CHEBYSHEV CENTER-BASED DIRECTION SELECTION FOR MULTI-OBJECTIVE OPTIMIZATION AND TRAINING PINNS

Anonymous authors
Paper under double-blind review

ABSTRACT

Physics-informed neural networks (PINNs) are a promising approach for solving partial differential equations (PDEs). Their training, however, is often difficult because multiple loss terms induced by PDE residuals and boundary or initial conditions must be optimized simultaneously. To address this difficulty, existing approaches often construct update directions by explicitly enforcing particular desirable properties, such as scale robustness and simultaneous descent. While effective in many cases, such property-by-property designs can make it unclear which conditions are essential, what geometric principle determines the selected update direction, and how different methods are structurally related. In this work, we formulate update-direction selection for PINN training as a Chebyshev-center problem in the dual cone. The proposed formulation selects a normalized direction that maximizes the minimum distance to the cone facets. The resulting formulation admits an efficient dual problem in a much lower-dimensional space and yields a convergence guarantee in the nonconvex setting. It also recovers the key desirable properties targeted by existing approaches without imposing them separately; rather, they follow from the single geometric criterion underlying the formulation. This makes the selected direction interpretable through a single geometric rule and provides a unified basis for systematically comparing related direction-selection methods. Experiments on several PINN benchmarks further demonstrate strong empirical performance of the proposed method.

1 INTRODUCTION

Partial differential equations (PDEs) are fundamental tools for modeling a wide range of phenomena in science and engineering, including fluid dynamics, heat transfer, and wave propagation. Since analytically solving complex PDEs is often intractable, a variety of numerical methods have been developed to approximate their solutions. Among recent learning-based approaches, physics-informed neural networks (PINNs) have emerged as a promising framework for solving PDEs (Raissi et al., 2019; Karniadakis et al., 2021). In PINNs, a neural network is trained to approximate the PDE solution using objectives constructed from the governing equation together with boundary and initial conditions. This flexibility has enabled PINNs to be used for a broad class of forward and inverse problems. At the same time, however, their practical success depends critically on how effectively they can be trained (Krishnapriyan et al., 2021; Cuomo et al., 2022).

Training PINNs is challenging in practice, as their performance can be highly sensitive to data sampling, model design, and training dynamics (Wu et al., 2023; Wang et al., 2021b;a). These challenges have motivated a broad literature on sampling strategies (Nabian et al., 2021; Tang et al., 2023; Gao et al., 2023), network design (Jagtap & Karniadakis, 2020; Cho et al., 2023; Zhao et al., 2024), and optimization (Wang et al., 2022; Bischof & Kraus, 2025). In this work, we focus on the optimization aspect of PINN training, where multiple loss terms induced by PDE residuals, boundary conditions, and initial conditions must be optimized simultaneously. This multi-loss structure can produce imbalanced gradient magnitudes across objectives, while also creating conflicts among their preferred update directions (Hwang & Lim, 2024; Liu et al., 2025). As a result, the key optimization challenge is how to construct an update direction from multiple loss terms so that different physical constraints are reflected in a balanced manner.

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.

Existing approaches to multi-loss optimization in PINNs can be broadly categorized into adaptive weighting methods and direction-selection methods. Adaptive weighting methods primarily aim to adjust the relative importance or the scales of different loss terms (Wang et al., 2021a; 2022; Bischof & Kraus, 2025), whereas direction-selection methods seek to construct an update direction from multiple loss-specific gradients (Hwang & Lim, 2024; Liu et al., 2025; Bu et al., 2026). Many such direction-selection methods are motivated by desirable properties of the update direction, such as equalized progress across objectives, simultaneous descent, or robustness to gradient-scale differences. While these constructions are meaningful and often effective, designing directions by imposing such properties one by one can make it less clear which conditions are more essential, what geometric principle determines the selected direction, and how these methods are structurally related. These observations motivate a direction-selection formulation based on a single geometric criterion, rather than on separately imposed properties.

To this end, we formulate the direction-selection problem as a Chebyshev-center problem over the dual cone of the set of loss gradients. The dual cone provides natural candidates for update directions as it collects vectors that are descent directions for every loss function. However, not all directions in the dual cone are equally desirable: a direction close to a facet may give only limited improvement to the corresponding objective. We therefore select a normalized direction at the center of the largest ball inscribed in the dual cone. Rather surprisingly, the single geometric formulation achieves several desirable properties all at once, which have been the individual targets of previous methods: scale robustness via normalized gradient comparisons, balanced treatment across objectives, and simultaneous descent. Moreover, the geometric interpretation, together with the corresponding primal and dual formulations, allows us to systematically compare related direction-selection methods.

In particular, we derive a dual problem over a low-dimensional simplex and show how to recover the primal update direction from its solution. The formulation's connection to Pareto stationarity also provides a stopping criterion and enables convergence analysis. Then we compare our method with some representative methods, including DCGD, HARMONIC, IMTL-G, ConFIG, MGDA and GAPO, and the comparison shows that the proposed formulation can explain prior cone-based choices, recover desirable properties without imposing them separately, and relax restrictive equality-based formulations. Finally, we validate the proposed method through numerical experiments on a range of PINN problems.

2 PRELIMINARIES

Physics-informed neural networks Physics-informed neural networks (PINNs) (Raissi et al., 2019) approximate the solution u of a partial differential equation by a neural network u_θ with parameters θ. Consider a PDE defined on a domain Ω, written as $\mathcal{N}[u](x,t) = 0$ for $(x,t) \in \Omega$, together with boundary and initial conditions $\mathcal{B}[u](x,t) = 0$ on $\partial\Omega$ and $\mathcal{I}[u](x,0) = 0$ on Ω. The goal is to find θ such that u_θ approximately satisfies these physical constraints. To this end, PINNs sample collocation points from the interior of the domain, the boundary, and the initial slice, and define separate loss terms for each constraint, e.g., $\mathcal{L}_{\mathrm{r}}(\theta) = \frac{1}{N_r}\sum_{j=1}^{N_r} \|\mathcal{N}[u_\theta](x_j^r, t_j^r)\|_2^2$, $\mathcal{L}_{\mathrm{b}}(\theta) = \frac{1}{N_b}\sum_{j=1}^{N_b} \|\mathcal{B}[u_\theta](x_j^b, t_j^b)\|_2^2$, and $\mathcal{L}_{\mathrm{i}}(\theta) = \frac{1}{N_i}\sum_{j=1}^{N_i} \|\mathcal{I}[u_\theta](x_j^i, 0)\|_2^2$. Balancing these constraint-specific losses is often critical for stable and effective PINN training. We next review approaches to this challenge.

Multi-objective perspectives on training PINNs One line focuses on adaptive weighting methods for PINNs, such as LRA (Wang et al., 2021a), NTK-based weighting (Wang et al., 2022), and ReLoBRaLo (Bischof & Kraus, 2025). Another line builds on classical multicriteria optimization, where a dual cone-based view of simultaneous descent was implicit in the characterization of descent directions (Fliege & Svaiter, 2000). Related to this classical viewpoint, gradient-based multi-objective and multi-task learning methods, including MGDA (Sener & Koltun, 2018), PCGrad (Yu et al., 2020), CAGrad (Liu et al., 2021a), and IMTL-G (Liu et al., 2021b), combine or modify gradients to reduce conflict among objectives. More recently, gradient-based multi-objective methods have also been developed for constrained reinforcement learning, preference-based MOO/MORL, and multi-objective LLM alignment, including GCPO (Zhou et al., 2023), PMGDA (Zhang et al., 2025), GAPO (Li et al., 2025), and RACO (Chen et al., 2026). In the PINN setting, DCGD (Hwang & Lim, 2024), ConFIG (Liu et al., 2025), and HARMONIC (Bu et al., 2026) are closely related

methods that construct update directions from multiple loss gradients. Other PINN-specific optimization strategies include the parameter-wise adaptive optimizer MultiAdam (Yao et al., 2023) and the minimax-based method BGDA (Bylinkin et al., 2026). In this paper, we focus on gradient-based methods for determining update directions and compare methods that are closely related to ours in Section 4.

Chebyshev centers of polyhedral sets The Chebyshev center is a classical notion for defining the center of a constrained geometric set (see Boyd & Vandenberghe, 2004). Given a norm $\|\cdot\|$, the depth of $x \in C$ for a bounded set $C \subset \mathbb{R}^n$ with nonempty interior is $\inf_{y \in \mathbb{R}^n \setminus C} \|x - y\|$, namely the radius of the largest norm ball centered at x and contained in C. A Chebyshev center is a point of maximum depth, or equivalently the center of a largest inscribed norm ball. For a polyhedron $C = \{x \mid a_i^\top x \leq b_i,\ i = 1, \ldots, m\}$, requiring a norm ball of radius R centered at x to lie in C is equivalent to $a_i^\top x + R\|a_i\|_* \leq b_i$ for all i, where $\|\cdot\|_*$ denotes the dual norm of $\|\cdot\|$. This notion of centrality for polytopes also extends to polyhedral cones, as explained in Section 3.1.

3 METHODOLOGY

3.1 CHEBYSHEV CENTER OF THE DUAL CONE FOR DIRECTION SELECTION

We consider a multi-objective optimization problem with m loss functions $\{\mathcal{L}_i\}_{i=1}^m$. At iteration t, let $g_i := \nabla \mathcal{L}_i(\theta_t)$ denote the gradient of $\mathcal{L}_i$. Under standard gradient descent, the rate of change in $\mathcal{L}_i$ along a direction d is determined by $g_i^\top d$; in particular, if $g_i^\top d < 0$, d is not a descent direction for $\mathcal{L}_i$. Therefore, depending on the choice of d, some losses may decrease while others increase. To avoid such conflicts, we consider directions satisfying $g_i^\top d \geq 0$ for all $i = 1, \ldots, m$. This simultaneous descent condition can be naturally expressed in terms of the dual cone.

Definition 3.1 (Dual cone)**.** Let $\mathbf{K} \subseteq \mathbb{R}^n$ be a cone. Then, the dual cone of $\mathbf{K}$ is defined as

$$\mathbf{K}^* = \{\, y \in \mathbb{R}^n \mid x^\top y \geq 0,\ \forall x \in \mathbf{K} \,\}.$$

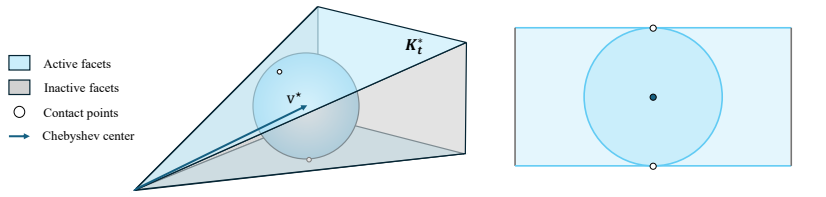
Let $\mathbf{K}_t := \mathrm{cone}(g_1, \ldots, g_m)$ be the cone generated by the gradients at iteration t, and let $\mathbf{K}_t^*$ denote its dual cone. If $d \in \mathbf{K}_t^*$, then $x^\top d \geq 0$ for all $x \in \mathbf{K}_t$. In particular, $g_i^\top d \geq 0$ for all $i = 1, \ldots, m$. Thus, choosing the update direction d from $\mathbf{K}_t^*$ guarantees a descent direction for all losses. As the dual cone may contain infinitely many feasible directions, an additional criterion is needed to choose an update direction. Inspired by this, in this work, we develop a centrality notion within the dual cone as the selection criterion, favoring directions that lie away from the cone boundaries.

Note that hyperplane $g_i^\top d = 0$ defines a face of the cone. Since $g_i^\top d$ determines the rate of change in $\mathcal{L}_i$, choosing a direction near this face yields a small rate of decrease for the corresponding loss. This motivates a centrality criterion that favors directions away from the cone boundary. To formalize this, we use the Chebyshev center, a geometric notion that defines a center of a polyhedral set.

To elaborate, we consider a pair of conjugate ℓ_p/ℓ_q norms with $1 \leq p \leq \infty$ and $q = p/(p-1)$. Let $\hat{g}_i^{(p)} := g_i/\|g_i\|_p$ denote the gradient normalized with respect to the ℓ_p norm. Throughout the paper, we assume that $\|g_i\|_p > 0$ for all i, so that each normalized gradient is well defined. Since $\hat{g}_i^{(p)}$ is a positive rescaling of g_i, the hyperplane $g_i^\top v = 0$ is equivalently written as $(\hat{g}_i^{(p)})^\top v = 0$. Then (the absolute value of) the distance from v to this hyperplane is $g_i^\top v/\|g_i\|_p = (\hat{g}_i^{(p)})^\top v$. Here, the Chebyshev-center criterion seeks a center v that maximizes the smallest of the distances to the hyperplanes. Because the dual cone is scale-invariant, this criterion requires normalization of v; otherwise, the distances can grow trivially along the same ray. Since our interest is in the update direction rather than its magnitude, we fix the scale by imposing $\|v\|_q \leq 1$. Note that finding a Chebyshev center boils down to computing the maximum admissible radius of an ℓ_q ball contained in the dual cone. Hence, we deduce the following formulation.

$$\max_{v,\, r} \quad r \quad \text{s.t.} \quad (\hat{g}_i^{(p)})^\top v \geq r,\ i = 1, \ldots, m, \quad \|v\|_q \leq 1. \tag{P}$$

Note that $(v, r) = (0, 0)$ is always feasible for Problem (P), and hence the optimal value satisfies $r^\star \geq 0$. Therefore, any optimal solution $v^\star$ satisfies $(\hat{g}_i^{(p)})^\top v^\star \geq 0$ for all $i = 1, \ldots, m$. Since $\hat{g}_i^{(p)}$ is a positive rescaling of g_i, it follows that $g_i^\top v^\star \geq 0$ for all i, and hence $v^\star \in \mathbf{K}_t^*$.



(a) Chebyshev ball in the dual cone (b) Schematic view of active and inactive constraints

Figure 1: Geometric illustration of the proposed direction-selection rule, shown in the Euclidean case $p = q = 2$ for visual clarity. (a) The update direction $v^\star$ is chosen from the dual cone $\mathbf{K}_t^*$ as the center of an admissible Chebyshev ball. (b) A schematic cross-sectional view shows that only active constraints are tangent to the ball, while inactive constraints remain separated. The same active/inactive interpretation applies under general ℓ_q balls.

