On Implicit Bias in Overparameterized Bilevel Optimization

Paul Vicol [1 2] **Jonathan Lorraine** [1 2] **Fabian Pedregosa** [3] **David Duvenaud** [1 2] **Roger Grosse** [1 2]

Abstract

Many problems in machine learning involve bilevel optimization (BLO), including hyperparameter optimization, meta-learning, and dataset distillation. Bilevel problems involve inner and outer parameters, each optimized for its own objective. Often, at least one of the two levels is underspecified and there are multiple ways to choose among equivalent optima. Inspired by recent studies of the implicit bias induced by optimization algorithms in single-level optimization, we investigate the implicit bias of different gradient-based algorithms for jointly optimizing the inner and outer parameters. We delineate two standard BLO methods—cold-start and warm-start BLO—and show that the converged solution or long-run behavior depends to a large degree on these and other algorithmic choices, such as the hypergradient approximation. We also show that the solutions from warm-start BLO can encode a surprising amount of information about the outer objective, even when the outer optimization variables are low-dimensional. We believe that implicit bias deserves as central a role in the study of bilevel optimization as it has attained in the study of single-level neural net optimization.

1. Introduction

Bilevel optimization (BLO) problems consist of two nested sub-problems, called the *outer* and *inner* problems, where the outer problem must be solved subject to the optimality of the inner problem. Let $\mathbf{u} \in \mathcal{U}$, $\mathbf{w} \in \mathcal{W}$ denote the outer and inner parameters, and let $F, f : \mathcal{U} \times \mathcal{W} \to \mathbb{R}$ denote the outer and inner objectives, respectively. The BLO problem is defined as:

$$\mathbf{u}^\star \in \text{``}\operatorname*{arg\,min}_{\mathbf{u} \in \mathcal{U}}\text{''} F(\mathbf{u}, \mathbf{w}^\star) \tag{1}$$

$$\mathbf{w}^\star \in \mathcal{S}(\mathbf{u}^\star) = \operatorname*{arg\,min}_{\mathbf{w} \in \mathcal{W}} f(\mathbf{u}^\star, \mathbf{w}) \tag{2}$$

[1]University of Toronto [2]Vector Institute [3]Google Research. Correspondence to: Paul Vicol <pvicol@cs.toronto.edu>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

where $\mathcal{S} : \mathcal{U} \rightrightarrows \mathcal{W}$ is a set-valued *response mapping*, from each outer parameter $\mathbf{u}$ to a set of optimal inner parameters $\mathcal{S}(\mathbf{u})$. We use quotes around the outer "$\arg\min$" to denote the ambiguity in the definition of the bilevel problem when the inner objective has multiple solutions, a point we expand upon in Section 2. Examples of bilevel optimization in machine learning include hyperparameter optimization (Domke, 2012; Maclaurin et al., 2015; MacKay et al., 2019; Lorraine et al., 2020; Shaban et al., 2019), dataset distillation (Wang et al., 2018; Zhao et al., 2021), influence function estimation (Koh & Liang, 2017), meta-learning (Finn et al., 2017; Franceschi et al., 2018; Rajeswaran et al., 2019), example reweighting (Bengio et al., 2009; Ren et al., 2018), neural architecture search (Zoph & Le, 2017; Liu et al., 2019) and adversarial learning (Goodfellow et al., 2014; Pfau & Vinyals, 2016) (see Table 1).

Most algorithmic contributions to BLO make the simplifying assumption that the inner problem has a unique solution (that is, $|\mathcal{S}(\mathbf{u})| = 1, \forall \mathbf{u} \in \mathcal{U}$), and give approximation methods that provably get close to the solution (Shaban et al., 2019; Pedregosa, 2016; Yang et al., 2021; Hong et al., 2020; Grazzi et al., 2020a). In practice, however, these algorithms are often run in settings where either the inner or outer problem is *underspecified*; i.e., the set of optima forms a non-trivial manifold. Underspecification often occurs due to overparameterization, e.g., given a learning problem with more parameters than datapoints, there are typically infinitely many solutions that achieve the optimal objective value (Belkin, 2021). Analyses of the dynamics of overparameterized single-objective optimization have shown that the nature of the converged solution, such as its pattern of generalization, depends in subtle ways on the details of the optimization dynamics (Lee et al., 2019; Arora et al., 2019; Bartlett et al., 2020; Amari et al., 2020); this general phenomenon is known as *implicit bias*. We extend this investigation to the bilevel setting: What are the implicit biases of practical BLO algorithms? What types of solutions do they favor, and what are the implications for how the trained models generalize?

We identify two sources of implicit bias in underspecified BLO: 1) implicit bias resulting from the optimization algorithm, either cold-start or warm-start BLO (described below); and 2) implicit bias resulting from the hypergradient approximation. Classic literature on bilevel optimiza-

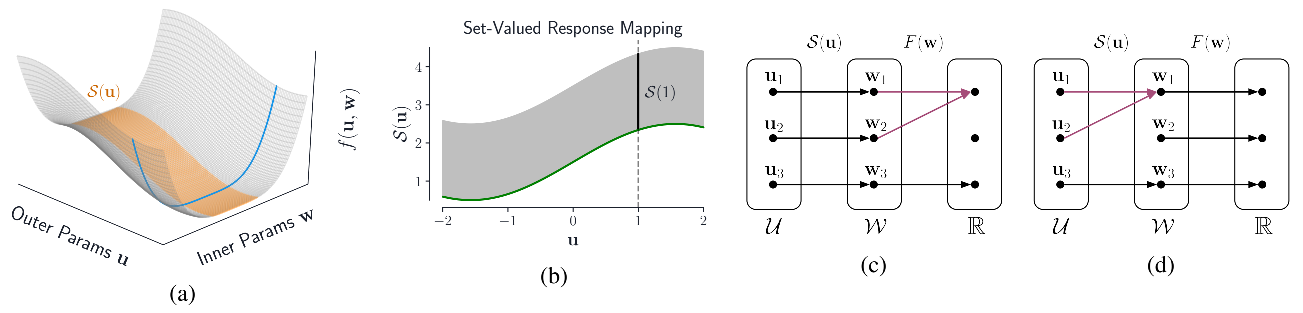


Figure 1: **Underspecification in BLO. (a)** Simplified visualization of inner underspecification, yielding a manifold of optimal solutions $\mathcal{S}(\mathbf{u}) = \arg\min_{\mathbf{w}} f(\mathbf{u},\mathbf{w})$ for each $\mathbf{u}$ (a slice is highlighted in blue for a fixed $\mathbf{u}$). The orange region along the floor of the valley highlights the set-valued best-response mapping. **(b)** The set-valued best-response for $\mathbf{u},\mathbf{w} \in \mathbb{R}$, where the shaded gray region contains values of $\mathbf{w}$ that achieve equivalent performance on the inner objective. The green curve highlights the minimum-norm inner solutions for each $\mathbf{u}$. **(c)** Outer underspecification due to F mapping a range of inner parameters to the same value; or **(d)** from $\mathcal{S}(\mathbf{u})$ associating a range of outer parameters with the same inner parameters.

Task	Inner (w)	Outer (u)	InnerU	OuterU
Data Distil.	NN weights	Synthetic data	✓	✗
Data Aug.	NN weights	Aug params	✓	✗
GANs	Discriminator	Generator	✓	✓
Meta-Learn.	NN weights	NN weights	✓	✓
NAS	NN weights	Architectures	✓	✓
Hyperopt	NN weights	Hyperparams	✓	✗
Example reweighting	NN weights	Example weights	✓	✓

Table 1: **Inner and outer overparameterization in common bilevel tasks.** For each task, we reference whether the common use-case includes inner/outer underspecification (InnerU/OuterU).

tion (Dempe, 2002) considers two ways to break ties between solutions in the set $\mathcal{S}(\mathbf{u})$: optimistic and pessimistic equilibria (discussed in Section 2). However, as we will show, neither describes the solutions obtained by the most common BLO algorithms. We focus on two algorithms that are relevant for practical machine learning, which we term *cold-start* and *warm-start* BLO. Cold-start BLO (Maclaurin et al., 2015; Metz et al., 2019; Micaelli & Storkey, 2020) defines the inner solution $\mathbf{w}^\star$ as the fixpoint of an update rule, which captures the notion of running an optimization algorithm to convergence. For each outer update, the inner parameters are re-initialized to $\mathbf{w}_0$ and the inner optimization is run from scratch. This is computationally expensive, as it requires full unrolls of the inner optimization for each outer step. Warm-start BLO is a tractable and widely-used alternative (Luketina et al., 2016; MacKay et al., 2019; Lorraine et al., 2020; Tang et al., 2020) that alternates gradient descent steps on the inner and outer parameters: in each step of warm-start BLO, the inner parameters are optimized from their *current values*, rather than the initialization $\mathbf{w}_0$.

In Section 3, we characterize solution concepts that capture the behavior of the cold- and warm-start BLO algorithms and show that, for quadratic inner and outer objectives, cold-start BLO yields minimum-norm outer parameters. In Section 4, we show that warm-start BLO induces an implicit bias on the inner parameter iterates, regularizing the updates to maintain proximity to previous solutions.

In addition to the BLO algorithm, another source of implicit bias is the hypergradient approximation used. In Sections 3 and 4, we investigate the effect of using approximate implicit differentiation to compute the hypergradient $\frac{dF}{d\mathbf{u}}$, where different approximations can lead to vastly different outer solutions.

Inner Underspecification. Most bilevel tasks in machine learning involve training a neural network in the inner level, which typically yields an underspecified problem, as shown in Table 1. Figure 1(a,b) illustrates set-valued response mappings in underspecified BLO.

Outer Underspecification. Outer underspecification can arise in two ways: 1) the mapping $\mathcal{S}(\mathbf{u})$ is a function (e.g., not set-valued) and maps a range of outer parameters to the same inner parameter; or 2) the outer objective F maps a range of inner parameters to the same objective value. These two pathways are illustrated in Figure 1(c,d).

2. Background

In this section, we provide an overview of key concepts that we build on.[1]

Hypergradient Approximation. We assume that F and f are differentiable, and consider gradient-based BLO algorithms, which require the hypergradient $\frac{dF(\mathbf{u},\mathbf{w}^\star)}{d\mathbf{u}} = \frac{\partial F(\mathbf{u},\mathbf{w}^\star)}{\partial \mathbf{u}} + \left(\frac{d\mathbf{w}^\star}{d\mathbf{u}}\right)^\top \frac{\partial F(\mathbf{u},\mathbf{w}^\star)}{\partial \mathbf{w}^\star}$. The main challenge to computing the hypergradient lies in computing the *response Jacobian* $\frac{d\mathbf{w}^\star}{d\mathbf{u}}$, which captures how the converged inner parameters depend on the outer parameters. Exactly computing this Jacobian is often intractable for large-scale problems. The two most common approaches to approximate it are: 1) *iterative differentiation*, which unrolls the inner optimization to reach an approximate best-response (BR) and backpropagates through the unrolled computation graph to compute the approximate BR Jacobian (Domke, 2012;

[1]We provide a table of notation in Appendix A.

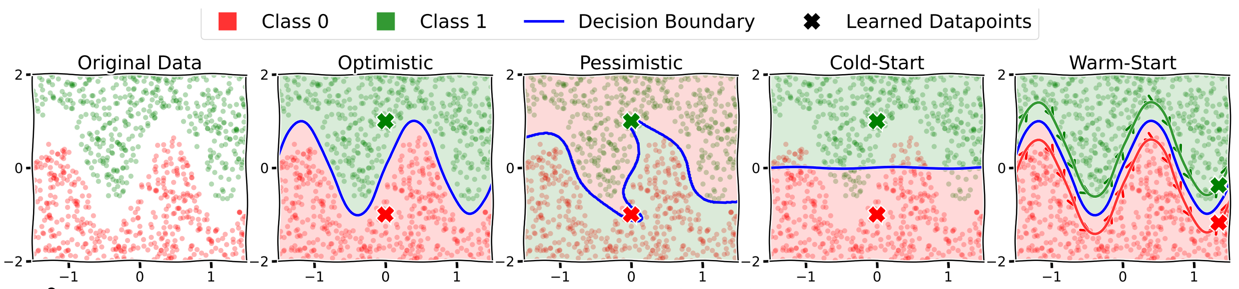


Figure 2: **Dataset distillation sketch illustrating four solution concepts for BLO: optimistic, pessimistic, cold-start, and warm-start.** We distill the dataset consisting of red and green points into two synthetic datapoints denoted by ✖ and ✖ (one per class). The learned datapoints ✖ and ✖ are the *outer parameters* and the inner parameters correspond to a model (classifier) trained on these synthetic datapoints. We assume an overparameterized inner model, which can fit the synthetic datapoints with many different decision boundaries. All of the solutions here correctly classify the synthetic datapoints (and are thus valid solutions to the inner problem) but they differ in their behavior on the original dataset. **Optimistic:** finds the decision boundary that correctly classifies all the original datapoints; **Pessimistic:** finds the decision boundary that achieves the worst loss on the original datapoints—it correctly classifies the synthetic datapoints but *incorrectly classifies* the original datapoints (note the *flipped red/green shading* on either side of the decision boundary); **Cold-start:** finds the min-norm solution that correctly classifies the synthetic datapoints; **Warm-start:** yields a trajectory of synthetic datapoints over time, which can allow for a model trained on two learned datapoints to fit the original data.

Maclaurin et al., 2015; Shaban et al., 2019); and 2) *implicit differentiation*, which leverages the implicit function theorem (IFT) to compute the BR Jacobian, assuming that the inner parameters are at a stationary point of the inner objective (Larsen et al., 1996; Chen & Hagan, 1999; Bengio, 2000; Foo et al., 2008; Pedregosa, 2016; Lorraine et al., 2020; Hataya et al., 2020b; Raghu et al., 2020). The IFT states that, assuming uniqueness of the inner solution and under certain regularity conditions, we can compute the response Jacobian as $\frac{\partial \mathbf{w}^\star(\mathbf{u})}{\partial \mathbf{u}} = \left(\frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top}\right)^{-1} \frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{u}}$. The term inside the inverse is the Hessian of the inner objective, which is typically intractable to store or invert directly (e.g., neural nets can easily have millions of parameters). Thus, multiplication by the inverse Hessian is typically approximated using an iterative linear solver, such as truncated conjugate gradient (CG) (Pedregosa, 2016), GMRES (Blondel et al., 2021), or the Neumann series (Liao et al., 2018; Lorraine et al., 2020). The (un-truncated) Neumann series for an invertible Hessian $\frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top}$ is defined as:

$$\left(\frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top}\right)^{-1} = \alpha \sum_{j=0}^{\infty} \left(\mathbf{I} - \alpha \frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top}\right)^j . \quad (3)$$

This series is known to converge when the largest eigenvalue of $\mathbf{I} - \alpha \frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top}$ is < 1. In practice, the Neumann series is typically truncated to the first K terms. Lorraine et al. (2020) showed that differentiating through i steps of unrolling starting from optimal inner parameters $\mathbf{w}^\star$ is equivalent to approximating the inverse Hessian with the first i terms in the Neumann series, a result we review in Appendix F. Approximate implicit differentiation can be implemented with efficient Hessian-vector products using modern autodiff libraries (Pearlmutter, 1994; Abadi et al., 2015; Paszke et al., 2017; Bradbury et al., 2018). However, when the inner problem is overparameterized, the Hessian $\frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top}$ is singular. In Section 3, we discuss how the K-term truncated Neumann series approximates the inverse of the *damped Hessian* $(\mathbf{H} + \epsilon \mathbf{I})^{-1}$, and we discuss the implications of this approximation.

