the solver converges, so it runs at every step. The forward needs the face in advance. Each context keeps its active set from the previous iteration, projects onto its face, and runs the full solver only when the result violates a constraint or a multiplier has the wrong sign.

Algorithm 1 ASPL

Require: contexts $\{\mathbf{x}_i\}_{i=1}^N$, network $f_{\boldsymbol{\theta}}$, solver map $\mathcal{T}$, solver steps K, step size η

$\bar{\mathbf{s}} \leftarrow \mathbf{0}$; $\mathcal{A}_i \leftarrow \emptyset$ for all i ▷ empty faces: first pass is full
repeat
  for all i **in parallel do**
    $\mathbf{y}_{\text{raw}} \leftarrow f_{\boldsymbol{\theta}}(\mathbf{x}_i)$ ▷ backbone
    $\bar{\mathbf{y}}_i \leftarrow \Pi_{\mathcal{C}_{\mathcal{A}_i}}(\mathbf{y}_{\text{raw}})$ ▷ **reduced forward** on the face of $\mathcal{A}_i$
    if $\bar{\mathbf{y}}_i \notin \mathcal{C}(\mathbf{x}_i)$ **or** $\mathbf{y}_{\text{raw}} - \bar{\mathbf{y}}_i \notin N_{\mathcal{C}(\mathbf{x}_i)}(\bar{\mathbf{y}}_i)$ **then**
      $\mathbf{s}_i \leftarrow \mathcal{T}^K(\bar{\mathbf{s}}, \mathbf{y}_{\text{raw}}; \mathbf{x}_i)$; read $\bar{\mathbf{y}}_i, \mathcal{A}_i$ off $\mathbf{s}_i$ ▷ full forward, identify (Thm. 2)
    end if ▷ accepted ⇒ exact projection
    $\mathbf{r}_i \leftarrow t\,\mathbf{Q}\mathbf{v}_i$ with $\mathbf{v}_i$ from (6), $\mathbf{c} = \nabla\ell(\bar{\mathbf{y}}_i)$ ▷ **reduced backward** (Thm. 1)
    $\mathbf{g}_i \leftarrow \left(\partial f_{\boldsymbol{\theta}}(\mathbf{x}_i)/\partial\boldsymbol{\theta}\right)^\top \mathbf{r}_i$ ▷ pull back to $\boldsymbol{\theta}$
  end for
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \frac{\eta}{N}\sum_i \mathbf{g}_i$; $\bar{\mathbf{s}} \leftarrow$ mean of the updated $\mathbf{s}_i$ ▷ step, warm start
until converged

In Algorithm 1, the contexts $\mathbf{x}_i$, the backbone $f_{\boldsymbol{\theta}}$, and the loss $\ell = \varphi(\cdot, \mathbf{x}_i)$ are those of (PCO), and η is the step size. $\mathcal{T}$ is the fixed-point map of the forward solver and $\mathcal{T}^K$ denotes K of its steps. Its state $\mathbf{s}_i$ determines the projection $\bar{\mathbf{y}}_i$, the split $(\mathbf{n}_1, \mathbf{n}_2, t)$ of (3), and the active set $\mathcal{A}_i$, whose face is the active manifold $\mathcal{C}_{\mathcal{A}}(\mathbf{x}_i)$ (Definition 1). The map $\Pi_{\mathcal{C}_{\mathcal{A}_i}}$ projects onto the face of $\mathcal{A}_i$. Initially $\mathcal{A}_i = \emptyset$, whose face is $\mathbb{R}^d$, so the projection returns $\mathbf{y}_{\text{raw}}$. The face projection returns the split through the multipliers of the active constraints. The multipliers have the right sign exactly when $\mathbf{y}_{\text{raw}} - \bar{\mathbf{y}}_i \in N_{\mathcal{C}(\mathbf{x}_i)}(\bar{\mathbf{y}}_i)$. Under Assumption A, feasibility and this normal-cone condition characterize the projection, so an accepted face projection is exact. The check fails when a parameter update moves the raw output across the boundary of the face. $\mathbf{P}$ and $\mathbf{Q}$ are the block Jacobians (4), $\mathbf{v}_i$ solves (6), $\mathbf{r}_i$ is the gradient with respect to the raw output, and $\mathbf{g}_i$ is its pull-back to $\boldsymbol{\theta}$. Our experiments train with the full solver at every step and apply the reduced backward to the face read off its final state, by Cholesky or CG. The feasibility and sign check runs only at evaluation in the polyhedral layers: when the solver output violates a constraint, they solve on the face and add violated rows and drop rows with wrong-sign multipliers. Appendix A2.2 gives the solvers and how each reads off $\mathcal{A}_i$, and Appendix A3.5 the pooled warm start.

3 Identification and Convergence

The reduced backward needs the active manifold. Any solver with convergent block-projection arguments finds it in finitely many steps.

Theorem 2 (Finite-time identification)**.** *Under Assumptions A–C, fix $\mathbf{y}_{\text{raw}} \in \mathbb{R}^d$ with split (3). There is $\varrho > 0$ such that each block projection maps every point at distance less than ϱ from its argument at the solution, $\bar{\mathbf{y}}+\mathbf{n}_1$ for $\mathcal{C}_1$ and $\bar{\mathbf{y}}+t\mathbf{n}_2$ for $\mathcal{C}_2$, into its active manifold $\mathcal{C}_{\mathcal{A},1}(\mathbf{x})$ or $\mathcal{C}_{\mathcal{A},2}(\mathbf{x})$. Hence, for any solver whose block-projection arguments converge to these points, the active manifold $\mathcal{C}_{\mathcal{A}}(\mathbf{x}) = \mathcal{C}_{\mathcal{A},1}(\mathbf{x}) \cap \mathcal{C}_{\mathcal{A},2}(\mathbf{x})$ (for constraint functions, the active set $\mathcal{A}$) is read off exactly after finitely many iterations.*

The solver therefore need not converge before the tight constraints are known. Once its iterates are within ϱ of the limit, the active set stops changing (Figure 1a). For a solver contracting at rate $1-\delta$ from a bounded start, with block-projection arguments Lipschitz in its state, this takes $\mathcal{O}(\delta^{-1}\log(1/\varrho))$ iterations. The proof applies the identification theorem of Hare & Lewis (2004) to the block projections.

From that iteration on, the reduced backward of Theorem 1 is exact.

Theorem 3 (Convergence of ASPL, informal)**.** *Let Φ be the objective of (PCO) with $\mathbf{y}_{\boldsymbol{\theta}} = \Pi_{\mathcal{C}(\cdot)} \circ f_{\boldsymbol{\theta}}$. Suppose Assumptions A–D hold at the training contexts, Φ has an L_Φ-Lipschitz gradient, the forward solver contracts toward its fixed point at rate $1-\delta$, the gradient readout is Lipschitz in the solver state, the backward solve is exact, and the iterates stay in $\mathcal{N}_\circ$. Then Algorithm 1 with $\eta = 1/L_\Phi$ and $K = \mathcal{O}(\delta^{-1}\log T)$ forward steps per iteration satisfies $\min_{\tau<T} \|\nabla\Phi(\boldsymbol{\theta}^{(\tau)})\|^2 = \mathcal{O}(1/T)$.*

Table 1: Parametric QP with d=2000 across active-set ratios. Relative suboptimality (RS) and total training wall-clock (s). All methods reach CV $\leq 10^{-4}$ except DC3, which fails the hard-feasibility bar at every ratio. ASPL stays flat in RS while ΠNet's RS grows by three orders of magnitude as the active set tightens. Batch size 64; ASPL entries are medians over four seeds. Best per metric in **bold**.

ratio	ASPL (Chol) RS / Time	ASPL (CG) RS / Time	ΠNet RS / Time	JAXOpt RS / Time	DC3 RS / Time
0.05	1.923e-4 / 90	1.698e-4 / **84**	**1.620e-4** / 172	1.880e-4 / 7886	6.084e+1 / 521
0.10	1.636e-4 / 105	**1.628e-4 / 90**	1.910e-4 / 171	2.623e-4 / 8217	1.160e+2 / 535
0.15	1.904e-4 / 128	**1.501e-4 / 104**	2.120e-4 / 184	3.208e-4 / 8135	1.578e+2 / 530
0.20	1.929e-4 / 147	**1.792e-4 / 116**	6.281e-4 / 176	4.943e-4 / 7938	1.976e+2 / 519
0.25	1.763e-4 / 178	**1.573e-4 / 127**	2.850e-3 / 178	6.798e-4 / 8285	2.398e+2 / 521
0.30	**1.599e-4** / 203	1.723e-4 / **135**	1.489e-2 / 186	1.316e-3 / 9560	2.820e+2 / 532
0.35	**2.276e-4** / 239	2.560e-4 / **150**	5.718e-2 / 193	6.573e-4 / 10630	3.334e+2 / 527
0.40	1.507e-4 / 267	**1.461e-4 / 162**	9.618e-2 / 193	1.347e-3 / 11547	3.758e+2 / 532
0.45	**1.504e-4** / 317	1.543e-4 / **183**	1.650e-1 / 193	1.025e-3 / 14049	4.284e+2 / 528

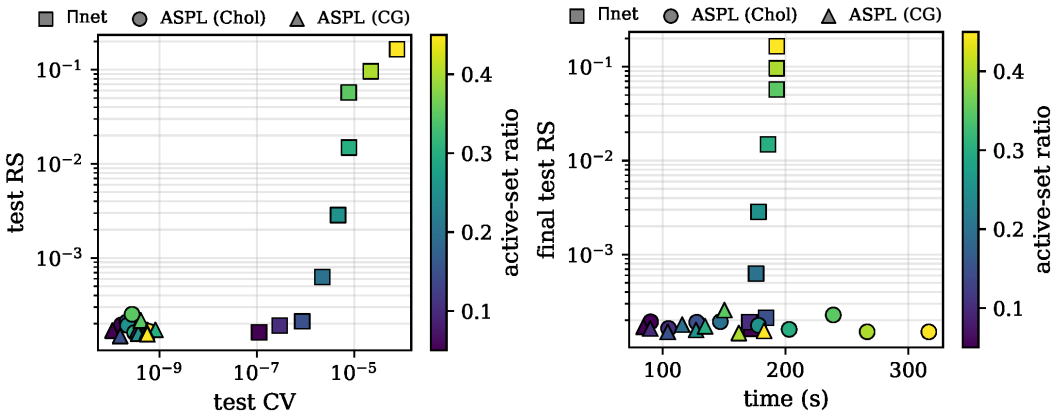

Figure 2: **QP across the active-set sweep** (color: active-set ratio). Left: test RS vs. test CV. Right: test RS vs. training wall-clock. The RS of ΠNet (squares) grows with the ratio, and both ASPL variants stay low.

Training with ASPL thus reaches the $\mathcal{O}(1/T)$ rate of gradient descent with exact gradients on smooth nonconvex objectives, with only $\mathcal{O}(\log T)$ projection-solver steps per iteration. The result is local: Assumption C(ii) keeps each training context on one active manifold, and iterates that move between manifolds are outside its scope.

4 Experiments

We evaluate ASPL on synthetic quadratic programs, where the exact optimum is known and the active-set ratio is controlled (Section 4.1), and on four robotics applications: multi-robot motion planning, inverse kinematics on real teleoperation logs, maze trajectory planning, and constrained flow matching for a generative policy (Sections 4.2–4.5). All learned methods share the backbone and data, feasibility is measured on the projected output, and GPU timings are on one A100. Appendix A4 gives full setups.

4.1 Parametric quadratic programming

We solve the parametric QP

$$\mathbf{y}^\star(\mathbf{x}) \;=\; \underset{\mathbf{y}\in\mathbb{R}^d}{\arg\min}\; \mathbf{y}^\top\mathbf{H}\mathbf{y} + \mathbf{q}^\top\mathbf{y} \quad \text{s.t.} \quad \mathbf{A}\mathbf{y} = \mathbf{x},\; \mathbf{C}\mathbf{y} \leq \mathbf{u}, \tag{7}$$

with $\mathbf{H} \succ 0$, d=2000 variables, n_{eq}=100 equalities, and n_{ineq}=2000 inequalities. The active-set ratio, the fraction of inequality constraints tight at the optimum, is swept from 0.05 to 0.45, and all

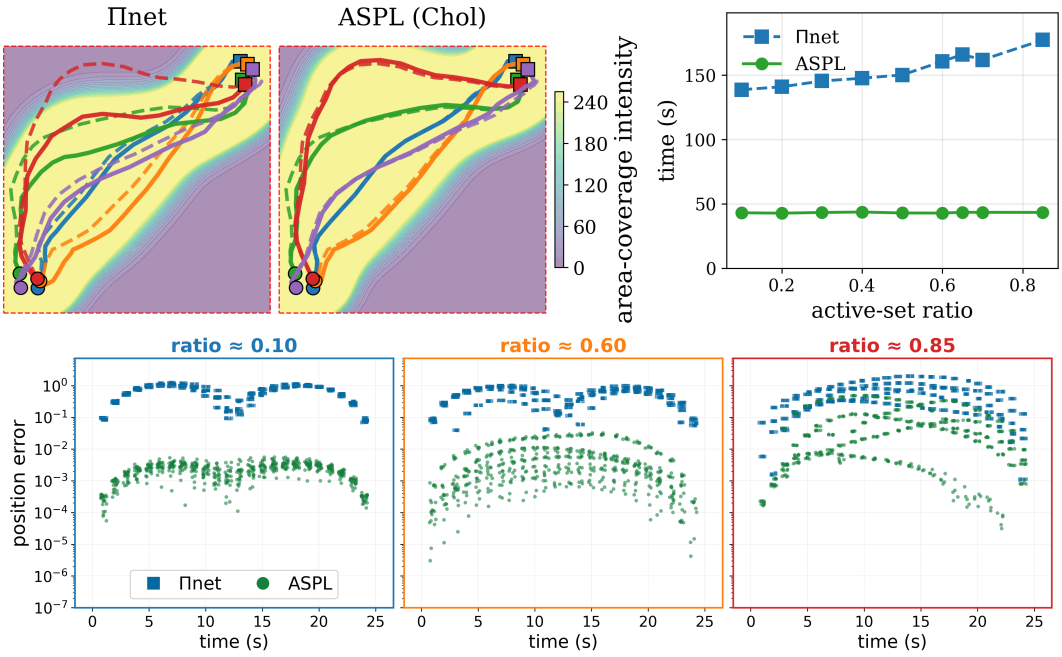

Figure 3: **Multi-robot motion planning.** Top left: predicted (solid) and ground-truth (dashed) trajectories on `r5h25` over the coverage map. ΠNet drifts from the ground truth and ASPL follows it. Top right: training wall-clock vs. active-set ratio on `r8h50`. The cost of ΠNet climbs and that of ASPL stays flat. Bottom: per-step position error on `r5h25` at three active-set ratios. The gap between ΠNet (squares) and ASPL (circles) narrows as the ratio grows.

methods share an MLP backbone trained against the ground truth (Appendix A4.1). We compare with ΠNet (Grontas et al., 2026), JAXOpt (Blondel et al., 2022), and DC3 (Donti et al., 2021), and run ASPL with a Cholesky (Chol) and a conjugate-gradient (CG) solve of its reduced system. We report the relative suboptimality RS of the objective of (7), clipped at zero because an infeasible point can reach a lower objective, the worst-case constraint violation CV over all equalities and inequalities, and the total training wall-clock.

ASPL keeps its accuracy as the constraints tighten (Table 1, Figure 2). Its RS stays near 2×10^{-4} over the whole sweep, while that of ΠNet grows by three orders of magnitude, and its constraint violation is several orders lower. Both variants train far faster than JAXOpt, and CG also trains faster than ΠNet at every ratio. DC3 never reaches the feasibility bar. The CG variant stays cheaper than Cholesky as the active set grows, because only Cholesky factors the growing normal block. With larger batches the gap narrows, and ΠNet is slightly more accurate at the smallest ratios (Appendix A4.1). The difference lies in the backward pass. At a fixed forward budget, the error of the Krylov gradient of ΠNet grows with the active-set ratio, while the reduced gradient of ASPL is exact once the active set is identified, as Theorems 1 and 2 predict, and cheaper to compute (Figure 9 in the appendix).

4.2 Multi-robot motion planning

We use the multi-robot benchmark of ΠNet (Grontas et al., 2026, §6.3–6.4): planar robots move from start to goal under workspace, velocity, acceleration, jerk, and pairwise collision-avoidance constraints, at two scales (`r5h25`: 5 robots over 25 steps, d=770; `r8h50`: 8 robots over 50 steps, d=2,432), with the active-set ratio $|\mathcal{A}|/d$ varied across cells. The network maps the start and goal configurations to all trajectories. Both methods train for 50 epochs with the same backbone and 200 forward iterations. ΠNet differentiates with 200 BiCGSTAB iterations at batch size 256, fixed by its reference runner, and ASPL uses batch size 512, so it takes half as many updates per epoch. ASPL follows the ground-truth trajectories where ΠNet drifts (Figure 3, top left). Its position error is orders of magnitude lower when few constraints bind, and the gap narrows as more of them bind (Figure 3, bottom). For 16 robots over 100 steps, ASPL's error stays below that of ΠNet over most

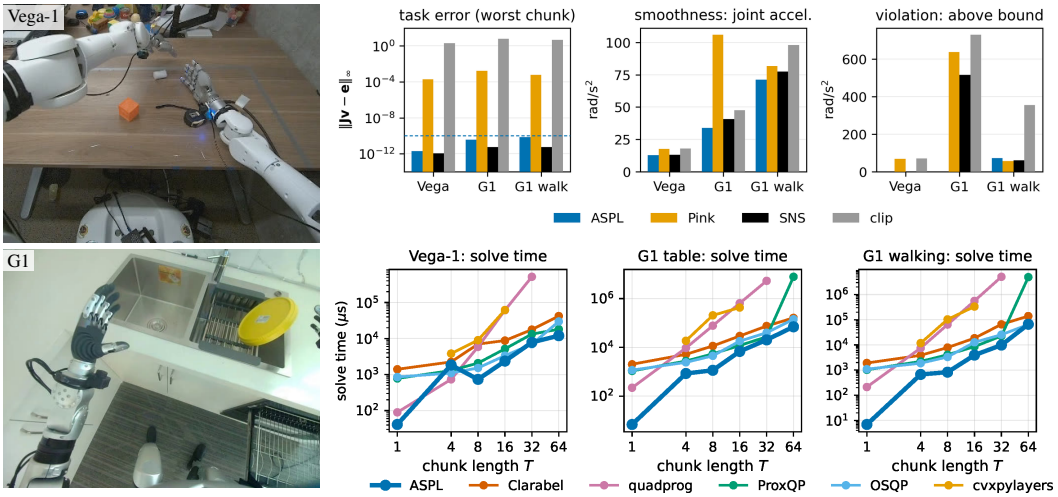

Figure 4: **IK on real logs.** Left: head-camera frames from the held-out teleoperation logs of the two robots. Right, top: worst-chunk task error $\|\mathbf{J}\mathbf{v} - \mathbf{e}\|_\infty$ (dashed: 10^{-10}), closed-loop joint acceleration, and bound violation of ASPL (blue) and per-frame controllers (clip: limits only). Right, bottom: solve time vs. chunk length against `Clarabel` (Goulart & Chen, 2026), `quadprog` (Goldfarb & Idnani, 1983), `ProxQP` (Bambade et al., 2022), `OSQP` (Stellato et al., 2020), and `cvxpylayers` (Agrawal et al., 2019); ASPL is below `Clarabel` at every length.

of the horizon at the three largest ratios (Appendix A4.3). The training time of ΠNet grows with the active-set ratio, while that of ASPL stays flat (Figure 3, top right).

4.3 Inverse Kinematics on Real Teleoperation Logs

Chunk-level differential IK projects a raw joint-velocity chunk onto the task equalities $\mathbf{J}(\mathbf{q}_t)\mathbf{v}_t = \mathbf{e}_t$, joint limits, and an acceleration coupling between frames. We run it in closed loop on held-out real teleoperation logs of a Dexmate Vega-1 (17 DoF; Niu et al., 2026) and a Unitree G1 humanoid (35 DoF, standing and walking; Unitree Robotics, 2026), with the same constraints for all methods. ASPL meets the task to below 10^{-10} on every chunk (Figure 4) and, in closed loop, tracks the operator to within a few millimeters, as the exact QP solver does. Under the same acceleration bound, the per-frame controllers Pink (Caron et al., 2024) and an SNS-style step (Flacco et al., 2015) drift by tens of millimeters on the two arm tasks. They track better only when walking, where the targets outrun the bound. Restricted to one frame, ASPL drifts as well, so its accuracy comes from planning over the whole chunk. With the bound ignored, the per-frame controllers track closely but violate it on every setting, while ASPL and `Clarabel` exceed it only when walking, where the targets outrun it. On the arm tasks ASPL moves the joints more smoothly (Figure 4). Batched, ASPL solves faster than `Clarabel` at every chunk length, and its gradient is three to four orders of magnitude cheaper than that of `cvxpylayers`, whose conic solver stops early and returns inaccurate gradients (Figure 4, bottom). As the output layer of an amortized IK network, it meets the task on every held-out frame and the joint limits on nearly all of them, whereas a regression head violates a limit on most frames.

4.4 Maze Trajectory Planning

The Graph-of-Convex-Sets (GCS) maze benchmark of Marcucci et al. (2023) plans a smooth path through a maze by choosing a sequence of free cells and fitting a Bézier curve inside them. We fix the cell path by breadth-first search and let ASPL amortize the remaining convex problem. An MLP maps the start and goal to the Bézier control points, and the layer projects them onto the polytope of the path, which pins the endpoints, sets the start and goal velocity to zero, enforces continuity up to second order at every cell junction, and keeps each control point inside its cell. We test a fixed maze with new start and goal points (S2) and a new maze for every query (S3), with maze sizes 10 to 30, against the exact solution of the same fixed-path problem computed with `Clarabel`. Every test output stays inside the corridor with the endpoints pinned (Figure 5). The control points are within a few percent of the fixed-path optimum, and the curves are slightly longer than optimal. On new

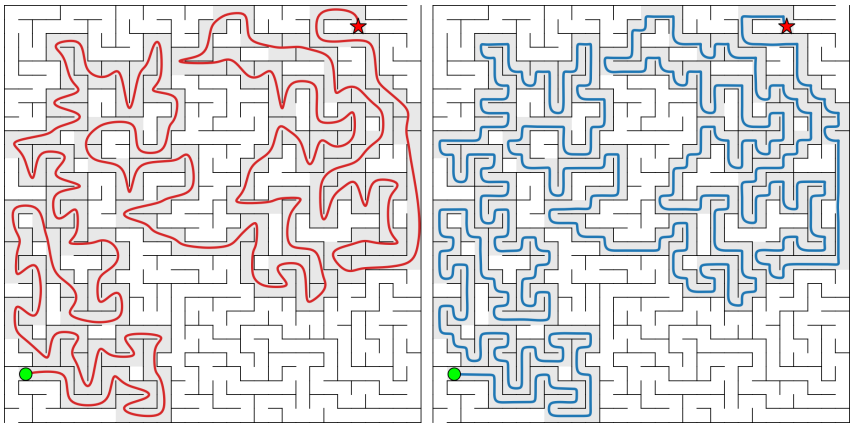

Figure 5: **Maze.** Same maze, start (green dot) and goal (red star): GCS (Marcucci et al., 2023) (left, Drake (Tedrake & the Drake Development Team, 2019) with convex relaxation and rounding; it also chooses the cell path, so its route differs) and ASPL (right) on the breadth-first-search cell path, inside whose corridor (gray) it stays.

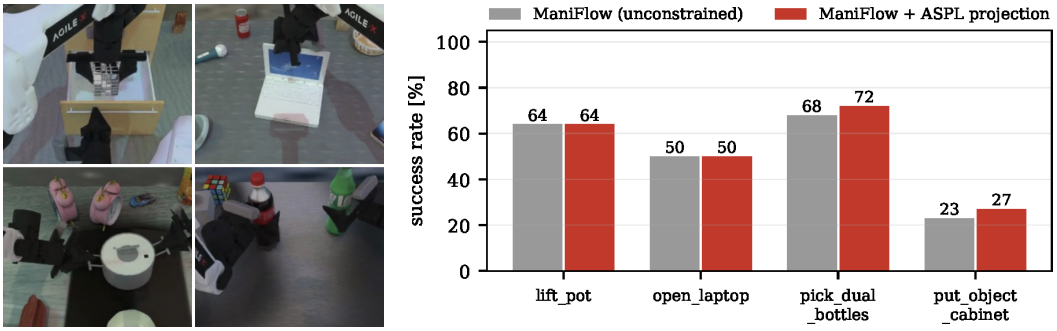

Figure 6: **Flow matching on RoboTwin 2.0.** Left: the four tasks (`put_object_cabinet`, `open_laptop`; `lift_pot`, `pick_dual_bottles`). Right: success over 100 episodes per task without (gray) and with (red) the ASPL projection; it never lowers success.

mazes the network is about as accurate as on the training maze, at nearly the same per-query cost of tens of milliseconds, so one network amortizes the fixed-path problem across mazes.

4.5 Constrained flow matching

ManiFlow (Yan et al., 2025) is a flow-matching policy that maps a point cloud to a chunk of joint commands for the twin-arm Aloha-Agilex robot. Nothing in its training enforces the joint limits or a smooth hand-over between consecutive chunks. We keep the trained policy and apply the ASPL layer at inference to every chunk, so this test uses the forward pass only. It projects the chunk onto joint-position, velocity, and acceleration bounds, an equality with the current joint state, and the real-time-chunking prefix equality of Black et al. (2025), so every executed chunk is feasible up to the solver tolerance. We evaluate on four RoboTwin 2.0 bimanual tasks (Chen et al., 2026), on the same 100 fixed episodes with and without the layer. The projection lowers success on no task (Figure 6). It adds a few successes on the two hardest tasks, a difference within episode-level variability for one training and one evaluation seed.

