LPGD: A General Framework for Backpropagation through Embedded Optimization Layers

Anselm Paulus [1 2] **Georg Martius** [1 2] **Vít Musil** [3]

Abstract

Embedding parameterized optimization problems as layers into machine learning architectures serves as a powerful inductive bias. Training such architectures with stochastic gradient descent requires care, as degenerate derivatives of the embedded optimization problem often render the gradients uninformative. We propose *Lagrangian Proximal Gradient Descent* (LPGD) a flexible framework for training architectures with embedded optimization layers that seamlessly integrates into automatic differentiation libraries. LPGD efficiently computes meaningful replacements of the degenerate optimization layer derivatives by re-running the forward solver oracle on a perturbed input. LPGD captures various previously proposed methods as special cases, while fostering deep links to traditional optimization methods. We theoretically analyze our method and demonstrate on historical and synthetic data that LPGD converges faster than gradient descent even in a differentiable setup.

1. Introduction

Optimization at inference is inherent to many prediction tasks, including autonomous driving (Paden et al., 2016), modeling physical systems (Cranmer et al., 2020), or robotic control (Kumar et al., 2016). Therefore, embedding optimization algorithms as building blocks of machine learning models serves as a powerful inductive bias. A recent trend has been to embed parameterized constrained optimization problems that can efficiently be solved to optimality (Amos & Kolter, 2017a; Agrawal et al., 2019a;b; Vlastelica et al., 2020; Sun et al., 2023; Sahoo et al., 2023).

Training such a *parameterized* optimization model is an instance of bi-level optimization (Gould et al., 2016), which is generally challenging. Whenever it is possible to propagate gradients through the optimization problem via an informative derivative of the solution mapping, the task is typically approached with standard stochastic gradient descent (GD) (Amos & Kolter, 2017a; Agrawal et al., 2019b). However, when the optimization problem has discrete solutions, the derivatives are typically degenerate, as small perturbations of the input do not affect the optimal solution. Previous works have proposed several methods to overcome this challenge, ranging from differentiable relaxations (Wang et al., 2019; Wilder et al., 2019a; Mandi & Guns, 2020; Djolonga & Krause, 2017) and stochastic smoothing (Berthet et al., 2020; Dalle et al., 2022), over proxy losses (Paulus et al., 2021), to finite-difference based techniques (Vlastelica et al., 2020).

The main contribution of this work is the unification of a variety of previous methods (McAllester et al., 2010; Vlastelica et al., 2020; Domke, 2010; Sahoo et al., 2023; Elmachtoub & Grigas, 2022; Blondel et al., 2020) into a general framework called *Lagrangian Proximal Gradient Descent* (LPGD). Motivated by traditional proximal optimization techniques (Moreau, 1962; Rockafellar, 1970; Nesterov, 1983; Figueiredo et al., 2007; Tseng, 2008; Beck & Teboulle, 2009; Combettes & Pesquet, 2011; Bauschke & Combettes, 2011; Nesterov, 2014; Parikh & Boyd, 2014), we derive LPGD as gradient descent on a smoothed envelope of a loss linearization. This fosters deep links between traditional and contemporary methods. We provide theoretical insights into the asymptotic behavior of our method, capturing a trade-off between smoothness and tightness of the introduced envelope.

We identify multiple practical use-cases of LPGD. On the one hand, when non-degenerate derivatives of the solution mapping exist, they can be computed as the limit of the LPGD update, providing a fast and simple alternative to previous methods based on differentiating the optimality conditions (Amos & Kolter, 2017a; Agrawal et al., 2019b; Wilder et al., 2019a; Mandi & Guns, 2020). On the other hand, when the derivatives are degenerate and GD fails,

[1]University of Tübingen, Tübingen, Germany [2]Max Planck Institute for Intelligent Systems, Tübingen, Germany [3]Masaryk University, Brno, Czech Republic. Correspondence to: Anselm Paulus <apaulus@tue.mpg.de>, Georg Martius <gmartius@uni-tuebingen.de>, Vít Musil <musil@fi.muni.cz>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

LPGD still allows learning the optimization parameters. This generalizes Vlastelica et al. (2020) to non-linear objectives, saddle-point problems, and learnable constraint parameters. Finally, we explore a new experimental direction by demonstrating on synthetic and historical data that LPGD can result in faster convergence than GD even when non-degenerate derivatives of the solution mapping exist.

2. Related work

Numerous implicit layers have been proposed in recent years, including neural ODEs (Chen et al., 2018; Dupont et al., 2019) and root-solving-based layers (Bai et al., 2019; 2020; Gu et al., 2020; Winston & Kolter, 2020; Fung et al., 2021; Ghaoui et al., 2021; Geng et al., 2021). In this work, we focus on optimization-based layers. A lot of research has been done on obtaining the gradient of such a layer, either by using the implicit function theorem to differentiate quadratic programs (Amos & Kolter, 2017a), conic programs (Agrawal et al., 2019b), ADMM (Sun et al., 2023), dynamic time warping (Xu et al., 2023), or by finite-differences (Domke, 2010; McAllester et al., 2010; Song et al., 2016; Lorberbom et al., 2019).

Another direction of related work has investigated optimization problems with degenerate derivatives of the solution mapping. The techniques developed for training these models range from continuous relaxations of SAT problems (Wang et al., 2019) and submodular optimization (Djolonga & Krause, 2017), over regularization of linear programs (Amos et al., 2019; Wilder et al., 2019a; Mandi & Guns, 2020; Paulus et al., 2020) to stochastic smoothing (Berthet et al., 2020; Dalle et al., 2022), learnable proxies (Wilder et al., 2019b) and generalized straight-through-estimators (Jang et al., 2017; Sahoo et al., 2023). Other works have built on geometric proxy losses (Paulus et al., 2021) and, again, finite-differences (Vlastelica et al., 2020; Niepert et al., 2021; Minervini et al., 2023).

Finally, a special case of an optimization layer is to embed an optimization algorithm as the final component of the prediction pipeline. This encompasses energy-based models (LeCun & Huang, 2005; Blondel et al., 2022), structured prediction (McAllester et al., 2010; Blondel, 2019; Blondel et al., 2020), smart predict-then-optimize (Ferber et al., 2020; Elmachtoub & Grigas, 2022) and symbolic methods such as SMT solvers (Fredrikson et al., 2023). Additional details of the closest related methods are in Appendix B.

3. Problem Setup

We consider a parameterized embedded constrained optimization problem of the form

$$\mathcal{L}^*(w) := \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathcal{L}(x, y, w) \tag{1}$$

where $w \in \mathbb{R}^k$ are the parameters, $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^m$ are the primal and dual feasible set, and $\mathcal{L} \in \mathcal{C}^1$ is a *Lagrangian*. The corresponding optimal solution is

$$z^*(w) = (x^*, y^*)(w) := \arg\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathcal{L}(x, y, w). \tag{2}$$

We assume strong duality holds for (1). For instance, this setup covers conic programs and quadratic programs, see Appendix C for details. Note, that the solution of (2) is in general set-valued. We assume that the solution set is non-empty and has a selection $z^*(w)$ continuous at w.[1] Throughout the paper, we assume access to an oracle that efficiently solves (2) to high accuracy. In our experiments, (2) is a conic program that we efficiently solve to high accuracy using the SCS solver (O'Donoghue et al., 2016).[2]

Our aim is to embed optimization problem (2) into a larger prediction pipeline. Given an input $\mu \in \mathbb{R}^p$ (e.g. an image), the parameters of the embedded optimization problem w are predicted by a parameterized backbone model $W_\theta \colon \mathbb{R}^p \to \mathbb{R}^k$ (e.g. a neural network with weights $\theta \in \mathbb{R}^r$) as $w = W_\theta(\mu)$. The embedded optimization problem (2) is then solved on the predicted parameters w returning the predicted solution $x^*(w)$, and its quality is measured by a loss function $\ell \colon \mathbb{R}^n \to \mathbb{R}$. The backbone and the loss function are assumed to be continuously differentiable.

Our goal is to train the prediction pipeline by minimizing the loss on a dataset of inputs $\{\mu_i\}_{i=1}^N$

$$\min_{\theta\in\mathbb{R}^r} \textstyle\sum_{i=1}^N \ell\big(x^*(W_\theta(\mu_i))\big) \tag{3}$$

using gradient backpropagation as in stochastic gradient descent or variations thereof (Kingma & Ba, 2015). However, the solution mapping does not need to be differentiable, and even when it is, the derivatives are often degenerate (e.g. they can be zero almost everywhere).[3] Therefore, we aim to derive informative replacements for the gradient $\nabla_w \ell(x^*(w))$, which can then be further backpropagated to the weights θ by standard automatic differentiation libraries (Abadi et al., 2015; Bradbury et al., 2018; Paszke et al., 2019). Note that the loss could also be composed of further learnable components that might be trained simultaneously. A list of symbols is provided in Appendix H.

[1]These assumptions e.g. follow from compactness of $\mathcal{X}$ and $\mathcal{Y}$ and the existence of a unique solution, given the continuity of $\mathcal{L}$.

[2]We use CVXPY (Diamond & Boyd, 2016; Agrawal et al., 2019a) for automatic reduction of parameterized convex optimization problems to conic programs in a differentiable way.

[3]Our method only assumes continuity of the solution mapping which is weaker than differentiability. Therefore, whenever the true gradients exist, the continuity assumption is also fulfilled.

4. Backgound: Proximal Point Method & Proximal Gradient Descent

The *Moreau envelope* (Moreau, 1962) $\mathrm{env}_{\tau f}\colon \mathbb{R}^n \to \mathbb{R}$ of a proper, lower semi-continuous, possibly non-smooth function $f\colon \mathbb{R}^n \to \mathbb{R}$ is defined for $\tau > 0$ as

$$\mathrm{env}_{\tau f}(\widehat{x}) := \inf_x f(x) + \tfrac{1}{2}\|x - \widehat{x}\|_2^2. \tag{4}$$

The envelope $\mathrm{env}_{\tau f}$ is a smoothed lower bound approximation of f (Rockafellar & Wets, 1998, Theorem 1.25). The corresponding *proximal map* $\mathrm{prox}_{\tau f}\colon \mathbb{R}^n \to \mathbb{R}^n$ is given by

$$\begin{aligned}\mathrm{prox}_{\tau f}(\widehat{x}) &:= \arg\min_x f(x) + \tfrac{1}{2\tau}\|x - \widehat{x}\|_2^2 \\ &= \widehat{x} - \tau\nabla\,\mathrm{env}_{\tau f}(\widehat{x})\end{aligned} \tag{5}$$

and can be interpreted as a gradient descent step on the Moreau envelope with step-size τ. For a more detailed discussion of the connection between proximal map and Moreau envelope see e.g. Rockafellar & Wets (1998); Bauschke & Combettes (2011); Parikh & Boyd (2014).

The *proximal point method* (Rockafellar, 1976; Güler, 1992; Bauschke & Combettes, 2011; Parikh & Boyd, 2014) aims to minimize f by iteratively updating $\widehat{x} \mapsto \mathrm{prox}_{\tau f}(\widehat{x})$. Now, assume that f decomposes as $f = g + h$, with g differentiable and h potentially non-smooth, and consider a linearization of g around $\widehat{x}$ given by

$$\widetilde{g}(x) := g(\widehat{x}) + \langle x - \widehat{x}, \nabla g(\widehat{x})\rangle. \tag{6}$$

The corresponding proximal map reads as

$$\begin{aligned}\mathrm{prox}_{\tau(\widetilde{g}+h)}(\widehat{x}) &= \arg\min_x \widetilde{g}(x) + h(x) + \tfrac{1}{2\tau}\|x - \widehat{x}\|_2^2 \\ &= \mathrm{prox}_{\tau h}\big(\widehat{x} - \tau\nabla g(\widehat{x})\big)\end{aligned} \tag{7}$$

and iterating $\widehat{x} \mapsto \mathrm{prox}_{\tau h}\big(\widehat{x} - \tau\nabla g(\widehat{x})\big)$ is called *proximal gradient descent* (Nesterov, 1983; Combettes & Pesquet, 2011; Parikh & Boyd, 2014).

5. Method

Our goal is to translate the idea of proximal methods to parameterized optimization models as in Section 3, by defining a *Lagrange-Moreau envelope* of the loss $w \mapsto \ell(x^*(w))$ on which we can perform gradient descent. In analogy to (4), given w and corresponding optimal solution x^*, the envelope should select an x in the proximity of x^* with a lower loss ℓ. The key concept is to replace as a measure of proximity the Euclidean distance with a *Lagrangian divergence* indicating how close x is to optimality given w.

5.1. Lagrangian Divergence

First we define the *Lagrangian difference* for $x \in \mathcal{X}$ and $w \in \mathbb{R}^k$ as

$$D_{\mathcal{L}}(x, y|w) := \mathcal{L}(x, y, w) - \mathcal{L}^*(w), \tag{8}$$

where $\mathcal{L}^*(w)$ is the optimal Lagrangian (1). We then define the *Lagrangian divergence*[4] $D^*_{\mathcal{L}}(\cdot|w)\colon \mathcal{X} \to \mathbb{R}^+$ as

$$\begin{aligned}D^*_{\mathcal{L}}(x|w) &:= \sup_{y\in\mathcal{Y}} D_{\mathcal{L}}(x, y|w) \\ &= \sup_{y\in\mathcal{Y}}\big[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)\big] \geq 0\end{aligned} \tag{9}$$

for $w \in \mathbb{R}^k$, where the last inequality follows from

$$\sup_{y\in\mathcal{Y}} \mathcal{L}(x, y, w) \geq \min_{\widetilde{x}\in\mathcal{X}}\max_{y\in\mathcal{Y}} \mathcal{L}(\widetilde{x}, y, w) = \mathcal{L}^*(w). \tag{10}$$

The divergence has the key property

$$D^*_{\mathcal{L}}(x|w) = 0 \quad\Leftrightarrow\quad x \in \mathcal{X} \text{ minimizes (2)} \tag{11}$$

which makes it a reasonable measure of optimality of x given w, for the proof, see Appendix F.

5.2. Lagrange-Moreau Envelope

Given $\tau > 0$, we say that $\ell_\tau\colon \mathbb{R}^k \to \mathbb{R}$ is the *lower Lagrange-Moreau envelope* ($\mathcal{L}$-envelope) if

$$\begin{aligned}\ell_\tau(w) &:= \min_{x\in\mathcal{X}} \ell(x) + \tfrac{1}{\tau}D^*_{\mathcal{L}}(x|w) \\ &= \min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} \ell(x) + \tfrac{1}{\tau}D_{\mathcal{L}}(x, y|w).\end{aligned} \tag{12}$$

The corresponding *lower Lagrangian proximal map* $z_\tau\colon \mathbb{R}^k \to \mathbb{R}^{n+m}$ is defined as

$$\begin{aligned}z_\tau(w) &:= \arg\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} \ell(x) + \tfrac{1}{\tau}D_{\mathcal{L}}(x, y|w) \\ &= \arg\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} \mathcal{L}(x, y, w) + \tau\ell(x).\end{aligned} \tag{13}$$

The *upper $\mathcal{L}$-envelope* $\ell^\tau\colon \mathbb{R}^k \to \mathbb{R}$ is defined analogously with maximization instead of minimization as

$$\begin{aligned}\ell^\tau(w) &:= \max_{x\in\mathcal{X}} \ell(x) - \tfrac{1}{\tau}D^*_{\mathcal{L}}(x|w) \\ &= \max_{x\in\mathcal{X}}\min_{y\in\mathcal{Y}} \ell(x) - \tfrac{1}{\tau}D_{\mathcal{L}}(x, y|w),\end{aligned} \tag{14}$$

and the corresponding *upper $\mathcal{L}$-proximal map* $z^\tau\colon \mathbb{R}^k \to \mathbb{R}^{n+m}$ ($\mathcal{L}$-proximal map) as

$$\begin{aligned}z^\tau(w) &:= \arg\max_{x\in\mathcal{X}}\min_{y\in\mathcal{Y}} \ell(x) - \tfrac{1}{\tau}D_{\mathcal{L}}(x, y|w) \\ &= \arg\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} \mathcal{L}(x, y, w) - \tau\ell(x).\end{aligned} \tag{15}$$

The lower and upper $\mathcal{L}$-envelope are lower and upper bound approximations of the loss $w \mapsto \ell(x^*(w))$, respectively. We emphasize that the solutions to (12) and (14) are in general set-valued and we assume that they are non-empty and admit a single-valued selection that is continuous at w, which we

[4]In some cases, the Lagrangian divergence coincides with the *Bregman divergence*, which generalizes the squared Euclidean distance, opening connections to *Mirror descent*, see Appendix D.

denote by (13) and (15). We also assume that strong duality holds for the upper and lower $\mathcal{L}$-envelopes.[5] We will also work with the *average* $\mathcal{L}$-envelope

$$\ell_{\tau}(w) := \tfrac{1}{2}\big[\ell_{\tau}(w) + \ell^{\tau}(w)\big]. \tag{16}$$

The different envelopes are closely related to right-, left- and double-sided directional derivatives.

