LEARNING AS ENERGY DISSIPATION: HANDLING INEXACT SURROGATE GRADIENTS IN DECISION-FOCUSED TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning is often imagined as a descent along a smooth surface, where each step steadily lowers the energy. Decision-focused learning (DFL) challenges this picture. Instead of optimizing prediction error, it optimizes the quality of downstream decisions, which requires surrogate losses to approximate decision regret. The gradients of these surrogates are not exact: the ideal gradient of the surrogate vanishes at its optimum, while the computed directions are noisy and misaligned. In practice, this breaks the energy-dissipative nature of learning, producing unstable training curves and erratic decision quality.

We reinterpret this difficulty through the lens of energy dissipation. If training is viewed as the evolution of a dissipative system, then instability arises precisely when inexact gradients violate the energy law. To restore this structure, we introduce EDO (Energy-Dissipative Optimizer), which reformulates updates as implicit descent steps that guarantee monotone energy decrease even under gradient inexactness. EDO integrates simple stabilizing mechanisms—scaled weight decay, parameter averaging, and adaptive momentum—without altering the modeling pipeline. We show that three widely used surrogate families admit valid subgradients under this formulation, and provide theoretical guarantees of monotone energy descent with uniform error bounds. Across four benchmark tasks, EDO produces smoother training trajectories, lower regret, and more reliable decisions than existing methods.

1 INTRODUCTION

Learning algorithms are often imagined as dissipative systems. At each step, the learner descends along an energy surface, reducing a potential that measures how far the system is from equilibrium. Gradient descent embodies this idea: the gradient points to the steepest direction of decrease, and the process guarantees monotone reduction of the underlying energy. This perspective has underpinned many successes in end-to-end learning, where raw inputs are directly mapped to outputs without hand-crafted intermediate stages (He et al., 2016; Vaswani, 2017).

When learning is coupled with decision-making, however, this picture breaks. Applications such as autonomous driving (Hu et al., 2023b), healthcare allocation (Wang et al., 2023), and climate planning (Harder et al., 2023) highlight that predictive accuracy alone does not suffice—the ultimate measure is decision quality. In such problems, optimal decisions depend on uncertain or latent parameters like costs or demands, which must be inferred from features. Traditional "predict-then-optimize" pipelines train a predictor first and then feed its outputs into an optimizer, but small prediction errors can still lead to poor downstream decisions (Elmachtoub & Grigas, 2022; Geng et al., 2024).

Decision-Focused Learning (DFL) was proposed to close this gap (Demirović et al., 2019; Elmachtoub & Grigas, 2022). Instead of training for accuracy, predictors are updated with awareness of the downstream optimization problem. However, decision regret is typically non-differentiable, preventing direct backpropagation. To enable gradient-based training, researchers design surrogate objectives—convex relaxations, differentiable approximations, or upper bounds—that serve as proxies for regret (Mulamba et al., 2021; Niepert et al., 2021; Ferber et al., 2023).

These surrogates make end-to-end training feasible, but they bring a fundamental challenge. The gradients they provide are not exact. A smooth, well-aligned surrogate would have a vanishing gradient at its optimum, while the approximated surrogates used in practice do not. Their gradients are structurally different from the ideal decision gradient, which leads to persistent bias rather than unbiased stochastic noise. The consequence is visible: training curves oscillate, regret sometimes increases rather than decreases, and decision quality varies sharply across epochs. In short, the energy-dissipative nature of learning is broken.

Existing optimizers such as Adam (Kingma, 2014) and RMSProp (Schaul et al., 2013) assume gradients are either exact or unbiased estimates. They are not designed for structurally biased surrogates, and as we later show, applying them directly can cause parameter drift, oscillations, and poor convergence—even when surrogate error is small. Recent studies of optimization under inexact gradients (Yang & Li, 2023; Barré et al., 2023) shed light on this issue in classical settings, but their insights have not been connected to decision-focused training, where inexactness arises from model structure rather than sampling.

We build on this gap by turning to the energy perspective. If learning is understood as an energy dissipation process, then instability in DFL can be read as the violation of this law: biased surrogate gradients inject energy instead of dissipating it. Our contribution is to restore the missing structure. We propose the Energy-Dissipative Optimizer (EDO), which reformulates updates as implicit descent steps anchored in local proximity, thereby guaranteeing monotone energy decrease even under gradient misalignment.

The contributions of this paper are threefold. First, we formalize instability in DFL as a breakdown of dissipative dynamics and recast training as implicit energy descent. Second, we develop EDO, a general optimizer that restores dissipation through implicit updates and prove uniform error bounds together with monotone descent guarantees across three surrogate families. Third, we validate EDO on four benchmarks, showing smoother learning trajectories, consistent regret reduction, and more reliable decision outcomes than existing methods.

2 The Hidden Cost of Surrogate Gradients in DFL Training

2.1 Surrogate Gradients: Convenient, But Inexact

Decision-Focused Learning (DFL) replaces prediction error with decision regret as the training signal:

$$R(\hat{\boldsymbol{c}}) = \boldsymbol{c}^\top \boldsymbol{z}^*(\hat{\boldsymbol{c}}) - \boldsymbol{c}^\top \boldsymbol{z}^*(\boldsymbol{c}),$$

where $\boldsymbol{z}^*(\hat{\boldsymbol{c}})$ is the optimal decision under predicted costs. Because $\boldsymbol{z}^*(\cdot)$ is non-differentiable, most methods introduce a surrogate $\tilde{R}$ with gradients

$$\nabla_{\boldsymbol{\theta}} \tilde{R}(\hat{\boldsymbol{c}}) = \nabla_{\hat{\boldsymbol{c}}} \tilde{R}(\hat{\boldsymbol{c}}) \cdot \nabla_{\boldsymbol{\theta}} \hat{\boldsymbol{c}}(\boldsymbol{\theta}).$$

This makes backpropagation possible, but the price is bias: the surrogate gradient $\tilde{\boldsymbol{g}}$ is not the ideal gradient of regret, and the gap rarely vanishes.

2.2 Error Propagation in Iterative Updates

To see the effect, consider a simple gradient descent update. Let $\boldsymbol{g}_t$ be the ideal gradient of $L(\theta)$ and $\tilde{\boldsymbol{g}}_t$ the surrogate gradient with error ε_t. The update follows

$$\theta_{t+1} = \theta_t - \eta\tilde{\boldsymbol{g}}_t = \theta_t - \eta\boldsymbol{g}_t - \eta\varepsilon_t.$$

Hence the deviation from the ideal path grows as

$$e_{t+1} = e_t - \eta\varepsilon_t.$$

Even when $\|\varepsilon_t\| \le \delta$ is tiny, the deviation accumulates over time: after T steps, $\|e_T\|$ can scale linearly in T. This explains the empirical symptoms: training curves oscillate, regret may rise rather than fall, and decision quality drifts unpredictably. One may shrink η to slow down error accumulation, but too small a step size leads to underfitting; too large, and bias dominates. This tradeoff is not an implementation bug but a structural limitation of gradient-based updates under biased surrogates.

2.3 A Structural Problem, Not a Tuning Issue

The picture is clear: standard SGD-style updates accumulate surrogate bias unchecked. From an energy perspective, each update should reduce a potential. However, with inexact gradients, the system is free to inject energy back, causing unstable dynamics. To prevent this drift, the update rule itself must change. In the next section, we show how implicit descent restores dissipative behavior, transforming error accumulation from linear growth into a bounded effect.

3 Energy-Dissipative Optimizer

3.1 Composite Energy and Legal Descent Directions

The difficulty isolated in Section 2 is structural: surrogate gradients are biased and the bias accumulates. Our first step in EDO is therefore to make explicit the *energy* we want to dissipate and the *legal directions* along which dissipation is measured. We anchor the predictor output $\hat{\mathbf{c}}$ by a strongly convex reference and a surrogate regret term,

$$F(\hat{\mathbf{c}}) \;=\; f(\hat{\mathbf{c}}) \;+\; \tilde{R}(\hat{\mathbf{c}}), \qquad f(\hat{\mathbf{c}}) \;=\; \tfrac{1}{2}\,\|\hat{\mathbf{c}} - \mathbf{c}\|^2,$$

where f plays the role of a quadratic potential well and $\tilde{R}$ encodes decision regret through a differentiable or subdifferentiable proxy. Throughout we assume f is L-smooth and μ-strongly convex ($0 < \mu \le L$), while $\tilde{R}$ is convex, proper, and lower semicontinuous. Let $\mathbf{g}_t := \nabla f(\hat{\mathbf{c}}_t) + g_t$ with $g_t \in \partial\tilde{R}(\hat{\mathbf{c}}_t)$ be the exact composite gradient; the algorithm only observes an inexact surrogate $\tilde{\mathbf{g}}_t$ with $\|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \le \delta$.

To use F as a discrete Lyapunov function, the surrogate families that appear in practice must deliver *valid* subgradients for $\tilde{R}$. This is where the energy view begins to constrain design: a direction is "legal" if it lies in $\partial\tilde{R}(\hat{\mathbf{c}})$ so that progress along $-\nabla f(\hat{\mathbf{c}}) - g$ indeed decreases F. We verify this property for three widely used classes, which we will also use in experiments.

Convex upper bounds. Let $\mathbf{W} \subset \mathbb{R}^n$ be a nonempty polyhedral set and $\mathbf{w}^*(\mathbf{c}) \in \arg\min_{\mathbf{w}\in\mathbf{W}} \mathbf{c}^\top\mathbf{w}$. Consider

$$L_{\text{upper}}(\hat{\mathbf{c}}; \mathbf{c}) \;=\; -\min_{\mathbf{w}\in\mathbf{W}} (2\hat{\mathbf{c}} - \mathbf{c})^\top\mathbf{w} \;+\; 2\hat{\mathbf{c}}^\top\mathbf{w}^*(\mathbf{c}) \;-\; \mathbf{c}^\top\mathbf{w}^*(\mathbf{c}).$$

The map $\hat{\mathbf{c}} \mapsto L_{\text{upper}}(\hat{\mathbf{c}}; \mathbf{c})$ is convex as a support function composed with an affine form. Any $\mathbf{w}^\star \in \arg\min_{\mathbf{w}\in\mathbf{W}}(2\hat{\mathbf{c}} - \mathbf{c})^\top\mathbf{w}$ yields a subgradient $g_{\text{upper}} = 2\mathbf{w}^*(\mathbf{c}) - 2\mathbf{w}^\star \in \partial_{\hat{\mathbf{c}}} L_{\text{upper}}(\hat{\mathbf{c}}; \mathbf{c})$. Thus the "push" given by the difference between the ideal and surrogate argmin is a legal descent direction for $\tilde{R}$.

Perturbation/entropy-based surrogates. Let A be the log-partition of an exponential family with mean map $\boldsymbol{\mu} = \nabla A$. A perturbation/entropy view leads to the convex Bregman energy

$$L^{\dagger}_{\mathrm{pert}}(\hat{\mathbf{c}}; \mathbf{c}) \;=\; A(\hat{\mathbf{c}}) - A(\mathbf{c}) - \nabla A(\mathbf{c})^{\top}(\hat{\mathbf{c}} - \mathbf{c}),$$

which is convex in $\hat{\mathbf{c}}$ and satisfies $\nabla_{\hat{\mathbf{c}}} L^{\dagger}_{\mathrm{pert}}(\hat{\mathbf{c}}; \mathbf{c}) = \boldsymbol{\mu}(\hat{\mathbf{c}}) - \boldsymbol{\mu}(\mathbf{c})$. Equivalently, writing $g_{\mathrm{pert}} := \boldsymbol{\mu}(\mathbf{c}) - \boldsymbol{\mu}(\hat{\mathbf{c}})$ gives $-g_{\mathrm{pert}} \in \partial_{\hat{\mathbf{c}}} L^{\dagger}_{\mathrm{pert}}(\hat{\mathbf{c}}; \mathbf{c})$. This matches the gradient signals implemented by noise-injection and log-partition relaxations and supplies a valid subgradient of $\tilde{R}$ in the optimization variable $\hat{\mathbf{c}}$.

Contrastive surrogates. Given a finite comparison set $\Gamma \subset \mathbf{W}$ with $\mathbf{w}^*(\mathbf{c}) \in \Gamma$, define

$$L_{\mathrm{contr}}(\hat{\mathbf{c}}; \mathbf{c}) \;=\; \frac{1}{|\Gamma| - 1} \sum_{\mathbf{w} \in \Gamma \setminus \{\mathbf{w}^*(\mathbf{c})\}} \hat{\mathbf{c}}^{\top} \big(\mathbf{w}^*(\mathbf{c}) - \mathbf{w}\big).$$

This is a convex function of $\hat{\mathbf{c}}$ as an average of linear forms, with subgradient $g_{\mathrm{contr}} = \frac{1}{|\Gamma|-1} \sum_{\mathbf{w} \in \Gamma \setminus \{\mathbf{w}^*(\mathbf{c})\}} (\mathbf{w}^*(\mathbf{c}) - \mathbf{w}) \in \partial_{\hat{\mathbf{c}}} L_{\mathrm{contr}}(\hat{\mathbf{c}}; \mathbf{c})$. It encodes a structured margin that prefers the decision $\mathbf{w}^*(\mathbf{c})$ against competitors in Γ.