5 Conclusion

ASPL differentiates the projection map through its active manifold: one reduced solve per backward pass, for any forward solver, with finite-time identification and an $\mathcal{O}(1/T)$ training rate. Its backward is cheaper than ambient implicit differentiation of equal accuracy; the savings shrink as the active set approaches the full dimension or more directions are curved, and in motion planning its accuracy lead over Πnet narrows as more constraints bind. Its guarantees are local and assume strict complementarity; degenerate solutions are future work.

AI USE STATEMENT

We used large language models (LLMs) for three purposes. (i) *Writing*: to edit and polish author-written text for grammar, clarity, and concision, and to format the manuscript and references. (ii) *Retrieval and discovery*: to search for related literature; every suggested reference was checked against the original source before it was cited. (iii) *Synthetic datasets*: LLM coding assistants helped write the scripts that generate the synthetic benchmark instances (e.g., the parametric quadratic programs of Section 4, whose optima are constructed from the KKT conditions); the authors specified the problem families and generation procedure and inspected, tested, and validated the generated data. LLMs were not used for research ideation, for proving the mathematical claims, or to draft sections of the paper. The authors take full responsibility for all content, including any text, references, or data produced with LLM assistance.

REPRODUCIBILITY STATEMENT

Complete proofs of Theorems 1–4 and of the propositions are given in Appendix A3, with the notation and variational-analysis preliminaries collected in Appendix A1.1 and the pooled warm-start update in Appendix A3.5. The constraint sets used in the experiments, their closed-form block projections, and the forward solvers are specified in Appendix A2.1 and A2.2. Experimental protocols, hyperparameters, and baseline configurations for all five experiments are given in Appendices A4 and A4.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *ICML*, 2017.

Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. In *NeurIPS*, 2019.

Shreenabh Agrawal, Manan Tayal, Aditya Singh, and Shishir Kolathaya. Neural control barrier functions from physics informed neural networks. *arXiv preprint arXiv:2504.11045*, 2025.

Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *ECC*, pp. 3420–3431, 2019. doi: 10.23919/ECC.2019.8796030.

Barrett Ames, Jeremy Morgan, and George Konidaris. IKFlow: Generating diverse inverse kinematics solutions. *IEEE Robotics and Automation Letters*, 7(3):7177–7184, 2022.

Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *ICML*, 2017.

Antoine Bambade, Sarah El-Kazdadi, Adrien Taylor, and Justin Carpentier. PROX-QP: Yet another quadratic programming solver for robotics and beyond. In *RSS*, 2022.

Gilles Bareilles, Franck Iutzeler, and Jérôme Malick. Newton acceleration on manifolds identified by proximal gradient methods. *Mathematical Programming*, 200(1):37–70, 2023. doi: 10.1007/s10107-022-01873-w.

Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, second edition, 2017.

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-Flow: Differentiating through flows for controlled generation. In *ICML*, volume 235 of *Proceedings of Machine Learning Research*, pp. 3462–3483, 2024.

Quentin Bertrand, Quentin Klopfenstein, Mathurin Massias, Mathieu Blondel, Samuel Vaiter, Alexandre Gramfort, and Joseph Salmon. Implicit differentiation for fast hyperparameter selection in non-smooth convex learning. *Journal of Machine Learning Research*, 23(149):1–43, 2022.

Dimitri P. Bertsekas. Projected Newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2):221–246, 1982.

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *NeurIPS*, 2025.

Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. In *NeurIPS*, 2022.

James P. Boyle and Richard L. Dykstra. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in Order Restricted Statistical Inference*, Lecture Notes in Statistics, pp. 28–47. Springer, New York, 1986. doi: 10.1007/978-1-4613-9940-7_3.

James V. Burke and Jorge J. Moré. On the identification of active constraints. *SIAM Journal on Numerical Analysis*, 25(5):1197–1211, 1988.

Stéphane Caron, Yann De Mont-Marin, Rohan Budhiraja, and Seung Hyeon Bang. Pink: Python inverse kinematics based on Pinocchio. https://github.com/stephane-caron/pink, 2024. Software.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, Yao Mu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *ICML*, 2026.

Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. In *ICML Workshop on RL for Real Life*, 2019.

Xiaobing Dai, Zewen Yang, Dian Yu, Fangzhou Liu, Hamid Sadeghian, Sami Haddadin, and Sandra Hirche. SafeFlow: Safe robot motion planning with flow matching via control barrier functions. *arXiv preprint arXiv:2504.08661*, 2025.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

Aris Daniilidis, Dmitriy Drusvyatskiy, and Adrian S. Lewis. Orthogonal invariance and identifiability. *SIAM Journal on Matrix Analysis and Applications*, 35(2):580–598, 2014.

Damek Davis and Wotao Yin. A three-operator splitting scheme and its optimization applications. *Set-Valued and Variational Analysis*, 25(4):829–858, 2017.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Mateo Díaz, Pedro Izquierdo Lehmann, Haihao Lu, and Jinwen Yang. Active set identification and rapid convergence for degenerate primal-dual problems. *arXiv preprint arXiv:2602.10436*, 2026.

Priya L. Donti, David Rolnick, and J. Zico Kolter. DC3: A learning method for optimization with hard constraints. In *ICLR*, 2021.

Dmitriy Drusvyatskiy and Adrian S. Lewis. Optimality, identifiability, and sensitivity. *Mathematical Programming*, 147:467–498, 2014.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *ICML*, pp. 272–279, 2008. doi: 10.1145/1390156.1390191.

Ferdinando Fioretto, Pascal Van Hentenryck, Terrence W. K. Mak, Cuong Tran, Federico Baldo, and Michele Lombardi. Lagrangian duality for constrained deep learning. In *ECML PKDD*, Lecture Notes in Computer Science, pp. 118–135. Springer, 2021. doi: 10.1007/978-3-030-67670-4_8.

Fabrizio Flacco, Alessandro De Luca, and Oussama Khatib. Control of redundant robots under hard joint constraints: Saturation in the null space. *IEEE Transactions on Robotics*, 31(3):637–654, 2015.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *ICLR*, 2025.

Donald Goldfarb and Ashok Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27(1):1–33, 1983.

Paul J. Goulart and Yuwen Chen. Clarabel: An interior-point solver for conic programs with quadratic objectives. *Mathematical Programming Computation*, 2026. doi: 10.1007/s12532-026-00320-7. Code: https://github.com/oxfordcontrol/Clarabel.rs.

Panagiotis D. Grontas, Antonio Terpin, Efe C. Balta, Raffaello D'Andrea, and John Lygeros. Πnet: Optimizing hard-constrained neural networks with orthogonal projection layers. In *ICLR*, 2026.

William W. Hager and Hongchao Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization*, 16(1):170–192, 2005. doi: 10.1137/030601880.

William W. Hager and Hongchao Zhang. A new active set algorithm for box constrained optimization. *SIAM Journal on Optimization*, 17(2):526–557, 2006.

William W. Hager and Hongchao Zhang. The limited memory conjugate gradient method. *SIAM Journal on Optimization*, 23(4):2150–2168, 2013.

William W. Hager and Hongchao Zhang. Projection onto a polyhedron that exploits sparsity. *SIAM Journal on Optimization*, 26(3):1773–1798, 2016.

Warren L. Hare and Adrian S. Lewis. Identifying active constraints via partial smoothness and prox-regularity. *Journal of Convex Analysis*, 11(2):251–266, 2004. URL https://journalofconvexanalysis.com/articles/jca11017/.

Franck Iutzeler and Jérôme Malick. Nonsmoothness in machine learning: specific structure, proximal identification, and applications. *Set-Valued and Variational Analysis*, 28:661–678, 2020.

Kazumi Kasaura, Shuwa Miura, Tadashi Kozuno, Ryo Yonetani, Kenta Hoshino, and Yohei Hosoe. Benchmarking actor-critic deep reinforcement learning algorithms for robotics control with action constraints. *IEEE Robotics and Automation Letters*, 8(8):4449–4456, 2023. doi: 10.1109/LRA.2023.3284378.

Visak Kumar, David Hoeller, Balakumar Sundaralingam, Jonathan Tremblay, and Stan Birchfield. Joint space control via deep reinforcement learning. In *IROS*, pp. 3619–3626, 2021. doi: 10.1109/IROS51168.2021.9636477.

Sangkyun Lee and Stephen J. Wright. Manifold identification in dual averaging for regularized stochastic online learning. *Journal of Machine Learning Research*, 13:1705–1744, 2012.

Adrian S. Lewis. Active sets, nonsmoothness, and sensitivity. *SIAM Journal on Optimization*, 13 (3):702–725, 2002.

Jingwei Liang, Jalal Fadili, and Gabriel Peyré. Activity identification and local linear convergence of forward–backward-type methods. *SIAM Journal on Optimization*, 27(1):408–437, 2017.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

Tobia Marcucci, Mark Petersen, David von Wrangel, and Russ Tedrake. Motion planning around obstacles with convex optimization. *Science Robotics*, 8(84), 2023. doi: 10.1126/scirobotics.adf7843.

Pablo Márquez-Neila, Mathieu Salzmann, and Pascal Fua. Imposing hard constraints on deep networks: Promises and limitations. *arXiv preprint arXiv:1706.02025*, 2017.

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

Geoffrey Négiar, Michael W. Mahoney, and Aditi S. Krishnapriyan. Learning differentiable solvers for systems with hard constraints. In *ICLR*, 2023.

Vlad Niculae, André F. T. Martins, Mathieu Blondel, and Claire Cardie. SparseMAP: Differentiable sparse structured inference. In *ICML*, 2018.

Dantong Niu, Zhuoyang Liu, Zekai Wang, Boning Shao, Zhao-Heng Yin, et al. T-Rex: Tactile-reactive dexterous manipulation. *arXiv preprint arXiv:2606.17055*, 2026.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, second edition, 2006.

Julian Nubert, Johannes Köhler, Vincent Berenz, Frank Allgöwer, and Sebastian Trimpe. Safe and fast tracking on a robot manipulator: Robust MPC and neural network control. *IEEE Robotics and Automation Letters*, 5(2):3050–3057, 2020. doi: 10.1109/LRA.2020.2975727.

Julie Nutini, Mark Schmidt, and Warren Hare. "Active-set complexity" of proximal gradient: How long does it take to find the sparsity pattern? *Optimization Letters*, 13(4):645–655, 2019.

Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. In *Journal of Optimization Theory and Applications*, 2016.

Xiang Pan, Tianyu Zhao, Minghua Chen, and Shengyu Zhang. DeepOPF: A deep neural network approach for security-constrained DC optimal power flow. *IEEE Transactions on Power Systems*, 36(3):1725–1735, 2021.

Seonho Park and Pascal Van Hentenryck. Self-supervised primal-dual learning for constrained optimization. In *AAAI*, 2023.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

Guy Pierra. Decomposition through formalization in a product space. *Mathematical Programming*, 28(1):96–115, 1984.

René A. Poliquin, R. Tyrrell Rockafellar, and Lionel Thibault. Local differentiability of distance functions. *Transactions of the American Mathematical Society*, 352(11):5231–5249, 2000.

Clarice Poon, Jingwei Liang, and Carola-Bibiane Schönlieb. Local convergence properties of SAGA/Prox-SVRG and acceleration. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4124–4132, 2018.

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. doi: 10.1016/j.jcp.2018.10.045.

Ahmed Rashwan, Keith Briggs, Chris Budd, and Lisa Maria Kreusser. Enforcing convex constraints in graph neural networks. In *NeurIPS*, 2025.

Moritz Reuss, Ömer Erdinç Yağmurlu, Fabian Wenzel, and Rudolf Lioutikov. Multimodal diffusion transformer: Learning versatile behavior from multimodal goals. In *RSS*, 2024.

R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

R. Tyrrell Rockafellar and Roger J.-B. Wets. *Variational Analysis*, volume 317 of *Grundlehren der Mathematischen Wissenschaften*. Springer, 1998.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, second edition, 2003.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *ICML*, 2023.

Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. OSQP: An operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. doi: 10.1007/s12532-020-00179-2.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by PID Lagrangian methods. In *ICML*, 2020.

Russ Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics. https://drake.mit.edu, 2019.

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

Unitree Robotics. UnifoLM-WBT-Dataset: Unitree G1 whole-body teleoperation datasets (LeRobot format). https://huggingface.co/collections/unitreerobotics/unifolm-wbt-dataset, 2026. Accessed 2026.

Utkarsh Utkarsh, Pengfei Cai, Alan Edelman, Rafael Gomez-Bombarelli, and Christopher Vincent Rackauckas. Physics-constrained flow matching: Sampling generative models with hard constraints. In *NeurIPS*, 2025.

Po-Wei Wang, Priya L. Donti, Bryan Wilder, and J. Zico Kolter. SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *ICML*, 2019.

Stephen J. Wright. Identifiable surfaces in constrained optimization. *SIAM Journal on Control and Optimization*, 31(4):1063–1079, 1993.

Ge Yan, Jiyue Zhu, Yuquan Deng, Shiqi Yang, Ri-Zhao Qiu, Xuxin Cheng, Marius Memmel, Ranjay Krishna, Ankit Goyal, Xiaolong Wang, and Dieter Fox. ManiFlow: A general robot manipulation policy via consistency flow training. In *CoRL*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2268–2293, 2025.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D diffusion policy: Generalizable visuomotor policy learning via simple 3D representations. In *RSS*, 2024.

Appendix

CONTENTS

A1 Addendum to Section 1: notation and related work . . . 15
A1.1 Notation and variational-analysis preliminaries . . . 15
A1.2 Extended related work . . . 16
A2 Addendum to Section 2: constraint sets, forward solvers, and training . . . 18
A2.1 Constraint sets and their projections . . . 18
A2.2 Forward solvers . . . 19
A2.3 Backward-pass cost of differentiable projection layers . . . 20
A2.4 Formal assumptions and the block-triangular form . . . 20
A2.5 Training procedure and the stochastic variant . . . 21
A3 Addendum to Section 3: stochastic variant and proofs . . . 23
A3.1 End-to-end convergence . . . 23
A3.2 Proof of Theorem 1 and corollaries . . . 24
A3.3 Proof of Theorem 2 (Finite-time identification) . . . 27
A3.4 Proofs of the propositions on the face-restricted forward . . . 28
A3.5 Pooled-EMA update . . . 29
A3.6 Proof of Theorem 3 . . . 29
A3.7 Proof of Theorem 4 . . . 30
A4 Addendum to Section 4: experimental details and further results . . . 31
A4.1 Parametric quadratic programming . . . 31
A4.2 Second-order cone programs with many cones . . . 34
A4.3 Multi-robot motion planning . . . 34
A4.4 Inverse kinematics on real teleoperation logs . . . 35
A4.5 Maze trajectory planning . . . 36
A4.6 Constrained flow matching . . . 37

A1 ADDENDUM TO SECTION 1: NOTATION AND RELATED WORK

A1.1 NOTATION AND VARIATIONAL-ANALYSIS PRELIMINARIES

Bold lower-case letters $(\mathbf{u}, \mathbf{y})$ are vectors, bold upper-case letters $(\mathbf{A}, \mathbf{P})$ are matrices, and calligraphic letters $(\mathcal{C}, \mathcal{A})$ are sets. $\|\cdot\|$ is the Euclidean norm, $\mathbf{1}_{\mathcal{I}}$ the indicator vector of an index set $\mathcal{I}$, and $\ker \mathbf{A}$ the kernel of $\mathbf{A}$. For closed $\mathcal{C} \subseteq \mathbb{R}^d$, $\Pi_{\mathcal{C}}(\mathbf{u}) \triangleq \arg\min_{\mathbf{z}\in\mathcal{C}} \frac{1}{2}\|\mathbf{z}-\mathbf{u}\|^2$ is the Euclidean projection, a singleton for convex $\mathcal{C}$ and for $\mathbf{u}$ within the reach of a prox-regular $\mathcal{C}$, and $\operatorname{dist}(\mathbf{u}, \mathcal{C}) \triangleq \min_{\mathbf{z}\in\mathcal{C}} \|\mathbf{u}-\mathbf{z}\|$. For a set $\mathcal{S}$, $\operatorname{aff}\mathcal{S}$ is its affine hull, $\operatorname{relint}\mathcal{S}$ (also $\operatorname{ri}\mathcal{S}$) its interior relative to $\operatorname{aff}\mathcal{S}$, and $\operatorname{rbd}\mathcal{S} \triangleq \overline{\mathcal{S}} \setminus \operatorname{relint}\mathcal{S}$ its relative boundary.

Definition 2 (Tangent and normal cones)**.** *For closed convex $\mathcal{C}$ and $\mathbf{y} \in \mathcal{C}$, $T_{\mathcal{C}}(\mathbf{y})$ is the closure of $\{\gamma(\mathbf{z}-\mathbf{y}) : \gamma \geq 0,\ \mathbf{z} \in \mathcal{C}\}$ and $N_{\mathcal{C}}(\mathbf{y}) \triangleq \{\mathbf{n} : \langle \mathbf{n}, \mathbf{z}-\mathbf{y}\rangle \leq 0\ \forall \mathbf{z} \in \mathcal{C}\}$ is its polar. For $\mathcal{C}$ prox-regular at $\mathbf{y}$, $\mathbf{v} \in N^P_{\mathcal{C}}(\mathbf{y})$ (a proximal normal) if $\langle \mathbf{v}, \mathbf{z}-\mathbf{y}\rangle \leq \sigma\|\mathbf{z}-\mathbf{y}\|^2$ for some $\sigma \geq 0$ and*

all $\mathbf{z} \in \mathcal{C}$ *near* $\mathbf{y}$*. The proximal, limiting, and Clarke normal cones then coincide near* $\mathbf{y}$*, and* $N_{\mathcal{C}}$ *denotes this common closed convex cone.*

If $\mathcal{C}$ is a smooth manifold near $\mathbf{y}$, $T_{\mathcal{C}}(\mathbf{y})$ and $N_{\mathcal{C}}(\mathbf{y})$ are its tangent and normal spaces. We write $T_{\mathcal{C}_{\mathcal{A}}}$ and $N_{\mathcal{C}_{\mathcal{A}}}$ for them at $\overline{\mathbf{y}}$.

Lewis (2002) call a closed set $\mathcal{C}$ *partly smooth at* $\mathbf{y}$ *relative to* a C^2 manifold $\mathcal{M} \ni \mathbf{y}$, $\mathcal{M} \subseteq \mathcal{C}$, if $\mathcal{C}$ is Clarke regular at the points of $\mathcal{M}$ near $\mathbf{y}$ and the normal-space and continuity conditions of Definition 1 hold. Clarke regularity is automatic for a set prox-regular at $\mathbf{y}$ (Poliquin et al., 2000), so Definition 1 is this notion, and for prox-regular sets the manifold is locally unique (Hare & Lewis, 2004).

Definition 3 (Friedrichs angle)**.** *For subspaces* $\mathcal{V}, \mathcal{W} \subseteq \mathbb{R}^d$*,* $\cos\theta_F(\mathcal{V}, \mathcal{W}) \triangleq \sup\{\langle \mathbf{u}, \mathbf{w}\rangle : \mathbf{u} \in \mathcal{V} \cap (\mathcal{V} \cap \mathcal{W})^{\perp},\ \mathbf{w} \in \mathcal{W} \cap (\mathcal{V} \cap \mathcal{W})^{\perp},\ \|\mathbf{u}\| = \|\mathbf{w}\| = 1\}$.

In the proofs, for polyhedral $\mathcal{C}_2 = \{\mathbf{l} \le \mathbf{z} \le \mathbf{u}\}$ at $\overline{\mathbf{y}}$, $\mathcal{J} \triangleq \{j : \overline{y}_j \in \{l_j, u_j\}\}$ is the bound-active set and $\mathcal{J}^c$ its complement, the free set. Their diagonal coordinate projectors are $\mathbf{N} = \operatorname{diag}(\mathbf{1}_{\mathcal{J}})$ and $\mathbf{F} = \mathbf{T} = \mathbf{I} - \mathbf{N}$, so $d_{\mathbf{N}} = |\mathcal{J}|$, and $\mathbf{X}_{\mathbf{UV}} \triangleq \mathbf{UXV}$. For a boundary-active cone in $\mathbb{R}^{n+1}$, $\mathbf{N} = \mathbf{I} - \mathbf{F}$ has rank n: the normal direction and the $n-1$ curved tangent directions. Mixed instances decompose block by block (Liang et al., 2017, Prop. 4.5).

A1.2 Extended related work

A1.2.1 Differentiable projection layers

OptNet (Amos & Kolter, 2017) introduces argmin-differentiation through quadratic-program (QP) layers via KKT linearization at $\mathcal{O}(d^3)$ per backward pass. cvxpylayers (Agrawal et al., 2019) generalizes this to arbitrary convex cone programs through the homogeneous self-dual embedding, sharing the dense-factor backward cost. Both methods solve the constraint exactly but pay $\mathcal{O}(d^3)$ per gradient step regardless of how few constraints are active — the active-manifold structure is invisible to them. DC3 (Donti et al., 2021) introduces a correction-completion approach: predict a partial output, then complete it to feasibility via a learned correction step. The cost is lower than full implicit differentiation but feasibility is approximate. RAYEN (Tordesillas et al., 2023) maps outputs into a fixed, input-independent convex set (linear, convex-quadratic, second-order-cone or LMI constraints) through a closed-form ray-based map, and HardNet (Min & Azizan, 2024) enforces input-dependent affine constraints through a closed-form correction; both are exact and fast but are restricted to these constraint classes and do not cover a general $\mathcal{C}(\mathbf{x})$.

ΠNet (Grontas et al., 2026) is the closest precedent and the primary baseline in our experiments. It solves the projection subproblem by Douglas–Rachford and computes implicit-differentiation gradients through a BiCGSTAB inner backward at cost $\mathcal{O}(\kappa \log(1/\varepsilon)\, d^2)$. ASPL identifies the active manifold (Theorem 2) and replaces this ambient backward with the reduced operator of Theorem 1, which is block lower-triangular on every active manifold and carries the curvature of SOC boundaries in its normal block. ProjNet (Rashwan et al., 2025) enforces input-dependent convex constraints in graph neural networks by Component-Averaged Dykstra projection with sparse vector clipping, and backpropagates a *surrogate gradient* of the iterative projection instead of the exact implicit gradient. This trades gradient correctness for GPU efficiency on the sparse outputs of GNNs. ASPL targets dense outputs whose feasible set has a low-codimension active manifold at the projected point, where the exact reduced gradient of Theorem 1 replaces both the ambient d-dimensional solve of ΠNet and OptNet and the surrogate gradient of ProjNet.

A1.2.2 Partial smoothness and active-set identification

The idea of identifying the active constraints and reducing the problem to its low-dimensional active block is one of the oldest tools in constrained optimization. Bertsekas's projected-Newton method (Bertsekas, 1982) is the prototype: at each step it freezes the bound-active coordinates and runs Newton on the free block, an approach later refined by Burke & Moré (1988), who gave the first general identification result for active inequality constraints, and by Wright (1993), who isolated *identifiable surfaces* as the right geometric object. The textbook treatment by Nocedal & Wright (2006, Ch. 16–17) consolidates the active-set / interior-point dichotomy. Hager and Zhang's series of bound-constrained algorithms develops this line with a focus on practical performance: the

active-set algorithm for box-constrained optimization of Hager & Zhang (2006), which uses their conjugate-gradient method with guaranteed descent (Hager & Zhang, 2005) on the free block, and their later limited-memory conjugate-gradient and sparse polyhedral-projection algorithms (Hager & Zhang, 2013; 2016). More recently, Nutini et al. (2019) quantified the *active-set complexity* of proximal-gradient methods on partly-smooth objectives, and Díaz et al. (2026) gave nonasymptotic guarantees that primal-dual methods (proximal point, PDHG, ADMM, extragradient) identify the active constraints of convex problems and then converge linearly, without assuming strict complementarity.

The variational-analysis tradition that underpins our identification result begins with Lewis (2002), who introduced *partial smoothness* as a unifying lens for active-set behavior in nonsmooth optimization. Hare & Lewis (2004) established finite-time identification of the active manifold for proximal maps under partial smoothness and strict complementarity — the result we apply (their Theorem 5.3) in the proof of Theorem 2. Daniilidis et al. (2014); Drusvyatskiy & Lewis (2014) broadened the framework to spectral and orthogonally invariant functions and connected it to second-order optimality. Liang et al. (2017) specialized the identification analysis to first-order operator-splitting methods, including Douglas–Rachford and forward–backward; their Proposition 4.5 (intersection of partly smooth sets) is central to our proofs. Parallel work by Lee & Wright (2012) on dual averaging, by Poon et al. (2018) on local linear convergence of forward–backward, by Iutzeler & Malick (2020) on exploiting proximal identification in machine learning (compression, acceleration, dimension reduction), and by Bareilles et al. (2023) on Newton-type acceleration after identification extends this analysis to other proximal-splitting settings.

The classical active-set and partial-smoothness literature solves one optimization problem, identifies its active manifold once, and then runs a faster local method on it. ASPL uses the same geometry inside a differentiable layer, in three ways.

First, existing differentiable projection layers (OptNet (Amos & Kolter, 2017), cvxpylayers (Agrawal et al., 2019), IInet (Grontas et al., 2026)) solve a d-dimensional system on every backward pass and never identify which constraints are tight. ASPL reads the active manifold off the forward solver (Theorem 2) and differentiates only along its flat and normal blocks, so $d_{\mathbf{N}}$ replaces d as the parameter that controls the backward cost (Appendix A2.3).

Second, the identification literature asks *which* constraints are active; Theorem 1 concerns the algebra of differentiating the projection once they are known. $\mathbf{K}_t$ is block lower-triangular on every active manifold; on flat polyhedral ones its normal block is $\mathbf{P}_{\mathbf{NN}} \succ 0$ and the backward reduces to one $d_{\mathbf{N}} \times d_{\mathbf{N}}$ Cholesky factorization (Corollary 3). On curved active manifolds (SOC boundaries), the curved tangent directions join the dense block, the curvature enters only through $\mathbf{Q}_{\mathbf{NN}}$, and the leading block stays scalar (Corollary 2).