5.3. Lagrangian Proximal Point Method

Our goal will be to perform gradient descent on the $\mathcal{L}$-envelope (12, 14). The gradients of the $\mathcal{L}$-envelopes read

$$\begin{aligned} \nabla \ell_{\tau}(w) &= \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}(z_{\tau}, w) - \mathcal{L}(z^*, w)\big],\\ \nabla \ell^{\tau}(w) &= \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}(z^*, w) - \mathcal{L}(z^{\tau}, w)\big], \end{aligned} \tag{17}$$

where we abbreviate $z_{\tau} = z_{\tau}(w)$ and $z^{\tau} = z^{\tau}(w)$. The proof is in Appendix F. In analogy to the proximal point method (5) we refer to GD using (17) as the *Lagrangian Proximal Point Method* (LPPM), or specifically, LPPM$_{\tau}$, LPPMτ and LPPM$^{\tau}$ for GD on ℓ_{τ}, $\ell\tau$ and ℓ^{τ}, respectively.

Example 1 (Direct Loss Minimization)**.** For an input $\mu \in \mathbb{R}^p$, label $x_{\text{true}} \in \mathcal{X}$, loss $\ell\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, feature map $\Psi\colon \mathcal{X} \times \mathbb{R}^p \to \mathbb{R}^k$ and an optimization problem of the form

$$x^*(w, \mu) = \arg\min_{x\in\mathcal{X}} -\langle w, \Psi(x, \mu)\rangle \tag{18}$$

the LPPM$_{\tau}$ update (17) reads

$$\nabla \ell_{\tau}(w) = \tfrac{1}{\tau}\big[\Psi(x^*, \mu) - \Psi(x_{\tau}, \mu)\big], \tag{19}$$

with $x^* = x^*(w, \mu)$ and

$$x_{\tau} = \arg\min_{x\in\mathcal{X}} -\langle w, \Psi(x, \mu)\rangle + \tau\ell(x, x_{\text{true}}). \tag{20}$$

This recovers the "towards-better" *Direct Loss Minimization* (DLM) update (McAllester et al., 2010), while the "away-from-worse" update corresponds to the LPPM$^{\tau}$ update, both of which were proposed in the context of taking the limit $\tau \to 0$, which aims to compute the true gradients.

5.4. Lagrangian Proximal Gradient Descent

LPPM requires computing the $\mathcal{L}$-proximal map (13) or (15). Due to the loss term, efficiently solving the involved optimization problem might not be possible or it requires choosing and implementing a custom optimization algorithm that is potentially much slower than the oracle used to solve the forward problem (2). Instead, we aim to introduce an approximation of the loss that allows solving the $\mathcal{L}$-proximal map with the forward solver oracle. We first observe that in many cases the parameterized Lagrangian takes the form

$$\mathcal{L}(x, y, w) = \langle x, c\rangle + \Omega(x, y, v), \tag{21}$$

with $w = (c, v)$, linear parameters $c \in \mathbb{R}^n$, non-linear parameters $v \in \mathbb{R}^{k-n}$ and continuously differentiable Ω. Our approximation, inspired by proximal gradient descent (7), is to consider a linearization $\widetilde{\ell}$ of the loss ℓ at x^*.[6] Importantly, the loss linearization is only applied *after the solver* and does not approximate or linearize the solution mapping. Abbreviating $\nabla\ell = \nabla\ell(x^*)$, we get the $\mathcal{L}$-proximal maps

$$\begin{aligned} \widetilde{z}_{\tau}(w) &:= \arg\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} \langle x, c\rangle + \Omega(x, y, v) + \tau\langle x, \nabla\ell\rangle\\ &= z^*(c + \tau\nabla\ell, v), \end{aligned} \tag{22}$$

$$\begin{aligned} \widetilde{z}^{\tau}(w) &:= \arg\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} \langle x, c\rangle + \Omega(x, y, v) - \tau\langle x, \nabla\ell\rangle\\ &= z^*(c - \tau\nabla\ell, v). \end{aligned} \tag{23}$$

As these are instances of the forward problem (2) on different parameters, they can be computed with the same solver oracle. The assumed strong duality and continuity of the solution of (2) also typically implies the same properties for the perturbed problems (22, 23). Note that warm-starting the solver with z^* can strongly accelerate the computation, often making the evaluation of the $\mathcal{L}$-proximal map much faster than the forward problem. This enables efficient computation of the $\mathcal{L}$-envelope gradient (17) as

$$\begin{aligned} \nabla\widetilde{\ell}_{\tau}(w) &= \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}\big(w, \widetilde{z}_{\tau}\big) - \mathcal{L}\big(w, z^*\big)\big],\\ \nabla\widetilde{\ell}^{\tau}(w) &= \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}\big(w, z^*\big) - \mathcal{L}\big(w, \widetilde{z}^{\tau}\big)\big]. \end{aligned} \tag{24}$$

In analogy to proximal gradient descent (7) we refer to gradient descent using (24) as *Lagrangian Proximal Gradient Descent* (LPGD), or more specifically, to LPGD$_{\tau}$, LPGDτ and LPGD$^{\tau}$ for GD on $\widetilde{\ell}_{\tau}$, $\widetilde{\ell}\tau$ and $\widetilde{\ell}^{\tau}$, respectively.

LPGD can also be viewed as computing gradients of the loss $\nabla_w \ell(x^*(w))$ standardly by rolling out the chain rule, but replacing every appearance of the optimization layer co-derivative with a certain finite-difference.[7] This shows that LPGD only affects the optimization layer, providing a derivative replacement that carries higher-order information than linear sensitivities. Further, this viewpoint directly implies that LPGD smoothly integrates into existing automatic differentiation frameworks (Abadi et al., 2015; Bradbury et al., 2018; Paszke et al., 2019), by simply replacing the backward pass operation of the optimization layer with the finite-difference, as summarized for LPGD$_{\tau}$ in Algorithm 1.

[5] In general, these assumptions cast strong restrictions on the allowed loss functions. This is one of the reasons why our main focus will be on loss approximations introduced in section 5.4.

[6] Note that other approximations of the loss can also be used depending on the parameterization supported by the forward solver. For example, if quadratic terms are supported we could consider a quadratic loss approximation.

[7] Note that this only requires a single additional solver evaluation, in contrast to traditional finite-difference gradient estimation requiring k solver evaluations, which is often intractable.

Algorithm 1 Forward and Backward Pass of LPGD$_\tau$

```
function ForwardPass($w$)
    $z^* \leftarrow$ SolverOracle($w$)    // Solve optimization (2)
    save $w$, $z^*$ for backward pass
    return $z^*$
end function

function BackwardPass($\nabla\ell = \nabla_z \ell(z^*)$, $\tau$)
    load $(c, v) = w$ and $z^*$ from forward pass
    $w_\tau \leftarrow c + \tau\nabla\ell, v$    // Perturb parameters
    $\widetilde{z}_\tau \leftarrow$ SolverOracle($w_\tau$)    // (22), warmstart with $z^*$
    $\nabla_w \widetilde{\ell}_\tau(w) = \frac{1}{\tau}\nabla_w\big[\mathcal{L}(\widetilde{z}_\tau, w) - \mathcal{L}(z^*, w)\big]$    // (24)
    return $\nabla_w \widetilde{\ell}_\tau(w)$    // Gradient of $\mathcal{L}$-envelope
end function
```

Example 2 (Blackbox Backpropagation)**.** For a linear program (LP)[8]

$$x^*(c) = \arg\min_{x\in\mathcal{X}} \langle x, c\rangle, \tag{25}$$

the LPGD$_\tau$ update (24) reads

$$\nabla\widetilde{\ell}_\tau(c) = \tfrac{1}{\tau}\big[\widetilde{x}_\tau(c) - x^*(c)\big] = \tfrac{1}{\tau}\big[x^*(c + \tau\nabla\ell) - x^*(c)\big],$$

which recovers the update rule in *Blackbox Backpropagation* (BB) (Vlastelica et al., 2020). The piecewise affine interpolation of the loss $c \mapsto \widetilde{\ell}(x^*(c))$ derived in BB agrees with the lower $\mathcal{L}$-envelope $\widetilde{\ell}_\tau$.

Example 3 (Implicit Differentiation by Perturbation)**.** For a regularized linear program

$$x^*(c) = \arg\min_{x\in\mathcal{X}} \langle x, c\rangle + \Omega(x) \tag{26}$$

with a strongly convex regularizer $\Omega\colon \mathcal{X} \to \mathbb{R}$, the LPGD$_{\overline{\tau}}$ update (24) reads

$$\begin{aligned}\nabla\widetilde{\ell}_{\overline{\tau}}(c) &= \tfrac{1}{2\tau}\big[\widetilde{x}_\tau(c) - \widetilde{x}^\tau(c)\big] \\ &= \tfrac{1}{2\tau}\big[x^*(c + \tau\nabla\ell) - x^*(c - \tau\nabla\ell)\big],\end{aligned} \tag{27}$$

recovering the update in Domke (2010), where only the limit case $\tau \to 0$ is considered.

5.5. Regularization & Augmented Lagrangian

To increase the smoothness of the $\mathcal{L}$-envelope, we augment the Lagrangian with a strongly convex regularizer

$$\mathcal{L}_\rho(x, y, w) := \mathcal{L}(x, y, w) + \tfrac{1}{2\rho}\|x - x^*\|_2^2 \tag{28}$$

with $\rho > 0$. Equivalently, we may re-introduce the quadratic regularizer from the Moreau envelope (4) into the $\mathcal{L}$-envelope (12) and $\mathcal{L}$-proximal map (13)

$$\begin{aligned}\ell_{\tau\rho}(w) := \min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} &\,\ell(x) + \tfrac{1}{\tau}D_{\mathcal{L}}(x, y|w) \\ &+ \tfrac{1}{2\rho}\|x - x^*\|_2^2,\end{aligned} \tag{29}$$

$$\begin{aligned}z_{\tau\rho}(w) := \arg\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} &\,\ell(x) + \tfrac{1}{\tau}D_{\mathcal{L}}(x, y|w) \\ &+ \tfrac{1}{2\rho}\|x - x^*\|_2^2.\end{aligned} \tag{30}$$

These definitions have an analogy for the upper envelope and for a linearized loss, which we omit for brevity. The LPPM$_\tau$ and LPGD$_\tau$ updates then take the form

$$\begin{aligned}\nabla\ell_{\tau\rho}(w) &= \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}(w, z_{\tau\rho}) - \mathcal{L}(w, z^*)\big], \\ \nabla\widetilde{\ell}_{\tau\rho}(w) &= \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}(w, \widetilde{z}_{\tau\rho}) - \mathcal{L}(w, z^*)\big].\end{aligned} \tag{31}$$

The augmentation does not alter the current optimal solution, but smoothens the Lagrange-Moreau envelope. This also has connections to Jacobian-regularization in the implicit function theorem, which we discuss in Appendix C. Note that using this quadratic regularization with LPGD requires the solver oracle to support quadratic objectives, which is the case for the conic program solver used in our experiments. We visualize the smoothing of the different $\mathcal{L}$-envelopes of the loss $c \mapsto \ell(x^*(c))$ in Figure 1, for a quadratic loss on the solution to the linear program (25) with $\mathcal{X} = [0, 1]^n$ and a one-dimensional random cut through the cost space.[9]

6. Method Analysis

We now theoretically analyze our method. Proofs of the statements are given in Appendix F. The following proposition characterizes the continuity properties that the $\mathcal{L}$-envelope inherits from the Lagrangian.

Proposition 6.1. *Assume that $\mathcal{L}$ is L-Lipschitz continuous in w. Then $\ell_\tau, \ell_{\overline{\tau}}, \ell^\tau$ are $\frac{2L}{\tau}$-Lipschitz continuous in w.*

This shows that by increasing τ we get smaller Lipschitz bounds of the $\mathcal{L}$-envelope, giving a form of "smoothening".

Next, we characterize the asymptotic behavior of the $\mathcal{L}$-envelope, $\mathcal{L}$-proximal map, LPPM and LPGD updates. First, we consider the limit as $\tau \to 0$. Let $X^*(w)$ and $\widehat{\mathcal{X}}(w)$ denote the *optimal solution set* and the *effective feasible set*, respectively, defined as

$$X^*(w) := \{x \in \mathcal{X} \mid \mathcal{D}^*_{\mathcal{L}}(x|w) = 0\}, \tag{32}$$

$$\widehat{\mathcal{X}}(w) := \{x \in \mathcal{X} \mid \mathcal{D}^*_{\mathcal{L}}(x|w) < \infty\}. \tag{33}$$

[8]For an LP over a polytope $\mathcal{X}$ the space of possible solutions is discrete. Whenever the solution is unique, which is true for almost every w, the solution mapping is locally constant (and hence continuous) around w. Therefore our continuity assumptions hold for almost all w.

[9]Here, the envelopes appear non-continuous due to the loss linearization. Namely, the point $x^*(c)$ at which we take the loss linearization varies with c, and therefore the function of which we take the envelope varies as well.

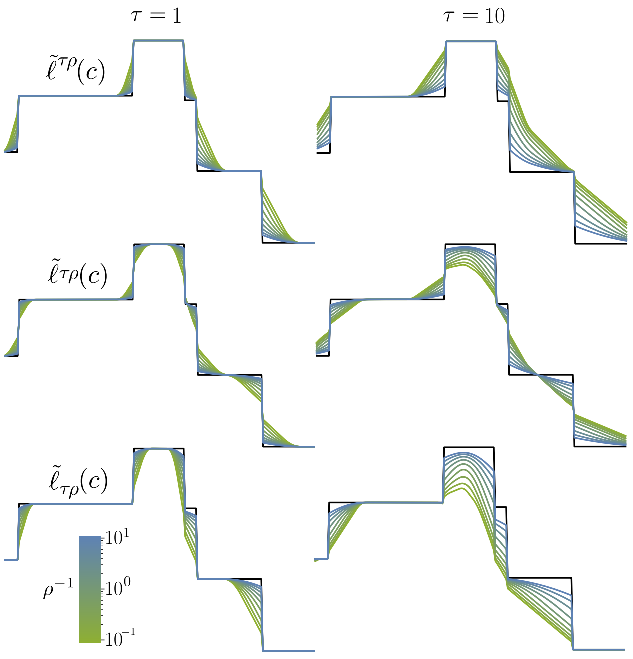


Figure 1: Visualization of the upper $\widetilde{\ell}^{\tau\rho}$, average $\widetilde{\ell}_{\tau}\rho$, and lower $\widetilde{\ell}_{\tau\rho}$ Lagrange-Moreau envelope for different temperatures τ and augmentation strengths ρ. The envelopes are smoothed approximations of the loss $c \mapsto \ell(x^*(c))$ (black).

Proposition 6.2. *Let w be such that $X^*(w) \neq \emptyset$. Assume $\mathcal{L}, \ell$ are lower semi-continuous and ℓ is finite-valued on $\mathcal{X}$. Then it holds that*

$$\lim_{\tau \to 0} \ell_\tau(w) := \min_{x^* \in X^*(w)} \ell(x^*) \tag{34}$$

and also

$$\lim_{\tau \to 0} x^*_\tau(w) \in \arg\min_{x^* \in X^*(w)} \ell(x^*). \tag{35}$$

whenever the limit exists.

Propositions 6.1 and 6.2 highlight that choosing τ determines a trade-off between smoothness and tightness of the approximation. Next, we show that the LPGD update converges to the true gradient.

Theorem 6.3. *Let $\mathcal{L} \in \mathcal{C}^2$ and assume the optimizer of (2) admits a differentiable selection $x^*(w)$ at w. Then*

$$\lim_{\tau \to 0} \nabla \widetilde{\ell}_\tau(w) = \nabla_w \ell(x^*(w)) = \lim_{\tau \to 0} \nabla \widetilde{\ell}^\tau(w). \tag{36}$$

The proof, also highlighting the connections between LPGD updates and finite-difference approximations of forward-mode derivatives, is given in Appendix F. Theorem 6.3 asserts that LPGD computes the same gradients in the limit as methods based on the implicit function theorem, such as *OptNet* (Amos & Kolter, 2017a), regularized LPs (Wilder et al., 2019a; Mandi & Guns, 2020), or differentiable conic programs (Agrawal et al., 2019b).

Next, we consider the limit $\tau \to \infty$. First, we have the result for the lower $\mathcal{L}$-proximal map (13).

Proposition 6.4. *Let w be such that $\widehat{\mathcal{X}}(w) \neq \emptyset$. Then*

$$\lim_{\tau \to \infty} x_\tau(w) \in \arg\min_{x \in \widehat{\mathcal{X}}(w)} \ell(x) \tag{37}$$

whenever the limit exists. For a linearized loss, we have

$$\lim_{\tau \to \infty} \widetilde{x}_\tau(w) \in \arg\min_{x \in \widehat{\mathcal{X}}(w)} \langle x, \nabla \ell \rangle = x_{FW}(w), \tag{38}$$

where x_{FW} is the solution to a Frank-Wolfe iteration LP (Frank & Wolfe, 1956)

Next proposition covers the case of the lower $\mathcal{L}$-proximal map (30) with a quadratic regularizer.