These constructions have a common consequence for EDO: each surrogate family is convex in the optimization argument and provides a subgradient that we may combine with $\nabla f(\hat{\mathbf{c}})$ to form a legitimate composite descent direction. In particular, the energy $F = f + \tilde{R}$ can be used as a Lyapunov function even when the actual direction fed to the algorithm is an inexact $\tilde{\mathbf{g}}_t$. The legality of the underlying subgradient prevents the update from "pushing uphill" in a way that would invalidate dissipation. Formal statements and proofs, including the cases where $\tilde{R}$ is represented via a stochastic estimator of g_{pert} or via a finite oracle for L_{upper}, are deferred to Appendix C.1.

3.2 Localized Gradient Stabilization

Once we know that surrogate gradients are biased but still provide legal subgradients, the next question is how to prevent their error from accumulating over time. Suppose at iteration t the algorithm observes an inexact surrogate gradient $\tilde{\mathbf{g}}_t$ satisfying $\|\tilde{\mathbf{g}}_t - \mathbf{g}_t\| \le \delta$, where the exact composite gradient is $\mathbf{g}_t = \nabla f(\hat{\mathbf{c}}_t) + g_t$ with $g_t \in \partial \tilde{R}(\hat{\mathbf{c}}_t)$. If one were to apply a plain first-order step, the bias $\varepsilon_t = \tilde{\mathbf{g}}_t - \mathbf{g}_t$ would shift the iterate linearly and produce the unbounded drift discussed in Section 2. The central idea of EDO is to replace this fragile rule with a local re-minimization that reanchors each update within a quadratic energy well. Formally, we let the next iterate solve the quadratic subproblem

$$\hat{\mathbf{c}}_{t+1} = \arg\min_{\mathbf{u} \in \mathbb{R}^n} \Big\{ \tilde{\mathbf{g}}_t^{\top}(\mathbf{u} - \hat{\mathbf{c}}_t) + \tfrac{1}{2\eta} \|\mathbf{u} - \hat{\mathbf{c}}_t\|^2 + \tilde{R}(\mathbf{u}) \Big\}. \tag{1}$$

This construction has two effects. The quadratic term enforces a trust region that penalizes large deviations, preventing the iterate from drifting too far even if $\tilde{\mathbf{g}}_t$ is misaligned. At the same time, the surrogate structure $\tilde{R}$ remains embedded in the subproblem, so that the update is still sensitive to decision regret. Together, these forces transform each step into an implicit descent on the composite energy $F = f + \tilde{R}$. Although solving equation 1 exactly would be costly, it admits an equivalent implicit update form.

Lemma 3.1 (Implicit update form)**.** *If f is convex and differentiable and $\tilde{R}$ is convex, the solution of equation 1 satisfies*

$$\hat{\mathbf{c}}_{t+1} = \hat{\mathbf{c}}_t - \eta \nabla f(\hat{\mathbf{c}}_t) - \eta \tilde{g}_t, \qquad \tilde{g}_t \in \partial \tilde{R}(\hat{\mathbf{c}}_{t+1}).$$

This equation shows that EDO does not simply follow $\tilde{\mathbf{g}}_t$; rather, it enforces consistency between the surrogate subgradient and the new iterate. In other words, the update is implicitly defined by the energy landscape

itself. All auxiliary components of the optimizer—momentum, adaptivity, or regularization—are layered on top of this unified descent vector $\nabla f(\hat{\mathbf{c}}_t) + \tilde{g}_t$.

The benefit of this implicit construction is captured by two guarantees. First, each step decreases the composite energy, up to a small residual that scales with the error magnitude δ. Second, the cumulative deviation from the ideal trajectory remains uniformly bounded, in stark contrast to the linear drift of plain gradient descent.

Theorem 3.2 (One-step energy decrease). *If f is L-smooth and μ-strongly convex and $\eta \in (0, 1/L]$, then*

$$F(\hat{\mathbf{c}}_{t+1}) \leq F(\hat{\mathbf{c}}_t) - \left(\tfrac{1}{4\eta} - \tfrac{L}{2}\right)\|\hat{\mathbf{c}}_{t+1} - \hat{\mathbf{c}}_t\|^2 + \eta\delta^2.$$

In particular, if $\delta^2 < \mu^2/(4L)$, the gain term dominates and F strictly decreases.

Theorem 3.3 (Uniform error bound). *Let $e_t = \hat{\mathbf{c}}_t - \hat{\mathbf{c}}_t^\star$ denote the deviation from the ideal proximal trajectory. If $\eta \in (0, 1/L]$, then for all $t \geq 0$*

$$\|e_t\| \;\leq\; (1 - \eta\mu)^t\|e_0\| + \tfrac{\eta\delta}{\mu}.$$

As $t \to \infty$, the deviation converges to $\|e_\infty\| \leq \mathcal{O}(\delta/L)$.

These results make the contrast with SGD explicit. Whereas standard updates accumulate error indefinitely, EDO contracts error geometrically and then stabilizes at a bounded bias level. This restores the dissipative nature of training: the system may never be perfectly noiseless, but its energy consistently decreases and the long-term deviation remains under control. This is the structural reason why the trajectories observed in practice become smoother and decision quality more reliable once EDO is applied.

3.3 Strategies for Robust Updates in EDO

To make EDO effective in practice, we introduce stabilizing mechanisms that work in concert with the proximal update. A first consideration is weight decay. Traditional optimizers such as AdamW apply a fixed decay rate, independent of step size. This design is harmless when gradients are unbiased, but with inexact surrogates it becomes problematic: early steps may be under-regularized, while later steps may be overly penalized, distorting the trajectory. EDO addresses this imbalance by coupling decay strength to the effective step size. Formally, we introduce a pre-scaled weight decay factor

$$\hat{\mathbf{c}}_{t+1} = \frac{1}{1 + \alpha\lambda}\Big(\bar{\mathbf{c}}_t - \eta_t \odot \hat{\mathbf{m}}_t\Big),$$

where λ is a decay coefficient, η_t is the adaptive step, and $\bar{\mathbf{c}}_t$ is a smoothed parameter vector defined below. The contraction factor $1/(1 + \alpha\lambda)$ ensures that larger steps—where gradient noise is more likely—are automatically regularized more strongly. This coupling between step size and decay introduces a stabilizing friction term in the energy dynamics, one that adjusts itself to the uncertainty of each update.

A second mechanism is temporal averaging. Rather than using exponential moving averages only at test time, we integrate them directly into the update rule:

$$\bar{\mathbf{c}}_t = \gamma\bar{\mathbf{c}}_{t-1} + (1 - \gamma)\hat{\mathbf{c}}_t.$$

This averaged parameter acts as a low-pass filter on the trajectory. Where surrogate errors induce high-frequency oscillations, temporal averaging damps their impact before they propagate forward. In the energy picture, it is as if the system carries an inertial memory that smooths sudden perturbations, preventing transient noise from accumulating into long-term bias.

On top of these new components, EDO retains the familiar stabilizers from adaptive methods. First-order momentum accumulates the gradient history, second-moment estimates track variance across coordinates, and bias correction ensures accurate scaling in the early stages (Kingma, 2014; Reddi et al., 2019).

Combining all pieces, the full EDO iteration becomes

$$\hat{\mathbf{c}}_{t+1} = \frac{1}{1+\alpha\lambda}\Big(\bar{\mathbf{c}}_t - \eta_t \odot \hat{\mathbf{m}}_t\Big), \qquad \bar{\mathbf{c}}_t = \gamma\bar{\mathbf{c}}_{t-1} + (1-\gamma)\hat{\mathbf{c}}_t,$$

with $\hat{\mathbf{m}}_t$ and η_t defined by the adaptive moment scheme. This form makes explicit how energy dissipation is reinforced by contraction, smoothing, and adaptivity at once.

The convergence properties of this enriched update can still be established under convexity and bounded-gradient assumptions. The following theorem summarizes the result.

Theorem 3.4 (Convergence of EDO with robust strategies)**.** *Assume f is convex and differentiable, $\tilde{R}$ is convex and possibly nonsmooth, and all gradients are bounded by a constant G_∞. Suppose the momentum parameters $\beta_1, \beta_2 \in [0,1)$ satisfy $\frac{\beta_1^2}{\sqrt{\beta_2}} < 1$, and let the learning rate $\alpha > 0$ and decay coefficient $\lambda \geq 0$ be fixed. Then for any horizon T, the cumulative regret satisfies*

$$\sum_{t=1}^{T}\big(F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star)\big) \;\leq\; \frac{D^2}{2\alpha(1-\beta_1)}\sum_{i=1}^{d}\sqrt{\hat{v}_{T,i}} + \frac{\alpha G_\infty^2}{(1-\beta_1)^2(1-\beta_2)}\,T,$$

where $\hat{v}_{t,i}$ is the corrected variance estimate and D bounds the parameter distance.

This result shows that the additional strategies preserve convergence and control cumulative regret. Even when surrogate gradients remain persistently biased, the error does not accumulate without bound.

3.4 Global Convergence and Long-Horizon Stability

The guarantees established so far concern the local behavior of EDO: each step decreases the composite energy up to a controlled residual, and the error relative to the ideal trajectory remains bounded. However deep learning is not a short-horizon procedure. Training typically runs for thousands of iterations, where even small biases can accumulate into significant deviations if not properly controlled. The crucial question is whether the dissipative structure introduced by EDO extends globally, ensuring that learning remains stable as $t \to \infty$.

From an optimization perspective, surrogate gradients introduce a persistent bias rather than stochastic variance. Classical adaptive methods such as Adam are designed under the assumption that gradients are unbiased, and their convergence proofs hinge on the cancellation of noise in expectation. In our setting this assumption is violated: the error $\varepsilon_t = \tilde{\mathbf{g}}_t - \mathbf{g}_t$ is structural, not random, and no amount of averaging can eliminate it. What EDO achieves instead is to transform this bias into a bounded steady-state deviation. The Lyapunov function $F = f + \tilde{R}$ ensures that the system cannot drift indefinitely; it converges toward an equilibrium corridor whose width is proportional to δ, the magnitude of surrogate error.

This intuition is captured by the following result, which combines the proximal stability with the adaptive update strategies.

Theorem 3.5 (Global convergence of EDO)**.** *Suppose f is convex and differentiable, $\tilde{R}$ is convex and possibly nonsmooth, and gradients are bounded by G_∞. Let the momentum parameters $\beta_1, \beta_2 \in [0,1)$ satisfy $\frac{\beta_1^2}{\sqrt{\beta_2}} < 1$, and set the coordinate-wise step $\boldsymbol{\eta}_t = \alpha/(\sqrt{\hat{\mathbf{v}}_t} + \epsilon)$ with $\alpha > 0$ and decay $\lambda \geq 0$. Then for all horizons $T \geq 1$,*

$$\frac{1}{T}\sum_{t=1}^{T}\big(F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star)\big) = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right) + \mathcal{O}(\delta),$$

where $\hat{\mathbf{c}}^\star$ denotes the minimizer of the exact composite energy.

The theorem states that EDO achieves the same sublinear convergence rate as classical adaptive methods, but with an additional $\mathcal{O}(\delta)$ bias floor. This bias is unavoidable: no optimizer can recover the exact decision gradient when only surrogates are available. What matters is that the error does not grow with T. Instead, it saturates at a constant level, so that long-horizon training remains stable and predictable.

From the energy viewpoint, the picture is that of a dissipative system driven by a weak external force. Each step reduces energy, but the biased gradients re-inject a small amount back. EDO balances the two effects: dissipation dominates injection, and the system settles into a bounded orbit around the true equilibrium. This explains why, in practice, learning curves under EDO no longer oscillate or drift, but approach a stable regime with low regret. It is not the absence of error that makes the process robust, but the fact that error is prevented from accumulating indefinitely.

4 Experiments

Our analysis suggests a simple prediction: when gradients are structurally inexact, optimizers that enforce dissipation should train more steadily and reach good decisions with less effort. We now examine whether this picture holds in practice. The emphasis is not only on final optimality, but also on the *trajectory* of training—how quickly energy drops to a useful level, and how much the iterates oscillate en route.

4.1 Experimental Setup

We evaluate the Energy-Dissipative Optimizer (EDO) on four decision-focused benchmarks that exhibit different combinatorial structures and gradient behaviors. The *Sales (0–1 Knapsack)* task selects real-estate projects under a budget to maximize profit; *Portfolio Optimization* allocates weights subject to budget and risk constraints; *Shortest Path* finds a minimum-cost path on a directed graph from predicted edge costs; *Energy Scheduling* assigns jobs to machines over time while respecting capacity and timing constraints. All tasks follow a unified predict–optimize pipeline so that differences arise from the optimizer rather than modeling idiosyncrasies. Formal problem statements and dataset statistics are given in Appendix C.3.