Optimistic and Pessimistic BLO. For scenarios where the inner problem has multiple solutions, i.e., $|\mathcal{S}(\mathbf{u})| > 1$, two solution concepts have been widely studied in the classical BLO literature: the *optimistic* (Sinha et al., 2017; Dempe et al., 2007; Dempe, 2002; Harker & Pang, 1988; Lignola & Morgan, 1995; 2001; Outrata, 1993) and *pessimistic* (Sinha et al., 2017; Dempe, 2002; Dempe et al., 2014; Loridan & Morgan, 1996; Lucchetti et al., 1987; Wiesemann et al., 2013; Liu et al., 2018; 2020) solutions.[2] In *optimistic* BLO, $\mathbf{w}$ is chosen such that it leads to the best outer objective value. In contrast, in *pessimistic* BLO, $\mathbf{w}$ is chosen such that it leads to the worst outer objective value. The corresponding equilibria are defined below, with the differences highlighted in purple and teal.

Definition 2.1. *Let $\mathcal{S} : \mathcal{U} \rightrightarrows \mathcal{W}$ be the set-valued response mapping $\mathcal{S}(\mathbf{u}) = \arg\min_{\mathbf{w} \in \mathcal{W}} f(\mathbf{u}, \mathbf{w})$. Then $(\mathbf{u}^\star, \mathbf{w}^\star)$ is an* optimistic/pessimistic equilibrium *if:*

$$\mathbf{u}^\star \in \arg\min_{\mathbf{u}} F(\mathbf{u}, \mathbf{w}^\star) \text{ s.t. } \mathbf{w}^\star \in \arg \min/\max_{\mathbf{w} \in \mathcal{S}(\mathbf{u}^\star)} F(\mathbf{u}^\star, \mathbf{w})$$

Dataset Distillation. Given an original (typically large) dataset, the task of *dataset distillation* (Maclaurin et al., 2015; Wang et al., 2018; Lorraine et al., 2020) is to learn a smaller *synthetic dataset* such that a model trained on the synthetic data will generalize well to the original data. In this problem, the inner parameters are the weights of a model, and the outer parameters are the synthetic datapoints. The inner objective is the loss of the inner model trained on the synthetic datapoints, and the outer objective is the loss

[2]See Sinha et al. (2017) for a comprehensive review.

of the inner model on the original dataset:

$$\mathbf{u}^\star \in \text{“}\arg\min_{\mathbf{u}}\text{”} \mathcal{L}(\mathbf{w}^\star(\mathbf{u}), \mathcal{D}_{\text{original}}), \tag{4}$$
$$\mathbf{w}^\star(\mathbf{u}) \in \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \mathcal{D}(\mathbf{u})) \tag{5}$$

Here, $\mathcal{L}$ denotes a loss function, $\mathcal{D}_{\text{original}}$ denotes the dataset we aim to distill, and $\mathcal{D}(\mathbf{u})$ denotes a synthetic dataset parameterized by $\mathbf{u}$. We focus on dataset distillation throughout this paper because the degree of inner and outer overparameterization can be modified by varying the number of original versus synthetic datapoints. The solutions for this task are also easy to visualize and interpret.

Visualizing Solution Concepts. Figure 2 illustrates four different solution concepts for a toy 2D problem: the optimistic and pessimistic solutions, as well as two new solution concepts—cold- and warm-start equilibria—defined in Section 3. We consider dataset distillation for binary classification, with red and green datapoints representing the two classes, and a sinusoidal ground-truth decision boundary. We distill this dataset into two synthetic datapoints, represented by ✖ and ✖. In each subplot, the blue curve shows the decision boundary of the inner classifier, and the background colors show which class is predicted on each side of the decision boundary. See the caption of Figure 2 for a description of each solution concept on this task.

The optimistic solution can be tractable to compute (Mehra & Hamm, 2019; Ji & Liang, 2021), while the pessimistic solution is known to be less tractable (Sinha et al., 2017). From Figure 2, however, it is unclear whether either of these solution concepts is useful. For example, in hyperparameter optimization, the optimistic and pessimistic solutions correspond to choosing hyperparameters such that the inner training loop achieves minimum or maximum performance on the validation set, and it is unclear why this would be beneficial. Most widely-used BLO algorithms do not aim to find either of these solution concepts, which motivates our study of the behavior of the algorithms used in practice.

3. Equilibrium Concepts

We aim to understand the behavior of two popular BLO algorithms: *cold-start* (App. H, Algorithm 1) and *warm-start* (App. H, Algorithm 2). A summary of the updates for each algorithm is shown in Table 2. In this section, we first define equilibrium notions that capture the solutions found by warm-start and cold-start BLO. We then discuss properties of these equilibria, in particular focusing on the norms of the resulting inner and outer parameters. Finally, we analyze the implicit bias induced by approximating the hypergradient with the truncated Neumann series.

Method	Inner Update
Full Cold-Start	$\mathbf{w}^\star_{k+1} = \Xi^{(\infty)}(\mathbf{u}_{k+1}, \mathbf{w}_0)$
Full Warm-Start	$\mathbf{w}^\star_{k+1} = \Xi^{(\infty)}(\mathbf{u}_{k+1}, \mathbf{w}^\star_k)$
Partial Warm-Start	$\mathbf{w}^\star_{k+1} = \Xi^{(T)}(\mathbf{u}_{k+1}, \mathbf{w}^\star_k)$

Table 2: Inner parameter updates for cold-start, full warm-start, and partial warm-start BLO.

3.1. Cold-Start Equilibrium

Here, we introduce a solution concept that captures the behavior of cold-start BLO. We consider an iterative optimization algorithm used to compute an approximate solution to the inner objective. We denote a step of inner optimization by $\mathbf{w}_{t+1} = \Xi(\mathbf{u}, \mathbf{w}_t)$; for gradient descent, we have $\Xi(\mathbf{u}, \mathbf{w}_t) = \mathbf{w}_t - \alpha \nabla_{\mathbf{w}} f(\mathbf{u}, \mathbf{w}_t)$. We denote K steps of inner optimization from $\mathbf{w}_0$ by $\Xi^{(K)}(\mathbf{u}, \mathbf{w}_0)$. Under certain assumptions (e.g., that f has a unique finite root and that the step size α for the update is chosen appropriately), repeated application of Ξ will converge to a fixpoint. We denote the fixpoint for an initialization $\mathbf{w}_0$ by $\Xi^{(\infty)}(\mathbf{u}, \mathbf{w}_0)$.

Definition 3.1. *Let* $\mathbf{r}(\mathbf{u}, \mathbf{w}) \triangleq \Xi^{(\infty)}(\mathbf{u}, \mathbf{w})$. *Then* $(\mathbf{u}^\star, \mathbf{w}^\star)$ *is a cold-start equilibrium for an initialization* $\mathbf{w}_0$ *if:*

$$\mathbf{u}^\star \in \arg\min_{\mathbf{u}} F(\mathbf{u}, \mathbf{w}^\star) \quad \textit{s.t.} \quad \mathbf{w}^\star = \mathbf{r}(\mathbf{u}^\star, \mathbf{w}_0)$$

In some cases, we can compute the fixpoint of the inner optimization analytically. In particular, when f is quadratic, the analytic solution minimizes the displacement from $\mathbf{w}_0$: $\Xi^{(\infty)}(\mathbf{u}, \mathbf{w}_0) = \arg\min_{\mathbf{w}\in\mathcal{S}(\mathbf{u})} \|\mathbf{w} - \mathbf{w}_0\|_2^2$.

3.2. Warm-Start Equilibrium

Next, we introduce a solution concept intended to capture the behavior of warm-start BLO. Warm-starting refers to initializing the inner optimization from the inner parameters obtained in the previous hypergradient computation. One can consider two variants of warm-starting: (1) using *full inner optimization*, that is, running the inner optimization to convergence starting from $\mathbf{w}_t$ to obtain the next iterate $\mathbf{w}_{t+1}$; or (2) *partial inner optimization*, where we approximate the solution to the inner problem via a few gradient steps. Full warm-start can be expressed as computing $\mathbf{w}_{k+1} = \Xi^{(\infty)}(\mathbf{u}, \mathbf{w}_k)$ in each iteration, starting from the previous inner solution $\mathbf{w}_k$ rather than $\mathbf{w}_0$. Similarly, partial warm-start can be expressed as $\mathbf{w}_k = \Xi^{(T)}(\mathbf{u}, \mathbf{w}_k)$, where T is often small (e.g., $T < 10$).

Definition 3.2. *Let* $\mathbf{r}(\mathbf{u}, \mathbf{w}) \triangleq \Xi^{(T)}(\mathbf{u}, \mathbf{w})$. *Then* $(\mathbf{u}^\star, \mathbf{w}^\star)$ *is a (full or partial) warm-start equilibrium if:*

$$\mathbf{u}^\star \in \arg\min_{\mathbf{u}} F(\mathbf{u}, \mathbf{w}^\star) \quad \textit{s.t.} \quad \mathbf{w}^\star = \mathbf{r}(\mathbf{u}^\star, \mathbf{w}^\star)$$

For finite T*, the solution is a partial warm-start equilibrium. As* $T \to \infty$*, we obtain the full warm-start equilibrium.*

3.3. Solution Properties

In this section, we examine properties of cold-start and warm-start equilibria—in particular, we analyze the norms of the inner and outer parameters given by different methods. First, we show that cold-start equilibria (with exact hypergradient computation) yield simple solutions for both the inner and outer problems: when F and f are quadratic, cold-start yields minimum-norm parameters at both levels. Then, we analyze the implicit bias induced by the hypergradient approximation, in particular by using the truncated Neumann series to estimate the inverse Hessian for implicit differentiation. To make the analysis tractable, we make the following assumption:

Assumption 3.1 (Quadratic Objectives)**.** *We assume that both the inner and outer objectives, f and F, are convex lower-bounded quadratics. We let:*

$$f(\mathbf{u},\mathbf{w}) = \frac{1}{2}\begin{bmatrix}\mathbf{w}^\top & \mathbf{u}^\top\end{bmatrix}\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^\top & \mathbf{C}\end{bmatrix}\begin{bmatrix}\mathbf{w}\\ \mathbf{u}\end{bmatrix} + \mathbf{d}^\top\mathbf{w} + \mathbf{e}^\top\mathbf{u} + c$$

where $\mathbf{A} \in \mathbb{R}^{|\mathcal{W}|\times|\mathcal{W}|}$ is positive semi-definite, $\mathbf{B} \in \mathbb{R}^{|\mathcal{W}|\times|\mathcal{U}|}$, $\mathbf{C} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$. The positive-definite assumption implies that f is a convex quadratic for a fixed $\mathbf{u}$. For the outer objective, we restrict our analysis to the case where F only depends directly on $\mathbf{w}$ (corresponding to a pure response *BLO problem, which encompasses many common applications including hyperparameter optimization):*

$$F(\mathbf{u},\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top\mathbf{P}\mathbf{w} + \mathbf{f}^\top\mathbf{w} + h\,,$$

where $\mathbf{P} \in \mathbb{R}^{|\mathcal{W}|\times|\mathcal{W}|}$ is positive semi-definite.

Note that we assume that the inner objective is a convex (but not strongly-convex) quadratic, which may have *many global minima*, providing a tractable setting to analyze implicit bias. This allows us to model an overparameterized linear regression problem with data matrix $\mathbf{\Phi} \in \mathbb{R}^{n\times p}$, which corresponds to a quadratic with curvature matrix $\mathbf{\Phi}^\top\mathbf{\Phi}$. When the number of parameters is greater than training examples ($p > n$), $\mathbf{\Phi}^\top\mathbf{\Phi}$ is PSD. In this case, f has a manifold of valid solutions (Wu & Xu, 2020).

Statement 3.1 (Cold-start BLO converges to a cold-start equilibrium.)**.** *Suppose f and F satisfy Assumption 3.1. If the inner parameters are initialized to $\mathbf{w}_0$ and learning rates are set appropriately to guarantee convergence, then the cold-start algorithm (Algorithm 1) using exact hypergradients converges to a cold-start equilibrium.*

Proof. The proof is provided in Appendix D.1. □

Implicit Bias of Cold-Start for Inner Solutions. Because cold-start BLO trains the inner parameters *from initialization to convergence* for each outer iteration, the inner solution inherits properties from the single-level optimization literature. For example, in linear regression trained with gradient descent, the inner solution will minimize displacement from the initialization $\|\mathbf{w} - \mathbf{w}_0\|_2^2$. Recent work aims to generalize such statements to other model classes and optimization algorithms – e.g., obtaining min-norm solutions with algorithm-specific norms (Gunasekar et al., 2018). Generalizing the results for various types of neural nets is an active research area (Vardi & Shamir, 2021).

Implicit Bias of Cold-Start for Outer Solutions. In the following theorem, we show that cold-start BLO with exact hypergradients, from outer initialization $\mathbf{u}_0$, converges to the outer solution $\mathbf{u}^\star$ that minimizes displacement from $\mathbf{u}_0$.

Theorem 3.1 (Min-Norm Outer Parameters)**.** *Consider cold-start BLO (Algorithm 1) with exact hypergradients starting from outer initialization $\mathbf{u}_0$. Assume that for each outer iteration, the inner parameters are re-initialized to $\mathbf{w}_0 = \mathbf{0}$ and optimized with an appropriate learning rate that ensures convergence. Then cold-start BLO converges to an outer solution $\mathbf{u}^\star$ with minimum L_2 distance from $\mathbf{u}_0$:*

$$\mathbf{u}^\star = \underset{\mathbf{u}\in \arg\min_{\mathbf{u}} F^\star(\mathbf{u})}{\arg\min} \|\mathbf{u} - \mathbf{u}_0\|^2\,. \tag{6}$$

Proof. The proof is provided in Appendix D.2. □

Next, we observe that the full warm-start and cold-start algorithms are equivalent for strongly-convex inner problems.

Remark 3.2 (Equivalence of Full Warm-Start and Cold-Start in the Strongly Convex Regime)**.** *When the inner problem $f(\mathbf{u},\mathbf{w})$ is strongly convex in $\mathbf{w}$ for each $\mathbf{u}$, then full warm-start (Alg. 2) and cold-start (Alg. 1) are equivalent.*

Proof. The proof is provided in Appendix D.3 □

We relate partial and full warm-start equilibria as follows.