Proposition 6.5. *The primal lower $\mathcal{L}$-proximal map (30) turns into the standard proximal map (5)*

$$\begin{aligned} \lim_{\tau \to \infty} x_{\tau\rho}(w) &= \arg\min_{x \in \widehat{\mathcal{X}}(w)} \left[\ell(x) + \tfrac{1}{2\rho}\|x - x^*\|_2^2\right] \\ &= \operatorname{prox}_{\rho\ell + I_{\widehat{\mathcal{X}}(w)}}(x^*), \end{aligned} \tag{39}$$

whenever the limit exists. For a linearized loss, it reduces to the Euclidean projection onto $\widehat{\mathcal{X}}(w)$

$$\begin{aligned} \lim_{\tau \to \infty} \widetilde{x}_{\tau\rho}(w) &= \arg\min_{x \in \widehat{\mathcal{X}}(w)} \left[\langle x, \nabla \ell \rangle + \tfrac{1}{2\rho}\|x - x^*\|_2^2\right] \\ &= P_{\widehat{\mathcal{X}}(w)}(x^* - \rho \nabla \ell). \end{aligned} \tag{40}$$

The LPPM$_\tau$ (17, 31) and LPGD$_\tau$ (24, 31) updates corresponding to the $\mathcal{L}$-proximal maps (37, 39) and (38, 40) have the interpretation of decoupling the update step, by first computing a "target" (e.g. $\widetilde{x}_{\tau\rho}$ via projected gradient descent with step-size ρ), and then minimizing the Lagrangian divergence to make the target the new optimal solution.

We discuss multiple examples that showcase the asymptotic variations of LPPM and LPGD. Here, we will work with the finite-difference version of the updates (17, 31), denoted by

$$\Delta \ell_\tau(w) := \tau \nabla \ell_\tau(w), \qquad \Delta \ell^\tau(w) := \tau \nabla \ell^\tau(w), \tag{41}$$

$$\Delta \ell_{\tau\rho}(w) := \tau \nabla \ell_{\tau\rho}(w), \quad \Delta \widetilde{\ell}_{\tau\rho}(w) := \tau \nabla \widetilde{\ell}_{\tau\rho}(w). \tag{42}$$

Example 4 (Identity with Projection). For an LP (25) it is $\widehat{\mathcal{X}}(w) = \mathcal{X}$ and we get the asymptotic regularized LPGD$_\tau$ update (31) in finite-difference form (42) as

$$\begin{aligned} \lim_{\tau \to \infty} \Delta \widetilde{\ell}_{\tau\rho}(c) &= \lim_{\tau \to \infty} \left[\widetilde{x}_{\tau\rho}(c) - x^*\right] \\ &= P_{\mathcal{X}}(x^* - \rho \nabla \ell) - x^*, \end{aligned} \tag{43}$$

where we used (40). In the limit of large regularization $\rho \to 0$ with division by ρ in analogy to Theorem 6.3, the above update converges to

$$\lim_{\rho\to 0}\lim_{\tau\to\infty}\tfrac{1}{\rho}\Delta\widetilde{\ell}_{\tau\rho}(c) = \lim_{\rho\to 0}\tfrac{1}{\rho}\big[P_{\mathcal{X}}(x^* - \rho\nabla\ell) - x^*\big]$$
$$= DP_{\mathcal{X}}(x^*|-\nabla\ell) = D^*P_{\mathcal{X}}(x^*|-\nabla\ell),$$

where DP and D^*P denote the directional derivative and coderivative of the projection P at x^*. This is closely related to the *Identity with Projection* method by Sahoo et al. (2023), in which the true gradient is replaced by backpropagating $-\nabla\ell$ through the projection onto a relaxation of $\mathcal{X}$.[10]

Example 5 (Smart Predict then Optimize)**.** The *Smart Predict then Optimize* (SPO) setting (Mandi et al., 2020; Elmachtoub & Grigas, 2022) embeds an LP (25) as the final component of the prediction pipeline and assumes access to the ground truth cost c_{true}. The goal is to optimize the SPO loss $\ell_{\text{SPO}}(x^*(c), c_{\text{true}}) = \langle x^*(c) - x^*(c_{\text{true}}), c_{\text{true}}\rangle$. Due to the discreteness of the LP, the SPO loss has degenerate gradients with respect to c, i.e. they are zero almost everywhere and undefined otherwise. Choosing $\tau = \frac{1}{2}$ for the upper $\mathcal{L}$-proximal map (15), we get

$$x^{\frac{1}{2}}(c) = \arg\max_{x\in\mathcal{X}}\langle x - x^*(c_{\text{true}}), c_{\text{true}}\rangle - 2\langle x - x^*, c\rangle$$
$$= \arg\max_{x\in\mathcal{X}}\langle x, c_{\text{true}} - 2c\rangle \tag{44}$$

which gives the lower and upper LPPM updates (17) in finite-difference form (41)

$$\Delta\ell_\tau(c) = x_\tau(c) - x^* \text{ and } \Delta\ell^{\frac{1}{2}}(c) = x^* - x^{\frac{1}{2}}(c). \tag{45}$$

Summing both the updates and taking the limit $\tau \to \infty$ yields the combined LPPM update

$$\lim_{\tau\to\infty}\big[\Delta\ell_\tau(c) + \Delta\ell^{\frac{1}{2}}(c)\big] = \lim_{\tau\to\infty}\big[x_\tau(c) - x^{\frac{1}{2}}(c)\big]$$
$$= x^*(c_{\text{true}}) - x^{\frac{1}{2}}(c) = \tfrac{1}{2}\nabla\ell_{\text{SPO+}}(c, c_{\text{true}}), \tag{46}$$

where we used (37). Note that as the SPO loss is already linear in x, LPPM and LPGD are equivalent. Update (46) recovers the gradient of the SPO+ loss

$$\ell_{\text{SPO+}}(c, c_{\text{true}}) := \sup_{x\in\mathcal{X}}\langle x, c_{\text{true}} - 2c\rangle + 2\langle x^*(c_{\text{true}}), c\rangle$$
$$- \langle x^*(c_{\text{true}}), c_{\text{true}}\rangle \tag{47}$$

introduced by Elmachtoub & Grigas (2022), which has found widespread applications.

Example 6 (Fenchel-Young Losses[11])**.** In the *structured prediction* setting we consider the regularized LP (26) as the final component of the prediction pipeline and assume access to the ground truth solutions x_{true}. The goal is to bring $x^*(c)$ close to x_{true} by minimizing any loss $\ell(x)$ that is minimized over $\mathcal{X}$ at x_{true}. We compute the asymptotic LPPM$_\tau$ update (17) in finite-difference form (41) as

$$\lim_{\tau\to\infty}\Delta\ell_\tau(c) = \lim_{\tau\to\infty}\big[x_\tau(c) - x^*\big]$$
$$= x_{\text{true}} - x^* = \nabla\ell_{\text{FY}}(c, x_{\text{true}}), \tag{48}$$

where we used (37) to compute the limit. This recovers the gradient of the Fenchel-Young loss[12]

$$\ell_{\text{FY}}(c, x_{\text{true}}) := \max_{x\in\mathcal{X}}\big[-\langle c, x\rangle - \Omega(x)\big] + \Omega(x_{\text{true}}) + \langle c, x_{\text{true}}\rangle$$
$$= \langle c, x_{\text{true}}\rangle + \Omega(x_{\text{true}}) - \min_{x\in\mathcal{X}}\big[\langle c, x\rangle + \Omega(x)\big]$$

defined by Blondel et al. (2020). Depending on the regularizer Ω and the feasible region $\mathcal{X}$, Fenchel-Young losses cover multiple structured prediction setups, including the *structured hinge* (Tsochantaridis et al., 2005), *CRF* (Lafferty et al., 2001), and *SparseMAP* (Niculae et al., 2018) losses.

Finally, note that solvers in general return approximate solutions, yielding an approximate $\mathcal{L}$-envelop denoted by $\widehat{\ell}_\tau(w)$, which we quantify in the following proposition.

Proposition 6.6. *Assume that the solvers for (2, 13) return ε-accurate solutions with δ-accurate objective values. Then*

$$|\widehat{\ell}_\tau(w) - \ell_\tau(w)| \leq \tfrac{2\delta}{\tau}. \tag{49}$$

Moreover, if $\nabla_w\mathcal{L}(x, y, w)$ is L-Lipschitz continuous in $(x, y) \in \mathcal{X} \times \mathcal{Y}$, it holds that

$$\|\nabla\widehat{\ell}_\tau(w) - \nabla\ell_\tau(w)\| \leq \tfrac{2L\varepsilon}{\tau}. \tag{50}$$

This shows that increasing the temperature τ helps reducing the error introduced by approximate solutions.

7. Experiments

Previous works have successfully applied variations of LPGD and LPPM to two types of experimental settings: 1) Producing informative gradient replacements when the derivative of the solution-mapping is degenerate (Rolínek et al., 2020a;b; Mandi et al., 2020; Sahoo et al., 2023; Ferber et al., 2023) and 2) efficiently computing the true gradient in the limit as $\tau \to 0$ when the derivative of the solution-mapping is non-degenerate (Domke, 2010; McAllester et al., 2010). However, our interpretation of LPGD, as updates capturing higher-order sensitivities than standard gradients, suggests the application to a third potential setting: 3) Using LPGD with finite values of τ even when the derivative of the solution-mapping is non-degenerate. We will compare LPGD to GD in two such cases: a) Learning the rules of Sudoku from synthetic data and b) tuning the parameters of a Markowitz control policy on historical trading data.

[10]Note that this also has close ties to the one-step gradient arising in implicit differentiation of fixed-point iterations by treating the inverse Jacobian as an identity function (Geng et al., 2021; Chang et al., 2022; Bai et al., 2022).

[11]Note that an analogous derivation holds for *generalized* Fenchel-Young losses (Blondel et al., 2022), in which the regularized LP is replaced with a regularized energy function.

[12]Note that Blondel et al. (2020) consider maximization.

Implementation. We build on the CVXPY ecosystem (Diamond & Boyd, 2016; Agrawal et al., 2018; 2019a) to implement LPGD for a large class of parameterized optimization problems. CVXPY automatically reduces parameterized optimization problems to conic programs in a differentiable way, which are then solved with the SCS solver (O'Donoghue et al., 2016). These solutions can then be differentiated based on the results of Agrawal et al. (2019b), which is natively implemented in CVXPY.[13] [14] As an alternative to the true conic program derivative computation based on Agrawal et al. (2019b), we implement Algorithm 1 (in all variations). This allows using LPGD for the large class of parameterized convex optimziation problems supported by CVXPY without modification. The code is available at github.com/martius-lab/diffcp-lpgd.

7.1. Learning the Rules of Sudoku

We consider a version of the Sudoku experiment proposed by Amos & Kolter (2017a). The task is to learn the rules of Sudoku in the form of linear programming constraints from pairs of incomplete and solved Sudoku puzzles. See Appendix G for details.

The results are reported in Figure 2. LPGD reaches a lower final loss than GD, which suggests that LPGD$_\tau$ produces better update steps than standard gradients. We observe faster convergence of LPGD compared to GD in terms of wallclock time, which is due to the faster backward pass computation resulting from warmstarting. Note that the forward solution time over training increases as the initial random optimization problem becomes more structured as the loss decreases. Additional results can be found in Appendix G.1.

7.2. Tuning a Markowitz Control Policy

We now consider the Markowitz Portfolio Optimization setting described by Agrawal et al. (2020, § 5). The task is to tune a convex optimization control policy that iteratively trades assets over a trading horizon. The policy parameters are initialized as a Markowitz model based on historical data. Differentiating through the optimization problem in the control policy allows tuning them to maximize the utility on simulated evolutions of the asset values. See Appendix G.2 for a more detailed description.

In Figure 3a, we report a sweep over the learning rate α. The best-performing GD run achieves an improvement of 23% over the Markowitz initialization, while the best-performing

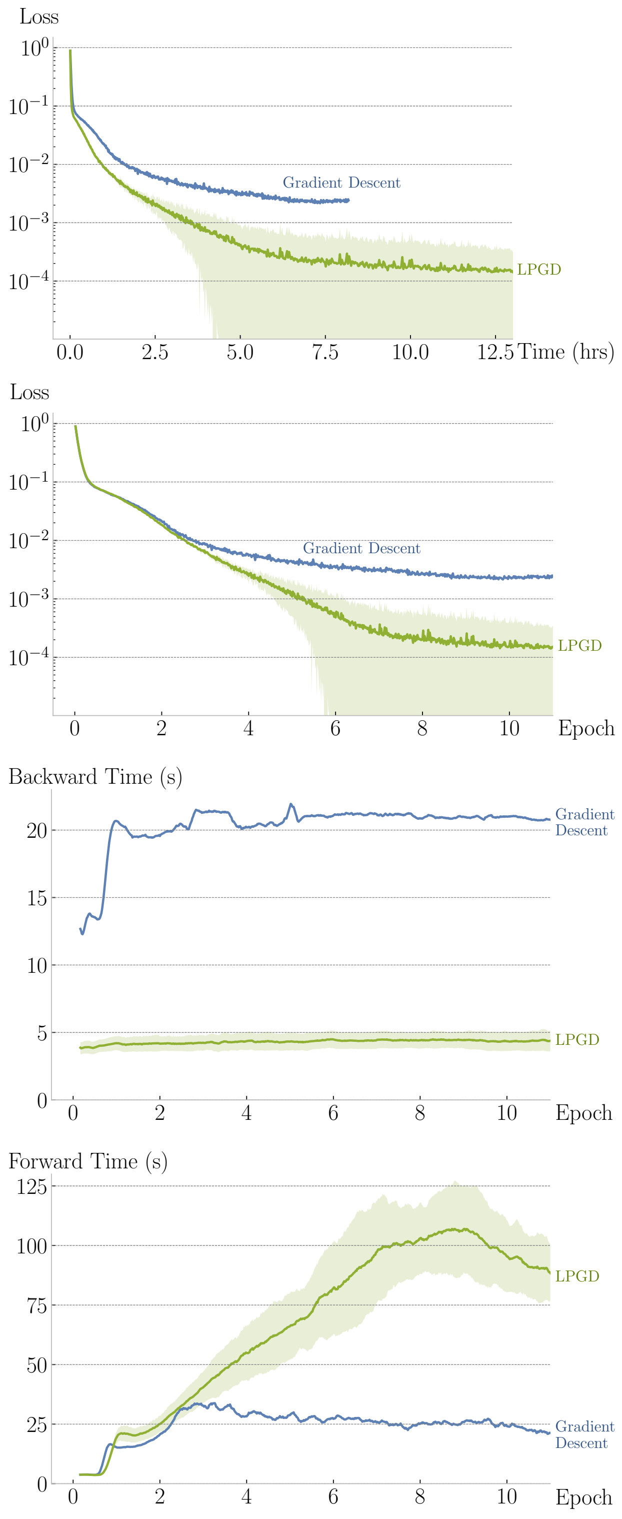


Figure 2: Comparison of LPGD and gradient descent (GD) on the Sudoku experiment. Reported train MSE over epochs, wall-clock time, and time spent in the backward and forward passes. Statistics are over 5 restarts.

[13]We modify this implementation to support the regularization term as described in Appendix C.

[14]Note that for many optimization problems the automatic reduction results in conic programs that have degenerate derivatives, in which case the method of Agrawal et al. (2019b) returns a heuristic quantity for the gradient without strong theoretical support.

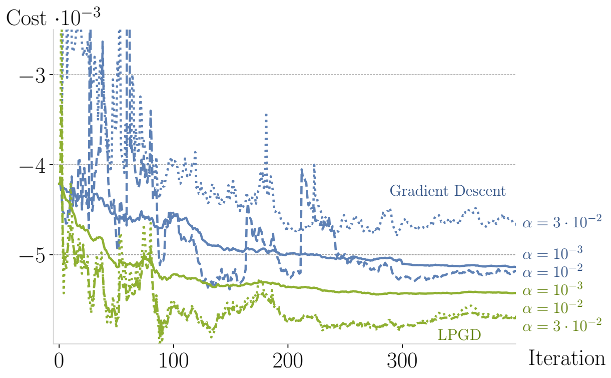
LPGD run achieves an improvement of 35%. Moreover, LPGD converges faster in terms of iterations, with both methods requiring similar runtime. Note that with a higher learning rate, GD becomes noisier than LPGD. For both methods, runs with $\alpha = 0.1$ diverged and terminated due to infeasible conic programs. In Figure 3b, we report a sweep on the temperature τ of LPGD. For low $\tau = 0.1$, LPGD performs badly, as numerical issues paired with the backpropagation through time make the training dynamics unstable. For $\tau = 1$, LPGD matches the performance of GD, as LPGD updates approximate the true gradients. The best performance is achieved for $\tau = 100$, for which LPGD provides higher-order information than the gradients. For $\tau = 1000$, the strong perturbations sometimes result in the infeasibility of the corresponding conic program, which causes the run to terminate. This shows the practical trade-off in selecting the temperature parameter τ, as discussed in the theoretical analysis of LPGD. In Figure 3c, we also conduct a sweep over the solver accuracy ϵ, showing that GD and LPGD have similar sensitivities to inaccurate solutions. Additional results can be found in Appendix G.2.

8. Conclusion

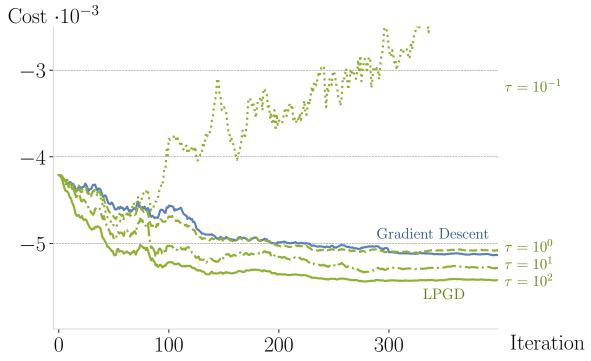
We propose *Lagrangian Proximal Gradient Descent* (LPGD), a flexible framework for learning parameterized optimization models. **LPGD unifies and generalizes various state-of-the-art contemporary optimization methods**, including *Direct Loss Minimization* (McAllester et al., 2010), *Blackbox Backpropagation* (Vlastelica et al., 2020), *Implicit Differentiation by Perturbation* (Domke, 2010), *Identity with Projection* (Sahoo et al., 2023), *Smart Predict then Optimize* (Elmachtoub & Grigas, 2022), and *Fenchel-Young losses* (Blondel et al., 2020; 2022), and **provides deep links to traditional optimization methods**.

