Lagrangian Proximal Gradient Descent for Learning Convex Optimization Models

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

Abstract

We propose *Lagrangian Proximal Gradient Descent* (LPGD), a flexible framework for learning convex optimization models. Like traditional proximal gradient methods, LPGD can be interpreted as optimizing a smoothed envelope of the possibly non-differentiable loss. The smoothening allows training models that do not provide informative gradients, such as discrete optimization models. We show that the LPGD update can efficiently be computed by rerunning the forward solver on a perturbed input. Moreover, we prove that the LPGD update converges to the gradient as the smoothening parameter approaches zero. Finally, we experimentally investigate the potential benefits of applying LPGD even in a fully differentiable setting.

1. Introduction

Optimization at inference is inherent to many prediction tasks, including autonomous driving (Paden et al., 2016), modelling physical systems (Cranmer et al., 2020), or robotic control (Kumar et al., 2016). Therefore, embedding optimization algorithms in machine learning models is a powerful inductive bias. A recent trend has been to embed constrained convex optimization problems that can efficiently be solved to optimality (Amos & Kolter, 2017; Agrawal et al., 2019a;b; Sun et al., 2022).

Training a convex optimization model is an instance of bilevel optimization (Gould et al., 2016), which is generally challenging. When it is possible to propagate informative gradients through the optimization problem, the task is typically approached with standard stochastic gradient descent (GD) (Amos & Kolter, 2017; Agrawal et al., 2019b). In some problems, such as discrete optimization layers, gradients are not informative. Previous works have proposed several methods to overcome this, 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). See Appendix A for a more detailed overview of the related work.

In this work, we unify and generalize previous finite-difference-based methods (McAllester et al., 2010; Domke, 2010; Vlastelica et al., 2020) in a general framework called *Lagrangian Proximal Gradient Descent* (LPGD). It is motivated by traditional proximal optimization techniques (Moreau, 1962; Parikh & Boyd, 2014; Boyd & Vandenberghe, 2014), fostering deep links between traditional and contemporary methods. Practically, in the differentiable setting, we show that gradients can be computed as the limit of the LPGD update, which provides an alternative to previous methods based on differentiating the optimality conditions (Amos & Kolter, 2017; Agrawal et al., 2019b). In the non-differentiable setting, LPGD allows learning the parameters even when GD fails, generalizing (Vlastelica et al., 2020) to non-linear objectives and learnable constraints. Further, we introduce regularization to stabilize training with LPGD. Finally, we empirically investigate to which extent LPGD can improve upon GD even in the fully differentiable setting.

2. Background & Notation

2.1. Problem Setup

We consider a model that contains an embedded constrained convex optimization procedure:

$$e \in \mathbb{R}^p \xrightarrow{W_\theta} w \in \mathbb{R}^k \xrightarrow{z^*} (x^*, y^*) \in \mathbb{R}^{n+m} \xrightarrow{\ell} \ell(x^*) \in \mathbb{R}$$

Given an input $e \in \mathbb{R}^p$, the model W_θ predicts parameters of the optimization problem $w \in \mathbb{R}^k$. A solver then finds an optimal primal-dual solution pair $z^* = (x^*, y^*) \in \mathbb{R}^n \times \mathbb{R}^m$ of the optimization problem, and the primal solution x^* is passed to a loss function $\ell(x^*)$.[1] The loss can further consist of additional layers that could be trained standardly. We focus on minimizing the loss with respect to the parameters θ.

We consider constrained optimization problems of the form

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

[1] A more general setup of a loss depending on both primal and dual solutions is discussed in Appendix C.3.

[1] Max Planck Institute for Intelligent Systems, Tübingen, Germany [2] Masaryk University, Brno, Czech Republic. Correspondence to: Anselm Paulus <anselm.paulus@tue.mpg.de>.

Published at the Differentiable Almost Everything Workshop of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. July 2023. Copyright 2023 by the author(s).

where $\mathcal{X} \subseteq \mathbb{R}^n$ is convex and $\mathcal{L}$ is a continuously differentiable *Lagrangian* that is convex in the primal variables x and affine in the dual variables y. The affinity assumption is not restrictive as inequalities can be enforced by introducing slack variables, see Appendix C.1 for details and examples covering conic and quadratic programs. Next, we assume strong duality

$$\mathcal{L}^*(w) := \min_{x \in \mathcal{X}} \max_{y} \mathcal{L}(x, y, w) = \max_{y} \min_{x \in \mathcal{X}} \mathcal{L}(x, y, w) \quad (2)$$

and, for simplicity, we assume $\mathcal{L}$ to be strictly convex in x, so that optimization (1) always attains a unique solution.[2]

2.2. Proximal Gradient Descent & Moreau Envelope

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

$$\operatorname{env}_{\tau f}(\widehat{x}) = \inf_x f(x) + \tfrac{1}{2\tau}\|x - \widehat{x}\|_2^2. \quad (3)$$

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

$$\begin{aligned} \operatorname{prox}_{\tau f}(\widehat{x}) &= \arg\inf_x f(x) + \tfrac{1}{2\tau}\|x - \widehat{x}\|_2^2 \\ &= \widehat{x} - \tau \nabla \operatorname{env}_{\tau f}(\widehat{x}). \end{aligned} \quad (4)$$

Intuitively, the proximal map searches for an x close to $\widehat{x}$ with a lower value of f, and the Moreau envelope is the sum of f and a Euclidean distance penalty at this x. The *proximal point method* (Parikh & Boyd, 2014; Güler, 1992) aims to minimize f by iteratively updating $\widehat{x} \mapsto \operatorname{prox}_{\tau f}(\widehat{x})$.

Now, assume that f decomposes as $f = g + h$ with g differentiable and h potentially non-smooth and consider

$$\widetilde{g}(x) := g(\widehat{x}) + \langle x - \widehat{x}, \nabla g(\widehat{x}) \rangle, \quad (5)$$

a linearization of g around $\widehat{x}$. This yields

$$\begin{aligned} \operatorname{prox}_{\tau(\widetilde{g}+h)}(\widehat{x}) &= \arg\inf_x \widetilde{g}(x) + h(x) + \tfrac{1}{2\tau}\|x - \widehat{x}\|_2^2 \\ &= \operatorname{prox}_{\tau h}(\widehat{x} - \tau \nabla g(\widehat{x})) \end{aligned} \quad (6)$$

and iterating $\widehat{x} \mapsto \operatorname{prox}_{\tau h}(\widehat{x} - \tau \nabla g(\widehat{x}))$ is called *proximal gradient descent* (Parikh & Boyd, 2014). For $h = I_{\mathcal{X}}$ this reduces to *projected gradient descent* (Parikh & Boyd, 2014).

3. Method

Our goal is to translate the idea of proximal methods to hybrid models (2.1) by defining a smoothed envelope of the loss $w \mapsto \ell(x^*(w))$ on which we can perform gradient descent. Given w and optimal solution x^*, this envelope should select an x in the proximity of x^* with a lower loss ℓ. The key concept is to replace the Euclidean distance term with a *Lagrangian divergence* indicating how close x is to optimality given w.

[2]For set-valued solution mapping see Appendix C.5.