Statement 3.2 (Inclusion of Partial Warm-Start Equilibria.)**.** *Every partial warm-start equilibrium is a full warm-start equilibrium. In addition, if $\Xi(\mathbf{u},\mathbf{w}) = \mathbf{w} - \alpha\nabla_{\mathbf{w}} f(\mathbf{u},\mathbf{w})$ with a fixed (non-decayed) step size α, then full-warm start equilibria are also partial warm-start equilibria.*

Proof. The proof is provided in Appendix D.4. □

3.3.1. Implicit Bias from Hypergrad Approx.

Neumann Series. Next, we investigate the impact of the hypergradient approximation on the converged outer solution. In practice, we use the truncated Neumann series to estimate the inverse Hessian. Note that:

$$\alpha\sum_{j=0}^{K}(\mathbf{I} - \alpha\mathbf{H})^j \approx (\mathbf{H} + \epsilon\mathbf{I})^{-1} \tag{7}$$

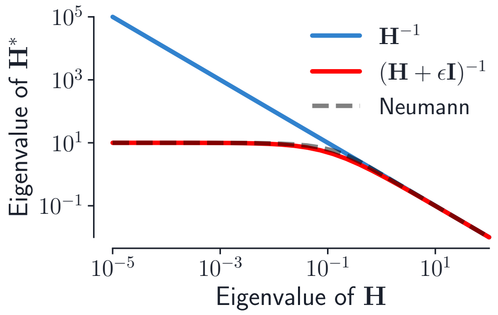


Figure 3: Eigenvalues of various matrix functions of $\mathbf{H}$ (denoted $\mathbf{H}^\star$ in the diagram): $\mathbf{H}^{-1}$, $(\mathbf{H}+\epsilon\mathbf{I})^{-1}$, and the Neumann series approximation $\alpha\sum_{j=0}^{K}(\mathbf{I}-\alpha\mathbf{H})^j$. Here, $\epsilon = \frac{1}{\alpha K}$.

where $\epsilon = \frac{1}{\alpha K}$. This is known from the spectral regularization literature (Gerfo et al., 2008; Rosasco, 2009). We show in Appendix E that the damped Hessian inverse $(\mathbf{H}+\epsilon\mathbf{I})^{-1}$ corresponds to the hypergradient of a proximally-regularized inner objective: $\hat{f}(\mathbf{u},\mathbf{w}) = f(\mathbf{u},\mathbf{w}) + \frac{\epsilon}{2}\|\mathbf{w}-\mathbf{w}'\|_2^2$, where $\mathbf{w}' \in \arg\min_{\mathbf{w}} f(\mathbf{u},\mathbf{w})$. The damping prevents the inner optimization from moving far in low-curvature directions. Consider the spectral decomposition of the real symmetric matrix $\mathbf{H} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$, where $\mathbf{D}$ is a diagonal matrix containing the eigenvalues of $\mathbf{H}$, and $\mathbf{U}$ is an orthogonal matrix. Then, $(\mathbf{H}+\epsilon\mathbf{I})^{-1} = (\mathbf{U}\mathbf{D}\mathbf{U}^\top+\epsilon\mathbf{I})^{-1} = \mathbf{U}^\top(\mathbf{D}+\epsilon\mathbf{I})^{-1}\mathbf{U}$. If λ is an eigenvalue of $\mathbf{H}$, then the corresponding eigenvalue of $\mathbf{H}^{-1}$ is $\frac{1}{\lambda}$. In contrast, the corresponding eigenvalue of $(\mathbf{H}+\epsilon\mathbf{I})^{-1}$ is $\frac{1}{\lambda+\epsilon}$. When $\epsilon \ll \lambda$, $\frac{1}{\lambda+\epsilon} \approx \frac{1}{\lambda}$; when $\lambda \ll \epsilon$, $\frac{1}{\lambda+\epsilon} \approx \frac{1}{\epsilon}$. Thus, the influence of small eigenvalues (associated with low-curvature directions) is diminished, and the truncated Neumann series primarily takes into account high-curvature directions. Figure 3 illustrates the relationship between the eigenvalues of $\mathbf{H}^{-1}$, $(\mathbf{H}+\epsilon\mathbf{I})^{-1}$, and $\alpha\sum_{j=0}^{K}(\mathbf{I}-\alpha\mathbf{H})^j$.

Unrolling. The implicit bias induced by approximating the hypergradient via K-step unrolled differentiation is qualitatively similar to the bias induced by the truncated Neumann series. For quadratic inner objectives f, the truncated Neumann series coincides with the result of differentiating through K steps of unrolled gradient descent, because the Hessian of a quadratic is constant over the inner optimization trajectory. Note that gradient descent on a quadratic converges (for step-size $\alpha \leq 1/\|\mathbf{H}\|_2$) more rapidly in high-curvature directions than in low-curvature directions. Thus, the approximate best response obtained by truncated unrolling only takes into account high-curvature directions of the inner objective, and is less sensitive to low-curvature directions. In turn, the response Jacobian will only capture how the inner parameters depend on the outer parameters in these high-curvature directions. Note that for general objectives f (e.g., when training neural networks), differentiating through unrolling only coincides with the Neumann series when the inner parameters are at a stationary point of f.

4. Empirical Overparameterization Results

Many large-scale empirical studies—in particular in the areas of hyperparameter optimization and data augmentation—use warm-start bilevel optimization (Hataya et al., 2020b; Ho et al., 2019; Mounsaveng et al., 2021; Peng et al., 2018; Tang et al., 2020). In this section, we introduce simple tasks based on dataset distillation, designed to provide insights into the phenomena at play. First, we show that when the inner problem is overparameterized, the inner parameters $\mathbf{w}$ can retain information associated with different settings of the outer parameters over the course of joint optimization. Then, we show that when the outer problem is overparameterized, the choice of hypergradient approximation can affect which outer solutions is found. Experimental details and extended results are provided in Appendix G.

4.1. Inner Overparameterization: Dataset Distillation

In dataset distillation, the original dataset is only accessed in the outer objective; thus, one might expect that the lower-dimensional distilled dataset would act as an information bottleneck between the objectives. Because the outer objective is only used directly to update the outer variables, it would seem intuitive that all of the information about the outer objective is compressed into the outer variables. While this is correct for the cold-start equilibrium, we show that it does not hold for the warm-start equilibrium: *a surprisingly large amount of information can leak from the original dataset to the inner variables (e.g., network weights).*

Consider a 2D binary classification task where the classes form concentric rings (Figure 4). We aim to learn *two distilled datapoints* (one per class) to model the circular decision boundary. One may *a priori* expect that this would not be possible when training a neural network on only the two distilled points; indeed, we observe poor decision boundaries for the original dataset when training a model to convergence on the synthetic datapoints (Figure 4, Top Right). However, when performing warm-start alternating updates on the MLP parameters and the learned datapoints, the datapoints follow a nontrivial *trajectory*, tracing out the decision boundary between classes over time (Fig. 4, middle three plots). The model trained jointly with the two learned datapoints fits to the full trajectory of those datapoints. Thus, warm-started BLO yields a model that achieves nearly the same outer loss as one trained directly on the original data, despite only using a single datapoint per class. See the caption of Figure 4 for details on interpreting this result.

Warm-Start Memory. Here we provide intuition for the warm-start behavior observed in Figure 4. In particular, we discuss how warm-start BLO induces a *memory effect*.

Figure 5 illustrates warm- and cold-start algorithms on a toy linear regression example where we can visualize the steps

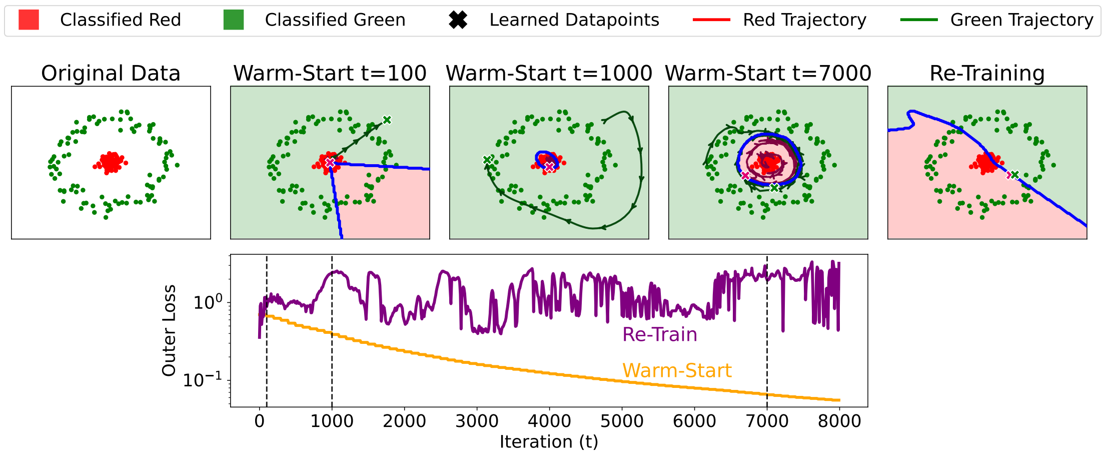


Figure 4: Dataset distillation for binary classification, with two learned datapoints (outer parameters) adapted jointly with the model weights (inner parameters). **Top left:** The original data distribution we wish to distill; **Top middle three plots:** Visualizations of snapshots during training with warm-start BLO, at iterations $t \in \{100, 1000, 7000\}$ (indicated by the dashed vertical lines in the bottom plot). In each middle figure, we show the *new portion of the synthetic datapoint trajectory* since the previous snapshot, shown by a curve with arrows; **Top right:** Decision boundary when re-training to convergence on the final values of the two distilled datapoints yields a poor solution for the original data. **Bottom:** We show the outer loss over the course of a single run of warm-start BLO corresponding to the middle three plots in the top row. We also demonstrate that none of the intermediate synthetic datapoint pairs along the trajectory is sufficient to fit the original data well, by re-training on each intermediate point (shown by the purple curve).

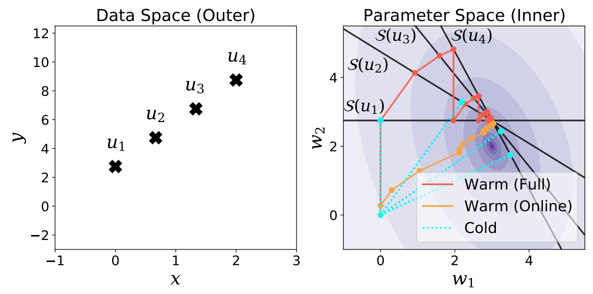
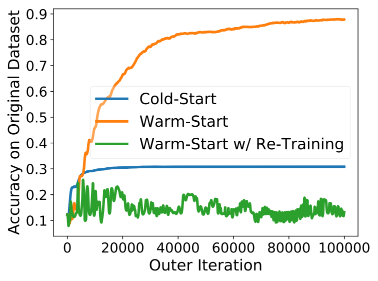


Figure 5: Parameter-space view of warm-start with full inner optimization, warm-start with partial inner optimization (denoted the “online” setting, which most closely resembles what is done in practice), and cold-start optimization.

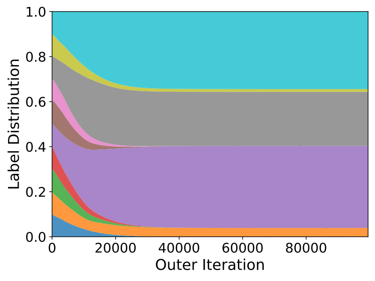
of each algorithm in the inner parameter space. The solution sets for four *fixed* values of a single synthetic datapoint are shown by the solid black lines in Figure 5, and outer loss contours are shown in the background.

We assume the inner parameters are initialized at the origin. Cold-start projects from the initialization onto the solution set corresponding to the current outer parameter: $\mathbf{w}_{k+1} = \Pi(\mathbf{0}, \mathcal{S}(\mathbf{u}_k))$, where $\Pi(\cdot, \mathcal{C})$ denotes projection onto the (convex) set $\mathcal{C}$. In contrast, full warm-start projects the previous inner parameters onto the current solution set: $\mathbf{w}_{k+1} = \Pi(\mathbf{w}_k, \mathcal{S}(\mathbf{u}_k))$. If we cycle through the solution sets repeatedly, full warm-start BLO is equivalent to the Kaczmarz algorithm (Karczmarz, 1937), a classic alternating projection algorithm for finding a point in the intersection of the constraint sets (see App. C.2 for details). In this case, the inner parameters will converge to the intersection of the solution sets $\{\mathcal{S}(\mathbf{u}_i)\}_{i=1}^4$, in effect yielding inner parameters that perform well for several outer parameters simultaneously. In the case of dataset distillation, this corresponds to model weights which fit all of the distilled datapoints over the outer optimization trajectory.

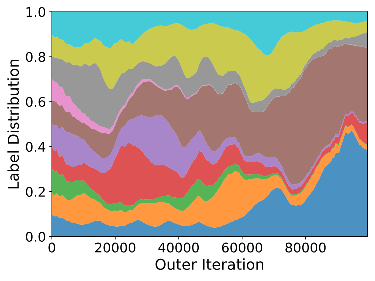
Warm-Start vs Cold-Start in High-Dimensions. To illustrate the warm-start phenomena in high-dimensional problems, we ran a dataset distillation task on MNIST using a linear classifier. Here, the BLO methods are provided with a single 28×28 image “canvas” whose pixels are learned together with a 10-dimensional vector of soft labels. Figure 6a shows accuracies when optimizing a *single synthetic datapoint and its soft label* on 10000 MNIST images. We visualize the soft label evolution in Figures 6b and 6c, showing classes 0-9 as colored regions. While the cold-start soft label quickly converges, warm-start continues to adapt the soft label over the course of joint optimization, effectively training the inner model on all 10 classes (e.g., by placing more weight on different classes at different points in time). We obtained similar results on MNIST, FashionMNIST, and CIFAR-10, with 1 or 10 synthetic datapoints (see Table 3). In addition to dataset distillation, we observe similar phenomena in hyperparameter optimization. We trained a linear data augmentation (DA) network that transforms inputs before feeding them into a classifier; here, the DA net parameters are hyperparameters $\mathbf{u}$, tuned on the validation set. We used subsampled training sets consisting of 50 datapoints—so that data augmentation is beneficial—and evaluated performance on the full validation set. Table 3 compares warm- and cold-start BLO on this task. Note that in order to compare with cold-start solutions (which need 10^3–10^4 inner optimization steps) we *require* tractable inner problems like linear models or MLPs.



(a) Accuracies



(b) Cold-start soft labels



(c) Warm-start soft labels

Figure 6: Dataset distillation using a linear classifier on MNIST.