LPGD computes updates as finite differences and only requires accessing the forward solver as a black-box oracle, which makes it extremely simple to implement. We also provide an implementation of LPGD that smoothly integrates it into the CVXPY ecosystem. Formulated as gradient descent on a loss function envelope, LPGD allows learning general objective and constraint parameters of saddlepoint problems even for solution mappings with degenerate derivatives.

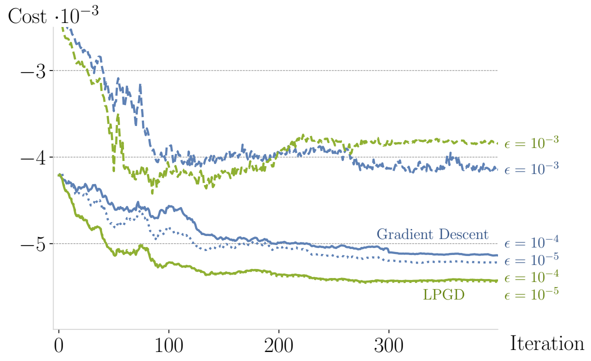
Various special cases of LPGD have shown impressive results in optimizing parameters of solution mappings with degenerate derivatives and speeding up the computation of non-degenerate derivatives. We explore a new experimental direction by using LPGD to efficiently compute more informative updates even when non-degenerate derivatives exist. We find on two experiments that LPGD can achieve faster convergence and better final results when compared to gradient descent.



(a) Learning rate α sweep.



(b) Temperature τ sweep.



(c) Solver accuracy ϵ sweep.

Figure 3: Comparison of LPGDτ and GD in the Markowitz portfolio optimization experiment. When not specified otherwise, we use $\alpha = 0.001$, $\epsilon = 0.0001$, $\tau = 100$ and $\rho = 0$.

Acknowledgements

We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Anselm Paulus. Georg Martius is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645. This work was supported by the ERC - 101045454 REAL-RL. We acknowledge the support from the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039B). This work was supported by the Cyber Valley Research Fund. Vít Musil was supported by the Czech Science Foundation Grant GA23-06963S.

Impact Statement

This paper presents theory-oriented work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

Agrawal, A., Verschueren, R., Diamond, S., and Boyd, S. A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60, 2018.

Agrawal, A., Amos, B., Barratt, S. T., Boyd, S. P., Diamond, S., and Kolter, J. Z. Differentiable convex optimization layers. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 9558–9570, 2019a. URL https://proceedings.neurips.cc/paper/2019/hash/9ce3c52fc54362e22053399d3181c638-Abstract.html.

Agrawal, A., Barratt, S., Boyd, S., Busseti, E., and Moursi, W. M. Differentiating through a cone program. *J. Appl. Numer. Optim*, 1(2):107–115, 2019b. URL https://doi.org/10.23952/jano.1.2019.2.02.

Agrawal, A., Barratt, S., Boyd, S., and Stellato, B. Learning convex optimization control policies. In *Learning for Dynamics and Control*, pp. 361–373. PMLR, 2020.

Amos, B. and Kolter, J. Z. Optnet: Differentiable optimization as a layer in neural networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 136–145. PMLR, 2017a. URL http://proceedings.mlr.press/v70/amos17a.html.

Amos, B. and Kolter, J. Z. OptNet Repository. https://github.com/locuslab/optnet, 2017b.

Amos, B., Koltun, V., and Kolter, J. Z. The limited multi-label projection layer. *CoRR*, abs/1906.08707, 2019. URL http://arxiv.org/abs/1906.08707.

Bai, S., Kolter, J. Z., and Koltun, V. Deep equilibrium models. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 688–699, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/01386bd6d8e091c2ab4c7c7de644d37b-Abstract.html.

Bai, S., Koltun, V., and Kolter, J. Z. Multiscale deep equilibrium models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3812f9a59b634c2a9c574610eaba5bed-Abstract.html.

Bai, S., Geng, Z., Savani, Y., and Kolter, J. Z. Deep equilibrium optical flow estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 610–620. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00070. URL https://doi.org/10.1109/CVPR52688.2022.00070.

Bauschke, H. H. and Combettes, P. L. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer, 2011. ISBN

978-1-4419-9466-0. doi: 10.1007/978-1-4419-9467-7. URL https://doi.org/10.1007/978-1-4419-9467-7.

Bauschke, H. H., Dao, M. N., and Lindstrom, S. B. Regularizing with bregman-moreau envelopes. *SIAM J. Optim.*, 28(4):3208–3228, 2018. doi: 10.1137/17M1130745. URL https://doi.org/10.1137/17M1130745.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009. doi: 10.1137/080716542. URL https://doi.org/10.1137/080716542.

Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J.-P., and Bach, F. Learning with differentiable pertubed optimizers. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9508–9519. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6bb56208f672af0dd65451f869fedfd9-Paper.pdf.

Blondel, M. Structured prediction with projection oracles. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 12145–12156, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/7990ec44fcf3d7a0e5a2add28362213c-Abstract.html.

Blondel, M., Martins, A. F. T., and Niculae, V. Learning with fenchel-young losses. *J. Mach. Learn. Res.*, 21:35:1–35:69, 2020. URL http://jmlr.org/papers/v21/19-021.html.

Blondel, M., Llinares-López, F., Dadashi, R., Hussenot, L., and Geist, M. Learning energy networks with generalized fenchel-young losses. In *NeurIPS*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/510cfd9945f8bde6f0cf9b27ff1f8a76-Abstract-Conference.html.

Boyd, S. P. and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, 2014. ISBN 978-0-521-83378-3. doi: 10.1017/CBO9780511804441. URL https://web.stanford.edu/%7Eboyd/cvxbook/.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Busseti, E., Moursi, W. M., and Boyd, S. P. Solution refinement at regular points of conic problems. *Comput. Optim. Appl.*, 74(3):627–643, 2019. doi: 10.1007/S10589-019-00122-9. URL https://doi.org/10.1007/s10589-019-00122-9.

Chang, M., Griffiths, T., and Levine, S. Object representations as fixed points: Training iterative refinement algorithms with implicit differentiation. In *NeurIPS*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/d301e2878a7ebadf1a95029e904fc7d0-Abstract-Conference.html.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 6572–6583, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/69386f6bb1dfed68692a24c8686939b9-Abstract.html.

Combettes, P. L. and Pesquet, J. Proximal splitting methods in signal processing. In Bauschke, H. H., Burachik, R. S., Combettes, P. L., Elser, V., Luke, D. R., and Wolkowicz, H. (eds.), *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, volume 49 of *Springer Optimization and Its Applications*, pp. 185–212. Springer, 2011. doi: 10.1007/978-1-4419-9569-8_10. URL https://doi.org/10.1007/978-1-4419-9569-8_10.

Cranmer, M. D., Greydanus, S., Hoyer, S., Battaglia, P. W., Spergel, D. N., and Ho, S. Lagrangian neural networks. *CoRR*, abs/2003.04630, 2020. URL https://arxiv.org/abs/2003.04630.

Dalle, G., Baty, L., Bouvier, L., and Parmentier, A. Learning with combinatorial optimization layers: a probabilistic approach. *CoRR*, abs/2207.13513, 2022. doi: 10.48550/arXiv.2207.13513. URL https://doi.org/10.48550/arXiv.2207.13513.

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

Djolonga, J. and Krause, A. Differentiable learning of submodular models. In *Advances in Neural Information Processing Systems*, pp. 1013–1023, 2017.

Domke, J. Implicit differentiation by perturbation. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A. (eds.), *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pp. 523–531. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper/2010/hash/6ecbdd6ec859d284dc13885a37ce8d81-Abstract.html.

Dupont, E., Doucet, A., and Teh, Y. W. Augmented neural odes. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 3134–3144, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/21be9a4bd4f81549a9d1d241981cec3c-Abstract.html.

Elmachtoub, A. N. and Grigas, P. Smart "Predict, then Optimize". *Manag. Sci.*, 68(1):9–26, 2022. doi: 10.1287/mnsc.2020.3922. URL https://doi.org/10.1287/mnsc.2020.3922.

Ferber, A. M., Wilder, B., Dilkina, B., and Tambe, M. Mipaal: Mixed integer program as a layer. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI, New York, NY, USA, February 7-12, 2020*, pp. 1504–1511. AAAI Press, 2020. URL https://ojs.aaai.org/index.php/AAAI/article/view/5509.

Ferber, A. M., Huang, T., Zha, D., Schubert, M., Steiner, B., Dilkina, B., and Tian, Y. Surco: Learning linear surrogates for combinatorial nonlinear optimization problems. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10034–10052. PMLR, 2023. URL https://proceedings.mlr.press/v202/ferber23a.html.

Figueiredo, M. A. T., Nowak, R. D., and Wright, S. J. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *J. Sel. Topics Signal Processing*, 1(4):586–597, 2007. doi: 10.1109/JSTSP.2007.910281. URL https://doi.org/10.1109/JSTSP.2007.910281.

Frank, M. and Wolfe, P. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956. doi: https://doi.org/10.1002/nav.3800030109. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800030109.

Fredrikson, M., Lu, K., Vijayakumar, S., Jha, S., Ganesh, V., and Wang, Z. Learning modulo theories. *CoRR*, abs/2301.11435, 2023. doi: 10.48550/arXiv.2301.11435. URL https://arxiv.org/abs/2301.11435.

Fung, S. W., Heaton, H., Li, Q., McKenzie, D., Osher, S. J., and Yin, W. Fixed point networks: Implicit depth models with jacobian-free backprop. *CoRR*, abs/2103.12803, 2021. URL https://arxiv.org/abs/2103.12803.

Geng, Z., Zhang, X., Bai, S., Wang, Y., and Lin, Z. On training implicit models. *CoRR*, abs/2111.05177, 2021. URL https://arxiv.org/abs/2111.05177.

Ghaoui, L. E., Gu, F., Travacca, B., Askari, A., and Tsai, A. Y. Implicit deep learning. *SIAM J. Math. Data Sci.*, 3(3):930–958, 2021. doi: 10.1137/20M1358517. URL https://doi.org/10.1137/20M1358517.

Gould, S., Fernando, B., Cherian, A., Anderson, P., Cruz, R. S., and Guo, E. On differentiating parameterized argmin and argmax problems with application to bi-level optimization. *arXiv preprint arXiv:1607.05447*, 2016.

Gu, F., Chang, H., Zhu, W., Sojoudi, S., and Ghaoui, L. E. Implicit graph neural networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/8b5c8441a8ff8e151b191c53c1842a38-Abstract.html.

Güler, O. New proximal point algorithms for convex minimization. *SIAM J. Optim.*, 2(4):649–664, 1992. doi: 10.1137/0802032. URL https://doi.org/10.1137/0802032.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Kumar, V., Todorov, E., and Levine, S. Optimal control with learned local models: Application to dexterous manipulation. In Kragic, D., Bicchi, A., and Luca, A. D. (eds.), *2016 IEEE International Conference on Robotics and Automation, ICRA 2016, Stockholm, Sweden, May 16-21, 2016*, pp. 378–383. IEEE, 2016. doi: 10.1109/ICRA.2016.7487156. URL https://doi.org/10.1109/ICRA.2016.7487156.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Brodley, C. E. and Danyluk, A. P. (eds.), *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pp. 282–289. Morgan Kaufmann, 2001.

LeCun, Y. and Huang, F. J. Loss functions for discriminative training of energy-based models. In Cowell, R. G. and Ghahramani, Z. (eds.), *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005*. Society for Artificial Intelligence and Statistics, 2005. URL http://www.gatsby.ucl.ac.uk/aistats/fullpapers/207.pdf.

Lorberbom, G., Jaakkola, T. S., Gane, A., and Hazan, T. Direct optimization through arg max for discrete variational auto-encoder. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 6200–6211, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/1a04f965818a8533f5613003c7db243d-Abstract.html.

Mandi, J. and Guns, T. Interior point solving for LP-based prediction+optimisation. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7272–7282. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/51311013e51adebc3c34d2cc591fefee-Paper.pdf.

Mandi, J., Demirovic, E., Stuckey, P. J., and Guns, T. Smart predict-and-optimize for hard combinatorial optimization problems. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI, New York, NY, USA, February 7-12, 2020*, pp. 1603–1610. AAAI Press, 2020. doi: 10.1609/aaai.v34i02.5521. URL https://doi.org/10.1609/aaai.v34i02.5521.

McAllester, D. A., Hazan, T., and Keshet, J. Direct loss minimization for structured prediction. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A. (eds.), *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada*, pp. 1594–1602. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper/2010/hash/ca8155f4d27f205953f9d3d7974bdd70-Abstract.html.

Minervini, P., Franceschi, L., and Niepert, M. Adaptive perturbation-based gradient estimation for discrete latent variable models. In Williams, B., Chen, Y., and Neville, J. (eds.), *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, February 7-14, 2023*, pp. 9200–9208. AAAI Press, 2023. doi: 10.1609/aaai.v37i8.26103. URL https://doi.org/10.1609/aaai.v37i8.26103.

Moreau, J. J. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 255: 2897–2899, 1962. URL https://hal.science/hal-01867195.

Nesterov, Y. A method for unconstrained convex minimization problem with the rate of convergence o(1/k$\hat{2}$). *Doklady AN USSR*, 269:543–547, 1983. URL https://cir.nii.ac.jp/crid/1570572699326076416.

Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014. ISBN 1461346916.

Niculae, V., Martins, A. F. T., Blondel, M., and Cardie, C. Sparsemap: Differentiable sparse structured inference. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3796–3805. PMLR, 2018. URL http://proceedings.mlr.press/v80/niculae18a.html.

Niepert, M., Minervini, P., and Franceschi, L. Implicit MLE: backpropagating through discrete exponential family distributions. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 14567–14579, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/7a430339c10c642c4b2251756fd1b484-Abstract.html.

O'Donoghue, B., Chu, E., Parikh, N., and Boyd, S. P. Conic optimization via operator splitting and homogeneous self-dual embedding. *J. Optim. Theory Appl.*, 169 (3):1042–1068, 2016. doi: 10.1007/s10957-016-0892-3. URL https://doi.org/10.1007/s10957-016-0892-3.

Oyama, D. and Takenawa, T. On the (non-)differentiability of the optimal value function when the optimal solution is unique. *Journal of Mathematical Economics*, 76(C): 21–32, 2018. doi: 10.1016/j.jmateco.2018.02. URL https://ideas.repec.org/a/eee/mateco/v76y2018icp21-32.html.

Paden, B., Cáp, M., Yong, S. Z., Yershov, D. S., and Frazzoli, E. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Trans. Intell. Veh.*, 1(1):33–55, 2016. doi: 10.1109/TIV.2016.2578706. URL https://doi.org/10.1109/TIV.2016.2578706.

Parikh, N. and Boyd, S. P. Proximal algorithms. *Found. Trends Optim.*, 1(3):127–239, 2014. doi: 10.1561/2400000003. URL https://doi.org/10.1561/2400000003.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Paulus, A., Rolínek, M., Musil, V., Amos, B., and Martius, G. CombOptNet: Fit the right NP-hard problem by learning integer programming constraints. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8443–8453. PMLR, 2021. URL http://proceedings.mlr.press/v139/paulus21a.html.

Paulus, M. B., Choi, D., Tarlow, D., Krause, A., and Maddison, C. J. Gradient estimation with stochastic softmax tricks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3df80af53dce8435cf9ad6c3e7a403fd-Abstract.html.

Rockafellar, R. T. *Convex Analysis*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, 1970. ISBN 978-1-4008-7317-3.

Rockafellar, R. T. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5):877–898, 1976. doi: 10.1137/0314056. URL https://doi.org/10.1137/0314056.

Rockafellar, R. T. and Wets, R. J. *Variational Analysis*, volume 317 of *Grundlehren der mathematischen Wissenschaften*. Springer, 1998. ISBN 978-3-540-62772-2. doi: 10.1007/978-3-642-02431-3. URL https://doi.org/10.1007/978-3-642-02431-3.

Rolínek, M., Musil, V., Paulus, A., P., M. V., Michaelis, C., and Martius, G. Optimizing rank-based metrics with blackbox differentiation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pp. 7617–7627. Computer Vision Foundation / IEEE, 2020a. doi: 10.1109/CVPR42600.2020.00764. URL https://openaccess.thecvf.com/content_CVPR_2020/html/Rolinek_Optimizing_Rank-Based_Metrics_With_Blackbox_Differentiation_CVPR_2020_paper.html.

Rolínek, M., Swoboda, P., Zietlow, D., Paulus, A., Musil, V., and Martius, G. Deep graph matching via blackbox differentiation of combinatorial solvers. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J. (eds.), *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXVIII*, volume 12373 of *Lecture Notes in Computer Science*, pp. 407–424. Springer, 2020b. doi: 10.1007/978-3-030-58604-1_25. URL https://doi.org/10.1007/978-3-030-58604-1_25.