Third, classical identification concerns one problem; during training it must hold across the data distribution while parameter updates move the active set. The pooled exponential-moving-average warm start needs $\Theta(d)$ memory instead of the $\Theta(Nd)$ of per-sample warm starts, and the switching rule (14) returns the forward to the full solver when the primal test fails, while the backward stays reduced. Theorem 3 gives the $\mathcal{O}(1/T)$ rate of full-gradient descent, and Theorem 4 shows that, for SGD with exact backward solves, $K_{\mathrm{dr}} = \mathcal{O}(\delta^{-1} \log T)$ forward steps from the pooled warm start keep the $\mathcal{O}(T^{-1/2})$ stationarity rate and, under a uniform identification radius, place every forward output on the correct active manifolds once T is large enough.

A1.2.3 SOFT-CONSTRAINED ALTERNATIVES

A large body of work (Raissi et al., 2019; Fioretto et al., 2021; Park & Van Hentenryck, 2023; Achiam et al., 2017; Chow et al., 2019; Stooke et al., 2020) replaces hard constraints with soft penalties. The methods are general and easy to implement but introduce a non-vanishing bias unless the penalty weight is tuned per instance, and they offer no feasibility guarantee at deployment. Lagrangian-dual methods (Fioretto et al., 2021; Park & Van Hentenryck, 2023) adapt the penalty during training; CPO (Achiam et al., 2017) adapts within a trust-region framework, and Lyapunov-constrained policies (Chow et al., 2019) project policy updates or actions onto linearized Lyapunov constraints, and Stooke et al. (2020) update the multiplier with a PID controller. None of these methods can guarantee zero violation at any single deployment query; conversely, Márquez-Neila

et al. (2017) found that imposing hard constraints through a generic Lagrangian solver brings little benefit over such penalties, which motivates making the hard-constraint path both exact and cheap.

ASPL is hard-constrained by design: every output, including those produced at intermediate training epochs, satisfies $\mathcal{C}(\mathbf{x})$ up to the forward solver's tolerance. The price is the K_{dr}-iteration forward DR subproblem and a reduced active-manifold backward; for flat polyhedral faces this is the $\mathcal{O}(d_{\mathbf{N}}^3)$ normal-block solve.

A1.2.4 Inference-time projection and constrained generative modeling

Building on flow matching (Lipman et al., 2023), a recent line of work (Utkarsh et al., 2025; Dai et al., 2025; Ben-Hamu et al., 2024) enforces constraints on generated outputs at inference time. PCFM (Utkarsh et al., 2025) introduces physics-constrained flow matching by adding a projection step at sampling time, but the velocity field is trained agnostic to feasibility, which can produce out-of-distribution terminal states that inflate inference-time projection cost; D-Flow (Ben-Hamu et al., 2024) uses optimization through the flow-matching ODE for constrained sampling; ManiFlow (Yan et al., 2025) adapts flow matching to robotic manipulation without enforcing hard constraints on the generated actions. ASPL-FM (Appendix A4.6) applies the projection layer to the terminal ODE state of a pretrained ManiFlow policy, so every executed chunk is feasible up to the solver tolerance.

A2 Addendum to Section 2: constraint sets, forward solvers, and training

A2.1 Constraint sets and their projections

This appendix lists the constraint families covered by ASPL, their two-set decomposition $\mathcal{C}(\mathbf{x}) = \mathcal{C}_1(\mathbf{x}) \cap \mathcal{C}_2(\mathbf{x})$ and per-set projections, and the forward solvers we use. The reduced backward depends only on the projection map and its active face, not on how the forward computes them (Proposition 1), so the forward solver is a free choice. Appendix A2.2 lists the solvers we use.

Example 1 (Box constraint)**.** *Take the instance-dependent box* $\mathcal{C}(\mathbf{x}) = \{\mathbf{z} \in \mathbb{R}^d : \mathbf{l}(\mathbf{x}) \le \mathbf{z} \le \mathbf{u}(\mathbf{x})\}$*, and let* $\mathbf{y} = \Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}})$*. Split the coordinates by whether they hit a bound:*

- ***Normal*** *(blocked):* i *with* $y_i = u_i(\mathbf{x})$ *or* $y_i = l_i(\mathbf{x})$*. The active manifold pins* $z_i = y_i$*; the projector* $\mathbf{N}$ *extracts these coordinates.*
- ***Tangent*** *(free):* j *with* $l_j(\mathbf{x}) < y_j < u_j(\mathbf{x})$*. The active manifold leaves* z_j *free; the projector* $\mathbf{T}$ *extracts these coordinates and* $\Pi_{\mathcal{C}_2}$ *is the identity along them.*

Then, near $\mathbf{y}$*,* $\mathcal{C}_{\mathcal{A}}(\mathbf{x}) = \{\mathbf{z} : z_i = y_i \ \forall\, i \text{ blocked}\}$*, and* $d_{\mathbf{N}}$ *is the number of blocked coordinates — typically far smaller than* d*.*

Polyhedra in the box-bounded affine form $\mathcal{C} = \{\mathbf{z} : \mathbf{A}\mathbf{z} = \mathbf{b}, \mathbf{l} \le \mathbf{z} \le \mathbf{u}\}$ cover linear MPC for legged-locomotion and manipulation, AC–DC dispatch in OPF, and resource allocation. After lifting general linear inequalities to slack form, every linear-inequality system fits this template. *Split:* $\mathcal{C}_1 = \{\mathbf{A}\mathbf{z} = \mathbf{b}\}$ (affine subspace), $\mathcal{C}_2 = [\mathbf{l}, \mathbf{u}]$ (box). *Projections:* $\Pi_{\mathcal{C}_1}(\mathbf{z}) = \mathbf{z} - \mathbf{A}^{+}(\mathbf{A}\mathbf{z} - \mathbf{b})$ (cached pseudoinverse, one matvec per call); $\Pi_{\mathcal{C}_2}(\mathbf{z}) = \mathrm{clip}(\mathbf{z}; \mathbf{l}, \mathbf{u})$ (element-wise). Second-order cones model friction cones and torque-magnitude constraints in manipulation, beam-pattern constraints in beamforming, robust counterparts of LPs under ellipsoidal uncertainty, and norm budgets in regularized supervised learning. The Lorentz cone $\mathcal{K}_{\mathrm{soc}} = \{(\mathbf{w}, t) : \|\mathbf{w}\|_2 \le t\}$ is the prototype. *Split:* $\mathcal{C}_1$ collects the affine equalities, $\mathcal{C}_2 = \mathcal{K}_{\mathrm{soc}}$ (or its Cartesian product over multiple cones). *Projection onto* $\mathcal{K}_{\mathrm{soc}}$*:* the closed-form rule of Bauschke & Combettes (2017) — if $\|\mathbf{w}\|_2 \le t$ return $(\mathbf{w}, t)$; if $\|\mathbf{w}\|_2 \le -t$ return $(\mathbf{0}, 0)$; else return $\frac{t+\|\mathbf{w}\|_2}{2\|\mathbf{w}\|_2}(\mathbf{w}, \|\mathbf{w}\|_2)$ — one norm and one rescale per cone.

Norm balls $\{\mathbf{z} : \|\mathbf{z} - \mathbf{c}\|_p \le r\}$ appear as control-input limits ($p = \infty$ box, $p = 2$ Euclidean), trust regions in adaptive control, and certified-robust output regions. *Split:* $\mathcal{C}_1$ is the (optional) ambient affine constraint, $\mathcal{C}_2$ the ball. *Projection:* closed form for $p \in \{1, 2, \infty\}$ — coordinate clip ($p = \infty$), radial rescale ($p = 2$), and the $\mathcal{O}(d \log d)$ soft-threshold rule of Duchi et al. (2008) ($p = 1$). Simplices and their products (the canonical multi-class output constraint) appear in policy networks, assignment problems, and categorical generative priors. *Split:* $\mathcal{C}_1 = \{\mathbf{1}^\top \mathbf{z} = 1\}$, $\mathcal{C}_2 = \{\mathbf{z} \ge \mathbf{0}\}$ (or

directly the simplex via the $\mathcal{O}(d \log d)$ sort-then-clip projector of Duchi et al. (2008)). Physical-AI deployments are typically intersections, e.g., joint-limit (box) $\cap$ friction cone (SOC) $\cap$ workspace half-spaces. Stacking the affine parts in $\mathcal{C}_1$ keeps its projection closed-form; the structured parts form a Cartesian product $\mathcal{C}_2$ when they act on disjoint coordinates, and otherwise the product-space construction of Pierra (1984) reduces them to two sets, and Theorem 1 extends block-by-block via Corollaries 2 and 3 on the direct-sum structure of partly smooth set products (Liang et al., 2017, Prop. 4.5).

A2.2 Forward solvers

The reduced backward of Theorem 1 needs only the projection $\Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}})$ and its active face, not the procedure that computes them (Proposition 1). Any exact projection method can therefore serve as the forward: an operator-splitting or dual method, an active-set method, or an interior-point method. The library provides the following solvers, and each experiment uses the one that fits its constraint set.

(i) Dual identification and a face solve for polytopes $\{\mathbf{z} : \mathbf{A}\mathbf{z} = \mathbf{b},\ \mathbf{G}\mathbf{z} \le \mathbf{h}\}$. Phase 1 runs projected gradient ascent on the dual $\max_{\boldsymbol{\lambda},\,\boldsymbol{\mu}\ge\mathbf{0}} -\frac{1}{2}\|\mathbf{A}^\top\boldsymbol{\lambda} + \mathbf{G}^\top\boldsymbol{\mu}\|^2 + (\boldsymbol{\lambda},\boldsymbol{\mu})^\top(\mathbf{A}\mathbf{y}_{\text{raw}} - \mathbf{b},\ \mathbf{G}\mathbf{y}_{\text{raw}} - \mathbf{h})$ with Barzilai–Borwein or FISTA steps, and the rows $\mathcal{I}$ with $\hat{\mu}_i > 0$ give the active face. Phase 2 projects exactly onto that face: with $\mathbf{A}_\mathcal{I} = [\mathbf{A}; \mathbf{G}_\mathcal{I}]$, it solves $\mathbf{A}_\mathcal{I}\mathbf{A}_\mathcal{I}^\top\boldsymbol{\nu} = \mathbf{A}_\mathcal{I}\mathbf{y}_{\text{raw}} - (\mathbf{b}; \mathbf{h}_\mathcal{I})$ and returns $\mathbf{y} = \mathbf{y}_{\text{raw}} - \mathbf{A}_\mathcal{I}^\top\boldsymbol{\nu}$, adding violated rows and dropping rows with negative multipliers until the KKT conditions hold; it is skipped when Phase 1 already meets the feasibility tolerance. The backward is the orthogonal projector onto $\ker \mathbf{A}_\mathcal{I}$ (Corollary 3). We use this solver on the maze benchmark. Training runs Phase 1 only and reads the face off its final multipliers. Phase 2 runs at evaluation.

(ii) Exact and inexact face solves. The Phase-2 system is solved either exactly, by a Cholesky factor of $\mathbf{A}\mathbf{A}^\top$ cached once plus a Cholesky of the $|\mathcal{I}| \times |\mathcal{I}|$ Schur complement, or inexactly, by Jacobi-preconditioned CG warm-started from the Phase-1 multipliers. The reduced backward offers the same two options, reported as ASPL (Chol) and ASPL (CG).

(iii) Conic dual ascent for $\{\mathbf{y} : \mathbf{b} - \mathbf{A}\mathbf{y} \in \mathcal{K}\}$ with $\mathcal{K}$ a product of Lorentz cones. It runs projected ascent on the self-dual multiplier, $\mathbf{s} \leftarrow \Pi_\mathcal{K}\big(\mathbf{s} + \alpha(\mathbf{A}\mathbf{y} - \mathbf{b})\big)$ with $\mathbf{y} = \mathbf{y}_{\text{raw}} - \mathbf{A}^\top\mathbf{s}$ (the dual gradient); since $\mathbf{s}_i = \mathbf{0}$ on every cone inactive at the solution, the support of $\mathbf{s}$ identifies the active cones. The step uses FISTA or Barzilai–Borwein step sizes and the closed-form projection onto each Lorentz cone given above. The backward classifies each cone of $\mathcal{K}$ as interior (Jacobian $\mathbf{I}$), apex (Jacobian $\mathbf{0}$), or boundary. At a point $(\mathbf{w}, t)$ with $|t| < r = \|\mathbf{w}\|$, which projects to the cone boundary, the Jacobian of $\Pi_{\mathcal{K}_{\text{soc}}}$ is $\left[\begin{smallmatrix} \frac{t+r}{2r}\mathbf{I} - \frac{t}{2r^3}\mathbf{w}\mathbf{w}^\top & \frac{\mathbf{w}}{2r} \\ \frac{\mathbf{w}^\top}{2r} & \frac{1}{2} \end{smallmatrix}\right]$; its $\mathbf{w}\mathbf{w}^\top$ term carries the curvature of the cone (Corollary 2). The reduced backward keeps only the boundary cones, so its size is set by the boundary cones ($k-1$ directions for each boundary cone of dimension k), not by d. The SOCP benchmark (Appendix A4.2) uses this forward with the dense $m \times m$ adjoint built from the same per-cone Jacobians.

(iv) Douglas–Rachford, for any two sets with computable projections. With state $\mathbf{s}$ and relaxation $\omega \in (0, 2)$, one step alternates the two block projections,

$$\begin{aligned} \mathbf{z}^{(k+1)} &= \Pi_{\mathcal{C}_2}\big(\tfrac{1}{2}(\mathbf{s}^{(k)} + \mathbf{y}_{\text{raw}})\big), \qquad \mathbf{t}^{(k+1)} = \Pi_{\mathcal{C}_1}\big(2\mathbf{z}^{(k+1)} - \mathbf{s}^{(k)}\big), \\ \mathbf{s}^{(k+1)} &= \mathbf{s}^{(k)} + \omega\big(\mathbf{t}^{(k+1)} - \mathbf{z}^{(k+1)}\big). \end{aligned} \tag{8}$$

At a fixed point $\mathbf{s}^\star$, $\mathbf{z}^\star = \mathbf{t}^\star = \Pi_\mathcal{C}(\mathbf{y}_{\text{raw}})$, and for convex blocks the iteration converges from any start (Bauschke & Combettes, 2017, Prop. 28.13, Cor. 28.10). The proofs in Appendix A3 use it as the worked example of a fixed-point solver $\mathcal{T}$. We run $K_{\text{dr}} = 200$ steps, with $\omega = 1$ on the QP benchmarks and $\omega = 1.7$ elsewhere (Appendix A4). The active face is read off the bounds that the last $\Pi_{\mathcal{C}_2}$ call clips. The cached affine projector $\Pi_{\mathcal{C}_1}$ is shared across iterations and the minibatch (data dependence enters only through $\mathbf{b}_{\text{eq}}(\mathbf{x})$), and $\Pi_{\mathcal{C}_2}$ is element-wise or per-cone. We use DR on the QP, motion-planning, and flow-matching experiments (the ASPL-FM instance is in Appendix A4.6.3).

Table 2: ASPL's backward operates on the $d_{\mathbf{N}}$-dimensional active manifold rather than the ambient $\mathbb{R}^d$. Rate: the stationarity rate of Theorem 4 (SGD with $\eta \propto T^{-1/2}$, exact backward solve, locally contracting forward; with full gradients and $\eta = 1/L_\Phi$ it is $\mathcal{O}(1/T)$, Theorem 3); †for CG, when the backward is solved to exactness. Chol: Cholesky; CG: conjugate gradient. $d_{\mathbf{N}} \le d$: size of the normal block $\mathcal{F}^\perp$; $\kappa_{\mathbf{N}} \le \kappa$: block vs. ambient IFT condition numbers.

Method	Backward cost	Rate
OptNet (Amos & Kolter, 2017)	$\mathcal{O}(d^3)$	—
cvxpyL. (Agrawal et al., 2019)	solver-dep.	—
Πnet (BiCG) (Grontas et al., 2026)	$\mathcal{O}(\kappa \log \frac{1}{\epsilon}\, d^2)$	—
ASPL (Chol)	$\mathcal{O}(d_{\mathbf{N}}^3)$	$\mathcal{O}(T^{-1/2})$
ASPL (CG)	$\mathcal{O}(\sqrt{\kappa_{\mathbf{N}}} \log \frac{1}{\epsilon}\, d_{\mathbf{N}}^2)$	$\mathcal{O}(T^{-1/2})^\dagger$

(v) Splitting with an active-set polish for inverse kinematics: $2T$ splitting steps are followed by an active-set polish, and a chunk not certified feasible is re-identified by an interior-point pass (Appendix A4.4).

A2.3 Backward-pass cost of differentiable projection layers

Table 2 compares the per-sample backward cost of ASPL with existing differentiable projection layers.

A2.4 Formal assumptions and the block-triangular form

The split (3) is the normal-cone sum rule $N_{\mathcal{C}}(\overline{\mathbf{y}}) = N_{\mathcal{C}_1}(\overline{\mathbf{y}}) + N_{\mathcal{C}_2}(\overline{\mathbf{y}})$ (Bauschke & Combettes, 2017, Cor. 16.50), with each $\mathbf{n}_i$ a proximal normal realized by its own block projection (Poliquin et al., 2000). For convex $\mathcal{C}_2$ one may take $t = 1$; for prox-regular $\mathcal{C}_2$ with prox-regularity radius r_2 any $t < r_2/\|\mathbf{n}_2\|$ works (cardinality sets, obstacle complements). The transversality condition $N_{\mathcal{C}_1}(\overline{\mathbf{y}}) \cap (-N_{\mathcal{C}_2}(\overline{\mathbf{y}})) = \{\mathbf{0}\}$ guarantees the sum rule; the pair $(\mathbf{n}_1, \mathbf{n}_2)$ is unique under the stronger condition $\operatorname{span} N_{\mathcal{C}_1}(\overline{\mathbf{y}}) \cap \operatorname{span} N_{\mathcal{C}_2}(\overline{\mathbf{y}}) = \{\mathbf{0}\}$ of Assumption C. For nonconvex $\mathcal{C}_1(\mathbf{x})$, the bound on $\|\mathbf{n}_1\|$ in Assumption B keeps $\overline{\mathbf{y}} + \mathbf{n}_1$ inside the region where $\Pi_{\mathcal{C}_1}$ is single-valued and differentiable. Assumption D is simple on flat manifolds: when $\mathcal{C}_1$ is affine and $\mathcal{C}_{\mathcal{A},2}$ is locally affine, $\mathbf{K_{NN}} = \mathbf{P_{NN}}$, which is nonsingular iff the two active manifolds are transversal and satisfies $\lambda_{\min}(\mathbf{P_{NN}}) = \sin^2\theta_F$ for the Friedrichs angle θ_F between them, so the uniform condition-number bound is a uniform lower bound on θ_F.

Theorem 1 collapses to a projector when both active manifolds are flat.

Corollary 1 (Flat active manifolds)**.** *If $\mathcal{C}_1(\mathbf{x})$ is affine with tangent projector $\mathbf{P}$ and $\mathcal{C}_{\mathcal{A},2}$ is locally affine at $\overline{\mathbf{y}}$, then (taking $t = 1$; the Jacobian (5) does not depend on the admissible t) $\mathbf{Q} = \mathbf{F} = \mathbf{T}$, $\mathbf{K_{NN}} = \mathbf{P_{NN}}$, and (6) becomes*

$$\mathbf{P_{NN}}\,\mathbf{v_N} \;=\; (\mathbf{Pc})_{\mathbf{N}}, \qquad \frac{\partial \ell}{\partial \mathbf{y}_{\text{raw}}} \;=\; (\mathbf{Pc})_{\mathbf{T}} - \mathbf{P_{TN}}\,\mathbf{v_N} \;=\; \Pi_{T_{\mathcal{C}_{\mathcal{A}}}(\overline{\mathbf{y}})}\,\mathbf{c}. \tag{9}$$

That is, the Jacobian of $\Pi_{\mathcal{C}(\mathbf{x})}$ at $\mathbf{y}_{\text{raw}}$ is the orthogonal projector onto the tangent space of the active manifold $\mathcal{C}_{\mathcal{A}} = \mathcal{C}_1 \cap \mathcal{C}_{\mathcal{A},2}$.

In cost, the normal stage is a $d_{\mathbf{N}} \times d_{\mathbf{N}}$ system, solvable in $\mathcal{O}(d_{\mathbf{N}}^3)$ by LU or, in the flat case ($\mathbf{K_{NN}} = \mathbf{P_{NN}} \succ 0$), in $\mathcal{O}(\sqrt{\kappa_{\mathbf{N}}} \log(1/\varepsilon)\, d_{\mathbf{N}}^2)$ by CG, with $\kappa_{\mathbf{N}} \triangleq \operatorname{cond} \mathbf{K_{NN}}$; when $\mathcal{C}_1 = \{\mathbf{Az} = \mathbf{b}\}$ with $m_1 < d_{\mathbf{N}}$ rows, the Woodbury identity moves the solve to $m_1 \times m_1$, so the cost is $\mathcal{O}(\min(d_{\mathbf{N}}, m_1)^3 + d \min(d_{\mathbf{N}}, m_1))$. Appendix A3.2 specializes the theorem to polyhedra and second-order cones and records how an operator-splitting solver realizes it.

Assumption A allows a nonconvex set as long as it has no inward corners near $\overline{\mathbf{y}}$: prox-regularity with reach $R(\mathbf{x})$ means

$$\Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}) \text{ is a singleton whenever } \operatorname{dist}(\mathbf{y}, \mathcal{C}(\mathbf{x})) < R(\mathbf{x}), \tag{10}$$

equivalently $\langle \mathbf{v}, \mathbf{z} - \mathbf{w} \rangle \le \frac{\|\mathbf{v}\|}{2R(\mathbf{x})} \|\mathbf{z} - \mathbf{w}\|^2$ for all $\mathbf{z}, \mathbf{w} \in \mathcal{C}(\mathbf{x})$ near $\overline{\mathbf{y}}$ and $\mathbf{v} \in N_{\mathcal{C}(\mathbf{x})}(\mathbf{w})$, so $\Pi_{\mathcal{C}(\mathbf{x})}$ is single-valued and Lipschitz around $\mathbf{y}_{\text{raw}}$. Closed convex sets are the case $R(\mathbf{x}) = \infty$; obstacle

complements $\{\mathbf{z} : \|\mathbf{z} - \mathbf{c}\| \geq r\}$ (reach r), spheres and rotation groups have positive reach, and cardinality sets $\{\|\mathbf{z}\|_0 \leq s\}$ are prox-regular at points with exactly s nonzeros. For such nonconvex sets Theorems 1–2 are local statements at a given projection; global convergence of the forward solver is guaranteed only for convex blocks or warm starts inside the local basin, and Theorem 4 additionally assumes the one-step contraction of Assumption G in either case.

Assumption B asks that each constraint set look locally like a smooth low-dimensional surface, the active manifold, around $\overline{\mathbf{y}}$, with the projection varying smoothly along it. Assumption C asks that the displacement from the projection to the raw output point firmly into the normal cone, not toward an adjacent active manifold, so the active manifold is stable under perturbations of the input; and the two blocks have linearly independent normals, so the split (3) is unique with $\mathbf{n}_i \in \operatorname{relint} N_{\mathcal{C}_i}(\overline{\mathbf{y}})$ (Appendix A3.2). The linear independence of the block normals is a LICQ-type condition; under Assumption B it is transversality of the active manifolds, $T_{\mathcal{C}_{\mathcal{A},1}}(\overline{\mathbf{y}}) + T_{\mathcal{C}_{\mathcal{A},2}}(\overline{\mathbf{y}}) = \mathbb{R}^d$, and it implies $N_{\mathcal{C}_1}(\overline{\mathbf{y}}) \cap (-N_{\mathcal{C}_2}(\overline{\mathbf{y}})) = \{\mathbf{0}\}$. Under Assumptions B and D, with $\mathbf{P}, \mathbf{Q}$ evaluated at a split (3) with $\mathbf{n}_i \in \operatorname{relint} N_{\mathcal{C}_i}(\overline{\mathbf{y}})$ (one exists whenever the sum rule holds, e.g. for polyhedral blocks), it holds automatically (as shown next), so it is an additional hypothesis only for Theorem 2. Assumption D asks that the single dense block of the backward system be well conditioned: the two constraint sets meet at a non-degenerate angle, not nearly tangent. This is the implicit-function-theorem regularity for (3). Under Assumption B it implies the transversality condition of Assumption C (a vector in $\operatorname{span} N_{\mathcal{C}_1}(\overline{\mathbf{y}}) \cap \operatorname{span} N_{\mathcal{C}_2}(\overline{\mathbf{y}})$ lies in $\operatorname{range} \mathbf{N} \cap \ker \mathbf{K}_t^\top$); in the flat case the two are equivalent, and the content of the assumption is the uniform bound on the condition number.