	Dataset Distillation		Data Aug. Net	
Method	MNIST	CIFAR-10	MNIST	CIFAR-10
Cold-Start	30.7 / 89.1	17.6 / 46.9	84.86	45.38
Warm-Start	90.8 / 97.5	50.3 / 59.8	92.81	59.30
Warm-Start + Re-Train	12.9 / 17.1	10.2 / 8.9	9.15	11.12

Table 3: **Cols 1-3:** Accuracy on original data with 1 or 10 synthetic samples. **Cols 4-6:** Learning a data augmentation network. Extended table in App. G.

Warm-Start Takeaways. Warm-start BLO yields outer parameters that *fail to generalize* under re-initialization of the inner problem. In both toy and high-dimensional problems, re-training with the final outer parameters (e.g., discarding the outer optimization trajectory) yields a model that performs poorly on the outer objective. In addition, warm-start BLO *leaks information* about the outer objective to the inner parameters, which can lead to overestimation of performance. For example, when adapting a small number of hyperparameters online, warm-start BLO may overfit the validation set, yielding a model that fails to generalize to the test set.

4.2. Outer Overparameterization: Anti-Distillation

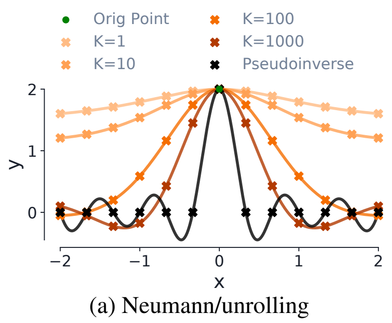
Our previous discussion focused on implicit bias resulting from the choice of cold- or warm-start BLO. Next, we show that when the outer problem is overparameterized, the hypergradient approximation we use can lead to different outer solutions. We propose a task related to dataset distillation, but where we have *more* learned datapoints than original dataset examples, which we term *anti-distillation* (Figure 7). Here, there are many valid ways to set the learned datapoints such that a model trained on those points achieves good performance on the original data.

Anti-Distillation. Consider the linear regression problem $f(\mathbf{u}, \mathbf{w}) = \frac{1}{2}\|\mathbf{\Phi}\mathbf{w} - \mathbf{u}\|_2^2$ where, for simplicity, $\mathbf{u}$ parameterizes only the targets, not the inputs. The minimum Frobenius norm best-response Jacobian is the Moore-Penrose pseudoinverse of the feature matrix, $\mathbf{\Phi}^+$ (see Appendix C.1), which we will approximate in different ways. For this problem, we have one original datapoint, and we learn 13 synthetic datapoints (Figure 7). Any solution that places one learned datapoint on top of the original point perfectly fits the outer objective. We use Fourier-basis regression with the feature function $\phi(x) = a_0 + \sum_{n=1}^{N} \left(a_n 2^{N-n} \cos(nx) + b_n 2^{N-n} \sin(nx)\right)$, where low frequency terms have larger magnitude than high frequency terms—this yields an inductive bias for smooth functions. Because of this difference in magnitude, the inner optimizer can more easily (i.e., with a smaller-magnitude weight update) fit the data by adjusting the regression coefficients on the low-frequency terms. Thus, the minimum-norm solution found by gradient descent will explain as much as possible using the low frequency terms.

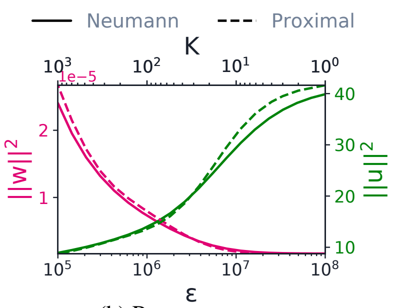
When we estimate the best-response Jacobian with a few of steps of unrolling, the inner optimizer will do its best to fit the data using the low-frequency basis terms, which correspond to high-curvature directions of the inner loss.

In Figure 7a, we show the solutions obtained for different truncations of the Neumann series; because this problem is quadratic, the K-step unrolled hypergradient coincides with the K-step Neumann series hypergradient. To demonstrate the relationship $\alpha \sum_{j=0}^{K} (\mathbf{I} - \alpha\mathbf{H})^j \approx (\mathbf{H} + \frac{1}{\alpha K}\mathbf{I})^{-1}$ empirically, we also show the distilled datasets obtained with the proximal best-response Jacobian, for various damping factors $\epsilon \in \{$1e-5, 1e-6, 1e-7, 1e-8$\}$, in Figure 10. We observe similar behavior to the Neumann series. In Figure 7b we plot the norms of the converged inner and outer parameters for a range of Neumann truncation lengths K and damping factors ϵ. We see that while the Neumann and proximal approximate hypergradients behave similarly, they are not equivalent. In Appendix G, Figure 13, we show similar behavior when using an MLP rather than a linear model.

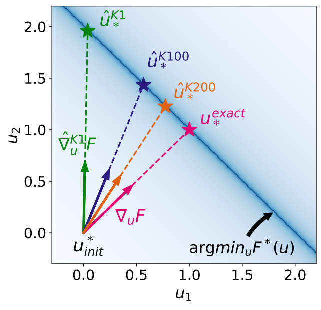
Figure 7c visualizes optimization trajectories in the outer parameter space to provide additional intuition for the behavior of the outer parameter norms in Figure 7b. We consider antidistillation with 1 original and 2 learned datapoints. We use $\hat{\nabla}_{\mathbf{u}}^{K} F(\mathbf{u}, \mathbf{w})$ to denote the approximate hypergradient obtained via implicit differentiation with the K-term truncated Neumann series. We show the true hypergradi-



(a) Neumann/unrolling



(b) Parameter norms



(c) Approximate hypergradients

Figure 7: **Antidistillation task for linear regression with an overparametrized outer objective.** We learn the y-component of 13 synthetic datapoints such that a regressor trained on those points will fit a single original dataset point, shown by the green dot at $(0, 2)$. Fig. **(a)** shows the learned datapoints (outer parameters) obtained via different truncated Neumann approximations to the hypergradient. Fig. **(b)** shows the norms of the outer parameters $\|\mathbf{u}\|^2$ as a function of K (for Neumann/unrolling) or ϵ (for proximal). We observe that better hypergradient approximations (e.g., larger K or smaller ϵ) lead to smaller norm outer parameters, because they account for both high- and low-curvature directions of the inner objective. Fig. **(c)** visualizes outer optimization trajectories in the outer parameter space, to provide intuition for the behavior in **(a)** and **(b)**. We consider antidistillation with 1 original datapoint and 2 learned datapoints. We show the true hypergradient $\nabla_{\mathbf{u}} F$, approximations using truncated Neumann series $\hat{\nabla}_{\mathbf{u}} F$, and the converged outer parameters for each setting, e.g., $\hat{\mathbf{u}}_\star^{K1}$. We observe that: 1) when the outer problem is overparameterized, approximate hypergradients converge to valid solutions to the outer objective; and 2) the exact hypergradient converges to the min-norm solution to the outer problem.

ent $\nabla_{\mathbf{u}} F$, approximations using truncated Neumann series $\hat{\nabla}_{\mathbf{u}} F$, and the converged outer parameters for each setting, e.g., $\hat{\mathbf{u}}_\star^{K1}$. We observe that: 1) when the outer problem is overparameterized, approximate hypergradients converge to valid solutions in $\arg\min_{\mathbf{u}} F^\star(\mathbf{u})$; and 2) the exact hypergradient converges to the min-norm outer solution.

5. Related Work

Extended related work is provided in Appendix B.

Overparameterization. Overparameterization has long been studied in single-level optimization, generating key insights such as neural network behavior in the infinite-width limit (Jacot et al., 2018; Sohl-Dickstein et al., 2020; Lee et al., 2019), double descent phenomena (Nakkiran et al., 2020; Belkin et al., 2019), the ability to fit random labels (Zhang et al., 2016), and the inductive biases of optimizers. However, prior work has not investigated implicit bias in bilevel optimization. Implicit bias has a long history in machine learning: many works have observed and studied the connection between early stopping and L_2 bias (Strand, 1974; Morgan & Bourlard, 1989; Friedman & Popescu, 2003; Yao et al., 2007). Interest in implicit bias has increased over the past few years (Nacson et al., 2019; Soudry et al., 2018; Suggala et al., 2018; Poggio et al., 2019; Ji & Telgarsky, 2019; Ali et al., 2019; 2020).

Prior Work using Warm Start. Many papers perform joint optimization of the inner and outer parameters (e.g., data augmentations together with a base model), such as Hataya et al. (2020a;b); Ho et al. (2019); Mounsaveng et al. (2021); Peng et al. (2018); Tang et al. (2020). Ghadimi & Wang (2018), Hong et al. (2020), and Ji et al. (2020) propose bilevel optimization algorithms that use warm-starting; however, they focus on analyzing the convergence rates of their algorithms and do not consider inner underspecification.

Gap Between Theory & Practice. Existing BLO theory typically assumes unique solutions to the inner (and sometimes outer) problem, and focuses on showing that approximation methods get (provably) close to the solution. Shaban et al. (2019) provide conditions where optimization with an approximate hypergradient $\hat{\mathbf{h}}$ from truncated unrolling converges to a BLO solution, but they assume uniqueness of the inner solution. Grazzi et al. (2020a;b) study iteration complexity and convergence of hypergradient approximations in the strongly-convex inner problem setting. Several other works focus on convergence rate analyses (Ji et al., 2021; Yang et al., 2021; Ji & Liang, 2021).

6. Conclusion

Most work on bilevel optimization has made the simplifying assumption that the solutions to the inner and outer problems are unique. However, this does not hold in many practical applications, such as hyperparameter optimization for overparameterized neural networks. We investigated overparameterized bilevel optimization, where either the inner or outer problems may admit non-unique solutions. We formalized warm- and cold-start equilibria, which correspond to common BLO algorithms. We analyzed the properties of these equilibria, and algorithmic choices such as the number of Neumann series terms used to approximate the hypergradient. We presented several tasks illustrating that these choices can significantly affect the solutions obtained in practice. More generally, we highlighted the importance of, and laid groundwork for, analyzing the effects of overparameterization in nested optimization problems.

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.

Ali, A., Kolter, J. Z., and Tibshirani, R. J. A continuous-time view of early stopping for least squares regression. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

Ali, A., Dobriban, E., and Tibshirani, R. The implicit regularization of stochastic gradient flow for least squares. In *International Conference on Machine Learning (ICML)*, pp. 233–244, 2020.

Amari, S.-i., Ba, J., Grosse, R., Li, X., Nitanda, A., Suzuki, T., Wu, D., and Xu, J. When does preconditioning help or hurt generalization? *arXiv preprint arXiv:2006.10732*, 2020.

Angelos, J., Grossman, G., Kaufman, E., Lenker, T., and Rakesh, L. Limit cycles for successive projections onto hyperplanes in RN. *Linear Algebra and its Applications*, 285, 1998.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

Bae, J. and Grosse, R. Delta-STN: Efficient bilevel optimization for neural networks using structured response Jacobians. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Bartlett, P. L., Long, P. M., Lugosi, G., and Tsigler, A. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 2020.

Belkin, M. Fit without fear: remarkable mathematical phenomena of deep learning through the prism of interpolation. *Acta Numerica*, 30:203–248, 2021.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, pp. 15849–15854, 2019.

Bengio, Y. Gradient-based optimization of hyperparameters. *Neural Computation*, 12(8):1889–1900, 2000.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *International Conference on Machine Learning (ICML)*, pp. 41–48, 2009.

Blondel, M., Berthet, Q., Cuturi, M., Frostig, R., Hoyer, S., Llinares-López, F., Pedregosa, F., and Vert, J.-P. Efficient and modular implicit differentiation. *arXiv preprint arXiv:2105.15183*, 2021.

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.

Chen, D. and Hagan, M. T. Optimal use of regularization and cross-validation in neural network modeling. In *International Joint Conference on Neural Networks (IJCNN)*, volume 2, pp. 1275–1280, 1999.

Cheung, T.-H. and Yeung, D.-Y. MODALS: Modality-agnostic automated data augmentation in the latent space. In *International Conference on Learning Representations (ICLR)*, 2021.

Dempe, S. *Foundations of bilevel programming*. Springer Science & Business Media, 2002.

Dempe, S., Dutta, J., and Mordukhovich, B. New necessary optimality conditions in optimistic bilevel programming. *Optimization*, 56, 2007.

Dempe, S., Mordukhovich, B. S., and Zemkoho, A. B. Necessary optimality conditions in pessimistic bilevel programming. *Optimization*, 2014.

Domke, J. Generic methods for optimization-based modeling. In *Artificial Intelligence and Statistics*, pp. 318–326, 2012.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, pp. 1126–1135, 2017.

Foo, C.-s., Do, C. B., and Ng, A. Y. Efficient multiple hyperparameter learning for log-linear models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 377–384, 2008.

Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. Bilevel programming for hyperparameter optimization and meta-learning. *arXiv preprint arXiv:1806.04910*, 2018.

Friedman, J. and Popescu, B. E. Gradient directed regularization for linear regression and classification. Technical report, Statistics Department, Stanford University, 2003.

Gerfo, L. L., Rosasco, L., Odone, F., Vito, E. D., and Verri, A. Spectral algorithms for supervised learning. *Neural Computation*, 20(7):1873–1897, 2008.

Ghadimi, S. and Wang, M. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

Grazzi, R., Franceschi, L., Pontil, M., and Salzo, S. On the iteration complexity of hypergradient computation. In *International Conference on Machine Learning (ICML)*, pp. 3748–3758, 2020a.

Grazzi, R., Pontil, M., and Salzo, S. Convergence properties of stochastic hypergradients. *arXiv preprint arXiv:2011.07122*, 2020b.

Gunasekar, S., Lee, J., Soudry, D., and Srebro, N. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning (ICML)*, pp. 1832–1841, 2018.

Ha, D., Dai, A., and Le, Q. V. Hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2017.

Harker, P. T. and Pang, J.-S. Existence of optimal solutions to mathematical programs with equilibrium constraints. *Operations Research Letters*, 7(2):61–64, 1988.

Hataya, R., Zdenek, J., Yoshizoe, K., and Nakayama, H. Faster Autoaugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision (ECCV)*, pp. 1–16, 2020a.

Hataya, R., Zdenek, J., Yoshizoe, K., and Nakayama, H. Meta approach to data augmentation optimization. *arXiv preprint arXiv:2006.07965*, 2020b.

Ho, D., Liang, E., Chen, X., Stoica, I., and Abbeel, P. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning (ICML)*, pp. 2731–2741, 2019.

Hong, M., Wai, H.-T., Wang, Z., and Yang, Z. A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic. *arXiv preprint arXiv:2007.05170*, 2020.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

Ji, K. and Liang, Y. Lower bounds and accelerated algorithms for bilevel optimization. *arXiv preprint arXiv:2102.03926*, 2021.

Ji, K., Yang, J., and Liang, Y. Bilevel optimization: Nonasymptotic analysis and faster algorithms. *arXiv preprint arXiv:2010.07962*, 2020.

Ji, K., Yang, J., and Liang, Y. Bilevel optimization: Convergence analysis and enhanced design. In *International Conference on Machine Learning (ICML)*, pp. 4882–4892, 2021.