Sahoo, S., Paulus, A., Vlastelica, M., Musil, V., Kuleshov, V., and Martius, G. Backpropagation through combinatorial algorithms: Identity with projection works. In *Proceedings of the Eleventh International Conference on Learning Representations*, ICLR, May 2023. URL https://openreview.net/forum?id=JZMR727O29.

Song, Y., Schwing, A. G., Zemel, R. S., and Urtasun, R. Training deep neural networks via direct loss minimization. In Balcan, M. and Weinberger, K. Q. (eds.), *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 2169–2177. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/songb16.html.

Sun, H., Shi, Y., Wang, J., Tuan, H. D., Poor, H. V., and Tao, D. Alternating differentiation for optimization layers. In *The Eleventh International Conference on Learning Representations*, ICLR, 2023. URL https://openreview.net/forum?id=KKBMz-EL4tD.

Tseng, P. On accelerated proximal gradient methods for convex-concave optimization, 2008. URL https://www.mit.edu/~dimitrib/PTseng/papers/apgm.pdf.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6:1453–1484, 2005. URL http://jmlr.org/papers/v6/tsochantaridis05a.html.

Vlastelica, M., Paulus, A., Musil, V., Martius, G., and Rolínek, M. Differentiation of blackbox combinatorial solvers. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BkevoJSYPB.

Wang, P., Donti, P. L., Wilder, B., and Kolter, J. Z. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6545–6554. PMLR, 2019. URL http://proceedings.mlr.press/v97/wang19e.html.

Weaver, N. *Lipschitz Algebras*. World Scientific, second edition, 2018.

Wilder, B., Dilkina, B., and Tambe, M. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pp. 1658–1665. AAAI Press, 2019a. doi: 10.1609/aaai.v33i01.33011658. URL https://doi.org/10.1609/aaai.v33i01.33011658.

Wilder, B., Ewing, E., Dilkina, B., and Tambe, M. End to end learning and optimization on graphs. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 4674–4685, 2019b. URL https://proceedings.neurips.cc/paper/2019/hash/8bd39eae38511daad6152e84545e504d-Abstract.html.

Winston, E. and Kolter, J. Z. Monotone operator equilibrium networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/798d1c2813cbdf8bcdb388db0e32d496-Abstract.html.

Xu, M., Garg, S., Milford, M., and Gould, S. Deep declarative dynamic time warping for end-to-end learning of alignment paths. *CoRR*, abs/2303.10778, 2023. doi: 10.48550/arXiv.2303.10778. URL https://doi.org/10.48550/arXiv.2303.10778.

A. Limitations

We are aware of a few limitations of our method and are committed to transparent communication of them.

Reformulating the $\mathcal{L}$-proximal map as an instance of the forward solver makes the implementation simple and efficient. However, such reformulation is not possible in all cases since it requires access to the linear coefficients of the Lagrangian via the solver interface. This same issue arises for the augmentation introduced in Section 5.5, which requires accessing the quadratic coefficients of the solver and potentially turns an LP from the forward pass into a QP on the backward pass. This does not apply to our implementation of LPGD for the SCS solver (O'Donoghue et al., 2016) in CVXPY, as the solver natively supports quadratic conic programs.

When considering extremely small values of $\tau > 0$, the LPGD update requires solving the optimization problem very accurately, which can be expensive. Otherwise, warm-starting the solver on the backward pass with the forward pass solution will already satisfy the stopping criterion. However, we advocate for larger values of τ as the increased smoothing is usually beneficial. In our experiment, due to the convexity of the Lagrangian we are able to use the forward solver oracle to solve the optimization problems (2) and (30) to a very high accuracy in reasonable time using the SCS solver (O'Donoghue et al., 2016), and therefore did not observe issues arising from inexact solutions.

Finally, choosing the right combination of hyperparameters τ and ρ can potentially require expensive tuning, as the different terms in the objective of the $\mathcal{L}$-envelope (31) can be of different magnitudes, depending on the problem at hand. A potential remedy is to normalize the terms onto a shared scale, which makes well-performing values of τ and ρ more transferable and interpretable. Future work might also take inspiration from adaptive methods such as AIMLE (Minervini et al., 2023) for automatically tuning τ and ρ on the fly.

B. Extended Related Work

In this section we discuss the methods that are closest related to our framework.

Direct Loss Minimization. In *Direct Loss Minimization* (McAllester et al., 2010) the goal is to directly optimize a structured prediction pipeline for a given task loss such as the BLEAU score, extending previous work using structured SVMs or CRFs. Given an input $\mu \in \mathbb{R}^p$ the prediction pipeline consists of a feature map $\Psi\colon \mathcal{X} \times \mathbb{R}^p \to \mathbb{R}^k\colon (x, \mu) \mapsto \Psi(x, \mu)$ and a corresponding parameterized Lagrangian (called score function) $\mathcal{L}(x, w) = \langle w, \Psi(x, \mu)\rangle$. The structured prediction is then computed by solving the embedded optimization problem

$$x^*(w, \mu) \coloneqq \arg\max_{x \in \mathcal{X}} \langle w, \Psi(x, \mu)\rangle \tag{51}$$

over a finite set of solutions $\mathcal{X}$. Finally a task-specific loss $\ell\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is used to compare the prediction to a label $x_{\text{true}} \in \mathcal{X}$. The goal is to optimize the loss over a dataset $\{(\mu_i, x_{\text{true},i})\}_{i=1}^N$, and the authors propose to optimize it using gradient descent. They show for the case $\mathcal{X} = \{-1, 1\}$ that the gradient can be computed using the limit of the finite difference

$$\nabla_w \ell(x^*(w, \mu)) = \pm \lim_{\tau \to 0} \tfrac{1}{\tau}\left[\Psi(x_{\pm\tau}, \mu) - \Psi(x^*, \mu)\right]$$

with

$$x_{\pm\tau} \coloneqq \arg\max_{x \in \mathcal{X}} \langle w, \Psi(x, \mu)\rangle \pm \tau \ell(x, x_{\text{true}}). \tag{52}$$

Where the two sign cases are called the "away-from-worse" and the "towards-better" update. The authors also discuss using relaxations of $\mathcal{X}$ as well has hidden variables that have similarities to our dual variables. The method is applied to phoneme-to-speech alignment. The DLM framework has also been generalized to non-linear objective functions (Song et al., 2016; Lorberbom et al., 2019), always considering the limit $\tau \to 0$, with applications to action classification, object detection, and semi-supervised learning of structured variational autoencoders.

Blackbox Backpropagation. In *Blackbox Backpropagation* (Vlastelica et al., 2020) the authors consider embedded combinatorial optimization problems with linear cost functions. Given a (potentially high-dimensional) input $\mu \in \mathbb{R}^p$ the prediction pipeline first computes the cost vector $c \in \mathbb{R}^n$ using a backbone model $W\colon \mathbb{R}^p \times \Theta \to \mathbb{R}^n\colon (\mu, \theta) \mapsto c$ and then solves the embedded linear optimization problem over a combinatorial space $\mathcal{X} \subset \mathbb{R}^n$

$$x^*(c) \coloneqq \arg\min_{x \in \mathcal{X}} \langle x, c\rangle. \tag{53}$$

Finally the optimal solution is used as the prediction and compared to a label $x_{\text{true}} \in \mathcal{X}$ on a given loss $\ell\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Because of the discrete solution space the gradient of the loss with respect to the cost vector (and therefore also the parameters θ) is uninformative, as it is either zero or undefined. The authors propose to replace the uninformative gradient w.r.t. c with the gradient of a piecewise-affine loss interpolation

$$\widetilde{\ell}_\tau(c) \coloneqq \ell(x^*(c)) - \tfrac{1}{\tau} \min_{x \in \mathcal{X}} \langle c, x^*(c) - \widetilde{x}_\tau(c)\rangle \tag{54}$$

where we defined

$$\widetilde{x}_\tau(c) \coloneqq \arg\min_{x \in \mathcal{X}} \langle x, c + \tau \nabla_x \ell(x^*)\rangle. \tag{55}$$

It has the gradient

$$\nabla\widetilde{\ell}_\tau(c) = -\tfrac{1}{\tau}[x^* - x^*(c + \tau\nabla_x \ell(x^*))]. \quad (56)$$

This approach has also been extended to learning discrete distributions in (Niepert et al., 2021).

Implicit Differentiation by Perturbation. *Implicit Differentiation by Perturbation* (Domke, 2010) was proposed in the context of graphical models and marginal inference. The parameters c now correspond to the parameters of an exponential family distribution and $\mathcal{X}$ is the marginal polytope. Computing the marginals requires solving the entropy-regularized linear program

$$\arg\max_{x\in\mathcal{X}} \langle x, c\rangle + \Omega(x), \quad (57)$$

where Ω denotes the entropy. The final loss function ℓ depends on the computed marginals and some data x_{true}. Approximate marginal inference can be seen as approximating the marginal polytope with a set of linear equality constraints

$$x^*(c) := \arg\max_{x\in\mathbb{R}^n, Ax=b} \langle x, c\rangle + \Omega(x), \quad (58)$$

which can be solved by loopy/tree-reweighted belief propagation. The authors show that in this case, as an alternative to implicit differentiation, the gradients of the loss with respect to the parameters can be computed by

$$\nabla_w \ell(x^*(w)) = \lim_{\tau\to 0} \tfrac{1}{\tau}[x^*(c+\tau\nabla_x\ell(x^*)) - x^*(c)]. \quad (59)$$

The method is applied to binary denoising, where the authors also test the double-sided perturbation case.

Identity with Projection. In *Identity with Projection* (Sahoo et al., 2023) the setup is the same as in Blackbox Backpropagation. The goal of this method is to speed up the backward pass computation by removing the second invocation of the solver oracle on the backward pass. In the basic version, the authors propose to replace the uninformative gradient through the solver $\nabla_c \ell(x^*(c))$ by simply treating the solver as a negative identity, and returning $\Delta_{\text{Id}} := -\nabla\ell(x^*)$ instead. Connections are drawn to the straight-through estimator. Further, the authors identify transformations of the cost vector $P\colon \mathbb{R}^k \to \mathbb{R}^k$ that leave the optimal solution unchanged (e.g. normalization), i.e. $x^*(P(c)) = x^*(c)$. They propose to refine the vanilla identity method by differentiating through the transformation, yielding the update $\Delta_{\text{Id}} := P'_{x^*}(-\nabla\ell(x^*))$. This is also shown to have an interpretation as differentiating through a projection onto a relaxation $\widetilde{\mathcal{X}}$ of the feasible space, i.e.

$$\Delta_{\text{Id}} = D^* P_{\widetilde{\mathcal{X}}}(x^*)(-\nabla\ell(x^*)). \quad (60)$$

Experimentally, the method is shown to be competitive with Blackbox Backpropagation and I-MLE.

Smart Predict then Optimize. In *Smart Predict then Optimize* (Elmachtoub & Grigas, 2022) setting the authors consider a linear program

$$x^*(c) := \arg\min_{x\in\mathcal{X}} \langle x, c\rangle \quad (61)$$

as the final component of a prediction pipeline. The cost parameters $c \in \mathbb{R}^n$ are not known with certainty at test time, and are instead predicted from an input $\mu \in \mathbb{R}^p$ via a prediction model $W_\theta\colon \mathbb{R}^p \to \mathbb{R}^n\colon \mu \mapsto c$ with parameters $\theta \in \Theta$. What distinguishes this setup from the one considered in e.g. Blackbox Backpropagation is that during training the true cost vectors c_{true} are available, i.e. there is a dataset $\{\mu_i, c_{\text{true},i}\}_{i=1}^N$. A naive approach would be to directly regress the prediction model onto the true cost vectors by minimizing the mean squared error

$$\ell_{\text{MSE}}(\mu, c_{\text{true}}) = \tfrac{1}{2}\|W_\theta(\mu) - c_{\text{true}}\|. \quad (62)$$

However, the authors note that this ignores the actual downstream performance metric, which is the regret or SPO loss

$$\ell_{\text{SPO}}(x^*(c), c_{\text{true}}) = \langle x^*(c) - x^*(c_{\text{true}}), c_{\text{true}}\rangle. \quad (63)$$

Unfortunately this loss does not have informative gradients, so the authors propose to instead optimize a convex upper bound, the SPO+ loss

$$\begin{aligned}\ell_{\text{SPO+}}(c, c_{\text{true}}) := \sup_{x\in\mathcal{X}} \langle x, c_{\text{true}} - 2c\rangle \\ + 2\langle x^*(c_{\text{true}}), c\rangle - \langle x^*(c_{\text{true}}), c_{\text{true}}\rangle\end{aligned} \quad (64)$$

a generalization of the hinge loss. Experimentally, results on shortest path problems and portfolio optimization demonstrate that optimizing the SPO+ loss offers significant benefits over optimizing the naive MSE loss.

Fenchel-Young Losses. *Fenchel Young Losses* (Blondel et al., 2020) were proposed in the context of structured prediction, in which the final prediction is the solution of a regularized linear program

$$x^*(c) := \arg\max_{x\in\mathcal{X}} \langle x, c\rangle - \Omega(x). \quad (65)$$

Supervision is assumed in the form of ground truth labels x_{true}. The authors propose to learn the parameters c by minimizing the convex Fenchel Young Loss

$$\begin{aligned}\ell_{\text{FY}}(c, x_{\text{true}}) &:= \max_{x\in\mathcal{X}} \big[\langle c, x\rangle - \Omega(x)\big] + \Omega(x_{\text{true}}) - \langle c, x_{\text{true}}\rangle \\ &= \langle c, x_{\text{true}}\rangle + \Omega(x_{\text{true}}) - \min_{x\in\mathcal{X}} \big[\langle c, x\rangle + \Omega(x)\big],\end{aligned}$$

for which a variety of appealing theoretical results are presented. Many known loss functions from structured prediction and probabilistic prediction are shown to be recovered by Fenchel-Young losses for specific choices of feasible region $\mathcal{X}$ and regularizer Ω, including the *structured hinge* (Tsochantaridis et al., 2005), *CRF* (Lafferty et al., 2001), and *SparseMAP* (Niculae et al., 2018) losses.

C. Implicit Differentiation with Augmentation

We inspect how the augmentation in (28) affects existing methods for computing the adjoint derivative of the augmented optimization problem

$$z^*_\rho(w) = (x^*_\rho(w), y^*_\rho(w)) \coloneqq \arg\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}} \mathcal{L}_\rho(x,y,w).$$

Quadratic Program. For a symmetric positive semi-definite matrix H we can write a quadratic program with inequality constraints as

$$\begin{aligned}(x^*, s^*) = &\arg\min_{x,s\geq 0} \tfrac{1}{2}x^T Hx + c^T y \\ &\text{subject to} \quad Ax + b + s = 0.\end{aligned} \tag{66}$$

In Lagrangian form, we can write it as

$$z^* = (x^*, s^*, y^*) = \arg\min_{x,s\geq 0}\max_y \mathcal{L}(H,c,A,b,x,y)$$

with the Lagrangian

$$\mathcal{L}(x,s,y,H,A,b,c) = \tfrac{1}{2}x^T Hx + c^T x + (Ax+b+s)^T y.$$

The augmentation in (28) augments the Lagrangian to

$$\begin{aligned}&\mathcal{L}_\rho(x,s,y,H,A,b,c)\\ &\quad= \tfrac{1}{2}x^T Hx + c^T x + (Ax+b+s)^T y + \tfrac{1}{2\rho}\|x - x^*\|_2^2\end{aligned}$$

and we write

$$z^*_\rho(H,A,b,c) = \arg\min_{x,s\geq 0}\max_y \mathcal{L}_\rho(x,s,y,H,A,b,c).$$

As described in (Amos & Kolter, 2017a), the optimization problem can be differentiated by treating it as an implicit layer via the KKT optimality conditions, which are given as

$$Hx + A^T y + c + \tfrac{1}{\rho}(x - x^*) = 0 \tag{67}$$
$$\operatorname{diag}(y)s = 0 \tag{68}$$
$$Ax + b + s = 0 \tag{69}$$
$$s \geq 0. \tag{70}$$

Assuming strict complementary slackness renders the inequality redundant and the conditions reduce to the set of equations

$$\begin{aligned}0 &= F_\rho(x,s,y,H,A,b,c)\\ &= \begin{pmatrix} Hx + A^T y + c + \frac{1}{\rho}(x - x^*) \\ \operatorname{diag}(y)s \\ Ax + b + s\end{pmatrix}\end{aligned} \tag{71}$$

which admits the use of the implicit function theorem. It states that under the regularity condition that $\partial F_\rho/\partial z$ is invertible, $z^*_\rho(w)$ can be expressed as an implicit function, and we can compute its Jacobian by linearizing the optimality conditions around the current solution

$$0 = \frac{\partial F_\rho}{\partial z}\frac{\partial z^*_\rho}{\partial w} + \frac{\partial F_\rho}{\partial w} \tag{72}$$

with

$$\frac{\partial F_\rho}{\partial z} = \begin{bmatrix} H + \frac{1}{\rho}I & 0 & A^T \\ 0 & \operatorname{diag}(\frac{y}{s}) & I \\ A & I & 0\end{bmatrix}. \tag{73}$$

It is now possible to compute the desired Vector-Jacobian-product as

$$\begin{aligned}\nabla_w \ell(x^*(w)) &= \frac{\partial z^*}{\partial w}^T \nabla\ell(x^*)\\ &= -\frac{\partial F_\rho}{\partial w}^T \frac{\partial F_\rho}{\partial z}^{-T} \nabla\ell(x^*),\end{aligned} \tag{74}$$

which involves solving a linear system. The augmentation term, therefore, serves as a regularizer for this linear system.