The radius r is determined by the closest facets to the center v; hence, only the closest facets are active at the optimum. Other ones may remain strictly farther away, as illustrated in Figure 1. Therefore, the formulation does not enforce that the distances to the hyperplanes are equal, but only requires them to be at least r.

3.2 Dual formulation to get optimal solution efficiently

We next derive the dual formulation of (P) for efficient computation of the update direction.

Proposition 3.2. *The dual of* (P) *is given by*

$$\min_{\alpha} \left\| \sum_{i=1}^{m} \alpha_i \hat{g}_i^{(p)} \right\|_p \quad \text{s.t.} \quad \sum_{i=1}^{m} \alpha_i = 1, \quad \alpha_i \geq 0,\ i = 1, \ldots, m. \tag{D}$$

Moreover, strong duality holds between (P) and (D).

By Proposition 3.2, (P) can be solved equivalently through its dual Problem (D). This is advantageous in our setting because the primal variable v lies in the parameter space $\mathbb{R}^n$, whose dimension n is typically very large in neural network training. In contrast, the dual variable α lies in $\mathbb{R}^m$, where m is the number of loss terms and is usually much smaller than n. In such a case, solving the dual is computationally more favorable than solving the primal problem directly, since it reduces the optimization to a simplex-constrained convex problem in a much lower-dimensional space.

Although solving (D) is computationally advantageous, our main interest is in the primal optimal direction $v^\star$ for the update. Solving the dual problem yields the optimal dual variable $\alpha^\star$, and strong duality ensures agreement of the optimal values, but it does not directly provide the primal direction itself. The following proposition shows that, when $r^\star > 0$, the primal optimal direction can be recovered explicitly from the dual optimal solution.

Proposition 3.3. *Assume that* $1 < p < \infty$. *Let* $(v^\star, r^\star)$ *be an optimal solution to the primal* (P)*, and let* $\alpha^\star$ *be an optimal solution to the dual* (D). *Define* $w^\star := \sum_{i=1}^{m} \alpha_i^\star \hat{g}_i^{(p)}$. *If* $r^\star > 0$*, then* $w^\star \neq 0$ *and the primal optimal direction is recovered as* $v^\star = \operatorname{sgn}(w^\star) \odot |w^\star|^{p-1} / \|w^\star\|_p^{p-1}$*, where* $\operatorname{sgn}(\cdot)$*,* $|\cdot|$*, and the power are applied componentwise, and* $\odot$ *denotes componentwise multiplication.*

Note that when $p = 2$, the recovery formula in Proposition 3.3 reduces to the simple Euclidean normalization $v^\star = \frac{w^\star}{\|w^\star\|_2}$. For $1 < p < \infty$, Proposition 3.3 shows that the primal update direction can be obtained directly from the dual optimal solution. Therefore, in practice, it is sufficient to solve (D), form $w^\star = \sum_{i=1}^{m} \alpha_i^\star \hat{g}_i^{(p)}$, and recover $v^\star$ using the formula in Proposition 3.3.

Remark 3.4. The cases $p = 1$ and $p = \infty$ are also well defined for both the primal and dual formulations. However, the closed-form map in Proposition 3.3 no longer directly recovers the primal direction from the dual solution, as the corresponding primal direction may not be uniquely determined. Related ℓ_∞-norm constraints also arise in classical multicriteria steepest-descent methods; in that setting, the resulting direction-selection problem can be formulated as a linear program (Fliege

& Svaiter, 2000). In Appendix A.3, we spell out the corresponding recovery step explicitly for our formulation.

The same primal–dual framework also provides a natural stopping criterion for the method. To formalize this criterion, we use Pareto stationarity as the first-order notion of optimality in multi-objective optimization.

Definition 3.5 (Pareto stationarity)**.** For the multi-objective problem $\min_{\theta\in\mathbb{R}^n}\big(\mathcal{L}_1(\theta),\dots,\mathcal{L}_m(\theta)\big)$ where each $\mathcal{L}_i$ is differentiable, a point $\theta^\star\in\mathbb{R}^n$ is *Pareto-stationary* if there exists $\alpha\in\mathbb{R}^m_+$ such that $\sum_{i=1}^m\alpha_i=1$ and $\sum_{i=1}^m\alpha_i\nabla\mathcal{L}_i(\theta^\star)=0$.

Proposition 3.6. *Let $(v^\star,r^\star)$ be an optimal solution to the primal* (P) *at θ. If $r^\star=0$, then θ is Pareto-stationary.*

Proposition 3.6 shows that the primal optimal value provides a natural stopping criterion for our method. In practice, however, requiring $r^\star=0$ exactly is too restrictive, so we use an approximate criterion instead. Specifically, we refer to a point θ_t as ϵ-Pareto-stationary if the corresponding primal optimal value satisfies $r_t^\star\le\epsilon$ for a prescribed tolerance $\epsilon>0$. By strong duality, this can be checked through $\|\sum_{i=1}^m\alpha_{t,i}\hat{g}_i^{(p)}(\theta_t)\|_p$, which is computed in the algorithm. With this stopping criterion, our method can be summarized in Algorithm 1. Since the dual is a convex problem over a simplex, one may use the Frank–Wolfe algorithm, as for MGDA (Sener & Koltun, 2018). At the same time, in the ℓ_2 case with small m, there exists a faster dual solver; see Appendix C. In the final step, the recovered ℓ_q-unit direction is scaled by the sum of its inner products with all loss gradients to form the final direction.

Algorithm 1 Chebyshev Center-Based Direction Search via the Dual Formulation (DualChebyshev)

Require: Initial point θ_0, norm parameter $p\in(1,\infty)$, step sizes $\{\eta_t\}_{t=0}^{T-1}$, threshold $\epsilon>0$

for $t=0,1,\dots,T-1$ **do**
  Compute ℓ_p-normalized gradients $\hat{g}_1^{(p)}(\theta_t),\dots,\hat{g}_m^{(p)}(\theta_t)$
  Solve the dual (D) to obtain $\alpha_t\in\Delta^m$
  Form $w_t\leftarrow\sum_{i=1}^m\alpha_{t,i}\,\hat{g}_i^{(p)}(\theta_t)$
  if $\|w_t\|_p\le\epsilon$ **then**
    terminate and return θ_t
  end if
  Recover the primal direction: $v_t\leftarrow\operatorname{sgn}(w_t)\odot|w_t|^{p-1}/\|w_t\|_p^{p-1}$
  Set the final update direction with adaptive scalar: $d_t\leftarrow(\sum_{i=1}^m g_i(\theta_t)^\top v_t)\,v_t$
  Update the iterate: $\theta_{t+1}\leftarrow\theta_t-\eta_t d_t$
end for
return θ_T

Remark 3.7. For the cases $p=1$ and $p=\infty$, the implementation differs from Algorithm 1. The dual objective becomes nonsmooth, so the dual problem is better handled through an LP reformulation rather than the Frank–Wolfe step used above. Moreover, as discussed in Remark 3.4, these cases require an additional recovery step because the closed-form map in Proposition 3.3 does not directly apply. Appendix A.5 describes how to solve the dual problem in the $p=1$ and $p=\infty$ cases, and Appendix A.3 describes the corresponding recovery step.

3.3 Convergence analysis

The following theorem provides convergence guarantees for Algorithm 1 in the nonconvex setting, under a smoothness assumption on the total loss compatible with the chosen ℓ_q geometry. Here, $\theta^\star$ denotes a reference point satisfying $\mathcal{L}(\theta^\star)\le\mathcal{L}(\theta_t)$ for all iterates considered. We measure Pareto stationarity by $S_p(\theta):=\min_{\lambda\in\Delta^m}\big\|\sum_{i=1}^m\lambda_i g_i(\theta)\big\|_p$, which is zero if and only if θ is Pareto stationary.

Theorem 3.8. *Assume that $\mathcal{L}_1,\dots,\mathcal{L}_m$ are differentiable and that $\mathcal{L}:=\sum_{i=1}^m\mathcal{L}_i$ is β-smooth with respect to the ℓ_q norm, where $\beta>0$. With step size $\eta=1/\beta$, Algorithm 1 either terminates before T updates at an ϵ-Pareto-stationary point, or satisfies*

$$\min_{0\le t<T}S_p(\theta_t)^2\le\frac{2\beta\big(\mathcal{L}(\theta_0)-\mathcal{L}(\theta^\star)\big)}{T},\qquad\min_{0\le t<T}\|\nabla\mathcal{L}(\theta_t)\|_p^2\le\frac{2\beta\big(\mathcal{L}(\theta_0)-\mathcal{L}(\theta^\star)\big)}{\epsilon^2T}.$$

Thus, unless the algorithm terminates earlier, the smallest Pareto-stationarity measure among the first T iterates is bounded by $O(T^{-1/2})$.

4 Comparative Analysis

In this section, we compare our method with several closely related direction-selection methods. Recall that ours selects a direction through the Chebyshev-center criterion over the dual cone, rather than by imposing individual properties. We will see that based on the primal and dual formulations for the Chebyshev center, we may systematically analyze and compare the related methods.

Since the related methods considered below are formulated in the Euclidean geometry, we focus in this section on the default case $p = q = 2$. Accordingly, we write $\hat{g}_i := \hat{g}_i^{(2)} = g_i/\|g_i\|_2$ throughout this section. For ease of comparison, we express all methods in this Euclidean notation. Figure 2 gives a geometric overview of the four direction-selection methods compared in this section. The construction details for this illustration are provided in Appendix B.1.

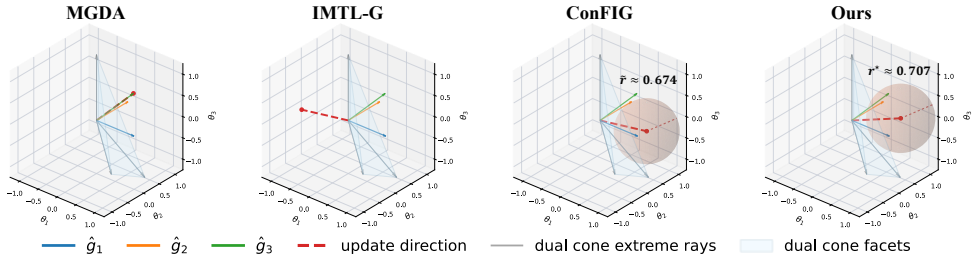


Figure 2: A three-objective example illustrating geometric differences with some representative methods.

4.1 Geometric comparison with cone-based methods

Cone-based methods use conic geometry to construct update directions from multiple loss gradients. In this subsection, we compare our method with DCGD and HARMONIC, focusing on how their constructions relate to our Chebyshev-center formulation.

DCGD Within the dual cone, three direction-selection variants are considered: Projection, Average, and Center (Hwang & Lim, 2024). Among them, the Center variant uses the angle bisector and was reported to achieve the best empirical performance. In the two-objective nondegenerate case, the angle bisector coincides with the direction selected by our Chebyshev-center formulation. Thus, for $m = 2$, the reported advantage of DCGD (Center) is consistent with the centrality criterion used in our method.

For $m \geq 3$, however, the notion of an angle bisector is no longer canonical, whereas our Chebyshev-center formulation may take any number of objectives.

HARMONIC HARMONIC (Bu et al., 2026) selects a direction in the harmonized cone $\mathbf{H}_t := \mathbf{K}_t \cap \mathbf{K}_t^*$. With $G_t = [g_1, \ldots, g_m]$, it enforces membership in both cones through $d = G_t\lambda$, $\lambda \geq 0_m$, and $G_t^\top G_t \lambda \geq 0_m$. It enumerates the extreme rays of this coefficient-space cone using the Double Description method, maps them through G_t, and averages the normalized rays.

In contrast, our method neither constructs $\mathbf{H}_t$ nor enumerates extreme rays. The Chebyshev-center problem (P) directly selects a direction in $\mathbf{K}_t^*$. Moreover, Proposition 3.3 gives $v^\star = w^\star/\|w^\star\|_2$, where $w^\star$ is a conic combination of the normalized gradients and hence lies in $\mathbf{K}_t$. Therefore, the $\mathbf{H}_t$ membership, explicitly enforced by HARMONIC, automatically holds in our method as an optimality consequence of the Chebyshev-center formulation.

4.2 Comparison with equality-based methods based on the primal formulation

We next compare our formulation with methods that determine the update direction through gradient–direction inner products. In particular, IMTL-G and ConFIG both aim to equalize these inner products across objectives, but they differ in whether the resulting direction is guaranteed to lie in the dual cone. Through (P), we view these methods as equality-based counterparts of our inequality-based Chebyshev-center formulation.

IMTL-G IMTL-G (Liu et al., 2021b) seeks a direction $d = \sum_{i=1}^{m} \alpha_i g_i$, with $\sum_{i=1}^{m} \alpha_i = 1$, whose normalized-gradient inner products are equal across objectives: $\hat{g}_1^\top d = \hat{g}_2^\top d = \cdots = \hat{g}_m^\top d$.

Thus, IMTL-G explicitly normalizes gradients and imposes equality of their inner products with the update direction. However, as noted in ConFIG, the common value can be negative; if $\hat{g}_1^\top d = \cdots = \hat{g}_m^\top d = c < 0$, then the resulting direction does not lie in $\mathbf{K}_t^*$.

In contrast, in our formulation, normalization is not introduced as a separate designed property; it follows from measuring distances to the defining hyperplanes of the dual cone. Moreover, the update direction is selected within $\mathbf{K}_t^*$, avoiding the negative inner product value described above.

ConFIG To enforce equal positive normalized-gradient inner products, ConFIG (Liu et al., 2025) solves $\hat{G}^\top w = \mathbf{1}_m$ using the pseudoinverse, where $\hat{G} = [\hat{g}_1, \ldots, \hat{g}_m]$, and normalizes the result as $v = w/\|w\|_2$. It then applies a positive scalar rescaling to form the final update. When the linear system $\hat{G}^\top w = \mathbf{1}_m$ is consistent, the normalized ConFIG direction coincides with the optimal direction of the following equality-constrained counterpart of (P):

$$\max_{v,\, r} \quad r \quad \text{s.t.} \quad \hat{g}_i^\top v = r,\ i = 1, \ldots, m, \quad \|v\|_2 \le 1. \tag{P$_=$}$$

By a change of variables, Problem (P$_=$) reduces to the pseudo-inverse minimum-norm system used by ConFIG. The detailed derivation is provided in Appendix B.2.