Ji, Z. and Telgarsky, M. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pp. 1772–1798, 2019.

Karczmarz, S. Angenaherte Auflosung von Systemen linearer Gleichungen. *Bull. Int. Acad. Pol. Sic. Let., Cl. Sci. Math. Nat.*, pp. 355–357, 1937.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, pp. 1885–1894, 2017.

Larsen, J., Hansen, L. K., Svarer, C., and Ohlsson, M. Design and regularization of neural networks: The optimal use of a validation set. In *IEEE Signal Processing Society Workshop*, pp. 62–71, 1996.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Liao, R., Xiong, Y., Fetaya, E., Zhang, L., Yoon, K., Pitkow, X., Urtasun, R., and Zemel, R. Reviving and improving recurrent back-propagation. In *International Conference on Machine Learning (ICML)*, 2018.

Lignola, M. B. and Morgan, J. Topological existence and stability for Stackelberg problems. *Journal of Optimization Theory and Applications*, 84(1):145–169, 1995.

Lignola, M. B. and Morgan, J. Existence of solutions to bilevel variational problems in Banach spaces. In *Equilibrium Problems: Nonsmooth Optimization and Variational Inequality Models*, pp. 161–174. Springer, 2001.

Liu, H., Simonyan, K., and Yang, Y. DARTS: Differentiable architecture search. In *International Conference on Learning Representations (ICLR)*, 2019.

Liu, J., Fan, Y., Chen, Z., and Zheng, Y. Pessimistic bilevel optimization: A survey. *International Journal of Computational Intelligence Systems*, 11(1):725–736, 2018.

Liu, J., Fan, Y., Chen, Z., and Zheng, Y. Methods for pessimistic bilevel optimization. In *Bilevel Optimization*, pp. 403–420. Springer, 2020.

Loridan, P. and Morgan, J. Weak via strong Stackelberg problem: New results. *Journal of Global Optimization*, 1996.

Lorraine, J. and Duvenaud, D. Stochastic hyperparameter optimization through hypernetworks. In *NIPS Meta-Learning Workshop*, 2017.

Lorraine, J., Vicol, P., and Duvenaud, D. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1540–1552, 2020.

Lucchetti, R., Mignanego, F., and Pieri, G. Existence theorems of equilibrium points in Stackelberg games with constraints. *Optimization*, 1987.

Luketina, J., Berglund, M., Greff, K., and Raiko, T. Scalable gradient-based tuning of continuous regularization hyperparameters. In *International Conference on Machine Learning (ICML)*, pp. 2952–2960, 2016.

MacKay, M., Vicol, P., Lorraine, J., Duvenaud, D., and Grosse, R. Self-Tuning Networks: Bilevel optimization of hyperparameters using structured best-response functions. In *International Conference on Learning Representations (ICLR)*, 2019.

Maclaurin, D., Duvenaud, D., and Adams, R. Gradient-based hyperparameter optimization through reversible learning. In *International Conference on Machine Learning (ICML)*, pp. 2113–2122, 2015.

Mehra, A. and Hamm, J. Penalty method for inversion-free deep bilevel optimization. *arXiv preprint arXiv:1911.03432*, 2019.

Metz, L., Maheswaranathan, N., Nixon, J., Freeman, D., and Sohl-Dickstein, J. Understanding and correcting pathologies in the training of learned optimizers. In *International Conference on Machine Learning (ICML)*, pp. 4556–4565, 2019.

Micaelli, P. and Storkey, A. Non-greedy gradient-based hyperparameter optimization over long horizons. *arXiv preprint arXiv:2007.07869*, 2020.

Mishachev, N. and Shmyrin, A. M. On realization of limit polygons in sequential projection method. *International Transaction Journal of Engineering, Management and Applied Sciences and Technologies*, 10(15):1015–1015, 2019.

Morgan, N. and Bourlard, H. Generalization and parameter estimation in feedforward nets: Some experiments. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 2, pp. 630–637, 1989.

Mounsaveng, S., Laradji, I., Ben Ayed, I., Vazquez, D., and Pedersoli, M. Learning data augmentation with online bilevel optimization for image classification. In *Winter Conference on Applications of Computer Vision*, pp. 1691–1700, 2021.

Nacson, M. S., Srebro, N., and Soudry, D. Stochastic gradient descent on separable data: Exact convergence with a fixed learning rate. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 3051–3059, 2019.

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. Deep double descent: Where bigger models and more data hurt. In *International Conference on Learning Representations (ICLR)*, 2020.

Outrata, J. V. Necessary optimality conditions for Stackelberg problems. *Journal of Optimization Theory and Applications*, 76 (2):305–320, 1993.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS Workshop Autodifferentiation*, 2017.

Pearlmutter, B. A. Fast exact multiplication by the Hessian. *Neural Computation*, 6(1):147–160, 1994.

Pedregosa, F. Hyperparameter optimization with approximate gradient. In *International Conference on Machine Learning (ICML)*, 2016.

Peng, X., Tang, Z., Yang, F., Feris, R. S., and Metaxas, D. Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2226–2234, 2018.

Pfau, D. and Vinyals, O. Connecting generative adversarial networks and actor-critic methods. In *NeurIPS Workshop on Adversarial Training*, 2016.

Poggio, T., Banburski, A., and Liao, Q. Theoretical issues in deep networks: Approximation, optimization and generalization. In *Proceedings of the National Academy of Sciences*, 2019.

Raghu, A., Raghu, M., Kornblith, S., Duvenaud, D., and Hinton, G. Teaching with commentaries. In *International Conference on Learning Representations (ICLR)*, 2020.

Rajeswaran, A., Finn, C., Kakade, S., and Levine, S. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning (ICML)*, pp. 4334–4343, 2018.

Riba, E., Mishkin, D., Ponsa, D., Rublee, E., and Bradski, G. Kornia: An open source differentiable computer vision library for PyTorch. In *Winter Conference on Applications of Computer Vision*, 2020. URL `https://arxiv.org/pdf/1910.02190.pdf`.

Rosasco, L. 9.520: Statistical Learning Theory and Applications, Lecture 7. Lecture Notes, 2009. URL `https://www.mit.edu/~9.520/spring09/Classes/class07_spectral.pdf`.

Shaban, A., Cheng, C.-A., Hatch, N., and Boots, B. Truncated back-propagation for bilevel optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1723–1732, 2019.

Sinha, A., Malo, P., and Deb, K. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.

Sohl-Dickstein, J., Novak, R., Schoenholz, S. S., and Lee, J. On the infinite width limit of neural networks with a standard parameterization. *arXiv preprint arXiv:2001.07301*, 2020.

Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

Stošić, M., Xavier, J., and Dodig, M. Projection on the intersection of convex sets. *Linear Algebra and its Applications*, 509:191–205, 2016.

Strand, O. N. Theory and methods related to the singular-function expansion and Landweber's iteration for integral equations of the first kind. *SIAM Journal on Numerical Analysis*, 11(4): 798–825, 1974.

Sucholutsky, I. and Schonlau, M. 'Less than one'-shot learning: Learning N classes from M< N samples. *arXiv preprint arXiv:2009.08449*, 2020.

Suggala, A., Prasad, A., and Ravikumar, P. K. Connecting optimization and regularization paths. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pp. 10608–10619, 2018.

Tang, Z., Gao, Y., Karlinsky, L., Sattigeri, P., Feris, R., and Metaxas, D. OnlineAugment: Online data augmentation with less domain knowledge. In *European Conference on Computer Vision (ECCV)*, 2020.

Vardi, G. and Shamir, O. Implicit regularization in ReLU networks with the square loss. In *Conference on Learning Theory*, pp. 4224–4258, 2021.

Wang, T., Zhu, J.-Y., Torralba, A., and Efros, A. A. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Wiesemann, W., Tsoukalas, A., Kleniati, P.-M., and Rustem, B. Pessimistic bilevel optimization. *SIAM Journal on Optimization*, 23(1):353–380, 2013.

Wu, D. and Xu, J. On the optimal weighted ℓ_2 regularization in overparameterized linear regression. *arXiv preprint arXiv:2006.05800*, 2020.

Wu, Y., Ren, M., Liao, R., and Grosse, R. Understanding short-horizon bias in stochastic meta-optimization. In *International Conference on Learning Representations (ICLR)*, 2018.

Yang, J., Ji, K., and Liang, Y. Provably faster algorithms for bilevel optimization. *arXiv preprint arXiv:2106.04692*, 2021.

Yao, Y., Rosasco, L., and Caponnetto, A. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2): 289–315, 2007.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*, 2016.

Zhao, B., Mopuri, K. R., and Bilen, H. Dataset condensation with gradient matching. In *International Conference on Learning Representations (ICLR)*, 2021.

Zoph, B. and Le, Q. V. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2017.

Acknowledgements

We thank James Lucas, Guodong Zhang, and David Acuna for helpful feedback on the manuscript. Resources used in this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute (www.vectorinstitute.ai/partners).

Appendix

This appendix is structured as follows:

- In Section A, we provide an overview of the notation we use throughout the paper.
- In Section B, we provide an extended discussion of background and related work.
- In Section C, we provide derivations of formulas used in the main body.
- In Section D, we provide proofs of the theorems in the main paper.
- In Section E, we derive the response Jacobian for a proximal inner objective, recovering a form of the IFT using the damped Hessian inverse.
- In Section F, we provide an overview of the result from Lorraine et al. (2020) that shows that differentiating through i steps of unrolling starting from optimal inner parameters $\mathbf{w}^\star$ is equivalent to approximating the inverse Hessian with the first i terms of the Neumann series.
- In Section G, we provide experimental details and extended results, as well as a link to an animation of the bilevel training dynamics for the dataset distillation task from Section 4.1.
- In Section H, we describe algorithms for cold-start and warm-start bilevel optimization.

A. Notation

BLO	Bilevel optimization
F	Outer objective
f	Inner objective
$\mathbf{u}$	Outer variables
$\mathbf{w}$	Inner variables
$\mathcal{U}$	Outer parameter space
$\mathcal{W}$	Inner parameter space
$\rightrightarrows$	Multi-valued mapping between two sets
$\mathcal{S}(\mathbf{u})$	Set-valued response mapping for $\mathbf{u}$: $\mathcal{S}(\mathbf{u}) = \arg\min_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$
$\mathbf{\Phi}$	Design matrix, where each row corresponds to an example
$\|\| \cdot \|\|_2^2$	(Squared) Euclidean norm
$\|\| \cdot \|\|_F^2$	(Squared) Frobenius norm
$\mathbf{A}^{+}$	Moore-Penrose Pseudoinverse of $\mathbf{A}$
$\mathbf{A}^{+_k}$	Shorthand for k-term Neumann series approximate inverse, $\sum_{j=0}^{k}(\mathbf{I} - \alpha\mathbf{A})^j$
$\mathbf{A}^{\dagger}$	Any matrix such that $\mathbf{x} = \mathbf{A}^{\dagger}\mathbf{b}$ is a solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$
$\mathbf{u}_k^{\star}$	A fixpoint of the outer problem obtained via the k-step unrolled hypergradient
$\hat{\nabla}_{\mathbf{u}}^{K10} F$	Hypergradient approximation using $K = 10$ terms of the Neumann series
α	Learning rate for inner optimization or step size for the Neumann series
β	Learning rate for outer optimization
k	Number of unrolling iterations / Neumann steps
$F^{\star}(\mathbf{u})$	$F(\mathbf{u}, \mathbf{w}^{\star})$ where $\mathbf{w}^{\star} \in \mathcal{S}(\mathbf{u})$
$\mathbf{w}^{\star}$	An inner solution, $\mathbf{w}^{\star} \in \mathcal{S}(\mathbf{u}) = \arg\min_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$
$\mathbf{w}_0$	An inner parameter initialization
$\mathcal{N}(\mathbf{A})$	Nullspace of $\mathbf{A}$
PSD	Positive semi-definite
Neumann series	If $(\mathbf{I} - \mathbf{A})$ is contractive, then $\mathbf{A}^{-1} = \sum_{j=0}^{\infty}(\mathbf{I} - \mathbf{A})^j$
Implicit Differentiation Hypergradient	$\frac{dF}{d\mathbf{u}} = \left(\frac{\partial \mathbf{w}^{\star}}{\partial \mathbf{u}}\right)^{\top} \left(\frac{\partial F(\mathbf{u}, \mathbf{w}^{\star})}{\partial \mathbf{w}}\right) = -\left(\frac{\partial^2 f(\mathbf{w}^{\star}, \mathbf{u})}{\partial \mathbf{w} \partial \mathbf{w}^{\top}}\right)^{-1} \left(\frac{\partial^2 f(\mathbf{w}^{\star}, \mathbf{u})}{\partial \mathbf{u} \partial \mathbf{w}}\right) \left(\frac{\partial F(\mathbf{u}, \mathbf{w}^{\star})}{\partial \mathbf{w}}\right)$
Optimize$(f(\cdot), \mathbf{w}_k)$	Perform an optimization process on f with initialization $\mathbf{w}_k$
HypergradApprox$(\mathbf{u}, \mathbf{w})$	Compute a hypergradient approximation at given inner and outer parameters
Optimistic BLO	$\mathbf{u}^{\star} \in \arg\min_{\mathbf{u}} F(\mathbf{u}, \mathbf{w}^{\star}) \quad \text{s.t.} \quad \mathbf{w}^{\star} \in \arg\min_{\mathbf{w} \in \mathcal{S}(\mathbf{u}^{\star})} F(\mathbf{u}^{\star}, \mathbf{w})$
Pessimistic BLO	$\mathbf{u}^{\star} \in \arg\min_{\mathbf{u}} F(\mathbf{u}, \mathbf{w}^{\star}) \quad \text{s.t.} \quad \mathbf{w}^{\star} \in \arg\max_{\mathbf{w} \in \mathcal{S}(\mathbf{u}^{\star})} F(\mathbf{u}^{\star}, \mathbf{w})$
HO	Hyperparameter optimization
NAS	Neural architecture search
DD	Dataset distillation
BR	Best-response
IFT	Implicit Function Theorem

Table 4: Summary of the notation and abbreviations used in this paper.

B. Extended Related Work

Hyperparameter Optimization (HO). There are three main approaches for gradient-based HO: 1) differentiating through unrolls of the inner problem, sometimes called *iterative differentiation* (Domke, 2012; Maclaurin et al., 2015; Shaban et al., 2019); 2) using *implicit differentiation* to compute the response Jacobian assuming that the inner optimization has converged (Larsen et al., 1996; Bengio, 2000; Foo et al., 2008; Pedregosa, 2016); and 3) using a *hypernetwork* (Ha et al., 2017) to approximate the best-response function locally, $\hat{\mathbf{w}}_\phi(\boldsymbol{\lambda}) \approx \mathbf{w}^\star(\boldsymbol{\lambda})$, such that the outer gradient can be computed using the chain rule through the hypernetwork, $\frac{\partial \mathcal{L}_V}{\partial \boldsymbol{\lambda}} = \frac{\partial \mathcal{L}_V}{\partial \hat{\mathbf{w}}\phi(\boldsymbol{\lambda})} \frac{\partial \hat{\mathbf{w}}\phi(\boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}}$. Hypernetworks have been applied to HO in (Lorraine & Duvenaud, 2017; MacKay et al., 2019; Bae & Grosse, 2020). MacKay et al. (2019) and Bae & Grosse (2020) have observed that STNs learn online hyperparameter schedules (e.g., for dropout rates and augmentations) that can outperform any fixed hyperparameter value. We believe that warm-start effects at least partially explain the observed improvements from hyperparameter schedules.