3.1. Lagrangian Divergence

For $x \in \mathcal{X}$, we have the inequality

$$\sup_y \mathcal{L}(x, y, w) \geq \inf_{x \in \mathcal{X}} \sup_y \mathcal{L}(x, y, w) = \mathcal{L}^*(w) \quad (7)$$

and therefore we define the *Lagrangian divergence*[3] as

$$D_{\mathcal{L}}(x, x^* | w) := \sup_y \mathcal{L}(w, x, y) - \mathcal{L}^*(w) \geq 0 \quad (8)$$

for $x \in \mathcal{X}$ and $w \in \mathbb{R}^k$. Note that x^* is determined by w, therefore $D_{\mathcal{L}}$ is a function of only x and w and we abbreviate it as $D_{\mathcal{L}}(x|w)$. The divergence holds the key property

$$D_{\mathcal{L}}(x|w) = 0 \text{ if and only if } x = x^*(w) \text{ for } x \in \mathcal{X}, \quad (9)$$

which makes it a good measure of optimality of x given w.

3.2. Lower Lagrange-Moreau Envelope

Given $\tau > 0$, we call the *lower Lagrange-Moreau envelope* ($\mathcal{L}$-envelope) the function $\ell_\tau \colon \mathbb{R}^k \to \mathbb{R}$ defined as

$$\begin{aligned} \ell_\tau(w) &:= \inf_{x \in \mathcal{X}} \ell(x) + \tfrac{1}{\tau} D_{\mathcal{L}}(x|w) \\ &= \inf_{x \in \mathcal{X}} \sup_y \ell(x) + \tfrac{1}{\tau}[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)]. \end{aligned} \quad (10)$$

The lower $\mathcal{L}$-envelope ℓ_τ is a smoothed lower bound approximation of the function $w \mapsto \ell(x^*(w))$. The smoothness of $\ell\tau$ is inherited from the smoothness of the Lagrangian. A function $z_{\tau\ell} \colon \mathbb{R}^k \to \mathbb{R}^{n+m}$ representing the corresponding *lower $\mathcal{L}$-proximal map* is given by

$$\begin{aligned} z_{\tau\ell}(w) &:= \arg \inf_{x \in \mathcal{X}} \sup_y \ell(x) + \tfrac{1}{\tau}[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)] \\ &= \arg \inf_{x \in \mathcal{X}} \sup_y \mathcal{L}(x, y, w) + \tau \ell(x). \end{aligned} \quad (11)$$

The objectives in (10) and (11) are strictly convex in x; hence the optimal points are uniquely attained.

3.3. Upper Lagrange-Moreau Envelope

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

$$\begin{aligned} \ell^\tau(w) &:= \sup_{x \in \mathcal{X}} \ell(x) - \tfrac{1}{\tau} D_{\mathcal{L}}(x|w) \\ &= - \inf_{x \in \mathcal{X}} \sup_y \tfrac{1}{\tau}[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)] - \ell(x). \end{aligned} \quad (12)$$

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

$$\begin{aligned} z^{\tau\ell}(w) &:= \arg \sup_{x \in \mathcal{X}} \ell(x) - \tfrac{1}{\tau} D_{\mathcal{L}}(x|w) \\ &= \arg \inf_{x \in \mathcal{X}} \sup_y \mathcal{L}(x, y, w) - \tau \ell(x). \end{aligned} \quad (13)$$

[3]In some cases, the Lagrangian divergence coincides with the *Bregman divergence*, a proximity measure generalizing the Euclidean distance, opening connections to *Mirror descent*, see Appendix C.2.

Analogously to ℓ_τ, the upper $\mathcal{L}$-envelope ℓ^τ is a smoothed upper bound of $w \mapsto \ell(x^*(w))$.[4] We have the simple relations

$$\ell^\tau = -(-\ell)_\tau \quad \text{and} \quad z^{\tau\ell} = z_{-\tau\ell} \tag{14}$$

between the $\mathcal{L}$-envelopes and $\mathcal{L}$-proximal maps.

As both the lower and upper envelope have desirable properties, we will also work with the *average* $\mathcal{L}$-envelope[5]

$$\ell_{\tau\!\!\!\!\tau}(w) := \tfrac{1}{2}[\ell_\tau(w) + \ell^\tau(w)]. \tag{15}$$

3.4. Differentiating the Lagrange-Moreau Envelopes

We perform gradient descent on the $\mathcal{L}$-envelope. By Danskin's theorem (Danskin, 1966), the gradient of the envelope reads

$$\begin{aligned} \nabla\ell_\tau(w) &= \tfrac{1}{\tau}[\nabla_w\mathcal{L}(w, z_{\tau\ell}) - \nabla_w\mathcal{L}(w, z^*)], \\ \nabla\ell^\tau(w) &= \tfrac{1}{\tau}[\nabla_w\mathcal{L}(w, z^*) - \nabla_w\mathcal{L}(w, z^{\tau\ell})], \end{aligned} \tag{16}$$

where we abbreviate $z_{\tau\ell} = z_{\tau\ell}(w)$ and $z^{\tau\ell} = z^{\tau\ell}(w)$.

Example (Direct Loss Minimization)**.** For a loss $\ell(x, x_{\text{true}})$, feature function Ψ and an optimization problem of the form

$$x^*(w, \xi) = \arg\min_{x\in\mathcal{X}} -\langle w, \Psi(x, \xi)\rangle \tag{17}$$

the gradient of the lower $\mathcal{L}$-envelope update is

$$\nabla_w\ell_\tau(w) = \tfrac{1}{\tau}[\Psi(x^*, \xi) - \Psi(x_{\tau\ell}, \xi)] \tag{18}$$

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

$$x_{\tau\ell}(w, \xi) = \arg\min_{x\in\mathcal{X}} -\langle w, \Psi(x, \xi)\rangle + \tau\ell(x, x_{\text{true}}). \tag{19}$$

This recovers the "towards-better" *Direct Loss Minimization* (DLM) update (McAllester et al., 2010), while the "away-from-worse" update corresponds to the upper $\mathcal{L}$-envelope gradient. The DLM framework has also been generalized to non-linear objective functions (Song et al., 2016; Lorberbom et al., 2019), always considering $\tau \to 0$.

3.5. Lagrangian Proximal Gradient Descent

In practice, due to the loss term, the $\mathcal{L}$-proximal map can be difficult to evaluate. As in the proximal gradient descent algorithm, we consider a linearization $\tilde{\ell}$ of the loss ℓ at x^* like in (5), which allows computing the $\mathcal{L}$-proximal map with the forward pass solver.[6] W.l.o.g., exposing the linear parameters of the Lagrangian as

$$\mathcal{L}(x, y, w) := \langle x, c\rangle + H(x, y, v) \tag{20}$$

with $w = (c, v)$ and abbreviating $\nabla\ell = \nabla\ell(x^*)$ we get

$$\begin{aligned} z_{\tau\tilde{\ell}} &= \arg\inf_{x\in\mathcal{X}} \sup_y \langle x, c\rangle + H(x, y, v) + \tau\langle x, \nabla\ell\rangle \\ &= z^*(c + \tau\nabla\ell, v). \end{aligned} \tag{21}$$

The upper $\mathcal{L}$-proximal map is computed analogously. This enables efficient gradient computation for the $\mathcal{L}$-envelope of the linearized loss using the forward solver[7] as

$$\nabla\tilde{\ell}_{\tau\!\!\!\!\tau}(w) = \tfrac{1}{2\tau}\big[\nabla_w\mathcal{L}\big(w, z_{\tau\tilde{\ell}}\big) - \nabla_w\mathcal{L}\big(w, z^{\tau\tilde{\ell}}\big)\big]. \tag{22}$$

When using this gradient, we refer to *Lagrangian Proximal Gradient Descent* (LPGD), or more specifically, to LPGD$_\tau$, LPGD$_{\tau\!\!\!\!\tau}$ and LPGD$^\tau$ for gradient descent on $\tilde{\ell}_\tau$, $\tilde{\ell}_{\tau\!\!\!\!\tau}$ and $\tilde{\ell}^\tau$.