Conic Program. A conic program (Boyd & Vandenberghe, 2014) is defined as

$$(x^*, s^*) = \arg\min_{x,s\in\mathcal{K}} c^T y \quad \text{subject to} \quad Ax + b + s = 0$$

where $\mathcal{K}$ is a cone. The Lagrangian of this optimization problem is

$$\mathcal{L}(x,s,y,A,b,c) = c^T x + (Ax+b+s)^T y \tag{75}$$

which allows an equivalent saddle point formulation given by

$$z^* = (x^*, s^*, y^*) = \arg\min_{x,s\in\mathcal{K}}\max_y \mathcal{L}(x,s,y,A,b,c).$$

The KKT optimality conditions are

$$A^T y + c = 0, \tag{76}$$
$$Ax + s + b = 0, \tag{77}$$
$$(s,y) \in \mathcal{K}\times\mathcal{K}^*, \tag{78}$$
$$s^T y = 0, \tag{79}$$

where $\mathcal{K}^*$ is the dual cone of $\mathcal{K}$. The skew-symmetric mapping

$$Q(A,b,c) = \begin{bmatrix} 0 & A^T & c \\ -A & 0 & b \\ -c^T & -b^T & 0\end{bmatrix} \tag{80}$$

is used in the homogenous self-dual embedding (O'Donoghue et al., 2016; Busseti et al., 2019), a feasibility problem that embeds the conic optimization problem. Agrawal et al. (2019b) solve and differentiate the self-dual embedding. We use the CVXPY implementation of this method as our baseline for computing the true adjoint derivatives of the optimization problem. The augmentation in (28) changes the stationarity condition and, thereby, the skew-symmetric mapping as

$$Q_\rho(A,b,c) = \begin{bmatrix} \frac{1}{\rho}I & A^T & c \\ -A & 0 & b \\ -c^T & -b^T & 0 \end{bmatrix}. \tag{81}$$

We adjust the CVXPY implementation accordingly for our experiments.

D. Relation to Mirror Descent

Standard Mirror Descent. Classical mirror descent is an algorithm for minimizing a function $\ell(x)$ over a closed convex set $\mathcal{X} \subseteq \mathbb{R}^n$. The algorithm is defined by the distance-generating function (or mirror map) $\phi\colon \mathbb{R}^n \to \mathbb{R}$, a strictly convex continuously differentiable function. The mirror descent algorithm also requires the assumption that the dual space of ϕ is all of $\mathbb{R}^n$, i.e. $\{\nabla\phi(x) \mid x \in \mathbb{R}^n\} = \mathbb{R}^n$, and that the gradient of ϕ diverges as $\|x\|_2 \to \infty$.

The Bregman divergence of the mirror map is defined as

$$D_\phi(x, \widehat{x}) := \phi(x) - \phi(\widehat{x}) - \langle x - \widehat{x}, \nabla\phi(\widehat{x})\rangle. \tag{82}$$

The lower[15] *Bregman-Moreau envelope* (Bauschke et al., 2018) $\mathrm{env}^\phi_{\tau f}\colon \mathbb{R}^n \to \mathbb{R}$ of a possibly non-smooth function $f\colon \mathbb{R}^n \to \mathbb{R}$ is defined for $\tau > 0$ as

$$\mathrm{env}^\phi_{\tau f}(\widehat{x}) := \min_x f(x) + \tfrac{1}{\tau} D_\phi(x, \widehat{x}). \tag{83}$$

The corresponding lower *Bregman-Moreau proximal map* $\mathrm{prox}^\phi_{\tau f}\colon \mathbb{R}^n \to \mathbb{R}^n$ is given by

$$\mathrm{prox}^\phi_{\tau f}(\widehat{x}) := \arg\inf_x f(x) + \tfrac{1}{\tau} D_\phi(x, \widehat{x}) \tag{84}$$

The superscript ϕ distinguishes the notation from the standard Moreau envelope (4) and proximal map (5).

Then, the mirror descent algorithm in proximal form is given by iteratively applying the Bregman-Moreau proximal map of $\tilde{\ell} + I_\mathcal{X}$ as

$$\begin{aligned} x_{k+1} &= \arg\min_x \tilde{\ell}(x) + I_\mathcal{X}(x) + \tfrac{1}{\tau} D_\phi(x, x_k) && (85)\\ &= \arg\min_{x\in\mathcal{X}} \langle x, \nabla\ell(x_k)\rangle + \tfrac{1}{\tau} D_\phi(x, x_k). && (86) \end{aligned}$$

Lagrangian Mirror Descent. In this section, we derive *Lagrangian Mirror Descent* (LMD), an alternative algorithm to LPGD inspired by mirror descent. We define

$$\mathcal{L}_w(x) := \sup_{y\in\mathcal{Y}} \mathcal{L}(x, y, w) \tag{87}$$

and assume that $\mathcal{L}_w$ is strongly convex and continuously differentiable in x. As the key step, we identify the distance-generating function (mirror map) as the Lagrangian, i.e. $\phi = \mathcal{L}_w$. This has the interpretation that distances are measured in terms of the Lagrangian, similar to the intuition behind the previously defined Lagrangian divergence.

This mirror map leads to the Bregman divergence

$$D_{\mathcal{L}_w}(x, \widehat{x}) = \mathcal{L}_w(x) - \mathcal{L}_w(\widehat{x}) - \langle x - \widehat{x}, \nabla\mathcal{L}_w(\widehat{x})\rangle \tag{88}$$

satisfying

$$\begin{aligned} &D_{\mathcal{L}_w}(x, x^*) \geq 0, && (89)\\ &D_{\mathcal{L}_w}(x, x^*) = 0 \Leftrightarrow x = x^*(w) \quad \text{for } x \in \mathcal{X}. && (90) \end{aligned}$$

We define the lower *Lagrange-Bregman-Moreau envelope* at w as the Bregman-Moreau envelope at $x^* = x^*(w)$, i.e.

$$\begin{aligned} \ell^\phi_\tau(w) &:= \mathrm{env}^{\mathcal{L}_w}_{\tau\ell + I_\mathcal{X}}(x^*) && (91)\\ &= \min_x \ell(x) + I_\mathcal{X}(x) + \tfrac{1}{\tau} D_{\mathcal{L}_w}(x, x^*) && (92)\\ &= \min_{x\in\mathcal{X}} \ell(x) + \tfrac{1}{\tau}(\mathcal{L}(x, w) - \mathcal{L}^*(w) \\ &\qquad - \langle x - x^*, \nabla_x\mathcal{L}(x^*, w)\rangle), && (93) \end{aligned}$$

and the corresponding lower *Lagrange-Bregman-Moreau proximal map*

$$\begin{aligned} x^\phi_\tau(w) &:= \mathrm{prox}^{\mathcal{L}_w}_{\tau\ell + I_\mathcal{X}}(x^*) && (94)\\ &= \arg\min_x \ell(x) + I_\mathcal{X}(x) + \tfrac{1}{\tau} D_{\mathcal{L}_w}(x, x^*) && (95)\\ &= \arg\min_{x\in\mathcal{X}} \ell(x) + \tfrac{1}{\tau}(\mathcal{L}(x, w) - \mathcal{L}^*(w) \\ &\qquad - \langle x - x^*, \nabla_x\mathcal{L}(x^*, w)\rangle). && (96) \end{aligned}$$

Again, the superscript ϕ distinguishes the notation from the lower Lagrange-Moreau envelope (12) and lower $\mathcal{L}$-proximal map (13).

Similar to how we defined LPGD as gradient descent on the Lagrange-Moreau envelope of the linearized loss, we define *Lagrangian Mirror Descent* (LMD) as gradient descent on the Lagrange-Bregman-Moreau envelope of the linearized loss, i.e.

$$\begin{aligned} \nabla_w \widetilde{\ell}^\phi_\tau(w) = \tfrac{1}{\tau}\nabla_w[&\mathcal{L}(\widetilde{x}^\phi_\tau, w) - \mathcal{L}(x^*, w) \\ &- \langle \widetilde{x}^\phi_\tau - x^*, \nabla_x\mathcal{L}(x^*, w)\rangle]. \end{aligned} \tag{97}$$

Again, the approximation allows efficiently computing the gradients using the forward solver as

$$\begin{aligned} \widetilde{x}^\phi_\tau(u, v) &= \arg\min_{x\in\mathcal{X}} \tau\langle x, \nabla\ell\rangle + \langle x, u\rangle \\ &\qquad + \Omega(x, y, v) - \langle x, \nabla_x\mathcal{L}(x^*, w)\rangle && (98)\\ &= x^*(u + \tau\nabla\ell - \nabla_x\mathcal{L}(x^*, w), v). && (99) \end{aligned}$$

[15] The terms lower and upper are replaced with left and right in Bauschke et al. (2018).

If $\mathcal{X} = \mathbb{R}^n$, then the original optimization problem (2) is unconstrained and we have the optimality condition $\nabla_x \mathcal{L}(x^*, w) = 0$. Therefore, the Bregman divergence

$$\begin{aligned} D_{\mathcal{L}_w}(x, x^*) &= \mathcal{L}_w(x) - \mathcal{L}_w(x^*) - \langle x - x^*, \nabla \mathcal{L}_w(x^*)\rangle \\ &= \mathcal{L}(x, w) - \mathcal{L}(x^*, w) \\ &= \mathcal{L}(x, w) - \mathcal{L}^*(w) = D_{\mathcal{L}}(x|w) \end{aligned}$$

coincides with the Lagrangian divergence (9). It follows that, in this case, the Lagrange-Moreau envelope and LPGD coincide with the Lagrange-Bregman-Moreau envelope and LMD, respectively.

E. Extension to General Loss Functions

Loss on Dual Variables. In the main text we considered losses depending only on the primal variables, i.e. $\ell(x)$. For a loss on the dual variables $\ell(y)$, if we assume strong duality of (1), we can reduce the situation to the primal case, as

$$y^*(w) = \arg\max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \mathcal{L}(x, y, w) \tag{100}$$

$$= \arg\min_{y\in\mathcal{Y}} \max_{x\in\mathcal{X}} -\mathcal{L}(x, y, w). \tag{101}$$

This amounts to simply negating the Lagrangian in all equations while swapping x and y.

Loss on Primal and Dual Variables. The situation becomes more involved for a loss function $L(x, y)$ depending on both primal and dual variables. If it decomposes into a primal and dual component, i.e. $L(x, y) = \ell_p(x) + \ell_d(y)$, we can compute the envelopes of the individual losses independently. Note that a linearization of the loss $\tilde{L}$ trivially decomposes this way. Adding the envelopes together yields a combined lower and upper envelope for the total loss as

$$\begin{aligned} (\ell_p)_\tau(w) + (\ell_d)_\tau(w) =& \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_p(x)] \\ &- \max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_d(y)], \end{aligned} \tag{102}$$

$$\begin{aligned} (\ell_d)^\tau(w) + (\ell_p)^\tau(w) =& \max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_d(y)] \\ &- \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_p(x)]. \end{aligned} \tag{103}$$

Assuming strong duality of these optimization problems, this leads to

$$\begin{aligned} (\ell_p)_\tau(w) + (\ell_d)_\tau(w) =& \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_p(x)] \\ &- \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_d(y)], \\ (\ell_d)^\tau(w) + (\ell_p)^\tau(w) =& \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_d(y)] \\ &- \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_p(x)]. \end{aligned}$$

Strong duality holds in particular for a linearized loss $\tilde{L}$, as this only amounts to a linear perturbation of the original optimization problem. Unfortunately, computing the average envelope

$$\begin{aligned} (\ell_p)_{\tau}(w) + (\ell_d)_{\tau}(w) = \tfrac{1}{2}\Big\{ & \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_p(x)] \\ & - \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_d(y)] \\ & + \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_d(y)] \\ & - \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_p(x)] \Big\} \end{aligned}$$

or its gradient would now require four evaluations of the solver, which can be expensive. To reduce the number of evaluations, we instead "combine" the perturbations of the primal and dual loss, i.e. we define

$$L_\tau(w) \coloneqq \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} L(x, y) + \tfrac{1}{\tau}[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)] \tag{104}$$

$$L^\tau(w) \coloneqq \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} L(x, y) + \tfrac{1}{\tau}[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)] \tag{105}$$

$$\begin{aligned} L_{\tau}(w) &\coloneqq \tfrac{1}{2}\{L_\tau(w) + L^\tau(w)\} \\ &= \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + L(x, y)] \\ &\quad - \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - L(x, y)]. \end{aligned} \tag{106}$$

The above definitions are valid even when we do not have strong duality of (1) and (104). Note that L_τ and L^τ are not necessarily lower and upper bounds of the loss anymore. However, these combined envelopes also apply to loss functions that do not separate into primal and dual variables, and computing their gradients requires fewer additional solver evaluations. For $L(x, y) = \ell(x)$ we have

$$L_\tau(w) = \ell_\tau(w), \quad L_{\tau}(w) = \ell_{\tau}(w), \quad L^\tau(w) = \ell^\tau(w). \tag{107}$$

In some of the proofs we will work with L instead of ℓ for full generality and reduce the situation to a primal loss ℓ with the relations above.

F. Proofs

Lemma F.1 (Equation (11)). *It holds that*

$$D^*_{\mathcal{L}}(x|w) = 0 \text{ if and only if } x \text{ minimizes (2) for } x \in \mathcal{X}.$$

Proof of Lemma F.1. If $x \in \mathcal{X}$ minimizes (2), this means

$$\sup_{y\in\mathcal{Y}} \mathcal{L}(x, y, w) = \inf_{\tilde{x}\in\mathcal{X}} \sup_{y\in\mathcal{Y}} \mathcal{L}(\tilde{x}, y, w) \tag{108}$$

and therefore

$$D^*_{\mathcal{L}}(x|w) = \sup_{y\in\mathcal{Y}} \big[\mathcal{L}(x,y,w) - \mathcal{L}^*(w)\big] \tag{109}$$

$$= \min_{\widetilde{x}\in\mathcal{X}} \max_{y\in\mathcal{Y}} \big[\mathcal{L}(\widetilde{x},y,w)\big] - \mathcal{L}^*(w) = 0. \tag{110}$$

If $D^*_{\mathcal{L}}(x|w) = 0$, we have from the definition of the Lagrangian divergence (9) that

$$\sup_{y\in\mathcal{Y}} \mathcal{L}(x,y,w) = \mathcal{L}^*(w) \tag{111}$$

$$= \min_{\widetilde{x}\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathcal{L}(\widetilde{x},y,w) \tag{112}$$

and hence x is a minimizer of (2). □

Lemma F.2 (Equation (17))**.** *Assume that $\mathcal{L}, \ell \in \mathcal{C}^1$ and assume that the solution mappings of optimization (2, 12, 14) admit continuous selections $x^*(w), x_\tau(w), x^\tau(w)$ at w. Then*

$$\nabla \ell_\tau(w) = \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}(w,z_\tau) - \mathcal{L}(z^*,w)\big], \tag{113}$$

$$\nabla \ell^\tau(w) = \tfrac{1}{\tau}\nabla_w\big[\mathcal{L}(z^*,w) - \mathcal{L}(z^\tau,w)\big]. \tag{114}$$

Proof of Lemma F.2. We assumed that $z^*(w)$ is a selection of the solution set continuous at w. We also assumed that $\mathcal{L}$ and ℓ are continuously differentiable. It then follows that

$$\nabla_w \ell_\tau(w) = \nabla_w \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \ell(x) + \tfrac{1}{\tau} D_{\mathcal{L}}(x,y|w) \tag{115}$$

$$\begin{aligned} &= \nabla_w \Big[\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} [\ell(x) + \tfrac{1}{\tau}\mathcal{L}(x,y,w)] \\ &\quad - \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \tfrac{1}{\tau}\mathcal{L}(x,y,w)\Big] \end{aligned} \tag{116}$$

$$\begin{aligned} &= \nabla_w \Big[\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \ell(x) + \tfrac{1}{\tau}\mathcal{L}(x,y,w)\Big] \\ &\quad - \nabla_w \Big[\min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \tfrac{1}{\tau}\mathcal{L}(x,y,w)\Big] \end{aligned} \tag{117}$$

$$\begin{aligned} &= \nabla_w \big[\ell(x_\tau) + \tfrac{1}{\tau}\mathcal{L}(x_\tau, y_\tau, w)\big] \\ &\quad - \nabla_w \big[\tfrac{1}{\tau}\mathcal{L}(x^*, y^*, w)\big] \end{aligned} \tag{118}$$

$$= \tfrac{1}{\tau}\nabla_w \big[\mathcal{L}(w,z_\tau) - \mathcal{L}(z^*,w)\big] \tag{119}$$

In the fourth equation we used the result by Oyama & Takenawa (2018, Proposition 4.1). The proof for the upper envelope is analogous. □

Proposition 6.1. *Assume that $\mathcal{L}$ is L-Lipschitz continuous in w. Then $\ell_\tau, \ell\tau, \ell^\tau$ are $\frac{2L}{\tau}$-Lipschitz continuous in w.*

Proof of Proposition 6.1. We have the optimal Lagrangian

$$\mathcal{L}^*(w) = \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathcal{L}(x,y,w) \tag{120}$$

and define

$$\mathcal{L}^*_\tau(w) := \min_{x\in\mathcal{X}} \max_{y\in\mathcal{Y}} \mathcal{L}(x,y,w) + \tau\ell(x). \tag{121}$$

Then the lower $\mathcal{L}$-envelope may be written as

$$\ell_\tau(w) = \tfrac{1}{\tau}\big[\mathcal{L}^*_\tau(w) - \mathcal{L}^*(w)\big]. \tag{122}$$

If $\mathcal{L}$ is L-Lipschitz in w for every $x \in \mathcal{X}, y \in \mathcal{Y}$, where L is independent of x and y, then both $\mathcal{L}^*$ and $\mathcal{L}^*_\tau$ are L-Lipschitz, see Weaver (2018, Proposition 1.32). Hence, ℓ_τ is $\frac{2L}{\tau}$-Lipschitz. The cases of $\ell\tau$ and ℓ^τ are analogous. □

Proposition 6.2. *Assume $\mathcal{L}, \ell$ are lower semi-continuous and ℓ is finite-valued on $\mathcal{X}$. Let w be a parameter for which*

$$X^*(w) := \{x \in \mathcal{X} \mid \mathcal{D}^*_{\mathcal{L}}(x|w) = 0\} \tag{123}$$

is nonempty. Then it holds that

$$\lim_{\tau\to 0} \ell_\tau(w) = \min_{x^*\in X^*(w)} \ell(x^*) \tag{124}$$

and

$$\lim_{\tau\to 0} x^*_\tau(w) \in \arg\min_{x\in X^*(w)} \ell(x^*) \tag{125}$$

whenever the limit exists.