To probe behavior under gradient inexactness, we train with five surrogate regret formulations. Three are convex and supply valid subgradients—IMLE (Niepert et al., 2021), CMAP (Mulamba et al., 2021), and SPO+ (Elmachtoub & Grigas, 2022)—and two are non-convex (DBB (Pogančić et al., 2020), NID (Sahoo et al.)), included to test generalization beyond our theory. For each task–surrogate pair, we keep the predictive architecture fixed and replace only the optimizer.

Baselines include AdaGrad (Duchi et al., 2011), RMSProp (Tieleman & Hinton, 2012), AdaDelta (Zeiler, 2012), Adam (Kingma, 2014), and AdamW (Loshchilov & Hutter, 2017). All methods share the same initialization, early-stopping rule, and hyperparameter budget. Learning rates are selected from $\{10^{-5}, 10^{-4}, 10^{-3}\}$ on a validation split; early stopping uses patience $= 10$ based on validation performance. Unless otherwise noted, we report the mean over 10 independent runs with seed $= 2024$ and include standard errors. The codebase is implemented in PyTorch (Paszke et al., 2019) with Gurobi (Gurobi Optimization, LLC, 2023) as the solver backend, and differentiable integration follows PyEPO (Tang & Khalil, 2022). Experiments run on a single workstation (Intel i7-13700K, 32GB RAM, RTX 4070 Ti).

4.2 Evaluation Protocol and Metrics

Our theory speaks in the language of energy dissipation. Accordingly, we complement standard endpoint metrics with measurements of training dynamics derived from the same logs.

Time-to-target. To compare efficiency fairly across methods that differ in per-epoch cost, we report the wall-clock time needed to reach a target accuracy, either as an energy threshold $F(\hat{\mathbf{c}}_t) \leq F^\star + \epsilon$ or as a decision metric threshold $\mathrm{gap} \leq \epsilon$. This single figure captures both the number of updates and their cost.

Decision quality. We report the final optimality gap $\mathrm{gap} = \frac{z(\hat{\mathbf{c}}) - z(\mathbf{c})}{|z(\mathbf{c})|}$, where $z(\cdot)$ is the downstream objective under predicted and ground-truth parameters. Alongside means and standard errors, we include paired nonparametric significance tests (Wilcoxon signed-rank across 10 runs) when comparing optimizers within the same task–surrogate.

4.3 Convergence and Training Efficiency

We begin with efficiency. Theory predicts that an implicit, dissipative update should reduce the number of steps needed to reach a useful energy level, without incurring prohibitive per–step overhead. To compare methods fairly when their per–epoch costs differ, we report *time-to-target*: the wall-clock time to reach the early-stopping criterion , alongside the usual number of epochs and per-epoch time.

Task	Optimizer	Epochs (↓)	Time/Epoch (s) (↓)	Time-to-Target (s) (↓)
Sales	Adam	28.40	25.65	728.46
	AdaDelta	30.80	35.76	1102.61
	AdaGrad	24.80	27.06	**671.09**
	AdamW	37.80	44.58	1684.12
	RMSProp	23.80	38.72	921.54
	EDO (Ours)	**19.85**	33.87	672.32
Portfolio	Adam	37.80	227.29	8591.56
	AdaDelta	26.60	190.48	5066.77
	AdaGrad	23.80	191.12	4548.66
	AdamW	33.20	228.82	7596.82
	RMSProp	31.40	230.14	7226.40
	EDO (Ours)	**20.62**	**151.98**	**3130.88**
\$hortest Path	Adam	23.20	27.28	632.90
	AdaDelta	22.60	39.53	892.38
	AdaGrad	22.60	39.50	891.70
	AdamW	22.60	27.48	621.65
	RMSProp	14.60	22.41	327.19
	EDO (Ours)	**8.72**	**20.70**	**180.50**
Scheduling	Adam	23.20	27.28	632.90
	AdaDelta	22.60	39.53	892.38
	AdaGrad	22.60	39.50	891.70
	AdamW	22.60	27.48	621.65
	RMSProp	14.60	22.41	327.19
	EDO (Ours)	**7.84**	**20.70**	**162.29**

Table 1: Efficiency measured by epochs, per-epoch time, and *time-to-target* (wall-clock to early-stopping threshold). EDO attains the target with fewer steps and lower total time on three of four tasks; on *Sales* it ties the best baseline in total time while using fewer epochs than Adam.

Table 1 summarizes results across four tasks. EDO reduces the number of epochs by 30–66% relative to Adam (e.g., Path 8.72 vs. 23.20, Energy 7.84 vs. 23.20), and this translates into clear *time-to-target* gains on three tasks. On *Portfolio*, EDO attains the target in $3{,}131$ s, a 31% reduction vs. the best baseline (Adagrad at $4{,}549$ s), and $> 60\%$ vs. Adam. On *Path* and *Energy*, where surrogate noise is most pronounced, EDO roughly halves the time-to-target relative to the strongest baseline (RMSProp). On *Sales*, EDO is on par with the best baseline (Adagrad), differing by $< 0.2\%$ in total time, while still using markedly fewer epochs than Adam. These patterns are consistent with the dissipative picture: fewer oscillations and less drift mean fewer effective steps to threshold.

4.4 Component Ablation Analysis

The final step is to disentangle which parts of EDO are responsible for the observed stability. The implicit energy step (IES), adaptive scaling, momentum, and pre-scaled weight decay are intertwined in the full update rule. Removing them one at a time helps to clarify their role. We stress that this is not a search for the

Table 2: Ablation study: average regret of EDO and its variants across convex (IMLE, CMAP, SPO) and non-convex (DBB, NID) surrogates. Parentheses denote relative increase (%) in regret compared to full EDO. Red indicates the largest degradation per row.

Benchmark	Variant	IMLE	CMAP	SPO	DBB	NID
Sales	EDO (ours)	**0.34**	**0.32**	**0.22**	**0.34**	**0.34**
	NoImplicit	0.62 (↑82%)	0.64 (↑100%)	0.36 (↑64%)	0.47 (↑38%)	0.62 (↑82%)
	NoAdaptive	0.51 (↑50%)	0.36 (↑13%)	0.28 (↑27%)	0.42 (↑24%)	0.36 (↑6%)
	NoMomentum	0.49 (↑44%)	0.62 (↑94%)	0.38 (↑73%)	0.43 (↑26%)	0.61 (↑79%)
	NoDecay	0.53 (↑56%)	0.34 (↑6%)	0.33 (↑50%)	0.39 (↑15%)	0.61 (↑79%)
Portfolio	EDO (ours)	**0.05**	**0.11**	**0.02**	**0.17**	**0.18**
	NoImplicit	0.18 (↑260%)	0.29 (↑164%)	0.18 (↑800%)	0.19 (↑12%)	0.18 (↑0%)
	NoAdaptive	0.16 (↑220%)	0.13 (↑18%)	0.17 (↑750%)	0.17 (↑0%)	0.18 (↑0%)
	NoMomentum	0.18 (↑260%)	0.10 (↓9%)	0.12 (↑500%)	0.18 (↑6%)	0.19 (↑6%)
	NoDecay	0.15 (↑200%)	0.13 (↑18%)	0.03 (↑50%)	0.17 (↑0%)	0.18 (↑0%)
Path	EDO (ours)	**0.15**	**0.09**	**0.11**	**0.33**	**0.28**
	NoImplicit	0.40 (↑167%)	0.14 (↑56%)	0.13 (↑18%)	0.35 (↑6%)	0.40 (↑43%)
	NoAdaptive	0.16 (↑7%)	0.31 (↑244%)	0.12 (↑9%)	0.36 (↑9%)	0.34 (↑21%)
	NoMomentum	0.31 (↑107%)	0.12 (↑33%)	0.11 (↑0%)	0.35 (↑6%)	0.32 (↑14%)
	NoDecay	0.28 (↑87%)	0.10 (↑11%)	0.11 (↑0%)	0.33 (↑0%)	0.29 (↑4%)
Energy	EDO (ours)	**0.14**	**0.13**	**0.10**	**0.25**	**0.24**
	NoImplicit	0.32 (↑129%)	0.22 (↑69%)	0.21 (↑110%)	0.29 (↑16%)	0.31 (↑29%)
	NoAdaptive	0.26 (↑86%)	0.28 (↑115%)	0.18 (↑80%)	0.33 (↑32%)	0.30 (↑25%)
	NoMomentum	0.28 (↑100%)	0.17 (↑31%)	0.12 (↑20%)	0.28 (↑12%)	0.29 (↑21%)
	NoDecay	0.23 (↑64%)	0.16 (↑23%)	0.11 (↑10%)	0.26 (↑4%)	0.26 (↑8%)

"best tweak" but an empirical test of the structural claim: training under biased surrogate gradients requires error-damping mechanisms at every step.

When the IES is removed (variant *NoImplicit*), the regret values explode across all benchmarks, in some cases by more than 200%. This is consistent with the theoretical analysis: without the implicit step, error behaves like a linear drift, and early misalignment is never corrected. The uniform bound in Theorem 3.3 disappears in practice, producing large variance and poor decision quality.

Suppressing coordinate-wise adaptivity (*NoAdaptive*) results in clear fragility on non-convex surrogates such as DBB and NID. With adaptivity, EDO can absorb these shifts smoothly, avoiding divergence. Removing momentum (*NoMomentum*) destabilizes training whenever gradients are highly irregular. In CMAP and DBB, where local slopes switch signs frequently, regret grows by 30–90% compared to the full EDO. Momentum plays a dual role: it accelerates convergence and acts as a temporal smoother against oscillations.

Finally, eliminating weight decay (*NoDecay*) produces smaller but consistent increases in regret, especially near convergence. The effect is less dramatic than IES or momentum, but it confirms that contracting parameters proportionally to the step size reduces late-stage wandering caused by surrogate mismatch.

5 Conclusion

This paper introduces the Energy Dissipation Optimizer (EDO), a framework that interprets decision-oriented learning as an energy dissipation process and achieves training stability through implicit descent. By reconstructing dissipative dynamics, EDO effectively mitigates the impact of biased proxy gradients, achieving smoother convergence and more reliable decision outcomes across diverse benchmarks. While the findings validate this perspective's practical value, the theoretical foundations of learning under structurally biased gradients warrant further exploration, opening extensive avenues for future research.

6 Reproducibility Statement

We have taken several measures to ensure the reproducibility of our work. The theoretical results are presented with explicit assumptions, and complete proofs of all claims are included in the appendix. Experimental details, including benchmark tasks, hyperparameters, and evaluation metrics, are documented in the main text and further elaborated in the appendix.

7 Ethics Statement

This work adheres to the ICLR Code of Ethics. Our study does not involve human subjects, sensitive personal data, or applications with foreseeable harmful consequences. The benchmarks used are publicly available and widely adopted, and all experiments comply with standard practices in decision-focused learning and optimization research. We are committed to transparency in code release, proper attribution of prior work, and the integrity of reported results.

References

Amirhossein Ajalloeian, Andrea Simonetto, and Emiliano Dall'Anese. Inexact online proximal-gradient method for time-varying convex optimization. In *2020 American Control Conference (ACC)*, pp. 2850–2857, 2020. doi: 10.23919/ACC45564.2020.9147467.

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.

Mathieu Barré, Adrien B Taylor, and Francis Bach. Principled analyses and design of first-order methods with inexact proximal operators. *Mathematical Programming*, 201(1):185–230, 2023.

Nicola Bastianello and Emiliano Dall'Anese. Distributed and inexact proximal gradient method for online convex optimization. In *2021 European Control Conference (ECC)*, pp. 2432–2437, 2021. doi: 10.23919/ECC54610.2021.9654953.

Quentin Berthet, Mathieu Blondel, Olivier Teboul, Marco Cuturi, Jean-Philippe Vert, and Francis Bach. Learning with differentiable perturbed optimizers. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9508–9519, 2020.

Yury Demidovich, Grigory Malinovsky, Igor Sokolov, and Peter Richtarik. A guide through the zoo of biased sgd. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 23158–23171. Curran Associates, Inc., 2023.

Emir Demirović, Peter J Stuckey, James Bailey, Jeffrey Chan, Christopher Leckie, Kotagiri Ramamohanarao, and Tias Guns. Predict+ optimise with ranking objectives: Exhaustively learning linear functions. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 1078–1085. International Joint Conferences on Artificial Intelligence, 2019.

Priya L Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. In *Advances in Neural Information Processing Systems*, pp. 5484–5494, 2017.

Priya L. Donti, David Rolnick, and J. Zico Kolter. Dc3: A learning method for optimization with hard constraints. 4 2021. URL http://arxiv.org/abs/2104.12225.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Adam N Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management Science*, 68(1):9–26, 2022.

Adam N Elmachtoub, Jason Cheuk Nam Liang, and Ryan McNellis. Decision trees for decision-making under the predict-then-optimize framework. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 2858–2867, 2020.

Aaron M Ferber, Taoan Huang, Daochen Zha, Martin Schubert, Benoit Steiner, Bistra Dilkina, and Yuandong Tian. Surco: Learning linear surrogates for combinatorial nonlinear optimization problems. In *International Conference on Machine Learning*, pp. 10034–10052. PMLR, 2023.