Near $\overline{\mathbf{y}} + t\mathbf{n}_2$ the projection $\Pi_{\mathcal{C}_2}$ is the gradient of the $C^{1,1}$ function $\frac{1}{2}\|\cdot\|^2 - \frac{1}{2}\operatorname{dist}^2(\cdot, \mathcal{C}_2)$ (Poliquin et al., 2000), so $\mathbf{Q}$ is symmetric and the flat subspace $\mathcal{F}$ of Section 2.1 is the kernel of the second fundamental form of $\mathcal{C}_{\mathcal{A},2}$ along $\mathbf{n}_2$. On a polyhedral face $\mathcal{F} = T_{\mathcal{C}_{\mathcal{A},2}}(\overline{\mathbf{y}})$; on a curved manifold $\mathcal{F} \subsetneq T_{\mathcal{C}_{\mathcal{A},2}}(\overline{\mathbf{y}})$ (one flat ruling direction on a Lorentz cone, none on a ball). In any orthonormal basis aligned with $\mathbb{R}^d = \mathcal{F} \oplus \mathcal{F}^\perp$, the matrix $\mathbf{K}_t$ is block lower-triangular with a scalar leading block,

$$\mathbf{K}_t \;=\; \begin{bmatrix} t\,\mathbf{I}_{|\mathcal{F}|} & \mathbf{0} \\ (\mathbf{I} - \mathbf{Q}_{\mathbf{NN}})\mathbf{P}_{\mathbf{NF}} & t\,\mathbf{Q}_{\mathbf{NN}} + (\mathbf{I} - \mathbf{Q}_{\mathbf{NN}})\mathbf{P}_{\mathbf{NN}} \end{bmatrix}. \tag{11}$$

A2.5 Training procedure and the stochastic variant

This appendix gives the minibatch variant (Algorithm 2), which replaces the full-batch gradient by a minibatch average and the warm start by a pooled exponential moving average of $\Theta(d)$ memory. Its convergence is Theorem 4. Algorithm 1 accepts a face projection when it is feasible and its multipliers have the right sign. The minibatch variant keeps the face under inexact solves, so it replaces this check by a primal and a dual test with thresholds, described below. In the ambient regime (Phase one) the forward runs on $\mathbb{R}^d$ without using the active manifold, as in the ΠnetGrontas et al., 2026) hierarchy. The backward is the reduced solve at the current solver state in both regimes: with $\mathbf{P}$ and $\mathbf{Q}$ taken at that state it equals implicit differentiation of $\mathcal{T}$ in $\mathbb{R}^d$, so the dense Krylov solve of Πnet is never needed.

$$\textbf{Ambient} \begin{cases} \mathbf{s}^\star = \mathcal{T}(\mathbf{s}^\star, \mathbf{y}_{\text{raw}}; \mathbf{x}), & \text{(S1}'\text{)} \\ \mathbf{K}_{\mathbf{NN}}^\top \mathbf{v}_{\mathbf{N}} = (\mathbf{Pc})_{\mathbf{N}},\ \mathbf{v}_{\mathbf{F}} = \frac{1}{t}\big[(\mathbf{Pc})_{\mathbf{F}} - \mathbf{K}_{\mathbf{NF}}^\top \mathbf{v}_{\mathbf{N}}\big], & \text{(S2}'\text{)} \\ \boldsymbol{\theta} \mapsto \boldsymbol{\theta} - \eta\, \nabla_{\boldsymbol{\theta}} \ell. & \text{(S3}'\text{)} \end{cases}$$

The manifold regime (Phase two, on the slice $\mathcal{S}_B$ defined below) replaces the forward by its face-restricted analogue and keeps the same backward:

$$\textbf{Manifold} \begin{cases} \mathbf{s}^\star = \mathcal{T}^F(\mathbf{s}^\star, \mathbf{y}_{\text{raw}}; \mathbf{x}) \quad \text{on } \mathcal{S}_B, & \text{(S1)} \\ \mathbf{K}_{\mathbf{NN}}^\top \mathbf{v}_{\mathbf{N}} = (\mathbf{Pc})_{\mathbf{N}},\ \mathbf{v}_{\mathbf{F}} = \frac{1}{t}\big[(\mathbf{Pc})_{\mathbf{F}} - \mathbf{K}_{\mathbf{NF}}^\top \mathbf{v}_{\mathbf{N}}\big], & \text{(S2)} \\ \boldsymbol{\theta} \mapsto \boldsymbol{\theta} - \eta \left(t\,\mathbf{Q}(\mathbf{v}_{\mathbf{F}} + \mathbf{v}_{\mathbf{N}})\right)^\top \frac{\partial f_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}, & \text{(S3)} \end{cases}$$

with the flat-manifold specialization (9), and the face-restricted step of the solver's fixed-point map $\mathcal{T}$ (any solver of Section 2.1)

$$\mathcal{T}^F(\mathbf{s}, \mathbf{y}_{\text{raw}}; \mathbf{x}) \;\triangleq\; \mathbf{T}\,\mathcal{T}(\mathbf{T}\mathbf{s} + \mathbf{N}\mathbf{z}^B,\ \mathbf{y}_{\text{raw}};\ \mathbf{x}) + \mathbf{N}\mathbf{z}^B, \tag{12}$$

Algorithm 2 ASPL, stochastic variant (minibatch, pooled warm start)

Parameters: $\vartheta \in (0,1)$, $\mu \in (0,1)$; pool rate $\alpha \in (0,1]$; margin ν; total steps T. Init: $\boldsymbol{\theta}^{(0)}$, $(\bar{\mathbf{s}}^{(0)}, \bar{\mathbf{v}}^{(0)})$, $t{=}0$.

Phase one (ambient). While $t < T$, execute (S1′)–(S3′) on $\mathbb{R}^d$ for batch $\mathcal{B}^{(t)}$, producing $\{\mathbf{s}_i^{(t)}, \mathbf{v}_i^{(t)}\}_{i\in\mathcal{B}^{(t)}}$:

If $\mathcal{U}^{(t)}{=}\emptyset$ and $E_{\mathcal{N}}^{(t)} < \vartheta E^{(t)}$, then $\vartheta \leftarrow \mu\vartheta$. If $E_{\mathcal{N}}^{(t)} \geq \vartheta E^{(t)}$, goto phase two.

Pool: $\bar{\mathbf{s}}^{(t+1)}{=}(1{-}\alpha)\bar{\mathbf{s}}^{(t)} + \frac{\alpha}{B}\sum_{i\in\mathcal{B}^{(t)}} \mathbf{s}_i^{(t)}$, $\bar{\mathbf{v}}^{(t+1)}{=}(1{-}\alpha)\bar{\mathbf{v}}^{(t)} + \frac{\alpha}{B}\sum_{i\in\mathcal{B}^{(t)}} \mathbf{v}_i^{(t)}$; $t \leftarrow t{+}1$.

Phase two (manifold). While $t < T$, execute (S1)–(S3) on $\mathcal{S}_B$ for batch $\mathcal{B}^{(t)}$, producing $\{\mathbf{s}_i^{(t)}, \mathbf{v}_i^{(t)}\}_{i\in\mathcal{B}^{(t)}}$:

If $E_{\mathcal{N}}^{(t)} < \vartheta E^{(t)}$, goto phase one.

Pool: $\bar{\mathbf{s}}^{(t+1)}{=}(1{-}\alpha)\bar{\mathbf{s}}^{(t)} + \frac{\alpha}{B}\sum_{i\in\mathcal{B}^{(t)}} \mathbf{s}_i^{(t)}$, $\bar{\mathbf{v}}^{(t+1)}{=}(1{-}\alpha)\bar{\mathbf{v}}^{(t)} + \frac{\alpha}{B}\sum_{i\in\mathcal{B}^{(t)}} \mathbf{v}_i^{(t)}$; $t \leftarrow t{+}1$.

Return $\boldsymbol{\theta}^{(T)}$.

where $\mathbf{T} \triangleq \mathbf{I} - \mathbf{N} = \mathbf{F}$ and $\mathbf{z}^B \triangleq \mathbf{N}\mathbf{s}^\star$ pins the normal coordinates of the solver state at their fixed-point values (for (8), $\mathbf{N}\mathbf{s}^\star = \mathbf{N}(\overline{\mathbf{y}} - \mathbf{n}_1)$, which differs from the bound values unless $\mathbf{N}\mathbf{n}_1 = \mathbf{0}$); pinning the state at the bound values instead changes the fixed point and can return an infeasible output. The iteration (S1) runs on the slice $\mathcal{S}_B \triangleq \{\mathbf{s} : \mathbf{N}\mathbf{s} = \mathbf{N}\mathbf{z}^B\}$, which contains $\mathbf{s}^\star$ (Proposition 2). Subproblem (S2) is the two-stage solve of Theorem 1; for polyhedral faces it reduces to a $d_{\mathbf{N}} \times d_{\mathbf{N}}$ Cholesky of $\mathbf{P}_{\mathbf{NN}}$ (or warm-started Jacobi-CG) plus one $(d - d_{\mathbf{N}}) \times d_{\mathbf{N}}$ product (Corollary 3). The switch between regimes uses two residuals,

$$E^{(t)} \triangleq \|\mathbf{y}_{\text{raw}}^{(t)} - \mathbf{y}^{(t)}\|_2, \qquad E_{\mathcal{N}}^{(t)} \triangleq \|\mathbf{N}^{(t)}(\mathbf{y}_{\text{raw}}^{(t)} - \mathbf{y}^{(t)})\|_2 \leq E^{(t)}, \tag{13}$$

the total projection displacement and its component aligned with the candidate normal cone. The KKT condition $\mathbf{y}_{\text{raw}} - \mathbf{y} \in N_{\mathcal{C}(\mathbf{x})}(\mathbf{y})$ forces the displacement to be purely normal at a true fixed point, so a small $E_{\mathcal{N}}^{(t)}/E^{(t)}$ signals a wrong candidate face (for $\mathcal{C}_1$ affine, the component of the displacement in $\operatorname{range}\mathbf{A}^\top$ is also normal, so the ratio is a heuristic threshold rather than exactly 1 at the fixed point). With threshold $\vartheta \in (0,1)$:

$$\text{REGIME}_t = \begin{cases} \text{manifold} & \text{if } E_{\mathcal{N}}^{(t)} \geq \vartheta\, E^{(t)}, \\ \text{ambient} & \text{if } E_{\mathcal{N}}^{(t)} < \vartheta\, E^{(t)}. \end{cases} \tag{14}$$

When $\mathcal{C}_2$ admits a tractable dual (boxes, SOCs, simplices), the block normal $\boldsymbol{\lambda}^{(t)} \triangleq \mathbf{n}_2^{(t)}$ of the split (3) serves as a Lagrange multiplier on $\mathcal{C}_2$; the undecided set

$$\mathcal{U}^{(t)} \triangleq \{\, j \,:\, \|\boldsymbol{\lambda}_j^{(t)}\| < \nu \text{ or } \boldsymbol{\lambda}_j^{(t)} \notin N_{\mathcal{C}_2}^{(t)}(\mathbf{y}^{(t)}) \,\} \tag{15}$$

collects components with weak or misaligned dual evidence, and we lower the threshold, $\vartheta \leftarrow \mu\vartheta$ ($\mu \in (0,1)$), whenever $\mathcal{U}^{(t)} = \emptyset$: clean dual evidence relaxes the primal test, and since ϑ only decreases, a later return to the ambient regime requires a larger drop of $E_{\mathcal{N}}^{(t)}/E^{(t)}$. If the per-sample active manifolds stop changing after some T_0 and strict complementarity holds with margin ν, the rule stays in the manifold regime for all $t \geq T_0$; we do not prove that such a T_0 exists; a stale candidate is detected by the primal check, which returns the algorithm to the ambient regime. Both residuals reuse forward-pass quantities, so the test in (14) adds one matvec, one norm, and one comparison per outer step. The pooled EMA states $\bar{\mathbf{s}}^{(t)}, \bar{\mathbf{v}}^{(t)}$ warm-start the forward solver and, with CG, the reduced backward at $\Theta(d)$ memory across the dataset.

The released code uses the simplest instance of this rule: it never restricts the forward to the face. At every step it runs the full forward solver, reads the active structure off its output, and differentiates on it. The polyhedral layers (QP, motion planning, maze) solve on the bound-active coordinates or the identified rows, and the conic layer (Appendix A4.2) rebuilds the cone Jacobians from the current dual iterate and, in our SOCP runs, solves the dense $m \times m$ adjoint system with them rather than the reduced one. The primal and dual tests above matter only for the optional face-restricted forward.

A3 Addendum to Section 3: stochastic variant and proofs

A3.1 End-to-end convergence

The notation and Assumptions E–G below serve both Theorem 3 (deterministic, Section 3; Assumption F is not needed there) and its stochastic counterpart, Theorem 4. For the stochastic counterpart, contexts are drawn from a distribution $\mathcal{D}$, the objective of (PCO) is replaced by its expectation over $\mathbf{x} \sim \mathcal{D}$, and Assumptions A–D are required for $\mathcal{D}$-almost every $\mathbf{x}$. Let $\boldsymbol{\theta}^{(t)}$ denote the backbone parameters at outer step t (superscripts (t) index outer steps; the split scalar t of (3) appears only inside the read-out $\mathbf{g}_{\mathbf{x}}$ and $\nabla\Phi$), and $(\overline{\mathbf{s}}^{(t)}, \overline{\mathbf{v}}^{(t)})$ the pooled primal and adjoint EMA states. The analysis applies to any forward solver expressible as a fixed-point map $\mathcal{T}$ that contracts locally (Assumption G) and whose state can be pooled across the dataset (the default is given in Appendix A2.1). For each context $\mathbf{x}$, $\mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})$ is the per-sample fixed point of $\mathcal{T}$ and $\mathbf{v}^\star(\boldsymbol{\theta}; \mathbf{x})$ the per-sample adjoint of (6) at that fixed point, for the cotangent $\mathbf{c} = \partial\ell/\partial\mathbf{y}$. The population objective is $\Phi(\boldsymbol{\theta}) \triangleq \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}\, \varphi(\Pi_{\mathcal{C}(\mathbf{x})}(f_{\boldsymbol{\theta}}(\mathbf{x})), \mathbf{x})$ as in (PCO), with per-sample loss $\ell \triangleq \varphi(\cdot, \mathbf{x})$, and $\Phi^\star \triangleq \inf_{\boldsymbol{\theta}} \Phi(\boldsymbol{\theta}) > -\infty$. For a forward state $\mathbf{s}$ we write $\mathbf{g}_{\mathbf{x}}(\mathbf{s}) \triangleq (\partial f_{\boldsymbol{\theta}}(\mathbf{x})/\partial\boldsymbol{\theta})^\top t\mathbf{Q}(\mathbf{s})\mathbf{v}(\mathbf{s})$ for the per-sample gradient read-out (S3) computed from $\mathbf{s}$ (with $\mathbf{Q}(\mathbf{s})$, $\mathbf{v}(\mathbf{s})$ the curvature operator and adjoint obtained from the split read off $\mathbf{s}$). At $\mathbf{s} = \mathbf{s}^\star$ it is the exact per-sample gradient (Theorem 1), so that $\mathbf{g}_{\mathbf{x}}(\mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})) = (\partial f_{\boldsymbol{\theta}}(\mathbf{x})/\partial\boldsymbol{\theta})^\top t\mathbf{Q}\, \mathbf{v}^\star(\boldsymbol{\theta}; \mathbf{x})$. The stochastic gradient at step t is $\mathbf{g}^{(t)} \triangleq \frac{1}{B}\sum_{i\in\mathcal{B}^{(t)}} \mathbf{g}_{\mathbf{x}_i}(\mathbf{s}_i^{(t)})$, where $\mathbf{s}_i^{(t)}$ is the per-sample forward output after K_{dr} steps from the warm start $\overline{\mathbf{s}}^{(t)}$. We write B for the batch size, T for total outer steps, and $\mathcal{N}_\circ$ for the bounded open parameter set of Assumption C(ii); Theorem 4 assumes that the iterates $\boldsymbol{\theta}^{(t)}$ and the segments between consecutive iterates stay in $\mathcal{N}_\circ$.

Assumption E (Smoothness)**.** *The maps $f_{\boldsymbol{\theta}}$ and ℓ are C^1, and the forward map $\mathcal{T}(\cdot, \cdot; \mathbf{x})$ is C^1 near its fixed points. Moreover, we assume that Φ is differentiable with L_Φ-Lipschitz gradient on $\mathcal{N}_\circ$ (this does not follow from the other conditions: $\Pi_{\mathcal{C}(\mathbf{x})}$ is only piecewise smooth, and smoothness of Φ relies on averaging over $\mathbf{x} \sim \mathcal{D}$), and that $\nabla\Phi(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x}}\big[(\partial f_{\boldsymbol{\theta}}(\mathbf{x})/\partial\boldsymbol{\theta})^\top t\mathbf{Q}\, \mathbf{v}^\star(\boldsymbol{\theta}; \mathbf{x})\big]$ on $\mathcal{N}_\circ$ (differentiation under the expectation).*

Assumption F (Stochastic gradients)**.** *Each iteration draws an i.i.d. batch $\mathcal{B}^{(t)} = \{\mathbf{x}_i\}_{i=1}^B \sim \mathcal{D}^B$, independent of $\mathcal{F}_t \triangleq \sigma(\boldsymbol{\theta}^{(t)}, \overline{\mathbf{s}}^{(t)}, \overline{\mathbf{v}}^{(t)})$. There exists $\sigma^2 < \infty$ such that, whenever $\boldsymbol{\theta}^{(t)} \in \mathcal{N}_\circ$,*

$$\mathbb{E}\big[\|\mathbf{g}^{(t)} - \mathbb{E}[\mathbf{g}^{(t)} \,|\, \mathcal{F}_t]\|^2 \,\big|\, \mathcal{F}_t\big] \le \sigma^2/B,$$

which holds whenever the per-sample read-out $\mathbf{g}_{\mathbf{x}}(\mathbf{s}_{\mathbf{x}}^{(t)})$ has conditional variance at most σ^2.

Assumption G (Forward contraction)**.** *Let $R_s \triangleq \sup \|\mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})\| < \infty$ over $\boldsymbol{\theta} \in \mathcal{N}_\circ$ and $\mathcal{D}$-a.e. $\mathbf{x}$, and $D \triangleq \max\{\|\overline{\mathbf{s}}^{(0)}\|, 3R_s\} + R_s$. There is $\delta \in (0, 1)$ such that the forward map ($\mathcal{T}$, and $\mathcal{T}^F$ of (12) with $\mathbf{N}\mathbf{z}^B = \mathbf{N}\mathbf{s}^\star$ in the manifold regime) satisfies $\|\mathcal{T}(\mathbf{s}, f_{\boldsymbol{\theta}}(\mathbf{x}); \mathbf{x}) - \mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})\| \le (1-\delta)\|\mathbf{s} - \mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})\|$ whenever $\|\mathbf{s} - \mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})\| \le D$. This is a condition on the forward solver, not a consequence of Assumption D.*

Theorem 4 (Convergence of stochastic ASPL)**.** *Under Assumptions A–G, with parts (ii) of Assumptions C and D holding for $\mathcal{D}$-almost every $\mathbf{x}$ in place of the training contexts, suppose $\boldsymbol{\theta}^{(t)}$ and the segments $[\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)}]$ lie in $\mathcal{N}_\circ$ for all t, the backward solve is exact, and the per-sample read-out $\mathbf{g}_{\mathbf{x}}$ satisfies $\|\mathbf{g}_{\mathbf{x}}(\mathbf{s}) - \mathbf{g}_{\mathbf{x}}(\mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x}))\| \le L_g\|\mathbf{s} - \mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})\|$ whenever $\|\mathbf{s} - \mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})\| \le D$. Run Algorithm 2 with $\eta = c_\eta T^{-1/2}$, $T \ge 16L_\Phi^2 c_\eta^2$, any $\alpha \in (0, 1]$, and $K_{\mathrm{dr}} \ge \delta^{-1}\max\{\ln 2, \frac{1}{2}\ln(1/\eta)\}$. Then*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\|\nabla\Phi(\boldsymbol{\theta}^{(t)})\|^2 \;\le\; \frac{4(\Phi(\boldsymbol{\theta}^{(0)}) - \Phi^\star)}{c_\eta\sqrt{T}} + \frac{3c_\eta L_g^2 D^2}{\sqrt{T}} + \frac{2c_\eta L_\Phi \sigma^2}{B\sqrt{T}}. \tag{16}$$

Consequently $\min_{t<T} \mathbb{E}\|\nabla\Phi(\boldsymbol{\theta}^{(t)})\|^2 \le \epsilon$ for $T = \mathcal{O}(\epsilon^{-2})$. If, moreover, $\mathcal{T}$ satisfies (S) with a constant L_u and the radius ϱ of Theorem 2 is bounded below by $\varrho_{\min} > 0$ uniformly over $\mathbf{x}$ and $\boldsymbol{\theta} \in \mathcal{N}_\circ$, then for $T > \big(L_u D\sqrt{c_\eta}/((1-\delta)\varrho_{\min})\big)^4$ every per-sample forward output of Algorithm 2 lies on the block active manifolds $\mathcal{C}_{\mathcal{A},i}(\mathbf{x})$ of the current $\boldsymbol{\theta}^{(t)}$, so the partition read off the forward pass is the correct one at every step.

The rate matches the optimal nonconvex stochastic-gradient rate $\mathcal{O}(T^{-1/2})$ with a T-independent constant; the only price of the pooled warm start is $K_{\mathrm{dr}} = \mathcal{O}(\delta^{-1}\log T)$ forward steps, which makes the gradient bias $\mathcal{O}(\sqrt{\eta})$. The pooled adjoint $\overline{\mathbf{v}}^{(t)}$ plays no role in the guarantee; it only warm-starts the Krylov backward. The proof (Appendix A3.7) is a biased-SGD descent argument. Our QP experiments train with AdamW under a cosine schedule (Appendix A4) rather than with SGD at $\eta \propto T^{-1/2}$, so they lie outside the literal scope of Theorem 4.

A3.2 Proof of Theorem 1 and corollaries

The proof never looks inside the forward solver. It writes the three identities (3) as one equation $G(\mathbf{u};\mathbf{y}) = \mathbf{0}$ in an auxiliary point $\mathbf{u}$, applies the implicit function theorem to this equation, and reads off the Jacobian of the projection. Step 1 shows that the split (3) is unique. Step 2 builds G. Step 3 shows that its Jacobian $\mathbf{K}_t$ is block lower-triangular. Step 4 applies the implicit function theorem, and Step 5 derives the two-stage solve.

A3.2.1 Proof of Theorem 1

Proof. Step 1: the split is unique and lies in the relative interiors. Assumption C(i) gives

$$\operatorname{span} N_{\mathcal{C}_1}(\overline{\mathbf{y}}) \cap \operatorname{span} N_{\mathcal{C}_2}(\overline{\mathbf{y}}) = \{\mathbf{0}\}. \tag{17}$$

In particular $N_{\mathcal{C}_1}(\overline{\mathbf{y}}) \cap (-N_{\mathcal{C}_2}(\overline{\mathbf{y}})) = \{\mathbf{0}\}$. Under this qualification the normal cone of the intersection is the sum of the block normal cones,

$$N_{\mathcal{C}}(\overline{\mathbf{y}}) = N_{\mathcal{C}_1}(\overline{\mathbf{y}}) + N_{\mathcal{C}_2}(\overline{\mathbf{y}}),$$

by Bauschke & Combettes (2017, Cor. 16.50) for convex blocks and by Rockafellar & Wets (1998, Thm. 6.42) for prox-regular blocks, which are Clarke regular. The residual $\mathbf{y}_{\mathrm{raw}} - \overline{\mathbf{y}} \in N_{\mathcal{C}}(\overline{\mathbf{y}})$ therefore splits as $\mathbf{n}_1 + \mathbf{n}_2$ with $\mathbf{n}_i \in N_{\mathcal{C}_i}(\overline{\mathbf{y}})$. The split is unique: if $\mathbf{n}_1 + \mathbf{n}_2 = \mathbf{n}_1' + \mathbf{n}_2'$, then $\mathbf{n}_1 - \mathbf{n}_1' = \mathbf{n}_2' - \mathbf{n}_2$ lies in both spans and vanishes by (17).

The split also lies in the relative interiors. Assumption C(i) places the residual in $\operatorname{relint} N_{\mathcal{C}}(\overline{\mathbf{y}})$, and $\operatorname{relint}(K_1 + K_2) = \operatorname{relint} K_1 + \operatorname{relint} K_2$ for convex cones (Rockafellar, 1970, Cor. 6.6.2). So the residual has a split with $\mathbf{n}_i' \in \operatorname{relint} N_{\mathcal{C}_i}(\overline{\mathbf{y}})$, and uniqueness gives $\mathbf{n}_i = \mathbf{n}_i'$ (Liang et al., 2017, Lem. A.5).

Finally, the block projections realize the split, which is (3). For a set that is prox-regular at $\overline{\mathbf{y}}$, a short enough normal $\mathbf{v}$ is a proximal normal and $\Pi(\overline{\mathbf{y}} + \mathbf{v}) = \overline{\mathbf{y}}$ (Poliquin et al., 2000). For $\mathcal{C}_2$ this holds for $t\mathbf{n}_2$ with t small enough, and for every $t \in (0, 1]$ when $\mathcal{C}_2$ is convex. For $\mathcal{C}_1$ the normal $\mathbf{n}_1$ is not rescaled. It is short enough by the bound in Assumption B when $\mathcal{C}_1$ is nonconvex, and automatically when $\mathcal{C}_1$ is convex. Near $\overline{\mathbf{y}} + \mathbf{n}_1$ the projection $\Pi_{\mathcal{C}_1}$ is the gradient of the $C^{1,1}$ function $\frac{1}{2}\|\cdot\|^2 - \frac{1}{2}\operatorname{dist}^2(\cdot, \mathcal{C}_1)$ (Poliquin et al., 2000), so its differential $\mathbf{P}$ is symmetric. The same argument makes $\mathbf{Q}$ symmetric.