Theorem 3.1. *It holds that*

$$\lim_{\tau\to 0} \nabla\tilde{\ell}_\tau(c, v) = \nabla_{(c,v)}\ell(x^*(c, v)) = \lim_{\tau\to 0} \nabla\tilde{\ell}^\tau(c, v). \tag{23}$$

The proof is given in Appendix C.4. The extension for set-valued solution mappings is provided in Appendix C.6.

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

$$x^*(c) = \arg\min_{x\in\mathcal{X}}\langle x, c\rangle, \tag{24}$$

the LPGD$_\tau$ update reduces to

$$\nabla\tilde{\ell}_\tau(c) = \tfrac{1}{\tau}[x^*(c + \tau\nabla\ell) - x^*(c)], \tag{25}$$

the update rule in *Blackbox Backpropagation* (Vlastelica et al., 2020). Moreover, their piecewise affine interpolation of the loss $c \mapsto \tilde{\ell}(x^*(c))$ corresponds to the lower $\mathcal{L}$-envelope $\tilde{\ell}_\tau$.

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

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

with a strongly convex regularizer H, the LPGD$_{\tau\!\!\!\!\tau}$ update is

$$\nabla\tilde{\ell}_{\tau\!\!\!\!\tau}(c) = \tfrac{1}{2\tau}[x^*(c + \tau\nabla\ell) - x^*(c - \tau\nabla\ell)], \tag{27}$$

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

[4]Note the link to the *Proximal hull* (Rockafellar & Wets, 1998, Example 1.44) corresponding to taking an upper envelope of a lower envelope. Note also that for a general loss, the optimization (12) and (13) can diverge. This will not be problematic for the linearized loss we will consider in Sec. 3.5.

[5]Lower, upper and average envelopes are closely related to the right-, left- and double-sided directional derivatives, see Appendix C.4.

[6]Note that the loss linearization is only applied *after the solver* and does not approximate/linearize the solution mapping.

[7]Note that warm-starting the solver with z^* strongly accelerates the computation of the $\mathcal{L}$-proximal map.

[8]LP does not satisfy the strict convexity of the Lagrangian, and the solution map is set-valued in general, cf. Appendix C.5.

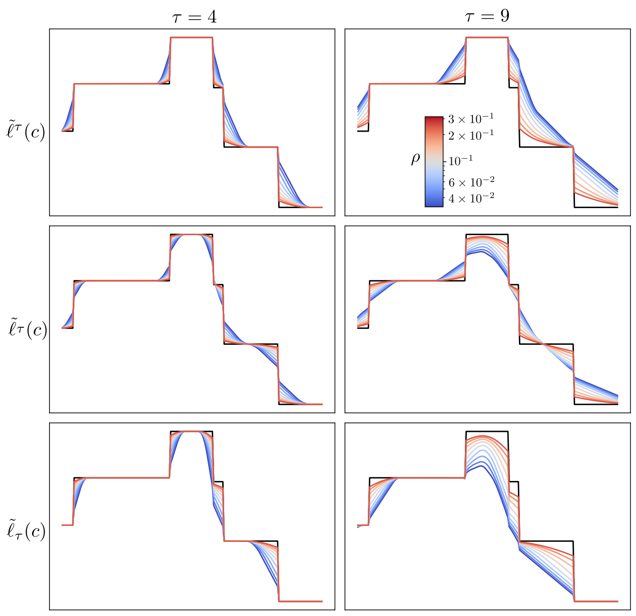

Figure 1. Visualization of the upper $\tilde{\ell}^{\tau}$, average $\tilde{\ell}_{\tau}$, and lower $\tilde{\ell}_{\tau}$ Lagrange-Moreau envelope for different temperatures τ and augmentation strengths ρ. The envelopes are smoothed approximations of the linearized loss $c \mapsto \tilde{\ell}_{\tau}(x^*(c))$, illustrated in black. In Lagrangian Proximal Gradient Descent (LPGD) we optimize the loss by gradient descent on the Lagrange-Moreau envelope.

3.6. Augmented Lagrangian

The $\mathcal{L}$-envelope inherits its smoothness from the Lagrangian. This motivates augmenting the Lagrangian with a strongly convex smoothing term as

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

The augmentation smoothens the Lagrangian envelope while not changing the current optimal solution.[9]

4. Experiments

4.1. Visualizations

We visualize the different $\mathcal{L}$-envelopes of the linearized loss $c \mapsto \widetilde{\ell}(x^*(c))$ in Fig. 4.1, for a quadratic loss on the solution to the linear program (24) with $\mathcal{X} = [0, 1]^n$ and a one-dimensional random cut through the cost space.

4.2. Sudoku

The main practical use-case of LPGD is the case when gradients of the loss are uninformative. However, we focus our experiments on fully differentiable problems, to investigate whether going beyond standard gradient-based methods is beneficial even in these cases.

[9]This also affects the adjoint derivative obtained from implicit differentiation, in which it introduces a regularizing term in the Jacobian. We discuss this in Appendix C.1.

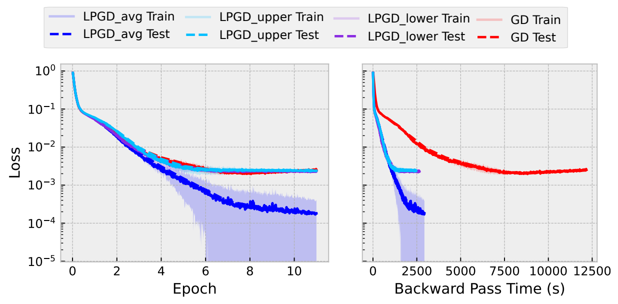

Figure 2. Comparison of LPGD and gradient descent (GD) on the Sudoku experiment. Reported are train and test MSE over epochs and wall-clock time spent in the backward pass. Statistics are over 5 restarts. Additional results can be found in Appendix B.

We consider the Sudoku experiment proposed by Amos & Kolter (2017). 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 B for detailed information on the experimental setup. We compare LPGD to gradient descent (GD), in which we use the CVXPY (Diamond & Boyd, 2016) implementation of (Agrawal et al., 2019b) to compute the gradients.

The results are reported in Fig. 4.2. LPGDτ reaches a lower final loss than GD, which shows that LPGDτ produces better update steps than standard gradients. LPGDτ also outperforms LPGD$^{\tau}$ and LPGD$_{\tau}$, which highlights that both lower and upper envelope carry relevant information for the optimization. Considering backward pass computation time, all variants of LPGD converge much faster than GD. This is due to the efficient computation of LPGD via warmstarting and parallelization. Additional results, including other metrics and wallclock time convergence, are reported in Appendix B.

5. Conclusion

We propose Lagrangian Proximal Gradient Descent (LPGD), a flexible framework for learning convex optimization models. It unifies and generalizes contemporary optimization methods while providing deep links to traditional ones.

LPGD approximates gradients as finite differences and only requires accessing the forward solver as a black-box oracle. Formulated as Gradient Descent (GD) on a loss function smoothening, LPGD allows learning general objective and constraint parameters even in the non-differentiable setting.

Empirically, we explore the potential benefits of LPGD over GD even in a fully differentiable setting. We find that in our synthetic setup, LPGD achieves faster convergence and better final results when compared to GD, which motivates applying LPGD in other fully differentiable experimental setups. Finally, an exciting direction for future work is to explore further the connections to Mirror descent (see Appendix C.2) and experimentally compare it to LPGD.