Truncation Bias. We restrict our focus to bilevel problems in which the outer parameters *affect the fixed points* of the inner problem—this includes dataset distillation and HO for most regularization hyperparameters, but not the *optimization hyperparameters* such as the learning rate and momentum. Truncation bias has been shown to lead to critical failures when used for tuning such hyperparameters (Wu et al., 2018; Metz et al., 2019). In contrast, greedy adaptation of regularization hyperparameters has been successful empirically, via population-based training (Jaderberg et al., 2017), hypernetwork-based HO (Lorraine & Duvenaud, 2017; MacKay et al., 2019; Bae & Grosse, 2020), and online implicit differentiation (Lorraine et al., 2020; Hataya et al., 2020b).

Data Augmentation. A special case of hyperparameter optimization that has received widespread attention is *automatic data augmentation* (Hataya et al., 2020b;a; Riba et al., 2020; Peng et al., 2018; Mounsaveng et al., 2021; Cheung & Yeung, 2021), where the aim is either to learn the strengths with which to apply different augmentations, or an *augmentation network* that takes an input image and potentially a source of random noise, and outputs an augmented example (Lorraine et al., 2020; Tang et al., 2020). The former approach typically involves tens to hundreds of outer parameters (the coefficients of pre-specified augmentations), while the latter approach involves millions of outer parameters (the weights of the augmentation network). The Population-Based Augmentation (PBA) algorithm (Ho et al., 2019) searches for augmentation schedules using Population-Based Training (Jaderberg et al., 2017); the authors found that training with the PBA schedule outperformed using the fixed final hyperparameters or training with a shuffled schedule (e.g., using the magnitudes of augmentations from the schedule, but in a random order). Dataset distillation (Wang et al., 2018) can be thought of as a special case of data augmentation, where the "augmentation" operation replaces an original dataset example with a learned, synthetic example. Maclaurin et al. (2015) were among the first to consider learning a training dataset in a bilevel formulation. Recent work has also looked at learning synthetic examples with soft labels, aiming to compress a dataset with N classes into $M < N$ synthetic examples (Sucholutsky & Schonlau, 2020).

Hysteresis. Luketina et al. (2016) are among the first we are aware of, who explicitly considered the effect of hysteresis (e.g., path-dependence) on the model obtained via alternating optimization of the model parameters and hyperparameters. The HO algorithm they introduce, dubbed T1-T2, uses the IFT with the identity matrix as an approximation to the inverse Hessian (e.g., equivalent to using $k = 0$ terms of the Neumann series). Interestingly, in contrast to more recent HO papers (particularly ones that focus on data augmentation), Luketina et al. (2016) found that re-training the model from scratch performed better than the online model. One potential explanation for this is the choice of hyperparameters: Luketina et al. (2016) adapted L2 coefficients and Gaussian noise added to inputs and activations, rather than augmentations.

C. Derivations

C.1. Minimum-Norm Response Jacobian

Suppose we have the following inner problem:

$$f(\mathbf{w}, \mathbf{u}) = \frac{1}{2} \|\boldsymbol{\Phi}\mathbf{w} - \mathbf{u}\|_2^2$$

The gradient, Hessian, and second-order mixed partial derivatives of f are:

$$\frac{\partial f(\mathbf{w}, \mathbf{u})}{\partial \mathbf{w}} = \boldsymbol{\Phi}^\top \boldsymbol{\Phi}\mathbf{w} - \boldsymbol{\Phi}^\top \mathbf{u} \qquad \frac{\partial^2 f(\mathbf{w}, \mathbf{u})}{\partial \mathbf{w} \partial \mathbf{w}^\top} = \mathbf{H} = \boldsymbol{\Phi}^\top \boldsymbol{\Phi} \qquad \frac{\partial^2 f(\mathbf{w}, \mathbf{u})}{\partial \mathbf{u} \partial \mathbf{w}} = \boldsymbol{\Phi}^\top$$

The minimum-norm response Jacobian is:

$$\frac{\partial \mathbf{w}^\star(\mathbf{u})}{\partial \mathbf{u}} = \underset{\mathbf{HM}=\mathbf{\Phi}}{\arg\min} ||\mathbf{M}||_F^2 \tag{8}$$

We need a solution to the linear system $\mathbf{HM} = \mathbf{\Phi}$. Assuming this system is satisfiable, the matrix $\mathbf{Q} = \mathbf{H}^+\mathbf{\Phi}$ is a solution that satisfies $||\mathbf{Q}||_F^2 \leq ||\mathbf{M}||_F^2$ for any matrix $\mathbf{M}$:

$$\mathbf{Q} = \mathbf{H}^+\mathbf{\Phi} = (\mathbf{\Phi}^\top\mathbf{\Phi})^+\mathbf{\Phi} = (\mathbf{\Phi}^\top\mathbf{\Phi})^{-1}\mathbf{\Phi} = \mathbf{\Phi}^+ \tag{9}$$

Thus, the minimum-norm response Jacobian is the Moore-Penrose pseudoinverse of the feature matrix, $\mathbf{\Phi}^+$.

C.2. Iterated Projection.

Alternating projections is a well-known algorithm for computing a point in the intersection of convex sets. Dijkstra's projection algorithm is a modified version which finds a specific point in the intersection of the convex sets, namely the point obtained by projecting the initial iterate onto the intersection. Iterated algorithms for projection onto the intersection of a set of convex sets are studied in (Stošić et al., 2016). Angelos et al. (1998) show that successive projections onto hyperplanes can converge to limit cycles. Mishachev & Shmyrin (2019) study algorithms based on sequential projection onto hyperplanes defined by equations of a linear system (like Kaczmarz), and show that with specifically-chosen systems of equations, the limit polygon of such an algorithm can be any predefined polygon.

Closed-Form Projection Onto the Solution Set Here, we provide a derivation of the formula we use to compute the analytic projections onto the solution sets given by different hyperparameters in Figure 5. This derivation is known in the literature on the Kaczmarz algorithm (Karczmarz, 1937); we provide it here for clarity.

Consider homogeneous coordinates for the data $\mathbf{x} = [x, 1]$ such that we can write the weights as $\mathbf{w} = [w_1, w_2]$ and the model as $\mathbf{w}^\top\mathbf{x} = y$. Given an initialization $\mathbf{w}_0$, we would like to find the point $\mathbf{w}^*$ such that:

$$\mathbf{w}^* = \arg \min_{\{\mathbf{w}|\mathbf{w}^\top\mathbf{x}=y\}} \frac{1}{2}||\mathbf{w} - \mathbf{w}_0||^2 \tag{10}$$

To find the solution to this problem, we write the Lagrangian:

$$L(\mathbf{w}, \lambda) = \frac{1}{2}||\mathbf{w} - \mathbf{w}_0||^2 + \lambda(\mathbf{w}^\top\mathbf{x} - y) \tag{11}$$

The solution to this problem is the stationary point of the Lagrangian. Taking the gradient and equating its components to 0 gives:

$$\nabla_{\mathbf{w}} L(\mathbf{w}, \lambda) = \mathbf{w} - \mathbf{w}_0 + \lambda\mathbf{x} = 0 \tag{12}$$

$$\nabla_\lambda L(\mathbf{w}, \lambda) = \mathbf{w}^\top\mathbf{x} - y = 0 \tag{13}$$

From the first equation, we have:

$$\mathbf{w} = \mathbf{w}_0 - \lambda\mathbf{x} \tag{14}$$

Plugging this into the second equation gives:

$$\mathbf{x}^\top\mathbf{w} - y = 0 \tag{15}$$

$$\mathbf{x}^\top(\mathbf{w}_0 - \lambda\mathbf{x}) - y = 0 \tag{16}$$

$$\mathbf{x}^\top\mathbf{w}_0 - \lambda\mathbf{x}^\top\mathbf{x} - y = 0 \tag{17}$$

$$\lambda\mathbf{x}^\top\mathbf{x} = \mathbf{x}^\top\mathbf{w}_0 - y \tag{18}$$

$$\lambda = \frac{\mathbf{x}^\top\mathbf{w}_0 - y}{\mathbf{x}^\top\mathbf{x}} \tag{19}$$

Finally, plugging this expression for λ back into the equation $\mathbf{w} = \mathbf{w}_0 - \lambda\mathbf{x}$, we have:

$$\mathbf{w} = \mathbf{w}_0 - \frac{\mathbf{x}^\top\mathbf{w}_0 - y}{\mathbf{x}^\top\mathbf{x}}\mathbf{x} \tag{20}$$

D. Proofs

First, we prove a well-known result on the implicit bias of gradient descent used to optimize convex quadratic functions (Lemma D.1), which we use in this section.

Lemma D.1 (Gradient Descent on a Quadratic Function Finds a Min-Norm Solution)**.** *Suppose we have a convex quadratic function:*

$$g(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{A}\mathbf{w} + \mathbf{b}^\top \mathbf{w} + c\,, \tag{21}$$

where $\mathbf{A} \in \mathbb{R}^{|\mathcal{W}|\times|\mathcal{W}|}$ *is symmetric positive semidefinite,* $\mathbf{b} \in \mathbb{R}^{|\mathcal{W}|}$*, and* $c \in \mathbb{R}$*. If we start from initialization* $\mathbf{w}_0$*, then gradient descent with an appropriate learning rate will converge to a solution* $\mathbf{w}^\star \in \arg\min_{\mathbf{w}} g(\mathbf{w})$ *which has minimum* L_2 *distance from* $\mathbf{w}_0$*:*

$$\mathbf{w}^\star \in \underset{\mathbf{w}\in\arg\min_{\mathbf{w}} g(\mathbf{w})}{\arg\min} \frac{1}{2}\|\mathbf{w}-\mathbf{w}_0\|_2^2\,. \tag{22}$$

Proof. Note that $\nabla_{\mathbf{w}} g(\mathbf{w}) = \mathbf{A}\mathbf{w}+\mathbf{b}$. By the lower-bounded assumption, the minimum of g is reached when $\nabla_{\mathbf{w}} g(\mathbf{w}) = 0$; one solution, which is the minimum-norm solution with respect to the origin, is $\mathbf{w} = -\mathbf{A}^+\mathbf{b}$, where $\mathbf{A}^+$ denotes the Moore-Penrose pseudoinverse of $\mathbf{A}$. The minimum-cost subspace is defined by:

$$\arg\min_{\mathbf{w}} g(\mathbf{w}) = \{\mathbf{w}^\star + \mathbf{w}' \mid \mathbf{w}' \in \mathcal{N}(\mathbf{A})\}\,, \tag{23}$$

where $\mathbf{w}^\star$ is any specific minimizer of g and $\mathcal{N}(\mathbf{A})$ denotes the nullspace of $\mathbf{A}$. The closed-form solution for the minimizer of g which minimizes the L_2 distance from some initialization $\mathbf{w}_0$ is:

$$\mathbf{w}^\star = -\mathbf{A}^+\mathbf{b} + (\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0\,. \tag{24}$$

Note that $(\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0$ is in the nullspace of $\mathbf{A}$, since $\mathbf{A}(\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0 = \mathbf{A}\mathbf{w}_0 - \mathbf{A}\mathbf{A}^+\mathbf{A}\mathbf{w}_0 = \mathbf{A}\mathbf{w}_0 - \mathbf{A}\mathbf{w}_0 = 0$. Next, we derive a closed-form expression for the result of k steps of gradient descent with a fixed learning rate α. We write the recurrence:

$$\begin{aligned}
\mathbf{w}_{k+1} &= \mathbf{w}_k - \alpha\nabla_{\mathbf{w}} g(\mathbf{w}_k) && (25)\\
&= \mathbf{w}_k - \alpha(\mathbf{A}\mathbf{w}_k + \mathbf{b}) && (26)
\end{aligned}$$

We can subtract the optimum from both sides, yielding:

$$\begin{aligned}
\mathbf{w}_{k+1} + \mathbf{A}^+\mathbf{b} &= \mathbf{w}_k - \alpha(\mathbf{A}\mathbf{w}_k + \mathbf{b}) + \mathbf{A}^+\mathbf{b} && (27)\\
&= \mathbf{w}_k - \alpha\mathbf{A}\mathbf{w}_k - \alpha\mathbf{b} + \mathbf{A}^+\mathbf{b} && (28)\\
&= \mathbf{w}_k - \alpha\mathbf{A}\mathbf{w}_k - \alpha\mathbf{A}\mathbf{A}^+\mathbf{b} + \mathbf{A}^+\mathbf{b} && (29)\\
&= (\mathbf{I}-\alpha\mathbf{A})\mathbf{w}_k + \mathbf{A}^+\mathbf{b} - \alpha\mathbf{A}\mathbf{A}^+\mathbf{b} && (30)\\
&= (\mathbf{I}-\alpha\mathbf{A})\mathbf{w}_k + (\mathbf{I}-\alpha\mathbf{A})\mathbf{A}^+\mathbf{b} && (31)\\
&= (\mathbf{I}-\alpha\mathbf{A})(\mathbf{w}_k + \mathbf{A}^+\mathbf{b}) && (32)
\end{aligned}$$

Thus, to obtain $\mathbf{w}_{k+1} + \mathbf{A}^+\mathbf{b}$, we simply multiply $\mathbf{w}_k + \mathbf{A}^+\mathbf{b}$ by $(\mathbf{I}-\alpha\mathbf{A})$. This allows us to write $\mathbf{w}_{k+1}$ as a function of the initialization $\mathbf{w}_0$:

$$\begin{aligned}
\mathbf{w}_{k+1} + \mathbf{A}^+\mathbf{b} &= (\mathbf{I}-\alpha\mathbf{A})^k(\mathbf{w}_0 + \mathbf{A}^+\mathbf{b}) && (33)\\
\mathbf{w}_{k+1} &= -\mathbf{A}^+\mathbf{b} + (\mathbf{I}-\alpha\mathbf{A})^k(\mathbf{w}_0 + \mathbf{A}^+\mathbf{b}) && (34)\\
&= -\mathbf{A}^+\mathbf{b} + (\mathbf{I}-\alpha\mathbf{A})^k((\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0 + \mathbf{A}^+\mathbf{A}\mathbf{w}_0 + \mathbf{A}^+\mathbf{b}) && (35)\\
&= -\mathbf{A}^+\mathbf{b} + (\mathbf{I}-\alpha\mathbf{A})^k((\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0) + (\mathbf{I}-\alpha\mathbf{A})^k(\mathbf{A}^+\mathbf{A}\mathbf{w}_0 + \mathbf{A}^+\mathbf{b}) && (36)\\
&= -\mathbf{A}^+\mathbf{b} + (\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0\,, && (37)
\end{aligned}$$

where the last equation follows from:

$$(\mathbf{I}-\alpha\mathbf{A})(\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0 = (\mathbf{I}-\alpha\mathbf{A}-\mathbf{A}^+\mathbf{A}+\alpha\mathbf{A}\mathbf{A}^+\mathbf{A})\mathbf{w}_0 = (\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0 \tag{38}$$

and thus $(\mathbf{I}-\alpha\mathbf{A})^k(\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0 = (\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0$, and the term $((\mathbf{I}-\mathbf{A}^+\mathbf{A})\mathbf{w}_0) + (\mathbf{I}-\alpha\mathbf{A})^k(\mathbf{A}^+\mathbf{A}\mathbf{w}_0 + \mathbf{A}^+\mathbf{b}$ goes to 0 as $k \to \infty$ because $\mathbf{A}^+\mathbf{A}\mathbf{w}_0$ and $\mathbf{A}^+\mathbf{b}$ are in the span of $\mathbf{A}$. Gradient descent will converge for learning rates $\alpha < 2\lambda_{\text{max}}^{-1}$, where λ_{max} is the maximum eigenvalue of $\mathbf{A}$. Thus, from Eq. 37 we see that gradient descent on the quadratic converges to the solution which minimizes the L_2 distance to the initialization $\mathbf{w}_0$. □

We also prove the following Lemma D.2, which we use in Theorem 3.1.