Theorem 4.1. *Let $(v^\star, r^\star)$ be an optimal solution of* (P)*, and let $(\tilde{v}, \tilde{r})$ be an optimal solution of its equality-constrained counterpart* (P$_=$)*. Then $r^\star \ge \tilde{r}$. Consequently, for every $i = 1, \ldots, m$, we have $\hat{g}_i^\top v^\star \ge \hat{g}_i^\top \tilde{v}$.*

Under the consistency condition above, Theorem 4.1 shows that, for every objective, our method dominates ConFIG with respect to the rate of change along the computed direction. Moreover, the improvement can be strictly larger when some inequality constraints in (P) are inactive at the optimum, as illustrated in the following example.

Example 4.2. Let $g_1 = c_1(1, 0, 0)$, $g_2 = c_2(0, 1, 0)$, and $g_3 = c_3(2, 2, 1)/3$, where $c_1, c_2, c_3 > 0$. Then the normalized gradients are $\hat{g}_1 = (1, 0, 0)$, $\hat{g}_2 = (0, 1, 0)$, and $\hat{g}_3 = (2, 2, 1)/3$. For (P$_=$), the optimal direction is $\tilde{v} = (1, 1, -1)/\sqrt{3}$, which gives rise to the ConFIG direction. In contrast, for (P), the optimal direction is $v^\star = (1, 1, 0)/\sqrt{2}$. Moreover, the corresponding inner products satisfy $\hat{g}_1^\top \tilde{v} = \hat{g}_2^\top \tilde{v} = \hat{g}_3^\top \tilde{v} = 1/\sqrt{3}$, whereas $\hat{g}_1^\top v^\star = \hat{g}_2^\top v^\star = 1/\sqrt{2}$ and $\hat{g}_3^\top v^\star = 2\sqrt{2}/3$. Therefore, our method yields strictly larger rates of change for all three objectives in this example. This illustrates that enforcing equalities for all inner products, as in ConFIG, can be more restrictive than the inequality-based formulation underlying our method.

We also note that in the two-objective case, the optimum of (P) necessarily activates both constraints, so our method and ConFIG coincide. Overall, ConFIG's property-driven construction imposes stronger conditions than necessary. In contrast, the Chebyshev-center geometry naturally recovers the intended properties in a relaxed form, where equality is required only for active constraints.

4.3 Comparison with minimum-norm methods based on the dual formulation

We next compare our dual formulation with MGDA and GAPO, which construct update directions by minimizing the norm of a convex combination of original or normalized gradients.

MGDA MGDA (Sener & Koltun, 2018) finds a minimum-norm convex combination of the original gradients by solving $\min_{\alpha \in \Delta^m} \| \sum_{i=1}^{m} \alpha_i g_i \|_2$. This formulation is closely related to our dual

formulation (D). The key difference is that MGDA uses the original gradients g_i, whereas our dual uses the normalized gradients $\hat{g}_i = g_i/\|g_i\|_2$. This distinction is particularly important for PINNs, where loss terms often have substantially different scales and the resulting imbalance can interfere with training (Wang et al., 2021a). Because MGDA is based on the original gradients, loss-scale imbalance can directly affect the selected direction. In contrast, our dual formulation determines the direction using normalized gradients, thereby removing the influence of gradient magnitudes. Such magnitude-independent direction selection is well suited to PINN training.

GAPO To reduce loss-scale sensitivity, GAPO (Li et al., 2025) solves $\min_{\alpha\in\Delta^m} \| \sum_{i=1}^m \alpha_i g_i/\|g_i\|_2^\rho\|_2$, where ρ controls the degree of gradient normalization. For $\rho = 1$, this problem coincides with our dual formulation (D) when $p = 2$. Consequently, at the same parameter point, the two formulations select the same direction before final scaling whenever the common minimum-norm combination is nonzero.

Our formulation derives this shared direction from the Chebyshev-center problem and applies the adaptive scalar defined above to construct the final update. Moreover, whereas ρ changes gradient normalization within a Euclidean minimum-norm problem, p determines the normalization, the primal and dual norms, and the primal recovery map in our formulation. Once p is fixed, no separate normalization exponent is required.

5 Numerical experiments

We evaluate the proposed method on five forward PDE benchmarks and an inverse coefficient-recovery problem under three data settings, comparing against Adam (Kingma & Ba, 2014), LRA, ReLoBRaLo, MGDA, PCGrad, IMTL-G, DCGD, ConFIG, and HARMONIC where applicable.

We run each experiment over three independent trials with different random seeds and report the mean and standard deviation of the final relative L^2 error. For forward problems, the error is computed on the solution, whereas for inverse problems, it is computed on the recovered coefficient.

5.1 Comparison on forward benchmark equations

We evaluate the ℓ_2-norm formulation of the proposed method on five forward PDE benchmarks solved with vanilla PINNs: two steady-state boundary-value problems (Helmholtz and 2D Kovasznay flow) and three time-dependent initial-boundary-value problems (Burgers' equation, the 2D varying-coefficient heat equation, and the Klein–Gordon equation). We compare it against the baselines listed above; implementation details are provided in Appendix D.2. Table 1 reports the forward results. In Tables 1 and 2, lower values are better; bold and underlined entries indicate the best and second-best results, respectively. A dash denotes an omitted entry: DCGD is omitted for three-term objectives, and ConFIG is omitted for two-term objectives where it coincides with ours.

Table 1: Final relative L^2 errors on five forward PDE benchmark problems. Each value reports the mean and standard deviation over three independent runs with different random seeds.

Method	Helmholtz	Kovasznay	Burgers'	Heat 2D VC	Klein–Gordon
Adam	0.0307 (0.0162)	0.0044 (0.0010)	0.1659 (0.0358)	2.9813 (0.6427)	0.0267 (0.0111)
LRA	1.0000 (0.0000)	0.0087 (0.0014)	0.4032 (0.0190)	0.9999 (0.0000)	1.0174 (0.0413)
ReLoBRaLo	0.0570 (0.0295)	0.0048 (0.0004)	0.1561 (0.0465)	2.6150 (0.4534)	0.0389 (0.0230)
MGDA	0.7542 (0.4058)	0.0095 (0.0014)	0.1258 (0.0114)	0.9370 (0.0219)	0.1494 (0.0478)
PCGrad	0.0075 (0.0014)	0.0057 (0.0012)	0.0706 (0.0046)	0.4185 (0.0829)	0.0466 (0.0194)
IMTL-G	0.0086 (0.0007)	0.0039 (0.0007)	0.2591 (0.0235)	1.0099 (0.0128)	0.0871 (0.0950)
DCGD	0.0097 (0.0061)	0.0052 (0.0025)	—	—	—
ConFIG	—	—	0.0708 (0.0191)	0.2401 (0.0359)	0.0339 (0.0068)
HARMONIC	0.0116 (0.0060)	0.0061 (0.0045)	0.0677 (0.0065)	0.2361 (0.0124)	**0.0143 (0.0031)**
Ours	**0.0072 (0.0033)**	**0.0017 (0.0005)**	**0.0599 (0.0042)**	**0.2316 (0.0190)**	0.0218 (0.0074)

Overall, these results demonstrate the superior empirical performance of the proposed method on both steady-state and time-dependent forward PDE problems.

5.2 COEFFICIENT RECOVERY IN AN INVERSE HEAT PROBLEM

We further evaluate the proposed method on an inverse heat problem, where the goal is to recover the unknown diffusion coefficient $a(x, y)$ from noisy observations of $u(x, y, t)$. We consider three data settings: Standard, Higher noise, and Fewer data. We compare against the same baselines listed above except DCGD, with further implementation details provided in Appendix D.3.

Figure 3 shows uniformly small reconstruction errors for our method, whereas Adam produces a distorted coefficient field with large structured errors. Table 2 shows strong coefficient-recovery performance across the tested data settings.

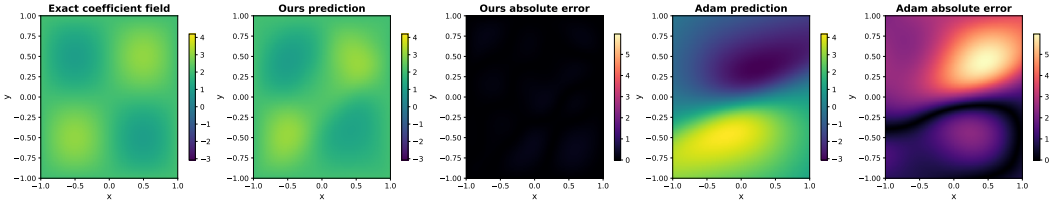


Figure 3: Inverse heat problem: ground-truth and reconstructed coefficient fields, with absolute errors for our method and Adam.

Table 2: Final relative L^2 errors of the recovered coefficient a for the inverse heat problem. Each value reports the mean and standard deviation over three independent runs.

Method	Standard	Higher noise	Fewer data
Adam	6.1510 (3.7224)	6.1409 (3.6977)	6.0555 (3.6322)
LRA	0.2390 (0.0000)	0.2390 (0.0000)	0.2390 (0.0000)
ReLoBRaLo	3.7730 (2.1329)	3.8501 (2.0031)	3.0611 (1.0022)
MGDA	0.2389 (0.0001)	0.2389 (0.0000)	0.2389 (0.0000)
PCGrad	0.1371 (0.0529)	0.1272 (0.0494)	0.1247 (0.0633)
IMTL-G	0.0599 (0.0005)	0.1069 (0.0085)	0.0686 (0.0026)
ConFIG	0.0612 (0.0024)	0.1038 (0.0042)	0.0655 (0.0034)
HARMONIC	0.0596 (0.0014)	**0.0988 (0.0110)**	0.0694 (0.0008)
Ours	**0.0593 (0.0002)**	0.1040 (0.0049)	**0.0631 (0.0026)**

Appendix E provides comparisons with GAPO, norm-choice analysis and ablations, multi-model adversarial attacks, direct updates without Adam, and computational cost analyses. In particular, the norm ablation shows problem-dependent benefits from $p \neq 2$, and $p = 1$ outperforms both tested $p = 2$ variants in attacks under an ℓ_∞ constraint, illustrating the practical value of the general norm formulation.

6 CONCLUSIONS

We proposed a Chebyshev center-based direction selection framework for multi-loss PINN training. We formulated update direction selection as finding a central direction that maximizes the minimum distance to the dual cone facets under a general ℓ_p/ℓ_q geometry. We derived its low-dimensional dual problem, showed how to recover the corresponding primal update direction, and obtained an efficient algorithm with a convergence guarantee in the nonconvex setting. The geometric interpretation and primal–dual formulations also enabled a systematic comparison with representative direction-selection methods, clarifying how our approach relates to existing constructions. Experiments on forward and inverse PINN benchmarks demonstrated the effectiveness of the proposed method.

Future work includes evaluating scalability with a large number of objectives and studying how mini-batch noise affects gradient normalization and direction selection. Beyond PINNs and adversarial attacks, the dual-cone and Chebyshev-center perspectives may also be useful for multi-task learning and multi-objective LLM alignment.

REFERENCES

Rafael Bischof and Michael A Kraus. Multi-objective loss balancing for physics-informed deep learning. *Computer Methods in Applied Mechanics and Engineering*, 439:117914, 2025.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Dohyun Bu, Yujung Byun, and Jong-Seok Lee. Harmonized cone for feasible and non-conflict directions in training physics-informed neural networks. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=PRYl1mO1go.

Dmitry Bylinkin, Mikhail Aleksandrov, Savelii Chezhegov, and Aleksandr Beznosikov. Enhancing stability of physics-informed neural network training through saddle-point reformulation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=EQNp3sFrY3.

Peter L Chen, Xiaopeng Li, Xi Chen, and Tianyi Lin. Reward-free alignment for conflicting objectives. *arXiv preprint arXiv:2602.02495*, 2026.

Junwoo Cho, Seungtae Nam, Hyunmo Yang, Seok-Bae Yun, Youngjoon Hong, and Eunbyung Park. Separable physics-informed neural networks. *Advances in Neural Information Processing Systems*, 36:23761–23788, 2023.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pp. 2206–2216. PMLR, 2020.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*, 2020.

Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HkeryxBtPB.

Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical methods of operations research*, 51(3):479–494, 2000.

Zhiwei Gao, Liang Yan, and Tao Zhou. Failure-informed adaptive sampling for pinns. *SIAM Journal on Scientific Computing*, 45(4):A1971–A1994, 2023.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Zhongkai Hao, Jiachen Yao, Chang Su, Hang Su, Ziao Wang, Fanzhi Lu, Zeyu Xia, Yichi Zhang, Songming Liu, Lu Lu, et al. Pinnacle: A comprehensive benchmark of physics-informed neural networks for solving pdes. *Advances in Neural Information Processing Systems*, 37:76721–76774, 2024.

Youngsik Hwang and Dong-Young Lim. Dual cone gradient descent for training physics-informed neural networks. *Advances in Neural Information Processing Systems*, 37:98563–98595, 2024.

Ameya D Jagtap and George Em Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5), 2020.

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

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in neural information processing systems*, 34:26548–26560, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Chengao Li, Hanyu Zhang, Yunkun Xu, Hongyan Xue, Xiang Ao, and Qing He. Gradient-adaptive policy optimization: Towards multi-objective alignment of large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11214–11232, 2025.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in neural information processing systems*, 34:18878–18890, 2021a.

Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. In *International conference on learning representations*, 2021b.

Qiang Liu, Mengyu Chu, and Nils Thuerey. ConFIG: Towards conflict-free training of physics informed neural networks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=APojAzJQiq.

Mohammad Amin Nabian, Rini Jasmine Gladstone, and Hadi Meidani. Efficient training of physics-informed neural networks via importance sampling. *Computer-Aided Civil and Infrastructure Engineering*, 36(8):962–977, 2021.

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.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

Kejun Tang, Xiaoliang Wan, and Chao Yang. Das-pinns: A deep adaptive sampling method for solving high-dimensional partial differential equations. *Journal of Computational Physics*, 476: 111868, 2023.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021a.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 384:113938, 2021b.

Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why pinns fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJx040EFvH.

Chenxi Wu, Min Zhu, Qinyang Tan, Yadhu Kartha, and Lu Lu. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 403:115671, 2023.

Jiachen Yao, Chang Su, Zhongkai Hao, Songming Liu, Hang Su, and Jun Zhu. Multiadam: Parameter-wise scale-invariant optimizer for multiscale training of physics-informed neural networks. In *International conference on machine learning*, pp. 39702–39721. PMLR, 2023.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in neural information processing systems*, 33: 5824–5836, 2020.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. *Advances in neural information processing systems*, 32, 2019.