References

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.

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, 2017. URL http://proceedings.mlr.press/v70/amos17a.html.

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.

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.

Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J., and Bach, F. R. Learning with differentiable perturbed optimizers. *CoRR*, abs/2002.08676, 2020. URL https://arxiv.org/abs/2002.08676.

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/.

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.

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.

Danskin, J. M. The theory of max-min, with applications. *SIAM Journal on Applied Mathematics*, 14(4):641–664, 1966. ISSN 00361399. URL http://www.jstor.org/stable/2946123.

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 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, 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.

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://doi.org/10.48550/arXiv.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.

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.

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.

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.

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.

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.

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.

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. 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.

Sahoo, S. S., Vlastelica, M., Paulus, A., Musil, V., Kuleshov, V., and Martius, G. Gradient backpropagation through combinatorial algorithms: Identity with projection works. *CoRR*, abs/2205.15213, 2022. doi: 10.48550/arXiv.2205.15213. URL https://doi.org/10.48550/arXiv.2205.15213.

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. *CoRR*, abs/2210.01802, 2022. doi: 10.48550/arXiv.2210.01802. URL https://doi.org/10.48550/arXiv.2210.01802.

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.

Vlastelica, M., Rolínek, M., and Martius, G. Neuro-algorithmic policies enable fast combinatorial generalization. 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. 10575–10585. PMLR, 2021. URL http://proceedings.mlr.press/v139/vlastelica21a.html.

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.

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 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, 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. 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 true informative gradient of such a layer, either by use of the implicit function theorem to differentiate quadratic programs (Amos & Kolter, 2017), conic programs (Agrawal et al., 2019b), ADMM (Sun et al., 2022), 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 research has investigated problems in which no informative gradients exist. The techniques for still training the model 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., 2022). Other works have built on geometric proxy losses (Paulus et al., 2021) and finite differences (Vlastelica et al., 2021; Niepert et al., 2021).

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).

B. Experiments

We are given pairs of incomplete and solved sudokus. The task is to learn a model that solves incomplete sudokus. Instead of limiting ourselves to the mini-sudoku case (4×4 grid) as in (Amos & Kolter, 2017), we consider the full 9×9 sudoku grid. It is modelled as a one-hot-encoding; hence the partial input e and solved output x have dimension $p = k = n = 9 \times 9 \times 9$. The dataset consists of $N = 9000$ training and 1000 test instances. The solution procedure is modelled as a generic box-constrained linear program

$$x^*(A, b; e) = \arg\min_{x \in [0,1]^n} \langle x, e \rangle \tag{29}$$

$$\text{subject to } Ay + b = 0 \tag{30}$$

with a sufficient number of constraints to represent the rules of mini-sudoku. This differs from the original formulation in (Amos & Kolter, 2017), as we replace the quadratic regularizer with box constraints. Such a formulation is possible as our method is not limited to quadratic programs.

We follow the training protocol described in (Amos & Kolter, 2017), using a modification of the public codebase for the paper. The model is trained by minimizing the mean square error loss between the predictions and one-hot encodings of the correctly solved sudokus, for which an informative gradient with respect to the constraints exists. For evaluation, we follow (Amos & Kolter, 2017) in refining the prediction by taking an argmax over the one-hot dimension and report the percentage of violated ground-truth sudoku constraints. The hyperparameters α, τ and ρ were selected in a grid search.

We solve the optimization problems using CVXPY (Diamond & Boyd, 2016; Agrawal et al., 2018). The optimization problem is automatically transformed to a conic program by CVXPY for which we implemented LPGD. We compare LPGD to the standard implementation of the conic program differentiation (Agrawal et al., 2019b), with the augmentation as described in Sec. C.1.

We chose the hyperparameters learning rate α, τ and ρ with a grid search. The best hyperparameters for LPGD are $\tau = 10^4$, $\rho = 10$, $\alpha = 0.1$, for gradient descent they are $\rho = 10^3$, $\alpha = 0.1$. We use these hyperparameters in our final evaluation.

The loss and error training curves are presented in Fig. 3 and Fig. 4. LPGD converges much faster than GD in terms of time spent in the backward pass. This is also visible in the convergence in terms of total wallclock time, although to a lesser extent, as the forward pass becomes more and more computationally demanding as training progresses and non-trivial constraints are discovered. It is also apparent that LPGD$_{\tau}$ achieves a lower final loss and error than GD, LPGD$^{\tau}$ and LPGD$_{\tau}$. This suggests that both lower and upper envelope carry relevant information for the optimization.

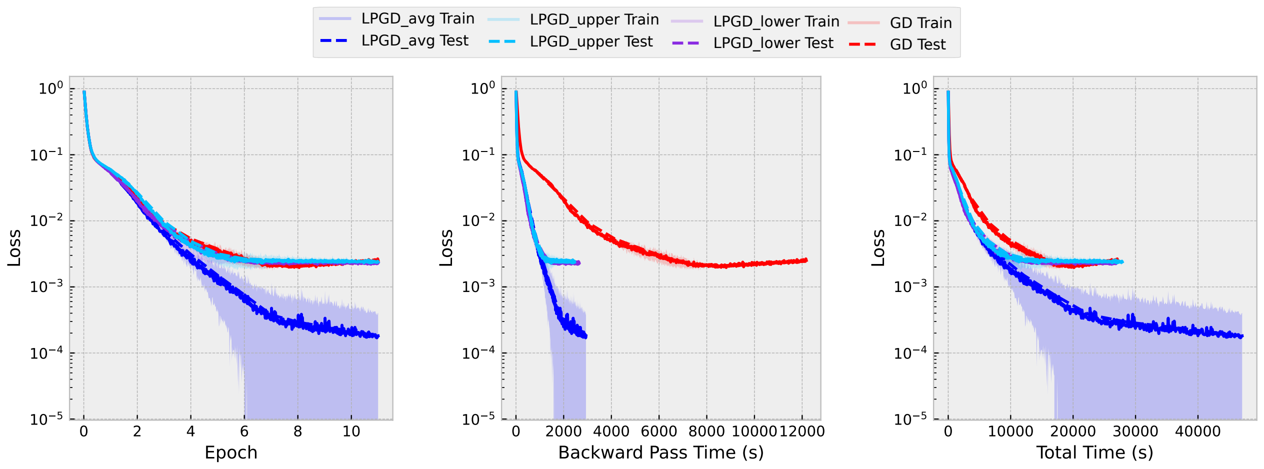

Figure 3. Results for the Sudoku experiment. Reported are the batched train and test loss over 11 epochs. We additionally report the loss over total wallclock time, and total time spent in the backward passes.