Lemma D.2. *Suppose f and F satisfy Assumption 3.1. Then, the function $F^\star(\mathbf{u}) \equiv F(\mathbf{u}, \mathbf{w}^\star)$, where $\mathbf{w}^\star$ is a solution to the inner problem, is a convex quadratic in $\mathbf{u}$ with a positive semi-definite curvature matrix.*

Proof. Any solution to the inner optimization problem for a given outer parameter $\mathbf{u}$, $\arg\min_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$, can be expressed as:

$$\mathbf{w}^\star = -\mathbf{A}^+(\mathbf{B}\mathbf{u} + \mathbf{d}) + \mathbf{c}\,, \tag{39}$$

where $\mathbf{c}$ is any element the kernel (or nullspace) of $\mathbf{A}$ and $\mathbf{A}^+$ is $\mathbf{A}$'s pseudoinverse. Plugging this inner solution into the outer objective F, we have:

$$F^\star(\mathbf{u}) \equiv F(\mathbf{w}^\star) = \frac{1}{2}\mathbf{w}^{\star\top}\mathbf{P}\mathbf{w}^\star + \mathbf{f}^\top\mathbf{w}^\star + h \tag{40}$$

$$= \frac{1}{2}\left(-\mathbf{A}^+\mathbf{B}\mathbf{u} - \mathbf{A}^+\mathbf{d} + \mathbf{c}\right)^\top \mathbf{P}\left(-\mathbf{A}^+\mathbf{B}\mathbf{u} - \mathbf{A}^+\mathbf{d} + \mathbf{c}\right) + \mathbf{f}^\top(-\mathbf{A}^+\mathbf{B}\mathbf{u} - \mathbf{A}^+\mathbf{d} + \mathbf{c}) + h \tag{41}$$

$$= \frac{1}{2}\mathbf{u}^\top \underbrace{\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{B}}_{\text{PSD}}\mathbf{u} + \mathbf{u}^\top(\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{d} - \mathbf{B}^\top\mathbf{A}^+\mathbf{f} - \mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{c}) \tag{42}$$

$$+ \left(\frac{1}{2}\mathbf{d}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{d} - \mathbf{f}^\top\mathbf{A}^+\mathbf{d} - \frac{1}{2}\mathbf{d}^\top\mathbf{A}^+\mathbf{P}\mathbf{c} + \frac{1}{2}\mathbf{c}^\top\mathbf{P}\mathbf{c} + \mathbf{f}^\top\mathbf{c} + h\right). \tag{43}$$

The final equation is a quadratic form in $\mathbf{u}$; we wish to show that the curvature matrix, $\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{B}$, is positive semi-definite. Note that $\mathbf{A}^+\mathbf{P}\mathbf{A}^+$ is PSD because $\mathbf{P}$ is PSD by assumption, and thus for any vector $\mathbf{v}$ we have:

$$\mathbf{v}^\top(\mathbf{A}^+\mathbf{P}\mathbf{A}^+)\mathbf{v} = \mathbf{v}^\top(\mathbf{A}^+)^\top\mathbf{P}\mathbf{A}^+\mathbf{v} = \left(\mathbf{A}^+\mathbf{v}\right)^\top\mathbf{P}\left(\mathbf{A}^+\mathbf{v}\right) \geq 0\,. \tag{44}$$

Next, because $\mathbf{A}^+\mathbf{P}\mathbf{A}^+$ is a PSD matrix, it can be expressed in the form $\mathbf{M}^\top\mathbf{M}$ for some PSD matrix $\mathbf{M}$ (e.g., the matrix square root of $\mathbf{A}^+\mathbf{P}\mathbf{A}^+$). Then, for any vector $\mathbf{u}$, we have:

$$\mathbf{u}^\top\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{B}\mathbf{u} = \mathbf{u}^\top\mathbf{B}^\top\mathbf{M}^\top\mathbf{M}\mathbf{B}\mathbf{u} = (\mathbf{M}\mathbf{B}\mathbf{u})^\top(\mathbf{M}\mathbf{B}\mathbf{u}) = \|\mathbf{M}\mathbf{B}\mathbf{u}\|_2^2 \geq 0 \tag{45}$$

Thus, $\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{B}$ is PSD. □

D.1. Proof of Statement 3.1

Statement D.1 (Cold-start BLO converges to a cold-start equilibrium.)**.** *Suppose f and F satisfy Assumption 3.1, and assume we have inner parameter initialization $\mathbf{w}_0$. Then, given appropriate learning rates for the inner and outer optimizations, the cold-start algorithm (Algorithm 1) using exact hypergradients converges to a cold-start equilibrium.*

Proof. By assumption, the inner objective f is a convex quadratic in $\mathbf{w}$ for each fixed $\mathbf{u}$, with positive semi-definite curvature matrix $\mathbf{A}$. By Lemma D.1, with an appropriately-chosen learning rate, the iterates of gradient descent in the inner loop of Algorithm 1 converge to the solution with minimum L_2 norm from the inner initialization $\mathbf{w}_0$: $\mathbf{w}_{k+1}^* = \arg\min_{\mathbf{w}\in\mathcal{S}(\mathbf{u}_k)} \frac{1}{2}\|\mathbf{w}-\mathbf{w}_0\|^2$. Because $\mathbf{w}_{k+1}^* \in \arg\min_{\mathbf{w}} f(\mathbf{w}, \mathbf{u}_k)$ and assuming that we compute the exact hypergradient, each outer step performs gradient descent on the objective $F^\star(\mathbf{u}) \equiv F(\mathbf{u}, \mathbf{w}^\star)$. By Lemma D.2, the outer objective $F^\star(\mathbf{u})$ is quadratic in $\mathbf{u}$ with a PSD curvature matrix, so that with an appropriate outer learning rate, the outer loop of Algorithm 1 will converge to a solution of $F^\star(\mathbf{u})$. Thus, we will have a final iterate $\mathbf{u}^\star \in \arg\min_{\mathbf{u}} F(\mathbf{u}, \mathbf{w}^\star)$ for which the corresponding inner solution is $\mathbf{w}^\star \in \arg\min_{\mathbf{w}\in\mathcal{S}(\mathbf{u}^\star)} \frac{1}{2}\|\mathbf{w}^\star - \mathbf{w}_0\|^2$, yielding a pair of inner and outer parameters $(\mathbf{u}^\star, \mathbf{w}^\star)$ that are a cold-start equilibrium. □

D.2. Proof of Theorem 3.1

Theorem D.3 (Cold-Start Outer Parameter Norm.)**.** *Suppose f and F satisfy Assumption 3.1, and suppose we run cold-start BLO (Algorithm 1) using the exact hypergradient, starting from outer parameter initialization $\mathbf{u}_0$. Assume that for each outer iteration, the inner parameters are re-initialized to $\mathbf{w}_0 = \mathbf{0}$ and optimized with an appropriate learning rate to convergence. Then cold-start BLO—with an appropriate learning rate for the outer optimization—converges to an outer solution $\mathbf{u}^\star$ with minimum L_2 distance from u_0:*

$$\mathbf{u}^\star = \operatorname*{arg\,min}_{\mathbf{u} \in \arg\min_{\mathbf{u}} F^\star(\mathbf{u})} \frac{1}{2}\|\mathbf{u} - \mathbf{u}_0\|^2 \tag{46}$$

Proof. Since the inner parameters are initialized at $\mathbf{w}_0 = \mathbf{0}$, the solution to the inner optimization problem found by gradient descent for a given outer parameter $\mathbf{u}$, $\arg\min_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$, can be expressed in closed-form as:

$$\mathbf{w}^\star = -\mathbf{A}^+(\mathbf{B}\mathbf{u} + \mathbf{d}), \tag{47}$$

where $\mathbf{A}^+$ denotes the Moore-Penrose pseudoinverse of $\mathbf{A}$. Plugging this min-norm inner solution into the outer objective F, we have:

$$F^\star(\mathbf{u}) \equiv F(\mathbf{w}^\star) = \frac{1}{2}\mathbf{w}^{\star\top}\mathbf{P}\mathbf{w}^\star + \mathbf{f}^\top\mathbf{w}^\star + h \tag{48}$$

$$= \frac{1}{2}\mathbf{u}^\top \underbrace{\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{B}}_{\text{PSD}}\mathbf{u} + \mathbf{u}^\top(\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{d} - \mathbf{B}^\top\mathbf{A}^+\mathbf{f}) + \left(\frac{1}{2}\mathbf{d}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{d} - \mathbf{f}^\top\mathbf{A}^+\mathbf{d} + h\right) \tag{49}$$

Then $F^\star(\mathbf{u})$ is quadratic in $\mathbf{u}$, and by Lemma D.2, the curvature matrix $\mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{B}$ is positive semi-definite. Let $\mathbf{Z} \equiv \mathbf{B}^\top\mathbf{A}^+\mathbf{P}\mathbf{A}^+\mathbf{B}$. Similarly to the analysis in Lemma D.1, the iterates $\mathbf{u}_k$ of gradient descent with learning rate α are given by:

$$\mathbf{u}_k = \mathbf{u}^\star + (\mathbf{I} - \alpha\mathbf{Z})^k(\mathbf{u}_0 - \mathbf{u}^\star), \tag{50}$$

where

$$\mathbf{u}^\star = \operatorname*{arg\,min}_{\mathbf{u} \in \arg\min_{\mathbf{u}} F^\star(\mathbf{u})} \frac{1}{2}\|\mathbf{u} - \mathbf{u}_0\|^2. \tag{51}$$

From Eq. (50), we see that the iterates of gradient descent converge exponentially to $\mathbf{u}^\star$, which is the outer solution with minimum L_2 distance from the outer initialization $\mathbf{u}_0$. □

D.3. Proof of Remark 3.2

Remark D.4 (Equivalence of Full Warm-Start and Cold-Start in the Strongly Convex Regime)**.** *When the inner problem $f(\mathbf{u}, \mathbf{w})$ is strongly convex in $\mathbf{w}$ for each $\mathbf{u}$, then the solution to the inner problem is unique. In this case, full warm-start (Algorithm 2 with $T \to \infty$) and cold-start (Algorithm 1), using exact hypergradients, are equivalent.*

Proof. If $f(\mathbf{u}, \mathbf{w})$ is strongly convex in $\mathbf{w}$ for each $\mathbf{u}$, then it has a unique global minimum for each $\mathbf{u}$. Thus, given an appropriate learning rate for the inner optimization, repeated application of the update Ξ will converge to this unique solution for any inner parameter initialization. That is, $\Xi^{(\infty)}(\mathbf{u}, \mathbf{w}_{\text{init}}) = \arg\min_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$ for any initialization $\mathbf{w}_{\text{init}} \in \mathcal{W}$. In particular, the fixpoint will be identical for cold-start and full warm-start, $\Xi^{(\infty)}(\mathbf{u}, \mathbf{w}_0) = \Xi^{(\infty)}(\mathbf{u}, \mathbf{w}_k)$ for any $\mathbf{w}_0$ and $\mathbf{w}_k$. Therefore, the inner solutions are identical, and yield identical hypergradients, so the iterates of the full warm-start and cold-start algorithms are equivalent. □

D.4. Proof of Statement 3.2

Statement D.2 (Inclusion of Partial Warm-Start Equilibria)**.** *Every partial warm-start equilibrium (with $T = 1$) is a full warm-start equilibrium ($T \to \infty$). In addition, if $\Xi(\mathbf{u}, \mathbf{w}) = \mathbf{w} - \alpha\nabla_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$ with a fixed (non-decayed) step size α, then the corresponding full-warm start equilibria are also partial warm-start equilibria.*

Proof. Let $(\mathbf{u}^\star, \mathbf{w}^\star)$ be an arbitrary partial warm-start equilibrium for $T = 1$. By definition, $\mathbf{u}^\star \in \arg\min_{\mathbf{u}} F(\mathbf{u}, \mathbf{w}^\star)$ and $\mathbf{w}^\star = \Xi^{(1)}(\mathbf{u}, \mathbf{w}^\star)$. Thus, $\nabla_{\mathbf{w}} f(\mathbf{u}, \mathbf{w}) = 0$, which entails that $\mathbf{w}^\star = \Xi^{(\infty)}(\mathbf{u}, \mathbf{w}^\star)$. Hence, $(\mathbf{u}^\star, \mathbf{w}^\star)$ is a full warm-start equilibrium.