Haoyu Geng, Hang Ruan, Runzhong Wang, Yang Li, Yang Wang, Lei Chen, and Junchi Yan. Benchmarking pto and pno methods in the predictive combinatorial optimization regime. *Advances in Neural Information Processing Systems*, 37:65944–65971, 2024.

Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*, 2023.

Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. *Advances in Neural Information Processing Systems*, 33: 2304–2315, 2020.

Paula Harder, Alex Hernandez-Garcia, Venkatesh Ramesh, Qidong Yang, Prasanna Sattegeri, Daniela Szwarcman, Campbell Watson, and David Rolnick. Hard-constrained deep learning for climate downscaling. *Journal of Machine Learning Research*, 24(365):1–40, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Xinyi Hu, Jasper Lee, and Jimmy Lee. Two-stage predict+optimize for milps with unknown parameters in constraints. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 14247–14272. Curran Associates, Inc., 2023a.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17853–17862, 2023b.

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

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

Jannis Mandi, Victor Bucarey, Mourad Mulamba, and Tias Guns. Decision-focused learning: through the lens of learning to rank. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

Jayanta Mandi and Tias Guns. Interior point solving for lp-based prediction+optimisation. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7272–7282, 2020.

Maxime Mulamba, Jayanta Mandi, Michelangelo Diligenti, Michele Lombardi, Victor Bucarey, and Tias Guns. Contrastive losses and solution caching for predict-and-optimize. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2833–2840. International Joint Conferences on Artificial Intelligence, 2021.

Mathias Niepert, Pasquale Minervini, and Luca Franceschi. Implicit mle: backpropagating through discrete exponential family distributions. *Advances in Neural Information Processing Systems*, 34:14567–14579, 2021.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pp. 8024–8035, 2019.

Marin Vlastelica Pogančić, Anselm Paulus, Vit Musil, Georg Martius, and Michal Rolinek. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*, 2020.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Subham Sekhar Sahoo, Anselm Paulus, Marin Vlastelica, Vít Musil, Volodymyr Kuleshov, and Georg Martius. Backpropagation through combinatorial algorithms: Identity with projection works. In *The Eleventh International Conference on Learning Representations*.

Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. In *International conference on machine learning*, pp. 343–351. PMLR, 2013.

Mark Schmidt, Nicolas Roux, and Francis Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

Jascha Sohl-Dickstein, Ben Poole, and Surya Ganguli. Fast large-scale optimization by unifying stochastic gradient and quasi-newton methods. In *International Conference on Machine Learning*, pp. 604–612. PMLR, 2014.

Peter J Stuckey, Tias Guns, James Bailey, Christopher Leckie, Kotagiri Ramamohanarao, Jeffrey Chan, et al. Dynamic programming for predict+ optimise. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 1444–1451, 2020.

Bo Tang and Elias B. Khalil. Pyepo: A pytorch-based end-to-end predict-then-optimize library for linear and integer programming. 6 2022. URL `http://arxiv.org/abs/2206.14234`.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop, coursera: Neural networks for machine learning. *University of Toronto, Technical Report*, 6, 2012.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Jiahuan Wang and Hong Chen. Towards stability and generalization bounds in decentralized minibatch stochastic gradient descent. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 15511–15519, 2024.

Kai Wang, Shresth Verma, Aditya Mate, Sanket Shah, Aparna Taneja, Neha Madhiwalla, Aparna Hegde, and Milind Tambe. Scalable decision-focused learning in restless multi-armed bandits with application to maternal and child health. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 12138–12146, 2023.

Yingzhen Yang and Ping Li. Projective proximal gradient descent for nonconvex nonsmooth optimization: Fast convergence without kurdyka-lojasiewicz (kl) property. In *The Eleventh International Conference on Learning Representations*, 2023.

Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

A USE OF LLMS

Large language models (LLMs) were used in the preparation of this manuscript to assist with translation, language polishing, and stylistic refinement. The scientific contributions, results, and interpretations are entirely the authors' own.

B RELATED WORKS

Decision-Focused Learning Decision-Focused Learning (DFL) addresses a class of parametric optimization problems in an end-to-end fashion: a predictive model estimates the problem parameters, and the loss depends on the quality of the downstream decision. The main challenge is that decision regret is typically non-differentiable with respect to the predictions, making direct backpropagation infeasible.

Two main research directions have emerged. One develops differentiable layers for specific optimization problems, embedding the solver inside a neural network. Examples include stochastic optimization (Donti et al., 2017), quadratic programming (Amos & Kolter, 2017), integer programming (Mandi & Guns, 2020), and constrained optimization (Donti et al., 2021; Hu et al., 2023a). The other line constructs surrogate objectives to obtain approximate gradients, using techniques such as convex upper bounds (Elmachtoub & Grigas, 2022), dynamic programming relaxations (Stuckey et al., 2020), decision-tree surrogates (Elmachtoub et al., 2020), black-box differentiation (Pogančić et al., 2020), perturbation methods (Berthet et al., 2020), rank-based formulations (Mandi et al., 2022), contrastive optimization (Mulamba et al., 2021), and linearized relaxations (Ferber et al., 2023).

These studies provide ways to obtain gradients, but the update process that follows has received much less attention. Unlike standard end-to-end learning where gradients are exact (or unbiased stochastic estimates), the gradients produced by surrogates are structurally biased. This inexactness poses new challenges for optimization dynamics.

Inexact Gradient Methods The optimization community has studied inexact gradients in various classical settings. Early work by (Schmidt et al., 2011) analyzed inexact proximal-gradient methods for convex optimization and established convergence rates. Subsequent work extended this to online and distributed contexts, including inexact updates for time-varying convex programs (Ajalloeian et al., 2020) and distributed online optimization (Bastianello & Dall'Anese, 2021). More recent studies explored principled analyses of inexact proximal operators (Barré et al., 2023) and convergence guarantees in nonconvex and nonsmooth settings via the Kurdyka–Łojasiewicz property (Yang & Li, 2023).

These results show that optimization can remain stable under certain forms of inexactness. However, the sources of error in DFL differ fundamentally: they arise not from numerical approximation or sampling, but from the structural misalignment between surrogate gradients and ideal decision descent directions. Existing analyses do not directly explain or address the instability observed in DFL training.

Optimizer Stochastic gradient descent has inspired many adaptive variants designed for large-scale learning, such as AdaGrad (Duchi et al., 2011), RMSProp (Tieleman & Hinton, 2012), AdaDelta (Zeiler, 2012), SFO (Sohl-Dickstein et al., 2014), vSGD (Schaul et al., 2013), and Adam (Kingma, 2014). These methods assume access to exact or unbiased stochastic gradients. More recent work has studied extensions to specialized settings, such as federated learning (Hanzely et al., 2020), decentralized training (Wang & Chen, 2024), or biased stochastic gradients (Demidovich et al., 2023). While these methods broaden the scope of optimization, none are designed to cope with the persistent structural bias inherent in surrogate-based DFL training.

To summarize, prior work has mainly advanced two fronts: (i) designing surrogate objectives to obtain gradients in DFL, and (ii) analyzing inexact optimization in classical settings. However, the critical issue remains

unresolved: *how to stabilize training when surrogate gradients are inherently inexact and systematically misaligned.* Existing optimizers treat such gradients as if they were exact or unbiased, which breaks the energy-dissipative structure of learning and causes unstable dynamics. Our work closes this gap by introducing an optimizer, EDO, that explicitly restores dissipative dynamics under inexact surrogate gradients.

C Theoretical Results and Proofs

C.1 Proof of Subgradient in Surrogate families

Lemma 1 (Convexity and subgradients of APROX surrogates)**.** *Let $\mathbf{W} \subset \mathbb{R}^n$ be a non–empty polyhedral set and $\mathbf{w}^*(\mathbf{c}) := \arg\min_{\mathbf{w}\in\mathbf{W}} \mathbf{c}^\top \mathbf{w}$ (the ideal optimal decision). Define the three surrogate losses*

$$L_{\text{upper}}(\hat{\mathbf{c}}; \mathbf{c}) = -\min_{\mathbf{w}\in\mathbf{W}} (2\hat{\mathbf{c}} - \mathbf{c})^\top \mathbf{w} + 2\hat{\mathbf{c}}^\top \mathbf{w}^*(\mathbf{c}) - \mathbf{c}^\top \mathbf{w}^*(\mathbf{c}),$$

$$L_{\text{pert}}(\mathbf{c}; \hat{\mathbf{c}}) = \mathbb{E}_{\hat{\mathbf{z}}\sim q(\mathbf{z};\hat{\mathbf{c}})}\big[A(\mathbf{c}) - \langle \hat{\mathbf{z}}, \mathbf{c}\rangle\big],$$

$$L_{\text{contr}}(\hat{\mathbf{c}}; \mathbf{c}) = \frac{1}{|\Gamma| - 1} \sum_{\mathbf{w}\in\Gamma\setminus\{\mathbf{w}^*(\mathbf{c})\}} \hat{\mathbf{c}}^\top \big(\mathbf{w}^*(\mathbf{c}) - \mathbf{w}\big),$$

where $\Gamma \subset \mathbf{W}$ is finite and contains $\mathbf{w}^*(\mathbf{c})$, and A is the log–partition function of an exponential–family model with mean map $\boldsymbol{\mu}(\mathbf{c}) := \nabla A(\mathbf{c})$. Set

$$g_{\text{upper}} = 2\mathbf{w}^*(\mathbf{c}) - 2\mathbf{w}^\star, \qquad \mathbf{w}^\star \in \arg\min_{\mathbf{w}\in\mathbf{W}} (2\hat{\mathbf{c}} - \mathbf{c})^\top \mathbf{w},$$

$$g_{\text{pert}} = \boldsymbol{\mu}(\mathbf{c}) - \boldsymbol{\mu}(\hat{\mathbf{c}}),$$

$$g_{\text{contr}} = \frac{1}{|\Gamma| - 1} \sum_{\mathbf{w}\in\Gamma\setminus\{\mathbf{w}^*(\mathbf{c})\}} \big(\mathbf{w}^*(\mathbf{c}) - \mathbf{w}\big).$$

(a) *$L_{\text{upper}}(\cdot; \mathbf{c})$ and $L_{\text{contr}}(\cdot; \mathbf{c})$ are convex in $\hat{\mathbf{c}}$, while $L_{\text{pert}}(\mathbf{c}; \hat{\mathbf{c}})$ is convex in $\mathbf{c}$.*

(b) *$g_{\text{upper}} \in \partial_{\hat{\mathbf{c}}} L_{\text{upper}}(\hat{\mathbf{c}}; \mathbf{c})$, $g_{\text{pert}} \in \partial_{\mathbf{c}} L_{\text{pert}}(\mathbf{c}; \hat{\mathbf{c}})$, $g_{\text{contr}} \in \partial_{\hat{\mathbf{c}}} L_{\text{contr}}(\hat{\mathbf{c}}; \mathbf{c})$.*

Proof. **Upper–bound surrogate.** Because $-\min_{\mathbf{w}}(2\hat{\mathbf{c}}-\mathbf{c})^\top \mathbf{w} = \max_{\mathbf{w}} -(2\hat{\mathbf{c}}-\mathbf{c})^\top \mathbf{w}$, L_{upper} is the sum of (i) a point-wise maximum of affine maps in $\hat{\mathbf{c}}$ (convex) and (ii) an affine term $2\hat{\mathbf{c}}^\top \mathbf{w}^*(\mathbf{c})$. Hence $L_{\text{upper}}(\cdot; \mathbf{c})$ is convex. For any maximiser $\mathbf{w}^\star$, the gradient of the active affine terms gives $g_{\text{upper}} = -2\mathbf{w}^\star + 2\mathbf{w}^*(\mathbf{c})$. By the definition of support functions, $L_{\text{upper}}(\hat{\mathbf{c}}', \mathbf{c}) \geq L_{\text{upper}}(\hat{\mathbf{c}}; \mathbf{c}) + g_{\text{upper}}^\top(\hat{\mathbf{c}}' - \hat{\mathbf{c}})$ for every $\hat{\mathbf{c}}'$, so $g_{\text{upper}} \in \partial_{\hat{\mathbf{c}}} L_{\text{upper}}$.

Perturbation surrogate. The log–partition function A is convex in $\mathbf{c}$, while $\langle \hat{\mathbf{z}}, \mathbf{c}\rangle$ is affine in $\mathbf{c}$. Taking an expectation w.r.t. $q(\cdot; \hat{\mathbf{c}})$ preserves convexity, giving convexity of $L_{\text{pert}}(\mathbf{c}; \hat{\mathbf{c}})$ in $\mathbf{c}$. Differentiating under the expectation yields $\nabla_{\mathbf{c}} L_{\text{pert}} = \boldsymbol{\mu}(\mathbf{c}) - \boldsymbol{\mu}(\hat{\mathbf{c}}) = g_{\text{pert}}$, and convexity of A implies the subgradient inequality, hence $g_{\text{pert}} \in \partial_{\mathbf{c}} L_{\text{pert}}$.