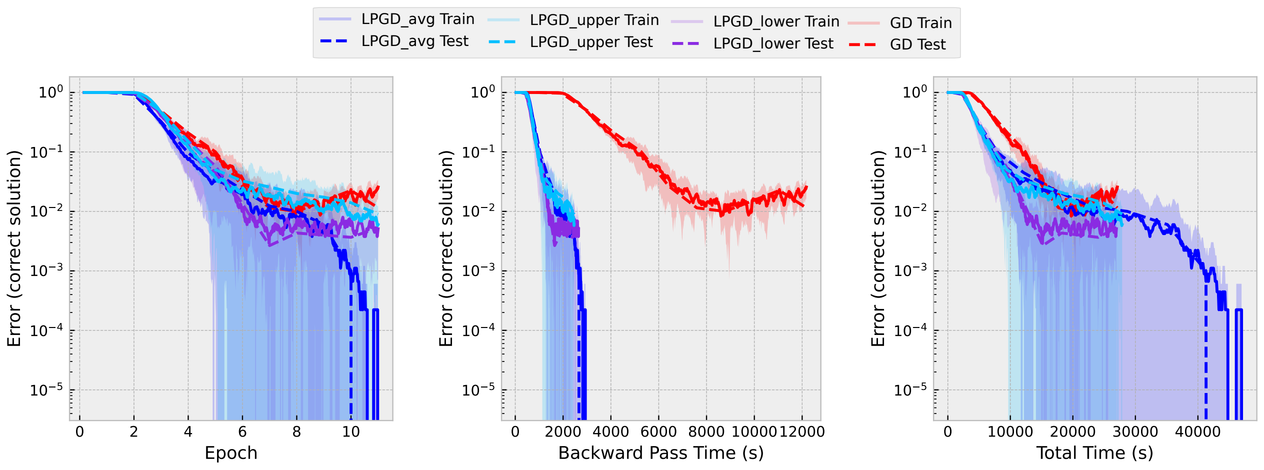

Figure 4. Results for the Sudoku experiment. Reported are the batched train and test error over 11 epochs. We additionally report the loss over total wallclock time, and total time spent in the backward passes. The error of an instance is zero if it is a valid solution to the sudoku, otherwise it is one.

C. Theoretical Considerations

C.1. Implicit Differentiation with Augmentation

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

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

$$(x^*, s^*) = \arg\min_{x, s \geq 0} \tfrac{1}{2} x^T H x + c^T y \quad \text{subject to} \quad Ax + b + s = 0. \tag{31}$$

In Lagrangian form, we can write it as

$$z^* = (x^*, s^*, y^*) = \arg \min_{x, s \geq 0} \max_y \mathcal{L}(H, c, A, b, x, y) \tag{32}$$

with the Lagrangian

$$\mathcal{L}(H, c, A, b, x, y) = x^T H x + c^T x + (Ax + b + s)^T y. \tag{33}$$

The augmentation in (28) augments the Lagrangian to

$$\mathcal{L}_\rho(H, c, A, b, x, y) = x^T H x + c^T x + (Ax + b + s)^T y + \tfrac{\rho}{2} \|x - x^*\|_2^2 \tag{34}$$

and we write

$$z_\rho^*(H, c, A, b) = \arg \min_{x, s \geq 0} \max_y \mathcal{L}_\rho(H, c, A, b, x, y). \tag{35}$$

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

$$Hx + A^T y + c + \rho(x - x^*) = 0 \qquad \operatorname{diag}(y) s = 0 \qquad Ax + b + s = 0 \qquad s \geq 0. \tag{36}$$

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

$$0 = F(x, s, y, H, c, A, b) = \begin{pmatrix} Hx + A^T y + c + \rho(x - x^*) \\ \operatorname{diag}(y) s \\ Ax + b + s \end{pmatrix} \tag{37}$$

which admits the use of the implicit function theorem. It states that under the regularity condition that $\frac{\partial F}{\partial z}$ 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

$$0 = \frac{\partial F}{\partial z} \frac{\partial z_\rho^*}{\partial w} + \frac{\partial F}{\partial w}. \tag{38}$$

with

$$\frac{\partial F}{\partial z} = \begin{bmatrix} H + \rho I & 0 & A^T \\ 0 & \operatorname{diag}(\frac{y}{s}) & I \\ A & I & 0 \end{bmatrix}. \tag{39}$$

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

$$\nabla_w \ell(x^*(w)) = \frac{\partial z^*}{\partial w}^T \nabla \ell(x^*) = -\frac{\partial F}{\partial w}^T \frac{\partial F}{\partial z}^{-T} \nabla \ell(x^*), \tag{40}$$

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

$$(x^*, s^*) = \arg\min_{x,s\in\mathcal{K}} c^T y \quad \text{subject to} \quad Ax + s + b = 0 \tag{41}$$

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

$$\mathcal{L}_\rho(A, b, c, x, s, y) = c^T x + (Ay + s + b)^T y \tag{42}$$

which allows an equivalent saddle point formulation given by

$$z^* = (x^*, s^*, y^*) = \arg\min_{x,s\in\mathcal{K}} \max_y \mathcal{L}(A, b, c, x, s, y). \tag{43}$$

The KKT optimality conditions are

$$A^T y + c = 0 \qquad\qquad Ax + s + b = 0 \qquad\qquad (s, y) \in \mathcal{K} \times \mathcal{K}^* \qquad\qquad s^T y = 0 \tag{44}$$

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

$$Q = Q(A, b, c) = \begin{bmatrix} 0 & A^T & c \\ -A & 0 & b \\ -c^T & -b^T & 0 \end{bmatrix} \tag{45}$$

is used in the homogenous self-dual embedding (O'Donoghue et al., 2016), 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

$$Q_\rho = Q_\rho(A, b, c) = \begin{bmatrix} \rho I & A^T & c \\ -A & 0 & b \\ -c^T & -b^T & 0 \end{bmatrix}. \tag{46}$$

We adjust the CVXPY implementation accordingly for our experiments.

C.2. 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

$$D_\phi(x, \widehat{x}) \coloneqq \phi(x) - \phi(\widehat{x}) - \langle x - \widehat{x}, \nabla\phi(\widehat{x})\rangle \tag{47}$$

The (left) Bregman-Moreau envelope (Bauschke et al., 2018) and the (left) Bregman-Moreau proximal map are defined as

$$\operatorname{env}^\phi_{\tau\ell}(\widehat{x}) \coloneqq \min_x \ell(x) + \tfrac{1}{\tau} D_\phi(x, \widehat{x}), \tag{48}$$

$$\begin{aligned} \operatorname{prox}^\phi_{\tau\ell}(\widehat{x}) &\coloneqq \arg\min_x \ell(x) + \tfrac{1}{\tau} D_\phi(x, \widehat{x}) \qquad (49)\\ &= \widehat{x} - \tau\nabla \operatorname{env}^\phi_{\tau\ell}(\widehat{x}). \qquad (50) \end{aligned}$$

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

$$\begin{aligned} x_{k+1} &= \arg\min_x \tilde{\ell}(x) + I_\mathcal{X}(x) + \tfrac{1}{\tau} D_\phi(x, x_k) \qquad (51)\\ &= \arg\min_{x\in\mathcal{X}} \langle x, \nabla\ell(x_k)\rangle + \tfrac{1}{\tau} D_\phi(x, x_k) \qquad (52) \end{aligned}$$

An equivalent form of this algorithm is

$$\eta_k = \nabla\phi(x_k) \tag{53}$$

$$\eta_{k+1} = \eta_k - \tau\nabla\ell(x_k) \tag{54}$$

$$\widetilde{x}_{k+1} = (\nabla\phi)^{-1}(\eta_{k+1}) \tag{55}$$

$$x_{k+1} = \arg\min_{x\in\mathcal{X}} D_\phi(x, \widetilde{x}_{k+1}) \tag{56}$$

which highlights that gradients are applied in the dual space of ϕ, and $\nabla\phi$ serves as the mapping between primal and ("mirrored") dual space.