Xiaoyuan Zhang, Xi Lin, and Qingfu Zhang. Pmgda: A preference-based multiple gradient descent algorithm. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2025.

Zhiyuan Zhao, Xueying Ding, and B. Aditya Prakash. PINNsformer: A transformer-based framework for physics-informed neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=DO2WFXU1Be.

Zixian Zhou, Mengda Huang, Feiyang Pan, Jia He, Xiang Ao, Dandan Tu, and Qing He. Gradient-adaptive pareto optimization for constrained reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11443–11451, 2023.

A Additional detail and proofs for section 3

A.1 Proof of proposition 3.2

Recall the primal problem

$$\begin{aligned} \max_{v,r} \quad & r \\ \text{s.t.} \quad & (\hat{g}_i^{(p)})^\top v \geq r, \quad i = 1, \ldots, m, \\ & \|v\|_q \leq 1. \end{aligned} \tag{P}$$

We first derive its dual. Rewriting the constraints as

$$(\hat{g}_i^{(p)})^\top v - r \geq 0, \quad i = 1, \ldots, m, \qquad 1 - \|v\|_q \geq 0,$$

the Lagrangian is

$$\mathcal{L}(v, r, \alpha, \beta) = r + \sum_{i=1}^{m} \alpha_i \left((\hat{g}_i^{(p)})^\top v - r \right) + \beta(1 - \|v\|_q),$$

where $\alpha_i \geq 0$ for all i and $\beta \geq 0$. Rearranging terms gives

$$\mathcal{L}(v, r, \alpha, \beta) = r\Big(1 - \sum_{i=1}^{m} \alpha_i\Big) + \Big(\sum_{i=1}^{m} \alpha_i \hat{g}_i^{(p)}\Big)^\top v - \beta\|v\|_q + \beta.$$

Hence the dual function is

$$\phi(\alpha, \beta) = \sup_{v,r} \mathcal{L}(v, r, \alpha, \beta).$$

For the supremum over r to be finite, the coefficient of r must vanish. Therefore,

$$\sum_{i=1}^{m} \alpha_i = 1.$$

Otherwise, if $1 - \sum_i \alpha_i > 0$, then letting $r \to +\infty$ yields $\phi(\alpha, \beta) = +\infty$, and if $1 - \sum_i \alpha_i < 0$, then letting $r \to -\infty$ yields $\phi(\alpha, \beta) = +\infty$.

Next, we examine when the supremum over v is finite. Define

$$w := \sum_{i=1}^{m} \alpha_i \hat{g}_i^{(p)}.$$

Then the part of the Lagrangian involving v is

$$w^\top v - \beta \|v\|_q.$$

By Hölder's inequality,

$$w^\top v \le \|w\|_p \|v\|_q.$$

Thus,

$$w^\top v - \beta \|v\|_q \le (\|w\|_p - \beta)\|v\|_q.$$

Therefore, if $\|w\|_p > \beta$, then there exists s with $\|s\|_q = 1$ and $w^\top s > \beta$. Hence, along $v = \tau s$,

$$w^\top v - \beta \|v\|_q = \tau(w^\top s - \beta) \to +\infty,$$

so the supremum over v is $+\infty$. Thus, for the dual function to be finite, it is necessary that

$$\|w\|_p \le \beta.$$

Under this condition, we now compute the value of the supremum. If $\|w\|_p \le \beta$, then

$$w^\top v - \beta \|v\|_q \le 0$$

for all v, and equality is attained at $v = 0$. Hence

$$\sup_v \big(w^\top v - \beta \|v\|_q\big) = 0 \qquad \text{whenever } \|w\|_p \le \beta.$$

Therefore, the dual function is

$$\phi(\alpha, \beta) = \beta$$

subject to

$$\sum_{i=1}^{m} \alpha_i = 1, \qquad \Big\| \sum_{i=1}^{m} \alpha_i \hat{g}_i^{(p)} \Big\|_p \le \beta, \qquad \alpha_i \ge 0,\ \beta \ge 0.$$

Minimizing over β yields

$$\beta^\star = \Big\| \sum_{i=1}^{m} \alpha_i \hat{g}_i^{(p)} \Big\|_p,$$

and thus the dual problem reduces to

$$\begin{aligned} \min_{\alpha} \quad & \Big\| \sum_{i=1}^{m} \alpha_i \hat{g}_i^{(p)} \Big\|_p \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i = 1, \\ & \alpha_i \ge 0, \quad i = 1, \dots, m. \end{aligned} \tag{D}$$

It remains to show strong duality. The primal (P) can be written as the minimization of $-r$ subject to

$$r - (\hat{g}_i^{(p)})^\top v \le 0, \qquad i = 1, \dots, m,$$

and

$$\|v\|_q \le 1.$$

The objective $-r$ is linear, each inequality $r - (\hat{g}_i^{(p)})^\top v \le 0$ is affine in (v, r), and the constraint $\|v\|_q \le 1$ defines a convex set. Hence, (P) is a convex optimization problem in the equivalent minimization form. Moreover, Slater's condition holds, because $(v, r) = (0, -1)$ is strictly feasible:

$$r - (\hat{g}_i^{(p)})^\top v = -1 < 0, \quad i = 1, \dots, m, \qquad \|v\|_q = 0 < 1.$$

Therefore, by Slater's theorem, strong duality holds between (P) and (D). □

A.2 PROOF OF PROPOSITION 3.3

Let $(v^\star, r^\star)$ be an optimal solution of the primal (P), and let $\alpha^\star$ be an optimal solution of the dual (D). Define

$$w^\star := \sum_{i=1}^{m} \alpha_i^\star \hat{g}_i^{(p)},$$

and let $\beta^\star := \|w^\star\|_p$.

Assume that $r^\star > 0$. Then $v^\star \neq 0$, since if $v^\star = 0$, feasibility of $(v^\star, r^\star)$ would imply

$$(\hat{g}_i^{(p)})^\top v^\star = 0 \geq r^\star$$

for all i, and hence $r^\star \leq 0$, a contradiction.

From the proof of Proposition 3.2, the Lagrangian is

$$\mathcal{L}(v, r, \alpha, \beta) = r + \sum_{i=1}^{m} \alpha_i \big((\hat{g}_i^{(p)})^\top v - r\big) + \beta(1 - \|v\|_q).$$

Since $1 < q < \infty$, the ℓ_q norm is differentiable at $v^\star \neq 0$. Therefore, the KKT stationarity condition with respect to v gives

$$\sum_{i=1}^{m} \alpha_i^\star \hat{g}_i^{(p)} - \beta^\star \nabla_v \|v^\star\|_q = 0.$$

Equivalently,

$$w^\star = \beta^\star \nabla_v \|v^\star\|_q. \tag{1}$$

We next show that $\|v^\star\|_q = 1$. By strong duality, the primal and dual optimal values coincide. From Proposition 3.2, the dual optimal value is $\beta^\star$, while the primal optimal value is $r^\star$. Hence

$$\beta^\star = r^\star.$$

Under the assumption $r^\star > 0$, we therefore have $\beta^\star > 0$ and $w^\star \neq 0$.

Now consider the complementary slackness condition associated with the constraint $\|v\|_q \leq 1$:

$$\beta^\star(1 - \|v^\star\|_q) = 0.$$

Since $\beta^\star > 0$, it follows that

$$\|v^\star\|_q = 1. \tag{2}$$

Since the gradient of the ℓ_q norm at $v^\star$ is

$$\nabla_v \|v^\star\|_q = \frac{\operatorname{sgn}(v^\star) \odot |v^\star|^{q-1}}{\|v^\star\|_q^{q-1}},$$

where the operations are applied componentwise, and $\|v^\star\|_q = 1$ by (2), equation (1) becomes

$$w^\star = \beta^\star \big(\operatorname{sgn}(v^\star) \odot |v^\star|^{q-1}\big). \tag{3}$$

Therefore, componentwise,

$$v_j^\star = \operatorname{sgn}(w_j^\star) \left(\frac{|w_j^\star|}{\beta^\star}\right)^{1/(q-1)}.$$

Since $q = p/(p-1)$, we have $1/(q-1) = p - 1$, and hence

$$v_j^\star = \frac{\operatorname{sgn}(w_j^\star)|w_j^\star|^{p-1}}{(\beta^\star)^{p-1}}.$$

Since $\beta^\star = \|w^\star\|_p$, this gives

$$v^\star = \frac{\operatorname{sgn}(w^\star) \odot |w^\star|^{p-1}}{\|w^\star\|_p^{p-1}}.$$

This completes the proof. □

A.3 THE $p = 1$ AND $p = \infty$ CASES FOR THE RECOVERY STEP

The main text focuses on $1 < p < \infty$, where the primal direction is uniquely recovered from the dual solution in closed form, as shown in Proposition 3.3. The cases $p = 1$ and $p = \infty$ are also well defined for both the primal and the dual problem. The difference is that the recovery step is no longer given by a closed-form map from the dual solution to a unique primal direction. Indeed, when $p = 1$ or $p = \infty$, the support direction associated with a dual vector can be non-unique, because the corresponding unit ℓ_q ball has flat faces.

Given an optimal dual solution $\alpha^\star$, define

$$w^\star := \sum_{i=1}^{m} \alpha_i^\star \hat{g}_i^{(p)}, \qquad r^\star := \|w^\star\|_p.$$

By strong duality, $r^\star$ is the optimal value of the primal problem. Hence, a valid primal optimal direction can be recovered by finding $v^\star$ such that

$$(\hat{g}_i^{(p)})^\top v^\star \geq r^\star, \qquad i = 1, \dots, m,$$

and

$$\|v^\star\|_q \leq 1.$$

Any $v^\star$ satisfying these conditions is primal optimal, since it attains the optimal value $r^\star$. Thus, in the $p = 1$ and $p = \infty$ cases, the dual solution provides the optimal radius $r^\star$, while the corresponding primal direction can be selected from the set of directions that attain this radius.

For $p = 1$, we have $q = \infty$. In this case, the norm constraint becomes

$$\|v\|_\infty \leq 1 \quad \Longleftrightarrow \quad -1 \leq v_j \leq 1, \qquad j = 1, \dots, n.$$

Therefore, the recovery step can be written as the linear feasibility problem

$$(\hat{g}_i^{(1)})^\top v \geq r^\star, \qquad i = 1, \dots, m,$$

$$-1 \leq v_j \leq 1, \qquad j = 1, \dots, n.$$

Equivalently, $v^\star$ can be obtained directly by solving the primal problem (P) as the following linear program:

$$\begin{aligned} \max_{v,r} \quad & r \\ \text{s.t.} \quad & (\hat{g}_i^{(1)})^\top v \geq r, \quad i = 1, \dots, m, \\ & -1 \leq v_j \leq 1, \quad j = 1, \dots, n. \end{aligned}$$

For $p = \infty$, we have $q = 1$. The recovery step is

$$(\hat{g}_i^{(\infty)})^\top v \geq r^\star, \qquad i = 1, \dots, m, \qquad \|v\|_1 \leq 1.$$

Introducing auxiliary variables $z_j \geq 0$, the ℓ_1-constraint can be written as

$$-z_j \leq v_j \leq z_j, \qquad j = 1, \dots, n,$$

$$\sum_{j=1}^{n} z_j \leq 1.$$

Thus, the recovery step becomes the linear feasibility problem

$$(\hat{g}_i^{(\infty)})^\top v \geq r^\star, \qquad i = 1, \dots, m,$$

$$-z_j \leq v_j \leq z_j, \qquad j = 1, \dots, n,$$

$$\sum_{j=1}^{n} z_j \leq 1, \qquad z_j \geq 0, \quad j = 1, \dots, n.$$

Similarly, $v^\star$ can be obtained directly by solving the primal problem (P) as the following linear program:

$$\begin{aligned} \max_{v,r,z} \quad & r \\ \text{s.t.} \quad & (\hat{g}_i^{(\infty)})^\top v \geq r, \quad i = 1, \dots, m, \\ & -z_j \leq v_j \leq z_j, \quad j = 1, \dots, n, \\ & \sum_{j=1}^{n} z_j \leq 1, \\ & z_j \geq 0, \quad j = 1, \dots, n. \end{aligned}$$

These formulations show that the $p = 1$ and $p = \infty$ cases can be handled through LP-based direction selection. However, unlike the $1 < p < \infty$ case, the primal direction is not recovered from $w^\star$ by a single closed-form map, and the LP formulation may still have multiple optimal directions. When this happens, the Chebyshev-center criterion alone does not select a unique direction, so an additional selection criterion may be required.

A.4 Proof of proposition 3.6

Let $(v^\star, r^\star)$ be an optimal solution of the primal (P) at θ, and assume that

$$r^\star = 0.$$

Let $\alpha^\star$ be an optimal solution of the dual (D). By strong duality from Proposition 3.2, the primal and dual optimal values coincide. Hence,

$$\left\| \sum_{i=1}^{m} \alpha_i^\star \hat{g}_i^{(p)} \right\|_p = 0,$$

which implies

$$\sum_{i=1}^{m} \alpha_i^\star \hat{g}_i^{(p)} = 0.$$

Since $\hat{g}_i^{(p)} = g_i / \|g_i\|_p$, where $g_i = \nabla \mathcal{L}_i(\theta)$, we obtain

$$\sum_{i=1}^{m} \frac{\alpha_i^\star}{\|g_i\|_p} \nabla \mathcal{L}_i(\theta) = 0.$$

Now define

$$\bar{\alpha}_i := \frac{\alpha_i^\star / \|g_i\|_p}{\sum_{j=1}^{m} \alpha_j^\star / \|g_j\|_p}, \qquad i = 1, \dots, m.$$

Since $\alpha_i^\star \geq 0$ and $\sum_{i=1}^{m} \alpha_i^\star = 1$, we have $\bar{\alpha}_i \geq 0$ for all i and

$$\sum_{i=1}^{m} \bar{\alpha}_i = 1.$$

Moreover,

$$\sum_{i=1}^{m} \bar{\alpha}_i \nabla \mathcal{L}_i(\theta) = \frac{1}{\sum_{j=1}^{m} \alpha_j^\star / \|g_j\|_p} \sum_{i=1}^{m} \frac{\alpha_i^\star}{\|g_i\|_p} \nabla \mathcal{L}_i(\theta) = 0.$$

Therefore, there exists $\bar{\alpha} \in \mathbb{R}_+^m$ such that

$$\sum_{i=1}^{m} \bar{\alpha}_i = 1 \qquad \text{and} \qquad \sum_{i=1}^{m} \bar{\alpha}_i \nabla \mathcal{L}_i(\theta) = 0.$$

By Definition 3.5, θ is Pareto-stationary. □

A.5 SOLVING THE DUAL PROBLEMS FOR $p = 1$ AND $p = \infty$

The dual problem (D) remains convex at the cases $p = 1$ and $p = \infty$, but the objective is nonsmooth. Instead of applying the Frank–Wolfe step in Algorithm 1 directly, these dual problems can be written as linear programs.