Proof of Proposition 6.2. Throughout the proof, let w fixed, and hence we omit the dependence in the notation. For $\tau > 0$ we define $f_\tau \colon \mathcal{X} \to \mathbb{R}$ by

$$f_\tau(x) = \ell(x) + \tfrac{1}{\tau} D^*_{\mathcal{L}}(x) \quad \text{for } x \in \mathcal{X}. \tag{126}$$

We have the monotonicity

$$f_\tau \le f_{\tau'} \quad \text{whenever} \quad \tau' \le \tau, \tag{127}$$

since $D^*_{\mathcal{L}}(x) \ge 0$, and taking the infimum over $x \in \mathcal{X}$ on both sides leads to

$$\ell_\tau \le \ell_{\tau'} \quad \text{whenever} \quad \tau' \le \tau. \tag{128}$$

Moreover, for any $\tau > 0$ and all $x^* \in X^*$, it is

$$\ell_\tau = \min_{x\in\mathcal{X}} f_\tau(x) \le f_\tau(x^*) = \ell(x^*). \tag{129}$$

Therefore ℓ_τ is bounded and monotone, hence the limit as $\tau \to 0^+$ exists and

$$\ell_0 := \lim_{\tau\to 0^+} \ell_\tau \le \inf_{x^*\in X^*} \ell(x^*). \tag{130}$$

Denote by $(x_\tau)_{\tau>0}$ minimizers of ℓ_τ, i.e. $\ell_\tau = f_\tau(x_\tau)$. Assume that $x_\tau \to x_0 \in \mathcal{X}$ as $\tau \to 0^+$. We show that x_0 minimizes $\ell(x)$ over X^*.

Since ℓ is lower semi-continuous, we have that

$$\liminf_{\tau\to 0^+} \ell(x_\tau) \ge \ell(x_0) \tag{131}$$

and, by definition, we also have that

$$\ell_\tau = \ell(x_\tau) + \tfrac{1}{\tau} D^*_{\mathcal{L}}(x_\tau). \tag{132}$$

Taking the $\liminf$ as $\tau \to 0^+$ in (132) yields

$$\begin{aligned} \ell_0 &= \liminf_{\tau\to 0^+} \ell_\tau \\ &= \liminf_{\tau\to 0^+} \left[\ell(x_\tau) + \tfrac{1}{\tau} D^*_{\mathcal{L}}(x_\tau)\right] \\ &\geq \liminf_{\tau\to 0^+} \ell(x_\tau) + \liminf_{\tau\to 0^+} \tfrac{1}{\tau} D^*_{\mathcal{L}}(x_\tau) \\ &\geq \ell(x_0) + \liminf_{\tau\to 0^+} \tfrac{1}{\tau} D^*_{\mathcal{L}}(x_\tau). \end{aligned} \tag{133}$$

Next, since $D^*_{\mathcal{L}}(x) \geq 0$, inequality (133) reduces to $\ell_0 \geq \ell(x_0)$, which together with estimate (130) gives

$$\ell(x_0) \leq \inf_{x^* \in X^*} \ell(x^*). \tag{134}$$

It remains to show that $x_0 \in X^*$. From (133) we know

$$\liminf_{\tau\to 0^+} \tfrac{1}{\tau} D^*_{\mathcal{L}}(x_\tau) < \infty. \tag{135}$$

We assume lower semi-continuity of $\mathcal{L}(x, y)$ in x for every $y \in \mathcal{Y}$, therefore the Lagrangian divergence

$$D^*_{\mathcal{L}}(x) = \sup_{y\in\mathcal{Y}} \mathcal{L}(x, y) - \mathcal{L}^* \tag{136}$$

is also lower semi-continuous, hence

$$\liminf_{\tau\to 0^+} D^*_{\mathcal{L}}(x_\tau) \geq D^*_{\mathcal{L}}(x_0). \tag{137}$$

Together with (135), this gives that $D^*_{\mathcal{L}}(x_0) = 0$, or equivalently, $x_0 \in X^*$. $\square$

Theorem 6.3. *Assume that $\mathcal{L} \in \mathcal{C}^2$ and assume that the solution mapping of optimization (2) admits a differentiable selection $x^*(w)$ at w. Then*

$$\lim_{\tau\to 0} \nabla\widetilde{\ell}_\tau(w) = \nabla_w \ell(x^*(w)) = \lim_{\tau\to 0} \nabla\widetilde{\ell}^\tau(w). \tag{138}$$

Proof of Theorem 6.3. In this proof we work in the more general setup described in Appendix E, in which the loss L can depend on both primal and dual optimal solutions. We aim to show that for a linear loss approximation $\widetilde{L}$, the LPGD update recovers the true gradient as τ approaches zero. We again assume the same form of the Lagrangian as in (21)

$$\mathcal{L}(z, w) = \langle z, u\rangle + \Omega(z, v) \tag{139}$$

with $w = (u, v)$ and get from Oyama & Takenawa (2018, Proposition 4.1)

$$\nabla_u \mathcal{L}^*(u, v) = \nabla_u \mathcal{L}(z^*, u, v) = z^*(u, v). \tag{140}$$

We define

$$\mathrm{d}w := \begin{pmatrix} \nabla_z L \\ 0 \end{pmatrix}. \tag{141}$$

Then it holds that

$$\begin{aligned} &\lim_{\tau\to 0} \nabla_w \widetilde{L}_\tau(w) \\ &\quad= \lim_{\tau\to 0} \tfrac{1}{\tau}[\nabla_w \mathcal{L}^*(w + \tau \mathrm{d}w) - \nabla_w \mathcal{L}^*(w)] && (142) \\ &\quad= \frac{\partial^2 \mathcal{L}^*}{\partial^2 w} \mathrm{d}w = \frac{\partial^2 \mathcal{L}^*}{\partial^2 w}^T \mathrm{d}w && (143) \\ &\quad= \frac{\partial^2 \mathcal{L}^*}{\partial^2 w}^T \begin{pmatrix} \nabla_z L \\ 0 \end{pmatrix} && (144) \\ &\quad= \frac{\partial^2 \mathcal{L}^*}{\partial w \partial u}^T \nabla_z L = \frac{\partial(\nabla_u \mathcal{L}^*)}{\partial w}^T \nabla_z L && (145) \\ &\quad= \frac{\partial z^*}{\partial w}^T \nabla_z L = \nabla_w L(z^*(w)). && (146) \end{aligned}$$

The main step in this derivation appears in the second-to-last equality by identifying the Jacobian of the solution mapping as a sub-matrix of the Hessian of the optimal Lagrangian function, which is a symmetric matrix under the conditions of Schwarz's theorem. A sufficient condition for this is the assumption that $\mathcal{L} \in \mathcal{C}^2$.[16] Exploiting the symmetry of the Hessian then allows computing the gradient, which is a co-derivative (backward-mode, vector-jacobian-product), as the limit of a finite-difference between two solver outputs, which usually only computes a derivative (forward-mode, jacobian-vector-product) from input perturbations Δw as

$$\begin{aligned} \Delta z &= \frac{\partial z^*}{\partial w} \Delta w && (147) \\ &= \lim_{\tau\to 0} \tfrac{1}{\tau}[z^*(w + \tau\Delta w) - z^*(w)] && (148) \\ &= \lim_{\tau\to 0} \tfrac{1}{\tau}[\nabla_u \mathcal{L}^*(w + \tau\Delta w) - \nabla_u \mathcal{L}^*(w)]. && (149) \end{aligned}$$

We observe that the finite-difference appearing in the LPGD update is the co-derivative counterpart of a finite-difference approximation of the derivative. This observation also fosters an interpretation of finite τ in the LPGD update: In forward-mode, checking how the solver reacts to finite perturbation of the parameters intuitively provides higher-order information than linear sensitivities to infinitesimal perturbations via derivatives. In backward-mode, the finite-difference in LPGD update has the equivalent advantage over standard co-derivatives, by capturing higher-order information instead of linear sensitivities.

Note that for $L(x, y) = \ell(x)$ this reduces to

$$\lim_{\tau\to 0} \nabla_w \tilde{\ell}_\tau(w) = \nabla_w \ell(x^*(w)). \tag{150}$$

[16] Note that a similar derivation already appeared in (Domke, 2010), but only for primal variables with linear parameters and without considering the benefits of finite values of τ.

An analogous proof and discussion also hold for the upper envelope $\widetilde{\ell}^{\tau}$ and average envelope $\widetilde{\ell}_{\tau}$, corresponding to the co-derivative counterparts of the left-sided and central finite-difference approximations of the derivative, respectively. □

Proposition 6.4. *Assume $\mathcal{L}, \ell$ are lower semi-continuous and ℓ is finite-valued on $\mathcal{X}$. Let w be a parameter for which*

$$\widehat{\mathcal{X}}(w) := \{x \in \mathcal{X} \mid \mathcal{D}^*_{\mathcal{L}}(x|w) < \infty\} \tag{151}$$

is nonempty. Then

$$\lim_{\tau\to\infty} x_\tau(w) \in \arg\min_{x\in\widehat{\mathcal{X}}(w)} \ell(x) \tag{152}$$

whenever the limit exists. For a linearized loss, we have

$$\lim_{\tau\to\infty} \widetilde{x}_\tau(w) \in \arg\min_{x\in\widehat{\mathcal{X}}(w)} \langle x, \nabla\ell\rangle = x_{FW}(w), \tag{153}$$

where x_{FW} is the solution to a Frank-Wolfe iteration LP (Frank & Wolfe, 1956)

Proof of Proposition 6.4. Throughout the proof, let w be a fixed parameter, we therefore omit the dependence in the notation. For $\tau > 0$ we use $f_\tau \colon \mathcal{X} \to \mathbb{R}$ defined in (126). Since $D^*_{\mathcal{L}} \geq 0$ on $\mathcal{X}$, we have that

$$f_\tau \geq f_{\tau'} \geq \ell \quad \text{on } \mathcal{X} \tag{154}$$

whenever $\tau' \geq \tau$. Now let $(x_\tau)_{\tau>0}$ be minimizers of (126) such that $x_\tau \to x_\infty \in \mathcal{X}$ as $\tau \to \infty$. We show that x_∞ minimizes $\ell(x)$ over $\widehat{\mathcal{X}}$.

To this end, let $(\tau_n)_{n=1}^\infty$ be a non-decreasing sequence such that $\tau_n \to \infty$ and denote $x_n = x_{\tau_n}$ and $f_n = f_{\tau_n}$, for short. Since $\widehat{\mathcal{X}}$ is nonempty and ℓ is finite-valued on $\mathcal{X}$, we have that $f_n(x_n) < \infty$ and $x_n \in \widehat{\mathcal{X}}$ for all $n \in \mathbb{N}$.

We assume lower semi-continuity of $\mathcal{L}(x, y)$ in x for every $y \in \mathcal{Y}$, therefore the Lagrangian divergence

$$D^*_{\mathcal{L}}(x) = \sup_{y\in\mathcal{Y}} \mathcal{L}(x, y) - \mathcal{L}^* \tag{155}$$

is also lower semi-continuous, i.e.

$$\liminf_{n\to\infty} D^*_{\mathcal{L}}(x_n) \geq D^*_{\mathcal{L}}(x_\infty). \tag{156}$$

As $x_n \in \widehat{\mathcal{X}}$ for all $n \in \mathbb{N}$ it follows that

$$\infty > \liminf_{n\to\infty} D^*_{\mathcal{L}}(x_n) \geq D^*_{\mathcal{L}}(x_\infty), \tag{157}$$

and therefore $x_\infty \in \widehat{\mathcal{X}}$ as well.

Next, by (154), the sequence $f_n(x_n)$ for $n \in \mathbb{N}$ is nonincreasing and bounded from below (by a minimum of ℓ on $\overline{\{x_n \mid n \in \mathbb{N}\}}$). Therefore, $f_n(x_n)$ is convergent. By lower semi-continuity of ℓ, we have that $\liminf_{n\to\infty} \ell(x_n) \geq \ell(x_\infty)$ and thus $\frac{1}{\tau_n} D^*_{\mathcal{L}}(x_n|w)$ converges to some $c \geq 0$. Altogether, for any $\hat{x} \in \widehat{\mathcal{X}}(w)$, we have

$$\begin{aligned}\ell(x_\infty) + c &\leq \liminf_{n\to\infty} \ell(x_n) + \liminf_{n\to\infty} \tfrac{1}{\tau_n} D^*_{\mathcal{L}}(x_n|w) \\ &\leq \liminf_{n\to\infty} \big[\ell(x_n) + \tfrac{1}{\tau_n} D^*_{\mathcal{L}}(x_n|w)\big] \\ &\leq \lim_{n\to\infty} \ell(\hat{x}) + \tfrac{1}{\tau_n} D^*_{\mathcal{L}}(\hat{x}|w) \\ &= \ell(\hat{x}).\end{aligned} \tag{158}$$

The particular choice $\hat{x} = x_\infty$ shows that $c = 0$. Therefore $\ell(x_\infty) \leq \ell(\hat{x})$ for any $\hat{x} \in \widehat{\mathcal{X}}$ as desired. The second part of the proposition follows directly by taking a loss linearization. □

Proposition 6.5. *Assume ℓ is continuous and finite-valued on $\mathcal{X}$. Let w be a parameter for which $\widehat{\mathcal{X}}(w)$ is nonempty. The primal lower $\mathcal{L}$-proximal map (30) turns into the standard proximal map (5)*

$$\lim_{\tau\to\infty} x_{\tau\rho}(w) = \arg\min_{x\in\widehat{\mathcal{X}}(w)} \big[\ell(x) + \tfrac{1}{2\rho}\|x - x^*\|_2^2\big] \tag{159}$$

$$= \operatorname{prox}_{\rho\ell + I_{\widehat{\mathcal{X}}(w)}}(x^*), \tag{160}$$

whenever the limit exists. For a linearized loss, it reduces to the Euclidean projection onto $\widehat{\mathcal{X}}(w)$

$$\lim_{\tau\to\infty} \widetilde{x}_{\tau\rho}(w) = \arg\min_{x\in\widehat{\mathcal{X}}(w)} \big[\langle x, \nabla\ell\rangle + \tfrac{1}{2\rho}\|x - x^*\|_2^2\big] \tag{161}$$

$$= P_{\widehat{\mathcal{X}}(w)}(x^* - \rho\nabla\ell). \tag{162}$$

Proof of Proposition 6.5. The proof is analogous to that of Proposition 6.4 with $f_\tau \colon \mathcal{X} \to \mathbb{R}$, where

$$f_\tau(x) = \ell(x) + \tfrac{1}{\tau}\mathcal{D}^*_{\mathcal{L}}(x|w) + \tfrac{1}{2\rho}\|x - x^*\|_2^2. \quad \square$$

Proposition 6.6. *Assume that the solvers for (2, 13) return ε-accurate solutions $\widehat{z}^*, \widehat{z}_\tau$ with δ-accurate objective values, i.e.*

$$\|\widehat{z}^* - z^*\| \leq \varepsilon, \tag{163}$$

$$|\mathcal{L}(\widehat{z}^*, w) - \mathcal{L}(z^*, w)| \leq \delta, \tag{164}$$

$$\|\widehat{z}_\tau - z_\tau\| \leq \varepsilon, \tag{165}$$

$$\big|[\mathcal{L}(\widehat{z}_\tau, w) + \tau\ell(\widehat{x}_\tau)] - [\mathcal{L}(z_\tau, w) + \tau\ell(x_\tau)]\big| \leq \delta. \tag{166}$$