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

$$\mathcal{L}_w(x) \coloneqq \sup_y \mathcal{L}(x, y, w) \tag{57}$$

and assume that $\mathcal{L}_w$ is strongly convex and continuously differentiable in x. As the key step, we define the distance-generating function (mirror map) as $\phi = \mathcal{L}_w$. 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

$$D_{\mathcal{L}_w}(x, \widehat{x}) = \mathcal{L}_w(x) - \mathcal{L}_w(\widehat{x}) - \langle x - \widehat{x}, \nabla\mathcal{L}_w(\widehat{x})\rangle \tag{58}$$

$$D_{\mathcal{L}_w}(x, x^*) \geq 0, \tag{59}$$

$$D_{\mathcal{L}_w}(x, x^*) = 0 \Leftrightarrow x = x^*(w) \quad \text{for } x \in \mathcal{X}. \tag{60}$$

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

$$\ell_\tau^{\mathcal{L}}(w) \coloneqq \operatorname{env}_{\tau\ell+I_\mathcal{X}}^{\mathcal{L}_w}(x^*) \tag{61}$$

$$= \min_x \ell(x) + I_\mathcal{X}(x) + \tfrac{1}{\tau} D_{\mathcal{L}_w}(x, x^*) \tag{62}$$

$$= \min_{x\in\mathcal{X}} \ell(x) + \tfrac{1}{\tau}(\mathcal{L}(x, w) - \mathcal{L}^*(w) - \langle x - x^*, \nabla_x\mathcal{L}(x^*, w)\rangle), \tag{63}$$

$$x_{\tau\ell}^{\mathcal{L}}(w) \coloneqq \operatorname{prox}_{\tau\ell+I_\mathcal{X}}^{\mathcal{L}_w}(x^*) \tag{64}$$

$$= \arg\min_x \ell(x) + I_\mathcal{X}(x) + \tfrac{1}{\tau} D_{\mathcal{L}_w}(x, x^*) \tag{65}$$

$$= \arg\min_{x\in\mathcal{X}} \ell(x) + \tfrac{1}{\tau}(\mathcal{L}(x, w) - \mathcal{L}^*(w) - \langle x - x^*, \nabla_x\mathcal{L}(x^*, w)\rangle). \tag{66}$$

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.

$$\nabla_w \ell_\tau^{\mathcal{L}}(w) = \tfrac{1}{\tau}\nabla_w[\mathcal{L}(x_{\tau\ell}^{\mathcal{L}}, w) - \mathcal{L}(x^*, w) - \langle x_{\tau\ell}^{\mathcal{L}} - x^*, \nabla_x\mathcal{L}(x^*, w)\rangle]. \tag{67}$$

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

$$x_{\tau\widetilde{\ell}}^{\mathcal{L}}(u, v) = \arg\inf_{x\in\mathcal{X}} \tau\langle x, \nabla\ell\rangle + \langle x, u\rangle + H(x, y, v) - \langle x, \nabla_x\mathcal{L}(x^*, w)\rangle \tag{68}$$

$$= x^*(u + \tau\nabla\ell - \nabla_x\mathcal{L}(x^*, w), v). \tag{69}$$

If $\mathcal{X} = \mathbb{R}^n$, then the original optimization problem (1) is unconstrained and we have the optimality condition $\nabla_x\mathcal{L}(x^*, w) = 0$. Therefore, the Bregman divergence

$$D_{\mathcal{L}_w}(x, x^*) = \mathcal{L}_w(x) - \mathcal{L}_w(x^*) - \langle x - x^*, \nabla\mathcal{L}_w(x^*)\rangle \tag{70}$$

$$= \mathcal{L}(x, w) - \mathcal{L}(x^*, w) = \mathcal{L}(x, w) - \mathcal{L}^*(w) = D_\mathcal{L}(x|w) \tag{71}$$

is equal to our Lagrangian divergence. It follows that, in this case, the Lagrange-Moreau envelope/LPGD is equivalent to the Lagrange-Bregman-Moreau envelope/LMD.

C.3. Extension to General Loss Functions

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

$$y^*(w) = \arg\sup_{y} \min_{x\in\mathcal{X}} \mathcal{L}(x, y, w) = \arg\inf_{y} \max_{x\in\mathcal{X}} -\mathcal{L}(x, y, w). \tag{72}$$

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

$$(\ell_p)_\tau(w) + (\ell_d)_\tau(w) = \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_p(x)] - \sup_{y} \inf_{x\in\mathcal{X}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_d(y)] \tag{73}$$

$$(\ell_d)^\tau(w) + (\ell_p)^\tau(w) = \sup_{y} \inf_{x\in\mathcal{X}} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_d(y)] - \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_p(x)]. \tag{74}$$

Assuming strong duality of these optimization problems, this leads to

$$(\ell_p)_\tau(w) + (\ell_d)_\tau(w) = \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_p(x)] - \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_d(y)] \tag{75}$$

$$(\ell_d)^\tau(w) + (\ell_p)^\tau(w) = \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_d(y)] - \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_p(x)] \tag{76}$$

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

$$(\ell_p)^{\tau}_{\tau}(w) + (\ell_d)^{\tau}_{\tau}(w) = \tfrac{1}{2}\{ \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_p(x)] - \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_d(y)] \tag{77}$$

$$+ \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + \ell_d(y)] - \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - \ell_p(x)]\} \tag{78}$$

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

$$L_\tau(w) \coloneqq \inf_{x\in\mathcal{X}} \sup_{y} L(x, y) + \tfrac{1}{\tau}[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)] \tag{79}$$

$$L^\tau(w) \coloneqq \inf_{x\in\mathcal{X}} \sup_{y} L(x, y) + \tfrac{1}{\tau}[\mathcal{L}(x, y, w) - \mathcal{L}^*(w)] \tag{80}$$

$$L^{\tau}_{\tau}(w) \coloneqq \tfrac{1}{2}\{L_\tau(w) + L^\tau(w)\} \tag{81}$$

$$= \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) + L(x, y)] - \inf_{x\in\mathcal{X}} \sup_{y} [\tfrac{1}{\tau}\mathcal{L}(x, y, w) - L(x, y)]. \tag{82}$$

Note that $L_\tau/L^\tau(w)$ are not necessarily lower/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

$$L_\tau(w) = \ell_\tau(w), \qquad L^{\tau}_{\tau}(w) = \ell^{\tau}_{\tau}(w), \qquad L^\tau(w) = \ell^\tau(w). \tag{83}$$

For the rest of the appendix, we will work with L instead of ℓ for full generality and reduce the situation to a primal loss ℓ with the relations above.

C.4. Recovery of True Gradient

We aim to show that for a linear loss approximation $\tilde{L}$, the LPGD update recovers the true gradient as τ approaches zero. We again expose the linear parameters of the Lagrangian

$$\mathcal{L}(z, u, v) = \langle z, u\rangle + H(z, v) \tag{84}$$

and get from Danskin's envelope theorem (Danskin, 1966)

$$\nabla_u \mathcal{L}^*(u, v) = \nabla_u \mathcal{L}(z^*, u, v) = z^*(u, v). \tag{85}$$

We define

$$\mathrm{d}w := \begin{pmatrix} \nabla_z L \\ 0 \end{pmatrix}. \tag{86}$$

Then it holds that

$$\begin{aligned}
\lim_{\tau \to 0} \nabla_w \tilde{L}_\tau(w) &= \lim_{\tau \to 0} \tfrac{1}{\tau}[\nabla_w \mathcal{L}^*(w + \tau \mathrm{d}w) - \nabla_w \mathcal{L}^*(w)] && (87)\\
&= \frac{\partial^2 \mathcal{L}^*}{\partial^2 w} \mathrm{d}w = \frac{\partial^2 \mathcal{L}^*}{\partial^2 w}^T \mathrm{d}w = \frac{\partial^2 \mathcal{L}^*}{\partial^2 w}^T \begin{pmatrix} \nabla_z L \\ 0 \end{pmatrix} && (88)\\
&= \frac{\partial^2 \mathcal{L}^*}{\partial w \partial u}^T \nabla_z L = \frac{\partial (\nabla_u \mathcal{L}^*)}{\partial w}^T \nabla_z L = \frac{\partial z^*}{\partial w}^T \nabla_z L = \nabla_w L(z^*(w)) && (89)
\end{aligned}$$