Step 2: one equation that every exact solver satisfies. Set $\mathbf{u}_1 \triangleq \overline{\mathbf{y}} + \mathbf{n}_1$. For $(\mathbf{u}, \mathbf{y})$ near $(\mathbf{u}_1, \mathbf{y}_{\mathrm{raw}})$ define

$$G(\mathbf{u};\mathbf{y}) \triangleq \Pi_{\mathcal{C}_1}(\mathbf{u}) - \Pi_{\mathcal{C}_2}\big(\Pi_{\mathcal{C}_1}(\mathbf{u}) + t\,(\mathbf{y} - \mathbf{u})\big).$$

The first term is the output of the $\mathcal{C}_1$ projection. The argument of $\Pi_{\mathcal{C}_2}$ adds to it the scaled remaining residual. At the solution, $\Pi_{\mathcal{C}_1}(\mathbf{u}_1) = \overline{\mathbf{y}}$ and

$$\Pi_{\mathcal{C}_1}(\mathbf{u}_1) + t\,(\mathbf{y}_{\mathrm{raw}} - \mathbf{u}_1) = \overline{\mathbf{y}} + t\,(\mathbf{y}_{\mathrm{raw}} - \overline{\mathbf{y}} - \mathbf{n}_1) = \overline{\mathbf{y}} + t\,\mathbf{n}_2,$$

so $G(\mathbf{u}_1;\mathbf{y}_{\mathrm{raw}}) = \overline{\mathbf{y}} - \Pi_{\mathcal{C}_2}(\overline{\mathbf{y}} + t\mathbf{n}_2) = \mathbf{0}$ by (3). Both block projections are C^1 near the points where they are evaluated, because each block is partly smooth and each $\mathbf{n}_i$ lies in the relative interior of its normal cone (Poliquin et al., 2000; Lewis, 2002). Their differentials there are $\mathbf{P}$ and $\mathbf{Q}$ of (4), and the chain rule gives

$$D_{\mathbf{u}}G(\mathbf{u}_1;\mathbf{y}_{\mathrm{raw}}) = \mathbf{P} - \mathbf{Q}(\mathbf{P} - t\mathbf{I}) = \mathbf{K}_t, \qquad D_{\mathbf{y}}G(\mathbf{u}_1;\mathbf{y}_{\mathrm{raw}}) = -t\,\mathbf{Q}.$$

Step 3: $\mathbf{K}_t$ is block lower-triangular. Since $\mathbf{Q}$ is symmetric, its fixed space $\mathcal{F}$ is invariant, $\mathbf{FQ} = \mathbf{QF} = \mathbf{F}$, and $\mathbf{Q} = \mathbf{F} + \mathbf{Q}_{\mathbf{NN}}$ with $\mathbf{Q}_{\mathbf{FN}} = \mathbf{Q}_{\mathbf{NF}} = \mathbf{0}$. Expanding $\mathbf{K}_t = \mathbf{P} + t\mathbf{Q} - \mathbf{QP}$ block by

block gives

$$\mathbf{K_{FF}} = \mathbf{P_{FF}} + t\mathbf{I} - \mathbf{P_{FF}} = t\mathbf{I}, \qquad \mathbf{K_{FN}} = \mathbf{P_{FN}} - \mathbf{P_{FN}} = \mathbf{0},$$
$$\mathbf{K_{NF}} = (\mathbf{I} - \mathbf{Q_{NN}})\mathbf{P_{NF}}, \qquad \mathbf{K_{NN}} = t\mathbf{Q_{NN}} + (\mathbf{I} - \mathbf{Q_{NN}})\mathbf{P_{NN}},$$

which is (11). The diagonal blocks are $t\mathbf{I}$ and $\mathbf{K_{NN}}$, and $\mathbf{K_{NN}}$ is nonsingular by Assumption D. Hence $\mathbf{K}_t$ is nonsingular.

Step 4: the implicit function theorem gives the Jacobian. Since $\mathbf{K}_t$ is nonsingular, the implicit function theorem gives a C^1 map $\mathbf{y} \mapsto \mathbf{u}(\mathbf{y})$ near $\mathbf{y}_{\text{raw}}$ with $\mathbf{u}(\mathbf{y}_{\text{raw}}) = \mathbf{u}_1$, $G(\mathbf{u}(\mathbf{y}); \mathbf{y}) = \mathbf{0}$, and

$$D\mathbf{u} \;=\; -\mathbf{K}_t^{-1} D_{\mathbf{y}} G \;=\; t\,\mathbf{K}_t^{-1}\mathbf{Q}.$$

We show that $\mathbf{p}(\mathbf{y}) \triangleq \Pi_{\mathcal{C}_1}(\mathbf{u}(\mathbf{y}))$ is the projection of $\mathbf{y}$ onto $\mathcal{C}$. Put $\mathbf{n}_1(\mathbf{y}) \triangleq \mathbf{u}(\mathbf{y}) - \mathbf{p}(\mathbf{y})$ and $\mathbf{n}_2(\mathbf{y}) \triangleq \mathbf{y} - \mathbf{p}(\mathbf{y}) - \mathbf{n}_1(\mathbf{y})$. The first is a proximal normal to $\mathcal{C}_1$ at $\mathbf{p}(\mathbf{y})$, since $\mathbf{p}(\mathbf{y})$ is the projection of $\mathbf{u}(\mathbf{y})$. The equation $G = \mathbf{0}$ says $\mathbf{p}(\mathbf{y}) = \Pi_{\mathcal{C}_2}(\mathbf{p}(\mathbf{y}) + t\mathbf{n}_2(\mathbf{y}))$, so $\mathbf{n}_2(\mathbf{y})$ is a proximal normal to $\mathcal{C}_2$ at $\mathbf{p}(\mathbf{y})$. A proximal normal to $\mathcal{C}_i \supseteq \mathcal{C}$ at a point of $\mathcal{C}$ is a proximal normal to $\mathcal{C}$, so $\mathbf{y} - \mathbf{p}(\mathbf{y}) = \mathbf{n}_1(\mathbf{y}) + \mathbf{n}_2(\mathbf{y})$ is a proximal normal to $\mathcal{C}$ at $\mathbf{p}(\mathbf{y})$. For $\mathbf{y}$ near $\mathbf{y}_{\text{raw}}$ its length is below the reach of $\mathcal{C}$ (Assumption A). There the projection onto $\mathcal{C}$ is single-valued, and a point with a proximal normal of such length is the projection (Poliquin et al., 2000). Hence $\mathbf{p}(\mathbf{y}) = \Pi_{\mathcal{C}}(\mathbf{y})$. Differentiating $\Pi_{\mathcal{C}}(\mathbf{y}) = \Pi_{\mathcal{C}_1}(\mathbf{u}(\mathbf{y}))$ at $\mathbf{y}_{\text{raw}}$ gives

$$D\Pi_{\mathcal{C}}(\mathbf{y}_{\text{raw}}) \;=\; \mathbf{P}\,D\mathbf{u} \;=\; t\,\mathbf{P}\mathbf{K}_t^{-1}\mathbf{Q},$$

which is (5).

Step 5: the two-stage solve. Since $\mathbf{P}$ and $\mathbf{Q}$ are symmetric, the backward pass for a cotangent $\mathbf{c}$ is

$$\frac{\partial \ell}{\partial \mathbf{y}_{\text{raw}}} \;=\; (D\Pi_{\mathcal{C}})^{\top}\mathbf{c} \;=\; t\,\mathbf{Q}\mathbf{K}_t^{-\top}\mathbf{P}\mathbf{c} \;=\; t\,\mathbf{Q}\mathbf{v}, \qquad \mathbf{K}_t^{\top}\mathbf{v} = \mathbf{P}\mathbf{c}.$$

In the basis of Step 3, $\mathbf{K}_t^{\top}$ is block upper-triangular with diagonal blocks $t\mathbf{I}$ and $\mathbf{K}_{\mathbf{NN}}^{\top}$ and off-diagonal block $\mathbf{K}_{\mathbf{NF}}^{\top}$. Back-substitution solves the second block row for $\mathbf{v_N}$ and then the first for $\mathbf{v_F}$, which is (6). The $\mathcal{F}$-stage is a scalar division and one product with $\mathbf{K}_{\mathbf{NF}}^{\top}$, at cost $\mathcal{O}(d\,d_{\mathbf{N}})$. The only factorization is that of $\mathbf{K_{NN}}$. □

Proof of Corollary 1. Take $t = 1$. Since $\mathcal{C}_1$ is affine, $\mathbf{P}$ is the orthogonal projector onto its direction space. Since $\mathcal{C}_{\mathcal{A},2}$ is locally affine, partial smoothness and strict complementarity make $\Pi_{\mathcal{C}_2}$, near $\overline{\mathbf{y}} + \mathbf{n}_2$, the affine projection onto $\operatorname{aff}\mathcal{C}_{\mathcal{A},2}$, so $\mathbf{Q} = \mathbf{T}$. Then $\mathbf{Q_{NN}} = \mathbf{0}$, $\mathbf{K_{NN}} = \mathbf{P_{NN}}$ and $\mathbf{K_{NF}} = \mathbf{P_{NT}}$, and (6) reads

$$\mathbf{P_{NN}}\mathbf{v_N} = (\mathbf{P}\mathbf{c})_{\mathbf{N}}, \qquad \mathbf{v_T} = (\mathbf{P}\mathbf{c})_{\mathbf{T}} - \mathbf{P_{TN}}\mathbf{v_N},$$

with gradient $\mathbf{Q}\mathbf{v} = \mathbf{v_T}$. To identify this gradient, let

$$\mathbf{M} \;\triangleq\; \mathbf{P} - \mathbf{P}\mathbf{N}(\mathbf{P_{NN}})^{-1}\mathbf{N}\mathbf{P}.$$

Then $\mathbf{v_T} = \mathbf{T}\mathbf{M}\mathbf{c}$. Since $\mathbf{N}\mathbf{M} = \mathbf{N}\mathbf{P} - \mathbf{P_{NN}}(\mathbf{P_{NN}})^{-1}\mathbf{N}\mathbf{P} = \mathbf{0}$, we have $\mathbf{T}\mathbf{M} = \mathbf{M}$, so the gradient is $\mathbf{M}\mathbf{c}$. The matrix $\mathbf{M}$ is symmetric. It fixes every vector of $\operatorname{range}\mathbf{P} \cap \operatorname{range}\mathbf{T}$ and annihilates $(\operatorname{range}\mathbf{P})^{\perp} + \operatorname{range}\mathbf{N}$, the orthogonal complement of that subspace. So $\mathbf{M}$ is the orthogonal projector onto $\operatorname{range}\mathbf{P} \cap \operatorname{range}\mathbf{T} = T_{\mathcal{C}_{\mathcal{A}}}(\overline{\mathbf{y}})$. □

A3.2.2 Proof of Proposition 1

Proof of Proposition 1. Let the solver iterate $\mathbf{s} \mapsto \mathcal{T}(\mathbf{s}, \mathbf{y}; \mathbf{x})$ and return $\overline{\mathbf{y}} = \mathsf{r}(\mathbf{s}^\star, \mathbf{y})$ through a C^1 read-out r. Since $\mathbf{I} - D_{\mathbf{s}}\mathcal{T}$ is invertible, the implicit function theorem gives a C^1 fixed-point map $\mathbf{y} \mapsto \mathbf{s}^\star(\mathbf{y})$ near $\mathbf{y}_{\text{raw}}$. The solver returns the projection for every such $\mathbf{y}$, so

$$\mathsf{r}(\mathbf{s}^\star(\mathbf{y}), \mathbf{y}) \;=\; \Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}) \qquad \text{near } \mathbf{y}_{\text{raw}}.$$

Differentiating this identity shows that implicit differentiation through $\mathcal{T}$ returns $D\Pi_{\mathcal{C}(\mathbf{x})}(\mathbf{y}_{\text{raw}})$. By Theorem 1 this is (5). It depends on the forward pass only through $(\overline{\mathbf{y}}, \mathbf{n}_1, \mathbf{n}_2, t)$, and $(\mathbf{n}_1, \mathbf{n}_2)$ is unique by Assumption C.

It remains to discuss the invertibility hypothesis for the three solvers named in the proposition. For the splitting step (8), the chain rule gives $\mathbf{I} - D_{\mathbf{s}}\mathcal{T} = \omega\,\mathbf{K}_{1/2}^{\top}$, which is nonsingular by Assumption D. Dykstra's algorithm uses $\mathbf{y}_{\text{raw}}$ only through its initialization, and it keeps the sum of its iterate and its two correction vectors $(\mathbf{p}, \mathbf{q})$ equal to $\mathbf{y}_{\text{raw}}$. Its state is therefore $(\mathbf{p}, \mathbf{q})$, with iterate $\mathbf{y}_{\text{raw}} - \mathbf{p} - \mathbf{q}$, and the hypothesis applies to this reduced map. On the full state the fixed points form a family indexed by $\mathbf{y}_{\text{raw}}$, and $\mathbf{I} - D_{\mathbf{s}}\mathcal{T}$ is singular. For an active-set QP, the KKT matrix with the active set frozen takes the place of $\mathbf{I} - D_{\mathbf{s}}\mathcal{T}$. It is invertible under LICQ, that is, when the rows of $\mathbf{A}$ and the active constraint gradients are linearly independent, and LICQ implies the linear-independence condition of Assumption C. □

Corollary 2 (Second-order cone)**.** *For $\mathcal{C}_2 = \{(\mathbf{w}, t) : \|\mathbf{w}\|_2 \le t\}$ (in this corollary and its proof, t and $\bar{t}$ denote the cone's scalar coordinate, not the split scalar of (3)), let $\overline{\mathbf{y}} = (\overline{\mathbf{w}}, \bar{t})$ be a boundary projection with $\|\overline{\mathbf{w}}\| = \bar{t} > 0$, and let the input to the cone projector be $\mathbf{u} = (\mathbf{a}, b)$ with $\rho = \|\mathbf{a}\|$ and $\mathbf{q} = \mathbf{a}/\rho = \overline{\mathbf{w}}/\bar{t}$. Then*

$$\mathbf{Q}_{\text{soc}} = \begin{bmatrix} \kappa(\mathbf{I} - \mathbf{q}\mathbf{q}^{\top}) + \frac{1}{2}\mathbf{q}\mathbf{q}^{\top} & \frac{1}{2}\mathbf{q} \\ \frac{1}{2}\mathbf{q}^{\top} & \frac{1}{2} \end{bmatrix}, \qquad \kappa \triangleq \bar{t}/\rho.$$

This operator has range equal to the tangent space, kills the normal direction, and equals the tangent projector only in the zero-curvature limit $\kappa = 1$. Substituting $\mathbf{Q}_{\text{soc}}$ into Theorem 1 gives a structured $d_{\mathbf{N}} \times d_{\mathbf{N}}$ solve; for Cartesian products of cones, $\mathbf{Q}$ is block diagonal; a boundary-active cone in $\mathbb{R}^{n+1}$ contributes one normal and $n-1$ curved tangent directions, i.e. n dimensions to $d_{\mathbf{N}}$.

Proof of Corollary 2. For an input $\mathbf{u} = (\mathbf{a}, b)$ in the middle region $-\|\mathbf{a}\| < b < \|\mathbf{a}\|$, the Lorentz-cone projector is

$$\Pi_{\mathcal{C}_2}(\mathbf{a}, b) = \left(\tfrac{\rho+b}{2}\mathbf{q},\ \tfrac{\rho+b}{2}\right), \qquad \rho = \|\mathbf{a}\|, \quad \mathbf{q} = \mathbf{a}/\rho.$$

Writing $\bar{t} = (\rho + b)/2$ and differentiating ρ and $\mathbf{q}$ gives

$$d\bar{t} = \tfrac{1}{2}\mathbf{q}^{\top} d\mathbf{a} + \tfrac{1}{2} db, \qquad d\mathbf{q} = \rho^{-1}(\mathbf{I} - \mathbf{q}\mathbf{q}^{\top}) d\mathbf{a}.$$

Thus

$$d\overline{\mathbf{w}} = \mathbf{q}\, d\bar{t} + \bar{t}\, d\mathbf{q} = \left[\kappa(\mathbf{I} - \mathbf{q}\mathbf{q}^{\top}) + \tfrac{1}{2}\mathbf{q}\mathbf{q}^{\top}\right] d\mathbf{a} + \tfrac{1}{2}\mathbf{q}\, db, \qquad d\bar{t} = \tfrac{1}{2}\mathbf{q}^{\top} d\mathbf{a} + \tfrac{1}{2} db,$$

with $\kappa = \bar{t}/\rho$, which is exactly the displayed $\mathbf{Q}_{\text{soc}}$.

The boundary manifold is $\{(\mathbf{w}, t) : \|\mathbf{w}\| = t > 0\}$. Its outward unit normal is $\mathbf{n} = (\mathbf{q}, -1)^{\top}/\sqrt{2}$, with tangent projector $\mathbf{T}_{\text{soc}} \triangleq \mathbf{I} - \mathbf{n}\mathbf{n}^{\top}$. Direct substitution shows $\mathbf{Q}_{\text{soc}}\mathbf{n} = \mathbf{0}$, while $(\mathbf{q}, 1)^{\top}/\sqrt{2}$ is an eigenvector with eigenvalue 1 and every angular tangent vector $(\mathbf{h}, 0)$ with $\mathbf{q}^{\top}\mathbf{h} = 0$ is an eigenvector with eigenvalue κ. Hence $\mathbf{Q}_{\text{soc}} = \mathbf{T}_{\text{soc}}\mathbf{R}\mathbf{T}_{\text{soc}}$ for a self-adjoint $\mathbf{R}$, and $\mathbf{Q}_{\text{soc}} \neq \mathbf{T}_{\text{soc}}$ unless $\kappa = 1$. Since $-\rho < b < \rho$, $\kappa = (\rho + b)/(2\rho) \in (0, 1)$, so the flat subspace $\mathcal{F}$ is the ruling $\operatorname{span}\{(\mathbf{q}, 1)\}$ and $\mathbf{N} = \mathbf{I} - \mathbf{F}$ has rank n: the normal $\mathbf{n}$ and the $n-1$ angular directions. Partial smoothness of $\mathcal{C}_2$ at boundary points relative to this manifold is (Liang et al., 2017, Ex. 3.4). Theorem 1 then applies with this explicit $\mathbf{Q}$. For products of cones, the projector and its derivative are block diagonal, so the reduced solve decomposes over active boundary cones. □

Corollary 3 (Polyhedral)**.** *For polyhedral $\mathcal{C}(\mathbf{x}) = \{\mathbf{A}\mathbf{z} = \mathbf{b}, \mathbf{l} \le \mathbf{z} \le \mathbf{u}\}$, the box-active split gives coordinate indicators $\mathbf{T} = \operatorname{diag}(\mathbf{1}_{\mathcal{J}^c})$, $\mathbf{N} = \operatorname{diag}(\mathbf{1}_{\mathcal{J}})$ ($\mathcal{J}$ bound-active, $\mathcal{J}^c$ free), and $\mathbf{P}$ is the linear projector onto $\ker(\mathbf{A})$ from the affine constraint; the active-manifold tangent space is $\ker(\mathbf{A}) \cap \operatorname{range}\mathbf{T}$, with orthogonal projector $\mathbf{P} - \mathbf{P}\mathbf{N}\mathbf{P}_{\mathbf{N}\mathbf{N}}^{-1}\mathbf{N}\mathbf{P}$. The closed-form solve (9) becomes*

$$\mathbf{P}_{\mathbf{N}\mathbf{N}}\,\mathbf{v}_{\mathbf{N}} = (\mathbf{P}\mathbf{c})_{\mathbf{N}}, \qquad \left(\frac{\partial \ell}{\partial \mathbf{y}_{\text{raw}}}\right)_{\mathbf{F}} = (\mathbf{P}\mathbf{c})_{\mathbf{F}} - \mathbf{P}_{\mathbf{F}\mathbf{N}}\mathbf{v}_{\mathbf{N}}, \qquad \left(\frac{\partial \ell}{\partial \mathbf{y}_{\text{raw}}}\right)_{\mathbf{N}} = \mathbf{0}, \tag{18}$$

at cost $\mathcal{O}(|\mathcal{J}|^3 + |\mathcal{J}|\,|\mathcal{J}^c|)$ beyond one application of $\mathbf{P}$ ($\mathbf{P}$ cached): a single $|\mathcal{J}| \times |\mathcal{J}|$ Cholesky on the normal block and one $|\mathcal{J}^c| \times |\mathcal{J}|$ product on the tangent block.

Proof of Corollary 3. For the box $\mathcal{C}_2 = [\mathbf{l}, \mathbf{u}]$ at $\mathbf{y}^\star$ with bound-active set $\mathcal{J}$ and free set $\mathcal{J}^c$, the active manifold, near $\mathbf{y}^\star$ equal to $\{\mathbf{z} \in \mathbb{R}^d : z_j = y_j^\star\ \forall j \in \mathcal{J}\}$, has tangent space $T_{\mathcal{C}_{\mathcal{A},2}} = \operatorname{span}\{\mathbf{e}_j : j \notin \mathcal{J}\}$ and normal space $N_{\mathcal{C}_{\mathcal{A},2}} = \operatorname{span}\{\mathbf{e}_j : j \in \mathcal{J}\}$, with diagonal projectors

$\mathbf{T} = \mathrm{diag}(\mathbf{1}_{\mathcal{J}^c})$ and $\mathbf{N} = \mathrm{diag}(\mathbf{1}_{\mathcal{J}})$. In this basis, $\mathbf{P_{NT}}$ and $\mathbf{P_{NN}}$ become the sub-blocks $\mathbf{P_{NF}}$ and (principal) $\mathbf{P_{NN}}$ of the projector $\mathbf{P}$ onto $\ker(\mathbf{A})$. Substituting into (9) gives (18). The cost is a single Cholesky factorization of $\mathbf{P_{NN}} \in \mathbb{R}^{|\mathcal{J}|\times|\mathcal{J}|}$ ($\succ \mathbf{0}$ by Assumption D) plus the product $\mathbf{P_{FN}v_N}$, total $\mathcal{O}(|\mathcal{J}|^3 + |\mathcal{J}|\,|\mathcal{J}^c|)$. □

A3.2.3 Example: one active bound and one linear equality

We illustrate Theorem 1 on the simplest nontrivial polyhedral instance: the box $\mathcal{C}_2 = \{\mathbf{z} \in \mathbb{R}^d : -\mathbf{1} \le \mathbf{z} \le \mathbf{1}\}$ intersected with a single linear equality $\mathcal{C}_1 = \{\mathbf{z} : \mathbf{a}^\top \mathbf{z} = 0\}$ for $\mathbf{a} \in \mathbb{R}^d$, $\|\mathbf{a}\| = 1$. Suppose the projected output $\mathbf{y}^\star$ has its j-th coordinate saturated at the upper bound $y_j^\star = 1$ and all other coordinates strictly interior in the box. Throughout, $\mathbf{e}_j$ denotes the j-th standard basis vector and $a_j \triangleq \mathbf{a}^\top \mathbf{e}_j$.

Near $\mathbf{y}^\star$ the active manifold coincides with

$$\mathcal{C}_{\mathcal{A}}(\mathbf{x}) \;=\; \{\mathbf{z} \in \mathbb{R}^d : z_j = 1,\ \mathbf{a}^\top \mathbf{z} = 0\},$$

a $(d-2)$-dimensional affine slice. Only constraint j of the box is active and the equality is always active, so the codimension of the active manifold inside the ambient $\mathbb{R}^d$ is 2, and the codimension inside $\mathcal{C}_2$ (which is what $d_{\mathbf{N}} = \dim N_{\mathcal{C}_{\mathcal{A},2}}$ counts) is 1. The tangent and normal projectors of $\mathcal{C}_2$'s active manifold and the projector onto $T_{\mathcal{C}_{\mathcal{A},1}} = \ker(\mathbf{a}^\top)$ are

$$\mathbf{T} \;=\; \mathbf{I} - \mathbf{e}_j\mathbf{e}_j^\top, \qquad \mathbf{N} \;=\; \mathbf{e}_j\mathbf{e}_j^\top, \qquad \mathbf{P} \;=\; \mathbf{I} - \mathbf{a}\mathbf{a}^\top.$$

$\mathbf{T}$ "frees" all coordinates except the saturated one j; $\mathbf{N}$ pins coordinate j; $\mathbf{P}$ projects orthogonally to the equality direction $\mathbf{a}$.

The named sub-blocks of $\mathbf{P}$ in the $T \oplus N$ basis come out as

$$\mathbf{P_{NN}} \;=\; \mathbf{e}_j^\top \mathbf{P}\mathbf{e}_j \;=\; 1 - a_j^2 \;\in\; (0, 1], \qquad \mathbf{P_{NT}} \;=\; \mathbf{e}_j^\top \mathbf{P}\mathbf{T} \;=\; -a_j\big(\mathbf{a}^\top \mathbf{T}\big) \;=\; -a_j\big(\mathbf{a} - a_j\mathbf{e}_j\big)^\top.$$

The scalar $\mathbf{P_{NN}} = 1 - a_j^2$ is positive precisely when $|a_j| < 1$, i.e., the equality direction $\mathbf{a}$ does not coincide with the active coordinate direction $\mathbf{e}_j$ — this is the worked-example incarnation of Assumption D. Here it holds automatically: $|a_j| = 1$ would force $\mathbf{a} = \pm\mathbf{e}_j$, hence $\mathbf{a}^\top \mathbf{y}^\star = \pm y_j^\star = \pm 1 \neq 0$, contradicting $\mathbf{y}^\star \in \mathcal{C}_1$.

Decompose the cotangent as $\mathbf{c} = \mathbf{c_T} + c_j\mathbf{e}_j$ with $\mathbf{c_T} = \mathbf{T}\mathbf{c}$ supported off coordinate j, and write $\tilde{\mathbf{a}} \triangleq \mathbf{T}\mathbf{a} = \mathbf{a} - a_j\mathbf{e}_j$. The closed-form backward (9) reduces to

$$v_j \;=\; \frac{(\mathbf{P}\mathbf{c})_j}{1 - a_j^2} \;=\; \frac{c_j - a_j\,\mathbf{a}^\top \mathbf{c}}{1 - a_j^2},$$

$$\frac{\partial \ell}{\partial \mathbf{y}_{\text{raw}}} \;=\; (\mathbf{P}\mathbf{c})_{\mathbf{T}} - \mathbf{P_{TN}} v_j \;=\; \mathbf{c_T} - \frac{\mathbf{a}^\top \mathbf{c_T}}{1 - a_j^2}\,\tilde{\mathbf{a}},$$

where the second line uses $(\mathbf{P}\mathbf{c})_{\mathbf{T}} = \mathbf{c_T} - (\mathbf{a}^\top \mathbf{c})\tilde{\mathbf{a}}$ and $\mathbf{P_{TN}} = -a_j\tilde{\mathbf{a}}$; it is the orthogonal projection of $\mathbf{c}$ onto $\{\mathbf{z} : z_j = 0,\ \mathbf{a}^\top \mathbf{z} = 0\}$, as Corollary 1 predicts. The total cost is one length-d dot product $\mathbf{a}^\top \mathbf{c_T}$, one scalar division and one axpy — $\mathcal{O}(d)$ overall. By contrast, ambient implicit differentiation would factor a dense $d \times d$ matrix at $\mathcal{O}(d^3)$. At $d = 10^3$ this is a $\sim 10^6\times$ reduction; the gap grows with d.