For $p = 1$, the dual problem is

$$\min_{\alpha \in \Delta^m} \left\| \sum_{i=1}^{m} \alpha_i \hat{g}_i^{(1)} \right\|_1 .$$

Using auxiliary variables $z_j \geq 0$, this is equivalent to

$$\begin{aligned} \min_{\alpha, z} \quad & \sum_{j=1}^{n} z_j \\ \text{s.t.} \quad & -z_j \leq \left(\sum_{i=1}^{m} \alpha_i \hat{g}_i^{(1)} \right)_j \leq z_j, \quad j = 1, \ldots, n, \\ & \sum_{i=1}^{m} \alpha_i = 1, \\ & \alpha_i \geq 0, \quad i = 1, \ldots, m, \\ & z_j \geq 0, \quad j = 1, \ldots, n. \end{aligned}$$

The optimal value is then

$$r^\star = \sum_{j=1}^{n} z_j^\star .$$

For $p = \infty$, the dual problem is

$$\min_{\alpha \in \Delta^m} \left\| \sum_{i=1}^{m} \alpha_i \hat{g}_i^{(\infty)} \right\|_\infty .$$

Introducing a scalar variable τ, this is equivalent to

$$\begin{aligned} \min_{\alpha, \tau} \quad & \tau \\ \text{s.t.} \quad & -\tau \leq \left(\sum_{i=1}^{m} \alpha_i \hat{g}_i^{(\infty)} \right)_j \leq \tau, \quad j = 1, \ldots, n, \\ & \sum_{i=1}^{m} \alpha_i = 1, \\ & \alpha_i \geq 0, \quad i = 1, \ldots, m, \\ & \tau \geq 0. \end{aligned}$$

The optimal value is

$$r^\star = \tau^\star .$$

Thus, in both $p = 1$ and $p = \infty$ cases, an optimal dual solution $\alpha^\star$ and the corresponding optimal value $r^\star$ can be obtained by solving a linear program.

A.6 PROOF OF THEOREM 3.8

For each iteration t, let α_t be an optimal solution of the dual (D) at θ_t, and define

$$w_t := \sum_{i=1}^{m} \alpha_{t,i} \hat{g}_i^{(p)}(\theta_t), \qquad v_t := \frac{\operatorname{sgn}(w_t) \odot |w_t|^{p-1}}{\|w_t\|_p^{p-1}}, \qquad d_t := \Big(\sum_{i=1}^{m} g_i(\theta_t)^\top v_t \Big) v_t .$$

Also let

$$\nabla \mathcal{L}(\theta_t) = \sum_{i=1}^{m} g_i(\theta_t),$$

and let $r_t^\star$ denote the optimal value of the primal (P) at θ_t.

If Algorithm 1 terminates at some $t \in \{0, \dots, T-1\}$, then it returns an ϵ-Pareto-stationary point by construction. We therefore consider the remaining case where the algorithm does not terminate for any $t = 0, \dots, T-1$. Then

$$\|w_t\|_p > \epsilon \qquad \text{for all } t = 0, \dots, T-1.$$

By strong duality, $\|w_t\|_p = r_t^\star$. Hence

$$r_t^\star > \epsilon \qquad \text{for all } t = 0, \dots, T-1. \tag{1}$$

We first establish the auxiliary relations used below.

Step 1: Lower bound on $\|d_t\|_q$.

Since α_t is optimal for the dual (D) and $r_t^\star > 0$ by (1), Proposition 3.3 implies that v_t is the corresponding optimal primal direction for (P). Therefore,

$$\big(\hat{g}_i^{(p)}(\theta_t)\big)^\top v_t \geq r_t^\star, \qquad i = 1, \dots, m.$$

Multiplying both sides by $\|g_i(\theta_t)\|_p$ yields

$$g_i(\theta_t)^\top v_t \geq r_t^\star \|g_i(\theta_t)\|_p, \qquad i = 1, \dots, m.$$

Summing over i, we obtain

$$\sum_{i=1}^{m} g_i(\theta_t)^\top v_t \geq \sum_{i=1}^{m} r_t^\star \|g_i(\theta_t)\|_p = r_t^\star \sum_{i=1}^{m} \|g_i(\theta_t)\|_p \geq r_t^\star \Big\| \sum_{i=1}^{m} g_i(\theta_t) \Big\|_p = r_t^\star \|\nabla \mathcal{L}(\theta_t)\|_p,$$

where the second inequality follows from the triangle inequality. Since $\|v_t\|_q = 1$ and $\sum_{i=1}^m g_i(\theta_t)^\top v_t \geq 0$, it follows that

$$\|d_t\|_q = \|\Big(\sum_{i=1}^{m} g_i(\theta_t)^\top v_t\Big) v_t\|_q = \sum_{i=1}^{m} g_i(\theta_t)^\top v_t \geq r_t^\star \|\nabla \mathcal{L}(\theta_t)\|_p.$$

Combining this with (1) gives

$$\|d_t\|_q^2 \geq \epsilon^2 \|\nabla \mathcal{L}(\theta_t)\|_p^2. \tag{2}$$

We next bound the Pareto-stationarity measure using the same update direction. Since $\alpha_t \in \Delta^m$ and all gradient norms are positive, the coefficients

$$\frac{\alpha_{t,i} / \|g_i(\theta_t)\|_p}{\sum_{j=1}^m \alpha_{t,j} / \|g_j(\theta_t)\|_p}, \qquad i = 1, \dots, m,$$

are nonnegative and sum to one. Thus, by the definition of S_p,

$$S_p(\theta_t) \leq \left\| \frac{\sum_{i=1}^m \frac{\alpha_{t,i}}{\|g_i(\theta_t)\|_p} g_i(\theta_t)}{\sum_{i=1}^m \frac{\alpha_{t,i}}{\|g_i(\theta_t)\|_p}} \right\|_p = \frac{r_t^\star}{\sum_{i=1}^m \alpha_{t,i} / \|g_i(\theta_t)\|_p},$$

where we used $\|w_t\|_p = r_t^\star$. Moreover, $\sum_i \alpha_{t,i} = 1$ implies

$$\sum_{i=1}^{m} \frac{\alpha_{t,i}}{\|g_i(\theta_t)\|_p} \geq \frac{1}{\max_i \|g_i(\theta_t)\|_p}.$$

Combining these inequalities with the lower bound on $\|d_t\|_q$ established above yields

$$S_p(\theta_t) \leq r_t^\star \max_i \|g_i(\theta_t)\|_p \leq r_t^\star \sum_{i=1}^{m} \|g_i(\theta_t)\|_p \leq \|d_t\|_q. \tag{3}$$

Step 2: Relation between $\nabla\mathcal{L}(\theta_t)$ and d_t.

By definition,

$$\nabla\mathcal{L}(\theta_t)^\top d_t = \Big(\sum_{i=1}^m g_i(\theta_t)\Big)^\top \Big(\sum_{i=1}^m g_i(\theta_t)^\top v_t\Big) v_t = \Big(\sum_{i=1}^m g_i(\theta_t)^\top v_t\Big)^2.$$

On the other hand, since $\|v_t\|_q = 1$,

$$\|d_t\|_q^2 = \|\Big(\sum_{i=1}^m g_i(\theta_t)^\top v_t\Big) v_t\|_q^2 = \Big(\sum_{i=1}^m g_i(\theta_t)^\top v_t\Big)^2.$$

Therefore,

$$\nabla\mathcal{L}(\theta_t)^\top d_t = \|d_t\|_q^2. \tag{4}$$

Step 3: Descent estimate and final bounds.

By the β-smoothness of $\mathcal{L}$ and the update $\theta_{t+1} = \theta_t - \eta d_t$,

$$\mathcal{L}(\theta_{t+1}) \le \mathcal{L}(\theta_t) - \eta\nabla\mathcal{L}(\theta_t)^\top d_t + \frac{\beta\eta^2}{2}\|d_t\|_q^2.$$

Using (4) and setting $\eta = 1/\beta$, we obtain

$$\mathcal{L}(\theta_{t+1}) \le \mathcal{L}(\theta_t) - \frac{1}{2\beta}\|d_t\|_q^2. \tag{5}$$

Summing (5) over $t = 0, \dots, T-1$ yields

$$\sum_{t=0}^{T-1} \|d_t\|_q^2 \le 2\beta\big(\mathcal{L}(\theta_0) - \mathcal{L}(\theta_T)\big) \le 2\beta\big(\mathcal{L}(\theta_0) - \mathcal{L}(\theta^\star)\big).$$

By (3), it follows that

$$\begin{aligned}\min_{0\le t<T} S_p(\theta_t)^2 &\le \frac{1}{T}\sum_{t=0}^{T-1} S_p(\theta_t)^2 \\ &\le \frac{1}{T}\sum_{t=0}^{T-1} \|d_t\|_q^2 \\ &\le \frac{2\beta\big(\mathcal{L}(\theta_0) - \mathcal{L}(\theta^\star)\big)}{T}.\end{aligned}$$

Similarly, applying (2) gives

$$\begin{aligned}\min_{0\le t<T} \|\nabla\mathcal{L}(\theta_t)\|_p^2 &\le \frac{1}{T}\sum_{t=0}^{T-1} \|\nabla\mathcal{L}(\theta_t)\|_p^2 \\ &\le \frac{1}{\epsilon^2 T}\sum_{t=0}^{T-1} \|d_t\|_q^2 \\ &\le \frac{2\beta\big(\mathcal{L}(\theta_0) - \mathcal{L}(\theta^\star)\big)}{\epsilon^2 T}.\end{aligned}$$

This proves both claims. □

B Additional detail and proof for section 4

B.1 Details of the comparative toy example

We consider a three-objective toy example in $\mathbb{R}^3$. Let

$$g_1 = (5, 0, 0), \qquad g_2 = (0, 3, 0), \qquad g_3 = \left(\frac{1}{5}, \frac{14}{15}, \frac{2\sqrt{5}}{15}\right).$$

Then $\|g_1\|_2 = 5$, $\|g_2\|_2 = 3$, and $\|g_3\|_2 = 1$. The normalized gradients are

$$\hat{g}_1 = (1, 0, 0), \qquad \hat{g}_2 = (0, 1, 0), \qquad \hat{g}_3 = \left(\frac{1}{5}, \frac{14}{15}, \frac{2\sqrt{5}}{15}\right).$$

MGDA solves

$$\min_{\alpha \in \Delta^3} \left\| \sum_{i=1}^{3} \alpha_i g_i \right\|_2^2.$$

For the gradients in this example, the solution is

$$\alpha_{\text{MGDA}} = (0, 0, 1),$$

and hence

$$v_{\text{MGDA}} = g_3 = \left(\frac{1}{5}, \frac{14}{15}, \frac{2\sqrt{5}}{15}\right).$$

Although $v_{\text{MGDA}} \in \mathbf{K}^*$, it lies close to the facet of the dual cone. This shows that MGDA is affected by gradient-scale imbalance, causing the selected direction to be biased toward g_3.

For IMTL-G, using its closed-form coefficient rule yields

$$\alpha_{\text{IMTL-G}} = \left(-\frac{13}{22}, -\frac{20}{11}, \frac{75}{22}\right).$$

The resulting aggregated direction before normalization is

$$\sum_{i=1}^{3} \alpha_i g_i = \left(-\frac{25}{11}, -\frac{25}{11}, \frac{5\sqrt{5}}{11}\right).$$

After normalization, this gives

$$v_{\text{IMTL-G}} = \left(-\sqrt{\frac{5}{11}}, -\sqrt{\frac{5}{11}}, \frac{1}{\sqrt{11}}\right).$$

Therefore, the normalized inner products are

$$\hat{g}_1^\top v_{\text{IMTL-G}} = \hat{g}_2^\top v_{\text{IMTL-G}} = \hat{g}_3^\top v_{\text{IMTL-G}} = -\sqrt{\frac{5}{11}}.$$

Thus $v_{\text{IMTL-G}} \notin \mathbf{K}^*$.

For ConFIG, the direction enforces equal normalized inner products,

$$\hat{g}_1^\top v = \hat{g}_2^\top v = \hat{g}_3^\top v.$$

Solving these equalities and normalizing the direction gives

$$v_{\text{ConFIG}} = \left(\sqrt{\frac{5}{11}}, \sqrt{\frac{5}{11}}, -\frac{1}{\sqrt{11}}\right).$$

Indeed,

$$\hat{g}_1^\top v_{\text{ConFIG}} = \hat{g}_2^\top v_{\text{ConFIG}} = \hat{g}_3^\top v_{\text{ConFIG}} = \sqrt{\frac{5}{11}}.$$

Therefore,

$$\tilde{r} = \sqrt{\frac{5}{11}} \approx 0.674.$$

Our method solves

$$\max_{\|v\|_2 \leq 1} \min_{i=1,2,3} \hat{g}_i^\top v.$$

In this example, the solution is

$$v_{\text{ours}} = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right).$$

The normalized inner products are

$$\hat{g}_1^\top v_{\text{ours}} = \hat{g}_2^\top v_{\text{ours}} = \frac{1}{\sqrt{2}}, \qquad \hat{g}_3^\top v_{\text{ours}} = \frac{17}{15\sqrt{2}}.$$

Hence

$$r^\star = \frac{1}{\sqrt{2}} \approx 0.707.$$

The third constraint is inactive because

$$\hat{g}_3^\top v_{\text{ours}} = \frac{17}{15\sqrt{2}} > \frac{1}{\sqrt{2}}.$$

Therefore,

$$\tilde{r} = \sqrt{\frac{5}{11}} < \frac{1}{\sqrt{2}} = r^\star.$$

This illustrates that enforcing equality for all objectives can be more restrictive than the max–min formulation.