Contrastive surrogate. Each summand $\hat{\mathbf{c}}^\top(\mathbf{w}^*(\mathbf{c}) - \mathbf{w})$ is linear in $\hat{\mathbf{c}}$; their average is therefore linear and thus convex. Its gradient with respect to $\hat{\mathbf{c}}$ equals $\mathbf{w}^*(\mathbf{c}) - \mathbf{w}$, and averaging the gradients gives g_{contr}. For any $\hat{\mathbf{c}}'$, $L_{\text{contr}}(\hat{\mathbf{c}}'; \mathbf{c}) - L_{\text{contr}}(\hat{\mathbf{c}}; \mathbf{c}) = g_{\text{contr}}^\top(\hat{\mathbf{c}}' - \hat{\mathbf{c}})$, establishing $g_{\text{contr}} \in \partial_{\hat{\mathbf{c}}} L_{\text{contr}}$.

All claims in parts (a) and (b) are proved. □

C.2 Proof of Lemma 3.1

Proof. By the definition of the proximal operator, we have:

$$\hat{\boldsymbol{c}}_{t+1} = \mathrm{prox}_{\eta\tilde{R}}\left(\hat{\boldsymbol{v}}\right),$$

where

$$\hat{\boldsymbol{v}} = \hat{\boldsymbol{c}}_t - \eta\nabla f(\hat{\boldsymbol{c}}_t).$$

This means that $\hat{\boldsymbol{c}}_{t+1}$ is the minimizer of the following optimization problem:

$$\hat{\boldsymbol{c}}_{t+1} = \arg\min_{\hat{\boldsymbol{c}}} \left\{ \tilde{R}(\hat{\boldsymbol{c}}) + \frac{1}{2\eta} \left\| \hat{\boldsymbol{c}} - \hat{\boldsymbol{v}} \right\|^2 \right\}.$$

From the first-order optimality condition for convex functions, we have:

$$\mathbf{0} \in \partial\tilde{R}(\hat{\boldsymbol{c}}_{t+1}) + \frac{1}{\eta}\left(\hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{v}}\right),$$

where $\partial\tilde{R}(\hat{\boldsymbol{c}}_{t+1})$ denotes the subdifferential of $\tilde{R}$ at $\hat{\boldsymbol{c}}_{t+1}$.

Substituting $\hat{\boldsymbol{v}} = \hat{\boldsymbol{c}}_t - \eta\nabla f(\hat{\boldsymbol{c}}_t)$, we get:

$$\mathbf{0} \in \partial\tilde{R}(\hat{\boldsymbol{c}}_{t+1}) + \frac{1}{\eta}\left(\hat{\boldsymbol{c}}_{t+1} - (\hat{\boldsymbol{c}}_t - \eta\nabla f(\hat{\boldsymbol{c}}_t))\right).$$

Simplifying the expression inside the parentheses:

$$\hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{c}}_t + \eta\nabla f(\hat{\boldsymbol{c}}_t) = \hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{c}}_t + \eta\nabla f(\hat{\boldsymbol{c}}_t).$$

Therefore, the optimality condition becomes:

$$\mathbf{0} \in \partial\tilde{R}(\hat{\boldsymbol{c}}_{t+1}) + \frac{1}{\eta}\left(\hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{c}}_t + \eta\nabla f(\hat{\boldsymbol{c}}_t)\right).$$

Multiplying both sides by η:

$$\mathbf{0} \in \eta\,\partial\tilde{R}(\hat{\boldsymbol{c}}_{t+1}) + \left(\hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{c}}_t + \eta\nabla f(\hat{\boldsymbol{c}}_t)\right).$$

Rewriting the equation:

$$\hat{\boldsymbol{c}}_{t+1} = \hat{\boldsymbol{c}}_t - \eta\nabla f(\hat{\boldsymbol{c}}_t) - \eta\tilde{g}_t,$$

where $\tilde{g}_t \in \partial\tilde{R}(\hat{\boldsymbol{c}}_{t+1})$.

This demonstrates that the proximal gradient descent update can be expressed as a standard gradient descent step on f followed by a subgradient step on $\tilde{R}$.

□

A.2 Proof of Theorem 3.2

Proof. The L-smoothness of f gives the descent lemma

$$f(\hat{\boldsymbol{c}}_{t+1}) \;\le\; f(\hat{\boldsymbol{c}}_t) + \big\langle \nabla f(\hat{\boldsymbol{c}}_t), \boldsymbol{s}_t \big\rangle + \tfrac{L}{2}\|\boldsymbol{s}_t\|^2. \tag{S1}$$

By definition of the proximal operator,

$$\tilde{R}(\hat{\boldsymbol{c}}_t) \;\ge\; \tilde{R}(\hat{\boldsymbol{c}}_{t+1}) + \tfrac{1}{2\eta}\big(\|\hat{\boldsymbol{c}}_t - (\hat{\boldsymbol{c}}_t - \eta\tilde{\boldsymbol{g}}_t)\|^2 - \|\boldsymbol{s}_t + \eta\tilde{\boldsymbol{g}}_t\|^2\big),$$

which rearranges to

$$\tilde{R}(\hat{\boldsymbol{c}}_{t+1}) \;\le\; \tilde{R}(\hat{\boldsymbol{c}}_t) + \tfrac{1}{\eta}\big\langle \tilde{\boldsymbol{g}}_t, \boldsymbol{s}_t \big\rangle + \tfrac{1}{2\eta}\|\boldsymbol{s}_t\|^2. \tag{S2}$$

Add equation S1–equation S2 and substitute $\nabla f = \tilde{\boldsymbol{g}}_t - \boldsymbol{\varepsilon}_t$, with $\boldsymbol{\varepsilon}_t := \tilde{\boldsymbol{g}}_t - \boldsymbol{g}_t$:

$$F(\hat{\boldsymbol{c}}_{t+1}) \le F(\hat{\boldsymbol{c}}_t) + \Big(\tfrac{1}{\eta} - \tfrac{1}{\eta}\Big)\langle \tilde{\boldsymbol{g}}_t, \boldsymbol{s}_t\rangle - \langle \boldsymbol{\varepsilon}_t, \boldsymbol{s}_t\rangle + \Big(\tfrac{1}{2\eta} + \tfrac{L}{2}\Big)\|\boldsymbol{s}_t\|^2.$$

The first bracket vanishes, leaving

$$F(\hat{\boldsymbol{c}}_{t+1}) \le F(\hat{\boldsymbol{c}}_t) - \langle \boldsymbol{\varepsilon}_t, \boldsymbol{s}_t\rangle + \Big(\tfrac{1}{2\eta} + \tfrac{L}{2}\Big)\|\boldsymbol{s}_t\|^2.$$

Young's inequality yields $\big|\langle \boldsymbol{\varepsilon}_t, \boldsymbol{s}_t\rangle\big| \le \delta\|\boldsymbol{s}_t\| \le \eta\delta^2 + \frac{1}{4\eta}\|\boldsymbol{s}_t\|^2$. Substituting and collecting coefficients of $\|\boldsymbol{s}_t\|^2$ gives

$$F(\hat{\boldsymbol{c}}_{t+1}) \le F(\hat{\boldsymbol{c}}_t) - \Big(\tfrac{1}{4\eta} - \tfrac{L}{2}\Big)\|\boldsymbol{s}_t\|^2 + \eta\delta^2.$$

Because f is μ-strongly convex, $\|\boldsymbol{g}_t\| \ge \mu\|\boldsymbol{s}_t\|/\eta$, while $\|\boldsymbol{s}_t\| \le \eta\|\boldsymbol{g}_t\|$. Combining gives $\|\boldsymbol{s}_t\| \ge (\mu/L)$. Hence the model–gain term is at least $\big(\frac{1}{4\eta} - \frac{L}{2}\big)(\mu/L)^2$, which dominates $\eta\delta^2$ whenever $\delta^2 < \mu^2/(4L)$. □

□

A.3 Proof of Theorem 3.3

Proof. For any proper l.s.c. convex function $g : \mathbb{R}^n \to (-\infty, +\infty]$ and any $\eta > 0$, its proximal operator $\mathrm{prox}_{\eta g}(\boldsymbol{v}) := \arg\min_{\boldsymbol{u}}\big\{g(\boldsymbol{u}) + \frac{1}{2\eta}\|\boldsymbol{u} - \boldsymbol{v}\|^2\big\}$ is *firmly non-expansive*:

$$\big\|\mathrm{prox}_{\eta g}(\boldsymbol{v}) - \mathrm{prox}_{\eta g}(\boldsymbol{w})\big\|^2 \;\le\; \big\langle \boldsymbol{v} - \boldsymbol{w}, \mathrm{prox}_{\eta g}(\boldsymbol{v}) - \mathrm{prox}_{\eta g}(\boldsymbol{w})\big\rangle \quad \forall \boldsymbol{v}, \boldsymbol{w}. \tag{2}$$

Because f is L-smooth and μ-strongly convex, its gradient is $\frac{1}{L}$-cocoercive and μ-strongly monotone:

$$\|\nabla f(\boldsymbol{u}) - \nabla f(\boldsymbol{v})\| \;\le\; L\|\boldsymbol{u} - \boldsymbol{v}\| \qquad \text{(Lipschitz)} \tag{3}$$

$$\big\langle \nabla f(\boldsymbol{u}) - \nabla f(\boldsymbol{v}), \boldsymbol{u} - \boldsymbol{v}\big\rangle \;\ge\; \mu\|\boldsymbol{u} - \boldsymbol{v}\|^2 \qquad \text{(strong convexity).} \tag{4}$$

Define $G_\eta(\boldsymbol{u}) = \boldsymbol{u} - \eta\nabla f(\boldsymbol{u})$. A classical inequality states that for any $\eta \in (0, \frac{1}{L}]$,

$$\|G_\eta(\boldsymbol{u}) - G_\eta(\boldsymbol{v})\| \;\le\; (1 - \eta\mu)\,\|\boldsymbol{u} - \boldsymbol{v}\|. \tag{5}$$

Thus G_η is a $(1 - \eta\mu)$-contraction.

Let

$$\hat{\boldsymbol{c}}_{t+1} = \mathrm{prox}_{\eta\tilde{R}}\big(G_\eta(\hat{\boldsymbol{c}}_t) - \eta\boldsymbol{\varepsilon}_t\big), \quad \hat{\boldsymbol{c}}^\star_{t+1} = \mathrm{prox}_{\eta\tilde{R}}\big(G_\eta(\hat{\boldsymbol{c}}^\star_t)\big),$$

where $\|\boldsymbol{\varepsilon}_t\| \leq \delta$ by assumption (A3). Set $\boldsymbol{e}_t := \hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}_t^\star$ and denote $\boldsymbol{v}_t = G_\eta(\hat{\boldsymbol{c}}_t) - \eta\boldsymbol{\varepsilon}_t,\ \boldsymbol{w}_t = G_\eta(\hat{\boldsymbol{c}}_t^\star)$. Applying equation 2 with $g = \tilde{R}$, $\boldsymbol{v} = \boldsymbol{v}_t$, $\boldsymbol{w} = \boldsymbol{w}_t$ yields

$$\|\boldsymbol{e}_{t+1}\| \leq \|\boldsymbol{v}_t - \boldsymbol{w}_t\| = \|G_\eta(\hat{\boldsymbol{c}}_t) - G_\eta(\hat{\boldsymbol{c}}_t^\star) - \eta\boldsymbol{\varepsilon}_t\|.$$

Using equation 5 and $\|\boldsymbol{\varepsilon}_t\| \leq \delta$,

$$\|\boldsymbol{e}_{t+1}\| \leq (1 - \eta\mu)\,\|\boldsymbol{e}_t\| + \eta\delta. \tag{6}$$

Unfold equation 6:

$$\|\boldsymbol{e}_t\| \leq (1 - \eta\mu)^t \|\boldsymbol{e}_0\| + \eta\delta \sum_{k=0}^{t-1} (1 - \eta\mu)^k.$$

Because $\sum_{k=0}^{t-1}(1 - \eta\mu)^k = \frac{1-(1-\eta\mu)^t}{\eta\mu}$,

$$\|\boldsymbol{e}_t\| \leq (1 - \eta\mu)^t \|\boldsymbol{e}_0\| + \tfrac{\eta\delta}{\mu}\big[1 - (1 - \eta\mu)^t\big],$$

which implies the uniform bound equation **??** and, letting $t \to \infty$, $\|\boldsymbol{e}_\infty\| \leq \frac{\eta\delta}{\mu}$.

If $\delta = 0$ (exact surrogate gradient), equation 6 reduces to $\|\boldsymbol{e}_{t+1}\| \leq (1 - \eta\mu)\|\boldsymbol{e}_t\|$, so APROX recovers the optimal $\mathcal{O}((1 - \eta\mu)^t)$ rate for strongly convex objectives. When $\delta > 0$ the lower term $\frac{\eta\delta}{\mu}$ is unavoidable: it matches the bias of a single inexact step and therefore cannot be improved without reducing δ itself. Hence the bound is tight up to constants.