A3.3 Proof of Theorem 2 (Finite-time identification)

Proof of Theorem 2. The proof has four steps. Step 1 shows that each block projection maps a whole ball around its argument at the solution into the block's active manifold. Step 2 combines the two blocks. Step 3 states the condition a solver must meet, and Step 4 bounds its number of iterations.

Fix a context $\mathbf{x}$ and drop it from the notation. By Step 1 of the proof of Theorem 1, the split (3) is unique and $\mathbf{n}_i \in \mathrm{relint}\, N_{\mathcal{C}_i}(\overline{\mathbf{y}})$. The arguments of the two block projections at the solution are

$$\mathbf{u}_1^\star \triangleq \overline{\mathbf{y}} + \mathbf{n}_1, \qquad \mathbf{u}_2^\star \triangleq \overline{\mathbf{y}} + t\mathbf{n}_2.$$

Step 1: identification is a property of the block projection. Take any sequence $\mathbf{u}^{(k)} \to \mathbf{u}_2^\star$ and let $\mathbf{p}^{(k)} \triangleq \Pi_{\mathcal{C}_2}(\mathbf{u}^{(k)})$. The projection $\Pi_{\mathcal{C}_2}$ is continuous near $\mathbf{u}_2^\star$ because $\mathcal{C}_2$ is prox-regular (Assumption B), so $\mathbf{p}^{(k)} \to \Pi_{\mathcal{C}_2}(\mathbf{u}_2^\star) = \overline{\mathbf{y}}$. The residual $\mathbf{u}^{(k)} - \mathbf{p}^{(k)}$ is a proximal normal to $\mathcal{C}_2$ at $\mathbf{p}^{(k)}$, and it converges to $t\mathbf{n}_2 \in \operatorname{relint} N_{\mathcal{C}_2}(\overline{\mathbf{y}})$. The indicator of $\mathcal{C}_2$ is prox-regular and partly smooth at $\overline{\mathbf{y}}$ relative to $\mathcal{C}_{\mathcal{A},2}$ (Hare & Lewis, 2004, Thm. 3.1). The identification theorem of Hare & Lewis (2004, Thm. 5.3) therefore gives K_2 with $\mathbf{p}^{(k)} \in \mathcal{C}_{\mathcal{A},2}$ for all $k \geq K_2$.

This holds for every sequence, so it holds on a ball: there is $\varrho_2 > 0$ with

$$\Pi_{\mathcal{C}_2}(\mathbf{u}) \in \mathcal{C}_{\mathcal{A},2} \qquad \text{whenever } \|\mathbf{u} - \mathbf{u}_2^\star\| < \varrho_2.$$

Otherwise some sequence $\mathbf{u}^{(k)} \to \mathbf{u}_2^\star$ would have $\Pi_{\mathcal{C}_2}(\mathbf{u}^{(k)}) \notin \mathcal{C}_{\mathcal{A},2}$ for every k, which contradicts the previous paragraph. The same argument at $\mathbf{u}_1^\star$ gives ϱ_1. Take $\varrho = \min(\varrho_1, \varrho_2)$.

Step 2: the intersection. By Assumption C(i) the block normals are linearly independent, so the two block active manifolds are transversal. At a point of strict complementarity the active manifold of $\mathcal{C}_1 \cap \mathcal{C}_2$ is then $\mathcal{C}_{\mathcal{A},1} \cap \mathcal{C}_{\mathcal{A},2}$ (Liang et al., 2017, Prop. 4.5). Consider a solver whose block-projection arguments satisfy $\mathbf{u}_1^{(k)} \to \mathbf{u}_1^\star$ and $\mathbf{u}_2^{(k)} \to \mathbf{u}_2^\star$. Step 1 gives $\Pi_{\mathcal{C}_1}(\mathbf{u}_1^{(k)}) \in \mathcal{C}_{\mathcal{A},1}$ for $k \geq K_1$ and $\Pi_{\mathcal{C}_2}(\mathbf{u}_2^{(k)}) \in \mathcal{C}_{\mathcal{A},2}$ for $k \geq K_2$. From $K \triangleq \max(K_1, K_2)$ on, both block manifolds, and with them the active manifold $\mathcal{C}_{\mathcal{A},1} \cap \mathcal{C}_{\mathcal{A},2}$ of $\mathcal{C}$ at $\overline{\mathbf{y}}$, are read off exactly. The iterate $\Pi_{\mathcal{C}_2}(\mathbf{u}_2^{(k)})$ need not lie in $\mathcal{C}_1$ before convergence: the solver identifies the manifold before it reaches a feasible point on it.

Step 3: convergent solvers. Let the solver be a fixed-point map $\mathcal{T}$ whose fixed point $\mathbf{s}^\star$ returns $\overline{\mathbf{y}}$, and index each block-projection argument by the state $\mathbf{s}^{(k)}$ it is formed from. Suppose that for a constant L_u

$$\|\mathbf{u}_i^{(k)} - \mathbf{u}_i^\star\| \;\leq\; L_u \,\|\mathbf{s}^{(k)} - \mathbf{s}^\star\|, \qquad i = 1, 2. \tag{S}$$

Then $\mathbf{s}^{(k)} \to \mathbf{s}^\star$ implies $\mathbf{u}_i^{(k)} \to \mathbf{u}_i^\star$, and Step 2 applies. Condition (S) concerns only how the solver forms the arguments of its block projections, not the constraint set.

Step 4: iteration bound. Suppose in addition that the solver converges linearly, $\|\mathbf{s}^{(k)} - \mathbf{s}^\star\| \leq (1-\delta)^k \|\mathbf{s}^{(0)} - \mathbf{s}^\star\|$, with the rate $1-\delta$ of Assumption G. By (S), both arguments formed from $\mathbf{s}^{(k)}$ lie within ϱ of Step 1 as soon as $\|\mathbf{s}^{(k)} - \mathbf{s}^\star\| < \varrho / L_u$. Because $(1-\delta)^k \leq e^{-\delta k}$, this holds whenever $k > \delta^{-1} \ln\big(L_u \|\mathbf{s}^{(0)} - \mathbf{s}^\star\| / \varrho\big)$. Counting the one extra step at which an argument formed from $\mathbf{s}^{(k)}$ may be evaluated,

$$K \;\leq\; 2 + \max\Big\{0, \Big\lfloor \delta^{-1} \ln\big(L_u \|\mathbf{s}^{(0)} - \mathbf{s}^\star\| / \varrho\big) \Big\rfloor\Big\} \;=\; \mathcal{O}\big(\delta^{-1} \log(\|\mathbf{s}^{(0)} - \mathbf{s}^\star\| / \varrho)\big). \qquad \square$$

A3.4 Proofs of the propositions on the face-restricted forward

For any solver map $\mathcal{T}$, the face-restricted map from (12) is

$$\mathcal{T}^F(\mathbf{s}, \mathbf{y}_{\text{raw}}; \mathbf{x}) \;=\; \mathbf{T}\,\mathcal{T}\big(\mathbf{T}\mathbf{s} + \mathbf{N}\mathbf{z}^B,\, \mathbf{y}_{\text{raw}};\, \mathbf{x}\big) + \mathbf{N}\mathbf{z}^B,$$

where $\mathbf{z}^B$ satisfies $\mathbf{N}\mathbf{z}^B = \mathbf{N}\mathbf{s}^\star$ (in practice $\mathbf{N}\mathbf{s}^{(K)}$ at the identification iterate K, which introduces an error $\mathcal{O}(\|\mathbf{N}(\mathbf{s}^{(K)} - \mathbf{s}^\star)\|)$).

Proposition 2 (Invariance and fixed point of $\mathcal{T}^F$)**.** *Let $\mathcal{S}_B \triangleq \{\mathbf{s} : \mathbf{N}\mathbf{s} = \mathbf{N}\mathbf{z}^B\}$. Then $\mathcal{T}^F(\mathbf{s}, \mathbf{y}_{\text{raw}}; \mathbf{x}) \in \mathcal{S}_B$ for every $\mathbf{s}$, and if $\mathbf{N}\mathbf{z}^B = \mathbf{N}\mathbf{s}^\star$ then $\mathbf{s}^\star \in \mathcal{S}_B$ is a fixed point of $\mathcal{T}^F$.*

Proof of Proposition 2. Since $\mathbf{N}\mathbf{T} = \mathbf{0}$ and $\mathbf{N}^2 = \mathbf{N}$, $\mathbf{N}\mathcal{T}^F(\mathbf{s}) = \mathbf{N}\mathbf{z}^B$. If $\mathbf{N}\mathbf{z}^B = \mathbf{N}\mathbf{s}^\star$ then $\mathbf{T}\mathbf{s}^\star + \mathbf{N}\mathbf{z}^B = \mathbf{s}^\star$, so $\mathcal{T}^F(\mathbf{s}^\star) = \mathbf{T}\mathcal{T}(\mathbf{s}^\star) + \mathbf{N}\mathbf{s}^\star = \mathbf{T}\mathbf{s}^\star + \mathbf{N}\mathbf{s}^\star = \mathbf{s}^\star$. $\square$

Proposition 3 (Local contraction of $\mathcal{T}^F$)**.** *Under Assumptions A–D and G, let $\mathbf{N}\mathbf{z}^B = \mathbf{N}\mathbf{s}^\star$, let $\mathcal{T}$ be differentiable at $\mathbf{s}^\star$ with Jacobian $\mathbf{J}_\mathcal{T}$, and put $\delta_{\mathcal{C}_\mathcal{A}} \triangleq 1 - \|\mathbf{T}\mathbf{J}_\mathcal{T}\mathbf{T}\|_2$. Then $\delta_{\mathcal{C}_\mathcal{A}} \geq \delta$, and for every $\epsilon > 0$ there is a neighborhood $\mathcal{U}$ of $\mathbf{s}^\star$ with $\|\mathcal{T}^F(\mathbf{s}) - \mathbf{s}^\star\| \leq (1 - \delta_{\mathcal{C}_\mathcal{A}} + \epsilon)\|\mathbf{s} - \mathbf{s}^\star\|$ for all $\mathbf{s} \in \mathcal{S}_B \cap \mathcal{U}$; if $\mathcal{T}$ is affine near $\mathbf{s}^\star$ (polyhedral blocks) one may take $\epsilon = 0$.*

Proof of Proposition 3. For $\mathbf{s} \in \mathcal{S}_B$, $\mathbf{T}\mathbf{s} + \mathbf{N}\mathbf{z}^B = \mathbf{s}$ and $\mathbf{s} - \mathbf{s}^\star \in \operatorname{range}\mathbf{T}$, so $\mathcal{T}^F(\mathbf{s}) - \mathbf{s}^\star = \mathbf{T}\big(\mathcal{T}(\mathbf{s}) - \mathcal{T}(\mathbf{s}^\star)\big) = \mathbf{T}\mathbf{J}_{\mathcal{T}}\mathbf{T}(\mathbf{s} - \mathbf{s}^\star) + o(\|\mathbf{s} - \mathbf{s}^\star\|)$, with no remainder when $\mathcal{T}$ is affine near $\mathbf{s}^\star$. The contraction in Assumption G gives $\|\mathbf{J}_{\mathcal{T}}\mathbf{h}\| \le (1-\delta)\|\mathbf{h}\|$ for every direction $\mathbf{h}$, hence $\|\mathbf{T}\mathbf{J}_{\mathcal{T}}\mathbf{T}\|_2 \le 1 - \delta$. □

A3.5 Pooled-EMA update

The pooled exponential-moving-average update of Algorithm 2 is

$$\overline{\mathbf{s}}^{(t+1)} = (1-\alpha)\overline{\mathbf{s}}^{(t)} + \alpha\,\tfrac{1}{B}\sum_{i\in\mathcal{B}^{(t)}} \mathbf{s}_i^{(t)}, \qquad \overline{\mathbf{v}}^{(t+1)} = (1-\alpha)\overline{\mathbf{v}}^{(t)} + \alpha\,\tfrac{1}{B}\sum_{i\in\mathcal{B}^{(t)}} \mathbf{v}_i^{(t)}, \tag{19}$$

with rate $\alpha \in (0,1]$ and per-sample fixed-point and adjoint $\mathbf{s}_i^{(t)}, \mathbf{v}_i^{(t)}$ produced by Algorithm 2 on batch $\mathcal{B}^{(t)}$. This replaces the $\Theta(Nd)$ memory of per-sample warm-start state with constant $\Theta(d)$ memory across the dataset.

Remark 1 (Partition used by the backward)**.** *Before the forward outputs have identified the active manifolds, the candidate partition $\mathcal{C}_{\mathcal{A}}^{(t)}(\mathbf{x})$ may differ from the true one, and a reduced solve on a wrong partition differentiates the wrong manifold. Theorems 3 and 4 do not assume that the partition read off an inexact forward state is correct: a wrong partition enters only through the calmness constant L_g of the read-out $\mathbf{g}_{\mathbf{x}}$, which must then also bound the resulting gradient error on the ball of radius D. Under the uniform-radius condition in the last part of Theorem 4, the partition is correct at every step once T is large enough. Both theorems are stated for the pinned values $\mathbf{N}\mathbf{z}^B = \mathbf{N}\mathbf{s}^\star$; the effect of the practical choice $\mathbf{N}\mathbf{s}^{(K)}$ is not quantified.*

A3.6 Proof of Theorem 3

Let Φ be the objective of (PCO) with $\mathbf{y}_{\boldsymbol{\theta}} = \Pi_{\mathcal{C}(\cdot)} \circ f_{\boldsymbol{\theta}}$, $\Phi^\star = \inf_{\boldsymbol{\theta}} \Phi$, and let $\mathbf{s}^\star(\boldsymbol{\theta}; \mathbf{x})$ and $\mathbf{g}_{\mathbf{x}}(\mathbf{s})$ denote the fixed point of the forward solver and the gradient read-out of Algorithm 1 computed from a forward state $\mathbf{s}$.

Theorem (Theorem 3, formal)**.** *Suppose Assumptions A–D hold at the training contexts, $f_{\boldsymbol{\theta}}$ and ℓ are C^1, Φ has an L_Φ-Lipschitz gradient on $\mathcal{N}_\circ$, and the forward map $\mathcal{T}$ of Algorithm 1 contracts toward its fixed point, $\|\mathcal{T}(\mathbf{s}) - \mathbf{s}^\star\| \le (1-\delta)\|\mathbf{s} - \mathbf{s}^\star\|$ whenever $\|\mathbf{s} - \mathbf{s}^\star\| \le D \triangleq 4\sup\|\mathbf{s}^\star\|$ (supremum over the training contexts and $\boldsymbol{\theta} \in \mathcal{N}_\circ$). Suppose also that the iterates $\boldsymbol{\theta}^{(\tau)}$ and the segments between them lie in $\mathcal{N}_\circ$, the backward solve is exact, and $\|\mathbf{g}_{\mathbf{x}}(\mathbf{s}) - \mathbf{g}_{\mathbf{x}}(\mathbf{s}^\star)\| \le L_g\|\mathbf{s} - \mathbf{s}^\star\|$ whenever $\|\mathbf{s} - \mathbf{s}^\star\| \le D$. With $\eta = 1/L_\Phi$ and $K \ge \delta^{-1}\max\{\ln 2,\, \ln(L_g D\sqrt{T})\}$,*

$$\min_{\tau<T}\|\nabla\Phi(\boldsymbol{\theta}^{(\tau)})\|^2 \;\le\; \frac{1}{T}\sum_{\tau=0}^{T-1}\|\nabla\Phi(\boldsymbol{\theta}^{(\tau)})\|^2 \;\le\; \frac{2L_\Phi\big(\Phi(\boldsymbol{\theta}^{(0)}) - \Phi^\star\big) + 1}{T}. \tag{20}$$

Proof of Theorem 3. The argument is the descent lemma for gradient descent with a biased gradient. Step 1 shows that the warm start stays bounded. Step 2 uses this to show that K forward steps make the gradient bias at most $T^{-1/2}$. Step 3 shows that a bias of this size adds at most $1/T$ to the rate.

Setup. The smoothness and contraction conditions of Theorem 3 are Assumptions E and G restricted to the training contexts, with $\overline{\mathbf{s}}^{(0)} = \mathbf{0}$ (the initialization of Algorithm 1) and hence $D = 4R_s$. Write $\mathbf{s}_i^{(t)}$ for the forward state of context $\mathbf{x}_i$ at step t (the τ of the statement), $\mathbf{s}_i^\star \triangleq \mathbf{s}^\star(\boldsymbol{\theta}^{(t)}; \mathbf{x}_i)$, and

$$\mathbf{g}^{(t)} \triangleq \frac{1}{N}\sum_i \mathbf{g}_{\mathbf{x}_i}(\mathbf{s}_i^{(t)}), \qquad \mathbf{b}_t \triangleq \mathbf{g}^{(t)} - \nabla\Phi(\boldsymbol{\theta}^{(t)}).$$

Theorem 1 and the chain rule (1), applied to each of the N terms of (PCO), give $\nabla\Phi(\boldsymbol{\theta}^{(t)}) = \frac{1}{N}\sum_i \mathbf{g}_{\mathbf{x}_i}(\mathbf{s}_i^\star)$.

Step 1: bounded warm starts. Let $M \triangleq 3R_s$, so that $D = M + R_s$. Suppose $\|\overline{\mathbf{s}}^{(t)}\| \le M$. Then $\|\overline{\mathbf{s}}^{(t)} - \mathbf{s}_i^\star\| \le D$, every inner iterate stays within distance D of $\mathbf{s}_i^\star$, and K applications of the contraction of Assumption G give

$$\|\mathbf{s}_i^{(t)} - \mathbf{s}_i^\star\| \le (1-\delta)^K D \le D/2,$$

because $K \geq \delta^{-1} \ln 2$. Hence $\|\mathbf{s}_i^{(t)}\| \leq R_s + D/2 = 3R_s = M$, and the average $\bar{\mathbf{s}}^{(t+1)} = \frac{1}{N}\sum_i \mathbf{s}_i^{(t)}$ also satisfies $\|\bar{\mathbf{s}}^{(t+1)}\| \leq M$. By induction the bound holds for all t.

Step 2: bias. By the Lipschitz read-out and $K \geq \delta^{-1} \ln(L_g D\sqrt{T})$,

$$\|\mathbf{b}_t\| \leq \frac{1}{N}\sum_i L_g \|\mathbf{s}_i^{(t)} - \mathbf{s}_i^\star\| \leq L_g D(1-\delta)^K \leq L_g D\, e^{-\delta K} \leq T^{-1/2}.$$

Step 3: descent. The gradient $\nabla\Phi$ is L_Φ-Lipschitz on the segment $[\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)}] \subset \mathcal{N}_\circ$, and $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \mathbf{g}^{(t)}$ with $\eta = 1/L_\Phi$. The descent lemma gives

$$\Phi(\boldsymbol{\theta}^{(t+1)}) \leq \Phi(\boldsymbol{\theta}^{(t)}) - \eta\langle \nabla\Phi, \nabla\Phi + \mathbf{b}_t\rangle + \tfrac{\eta}{2}\|\nabla\Phi + \mathbf{b}_t\|^2 = \Phi(\boldsymbol{\theta}^{(t)}) - \tfrac{1}{2L_\Phi}\|\nabla\Phi(\boldsymbol{\theta}^{(t)})\|^2 + \tfrac{1}{2L_\Phi}\|\mathbf{b}_t\|^2.$$

Summing over $t < T$ and using $\Phi(\boldsymbol{\theta}^{(T)}) \geq \Phi^\star$,

$$\frac{1}{T}\sum_{t<T} \|\nabla\Phi(\boldsymbol{\theta}^{(t)})\|^2 \leq \frac{2L_\Phi(\Phi(\boldsymbol{\theta}^{(0)}) - \Phi^\star)}{T} + \frac{1}{T}\sum_{t<T}\|\mathbf{b}_t\|^2,$$

and $\|\mathbf{b}_t\|^2 \leq 1/T$ from Step 2 yields (20).

An accepted face projection does not change the argument. Feasibility and the multiplier signs certify that it is the exact projection, so the context contributes the exact read-out (Theorem 1) with zero bias, and it adds no state to the pooled average, which stays bounded by M. □

A3.7 Proof of Theorem 4

Proof of Theorem 4. The proof follows the proof of Theorem 3, with conditional expectations in place of exact gradients, and adds an identification step. Let $\mathcal{F}_t$ be as in Assumption F, and let $M \triangleq \max\{\|\bar{\mathbf{s}}^{(0)}\|, 3R_s\}$, so that $D = M + R_s$. Write $\mathbf{s}_i^\star \triangleq \mathbf{s}^\star(\boldsymbol{\theta}^{(t)}; \mathbf{x}_i)$.

Step 1: bounded warm starts. Suppose $\|\bar{\mathbf{s}}^{(t)}\| \leq M$. Then $\|\bar{\mathbf{s}}^{(t)} - \mathbf{s}^\star(\boldsymbol{\theta}^{(t)}; \mathbf{x})\| \leq D$, and the contraction of Assumption G applies along the whole inner run. Since $(1-\delta)^{K_{\mathrm{dr}}} \leq \frac{1}{2}$,

$$\|\mathbf{s}_i^{(t)}\| \leq \|\mathbf{s}_i^\star\| + \tfrac{1}{2}\|\bar{\mathbf{s}}^{(t)} - \mathbf{s}_i^\star\| \leq \tfrac{1}{2}M + \tfrac{3}{2}R_s \leq M.$$

The pooled update (19) is a convex combination of $\bar{\mathbf{s}}^{(t)}$ and an average of the $\mathbf{s}_i^{(t)}$, so $\|\bar{\mathbf{s}}^{(t+1)}\| \leq M$. By induction this holds for all t.

Step 2: bias. Let

$$\mathbf{b}_t \triangleq \mathbb{E}[\mathbf{g}^{(t)} \mid \mathcal{F}_t] - \nabla\Phi(\boldsymbol{\theta}^{(t)}) = \mathbb{E}_{\mathbf{x}}\big[\mathbf{g}_{\mathbf{x}}(\mathbf{s}_{\mathbf{x}}^{(t)}) - \mathbf{g}_{\mathbf{x}}(\mathbf{s}^\star(\boldsymbol{\theta}^{(t)}; \mathbf{x}))\big],$$

where the second form uses the gradient representation of Assumption E. The choice of K_{dr} gives $(1-\delta)^{K_{\mathrm{dr}}} \leq e^{-\delta K_{\mathrm{dr}}} \leq \sqrt{\eta}$, so each forward output satisfies $\|\mathbf{s}_{\mathbf{x}}^{(t)} - \mathbf{s}^\star(\boldsymbol{\theta}^{(t)}; \mathbf{x})\| \leq \sqrt{\eta}\, D$, and the Lipschitz read-out gives $\|\mathbf{b}_t\|^2 \leq L_g^2 D^2 \eta$.

Step 3: descent. The descent lemma on the segment $[\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)}] \subset \mathcal{N}_\circ$ uses three bounds. First, Assumption F gives $\mathbb{E}[\|\mathbf{g}^{(t)}\|^2 | \mathcal{F}_t] \leq \|\nabla\Phi + \mathbf{b}_t\|^2 + \sigma^2/B$. Second, $-\eta\langle\nabla\Phi, \mathbf{b}_t\rangle \leq \frac{\eta}{2}\|\nabla\Phi\|^2 + \frac{\eta}{2}\|\mathbf{b}_t\|^2$. Third, $\eta \leq 1/(4L_\Phi)$ (guaranteed by $T \geq 16 L_\Phi^2 c_\eta^2$) gives $\frac{L_\Phi \eta^2}{2}\|\nabla\Phi + \mathbf{b}_t\|^2 \leq \frac{\eta}{4}(\|\nabla\Phi\|^2 + \|\mathbf{b}_t\|^2)$. Together with Step 2,

$$\mathbb{E}[\Phi(\boldsymbol{\theta}^{(t+1)}) \mid \mathcal{F}_t] \leq \Phi(\boldsymbol{\theta}^{(t)}) - \tfrac{\eta}{4}\|\nabla\Phi(\boldsymbol{\theta}^{(t)})\|^2 + \tfrac{3\eta}{4}L_g^2 D^2 \eta + \tfrac{L_\Phi \eta^2}{2}\tfrac{\sigma^2}{B}.$$

Step 4: telescoping. Take total expectations, sum over $t < T$, use $\Phi \geq \Phi^\star$, and divide by $\eta T/4$:

$$\frac{1}{T}\sum_{t<T}\mathbb{E}\|\nabla\Phi(\boldsymbol{\theta}^{(t)})\|^2 \leq \frac{4(\Phi(\boldsymbol{\theta}^{(0)}) - \Phi^\star)}{\eta T} + 3L_g^2 D^2 \eta + \frac{2L_\Phi \eta \sigma^2}{B}.$$

At $\eta = c_\eta T^{-1/2}$ this is (16). The bound on $\min_{t<T} \mathbb{E}\|\nabla\Phi(\boldsymbol{\theta}^{(t)})\|^2$ follows because the minimum is at most the average.

Step 5: identification. Under the additional hypotheses, apply Step 4 of the proof of Theorem 2 to each sample, starting from $\|\bar{\mathbf{s}}^{(t)} - \mathbf{s}^\star(\boldsymbol{\theta}^{(t)}; \mathbf{x})\| \le D$. The state before the last forward step satisfies

$$\|\mathbf{s}^{(K_{\mathrm{dr}}-1)} - \mathbf{s}^\star\| \le (1-\delta)^{K_{\mathrm{dr}}-1} D \le \sqrt{\eta}\, D/(1-\delta),$$

which is below $\varrho_{\min}/L_u$ once $T > \left(L_u D\sqrt{c_\eta}/((1-\delta)\varrho_{\min})\right)^4$. Both block-projection arguments of the last forward step then lie within $\varrho_{\min}$ of their limits, and Step 1 of that proof places the forward outputs on $\mathcal{C}_{\mathcal{A},i}(\mathbf{x})$. □

A4 Addendum to Section 4: Experimental details and further results

All training runs use a single NVIDIA A100 (80 GB) on a shared cluster. The software stack is JAX 0.9.2, Optax 0.2.6, Flax 0.12.5 with CUDA 12.1. ASPL forward and backward are pure-JAX implementations; Πnet is run from the upstream reference (Grontas et al., 2026), with reproduction verified on the $d = 50$ box-QP cell to within $\pm 2\%$ of reported loss.