The main step in this derivation was to identify the Jacobian of the solution mapping as a sub-matrix of the Hessian of the optimal Lagrangian function, which is a symmetric matrix.[10] Exploiting the symmetry of the Hessian then allows computing the gradient, which is a co-derivative (backward-mode), as the finite difference between two solver outputs, which usually only computes a directional (forward-mode) derivative as

$$\begin{aligned}
\Delta z = \frac{\partial z^*}{\partial w} \Delta w &= \lim_{\tau \to 0} \tfrac{1}{\tau}[z^*(w + \tau \Delta w) - z^*(w)] && (90)\\
&= \lim_{\tau \to 0} \tfrac{1}{\tau}[\nabla_u \mathcal{L}^*(w + \tau \Delta w) - \nabla_u \mathcal{L}^*(w)]. && (91)
\end{aligned}$$

We observe that computing the gradient of the $\mathcal{L}$-envelope is the backward-mode counterpart of the forward-mode right-sided directional derivative. This observation also gives an interpretation of not taking the limit in theLPGD 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. The finite difference in the gradient of the $\mathcal{L}$-envelope serves the equivalent purpose in backward-mode, by back-propagating higher-order information instead of linear sensitivities as in standard directional co-derivatives (back-propagation).

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

$$\lim_{\tau \to 0} \nabla_w \tilde{\ell}_\tau(w) = \nabla_w \ell(x^*(w)). \tag{92}$$

An analogous proof and discussion also hold for the upper envelope $\tilde{\ell}^\tau$ and average envelope $\tilde{\ell}_\tau$, corresponding to the left- and double-sided directional derivatives.

C.5. Background: Set-valued Solution Mappings

In the main text, we assumed that the Lagrangian is strictly convex, which entails a unique solution to the optimization problem. We now turn to the more general case, in which the optimization problem does not have a unique solution i.e. the solution mapping is set-valued.

$$S : \mathbb{R}^m \rightrightarrows \mathbb{R}^n : w \mapsto S(w) = \{z^* \in \mathbb{R}^n \mid \mathcal{L}(z^*, w) = \mathcal{L}^*(w)\} \tag{93}$$

We still assume access to a solver oracle that returns a single element of the solution set $z^* \in S(w)$.

One possible approach to generalizing the concepts of derivatives and co-derivatives to set-valued mappings is the machinery of graphical differentiation described in detail in Rockafellar & Wets (1998).

As the name suggests, this approach relies on a graphical interpretation of the derivative and co-derivative mappings to generalize them from single-valued functions to set-valued mappings.

[10]Note that a similar derivation already appeared in (Domke, 2010), but only for primal variables without considering the benefits of finite values of τ over the limit.

Before we state the definition of graphical derivatives, we first develop a description of derivatives of single-valued functions in terms of graphs that can be generalized to set-valued functions. For a single-valued function $z^*(w)$ the directional derivative

$$Dz^*(w) : \mathbb{R}^m \to \mathbb{R}^n : \Delta w \mapsto \frac{\partial z^*}{\partial w}(w)\Delta w \tag{94}$$

is a function mapping perturbations of the input Δw to corresponding changes in the output. In a graphical interpretation, it gives a first-order local approximation of the function graph

$$\operatorname{Gph} z^* = \{(w, z) \mid z = z^*(w)\} \tag{95}$$

$$\operatorname{Gph} Dz^*(w) = \{(\Delta w, \Delta z) \mid \Delta z = \frac{\partial z^*}{\partial w}(w)\Delta w\}. \tag{96}$$

Likewise, the graph of the directional co-derivative

$$D^*z^*(w) : \mathbb{R}^n \to \mathbb{R}^m : \mathrm{d}z \mapsto \frac{\partial z^*}{\partial w}^T(w)\mathrm{d}z \tag{97}$$

is given by

$$\operatorname{Gph} D^*z^*(w) = \{(\mathrm{d}w, \mathrm{d}z) \mid \mathrm{d}w = \frac{\partial z^*}{\partial w}^T(w)\mathrm{d}z\} \tag{98}$$

We observe that up to a sign flip, the vectors in the graph of the derivative and the co-derivative are orthogonal

$$\begin{aligned}
\langle(\mathrm{d}w, -\mathrm{d}z), (\Delta w, \Delta z)\rangle &= \langle \mathrm{d}w, \Delta w\rangle - \langle -\mathrm{d}z, \Delta z\rangle && (99)\\
&= \langle \tfrac{\partial z^*}{\partial w}(\widehat{w})^T \mathrm{d}z, \Delta w\rangle - \langle \mathrm{d}z, \tfrac{\partial z^*}{\partial w}(\widehat{w})\Delta w\rangle && (100)\\
&= \langle \mathrm{d}z, \tfrac{\partial z^*}{\partial w}(\widehat{w})\Delta w\rangle - \langle \mathrm{d}z, \tfrac{\partial z^*}{\partial w}(\widehat{w})\Delta w\rangle && (101)\\
&= 0 && (102)
\end{aligned}$$

We can, therefore, equivalently define the co-derivative as the function with the graph

$$\operatorname{Gph} D^*z^*(w) = \{(\mathrm{d}w, \mathrm{d}z) \mid (\mathrm{d}w, -\mathrm{d}z) \perp \operatorname{Gph} Dz^*(w)\}. \tag{103}$$

These properties can now be generalized to graphical derivatives of set-valued mappings (Rockafellar & Wets, 1998, Definition 8.33). A local approximation to the graph of the set-valued mapping gives the graph of the directional derivative. In general, a local approximation of a set C at x is given by its tangent cone $T_S(x)$ (Rockafellar & Wets, 1998, Definition 6.1), therefore the graphical derivative is given by

$$DS(w \mid z) : \mathbb{R}^m \rightrightarrows \mathbb{R}^n : \Delta w \mapsto \{\Delta z \mid (\Delta w, \Delta z) \in T_{\operatorname{Gph} S}(w, z)\} \tag{104}$$

$$\operatorname{Gph} DS(w \mid z^*) = T_{\operatorname{Gph} S}(w, z^*) \tag{105}$$

The graphical co-derivative is obtained as all the vectors orthogonal to $\operatorname{Gph} DS(w \mid z)$ after a sign flip in the second component. The vectors orthogonal to a tangent cone $T_S(x)$ are described by the regular normal cone $\widehat{N}_S(x)$ (Rockafellar & Wets, 1998, Definition 6.3), (Rockafellar & Wets, 1998, Proposition 6.5). Therefore, we can express the graphical co-derivative as

$$D^*S(w \mid z^*) : \mathbb{R}^n \rightrightarrows \mathbb{R}^m : \mathrm{d}z \mapsto \{\mathrm{d}w \mid (\mathrm{d}w, -\mathrm{d}z) \in \widehat{N}_{\operatorname{Gph} S}(w, z^*)\}. \tag{106}$$

We can back-propagate sub-differentials using the co-derivative mapping chain rule

$$\partial_w L(S(w \mid z^*)) = D^*S(w \mid z^*)(\nabla L(z^*)) \tag{107}$$

C.6. Recovery of True Sub-differential

Our goal is to show that the sub-differential of the Lagrange-Moreau envelope of the linearized loss $\partial_{(u,v)}\tilde{L}_\tau(u, v \mid z^*)$ converges in a set-valued sense to the sub-differential of the loss $\partial_{(u,v)}L(S(u, v \mid z^*))$. We will show that this only holds in the form of an inclusion.