B.2 Derivation of the equivalence between ConFIG and the equality-constrained problem

Assume that the linear system $\hat{G}^\top w = \mathbf{1}_m$ is consistent, and let w_0 be a solution. Then $(v, r) = (w_0/\|w_0\|_2,\ 1/\|w_0\|_2)$ is feasible for $(P_=)$ with $r > 0$. Hence, its optimal value is positive, and it suffices to consider feasible pairs with $r > 0$. For any such pair (v, r), define $w = v/r$. Then the equality constraints become

$$\hat{g}_i^\top w = 1, \qquad i = 1, \dots, m,$$

or equivalently,

$$\hat{G}^\top w = \mathbf{1}_m, \qquad \hat{G} = [\hat{g}_1, \hat{g}_2, \dots, \hat{g}_m].$$

Moreover, since $v = rw$ and $\|v\|_2 \le 1$, we have

$$r\|w\|_2 = \|v\|_2 \le 1,$$

which implies

$$r \le \frac{1}{\|w\|_2}.$$

Therefore, maximizing r in (P$_=$) is equivalent to minimizing $\|w\|_2$ subject to $\hat{G}^\top w = \mathbf{1}_m$, that is,

$$\min_w \|w\|_2 \qquad \text{s.t.} \quad \hat{G}^\top w = \mathbf{1}_m.$$

The minimum-norm solution of this system is given by

$$w^\star = (\hat{G}^\top)^+ \mathbf{1}_m.$$

Hence, the optimal direction of (P$_=$) is obtained by normalizing $w^\star$:

$$v^\star = \frac{w^\star}{\|w^\star\|_2}.$$

Therefore, the normalized direction used by ConFIG coincides with the optimal direction of the equality-constrained counterpart of our primal (P).

B.3 PROOF OF THEOREM 4.1

Let $\mathcal{F}$ and $\mathcal{F}_=$ denote the feasible sets of Problems (P) and (P$_=$), respectively.

Since every equality implies the corresponding inequality, we have

$$\mathcal{F}_= \subseteq \mathcal{F}.$$

Therefore, (P) is a relaxation of Problem (P$_=$). Since both problems maximize the same objective r, the optimal value over the larger feasible set cannot be smaller than that over the smaller one. Hence,

$$r^\star \geq \tilde{r}.$$

By feasibility of $(v^\star, r^\star)$ for (P), we have

$$\hat{g}_i^\top v^\star \geq r^\star, \qquad i = 1, \ldots, m.$$

Combining this with $r^\star \geq \tilde{r}$, we obtain

$$\hat{g}_i^\top v^\star \geq \tilde{r}, \qquad i = 1, \ldots, m.$$

Since $(\tilde{v}, \tilde{r})$ is feasible for Problem (P$_=$), its equality constraints imply

$$\hat{g}_i^\top \tilde{v} = \tilde{r}, \qquad i = 1, \ldots, m.$$

Therefore,

$$\hat{g}_i^\top v^\star \geq \hat{g}_i^\top \tilde{v}, \qquad i = 1, \ldots, m.$$

This completes the proof. □

C PSEUDOCODE FOR ALGORITHMS

Algorithm 2 Exact Dual Solver for Two Loss Terms

Require: ℓ_2-normalized gradients $\hat{g}_1, \hat{g}_2 \in \mathbb{R}^n$
Ensure: Dual weights $\alpha^\star \in \Delta^2$
1: Set $\alpha^\star \leftarrow (1/2,\ 1/2)$
2: **return** $\alpha^\star$

Algorithm 2 follows from the closed-form structure of the two-loss dual problem. In this case, the dual objective reduces to minimizing

$$\|\alpha \hat{g}_1 + (1-\alpha)\hat{g}_2\|_2^2 \quad \text{over} \quad \alpha \in [0, 1].$$

Since both gradients are ℓ_2-normalized, this objective is minimized at $\alpha = 1/2$ whenever $\hat{g}_1 \neq \hat{g}_2$. If $\hat{g}_1 = \hat{g}_2$, every $\alpha \in [0, 1]$ gives the same direction, so the symmetric choice $\alpha = 1/2$ remains valid.

Algorithm 3 Exact Dual Solver for Three Loss Terms

Require: ℓ_2-normalized gradients $\hat{g}_1, \hat{g}_2, \hat{g}_3 \in \mathbb{R}^n$, feasibility tolerance $\tau > 0$, numerical tolerance $\varepsilon > 0$

Ensure: Dual weights $\alpha^\star \in \Delta^3$

Form the Gram matrix $G \in \mathbb{R}^{3\times 3}$ with $G_{ij} = \hat{g}_i^\top \hat{g}_j$
Find $\alpha^0 \in \Delta^3$ satisfying $G\alpha^0 = 0$, if it exists
if such α^0 exists **then**
    return α^0
end if
$z \leftarrow G^\dagger \mathbf{1}$
$c \leftarrow \mathbf{1}^\top z$
if $c > \varepsilon$ and $\|Gz - \mathbf{1}\|_\infty \le \varepsilon$ **then**
    $\tilde{\alpha} \leftarrow z/c$
    if $\tilde{\alpha}_i \ge -\tau$ for all $i = 1, 2, 3$ **then**
        $\tilde{\alpha} \leftarrow (\tilde{\alpha})_+$
        $\tilde{\alpha} \leftarrow \tilde{\alpha}/(\mathbf{1}^\top \tilde{\alpha})$
        return $\tilde{\alpha}$
    end if
end if
Find

$$(i^\star, j^\star) \in \arg\min_{1 \le i < j \le 3} G_{ij}.$$

$\alpha^\star \leftarrow (0, 0, 0)$
$\alpha^\star_{i^\star} \leftarrow 1/2$
$\alpha^\star_{j^\star} \leftarrow 1/2$
return $\alpha^\star$

Algorithm 3 uses the simple geometric structure of the ℓ_2 dual problem over Δ^3. Its solution is either a zero-norm convex combination, an interior simplex point, or an edge solution. The first case corresponds to $0 \in \mathrm{conv}\{\hat{g}_1, \hat{g}_2, \hat{g}_3\}$. For an interior optimum with positive objective value, the KKT conditions imply $G\alpha = \lambda \mathbf{1}$ for some $\lambda > 0$. We therefore first check whether $z = G^\dagger \mathbf{1}$ satisfies $Gz = \mathbf{1}$, up to numerical tolerance. If it does and $\tilde{\alpha} = z/(\mathbf{1}^\top z)$ is feasible, we return this candidate. Otherwise, an optimal solution can be found on an edge. Because the gradients are ℓ_2-normalized, the exact edge solution is the midpoint of the pair with the smallest Gram entry G_{ij}.

Algorithm 4 DUALCHEBYSHEV-EXACT-ADAM

Require: Initial point θ_0, step sizes $\{\eta_t\}_{t=0}^{T-1}$, Adam parameters $\beta_1, \beta_2, \epsilon_{\text{Adam}}$, threshold $\delta > 0$
Ensure: Updated parameters θ

Initialize Adam moments $M_0 \leftarrow 0$, $S_0 \leftarrow 0$
for $t = 0, 1, \ldots, T-1$ **do**
  Compute grouped gradients $g_1(\theta_t), \ldots, g_m(\theta_t)$
  Compute ℓ_2-normalized gradients $\hat{g}_1(\theta_t), \ldots, \hat{g}_m(\theta_t)$
  if $m = 2$ **then**
    Apply the exact two-loss solver in Algorithm 2 and obtain $\alpha_t \in \Delta^2$
  else if $m = 3$ **then**
    Apply the exact three-loss solver in Algorithm 3 and obtain $\alpha_t \in \Delta^3$
  else
    Apply Frank–Wolfe to the dual (D) and obtain $\alpha_t \in \Delta^m$
  end if
  Form

$$w_t \leftarrow \sum_{i=1}^{m} \alpha_{t,i} \hat{g}_i(\theta_t)$$

  if $\|w_t\|_2 \le \delta$ **then**
    terminate and return θ_t
  end if
  Recover the primal direction:

$$v_t \leftarrow \frac{w_t}{\|w_t\|_2}$$

  Set the final update direction with adaptive scalar:

$$d_t \leftarrow \left(\sum_{i=1}^{m} g_i(\theta_t)^\top v_t \right) v_t$$

  $M_{t+1} \leftarrow \beta_1 M_t + (1 - \beta_1) d_t$
  $S_{t+1} \leftarrow \beta_2 S_t + (1 - \beta_2)(d_t \odot d_t)$
  $\widehat{M}_{t+1} \leftarrow M_{t+1} / (1 - \beta_1^{t+1})$
  $\widehat{S}_{t+1} \leftarrow S_{t+1} / (1 - \beta_2^{t+1})$
  $\theta_{t+1} \leftarrow \theta_t - \eta_t \, \widehat{M}_{t+1} / (\sqrt{\widehat{S}_{t+1}} + \epsilon_{\text{Adam}})$
end for
return θ_T

Algorithm 4 combines the exact dual solvers above with an Adam-style parameter update. When $m = 2$ or $m = 3$, the dual weights are computed by Algorithms 2 and 3, respectively. For larger numbers of loss terms, the dual subproblem is solved using a generic Frank–Wolfe procedure. The resulting aggregate direction is then used to recover the primal direction and the final scaled update direction, exactly as in Algorithm 1, before applying Adam moments to the update.

D EXPERIMENTAL DETAILS

D.1 SOFTWARE AND HARDWARE ENVIRONMENTS

We conduct all experiments in a conda environment with Python 3.11, PyTorch 2.7.0+cu128, and CUDA 12.8. Our experiments are run on three servers: a Rocky Linux 9.7 server with 2 Intel Xeon Silver 4210R CPUs and 4 NVIDIA GeForce RTX 3090 24 GB GPUs; an Ubuntu 24.04.4 LTS server with 2 Intel Xeon Silver 4210R CPUs and 4 NVIDIA GeForce RTX 2080 Ti 11 GB GPUs; and an Ubuntu 24.04.4 LTS server with two Intel Xeon 6517P CPUs and three NVIDIA RTX PRO 6000 Blackwell Server Edition 96 GB GPUs. Each complete experiment is run entirely on a single server to ensure fair comparison across methods.

D.2 Details for section 5.1

Benchmark equations The Helmholtz problem is defined by

$$\begin{aligned}&\Delta u(x,y) + k^2 u(x,y) = f(x,y), \qquad (x,y) \in \Omega,\\ &u(x,y) = 0, \qquad (x,y) \in \partial\Omega,\\ &\Omega = [-1,1] \times [-1,1].\end{aligned}$$

We use the exact solution

$$u^\star(x,y) = \sin(a_1 \pi x)\sin(a_2 \pi y),$$

which induces the forcing term

$$f(x,y) = \big(k^2 - (a_1^2 + a_2^2)\pi^2\big)\sin(a_1 \pi x)\sin(a_2 \pi y).$$

In our experiments, we set $k = 1$, $a_1 = 1$, and $a_2 = 4$.

The 2D Kovasznay benchmark is governed by the steady incompressible Navier–Stokes equations

$$\begin{aligned}&u u_x + v u_y + p_x - \nu(u_{xx} + u_{yy}) = 0, \qquad (x,y) \in \Omega,\\ &u v_x + v v_y + p_y - \nu(v_{xx} + v_{yy}) = 0, \qquad (x,y) \in \Omega,\\ &u_x + v_y = 0, \qquad (x,y) \in \Omega,\end{aligned}$$

where

$$\Omega = [-0.5, 1] \times [-0.5, 1.5].$$

Dirichlet boundary conditions are imposed from the exact solution

$$\begin{aligned}u^*(x,y) &= 1 - e^{\lambda x}\cos(2\pi y),\\ v^*(x,y) &= \frac{\lambda}{2\pi} e^{\lambda x}\sin(2\pi y),\\ p^*(x,y) &= \frac{1}{2}\big(1 - e^{2\lambda x}\big),\end{aligned}$$

with

$$\lambda = \frac{1}{2\nu} - \sqrt{\frac{1}{4\nu^2} + 4\pi^2}.$$

We set $\nu = 1/40$.

The viscous Burgers benchmark is defined by

$$\begin{aligned}&u_t(x,t) + u(x,t)u_x(x,t) - \nu u_{xx}(x,t) = 0, \qquad (x,t) \in \Omega \times [0,1],\\ &u(x,0) = -\sin(\pi x), \qquad x \in \Omega,\\ &u(-1,t) = u(1,t) = 0, \qquad t \in [0,1],\\ &\Omega = [-1,1].\end{aligned}$$

In our experiments, we use $\nu = 0.01/\pi$. Since this benchmark does not admit a closed-form exact solution under the above initial and boundary conditions, we evaluate it using a high-resolution numerical reference solution generated offline.

The 2D varying-coefficient heat benchmark is given by

$$\begin{aligned}&u_t(x,y,t) - a(x,y)\big(u_{xx}(x,y,t) + u_{yy}(x,y,t)\big) = f(x,y,t), \qquad (x,y,t) \in \Omega \times [0,5],\\ &u(x,y,t) = 0, \qquad (x,y,t) \in \partial\Omega \times [0,5],\\ &u(x,y,0) = 0, \qquad (x,y) \in \Omega,\\ &\Omega = [0,1] \times [0,1],\end{aligned}$$

where

$$f(x,y,t) = 200\sin(\pi x)\sin(5\pi y)\sin(\pi t).$$

Here $a(x,y)$ denotes the spatially varying diffusion coefficient taken from the PINNacle benchmark dataset (Hao et al., 2024). For evaluation, we use the corresponding reference solution provided in the benchmark as a COMSOL-exported solution field.

Table 3: Learning rates selected for the final reported runs in Section 5.1. For each method and benchmark, the selection is made from $\{10^{-3}, 3 \times 10^{-4}\}$.