A4.1 Parametric quadratic programming

We report two accuracy metrics. With $J(\mathbf{y}) = \mathbf{y}^\top \mathbf{H}\mathbf{y} + \mathbf{q}^\top \mathbf{y}$, the relative suboptimality is $RS \triangleq \max\big(0,\ (J(\hat{\mathbf{y}}) - J(\mathbf{y}^\star))/|J(\mathbf{y}^\star)|\big)$ and the worst-case constraint violation is $CV \triangleq \max\big(\|\mathbf{A}\hat{\mathbf{y}} - \mathbf{x}\|_\infty,\ \|\max(\mathbf{C}\hat{\mathbf{y}} - \mathbf{u}, \mathbf{0})\|_\infty\big)$.

We solve the parametric QP of Section 4.1. The polyhedral feasible set $\mathcal{C}(\mathbf{x}) = \{\mathbf{y} : \mathbf{A}\mathbf{y} = \mathbf{x},\ \mathbf{C}\mathbf{y} \le \mathbf{u}\}$ admits the affine ∩ box decomposition after the standard slack lift $\mathbf{C}\mathbf{y} + \boldsymbol{\sigma} = \mathbf{u}$, $\boldsymbol{\sigma} \ge \mathbf{0}$. Corollary 3 then differentiates the lifted projection in closed form: $\mathbf{T}$ selects free lifted coordinates, $\mathbf{N}$ selects bound-active slack coordinates, and $\mathbf{P}$ is the orthogonal projector onto the lifted affine nullspace.

We construct nine QP cells sharing ambient dimension ($d = 2000$), equality count ($n_{\mathrm{eq}} = 100$), and inequality count ($n_{\mathrm{ineq}} = 2000$), with controlled active-set ratio $|\mathcal{A}|/n_{\mathrm{ineq}} \in \{0.05, 0.10, \ldots, 0.45\}$ at the optimum. Each cell contains roughly 8,000 training instances. Cells are sampled by drawing $\mathbf{H} = \mathbf{M}\mathbf{M}^\top + 0.1\,\mathbf{I}_d$ with $M_{ij} \sim \mathcal{N}(0, 1/d)$ (a scaled Wishart matrix, shifted to be positive definite) and $\mathbf{q} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, and tuning the bound $\mathbf{u}$ so that the target number of inequalities is tight at the optimum. The recorded $\mathbf{y}^\star$ is the exact optimum with a uniquely identifiable active set, with KKT residual $\le 8\cdot 10^{-14}$, minimum active multiplier $\ge 6\cdot 10^{-3}$, and minimum inactive slack ≥ 1.91.

The backbone is a 4-layer MLP with $d_{\mathrm{model}} = 512$ and GeLU activations. We optimize with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 10^{-4}) at learning rate 10^{-2} under a cosine schedule with 200 warm-up steps, batch size 64, for 50 epochs ($\sim$ 6,200 steps per cell). The Douglas–Rachford forward uses $K_{\mathrm{dr}} = 200$ iterations with $\omega = 1$, and the pooled-EMA warm-start uses rate $\alpha = 0.05$.

Table 1 gives the batch-64 results of Section 4.1; we also repeat the sweep at batch size 2048, where the JAXOpt and DC3 columns are copied from Table 1 (not rerun). Table 3 reports relative suboptimality and total wall-clock at d=2000 across the active-set sweep; Figure 8 repeats the comparison at d=1000 for ASPL (CG) and Πnet. ASPL is faster than Πnet at every ratio. Πnet has the lower RS at the smallest ratios (≤ 0.15 at d=2000, ≤ 0.20 at d=1000) and ASPL from there on: Πnet's RS grows by a factor of about 33 across the d=2000 sweep and about 380 across the d=1000 sweep, while ASPL's varies by at most a factor of 3. Figure 7 shows the d=2000 comparison on the RS–CV and RS–time planes.

A4.1.1 Backward accuracy on the lifted QP set (Figure 9)

The feasible set is $\{\mathbf{w} = (\mathbf{y}, \boldsymbol{\sigma}) \in \mathbb{R}^{d+p} : \mathbf{A}\mathbf{y} = \mathbf{x},\ \mathbf{C}\mathbf{y} + \boldsymbol{\sigma} = \mathbf{u},\ \boldsymbol{\sigma} \ge \mathbf{0}\}$ with d=p=2000, n_{eq}=100, and $\mathbf{A}, \mathbf{C}$ i.i.d. Gaussian with entries of variance $1/d$, i.e. $\mathcal{C}_1$ is the affine set $\{\mathbf{M}\mathbf{w} = \mathbf{r}\}$ with $\mathbf{M} = \left[\begin{smallmatrix}\mathbf{A} & \mathbf{0}\\ \mathbf{C} & \mathbf{I}\end{smallmatrix}\right]$ and $\mathcal{C}_2 = \{\boldsymbol{\sigma} \ge \mathbf{0}\}$. For each instance we draw $\mathbf{y}^\star \sim \mathcal{U}[-1, 1]^d$, a tight set

Table 3: Parametric QP with d=2000 and batch size 2048 across active-set ratios. Relative suboptimality (RS) and total training wall-clock (s). All methods reach CV $\leq 10^{-4}$ except DC3, which fails the hard-feasibility bar at every ratio. JAXOpt and DC3 entries are copied from Table 1. Best per metric in **bold**.

ratio	ASPL (CG) *RS* / Time	ΠNet *RS* / Time	JAXOpt *RS* / Time	DC3 *RS* / Time
0.05	3.493e-4 / **63**	2.830e-4 / 74	**1.880e-4** / 7886	6.084e+1 / 521
0.10	3.220e-4 / **66**	2.774e-4 / 79	**2.623e-4** / 8217	1.160e+2 / 535
0.15	3.350e-4 / **67**	**3.076e-4** / 84	3.208e-4 / 8135	1.578e+2 / 530
0.20	**3.354e-4** / **69**	4.462e-4 / 82	4.943e-4 / 7938	1.976e+2 / 519
0.25	**3.321e-4** / **72**	6.819e-4 / 83	6.798e-4 / 8285	2.398e+2 / 521
0.30	**3.485e-4** / **73**	1.276e-3 / 84	1.316e-3 / 9560	2.820e+2 / 532
0.35	**2.981e-4** / **74**	2.590e-3 / 87	6.573e-4 / 10630	3.334e+2 / 527
0.40	**2.858e-4** / **76**	4.764e-3 / 89	1.347e-3 / 11547	3.758e+2 / 532
0.45	**2.646e-4** / **76**	9.420e-3 / 89	1.025e-3 / 14049	4.284e+2 / 528

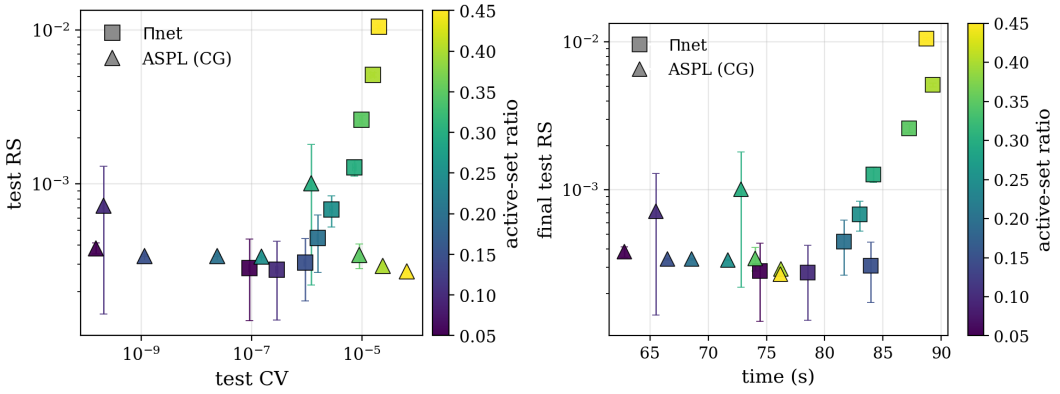

Figure 7: QP Pareto frontiers across the active-set sweep for batch size 2048. Left: test RS vs. test CV. Right: test RS vs. training wall-clock time.

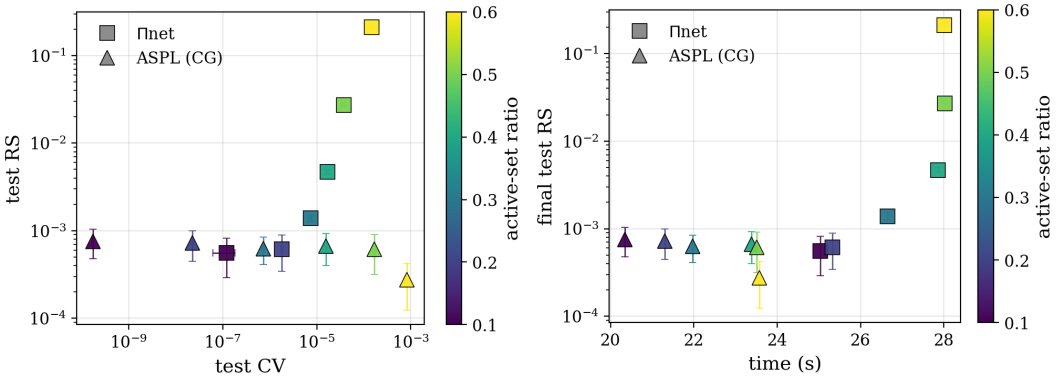

Figure 8: QP Pareto frontiers across the active-set sweep at $d = 1000$ and batch size 2048. Left: test RS vs. test CV. Right: test RS vs. training wall-clock time.

$\mathcal{A}$ of $\mathrm{round}((|\mathcal{A}|/p)\,p)$ inequalities with $\sigma^\star_{\mathcal{A}} = 0$ and $\sigma^\star_i \sim \mathcal{U}[0.2, 2]$ otherwise, set $\mathbf{r} = \mathbf{M}\mathbf{w}^\star$, and $\mathbf{y}_{\text{raw}} = \mathbf{w}^\star + \mathbf{M}^\top\boldsymbol{\nu} + \mathbf{n}_2$ with $\boldsymbol{\nu} \sim \mathcal{N}(\mathbf{0}, 0.09\,\mathbf{I})$ and $(\mathbf{n}_2)_{\sigma_i} \sim -\mathcal{U}[0.05, 1]$ for $i \in \mathcal{A}$ (zero elsewhere), so that $\Pi_{\mathcal{C}}(\mathbf{y}_{\text{raw}}) = \mathbf{w}^\star$ with active set $\mathcal{A}$ and strict-complementarity margin at least 0.05. The loss cotangent is $\mathbf{c} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The forward is the DR iteration (8) with ω=1.7 started at $\mathbf{s}^{(0)} = \mathbf{y}_{\text{raw}}$; the identified set at iteration k is the set of clipped slack coordinates. ASPL applies (18) on that set (Cholesky of $\mathbf{P}_{\mathbf{NN}}$ with the cached projector $\mathbf{P}$ onto $\ker \mathbf{M}$). The ambient baselines implicitly differentiate the DR fixed point at the same state, solving $(\mathbf{I} - D_{\mathbf{s}}\mathcal{T})^\top \mathbf{v} = \mathbf{r}_{\mathbf{c}}$, with $\mathbf{r}_{\mathbf{c}} = \frac{1}{2}\mathbf{Q}\mathbf{c}$ for the cotangent $\mathbf{c}$, with BiCGSTAB (25 iterations and tolerance 10^{-5} as in the

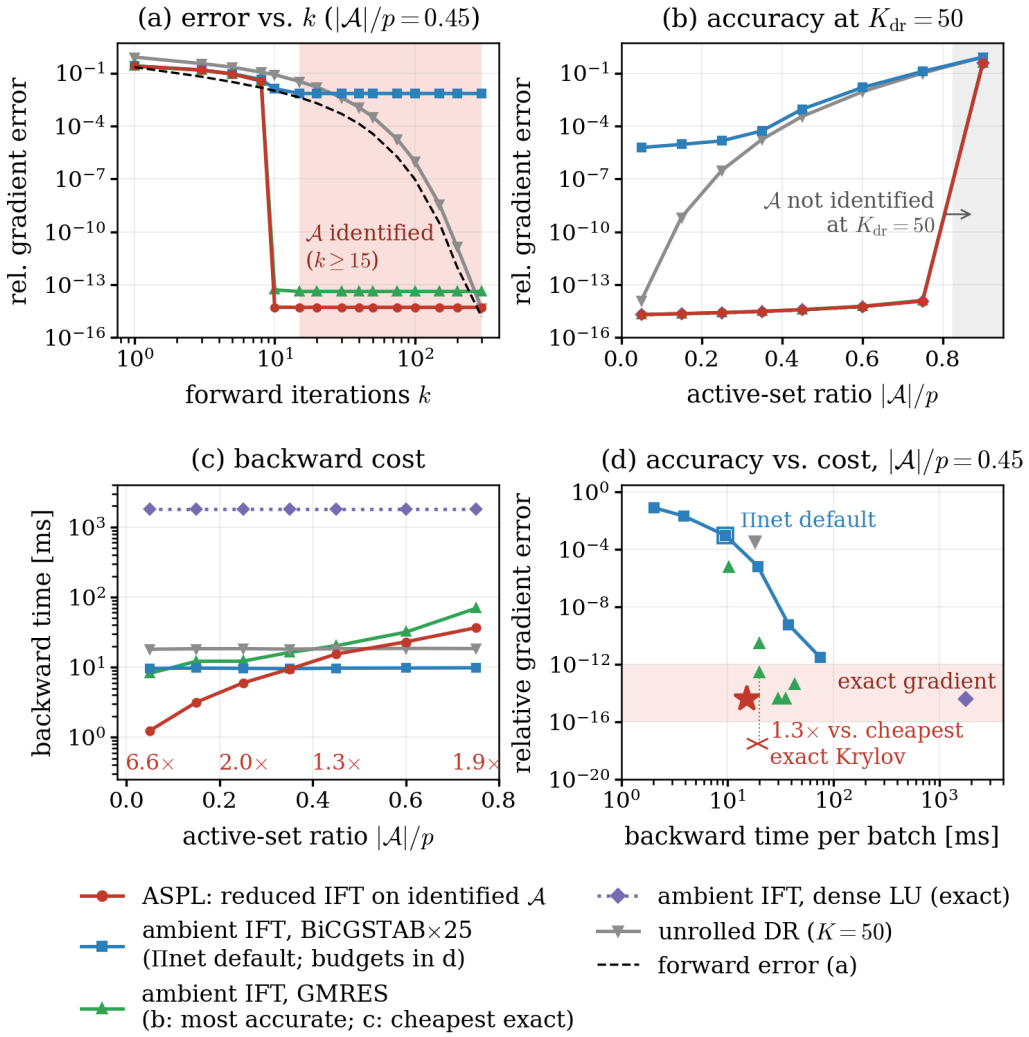

Figure 9: **Theorem 2 in detail.** Lifted QP feasible set of Section 4.1 (d=p=2000, ambient dimension 4000) with known projection. **(a)** Error vs. forward iterations at $|\mathcal{A}|/p$=0.45 (medians over 24 instances, 3 seeds): once $\mathcal{A}$ is identified ($k \geq 15$, shaded) the ASPL gradient matches the reference to $\approx 10^{-14}$ although the forward has not converged (dashed). **(b)** Error at K_{dr}=50: ASPL matches exact ambient IFT (dense LU, converged GMRES) wherever the forward has identified $\mathcal{A}$, while ΠnetQP's BiCGSTAB×25 degrades by four orders of magnitude as the ratio grows. **(c)** Backward time (batch 64, one A100); labels: ASPL's speedup over the cheapest Krylov configuration reaching 10^{-12} accuracy. **(d)** Accuracy vs. cost at $|\mathcal{A}|/p$=0.45 as each baseline's budget varies. Panels (b)–(d): 64 instances from one seed.

ΠnetQP configuration, or larger budgets), GMRES, or a dense LU of the analytic 4000×4000 matrix; unrolling back-propagates through K DR steps. All methods share the forward state, the cached factorization of $\mathbf{M}\mathbf{M}^\top$, and the cached projector $\mathbf{P}$ onto $\ker \mathbf{M}$ (precomputed once per affine block in 4 ms and 21 ms, excluded from all timings). ASPL pads the identified bound set to the batch maximum rounded up to a multiple of 64, computed from the forward's own identification. BiCGSTAB is a breakdown-guarded fixed-budget implementation; GMRES uses restarts 10–50 with relative tolerance 10^{-12}. The reference gradient is the orthogonal projection of $\mathbf{c}$ onto $\ker \mathbf{M} \cap \{\mathbf{w} : \sigma_{\mathcal{A}} = 0\}$ computed from the true active set by the free-coordinate formula $\mathbf{c}_F - \mathbf{M}_F^\top (\mathbf{M}_F \mathbf{M}_F^\top)^{-1} \mathbf{M}_F \mathbf{c}_F$ (panels b–d), or by dense implicit differentiation at the fully converged fixed point (3000 DR iterations; panel a); the two references agree to $2 \cdot 10^{-14}$ wherever the active set is identified. Timings are

medians of 7 repetitions of the jitted backward (JAX, float64, one A100 80GB, batch 64), excluding compilation.

A4.2 Second-order cone programs with many cones

Apart from the collision cones of motion planning, the sets in Section 4 are polyhedral. This benchmark tests the curved case of Corollary 2 with many cones, and a varying share of them active. Each instance is

$$\min_{\mathbf{y}_1} \ \mathbf{c}^\top \mathbf{y}_1 \quad \text{s.t.} \quad \mathbf{A}\mathbf{y}_1 + \mathbf{y}_2 = \mathbf{b}(\mathbf{x}), \ \ \mathbf{y}_2 \in \mathcal{K} = \mathcal{K}_{\text{soc}}^1 \times \cdots \times \mathcal{K}_{\text{soc}}^{50}, \tag{21}$$

with 50 Lorentz cones of dimension 10 (m=500 slack coordinates). Nine cells sweep the fraction of cones active at the optimum from 10% to 90%, with d_{y_1} equal to the number of active cones (5 to 45). An MLP maps $\mathbf{b}(\mathbf{x})$ to $\mathbf{y}_1$, and the layer projects its output. ASPL uses the conic dual solver of Appendix A2.2 with the dense $m \times m$ adjoint built from the per-cone Jacobians at the current iterate. We compare with ΠNet (Grontas et al., 2026) with a small regularization, which it needs to reach feasibility here, with the exact solvers SCS (O'Donoghue et al., 2016) (native and through CVXPY (Diamond & Boyd, 2016)), and with `cvxpylayers` (Agrawal et al., 2019) trained end to end. The reference is `Clarabel` (Goulart & Chen, 2026). A test output counts as feasible if its cone violation is at most 10^{-4}. The learned methods train for 6000 iterations on one A100; single-instance and batch (512 instances) inference times are measured on the same GPU, and the exact solvers run on CPU.

Table 4: SOCP with 50 cones, ranges over the nine active-cone ratios (10%–90%). Feasible: share of test outputs with cone violation $\leq 10^{-4}$. Train: training time of the learned methods. Times in ms per single instance and per batch of 512.

Method	Feasible (%)	Train (s)	Single (ms)	Batch (ms)
ASPL (conic dual)	100	213–218	95–98	265–279
ΠNet, regularized	92–100	269–291	86–90	434–461
SCS	100	–	8–42	4,354–24,025
CVXPY (SCS)	100	–	13–93	6,545–59,801
`cvxpylayers`, trained	0–1	3,532–10,624	912–13,409	4,197–17,773

ASPL is feasible on every test instance at every ratio, and its relative suboptimality differs from that of SCS by at most 1.5×10^{-3}. Regularized ΠNet reaches the same objective but misses feasibility on up to 8% of the instances. ASPL needs about a quarter less training time than ΠNet and projects a batch about 1.6× faster, and its time stays flat as the share of active cones grows from 10% to 90%. The exact solvers are feasible but take seconds per batch, and `cvxpylayers` trained end to end does not reach feasibility within its budget.

A4.3 Multi-robot motion planning

Following the multi-robot trajectory-generation setting of ΠNet (Grontas et al., 2026, §6.3–6.4), N agents traverse H steps with position $\mathbf{y}_t^i \in \mathbb{R}^2$, velocity $\mathbf{v}_t^i \in \mathbb{R}^2$, and control $\mathbf{u}_t^i \in \mathbb{R}^2$. The decision variable $\mathbf{z} = (\{\mathbf{y}_t^i\}_{t=0}^H, \{\mathbf{v}_t^i\}_{t=0}^H, \{\mathbf{u}_t^i\}_{t=0}^{H-1}) \in \mathbb{R}^{N(6H+4)}$ minimizes a reach-and-effort cost subject to dynamics, joint/velocity/acceleration/jerk limits, and pairwise collision avoidance:

$$\begin{aligned} \min_{\mathbf{z}} \ & \sum_{i,t} \|\mathbf{u}_t^i\|^2 + \lambda_g \sum_i \|\mathbf{y}_H^i - \mathbf{g}^i\|^2 + \lambda_c \, \Phi_{\text{cov}}(\{\mathbf{y}^i\}; \mathbf{x}) \\ \text{s.t.} \ & \mathbf{y}_{t+1}^i = \mathbf{y}_t^i + \Delta t \, \mathbf{v}_t^i, \ \ \mathbf{v}_{t+1}^i = \mathbf{v}_t^i + \Delta t \, \mathbf{u}_t^i, \ \ (\mathbf{y}_0^i, \mathbf{v}_0^i) \text{ given}, \\ & \mathbf{l} \leq \mathbf{y}_t^i \leq \mathbf{u}, \ \ |\mathbf{v}_t^i| \leq \mathbf{v}_{\max}, \ \ |\mathbf{u}_t^i| \leq \mathbf{u}_{\max}, \ \ |\mathbf{u}_{t+1}^i - \mathbf{u}_t^i| \leq \mathbf{j}_{\max}, \\ & \|\mathbf{y}_t^i - \mathbf{y}_t^j\| \geq d_{\text{safe}} \ \ \forall i \neq j, \ \ t = 0, \ldots, H. \end{aligned} \tag{22}$$

Box, velocity, acceleration, and jerk limits stack into affine inequalities through the finite-difference operators D_1, D_2, D_3. Dynamics fold into an affine equality. The pairwise distance constraint is nonconvex; we linearize it around the previous iterate as $\mathbf{n}_{ij}^\top(\mathbf{y}_t^i - \mathbf{y}_t^j) \geq d_{\text{safe}}$, or lift it to the Lorentz inequality $\|\Delta\mathbf{y}_t^{ij} - d_{\text{safe}}\mathbf{n}_{ij}\|_2 \leq \mathbf{n}_{ij}^\top \Delta\mathbf{y}_t^{ij}$. The composite is polyhedral ∩ SOC, and Theorem 1 applies block-by-block via Corollaries 2 and 3.

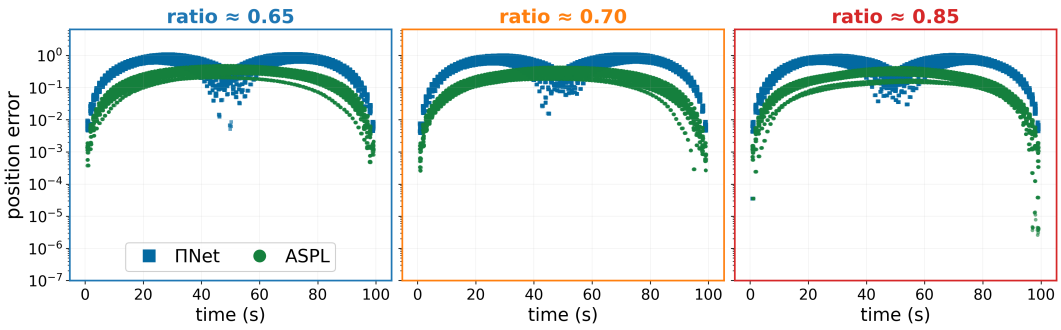

Figure 10: Per-(instance, step) position error for ΠNet and ASPL, evaluated across 16 robots over a horizon of 100 steps at active-set ratios 0.65, 0.70, 0.85 (left to right). *Position error* is the per-step Euclidean distance to the ground-truth trajectory. At all three ratios, ASPL's error is below ΠNet's over most of the horizon; near mid-horizon the two overlap.

The benchmark has three sizes (`r5h25`: N=5, H=25, d=770; `r8h50`: N=8, H=50, d=2432; `r16h100`: N=16, H=100, d=9664), each instantiated at nine controlled active-set ratios $|\mathcal{A}|/d \in \{0.10, 0.20, 0.30, 0.40, 0.50, 0.60, 0.65, 0.70, 0.85\}$ (denoted `r1`, `r2`, …, `r9`). For `r16h100` we report only the per-step position error at `r7`–`r9` (Figure 10).

Both methods train for 50 epochs, learning rate 3×10^{-4}, two-layer projected MLP with 400 features. ΠNet: $K_{\mathrm{dr}} = 200$, $\omega = 1.7$, $n_{\mathrm{iter,bwd}} = 200$, batch size 256 (the reference implementation's default). ASPL: $K_{\mathrm{dr}} = 200$, $\omega = 1.7$, batch size 512, both ASPL (Chol) (direct Cholesky on the $d_{\mathbf{N}} \times d_{\mathbf{N}}$ normal block) and ASPL (CG) (conjugate gradient) variants of Theorem 1. On `r16h100` the gap in per-step position error is under one order of magnitude (Figure 10).