Define the gradient of the optimal Lagrangian with respect to the non-linear parameters at the selected optimal solution as

$$g^* := \nabla_v \mathcal{L}(z^*, u, v). \tag{108}$$

We can then compactly write

$$(z^*, g^*) = \nabla_{(u,v)}\mathcal{L}(z^*, u, v) \in \partial_{(u,v)}\mathcal{L}^*(u, v) =: S^+(u, v) \tag{109}$$

where we defined the *extended* solution mapping $S^+ : \mathbb{R}^k \mapsto \mathbb{R}^k$, which contains the gradient with respect to v in addition to the optimal solution set.

The lower Lagrange-Moreau envelope of the linearized loss at z^* is defined as

$$\begin{aligned}
\tilde{L}_\tau(u, v \mid z^*) &= \inf_{x\in\mathcal{X}} \sup_y \langle z, \nabla L(z^*)\rangle + \tfrac{1}{\tau}[\mathcal{L}(z, u, v) - \mathcal{L}(z^*, u, v)] && (110)\\
&= \tfrac{1}{\tau}[\inf_{x\in\mathcal{X}} \sup_y \langle z, u + \tau\nabla L(z^*)\rangle + H(z, v) - \mathcal{L}(z^*, u, v)] && (111)\\
&= \tfrac{1}{\tau}[\mathcal{L}^*(u + \tau\nabla L(z^*), v) - \mathcal{L}(z^*, u, v)] && (112)
\end{aligned}$$

Note that we make the dependence on z^* explicit, as z^* is not uniquely determined anymore by (u, v). With a slight abuse of set-valued notation, the sub-differential of the Lagrange-Moreau envelope can again be written as the finite difference between two extended solver solutions

$$\begin{aligned}
\partial_{(u,v)}\tilde{L}_\tau(u, v \mid z^*) &= \tfrac{1}{\tau}[\partial_{(u,v)}\mathcal{L}^*(u + \tau\nabla L(z^*), v) - \nabla_{(u,v)}\mathcal{L}(z^*, u, v)] && (113)\\
&= \tfrac{1}{\tau}[S^+(u + \tau\nabla L(z^*), v) - (z^*, g^*)] && (114)\\
&\xrightarrow[\tau\to 0]{} DS^+(u, v \mid z^*, g^*)(\nabla L(z^*), 0). && (115)
\end{aligned}$$

In the final step, we used that the finite difference converges to the graphical derivative (Rockafellar & Wets, 1998, Eq. 8(14)).

Next, we relate the graphical derivative to the desired graphical co-derivative in (107). A promising statement in this direction is given in (Rockafellar & Wets, 1998, Theorem 13.57), which states that under some regularity conditions, the graphical derivative of a sub-differential mapping is contained in its graphical co-derivative. We reduce our situation to this case by phrasing the solution mapping as a sub-differential mapping. To see this, we inspect the graphs of the solution mapping

$$\begin{aligned}
\operatorname{Gph} S^+ &= \{(u, v \mid z^*, g^*) \mid 0 \in \partial_z(u^T z^* + H(z^*, v) + I_z(z^*)), g^* \in \nabla_v H(z^*, v) + \partial I_{\mathbb{R}^m}(v)\} && (116)\\
&= \{(u, v \mid z^*, g^*) \mid (-u, g) \in \partial(H + I_C)(z, v)\} && (117)
\end{aligned}$$

and the graph of the sub-differential

$$\begin{aligned}
\operatorname{Gph} \partial(H + I_C) &= \{(z^*, v \mid u, g^*) \mid (u, g^*) \in \partial(H + I_C)(z^*, v)\} && (118)\\
&= \{(z^*, v \mid -u, g^*) \mid (-u, g^*) \in \partial(H + I_C)(z^*, v)\}. && (119)
\end{aligned}$$

We observe that the graphs are equal up to permutation and sign flips of the coordinates. This allows us to write

$$\begin{aligned}
(\mathrm{d}u, \mathrm{d}v) \in D^*S^+(u, v \mid z^*, g^*)(\mathrm{d}z, 0) &\Leftrightarrow (\mathrm{d}u, \mathrm{d}v, -\mathrm{d}z, 0) \in \mathcal{N}_{\operatorname{Gph} S^+}(u, v \mid z^*, g^*) && (120)\\
&\Leftrightarrow (-\mathrm{d}z, \mathrm{d}v, -\mathrm{d}u, 0) \in \mathcal{N}_{\operatorname{Gph} \partial(H+I_C)}(z^*, v \mid -u, g^*) && (121)\\
&\Leftrightarrow (-\mathrm{d}z, \mathrm{d}v) \in D^*\partial(H + I_C)(z^*, v \mid -u, g^*)(\mathrm{d}u, 0) && (122)\\
&\Leftarrow (-\mathrm{d}z, \mathrm{d}v) \in D\partial(H + I_C)(z^*, v \mid -u, g^*)(\mathrm{d}u, 0) && (123)\\
&\Leftrightarrow (\mathrm{d}u, 0, -\mathrm{d}z, \mathrm{d}v) \in T_{\operatorname{Gph} \partial(H+I_C)}(z^*, v \mid -u, g^*) && (124)\\
&\Leftrightarrow (\mathrm{d}z, 0, \mathrm{d}u, \mathrm{d}v) \in T_{\operatorname{Gph} S^+}(u, v \mid z^*, g^*) && (125)\\
&\Leftrightarrow (\mathrm{d}u, \mathrm{d}v) \in DS^+(u, v \mid z^*, g^*)(\mathrm{d}z, 0) && (126)
\end{aligned}$$

The key step in this derivation (the implication) is precisely the statement of (Rockafellar & Wets, 1998, Theorem 13.57). It requires $H + I_C$ to be prox-regular and subdifferentilly continuous at (z^*, u) for $(-u, g^*)$. This is the case for all proper, lower semi-continuous, convex functions (Rockafellar & Wets, 1998, Example 13.30), which also includes indicator functions of convex sets. As C is convex and H is assumed to be convex in z and v and affine in y, these conditions are fulfilled, and the statement holds.

Finally, this gives us the desired result

$$\partial_{(u,v)}\tilde{L}_\tau(u, v \mid z^*) \xrightarrow[\tau\to 0]{} DS^+(u, v \mid z^*, g^*)(\nabla L(z^*), 0) \subseteq D^*S^+(u, v \mid z^*, g^*)(\nabla L(z^*), 0) \tag{127}$$

$$= D^*S(u, v \mid z^*)(\nabla L(z^*)) \tag{128}$$

$$= \partial_{(u,v)} L(S(u, v \mid z^*)). \tag{129}$$

For a primal loss $L(x, y) = \ell(x)$ this reduces to

$$\partial_{(c,v)}\tilde{\ell}_\tau(c, v \mid x^*) \xrightarrow[\tau\to 0]{} DS^+(c, v \mid x^*, g^*)(\nabla \ell(x^*), 0) \subseteq D^*S^+(c, v \mid x^*, g^*)(\nabla \ell(x^*), 0) \tag{130}$$

$$= D^*S(c, v \mid x^*)(\nabla \ell(x^*)) \tag{131}$$

$$= \partial_{(c,v)} \ell(S(c, v \mid x^*)). \tag{132}$$