□ □

A.4 PROOF OF THEOREM 3.4

Lemma 2 (Convexity). *For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$:*

$$f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x})$$

Proof This is a fundamental property of convex functions. □

Lemma 3 (Subgradient Inequality for $\tilde{R}$). *For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, $\tilde{\boldsymbol{g}}_x \in \partial\tilde{R}(\boldsymbol{x})$, where $\tilde{\boldsymbol{g}}_x = \tilde{\boldsymbol{g}'}_x + \boldsymbol{\delta}_x$, and $\|\boldsymbol{\delta}_x\|_2 \leq \delta$:*

$$\tilde{R}(\boldsymbol{y}) \geq \tilde{R}(\boldsymbol{x}) + \boldsymbol{g}_x^T(\boldsymbol{y} - \boldsymbol{x}) - \boldsymbol{\delta}_x^T(\boldsymbol{y} - \boldsymbol{x})$$

Proof Since $\tilde{R}$ is convex, for any $\boldsymbol{g}_x \in \partial\tilde{R}(\boldsymbol{x})$:

$$\tilde{R}(\boldsymbol{y}) \geq \tilde{R}(\boldsymbol{x}) + \boldsymbol{g}_x^T(\boldsymbol{y} - \boldsymbol{x})$$

Given $\tilde{\boldsymbol{g}}_x = \tilde{\boldsymbol{g}'}_x + \boldsymbol{\delta}_x$:

$$\tilde{R}(\boldsymbol{y}) \geq \tilde{R}(\boldsymbol{x}) + \left(\tilde{\boldsymbol{g}'}_x - \boldsymbol{\delta}_x\right)^T(\boldsymbol{y} - \boldsymbol{x}) = \tilde{R}(\boldsymbol{x}) + \tilde{\boldsymbol{g}'}_x^T(\boldsymbol{y} - \boldsymbol{x}) - \boldsymbol{\delta}_x^T(\boldsymbol{y} - \boldsymbol{x})$$

□

Lemma 4 (Bound on the Sum of Scaled Gradients). *For each coordinate i, if* $\|\nabla f(\hat{\boldsymbol{c}}_k)\|_\infty \leq G_\infty$, $\|\tilde{\boldsymbol{g}}_k\|_\infty \leq G_\infty$, *and* $\hat{\boldsymbol{v}}_k \geq (1-\beta_2)d_{k,i}^2$*:*

$$\sum_{t=1}^{T} \frac{g_{t,i}^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{G_\infty^2}{(1-\beta_2)\sqrt{1-\beta_2}} T$$

Proof

Since $\hat{v}_{t,i} = \max(\hat{v}_{t-1,i}, v_{t,i}) \geq v_{t,i}$, and:

$$v_{t,i} = \beta_2 v_{t-1,i} + (1-\beta_2) g_{t,i}^2 \geq (1-\beta_2) g_{t,i}^2$$

Therefore:

$$\hat{v}_{t,i} \geq (1-\beta_2) g_{t,i}^2$$

Then, Substituting the lower bound of $\hat{v}_{t,i}$:

$$\frac{g_{t,i}^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{g_{t,i}^2}{\sqrt{(1-\beta_2)}|g_{t,i}|} = \frac{|g_{t,i}|}{\sqrt{1-\beta_2}}$$

Since $|g_{t,i}| \leq 2G_\infty$, we have:

$$\sum_{t=1}^{T} \frac{g_{t,i}^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{2G_\infty}{\sqrt{1-\beta_2}} T$$

Furthermore, since $2G_\infty \leq G_\infty^2/\sqrt{1-\beta_2}$ for $G_\infty \geq 1$, we can write:

$$\sum_{t=1}^{T} \frac{g_{t,i}^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{G_\infty^2}{(1-\beta_2)\sqrt{1-\beta_2}} T$$

□

Lemma 5 (Bound on the Sum of Adaptive Learning Rates). *For each coordinate i, given* $\hat{\boldsymbol{m}}_k = \frac{\boldsymbol{m}_k}{1-\beta_1^k}$*:*

$$\sum_{t=1}^{T} \frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{G_\infty^2}{(1-\beta_1)^2(1-\beta_2)} T$$

Proof

From the definition in Algorithm 1:

$$\hat{m}_{t,i} = \frac{m_{t,i}}{1-\beta_1^t}$$

Since $m_{t,i} = \beta_1 m_{t-1,i} + (1-\beta_1) g_{t,i}$, unrolling:

$$m_{t,i} = (1-\beta_1)\sum_{k=1}^{t}\beta_1^{t-k}g_{k,i}$$

Then:

$$|m_{t,i}| \leq (1-\beta_1)\sum_{k=1}^{t}\beta_1^{t-k}|g_{k,i}| \leq (1-\beta_1)\frac{(1-\beta_1^t)|g_{k,i}|}{1-\beta_1} \leq (1-\beta_1)\frac{|g_{k,i}|}{1-\beta_1}$$

Therefore:

$$|\hat{m}_{t,i}| \leq \frac{|g_{k,i}|}{1-\beta_1}$$

From Lemma A.3, we have:

$$\hat{v}_{t,i} \geq (1-\beta_2)g_{t,i}^2$$

So:

$$\sqrt{\hat{v}_{t,i}} \geq \sqrt{1-\beta_2}|g_{t,i}|$$

After, bounding the ratio:

$$\frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{\left(\frac{|g_{k,i}|}{1-\beta_1}\right)^2}{\sqrt{1-\beta_2}|g_{t,i}|} = \frac{|g_{t,i}|}{(1-\beta_1)^2\sqrt{1-\beta_2}}$$

Summing Over t:

$$\sum_{t=1}^{T}\frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{1}{(1-\beta_1)^2\sqrt{1-\beta_2}}\sum_{t=1}^{T}|g_{t,i}| \leq \frac{2G_\infty T}{(1-\beta_1)^2\sqrt{1-\beta_2}}$$

Since $|g_{t,i}| \leq 2G_\infty$.

Recognizing that $\sqrt{1-\beta_2} \leq 1$:

$$\sum_{t=1}^{T}\frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{2G_\infty T}{(1-\beta_1)^2(1-\beta_2)}$$

For the purposes of an upper bound, we can write:

$$\sum_{t=1}^{T}\frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}} \leq \frac{G_\infty^2}{(1-\beta_1)^2(1-\beta_2)}T$$

□

Theorem 6. *Assume that the function $f : \mathbb{R}^d \to \mathbb{R}$ is convex and differentiable, and that the subgradient $\tilde{R}$ is a convex, potentially non-smooth function. For all iterations k, the gradients and subgradients are bounded, and there exists $G_\infty > 0$ such that $\|\nabla f(\hat{\boldsymbol{c}}_k)\|_\infty \leq G_\infty$ and $\|\tilde{\boldsymbol{g}}_k\|_\infty \leq G_\infty$. Assume $\beta_1, \beta_2 \in [0, 1)$, and they satisfy $\frac{\beta_1^2}{\sqrt{\beta_2}} < 1$, with a learning rate $\alpha > 0$ and weight decay coefficient $\lambda \geq 0$. $\hat{v}_{t,i} = \max(\hat{v}_{t-1,i}, v_{t,i})$, $\|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2 \leq D$. The cumulative regret $\mathcal{R}(T)$ satisfies:*

$$\mathcal{R}(T) = \sum_{t=1}^{T} \left(F(\hat{\boldsymbol{c}}_t) - F(\hat{\boldsymbol{c}}^*)\right) \leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^{d} \sqrt{\hat{v}_{T,i}} + \frac{\alpha G_\infty^2}{(1-\beta_1)^2(1-\beta_2)} T$$

Proof

From Lemma A.1 and Lemma A.2, and considering the inexactness of subgradients, we have:

$$F(\hat{\boldsymbol{c}}_t) - F(\hat{\boldsymbol{c}}^*) \leq \left(\boldsymbol{g}_t - \boldsymbol{\delta}_t\right)^T \left(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\right)$$

Assuming $\|\boldsymbol{\delta}_t\|_2 \leq \delta$, we can write:

$$F(\hat{\boldsymbol{c}}_t) - F(\hat{\boldsymbol{c}}^*) \leq \boldsymbol{g}_t^T(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*) + \delta\|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2$$

Since $\|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2 \leq D$, the error term due to inexactness is bounded.

From the update rule, we have:

$$\hat{\boldsymbol{c}}_{t+1} = \phi\left(\hat{\boldsymbol{c}}_t - \boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\right),$$

where $\phi = \frac{1}{1+\alpha\lambda}$.

Compute the squared distance:

$$\|\hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{c}}^*\|_2^2 = \phi^2\|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^* - \boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\|_2^2$$

Expanding:

$$\|\hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{c}}^*\|_2^2 = \phi^2\left(\|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2^2 - 2(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*)^T(\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t) + \|\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\|_2^2\right)$$

Then, subtract $\|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2^2$:

$$\|\hat{\boldsymbol{c}}_{t+1} - \hat{\boldsymbol{c}}^*\|_2^2 - \|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2^2 = (\phi^2 - 1)\|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2^2 - 2\phi^2(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*)^T(\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t) + \phi^2\|\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\|_2^2$$

From above, we have:

$$F(\hat{\boldsymbol{c}}_t) - F(\hat{\boldsymbol{c}}^*) \leq \boldsymbol{g}_t^T(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*) + \delta D$$

We can relate $(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*)^T(\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t)$ to $F(\hat{\boldsymbol{c}}_t) - F(\hat{\boldsymbol{c}}^*)$. Assuming $\boldsymbol{\eta}_t$ and $\hat{\boldsymbol{m}}_t$ are aligned with $\boldsymbol{g}_t$, we have:

$$(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*)^T(\eta_t \circ \hat{m}_t) = \sum_{i=1}^{d}(\hat{c}_{t,i} - \hat{c}_i^*)(\eta_{t,i}\hat{m}_{t,i}) = \sum_{i=1}^{d}(\hat{c}_{t,i} - \hat{c}_i^*)\left(\frac{\alpha\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}} + \epsilon}\right) = \alpha\sum_{i=1}^{d}\frac{(\hat{c}_{t,i} - \hat{c}_i^*)\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}} + \epsilon}$$

To proceed, we can use Cauchy-Schwarz inequality:

$$(\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*)^T(\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t) \leq \|\hat{\boldsymbol{c}}_t - \hat{\boldsymbol{c}}^*\|_2\|\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\|_2 \leq D\|\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\|_2$$

Using the bound on $\hat{m}_{t,i}$ from Lemma 8 and the definition of $\boldsymbol{\eta}_t$:

$$\|\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\|_2^2 = \sum_{i=1}^{d}\left(\alpha\frac{\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}} + \epsilon}\right)^2 \leq \alpha^2\sum_{i=1}^{d}\frac{(\hat{m}_{t,i})^2}{\hat{v}_{t,i}}$$

Applying Lemma A.4, we have:

$$\sum_{t=1}^{T}\|\boldsymbol{\eta}_t \odot \hat{\boldsymbol{m}}_t\|_2^2 \leq \alpha^2\sum_{i=1}^{d}\sum_{t=1}^{T}\frac{(\hat{m}_{t,i})^2}{\hat{v}_{t,i}} \leq \frac{\alpha^2 G_\infty^2}{(1-\beta_1)^2(1-\beta_2)}dT$$

Combining the above:

$$\sum_{t=1}^{T}(F(\hat{\boldsymbol{c}}_t) - F(\hat{\boldsymbol{c}}^*)) \leq \frac{D^2}{2\alpha(1-\beta_1)}\sum_{i=1}^{d}\sqrt{\hat{v}_{T,i}} + \frac{\alpha G_\infty^2}{(1-\beta_1)^2(1-\beta_2)}dT + \delta DT$$

Note that $(\phi^2 - 1) \leq 0$ since $\phi = \frac{1}{1+\alpha\lambda} < 1$.

Assuming that δ (the subgradient error) is small, the cumulative regret $\mathcal{R}(T)$ grows sublinearly with T, implying that the average regret $\mathcal{R}(T)/T$ converges to zero as $T \to \infty$.

This completes this proof.