A4.4 Inverse kinematics on real teleoperation logs

Differential inverse kinematics is a parametric projection in the sense of Section 2, and every datum of the problem is a parameter. Given the measured configuration $\mathbf{q}_t$ and the operator's target, a frame asks for the joint velocity closest to a raw command $\mathbf{v}_t^{\mathrm{raw}}$ (the output of a policy or of a retargeting stage) that realizes the task twist and respects the limits: $\mathbf{J}(\mathbf{q}_t)\mathbf{v}_t = \mathbf{e}_t$ and $\mathbf{v}_{\min}(\mathbf{q}_t) \le \mathbf{v}_t \le \mathbf{v}_{\max}(\mathbf{q}_t)$, where the box carries both the velocity limits and the joint-position limits divided by the timestep. Policies emit *chunks* of T frames, and smoothness is a constraint on the chunk, not on a frame: $|\mathbf{v}_{t+1} - \mathbf{v}_t| \le a_{\max}\Delta t$. The layer is therefore the projection of the raw chunk $\mathbf{V}^{\mathrm{raw}} \in \mathbb{R}^{n_{\mathrm{dof}}T}$ onto $\mathcal{C}(\mathbf{x})$ with $\mathbf{x} = (\mathbf{J}_{1:T}, \mathbf{e}_{1:T}, \mathbf{v}_{\min,1:T}, \mathbf{v}_{\max,1:T}, a_{\max})$: block-diagonal equalities, a box, and a banded coupling. We lift the coupling into the equality block with $\mathbf{w}_t = \mathbf{v}_{t+1} - \mathbf{v}_t$, so both proximal maps of the splitting are closed form (one reduced affine solve whose $T \times T$ chain factor is shared by every joint, and a clip), and the identified face is solved exactly by the reduced system of Theorem 1: the box rows are unit vectors and drop out, leaving a Cholesky on the equality rows plus the few active coupling rows. The layer certifies its output; the 2–5% of chunks where the operator's target jumped (the recorded robot itself misses these targets by 0.5–2.5 rad/s) are re-identified with an interior-point pass on that sub-batch only, and finished by the same exact face solve.

All instances come from real teleoperation logs at 30 Hz, with targets given by forward kinematics of the recorded joint commands and raw commands equal to the recorded joint velocity plus policy-like noise (0.3σ per coordinate). (i) The Dexmate Vega-1 from the tactile teleoperation corpus of Niu et al. (2026): torso and both arms, n_{dof}=17, 3D position task per hand. (ii) The Unitree G1 humanoid from Unitree's public whole-body teleoperation datasets (Unitree Robotics, 2026): all 29 joints plus the floating base (n_{dof}=35), position task for both wrists and a full 6D task for both feet (18 rows), on a standing table task and on a task that includes walking. Held-out episodes are never used for calibration.

We compare with QP solvers on the same chunk problem (`Clarabel` (Goulart & Chen, 2026), `quadprog`, `ProxQP` (Bambade et al., 2022), `OSQP` (Stellato et al., 2020)), the differentiable `cvxpylayers` (Agrawal et al., 2019), and the per-frame IK that teleoperation stacks actually run: Pink (Caron et al., 2024), the differential-IK layer used by Niu et al. (2026), and an SNS-style task-exact step (Flacco et al., 2015). The main test is *closed loop*: on six held-out episodes per robot,

each method's velocity is executed (integrated) and fed back, so per-frame compromises accumulate. Every method is subject to the same task, the same limits and the same acceleration bound, the 90th percentile of the operator's own recorded joint acceleration, imposed on the next frame relative to the executed history. ASPL runs in receding horizon (T=8, execute one frame, replan, warm-started from the previous solution); the per-frame controllers see the bound as a tightened box on the frame they solve. When a target is unreachable within the constraints, every method scales the task the same way.

Section 4.3 and Figure 4 summarize the closed-loop and timing results. Three details are not in the main text. A controller that sees one frame must scale the task whenever the bound and the target conflict and then never catches up, which is why the per-frame controllers drift; clipping the raw command drifts to 17–87 mm. On walking, ASPL's RMS joint acceleration is 13% below that of bounded PINK and 16% below the shipped per-frame controllers. Latency per control step at batch 1 is 6–14 ms for ASPL against 4–11 ms for `Clarabel`, well inside the 33 ms period; the advantage in time is in the batched regime: 4–5× over `Clarabel` at T=16, 2–3.5× at T=64, and 25–140× over `quadprog` and `cvxpylayers` at T=16.

Of the solvers above, only `cvxpylayers` also returns a derivative. Because the layer is affine on the identified face, its adjoint is one reduced solve with the factor the forward already holds: it agrees with central finite differences of the exact solution to within the reference's own accuracy (10^{-7} to 3×10^{-3}, the same figures a converged 200-step unroll attains) at 36–150× lower cost than that unroll; the comparison with `cvxpylayers` is in Section 4.3. We use it for an amortized IK network in the spirit of learned IK (Ames et al., 2022), one MLP from the frame's parameters to a joint velocity trained on past episodes and tested on held-out frames. A regression head violates a limit on 67–78% of frames and misses the task by 0.17–0.74 rad/s; clipping fixes the limits and not the task. With the ASPL layer as the output head the same network satisfies the task to 4×10^{-11} on every held-out frame and the limits on 99.6–99.9% of them, at 5–7 μs per frame, whether the layer is trained through or only applied at inference.

A4.5 MAZE TRAJECTORY PLANNING

We use the maze example of Marcucci et al. (2023), which casts smooth obstacle-avoiding trajectory planning as a Graph-of-Convex-Sets (GCS) program: each free cell of a polytope decomposition is a node, edges encode cell adjacency, and the full problem selects a cell sequence and fits a Bézier curve to it. We fix the cell path by BFS and train ASPL to amortize the resulting convex subproblem: one MLP forward pass followed by the projection onto that path's polytope.

For a maze of side $M_{\text{maze}} \in \{10, 15, 20, 25, 30\}$, we first run BFS on the cell-adjacency graph to extract the unique corner-to-corner cell sequence $P^\star = (c_1, \dots, c_{M_P})$. The decision variable stacks N_c=7 Bézier control points (degree-6 segment) per cell:

$$\mathbf{z} \;=\; \left[\mathbf{c}_{1,1}, \dots, \mathbf{c}_{1,N_c}, \dots, \mathbf{c}_{M_P,1}, \dots, \mathbf{c}_{M_P,N_c}\right] \in \mathbb{R}^{2M_P N_c}. \tag{23}$$

Equality constraints $\mathbf{A}_{\text{eq}}\mathbf{z}$=$\mathbf{b}(\mathbf{p}_0, \mathbf{p}_g)$ pin the trajectory endpoints to the (per-sample) start/goal points $\mathbf{p}_0, \mathbf{p}_g$, enforce zero start/goal velocity and acceleration, and impose C^0/C^1/C^2 continuity at each of the M_P-1 internal segment junctions ($n_{\text{eq}} = 12 + 6(M_P-1)$ rows). Box constraints $\mathbf{l} \le \mathbf{z} \le \mathbf{u}$ confine every control point to the polytope of its parent cell. The cost

$$J(\mathbf{z}) \;=\; \tfrac{1}{2}\mathbf{z}^\top \mathbf{H}\mathbf{z}, \qquad \mathbf{H} \;=\; \mathbf{D}^\top\mathbf{D} \,+\, \epsilon\mathbf{I}, \tag{24}$$

penalizes the squared second difference along consecutive control points (the discrete analogue of the curve's curvature), with ϵ=10^{-3} a strict-convexity regularizer. This is the GCS subproblem with the path fixed; its exact solution $\mathbf{z}^\star$, computed with an interior-point solver (`Clarabel` through CVXPY), is the reference for all metrics.

We test two scenarios: **S2**, a fixed maze layout with start and goal varying across samples, and **S3**, a new maze and endpoint pair per sample. Datasets have 1024 (S2) and 2048 (S3) instances with an 80/10/10 train/validation/test split. The backbone is a small MLP; a banded-Cholesky spectral prefilter $\mathbf{S} = (\mathbf{I} + 10\mathbf{H})^{-1}$ is applied to its output before the projection, and a tube constraint (δ=0.45 on interior control points) and C^3 continuity rows at the junctions are part of the projected constraint set. The projection layer runs 200 iterations of a dual-ascent forward. The training loss is the mean-squared error to $\mathbf{z}^\star$ for $M_{\text{maze}} \in \{10, 15\}$ and the $\mathbf{H}$-weighted error $\|\mathbf{D}(\hat{\mathbf{z}} - \mathbf{z}^\star)\|^2$ for $M_{\text{maze}} \in \{20, 25, 30\}$.

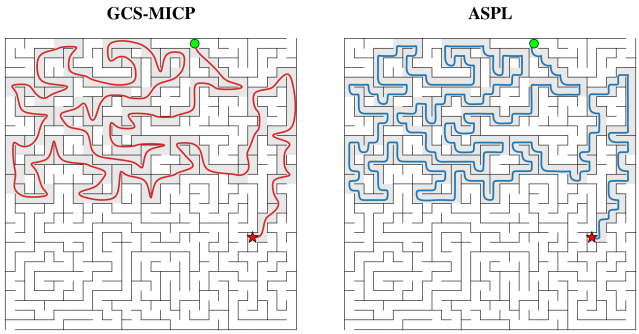

Figure 11: Example maze trajectory on scenario S3 (novel maze; an S2 example is in Figure 5). The right panel is the ASPL output; the left panel shows a trajectory from the GCS pipeline of Marcucci et al. (2023) for visual comparison only (the quantitative reference is the fixed-path QP solution $\mathbf{z}^\star$).

Figure 11 shows an example on a new maze (S3). Every test output satisfies the cell-polytope box constraints, and the equality constraints, including the pinned endpoints, hold up to the projection's solver tolerance ($\leq 10^{-3}$ in our audit).

A4.6 Constrained flow matching

We deploy ASPL on top of the point-cloud manipulation policy ManiFlow (Yan et al., 2025) (architecture in Appendix A4.6.1) on the Aloha-Agilex twin-arm robot in SAPIEN. With observation $\mathbf{o} \in \mathcal{O}$ and action chunk $\mathbf{A}_{\boldsymbol{\theta}}(\mathbf{o})$, every bimanual chunk must satisfy box bounds on the joints, finite-difference velocity and acceleration limits, an initial-state equality $\mathbf{a}_1{=}\mathbf{q}_0$, and the real-time chunking (RTC) (Black et al., 2025) prefix equality $\mathbf{a}_{1:d}{=}\mathbf{c}$ — a polyhedral set $\mathcal{C}(\mathbf{o})$ covered by Corollary 3. *Constrained flow matching* is then

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \;\; \mathbb{E}_{\mathbf{o}}\left[\,\varphi(\mathbf{A}_{\boldsymbol{\theta}}(\mathbf{o}); \mathbf{o})\,\right] \quad \text{s.t.} \quad \mathbf{A}_{\boldsymbol{\theta}}(\mathbf{o}) \in \mathcal{C}(\mathbf{o}) \quad \text{for every } \mathbf{o}, \tag{CFM}$$

with φ the regression + flow-matching loss defined below. As in Section 4.5, the projection is applied at inference only.

We train the ManiFlow-PointCloud policy on RoboTwin 2.0 (Chen et al., 2026) (one training seed) and evaluate it on the same 100 fixed episodes per task (one evaluation seed) for `lift_pot`, `open_laptop`, `pick_dual_bottles`, and `put_object_cabinet`, with and without the $\Pi_{\mathcal{C}}$ layer (per-task joint, velocity, and acceleration bounds of Appendix A4.6.2, plus the initial-state and RTC equalities). Section 4.5 and Figure 6 report the success rates and the per-episode changes on `put_object_cabinet` and `pick_dual_bottles`; on `lift_pot` and `open_laptop` the gains and losses balance.

Flow matching for action chunks. We build on the flow-matching construction of Lipman et al. (2023); Liu et al. (2023). A chunked manipulation policy maps an observation buffer $\mathbf{o} \in \mathcal{O}$ (visual, proprioception, optional language) to an action chunk $\mathbf{A} \in \mathbb{R}^{H\times n}$ that the controller executes. Treating the chunk as a point in $\mathbb{R}^{Hn}$, flow matching trains a time-conditioned velocity field $\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t \mid \mathbf{o}) : \mathbb{R}^{Hn}{\times}[0,1]{\times}\mathcal{O}{\to}\mathbb{R}^{Hn}$ to align with the straight-line conditional flow between Gaussian noise and the demonstration target. Given $(\mathbf{o}, \mathbf{A}_{\text{gt}}) \sim \mathcal{D}$, $\mathbf{x}_0 \sim \mathcal{N}(0, \mathbf{I})$, $\mathbf{x}_1 = \mathbf{A}_{\text{gt}}$, $t \sim \mathcal{U}[0,1]$, and $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$, the flow-matching loss is

$$\mathcal{L}_{\text{FM}}(\boldsymbol{\theta}) \;=\; \mathbb{E}_{(\mathbf{o},\mathbf{x}_1),\,\mathbf{x}_0,\,t}\left\| \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t \mid \mathbf{o}) - (\mathbf{x}_1{-}\mathbf{x}_0) \right\|^2. \tag{25}$$

At inference, an action chunk is generated by integrating $\dot{\mathbf{x}}_t = \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t \mid \mathbf{o})$ from $\mathbf{x}_0$ to $\mathbf{x}_1$ with an explicit Euler scheme on K uniform steps,

$$\mathbf{x}_{t_{k+1}} \;=\; \mathbf{x}_{t_k} \;+\; \tfrac{1}{K}\,\mathbf{v}_{\boldsymbol{\theta}}\big(\mathbf{x}_{t_k},\, t_k \mid \mathbf{o}\big), \qquad k = 0, \ldots, K-1, \tag{26}$$

and denormalized to give the raw action chunk $\mathbf{A}_{\text{raw}} \in \mathbb{R}^{H\times n}$. We abbreviate the entire forward map $\mathbf{o} \mapsto \mathbf{A}_{\text{raw}}$ by $f_{\boldsymbol{\theta}}$. A complementary consistency-training term (Song et al., 2023; Frans et al., 2025) ties $\mathbf{v}_{\boldsymbol{\theta}}$ at intermediate times to a slowly evolving EMA target so that K can be reduced to 1–2 steps at deployment. The relevant property for us is that $f_{\boldsymbol{\theta}}$ is differentiable in $\boldsymbol{\theta}$ end-to-end through (26), so a projection-layer loss can be backpropagated through the ODE rollout exactly as through any feedforward network.

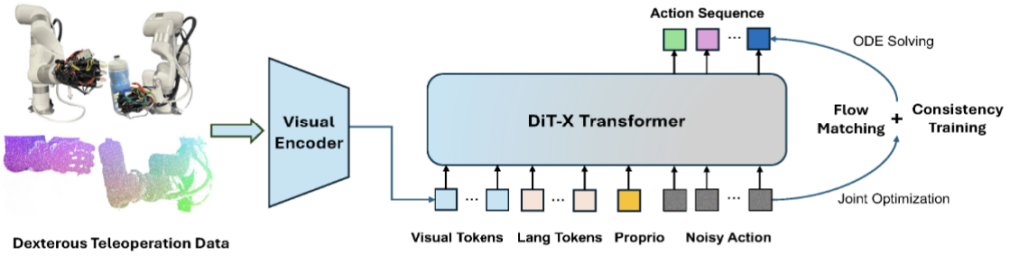

Figure 12: ManiFlow (Yan et al., 2025) pipeline that ASPL wraps. A point-cloud encoder (DP3 (Ze et al., 2024)) produces visual tokens; together with language and proprioceptive tokens these are consumed by the DiT-X transformer that parametrizes the velocity field $\mathbf{v}_{\boldsymbol{\theta}}$. The Euler ODE (26) integrates from Gaussian noise to a denoised chunk; ASPL applies the projector $\Pi_{\mathcal{C}(\mathbf{o})}$ to the terminal ODE state at inference; the policy is trained with the standard ManiFlow losses. Image credit: ManiFlow paper.

A4.6.1 Backbone and DiT-X block

For the RoboTwin 2.0 bimanual setup we use $H{=}16$ tokens of action and $n{=}14$ joint DoFs (two 7-DoF arms), so the flattened chunk lives in $\mathbb{R}^{Hn}$ with $Hn{=}224$. By ManiFlow's convention the first token $\mathbf{a}_0$ is a non-executed reference and the executed slice is $\mathbf{A}_{\text{exec}} = \mathbf{A}[1{:}H]$, the $H{-}1{=}15$ tokens after the reference (the configured $n_{\text{action}}{=}16$ is truncated to them); consequently the initial-state equality $\mathbf{a}_1 = \mathbf{q}_0$ must select column block $[n, 2n)$ of $\mathbf{y}$, not $[0, n)$.

The DiT-X transformer (Yan et al., 2025; Peebles & Xie, 2023) stacks $L{=}3$ blocks with $H_{\text{head}}{=}4$ attention heads at embedding dimension $n_{\text{emb}}{=}256$. Each block applies AdaLN-Zero conditioning on the timestep token and an adaptive cross-attention mechanism that lets action tokens attend to the multi-modal stream through learned scale-shift parameters; we use the architecture as released (Figure 12). The visual conditioner is the DP3 (Ze et al., 2024) point-cloud encoder (1024-dim feature); language conditioning, when present, comes from a frozen T5 encoder; proprioception is a 2-layer MLP with token-level masking probability 0.2 during training. Three streams interact inside one block: an action stream of H tokens carrying $\mathbf{x}_{t_k}$, a multi-modal observation stream, and a sinusoidally embedded timestep stream. Each block applies AdaLN-Zero modulation, self-attention on the action stream, adaptive cross-attention (Reuss et al., 2024) where action tokens query the observation stream, and a $4\times$-expansion feed-forward MLP, all with residual connections; AdaLN-Zero initialization makes the block start as the identity. The ODE is integrated for $K{=}10$ Euler steps both at training and at deployment (parity with the released checkpoint); the projector $\Pi_{\mathcal{C}(\mathbf{o})}$ adds ~ 6 ms per chunk on top of ODE integration.

A4.6.2 Constraint set $\mathcal{C}(\mathbf{o})$ and polyhedral instance

We expose every constraint in factored form so it lifts directly to the polyhedral ASPL algorithm, with $\mathbf{1}_H$ the all-ones vector in $\mathbb{R}^H$ and $\otimes$ the Kronecker product. *Joint position bounds* (C_1, box) tighten the URDF lower/upper bounds to the demonstration min/max:

$$\big(\mathbf{1}_H \otimes \mathbf{q}_{\min}\big) \;\le\; \mathbf{y} \;\le\; \big(\mathbf{1}_H \otimes \mathbf{q}_{\max}\big), \tag{27}$$

with the range $[\mathbf{q}_{\min}, \mathbf{q}_{\max}]$ widened about its midpoint by a factor 1.2, contributing $448 = 2Hn$ box bounds. *Joint-velocity inequalities* (C_2, affine) use the first-order finite-difference operator $D_1 \in \mathbb{R}^{(H-1)n \times Hn}$:

$$-\boldsymbol{\delta}_{\max} \;\le\; D_1\mathbf{y} \;\le\; \boldsymbol{\delta}_{\max}, \qquad \boldsymbol{\delta}_{\max} \;=\; 1.2 \cdot \mathtt{p99}\big(\{\,|\mathbf{a}^{(\mathrm{gt})}_{t+1} - \mathbf{a}^{(\mathrm{gt})}_{t}|\,\}\big), \tag{28}$$

contributing $2(H{-}1)n \;=\; 420$ inequalities; the $1.2\times$ margin is chosen so that every ground-truth trajectory satisfies the bound. *Joint-acceleration inequalities* (C_3, affine, anti-oscillation) use the second difference $D_2 \in \mathbb{R}^{(H-2)n \times Hn}$:

$$-\boldsymbol{\alpha}_{\max} \;\le\; D_2\mathbf{y} \;\le\; \boldsymbol{\alpha}_{\max}, \qquad \boldsymbol{\alpha}_{\max} \;=\; 1.5 \cdot \mathtt{p99}\big(\{\,|\mathbf{a}^{(\mathrm{gt})}_{t+1} - 2\mathbf{a}^{(\mathrm{gt})}_{t} + \mathbf{a}^{(\mathrm{gt})}_{t-1}|\,\}\big), \tag{29}$$

contributing $2(H{-}2)n \;=\; 392$ inequalities; this penalizes oscillation. *Initial-state equality* (C_4) requires the first executed action to match current proprioception:

$$\mathbf{E}_{\text{init}}\mathbf{y} \;=\; \mathbf{q}_0, \qquad \mathbf{E}_{\text{init}} \in \mathbb{R}^{n \times Hn}, \;\; (\mathbf{E}_{\text{init}})_{j,n+j} = 1, \tag{30}$$

where the column block $[n, 2n)$ targets $\mathbf{a}_1$ (the first executed token), not the discarded reference $\mathbf{a}_0$. *Real-time chunking consistency* (Black et al., 2025) (C_5, affine equality) pins the committed prefix $\mathbf{c} \in \mathbb{R}^{dn}$ ($d = 4$ tokens, 56 components) from the previous chunk:

$$\mathbf{E}_{\text{rtc}}\mathbf{y} \;=\; \mathbf{c}, \qquad \mathbf{E}_{\text{rtc}} \in \mathbb{R}^{dn \times Hn}, \;\; (\mathbf{E}_{\text{rtc}})_{i,i} = 1, \;\; i \in [0, dn); \tag{31}$$

sliding-window updating preserves $d-1 = 3$ executed-token overlap between consecutive chunks. Stacking equalities $(\mathbf{E}_{\text{init}}; \mathbf{E}_{\text{rtc}})$, inequalities $(D_1; D_2)$, and the box bounds gives the unified set

$$\mathcal{C}(\mathbf{o}) \;=\; \left\{ \mathbf{y} \in \mathbb{R}^{Hn} : \mathbf{A}_{\text{eq}}\mathbf{y} = \mathbf{b}_{\text{eq}}(\mathbf{o}),\; \mathbf{l}_{\text{ineq}} \le \mathbf{C}_{\text{ineq}}\mathbf{y} \le \mathbf{u}_{\text{ineq}},\; \mathbf{l}_{\text{box}} \le \mathbf{y} \le \mathbf{u}_{\text{box}} \right\}, \tag{32}$$

with $n_{\text{eq}} \in [14, 70]$, n_{ineq}=812, n_{box}=448 — the polyhedral instance of Corollary 3, so Theorem 1 would supply the backward if the policy were trained through the layer (here it is applied at inference only).

A4.6.3 Unified Douglas–Rachford projector

We treat $\Pi_{\mathcal{C}(\mathbf{o})}$ as a single layer that projects jointly across all five constraint families. Following Grontas et al. (2026); Bauschke & Combettes (2017), we lift the inequalities to slacks $\mathbf{s}_{\text{ineq}} \in \mathbb{R}^{n_{\text{ineq}}}$ and form the lifted variable $\overline{\mathbf{y}} = (\mathbf{y}; \mathbf{s}_{\text{ineq}})$. The lifted equality matrix and right-hand side are

$$\overline{\mathbf{A}}_{\text{eq}} = \begin{pmatrix} \mathbf{A}_{\text{eq}} & \mathbf{0} \\ \mathbf{C}_{\text{ineq}} & -\mathbf{I} \end{pmatrix}, \qquad \overline{\mathbf{b}}_{\text{eq}}(\mathbf{o}) = \begin{pmatrix} \mathbf{b}_{\text{eq}}(\mathbf{o}) \\ \mathbf{0} \end{pmatrix}, \tag{33}$$

with lifted box $\bar{\mathbf{l}} = (\mathbf{l}_{\text{box}}; \mathbf{l}_{\text{ineq}})$, $\overline{\mathbf{u}} = (\mathbf{u}_{\text{box}}; \mathbf{u}_{\text{ineq}})$. The two prox sets are then $\mathcal{C}_1 = \{\overline{\mathbf{y}} : \overline{\mathbf{A}}_{\text{eq}}\overline{\mathbf{y}} = \overline{\mathbf{b}}_{\text{eq}}\}$ and $\mathcal{C}_2 = [\bar{\mathbf{l}}, \overline{\mathbf{u}}]$. The cached affine projector is $\Pi_{\mathcal{C}_1}(\overline{\mathbf{y}}) = \overline{\mathbf{y}} - \overline{\mathbf{A}}_{\text{eq}}^{+}(\overline{\mathbf{A}}_{\text{eq}}\overline{\mathbf{y}} - \overline{\mathbf{b}}_{\text{eq}})$ with $\overline{\mathbf{A}}_{\text{eq}}^{+}$ precomputed once per task. The DR step is the over-relaxed scheme of Davis & Yin (2017); Bauschke & Combettes (2017) with ω=1.7, σ=0.1:

$$\overline{\mathbf{s}}^{(k+1)} = \overline{\mathbf{s}}^{(k)} + \omega\Big(\Pi_{\mathcal{C}_2}\big(2\Pi_{\mathcal{C}_1}(\overline{\mathbf{s}}^{(k)}) - \overline{\mathbf{s}}^{(k)}\big) - \Pi_{\mathcal{C}_1}(\overline{\mathbf{s}}^{(k)})\Big), \tag{34}$$

followed by the data-fit weighting $\overline{\mathbf{y}}^{\star} \leftarrow (\sigma\mathbf{D}^2\mathbf{y}_{\text{raw}} + \Pi_{\mathcal{C}_1}(\overline{\mathbf{s}}))/(1 + \sigma\mathbf{D}^2)$ on the primal block. Iterating K_{dr}=200 steps meets the equality and velocity constraints to 10^{-7} and the position box to 10^{-3} (unit tests below). The policy is not trained through the projector, so no backward pass is needed. Per-chunk wall-clock is dominated by 200 matrix-vector products with $\overline{\mathbf{A}}_{\text{eq}}^{+}$ ($1036 \times (n_{\text{eq}}{+}812)$): 6.1 ms on a single CPU thread, $\le$ 1 ms/chunk batched at 32. The DiT-X forward dominates by an order of magnitude.