Method	Helmholtz	Kovasznay	Burgers	Heat 2D VC	Klein–Gordon
Adam	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}
LRA	3×10^{-4}	10^{-3}	10^{-3}	3×10^{-4}	10^{-3}
ReLoBRaLo	10^{-3}	10^{-3}	10^{-3}	10^{-3}	3×10^{-4}
MGDA	10^{-3}	10^{-3}	10^{-3}	10^{-3}	10^{-3}
PCGrad	10^{-3}	10^{-3}	10^{-3}	3×10^{-4}	3×10^{-4}
IMTL-G	10^{-3}	3×10^{-4}	3×10^{-4}	10^{-3}	3×10^{-4}
DCGD	3×10^{-4}	3×10^{-4}	–	–	–
ConFIG	–	–	10^{-3}	10^{-3}	3×10^{-4}
HARMONIC	3×10^{-4}	3×10^{-4}	10^{-3}	10^{-3}	3×10^{-4}
Ours	3×10^{-4}	10^{-3}	10^{-3}	10^{-3}	3×10^{-4}

The Klein–Gordon benchmark is defined by

$$
\begin{aligned}
&u_{tt}(x,t) - u_{xx}(x,t) + u(x,t)^3 = f(x,t), \qquad (x,t) \in \Omega \times [0,1],\\
&u(x,0) = x, \qquad x \in \Omega,\\
&u_t(x,0) = 0, \qquad x \in \Omega,\\
&u(x,t) = u^*(x,t), \qquad (x,t) \in \partial\Omega \times [0,1],\\
&\Omega = [0,1].
\end{aligned}
$$

We use the manufactured exact solution

$$u^*(x,t) = x\cos(5\pi t) + (tx)^3,$$

which yields

$$f(x,t) = -25\pi^2 x\cos(5\pi t) + 6tx^3 - 6t^3 x + \big(x\cos(5\pi t) + (tx)^3\big)^3.$$

Network architecture and training setup For all forward benchmark problems, we use a fully connected multilayer perceptron with three linear layers, hidden width 50, and hyperbolic tangent activations. All weights are initialized with Xavier normal initialization (Glorot & Bengio, 2010), and all biases are initialized to zero. Each model is trained for 50,000 optimization steps on a single GPU. Following the implementation used throughout experiments, all methods adopt Adam (Kingma & Ba, 2014) as the base optimizer with fixed parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$, and no learning-rate decay is used.

Sampling strategy For all forward benchmark problems, training points are resampled at every optimization step. Interior collocation points are drawn by uniform random sampling over the corresponding spatial or spatio-temporal domain. For boundary constraints, points are sampled uniformly on each boundary component; for the rectangular 2D domains, the boundary samples are distributed approximately evenly across the four edges, while for the 1D time-dependent problems they are sampled on the left and right spatial boundaries with uniformly sampled time coordinates. For initial conditions, the time coordinate is fixed at $t = 0$ and the spatial coordinates are sampled uniformly over the spatial domain. In the experiments, we use $N_r = 5120$ and $N_b = 256$ for the two steady-state problems, and $N_r = 2560$, $N_b = 256$, and $N_i = 256$ for the three time-dependent problems.

Optimization hyperparameters To account for differences in optimization behavior across methods, we select the learning rate separately for each method and benchmark from the set $\{10^{-3}, 3 \times 10^{-4}\}$. Among the two candidates, we use the one that yields the better final relative L^2 error. The selected learning rates are summarized in Table 3. For LRA, we set $\alpha = 0.1$ and update the loss weights every 10 steps. For ReLoBRaLo, we use $\alpha = 0.95$, $\beta = 0.99$, $\tau = 1.0$, and $\epsilon = 10^{-8}$. For MGDA, we use a Frank–Wolfe subsolver with a maximum of 20 iterations and tolerance 10^{-6}. For our method, we set the stopping threshold to 10^{-6}.

Implementation detail for our method The main text presents our method using a generic dual subproblem solver. In the forward experiments in Section 5.1, which use the ℓ_2 setting, we use the exact variant of our method for the two-loss and three-loss cases. In particular, for all runs of our method reported in Section 5.1, we use the implementation described in Appendix C, which solves the dual problem exactly when $m = 2$ or $m = 3$.

Evaluation metric We report the relative L^2 error as the evaluation metric. Given predictions $\hat{u}(x_i)$ and reference values $u_{\mathrm{ref}}(x_i)$ evaluated at N points, we compute

$$\frac{\left(\sum_{i=1}^{N} \|\hat{u}(x_i) - u_{\mathrm{ref}}(x_i)\|_2^2\right)^{1/2}}{\left(\sum_{i=1}^{N} \|u_{\mathrm{ref}}(x_i)\|_2^2\right)^{1/2}}.$$

For scalar-valued problems, the inner ℓ_2 norm reduces to the absolute value. For the Helmholtz equation, 2D Kovasznay flow, and Klein–Gordon equation, u_{ref} is given by the corresponding analytic or manufactured solution. For Burgers' equation, u_{ref} is the offline high-resolution numerical reference described above. For the 2D varying-coefficient heat equation, u_{ref} is the benchmark reference solution field taken from the PINNacle dataset.

For Helmholtz and Kovasznay, we evaluate on a 256×256 spatial grid. For Burgers' and Klein–Gordon, we use a 256×256 space–time grid. Each of these grids contains $65{,}536$ evaluation points. For Heat 2D VC, we use 983 spatial locations at 51 time instants, totaling $50{,}133$ evaluation points.

D.3 Details for section 5.2

Inverse problem setup We consider an inverse diffusion problem in which the solution $u(x, y, t)$ and the spatially varying diffusion coefficient $a(x, y)$ are learned simultaneously. The governing equation is

$$u_t(x, y, t) - \nabla \cdot \big(a(x, y)\nabla u(x, y, t)\big) = f(x, y, t), \qquad (x, y, t) \in \Omega \times [0, 1],$$
$$\Omega = [-1, 1] \times [-1, 1].$$

We construct the benchmark from the reference fields

$$u^\star(x, y, t) = e^{-t} \sin(\pi x) \sin(\pi y), \qquad a^\star(x, y) = 2 + \sin(\pi x) \sin(\pi y),$$

and choose $f(x, y, t)$ accordingly so that $(u^\star, a^\star)$ satisfies the PDE.

Observation data and data settings We adopt the inverse heat benchmark setup and its fixed pool of $2{,}500$ observation locations from PINNacle (Hao et al., 2024). We use noisy pointwise observations of u together with boundary supervision for a. The Standard and Higher-noise settings use the full pool, whereas the Fewer-data setting uses $1{,}250$ locations sampled uniformly without replacement. More precisely, given observation points $\{(x_j, y_j, t_j)\}_{j=1}^{N_{\mathrm{obs}}}$, we construct the observed values as

$$\tilde{u}_j = u^\star(x_j, y_j, t_j) + \varepsilon_j, \qquad \varepsilon_j \sim \mathcal{N}(0, \sigma^2).$$

For each data setting, the noisy observation set is generated with a fixed observation seed of 0, shared across the three runs, and kept fixed throughout training. In addition, we impose coefficient boundary supervision by matching $a(x, y)$ to $a^\star(x, y)$ on $\partial\Omega \times [0, 1]$.

We consider three data settings in Section 5.2: *Standard* with $N_{\mathrm{obs}} = 2500$ and $\sigma = 0.1$, *Higher noise* with $N_{\mathrm{obs}} = 2500$ and $\sigma = 0.2$, and *Fewer data* with $N_{\mathrm{obs}} = 1250$ and $\sigma = 0.1$.

Network architecture and training setup For the inverse heat problem, we use a parallel fully connected network with two separate branches, one for the solution field $u(x, y, t)$ and the other for the coefficient field $a(x, y)$. Each branch is implemented as a multilayer perceptron with five linear layers, hidden width 50, and hyperbolic tangent activations. To reflect the fact that the coefficient is time-independent, the a-branch takes only the spatial coordinates as effective inputs, while the time coordinate is masked out. All weights are initialized with Xavier normal initialization, and all biases are initialized to zero. Each model is trained for 30,000 optimization steps.

Table 4: Learning rates selected for the final reported runs in Section 5.2. For each method and data setting, the selection is made from $\{10^{-3}, 3 \times 10^{-4}\}$.

Method	Standard	Higher noise	Fewer data
Adam	3×10^{-4}	3×10^{-4}	3×10^{-4}
LRA	3×10^{-4}	3×10^{-4}	10^{-3}
ReLoBRaLo	10^{-3}	10^{-3}	10^{-3}
MGDA	3×10^{-4}	3×10^{-4}	3×10^{-4}
PCGrad	3×10^{-4}	3×10^{-4}	3×10^{-4}
IMTL-G	10^{-3}	10^{-3}	10^{-3}
ConFIG	10^{-3}	10^{-3}	10^{-3}
HARMONIC	10^{-3}	10^{-3}	10^{-3}
Ours	10^{-3}	10^{-3}	10^{-3}

Table 5: Comparison with GAPO on five forward PDE benchmarks. Values are the mean and standard deviation of the final relative L^2 error over three independent runs. The results for our method are also reported in Table 1.

Benchmark	GAPO ($\rho = 1$)	GAPO ($\rho = 2$)	Ours ($p = 2$)
Helmholtz	0.0073 (0.0019)	0.0432 (0.0312)	0.0072 (0.0033)
Kovasznay	0.0055 (0.0023)	0.0880 (0.0061)	0.0017 (0.0005)
Burgers'	0.0597 (0.0106)	0.0947 (0.0066)	0.0599 (0.0042)
Heat 2D VC	0.2388 (0.0336)	0.6211 (0.0397)	0.2316 (0.0190)
Klein–Gordon	0.0197 (0.0046)	0.0389 (0.0106)	0.0218 (0.0074)

Sampling strategy At each optimization step, the collocation points and the coefficient-boundary points are resampled. The collocation points are drawn by uniform random sampling over $\Omega \times [0, 1]$, while the boundary points for a are sampled uniformly on the four spatial boundary components of $\partial\Omega \times [0, 1]$, with the samples distributed approximately evenly across the four edges. In contrast, the noisy observation set for u is generated once before training and then kept fixed throughout optimization. In the final experiments, we use $N_r = 5120$ collocation points and $N_{b,a} = 1024$ coefficient-boundary points per step. The number of observation points is $N_{\text{obs}} = 2500$ for the Standard and Higher-noise settings, and $N_{\text{obs}} = 1250$ for the Fewer-data setting.

Optimization hyperparameters The optimization hyperparameters are chosen in the same manner as in the forward experiments. In particular, for each method and data setting, we select the learning rate from $\{10^{-3}, 3 \times 10^{-4}\}$ and use the one that yields the better final relative L^2 error. The selected learning rates are summarized in Table 4. Method-specific hyperparameters are the same as those described above for the forward experiments.

Evaluation metric As in the forward experiments, we use the relative L^2 error as the evaluation metric. For the inverse problem, however, the reported error is computed on the recovered coefficient field $a(x, y)$ rather than on the solution field $u(x, y, t)$. We evaluate the recovered coefficient on a 64×64 spatial grid containing $4{,}096$ points.

E Additional experimental results

E.1 Comparison with GAPO

Experimental setup We evaluate GAPO with $\rho \in \{1, 2\}$ on the five forward PDE benchmarks. We use the network architectures, sampling strategy, training budget, and Adam settings described in Appendix D.2. For each value of ρ and each benchmark, we select the learning rate from $\{10^{-3}, 3 \times 10^{-4}\}$ using the same procedure as in the main experiments. Each setting is evaluated over three independent runs. Table 5 reports the mean and standard deviation of the final relative L^2 error. For reference, the table also includes the results of our $p = 2$ method reported in Table 1.

Table 6: Effect of the norm parameter p on the five forward PDE benchmarks. Entries report the mean and standard deviation of the final relative L^2 error over three independent runs. Lower is better. The lowest and second-lowest mean errors in each column are shown in **bold** and underlined, respectively.

p	Helmholtz	Kovasznay	Burgers'	Heat 2D VC	Klein–Gordon
1	0.0111 (0.0053)	0.0084 (0.0047)	0.1767 (0.0295)	0.2579 (0.0560)	0.0294 (0.0139)
1.5	0.0166 (0.0084)	0.0060 (0.0002)	0.0842 (0.0368)	**0.2214 (0.0122)**	0.0223 (0.0123)
2	0.0099 (0.0026)	**0.0024 (0.0011)**	**0.0759 (0.0080)**	0.2276 (0.0189)	**0.0204 (0.0026)**
3	**0.0082 (0.0012)**	0.0322 (0.0273)	0.0934 (0.0198)	0.2595 (0.0497)	0.0305 (0.0210)
4	0.0177 (0.0127)	0.0031 (0.0005)	0.1283 (0.0396)	0.3036 (0.0635)	0.0257 (0.0100)
∞	1.0396 (0.0552)	0.1799 (0.0311)	0.5223 (0.2226)	0.9960 (0.0137)	0.5808 (0.2510)

Results GAPO with $\rho = 1$ and our method yield similar mean errors on Helmholtz and Burgers'. Our method achieves lower mean errors on Kovasznay and Heat 2D VC, whereas GAPO with $\rho = 1$ achieves a lower mean error on Klein–Gordon. Thus, neither method achieves the lowest mean error across all five benchmarks. GAPO with $\rho = 2$ yields higher mean errors than both alternatives on every benchmark.

Although GAPO with $\rho = 1$ and our $p = 2$ method share the same dual problem, this equivalence concerns direction selection at a given parameter point. It does not imply identical training trajectories when the final update scaling differs. The results therefore complement the formulation-level comparison without establishing a uniform empirical advantage for either method.

E.2 Effect of the norm parameter

For conjugate exponents p and q, the norm parameter p determines the gradient normalization, the primal ℓ_q constraint, the dual norm, and the recovery of the primal direction. We examine how this choice affects performance across the five forward PDE benchmarks.

The default choice of $p = 2$ We use $p = 2$ as the default for two reasons. First, the Euclidean formulation is invariant under orthogonal changes of coordinates. Specifically, let $r_2(g_1, \ldots, g_m)$ denote the optimal value of (P) with $p = q = 2$. For any orthogonal matrix Q,

$$r_2(Qg_1, \ldots, Qg_m) = r_2(g_1, \ldots, g_m).$$

Indeed, $\|Qg_i\|_2 = \|g_i\|_2$, and the change of variables $v \mapsto Qv$ preserves both the normalized-gradient inner products and the constraint $\|v\|_2 \le 1$. Thus, rotating or reflecting all gradients leaves the optimal value unchanged. By contrast, this invariance does not generally hold for $p \neq 2$, making the Euclidean criterion independent of the choice of orthogonal coordinate axes.

Second, for $p = 2$, the dual problem with two or three objectives can be solved both efficiently and exactly using the solvers in Appendix C. This provides a computational advantage in our PINN experiments, which involve two or three objectives.

In these experiments, we impose neither coordinatewise bounds nor sparsity constraints on parameter updates that would explicitly motivate a non-Euclidean geometry. Together, orthogonal invariance and efficient exact computation therefore make $p = 2$ a natural default for our setting. This choice does not, however, guarantee the lowest final prediction error on every problem.