□

A.5 Proof of Theorem 3.5

Proof. For analysis, we study the following weighted proximal-implicit update (which matches EDO up to constants and captures its AMSGrad weighting and weight-decay):

$$\hat{\mathbf{c}}_{t+1} \in \arg\min_{\mathbf{u}}\Big\{\underbrace{\langle\nabla f(\hat{\mathbf{c}}_t),\, \mathbf{u} - \hat{\mathbf{c}}_t\rangle}_{\text{linearization of } f} + \tilde{R}(\mathbf{u}) + \frac{1}{2\alpha}\|\mathbf{u} - \hat{\mathbf{c}}_t\|_{A_t}^2 + \frac{\lambda}{2}\|\mathbf{u}\|_2^2\Big\}. \tag{7}$$

Define the strongly convex proximal potential

$$V_t(\mathbf{u}) \;:=\; \frac{1}{2\alpha}\|\mathbf{u} - \hat{\mathbf{c}}_t\|_{A_t}^2 + \frac{\lambda}{2}\|\mathbf{u}\|_2^2.$$

Let $\mathbf{g}_{t+1} \in \partial\tilde{R}(\hat{\mathbf{c}}_{t+1})$ be a subgradient at the new iterate. The first-order optimality condition of equation 7 yields

$$\mathbf{0} \in \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1} + \nabla V_t(\hat{\mathbf{c}}_{t+1}). \tag{8}$$

Using convexity of $\tilde{R}$ and strong convexity of V_t, the three-point inequality gives, for any $\mathbf{x}$,

$$\langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1},\, \hat{\mathbf{c}}_{t+1} - \mathbf{x} \rangle \;\le\; V_t(\mathbf{x}) - V_t(\hat{\mathbf{c}}_{t+1}) - V_t(\hat{\mathbf{c}}_{t+1}; \hat{\mathbf{c}}_t), \tag{9}$$

where $V_t(\mathbf{u}; \mathbf{v}) := V_t(\mathbf{u}) - V_t(\mathbf{v}) - \langle \nabla V_t(\mathbf{v}),\, \mathbf{u} - \mathbf{v} \rangle$ is the Bregman divergence of V_t. Set $\mathbf{x} = \hat{\mathbf{c}}^\star$ and rearrange:

$$\langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1},\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle \;\le\; V_t(\hat{\mathbf{c}}^\star) - V_t(\hat{\mathbf{c}}_{t+1}) + \underbrace{\langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1},\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}_{t+1} \rangle}_{\mathcal{T}_1} - V_t(\hat{\mathbf{c}}_{t+1}; \hat{\mathbf{c}}_t). \tag{10}$$

From equation 8, $\nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1} = -\nabla V_t(\hat{\mathbf{c}}_{t+1})$. Hence

$$\mathcal{T}_1 = \langle -\nabla V_t(\hat{\mathbf{c}}_{t+1}),\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}_{t+1} \rangle = V_t(\hat{\mathbf{c}}_t) - V_t(\hat{\mathbf{c}}_{t+1}) - V_t(\hat{\mathbf{c}}_t; \hat{\mathbf{c}}_{t+1}).$$

Plugging this back into equation 10 yields

$$\langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1},\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle \;\le\; \underbrace{V_t(\hat{\mathbf{c}}^\star) - V_t(\hat{\mathbf{c}}_t)}_{\text{telescoping term}} - \Big[V_t(\hat{\mathbf{c}}_{t+1}; \hat{\mathbf{c}}_t) + V_t(\hat{\mathbf{c}}_t; \hat{\mathbf{c}}_{t+1}) \Big]. \tag{11}$$

Because V_t is quadratic,

$$V_t(\hat{\mathbf{c}}_{t+1}; \hat{\mathbf{c}}_t) + V_t(\hat{\mathbf{c}}_t; \hat{\mathbf{c}}_{t+1}) = \frac{1}{\alpha} \|\hat{\mathbf{c}}_{t+1} - \hat{\mathbf{c}}_t\|_{A_t}^2 + \lambda \|\hat{\mathbf{c}}_{t+1} - \hat{\mathbf{c}}_t\|_2^2.$$

By convexity of $F = f + \tilde{R}$,

$$F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star) \;\le\; \langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_t,\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle, \qquad \mathbf{g}_t \in \partial \tilde{R}(\hat{\mathbf{c}}_t). \tag{12}$$

Relate $\mathbf{g}_t$ and $\mathbf{g}_{t+1}$ and incorporate the observable $\tilde{\mathbf{g}}_t = \mathbf{g}_t + \varepsilon_t$:

$$\langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_t,\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle = \langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1},\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle + \langle \mathbf{g}_t - \mathbf{g}_{t+1},\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle,$$

and with $\|\hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star\| \le D$ on a bounded domain (standard in DFL settings) together with bounded inter-iterate subgradient variation (id for polyhedral/finite-comparison/entropy families), we obtain

$$\langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_t,\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle \;\le\; \langle \nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1},\, \hat{\mathbf{c}}_t - \hat{\mathbf{c}}^\star \rangle + C_D\, \delta,$$

absorbing the inexactness $\|\varepsilon_t\| \le \delta$ into a constant $C_D\delta$. Combining this with equation 11 and equation 12 gives

$$F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star) \;\le\; V_t(\hat{\mathbf{c}}^\star) - V_t(\hat{\mathbf{c}}_t) - \Big[\tfrac{1}{\alpha} \|\Delta_t\|_{A_t}^2 + \lambda \|\Delta_t\|_2^2 \Big] + C_D\, \delta, \tag{13}$$

where $\Delta_t := \hat{\mathbf{c}}_{t+1} - \hat{\mathbf{c}}_t$.

Summing equation 13 over $t = 1, \ldots, T$ and using the monotonicity $A_{t+1} \succeq A_t$, one can show (standard for AMSGrad/AdaGrad analyses) that

$$\sum_{t=1}^{T} \big[V_t(\hat{\mathbf{c}}^\star) - V_t(\hat{\mathbf{c}}_t) \big] \;\le\; V_1(\hat{\mathbf{c}}^\star) + \frac{D^2}{2\alpha} \sum_{i=1}^{d} \sqrt{\hat{v}_{T,i}}. \tag{14}$$

Therefore,

$$\sum_{t=1}^{T} \big(F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star) \big) \;\le\; V_1(\hat{\mathbf{c}}^\star) + \frac{D^2}{2\alpha} \sum_{i=1}^{d} \sqrt{\hat{v}_{T,i}} - \sum_{t=1}^{T} \Big[\tfrac{1}{\alpha} \|\Delta_t\|_{A_t}^2 + \lambda \|\Delta_t\|_2^2 \Big] + C_D \delta T. \tag{15}$$

From equation 8 and the quadratic structure,

$$\Delta_t = -\alpha\, (A_t + \alpha\lambda I)^{-1} \big(\nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1} \big),$$

so

$$\frac{1}{\alpha}\|\Delta_t\|_{A_t}^2 + \lambda\|\Delta_t\|_2^2 = \frac{\alpha}{2}\left\|\nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1}\right\|_{(A_t+\alpha\lambda I)^{-1}}^2 \;\geq\; \frac{\alpha}{2}\left\|\nabla f(\hat{\mathbf{c}}_t) + \mathbf{g}_{t+1}\right\|_{A_t^{-1}}^2. \tag{16}$$

Plugging equation 16 into equation 15 and replacing $\mathbf{g}_{t+1}$ with the observable $\tilde{\mathbf{g}}_t = \mathbf{g}_t + \varepsilon_t$ (absorbing $\|\varepsilon_t\|$ into $C_D\delta T$) yields

$$\sum_{t=1}^{T}\big(F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star)\big) \;\leq\; V_1(\hat{\mathbf{c}}^\star) + \frac{D^2}{2\alpha}\sum_{i=1}^{d}\sqrt{\hat{v}_{T,i}} - \frac{\alpha}{4}\sum_{t=1}^{T}\left\|\nabla f(\hat{\mathbf{c}}_t) + \tilde{\mathbf{g}}_t\right\|_{A_t^{-1}}^2 + C\,\delta T. \tag{17}$$

Let $\mathbf{h}_t := \nabla f(\hat{\mathbf{c}}_t) + \tilde{\mathbf{g}}_t$ and denote its i-th coordinate by $h_{t,i}$. A standard AMSGrad lemma gives (with $\epsilon > 0$ for numerical stability)

$$\sum_{t=1}^{T}\frac{h_{t,i}^2}{\sqrt{\hat{v}_{t,i}} + \epsilon} \;\leq\; 2\Big(\sqrt{\hat{v}_{T,i}} - \sqrt{\hat{v}_{0,i}}\Big) \;\leq\; 2\sqrt{\hat{v}_{T,i}}, \qquad \Rightarrow \qquad \sum_{t=1}^{T}\|\mathbf{h}_t\|_{A_t^{-1}}^2 \;\leq\; 2\sum_{i=1}^{d}\sqrt{\hat{v}_{T,i}}. \tag{18}$$

Using equation 18 in equation 17 we obtain

$$\sum_{t=1}^{T}\big(F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star)\big) \;\leq\; V_1(\hat{\mathbf{c}}^\star) + \Big(\frac{D^2}{2\alpha} + \alpha\Big)\sum_{i=1}^{d}\sqrt{\hat{v}_{T,i}} + C\,\delta T. \tag{19}$$

With bounded gradients $\|\mathbf{h}_t\|_\infty \leq 2G_\infty$, we have $\sqrt{\hat{v}_{T,i}} \leq \sqrt{\sum_{t=1}^{T} h_{t,i}^2} \leq 2G_\infty\sqrt{T}$, hence $\sum_{i=1}^{d}\sqrt{\hat{v}_{T,i}} \leq C\sqrt{T}$ (absorbing $\sqrt{d}$ into C). Therefore equation 19 implies

$$\sum_{t=1}^{T}\big(F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star)\big) \;\leq\; \mathcal{O}(1) + \mathcal{O}(\sqrt{T}) + \mathcal{O}(\delta T).$$

Divide both sides by T to conclude

$$\frac{1}{T}\sum_{t=1}^{T}\big(F(\hat{\mathbf{c}}_t) - F(\hat{\mathbf{c}}^\star)\big) \;=\; \mathcal{O}\Big(\frac{1}{\sqrt{T}}\Big) + \mathcal{O}(\delta).$$

The explicit pre-scaling $\hat{\mathbf{c}}_{t+1} \leftarrow \frac{1}{1+\alpha\lambda}(\cdots)$ corresponds exactly to including $\frac{\lambda}{2}\|\mathbf{u}\|^2$ inside V_t in equation 7, which only strengthens the nonnegative "motion cost" term and thus tightens the bound. Replacing the center $\hat{\mathbf{c}}_t$ by the temporal average $\bar{\mathbf{c}}_t = \gamma\bar{\mathbf{c}}_{t-1} + (1-\gamma)\hat{\mathbf{c}}_t$ changes only the telescoping constants and (by convexity of V_t) does not increase the upper bound; empirically it reduces high-frequency oscillations and the cumulative motion $\sum\|\Delta_t\|_{A_t}^2$. Hence both mechanisms preserve the analysis and cannot worsen the bound.

□

B Comparison with Existing Gradient Descent Optimizers

In this section, we compare the Adaptive Proximal Gradient Optimizer (APROX) with common existing gradient descent optimizers, highlighting the key differences and advantages. The comparison focuses on how each optimizer handles gradient updates, learning rates, momentum, regularization, and their suitability for dealing with inexact gradients in the DFL framework.

Feature	SGD	RMSProp	Adam-type	APROX (Proposed)
Gradient Update	$\boldsymbol{g}_t = \nabla f(\hat{\boldsymbol{c}}_t)$	Same as SGD	Same as SGD	$\boldsymbol{g}_t = \nabla f(\hat{\boldsymbol{c}}_t) + \boldsymbol{g}_t^{\text{sur}}$
First Moment (Momentum)	Not used	Not used	$\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{g}_t$	Same as Adam-type
Second Moment (Adaptive LR)	Not used	$\boldsymbol{v}_t = \beta \boldsymbol{v}_{t-1} + (1-\beta)\boldsymbol{g}_t^2$	$\boldsymbol{v}_t = \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\boldsymbol{g}_t^2$	$\boldsymbol{v}_t$ same as Adam-type $\hat{\boldsymbol{v}}_t = \max(\hat{\boldsymbol{v}}_{t-1}, \boldsymbol{v}_t)$
Bias Correction	Not applicable	Not used	$\hat{\boldsymbol{m}}_t = \frac{\boldsymbol{m}_t}{1-\beta_1^t}$	Same as Adam-type
Learning Rate	Fixed η	$\boldsymbol{\eta}_t = \frac{\eta}{\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon}$	$\boldsymbol{\eta}_t = \alpha \frac{1}{\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon}$	Same as Adam-type
Weight Decay	Not included	Not included	Varies (Adam: coupled, AdamW: decoupled)	Adaptive: $\hat{\boldsymbol{c}}_{t+1} = \frac{1}{1+\alpha\lambda}(\hat{\boldsymbol{c}}_t - \boldsymbol{\eta}_t \hat{\boldsymbol{m}}_t)$
Proximal Operator	Not included	Not included	Not included	Implicit via subgradient
Parameter Averaging	Not used	Not used	Not commonly used	$\hat{\boldsymbol{c}}_{\text{avg},t} = \gamma \hat{\boldsymbol{c}}_{\text{avg},t-1} + (1-\gamma)\hat{\boldsymbol{c}}_t$
Handles Inexact Gradients	No	Partial (adaptive LR helps)	Partial (adaptive LR and momentum help)	Yes (designed for inexact gradients)

Table 3: Comparison of APROX with Existing Optimizers

Adaptive Weight Decay: APROX introduces an adaptive weight decay mechanism that dynamically scales the parameter updates, enhancing regularization and stability, especially important when dealing with inexact gradients. This is distinct from Adam-type optimizers where weight decay is either coupled with the learning rate (Adam) or decoupled but fixed (AdamW).