Then for the approximate lower $\mathcal{L}$-envelope

$$\widehat{\ell}_\tau(w) := \ell(\widehat{x}_\tau) + \tfrac{1}{\tau}\big[\mathcal{L}(\widehat{z}_\tau, w) - \mathcal{L}(\widehat{z}^*, w)\big] \tag{167}$$

it holds that

$$|\widehat{\ell}_\tau(w) - \ell_\tau(w)| \leq \frac{2\delta}{\tau}. \tag{168}$$

Moreover, if $\nabla_w\mathcal{L}(x, y, w)$ is L-Lipschitz continuous in $(x, y) \in \mathcal{X} \times \mathcal{Y}$, it holds that

$$\|\nabla\widehat{\ell}_\tau(w) - \nabla\ell_\tau(w)\| \leq \frac{2L\varepsilon}{\tau}. \tag{169}$$

Proof of Proposition 6.6. We have

$$\begin{aligned}&|\widehat{\ell}_\tau(w) - \ell_\tau(w)|\\ &\quad\leq \Big|\big[\ell(\widehat{x}_\tau) + \tfrac{1}{\tau}\mathcal{L}(\widehat{z}_\tau, w)\big] - \big[\ell(x_\tau) + \tfrac{1}{\tau}\mathcal{L}(z_\tau, w)\big]\Big|\\ &\qquad + \big|\tfrac{1}{\tau}\mathcal{L}(\widehat{z}^*, w) - \tfrac{1}{\tau}\mathcal{L}(z^*, w)\big| \leq \tfrac{2\delta}{\tau},\end{aligned}$$

which shows the first part of the proposition. The second part follows from

$$\begin{aligned}&\big\|\nabla\widehat{\ell}_\tau(w) - \nabla\ell_\tau(w)\big\|\\ &\quad\leq \Big\|\tfrac{1}{\tau}\big[\nabla_w\mathcal{L}(\widehat{z}_\tau, w) - \nabla_w\mathcal{L}(\widehat{z}^*, w)\big]\\ &\qquad + \tfrac{1}{\tau}\big[\nabla_w\mathcal{L}(z_\tau, w) - \nabla_w\mathcal{L}(z^*, w)\big]\Big\|\\ &\quad\leq \tfrac{1}{\tau}\big\|\nabla_w\mathcal{L}(\widehat{z}_\tau, w) - \nabla_w\mathcal{L}(z_\tau, w)\big\|\\ &\qquad + \tfrac{1}{\tau}\big\|\nabla_w\mathcal{L}(\widehat{z}^*, w) - \nabla_w\mathcal{L}(z^*, w)\big\|\\ &\quad\leq \tfrac{L}{\tau}\|\widehat{z}_\tau - z_\tau\| + \tfrac{L}{\tau}\|\widehat{z}^* - z^*\| \leq \tfrac{2L\varepsilon}{\tau}.\end{aligned} \tag{170}$$

The approximation of the upper $\mathcal{L}$-envelope has an analogous proof. □

G. Experiments

G.1. Learning the Rules of Sudoku

Instead of the mini-Sudoku case (4×4 grid) in Amos & Kolter (2017a), we consider the full 9×9 Sudoku grid. The Sudoku board is modelled as a one-hot-encoding, with incomplete input and solved label $x_{\text{inc}}, x_{\text{true}} \in \{0, 1\}^{9\times9\times9}$. The optimization problem is modelled as a generic box-constrained linear program

$$\begin{aligned} x^*(A, b; x_{\text{inc}}) = &\arg\min_{x\in\mathcal{X}} \langle x, x_{\text{inc}}\rangle \\ &\text{subject to} \quad Ax + b = 0\end{aligned} \tag{171}$$

with $\mathcal{X} = [0, 1]^{9\times9\times9}$ and a sufficient number of constraints m to represent the rules of the LP Sudoku.[17] The saddle-point formulation of (171) is

$$z^*(A, b; x_{\text{inc}}) = \arg\min_{x\in\mathcal{X}} \max_{y\in\mathbb{R}^m} \mathcal{L}(x, y, A, b, x_{\text{inc}}), \tag{172}$$

with $\mathcal{L}(x, y, A, b, c) = \langle x, c\rangle + \langle y, Ax + b\rangle$. Note that we have the effective feasible set

$$\widehat{\mathcal{X}}(A, b) = \{x \in [0, 1]^{9\times9\times9} \mid Ax + b = 0\}. \tag{173}$$

The constraint parameters A and b are themselves parameterized such that at least one feasible point exists, which means that the optimization problem has a finite optimal solution, implying strong duality of (172).

We follow the training protocol described by Amos & Kolter (2017a) that minimizes the mean square error between predictions $x^*(A, b; x_{\text{inc}})$ and one-hot encodings of the correctly solved Sudokus x_{true}. For evaluation, we follow Amos & Kolter (2017a) in refining the predictions by taking an argmax over the one-hot dimension and report the percentage of violated ground-truth Sudoku constraints as the error.

[17]The formulation as an LP differs from the original formulation by Amos & Kolter (2017a), in which a quadratic regularizer has to be added to meet the method requirements.

We modify the public codebase from (Amos & Kolter, 2017b). The dataset consists of 9000 training and 1000 test instances. We choose the hyperparameters learning rate α, τ and ρ with a grid search. The best hyperparameters for LPGD are $\tau = 10^4$, $\rho = 0.1$, $\alpha = 0.1$, for gradient descent they are $\rho = 10^{-3}$, $\alpha = 0.1$. We use these hyperparameters in our evaluation.

Additional metrics are reported in Figure 4 and Figure 5. We observe that there is not significant difference between train and test metrics, which shows that our formulation of Sudoku (171) allows generalizing across instances. The results also contain the upper and lower variations LPGD$_\tau$ and LPGD$^\tau$ in addition to the average LPGDτ. We observe that LPGDτ outperforms LPGD$^\tau$ and LPGD$_\tau$, highlighting that both the lower and upper envelopes carry relevant information for the optimization. This is intuitively understandable by considering that in Figure 1 LPGD$_\tau$ and LPGD$^\tau$ provide non-zero gradients in different subsets of the domain, while LPGDτ gives informative gradients in both subsets.

G.2. Tuning a Markowitz Control Policy

In the Markowitz Portfolio Optimization setting described in Agrawal et al. (2020, §5), the task is to iteratively trade assets in a portfolio such that a utility function based on returns and risk is maximized over a trading horizon. The trading policy is a convex optimization control policy, which determines trades by solving a parameterized convex optimization problem. It maximizes a parameterized objective that trades off the expected return and the risk of the post-trade portfolio, subject to a constraint that ensures self-financing trades. The parameters are initialized based on historical data, and differentiating through the optimization problem allows tuning them to maximize the utility on simulated evolutions of the asset values. We refer the reader to Agrawal et al. (2020, §5) for a detailed description.

We conduct a sweep over the solver accuracy ϵ, the results are reported in Figure 6. The results show that LPGD is more sensitive to inaccurate solutions than GD, as the update relies on accurate solutions for the finite difference. Finally, we report statistics for $\tau = 100$, $\alpha = 0.001$, $\epsilon = 0.0001$ over 3 training seeds. We do not observe high variability in the performance.

H. List of Symbols

x	primal variables
s	optional slack variables
y	dual variables
$z = (x, y)$	primal & dual variables
$z^* = (x^*, y^*)$	optimal variables (2)
$\mathcal{X}, \mathcal{Y}$	primal and dual feasible sets
$\widetilde{\mathcal{X}}$	primal effective feasible set (32)
w	all parameters
u	linear parameters
v	non-linear parameters
c	linear primal parameters
b	linear dual parameters
$\mathcal{L}$	Lagrangian
$\mathcal{L}^*$	optimal Lagrangian (1)
Ω	non-linear part of Lagrangian
τ	perturbation strength parameter
$\mathrm{env}_{\tau f}$	Moreau envelope (4)
$\mathrm{prox}_{\tau f}$	Proximal map (5)
ℓ	loss on primal variables
$D_{\mathcal{L}}$	Lagrangian difference (8)
$D^*_{\mathcal{L}}$	Lagrangian divergence (9)
$\ell_\tau, \ell^\tau, \ell{\scriptstyle\tau}$	$\mathcal{L}$-envelopes (12, 14, 16)
$z_\tau = (x_\tau, y_\tau)$	lower $\mathcal{L}$-proximal map (13)
$z^\tau = (x^\tau, y^\tau)$	upper $\mathcal{L}$-proximal map (15)
$\nabla\ell_\tau, \nabla\ell^\tau, \nabla\ell{\scriptstyle\tau}$	LPPM updates (17)
$\Delta\ell_\tau, \Delta\ell^\tau, \Delta\ell{\scriptstyle\tau}$	LPPM finite-differences (41)
$\widetilde{\ell}$	linearization of ℓ at x^* (6)
$\widetilde{\ell}_\tau, \widetilde{\ell}^\tau, \widetilde{\ell}{\scriptstyle\tau}$	$\mathcal{L}$-envelopes of lin. loss $\widetilde{\ell}$
$\widetilde{z}_\tau = (\widetilde{x}_\tau, \widetilde{y}_\tau)$	lin. lower $\mathcal{L}$-proximal map (22)
$\widetilde{z}^\tau = (\widetilde{x}^\tau, \widetilde{y}^\tau)$	lin. upper $\mathcal{L}$-proximal map (23)
$\nabla\widetilde{\ell}_\tau, \nabla\widetilde{\ell}^\tau, \nabla\widetilde{\ell}{\scriptstyle\tau}$	LPGD updates (24)
$\Delta\widetilde{\ell}_\tau, \Delta\widetilde{\ell}^\tau, \Delta\widetilde{\ell}{\scriptstyle\tau}$	LPGD finite-differences
ρ	augmentation strength parameter
$\mathcal{L}_\rho$	augmented Lagrangian (28)
$\ell_{\tau\rho}, \ell^{\tau\rho}, \ell{\scriptstyle\tau\rho}$	aug. $\mathcal{L}$-envelopes (29)
$z_{\tau\rho} = (x_{\tau\rho}, y_{\tau\rho})$	aug. lower $\mathcal{L}$-proximal map (30)
$z^{\tau\rho} = (x^{\tau\rho}, y^{\tau\rho})$	aug. upper $\mathcal{L}$-proximal map
$\nabla\ell_{\tau\rho}, \nabla\ell^{\tau\rho}, \nabla\ell{\scriptstyle\tau\rho}$	aug. LPPM updates (31)
$\Delta\ell_{\tau\rho}, \Delta\ell^{\tau\rho}, \Delta\ell{\scriptstyle\tau\rho}$	aug. LPPM finite-differences (42)
$\widetilde{\ell}_{\tau\rho}, \widetilde{\ell}^{\tau\rho}, \widetilde{\ell}{\scriptstyle\tau\rho}$	aug. $\mathcal{L}$-envelopes of lin. loss $\widetilde{\ell}$
$\widetilde{z}_{\tau\rho} = (\widetilde{x}_{\tau\rho}, \widetilde{y}_{\tau\rho})$	aug. lin. lower $\mathcal{L}$-proximal map
$\widetilde{z}^{\tau\rho} = (\widetilde{x}^{\tau\rho}, \widetilde{y}^{\tau\rho})$	aug. lin. upper $\mathcal{L}$-proximal map
$\nabla\widetilde{\ell}_{\tau\rho}, \nabla\widetilde{\ell}^{\tau\rho}, \nabla\widetilde{\ell}{\scriptstyle\tau\rho}$	aug. lin. LPGD updates (31)
$\Delta\widetilde{\ell}_{\tau\rho}, \Delta\widetilde{\ell}^{\tau\rho}, \Delta\widetilde{\ell}{\scriptstyle\tau\rho}$	aug. lin. LPGD finite-differences (42)
L	loss on primal & dual variables
$\widetilde{L}$	linearization of L at z^*

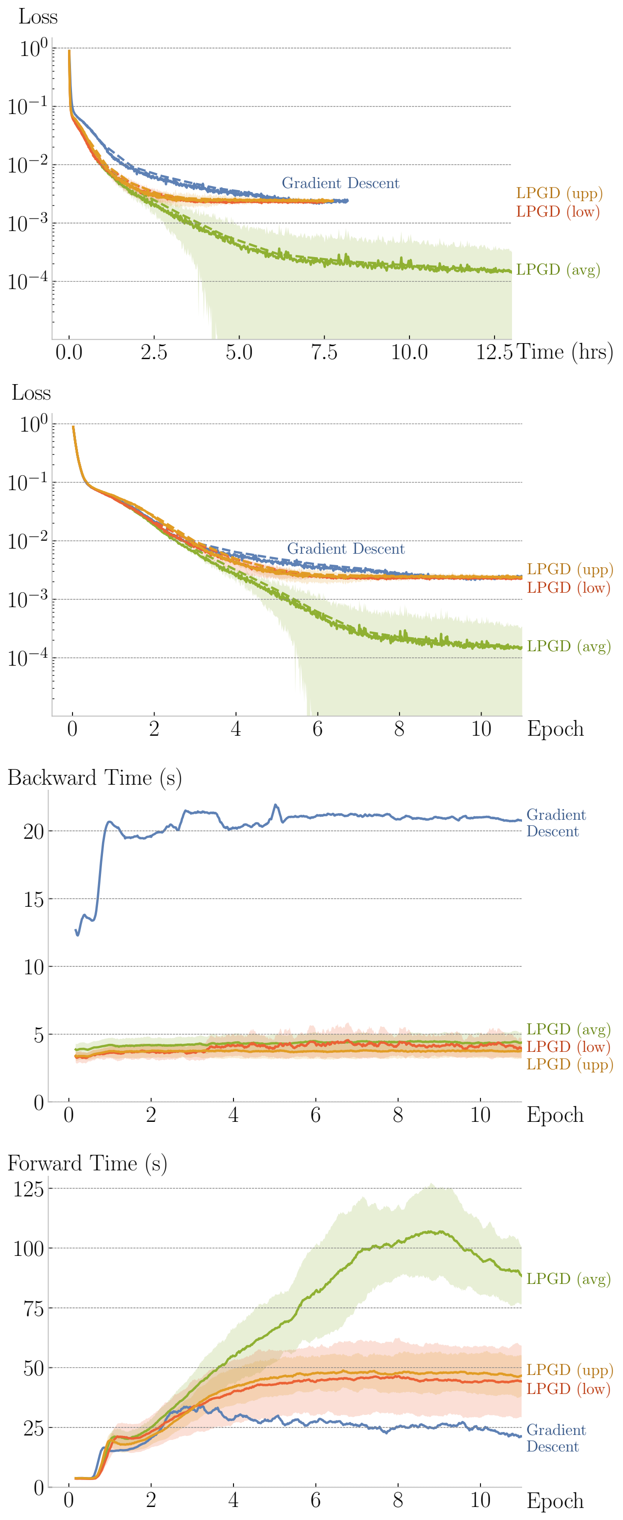


Figure 4: Comparison of LPGD variations and gradient descent (GD) on the Sudoku experiment. Reported are train and test MSE over epochs, wall-clock time, and time spent in the backward and forward passes. Dashed lines correspond to test data. Statistics are over 5 restarts.

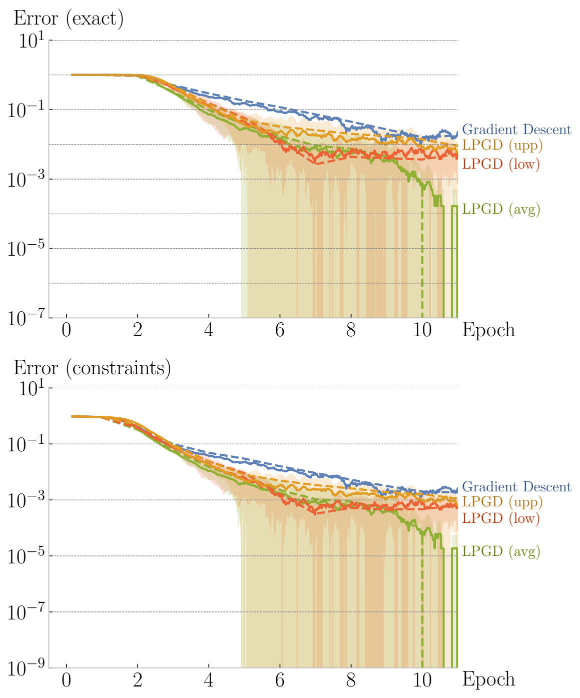


Figure 5: Comparison of LPGD and gradient descent (GD) on the Sudoku experiment. Reported are train and test errors over epochs. Dashed lines correspond to test data. The exact error refers to the proportion of incorrect solutions (at least one violated Sudoku constraint), while the constraint error refers to the proportion of violated Sudoku constraints. Statistics are over 5 restarts.

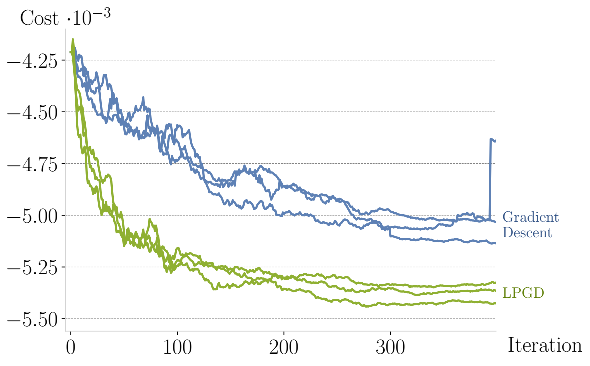


Figure 6: Comparison of LPGDτ and GD in the Markowitz portfolio optimization experiment. Reported is a sweep over training seeds, using $\alpha = 0.001$, $\epsilon = 0.0001$, $\tau = 100$ and $\rho = 0$.