Experimental setup We compare $p \in \{1, 1.5, 2, 3, 4, \infty\}$ on Helmholtz, 2D Kovasznay flow, Burgers' equation, the 2D varying-coefficient heat equation, and the Klein–Gordon equation. We use the same network architectures, sampling strategy, training budget, and Adam settings as in the main forward experiments described in Appendix D.2. For each norm parameter and benchmark, we select the learning rate from $\{10^{-3}, 3 \times 10^{-4}\}$ using the same selection procedure as in the main experiments. Each setting is evaluated over three independent runs.

Table 6 reports the mean and standard deviation of the final relative L^2 error. All entries, including the $p = 2$ results, are obtained from separate runs for this comparison.

Results No single tested value of p attains the lowest mean error on all five benchmarks. The lowest mean error is obtained with $p = 3$ on Helmholtz and $p = 1.5$ on Heat 2D VC, whereas $p = 2$ performs best on Kovasznay, Burgers', and Klein–Gordon. The $p = \infty$ setting performs substantially worse across all five benchmarks. These results show that non-Euclidean choices can yield lower mean errors on some problems, while $p = 2$ remains a useful default. Both the orthogonal invariance of the Euclidean direction-selection criterion and the availability of efficient exact solvers for two or three objectives support this default choice, without guaranteeing the lowest final prediction error on every problem.

E.3 Multi-model adversarial attacks

While $p = 2$ provides a useful default for our PINN experiments, an explicit norm constraint can motivate a different choice of geometry.

We consider a multi-model adversarial attack in which, for each input image, a single perturbation is optimized to cause misclassification by multiple classifiers simultaneously. The perturbation is constrained by an ℓ_∞ budget, providing an explicit norm structure for direction selection. Setting $p = 1$ gives $q = \infty$, so our primal formulation selects a direction directly within the corresponding ℓ_∞ unit ball. This provides a concrete setting in which a non-Euclidean choice of p is motivated by the application.

Experimental setup We jointly attack three models from the RobustBench model zoo (Croce et al., 2020): `Wong2020Fast` (Wong et al., 2020), `Ding2020MMA` (Ding et al., 2020), and `Zhang2019You` (Zhang et al., 2019). We use 1,000 CIFAR-10 (Krizhevsky et al., 2009) test images that are correctly classified by all three models. Each model contributes an untargeted difference-of-logits-ratio (DLR) objective (Croce & Hein, 2020). All methods use an ℓ_∞ perturbation budget of $8/255$, a step size of $2/255$, the same projection rule, and 100 iterations.

Direction construction At attack iteration k, let $g_i = \nabla_\delta \mathcal{L}_i(x + \delta_k, y)$ be the gradient of the attack objective for model i, and write $\hat{g}_i^{(p)} = g_i / \|g_i\|_p$. Here, the attack objectives are maximized.

For $p = 1$, we solve the primal problem independently for each input:

$$\begin{aligned} \max_{v,r} \quad & r \\ \text{subject to} \quad & (\hat{g}_i^{(1)})^\top v \geq r, \qquad i = 1, \dots, m, \\ & -1 \leq v_j \leq 1, \qquad j = 1, \dots, d, \end{aligned}$$

where d is the input dimension. We solve this linear program and use the resulting primal direction $v^\star$ directly as the attack direction $u_1 = v^\star$, without a sign transformation.

For $p = 2$, we solve the dual problem (D) with the ℓ_2-normalized gradients and form

$$w_2 = \sum_{i=1}^{m} \alpha_{2,i}^\star \hat{g}_i^{(2)}.$$

We compare two mappings of this vector into the ℓ_∞ unit ball:

$$u_{2,\text{sign}} = \text{sign}(w_2), \qquad u_{2,\text{ray}} = \begin{cases} w_2 / \|w_2\|_\infty, & w_2 \neq 0, \\ 0, & w_2 = 0. \end{cases}$$

For nonzero w_2, the ray-preserving mapping rescales the vector to the boundary of the unit ball without changing its ray, whereas the sign mapping generally changes the ray.

For MGDA (Sener & Koltun, 2018), we instead use the unnormalized objective gradients:

$$\alpha_{\text{MGDA}}^\star \in \arg\min_{\alpha \in \Delta^m} \left\| \sum_{i=1}^{m} \alpha_i g_i \right\|_2, \qquad u_{\text{MGDA}} = \text{sign}\left(\sum_{i=1}^{m} \alpha_{\text{MGDA},i}^\star g_i \right).$$

For every method, we update the perturbation using

$$\delta_{k+1} = \Pi_{\mathcal{D}(x)}(\delta_k + \eta u_k), \qquad \mathcal{D}(x) = \left\{ \delta : \|\delta\|_\infty \leq \epsilon,\ x + \delta \in [0,1]^d \right\},$$

with the same step size η and Euclidean projection $\Pi_{\mathcal{D}(x)}$.

Table 7: Simultaneous attack success rates on three CIFAR-10 classifiers under an ℓ_∞ perturbation budget of $8/255$. Values are the mean and standard deviation over three attack seeds and are reported in percentage points. Higher is better; the best mean is shown in bold.

Method	Final A-All (%)
Ours ($p = 1$)	**13.07 (0.21)**
Ours ($p = 2$, sign)	9.47 (0.21)
Ours ($p = 2$, ℓ_∞-ray)	4.77 (0.06)
MGDA (sign)	8.70 (0.10)

Table 8: Final relative L^2 errors using direct updates without Adam. Values are the mean and standard deviation over three independent runs at the selected learning rate. ConFIG is omitted for the two-objective benchmarks, where it coincides with our method. "Diverged" indicates that NaN or infinite values prevented a complete three-run aggregate.

Method	Helmholtz	Kovasznay	Burgers'	Heat 2D VC	Klein–Gordon
MGDA	0.9999 (0.0000)	0.7453 (0.0736)	1.0010 (0.1031)	1.7050 (0.5648)	1.0449 (0.0186)
ConFIG	—	—	0.9233 (0.0903)	1.2146 (0.0763)	1.1343 (0.0121)
HARMONIC	0.0966 (0.0388)	0.6838 (0.0052)	0.9359 (0.1050)	1.3216 (0.5130)	Diverged
GAPO ($\rho = 1$)	0.9988 (0.0009)	0.6883 (0.0004)	0.9294 (0.1095)	1.0001 (0.0005)	1.0682 (0.0058)
Ours	0.1613 (0.0953)	0.6838 (0.0052)	0.9254 (0.0914)	1.1979 (0.3087)	1.3522 (0.0233)

Results Table 7 reports the mean and standard deviation of the final simultaneous attack success rate (A-All), defined as the percentage of images misclassified by all three models after perturbation, over three independent runs. Higher values indicate more successful attacks. Our $p = 1$ formulation achieves the highest mean success rate, exceeding the $p = 2$ sign variant by 3.60 percentage points and MGDA sign by 4.37 percentage points. The $p = 2$ sign variant also outperforms the ray-preserving variant, showing that the sign transformation materially affects attack performance in this experiment.

Applying the sign operator generally changes the direction selected by the Euclidean multi-objective problem. By contrast, the $p = 1$ formulation incorporates the ℓ_∞ direction constraint directly into the max–min problem. The results support choosing the norm geometry to match the application's constraints and illustrate the applicability of our formulation beyond PINN training.

E.4 Direct updates without Adam

Experimental setup Theorem 3.8 analyzes the direct update in Algorithm 1 and does not directly cover Adam's momentum and coordinatewise adaptive scaling. We evaluate MGDA, ConFIG, HARMONIC, GAPO with $\rho = 1$, and our method with $p = 2$ on the five forward PDE benchmarks using direct updates. For each method, we apply its computed update direction directly to the parameters, without momentum or coordinatewise adaptive scaling. For our method, this gives $\theta_{t+1} = \theta_t - \eta_t d_t$, where d_t includes the adaptive scalar in Algorithm 1.

We retain the network architectures, sampling strategy, and training budget described in Appendix D.2. Because direct updates are sensitive to the magnitude of the computed direction, we use the lower learning-rate candidates $\{3 \times 10^{-4}, 10^{-4}\}$ for all methods. For each method and benchmark, we select the learning rate with the lower mean final relative L^2 error over three independent runs. Table 8 reports the corresponding means and standard deviations.

Results On Helmholtz, HARMONIC achieves the lowest mean error, followed by our method. On Kovasznay, HARMONIC and our method both yield a mean error of 0.6838 at the reported precision. However, the errors on Burgers', Heat 2D VC, and Klein–Gordon remain close to or above 1 across the tested methods, with HARMONIC diverging on Klein–Gordon.

Compared with the Adam-based results in Table 1 and Table 5, direct updates yield substantially higher mean errors for the corresponding methods and benchmarks. Thus, under the tested learning rates and training budget, direction selection alone does not recover the accuracy obtained with

Adam. These results highlight the practical importance of combining direction selection with Adam in our PINN experiments.

E.5 Computational Cost

Direction-computation complexity Let n denote the number of trainable parameters and m the number of objectives. We compare the arithmetic cost of constructing an update direction after the individual objective gradients have been computed. This separates the cost of combining gradients from that of evaluating the losses and computing their gradients. Table 9 summarizes the costs derived from the respective direction constructions, assuming $n \geq m$ and standard dense linear algebra.

Table 9: Direction-computation costs, excluding objective-gradient computation. Here, K is the number of Frank–Wolfe iterations, R is the number of coefficient-space rays enumerated by HARMONIC, and T_{DD} is the cost of its double-description enumeration.

Method / solver	Arithmetic cost
Ours, exact ($p = 2,\ m = 2$)	$O(nm)$
Ours, exact ($p = 2,\ m = 3$)	$O(nm^2 + m^3)$
Ours, matrix-free Frank–Wolfe	$O(Knm)$
ConFIG, general dense solve	$O(nm^2 + m^3)$
IMTL-G	$O(nm^2 + m^3)$
HARMONIC, ray enumeration	$O(nm^2 + nmR) + T_{\mathrm{DD}}$

For our method with $p = 2$, the two-objective solver requires only vector operations. The three-objective solver forms a small Gram matrix and evaluates a fixed number of interior and boundary candidates, with costs of $O(nm^2)$ and $O(m^3)$, respectively. Gradient normalization, direction recovery, and the adaptive scalar require at most $O(nm)$ additional work. Thus, both exact solvers have $O(n)$ cost for their respective fixed numbers of objectives.

For ConFIG (Liu et al., 2025), the general direction construction uses a least-squares or pseudoinverse solve involving the normalized-gradient matrix, with cost $O(nm^2 + m^3)$. Its specialized two-objective construction also has $O(n)$ cost. For IMTL-G (Liu et al., 2021b), forming the matrix of inner products between gradient differences costs $O(nm^2)$, followed by an $O(m^3)$ coefficient solve.

HARMONIC (Bu et al., 2026) first forms a Gram matrix at cost $O(nm^2)$. It then enumerates the extreme rays of the coefficient cone using the double-description method and maps the resulting R rays into parameter space. The latter operation costs $O(nmR)$, giving a total of $O(nm^2 + nmR) + T_{\mathrm{DD}}$. The enumeration cost depends on the cone geometry and the intermediate rays generated during the procedure.

Scaling with parameters and objectives For a fixed small number of objectives, our exact Euclidean solvers scale linearly with the parameter dimension, matching the dependence on n of the compared direction constructions. The distinction becomes more relevant as the number of objectives increases.

For $1 < p < \infty$, our dual can be optimized using matrix-free Frank–Wolfe iterations. Each iteration evaluates a weighted gradient combination and its inner products with the objective gradients, requiring $O(nm)$ work. Thus, K iterations cost $O(Knm)$, without explicitly forming an $m \times m$ Gram matrix or enumerating cone rays. This provides an alternative to the dense solves in ConFIG and IMTL-G and the ray enumeration in HARMONIC. The resulting cost depends on the number of iterations needed for the desired solver accuracy.

Training time To compare the practical computational cost of the competing methods, we record the mean wall-clock time required to complete the full training run for each setting. The forward benchmark problems are run for $50{,}000$ training steps, whereas the inverse benchmark settings are

Table 10: Mean wall-clock time required to complete $50{,}000$ training steps for each forward benchmark problem.

Method	Helmholtz	Kovasznay	Burgers	Heat 2D VC	Klein–Gordon
Adam	268.07	508.71	293.77	351.97	442.28
LRA	283.39	525.18	320.22	334.14	429.72
ReLoBRaLo	323.73	526.12	361.85	372.15	473.33
MGDA	288.43	530.61	616.87	408.62	574.87
PCGrad	290.90	513.03	361.78	395.85	461.49
IMTL-G	302.55	525.45	338.28	371.99	477.12
DCGD	282.54	567.91	–	–	–
ConFIG	–	–	342.06	387.18	463.90
HARMONIC	295.28	578.07	331.61	363.10	534.58
Ours	311.12	540.58	366.63	405.21	501.02

run for $30{,}000$ training steps. The measured runtimes, reported in seconds, are summarized in Tables 10 and 11, respectively.

Table 11: Mean wall-clock time required to complete $30{,}000$ training steps for each inverse benchmark setting.

Method	Standard	Higher noise	Fewer data
Adam	642.93	574.93	587.18
LRA	632.66	578.48	597.58
ReLoBRaLo	614.49	607.36	632.04
MGDA	649.07	676.26	669.19
PCGrad	647.41	610.27	626.52
IMTL-G	612.95	601.61	596.25
ConFIG	633.47	613.84	655.04
HARMONIC	637.30	605.30	609.60
Ours	676.37	648.24	624.63

Across the forward benchmarks, our method takes approximately 5.6%, 6.6%, and 3.0% more training time than IMTL-G, ConFIG, and HARMONIC, respectively, on average. These percentages are computed by averaging the per-benchmark relative time differences over the benchmarks shared by each pair: five for IMTL-G and HARMONIC, and three for ConFIG. For the inverse settings, the corresponding averages are 7.6%, 2.6%, and 5.2%. Thus, the proposed method has comparable full-training cost to these baselines in the evaluated settings.

AI USE STATEMENT

We used OpenAI's GPT-5.6 Sol to assist with language editing and manuscript organization, reviewing existing proofs for consistency and clarity, and code development. The authors reviewed and revised the AI-assisted text, checked the mathematical arguments, and reviewed and tested the AI-assisted code. The authors take full responsibility for the final content of this paper, including its theoretical claims, implementation, and experimental results.