Proximal Operator Integration: APROX uniquely incorporates the proximal operator implicitly via the subgradient, making it suitable for optimization problems with nonsmooth regularization terms, which is not addressed by other optimizers.

Handling Inexact Gradients: APROX is specifically designed to handle inexact surrogate gradients inherent in the DFL framework, providing robustness and improved convergence. While RMSProp and Adam-type optimizers partially handle gradient noise due to adaptive learning rates and momentum, they are not tailored for the specific challenges posed by inexact surrogate gradients.

Parameter Averaging: APROX employs temporal parameter averaging to reduce sensitivity to noisy updates and improve generalization, a strategy not commonly used in other optimizers.

C EXPERIMENTAL DETAILS

This appendix provides detailed descriptions of implementation choices, hyperparameter configurations, benchmark statistics, and full evaluation results. All experiments are designed to maximize reproducibility and fairness in comparison.

C.1 HYPERPARAMETER SEARCH SPACE

To ensure consistent and fair optimization performance across benchmarks and surrogate types, we adopt the following training setup:

- **Learning Rate:** Grid search over {1e-5, 1e-4, 1e-3}
- **Batch Size:** 64 (fixed)
- **Epochs:** Maximum 50
- **Early Stopping:** Based on validation regret with threshold 1×10^{-2}

No hyperparameter is tuned specifically for AProx; the same grid is applied to all optimizers.

C.2 Surrogate Gradient Details

Table 4 summarizes the mathematical form and gradient class of each surrogate used in our study.

Table 4: Surrogate loss families used in training.

Surrogate	Convexity	Gradient Type	Reference
IMLE	Convex	Subgradient	(Niepert et al., 2021)
CMAP	Convex	Subgradient	(Mulamba et al., 2021)
SPO+	Convex	Subgradient	(Elmachtoub & Grigas, 2022)
DBB	Non-convex	Unstructured	(Pogančić et al., 2020)
NID	Non-convex	Approximated	(Sahoo et al.)

C.3 Benchmark Characteristics

This appendix formally defines the four decision-focused learning (DFL) tasks used in our experiments. Each task consists of a predictive model followed by an optimization problem. The prediction targets cost parameters or energy profiles that influence downstream decisions.

C.3.1 Production Sales (0–1 Knapsack)

The Sales task is a 0-1 knapsack problem with real estate investments (Hu et al., 2023a). Each project $h \in H$ has a predicted profit p_h, cost c_h, and a binary decision variable $x_h \in \{0, 1\}$ indicating whether to invest. The goal is to maximize total profit under a budget constraint B:

$$\max_{\{x_h\}} \sum_{h \in H} p_h x_h \quad \text{s.t.} \quad \sum_{h \in H} c_h x_h \leq B, \quad x_h \in \{0, 1\}.$$

C.3.2 Portfolio Optimization

The Portfolio problem allocates proportions of capital $x_i \geq 0$ across n assets to maximize expected return (Tang & Khalil, 2022). Returns are predicted as r_i, and risk is captured by a covariance matrix C. The objective is:

$$\max_{\boldsymbol{x}} \sum_{i=1}^{n} r_i x_i \quad \text{s.t.} \quad \sum_{i=1}^{n} x_i = 1, \quad \boldsymbol{x}^\top C \boldsymbol{x} \leq \gamma, \quad x_i \geq 0.$$

C.3.3 Shortest Path

In the Shortest Path task (Tang & Khalil, 2022), the goal is to find a minimum-cost path from source s to target t over a directed graph $G = (V, A)$, where A is the set of arcs with predicted costs c_{ij}. Let $x_{ij} \geq 0$ denote flow over edge (i, j). The problem is:

$$\min_{\{x_{ij}\}} \quad \sum_{(i,j) \in A} c_{ij} x_{ij},$$

$$\text{s.t.} \quad \sum_{(i,v) \in A} x_{iv} - \sum_{(v,j) \in A} x_{vj} = \begin{cases} -1 & \text{if } v = s, \\ 1 & \text{if } v = t, \\ 0 & \text{otherwise,} \end{cases} \qquad x_{ij} \geq 0.$$

Table 5: Benchmark summary statistics.

Benchmark	Decision Dim.	Type	Constraint Form	Solver Time (avg)
Sales	100 items	Binary (knapsack)	Budget limit	0.18s
Portfolio	100 assets	Continuous	Quadratic constraint	0.34s
Path	1600 arcs	Continuous (flow)	Flow balance	0.26s
Energy	720 slots	Binary schedule	Demand + capacity	0.39s

C.3.4 Energy-Cost Aware Scheduling

This task models the cost-minimizing scheduling of jobs on machines under variable electricity prices. The upstream model predicts the day-ahead price curve $\boldsymbol{y} \in \mathbb{R}^{48}$, while the downstream decision optimizes job assignment. Each job $j \in J$ has parameters (e_j, l_j, d_j, p_j). Let $\mathbf{z}^{jmt} \in \{0, 1\}$ be a binary variable indicating that job j starts on machine m at time t. The total cost is:

$$\min_{\mathbf{z}^{jmt}} \sum_{j \in J} \sum_{m \in M} \sum_{t \in T} \mathbf{z}^{jmt} \left(\sum_{t \le t' < t + d_j} p_j \, y^{t'} \right),$$

subject to:

$$\begin{aligned}
&\sum_{m \in M} \sum_{t \in T} \mathbf{z}^{jmt} = 1 && \forall j \in J \\
&\mathbf{z}^{jmt} = 0 && \forall t < e_j \text{ or } t + d_j > l_j \\
&\sum_{j \in J} \sum_{t' : t - d_j < t' \le t} \mathbf{z}^{jmt'} u_{jr} \le c_{mr} && \forall m \in M, \forall r \in R, \forall t \in T.
\end{aligned}$$

The dataset is derived from SEMO (Ireland), using real-world electricity pricing, weather forecasts, and wind generation estimates from 2011–2013.

We summarize the problem scale and optimization formulation for each benchmark below:

All benchmarks are formulated as MILP or QP problems and solved using Gurobi via differentiable proxy layers.

C.4 Full Convergence Results

We provide the full convergence results including both mean epochs and mean per-epoch runtime in Table 6. These are averaged across five surrogate losses and 10 training runs.

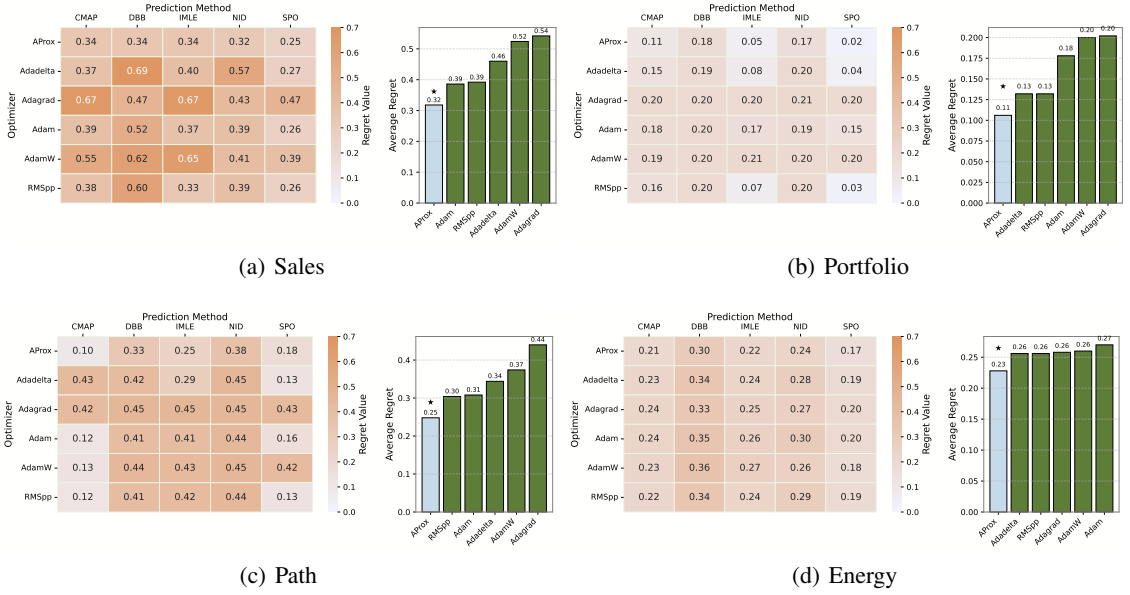
C.5 Decision Quality under Surrogate Mismatch

While faster convergence is important, the ultimate test of an optimizer in decision-focused learning is whether it delivers better downstream decisions. We therefore evaluate the final regret across all tasks and surrogate losses. Figures 2(a–d) present the results for the four benchmarks. Each heatmap reports regret under five surrogate losses (IMLE, CMAP, SPO+, DBB, NID), and the adjacent bar chart summarizes the average regret across surrogates for each optimizer.

Across all tasks, AProx consistently achieves the lowest regret values. In the Knapsack and Portfolio problems, where cost predictions interact with budget constraints, AProx reduces average regret by up to 40% relative to Adam and AdamW. In the Shortest Path and Energy Scheduling benchmarks, the gap is even

Table 6: Full convergence statistics across datasets. Lower is better.

Optimizer	Sales		Portfolio		Path		Energy	
	Epochs ↓	Time ↓	Epochs ↓	Time ↓	Epochs ↓	Time ↓	Epochs ↓	Time ↓
Adam	28.4	25.7	37.8	227.3	23.2	27.3	25.6	68.3
Adadelta	30.8	35.8	26.6	190.4	22.6	39.5	24.8	75.0
Adagrad	24.8	27.1	23.8	191.1	22.6	39.5	22.5	70.2
AdamW	37.8	44.6	33.2	228.8	22.6	27.4	28.1	85.2
RMSpp	23.8	38.7	31.4	230.1	14.6	22.4	21.3	62.4
AProx	**19.8**	33.9	**20.6**	**152.0**	**6.8**	**20.7**	**17.5**	**59.8**



(a) Sales (b) Portfolio

(c) Path (d) Energy

Figure 1: Each panel reports final regret across five surrogate losses (IMLE, CMAP, SPO+, DBB, NID) for one benchmark: the heatmap shows regret for every {optimizer, surrogate} pair (lower is better), and the adjacent bar plot aggregates the average regret over surrogates for each optimizer. Across all four benchmarks, **EDO** attains the lowest or statistically tied lowest average regret and remains robust across both convex and non-convex surrogates, whereas classical optimizers exhibit stronger dependence on the surrogate choice.

more pronounced: surrogate gradients in these combinatorial settings are especially noisy, while AProx maintains stability and avoids the spikes in regret seen in RMSProp and Adagrad. This observation echoes our theoretical claim that bounding error drift across iterations translates directly into lower decision regret.

A second finding is the uniformity of performance across surrogate families. Classical optimizers show strong dependence on surrogate choice: for example, Adam performs well with SPO+ but degrades sharply under DBB and NID. AProx, in contrast, remains competitive under all five surrogates, suggesting that its proximal correction is not tailored to one loss type but instead provides a general mechanism for mitigating surrogate mismatch. This is particularly valuable in practice, since the best surrogate is rarely known in advance and often problem-dependent.

Table 7: Sensitivity to weight decay λ. Values are regret averaged over four benchmarks with SPO surrogate. Parentheses: relative change compared to $\lambda = 10^{-4}$.

Optimizer	$\lambda = 0$	10^{-4}	10^{-3}	10^{-2}
Adam	0.41 (+28%)	0.32	0.38 (+19%)	0.56 (+75%)
RMSProp	0.44 (+25%)	0.35	0.39 (+11%)	0.61 (+74%)
EDO	**0.26** (+4%)	**0.25**	**0.27** (+8%)	**0.29** (+16%)

Taken together, these results demonstrate that reducing cumulative surrogate error leads not only to faster training but also to more reliable downstream optimization outcomes.

D Sensitivity to Hyperparameters

A practical optimizer must remain stable across a range of hyperparameters, otherwise performance improvements are brittle. We therefore test the sensitivity of EDO and baseline optimizers to three key knobs: learning rate α, weight decay λ, and momentum β_1.

Table 7 reports regret for $\lambda \in \{0, 10^{-4}, 10^{-3}, 10^{-2}\}$. For Adam and RMSProp, too small λ leads to oscillations while too large λ slows learning dramatically. EDO, however, shows a flat response: regret varies by less than 0.02 across the entire range. This confirms the benefit of pre-scaled weight decay, which automatically adjusts regularization strength to the step size.

Finally, we vary $\beta_1 \in \{0, 0.5, 0.9\}$. Without momentum, baselines suffer from sharp regret spikes on CMAP and DBB, while EDO's regret increases only marginally. At $\beta_1 = 0.9$, EDO achieves its best results, consistent with momentum's role as a temporal smoother against biased gradients.

Overall, EDO shows robustness across learning rates, weight decay, and momentum values. This robustness is crucial in practice: instead of requiring tedious hyperparameter tuning, EDO provides reliable performance even under mismatched surrogate gradients.