Next, let $(\mathbf{u}^\star, \mathbf{w}^\star)$ be an arbitrary full warm-start equilibrium. By definition, this means that $\mathbf{w}^\star = \Xi^{(\infty)}(\mathbf{u}^\star, \mathbf{w}^\star)$. By assumption, $\Xi(\mathbf{u}, \mathbf{w}) = \mathbf{w} - \alpha \nabla_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$ with a fixed step size α. The fixed step size combined with the $T \to \infty$ limit excludes the possibility that there is a finite-length cycle such that the inner optimization arrives back to the initial point $\mathbf{w}^\star$ after a finite number of gradient steps T. Thus, we must have $\Xi^{(1)}(\mathbf{u}, \mathbf{w}) = \mathbf{w}$, so $(\mathbf{u}^\star, \mathbf{w}^\star)$ is a partial warm-start equilibrium. □

E. Proximal Best-Response

Consider the proximal objective $\hat{f}(\mathbf{u}, \mathbf{w}) = f(\mathbf{u}, \mathbf{w}) + \frac{\epsilon}{2} ||\mathbf{w} - \mathbf{w}'||^2$. Here we will treat $\mathbf{w}_k$ as a constant (e.g., we won't consider its dependence on $\mathbf{u}$). Let $\mathbf{w}^\star(\mathbf{u}) \in \arg\min_{\mathbf{w}} \hat{f}(\mathbf{u}, \mathbf{w})$ be a fixed point of $\hat{f}$. We want to compute the response Jacobian $\frac{\partial \mathbf{w}^\star(\mathbf{u})}{\partial \mathbf{u}}$. Since $\mathbf{w}^\star$ is a fixed point, we have:

$$\frac{\partial \hat{f}(\mathbf{u}, \mathbf{w}^\star(\mathbf{u}))}{\partial \mathbf{w}} = 0 \tag{52}$$

$$\frac{\partial f(\mathbf{u}, \mathbf{w}^\star(\mathbf{u}))}{\partial \mathbf{w}} + \epsilon(\mathbf{w}^\star(\mathbf{u}) - \mathbf{w}') = 0 \tag{53}$$

$$\frac{\partial}{\partial \mathbf{u}} \frac{\partial f(\mathbf{u}, \mathbf{w}^\star(\mathbf{u}))}{\partial \mathbf{w}} + \epsilon \frac{\partial \mathbf{w}^\star(\mathbf{u})}{\partial \mathbf{u}} = 0 \tag{54}$$

$$\frac{\partial^2 f(\mathbf{u}, \mathbf{w}^\star(\mathbf{u}))}{\partial \mathbf{u} \partial \mathbf{w}} + \frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top} \frac{\partial \mathbf{w}^\star(\mathbf{u})}{\partial \mathbf{u}} + \epsilon \frac{\partial \mathbf{w}^\star(\mathbf{u})}{\partial \mathbf{u}} = 0 \tag{55}$$

$$\left(\frac{\partial^2 f}{\partial \mathbf{w}^2} + \epsilon I \right) \frac{\partial \mathbf{w}^\star(\mathbf{u})}{\partial \mathbf{u}} = - \frac{\partial^2 f(\mathbf{u}, \mathbf{w}^\star(\mathbf{u}))}{\partial \mathbf{u} \partial \mathbf{w}} \tag{56}$$

$$\frac{\partial \mathbf{w}^\star}{\partial \mathbf{u}} = - \left(\frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{w}^\top} + \epsilon I \right)^{-1} \frac{\partial^2 f}{\partial \mathbf{u} \partial \mathbf{w}} \tag{57}$$

F. Equivalence Between Unrolling and Neumann Hypergradients

In this section, we review the result from (Lorraine et al., 2020), which shows that when we are at a converged solution to the inner problem $\mathbf{w}^\star \in \mathcal{S}(\mathbf{u})$, then computing the hypergradient by differentiating through k steps of unrolled gradient descent on the inner objective is equivalent to computing the hypergradient with the k-term truncated Neumann series approximation to the inverse Hessian.

In this derivation, the inner and outer objectives are arbitrary—we do not need to assume that they are quadratic. The SGD recurrence for unrolling the inner optimization is:

$$\mathbf{w}_{i+1} = \mathbf{w}_i(\mathbf{u}) - \alpha \frac{\partial f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u})} \tag{58}$$

Then,

$$\frac{\partial \mathbf{w}_{i+1}}{\partial \mathbf{u}} = \frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}} - \alpha \frac{\partial}{\partial \mathbf{w}_i(\mathbf{u})} \left(\frac{\partial f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u})} \frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}} + \frac{\partial f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{u}} \right) \tag{59}$$

$$= \frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}} - \alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{w}_i(\mathbf{u})} \frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}} - \alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{u}} \tag{60}$$

$$= -\alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{u}} + \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{w}_i(\mathbf{u})} \right) \frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}} \tag{61}$$

We can similarly expand out $\frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}}$ as:

$$\frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}} = -\alpha \frac{\partial^2 f(\mathbf{w}_{i-1}(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_{i-1} \partial \mathbf{u}} + \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}_{i-1}(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_{i-1}(\mathbf{u}) \partial \mathbf{w}_{i-1}(\mathbf{u})} \right) \frac{\partial \mathbf{w}_{i-1}(\mathbf{u})}{\partial \mathbf{u}} \tag{62}$$

Plugging in this expression for $\frac{\partial \mathbf{w}_i(\mathbf{u})}{\partial \mathbf{u}}$ into the expression for $\frac{\partial \mathbf{w}_{i+1}(\mathbf{u})}{\partial \mathbf{u}}$, we have:

$$\frac{\partial \mathbf{w}_{i+1}(\mathbf{u})}{\partial \mathbf{u}} = -\alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{u}} + \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{w}_i(\mathbf{u})} \right) \tag{63}$$

$$\times \left[-\alpha \frac{\partial^2 f(\mathbf{w}_{i-1}(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_{i-1}(\mathbf{u}) \partial \mathbf{u}} + \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}_{i-1}(\mathbf{u}), \mathbf{u})}{\partial \mathbf{u}_{i-1}(\mathbf{u}) \partial \mathbf{w}_{i-1}(\mathbf{u})} \right) \frac{\partial \mathbf{w}_{i-1}(\mathbf{u})}{\partial \mathbf{u}} \right] \tag{64}$$

Expanding, we have:

$$\frac{\partial \mathbf{w}_{i+1}(\mathbf{u})}{\partial \mathbf{u}} = -\alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{u}} + \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}_i(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_i(\mathbf{u}) \partial \mathbf{w}_i(\mathbf{u})} \right) \left(-\alpha \frac{\partial^2 f(\mathbf{w}_{i-1}(\mathbf{u}), \mathbf{u})}{\partial \mathbf{w}_{i-1}(\mathbf{u}) \partial \mathbf{u}} \right) \tag{65}$$

$$+ \prod_{k<j} \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}, \mathbf{u})}{\partial \mathbf{w} \partial \mathbf{w}^\top} \bigg|_{\mathbf{u}, \mathbf{w}_{i-k}(\mathbf{u})} \right) \frac{\partial \mathbf{w}_{i-1}(\mathbf{u})}{\partial \mathbf{u}} \tag{66}$$

Telescoping this sum, we have:

$$\frac{\partial \mathbf{w}_{i+1}(\mathbf{u})}{\partial \mathbf{u}} = \sum_{j \le i} \left[\prod_{k<j} \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}, \mathbf{u})}{\partial \mathbf{w} \partial \mathbf{w}^\top} \bigg|_{\mathbf{u}, \mathbf{w}_{i-k}(\mathbf{u})} \right) \right] \left(-\alpha \frac{\partial^2 f(\mathbf{u})}{\partial \mathbf{w} \partial \mathbf{u}} \bigg|_{\mathbf{u}, \mathbf{w}_{i-j}(\mathbf{u})} \right) \tag{67}$$

If we start unrolling from a stationary point of the proximal objective, then all the $\mathbf{w}_i$ will be equal, so this simplifies to:

$$\frac{\partial \mathbf{w}_{i+1}(\mathbf{u})}{\partial \mathbf{u}} = \left[\sum_{j \le i} \left(\mathbf{I} - \alpha \frac{\partial^2 f(\mathbf{w}, \mathbf{u})}{\partial \mathbf{w} \partial \mathbf{w}^\top} \right)^j \right] \left(-\alpha \frac{\partial^2 f(\mathbf{w}, \mathbf{u})}{\partial \mathbf{w} \partial \mathbf{u}} \right) \tag{68}$$

This recovers the Neumann series approximation to the inverse Hessian.

G. Experimental Details and Extended Results

Compute Environment. All experiments were implemented using JAX (Bradbury et al., 2018), and were run on NVIDIA P100 GPUs. Each instance of the dataset distillation and antidistillation task took approximately 5 minutes of compute on a single GPU.

G.1. Details and Extended Results for Dataset Distillation.

Details. For our dataset distillation experiments, we trained a 4-layer MLP with 200 hidden units per layer and ReLU activations. For warm-start joint optimization, we computed hypergradients by differentiating through $K = 1$ steps of unrolling, and updated the hyperparameters (learned datapoints) and MLP parameters using alternating gradient descent, with one step on each. We used SGD with learning rate 0.001 for the inner optimization and Adam with learning rate 0.01 for the outer optimization.

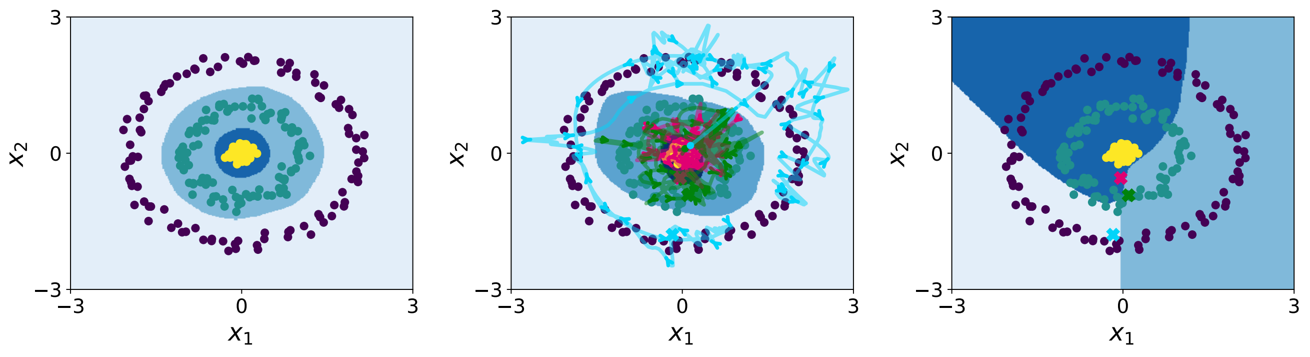
Link to Animations. Here is a link to an document containing animations of the bilevel training dynamics for the dataset distillation task. We visualize the dynamics of warm-started bilevel optimization in the setting where the inner problem is overparameterized, by animating the trajectories of the learned datapoints over time, and showing how the decision boundary of the model changes over the course of joint optimization.

Extended Results. Here, we show additional dataset distillation results, using a similar setup to Section 4.1.

Figure 8 shows the results where we fit a three-class problem (e.g., three concentric rings) using three learned datapoints. Figure 9 shows the results for fitting three classes using *only two datapoints*, where both the coordinates and *soft labels* are learned. For each training datapoint, in addition to learning its x- an y-coordinates, we learn a C-dimensional vector (where C is the number of classes, in this example $C = 3$) representing the unnormalized class label: this vector is normalized with a softmax when we perform cross-entropy training of the inner model (e.g., we do not train on one-hot labels). Joint adaptation of the model parameters and learned data is able to fit three classes by changing the learned class label for one of the datapoints during training.

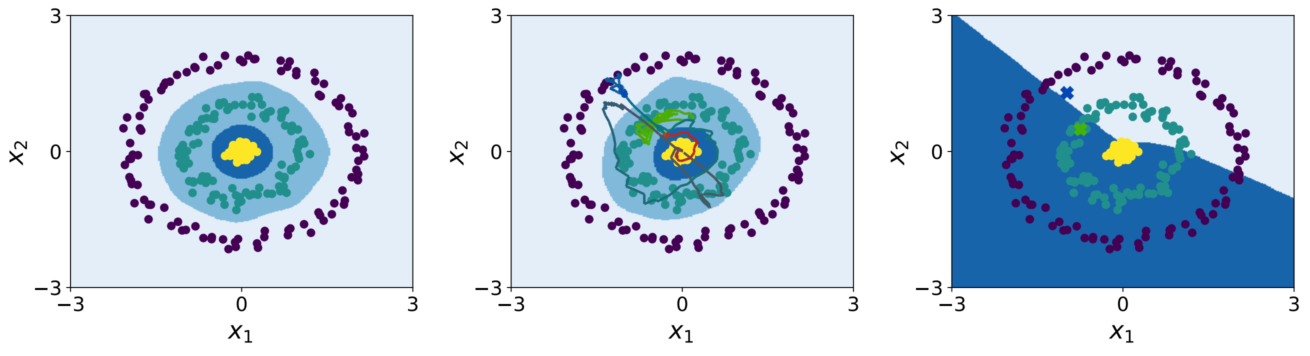
	Dataset Distillation			Data Augmentation Net		
Method	**MNIST**	**Fashion**	**C-10**	**MNIST**	**Fashion**	**C-10**
Cold-Start	30.7 / 89.1	33.2 / 83.0	17.6 / 46.9	84.86	84.04	45.38
Warm-Start	90.8 / 97.5	88.2 / 94.2	50.3 / 59.8	92.81	89.51	59.30
Warm-Start + **Re-Train**	12.9 / 17.1	7.0 / 12.6	10.2 / 8.9	9.15	25.32	11.12

Table 5: **Cols 1-3:** Accuracy on original data with 1/10 synthetic samples. **Cols 4-6:** Learning a data augmentation network.



(a) Training on original data. (b) Warm-start joint optimization. (c) Re-training on final points.

Figure 8: Dataset distillation results fitting three classes with three learned datapoints. Similarly to the results in Sec. 4.1, when using warm-start joint optimization, the three learned datapoints are adapted during training to trace out the data in their respective classes, guiding the network to learn a decision boundary that performs well on the original data. Cold-start re-training yields a model that correctly classifies the three learned datapoints, but has poor validation performance.



(a) Training on original data. (b) Warm-start joint optimization. (c) Re-training on final points.

Figure 9: Dataset distillation results fitting three classes with *only two* learned datapoints. In the warm-start plot, the color of the trajectory of each learned datapoint indicates its soft class membership, with magenta, green, and blue corresponding to the inner, middle, and outer rings, respectively. Darker/gray colors indicate soft labels that place approximately equal probability on each class. We see that although we only have two learned datapoints, the class labels change over the course of training such that all three classes are covered. Cold-start re-training yields a model that correctly classifies the three learned datapoints, but has poor validation performance.

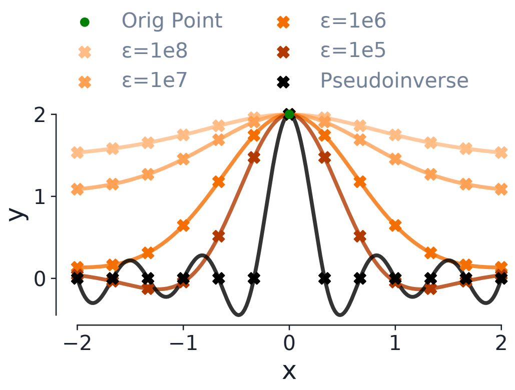


Figure 10: **Antidistillation task for linear regression with an overparameterized outer objective.** This plot uses the same setup as Figure 7, but shows learned datapoints obtained via the proximal best-response Jacobian, for various damping factors ϵ.

G.2. Details and Extended Results for Anti-Distillation.

Details. For the anti-distillation results in Section 4.2, we used a Fourier basis consisting of 10 $\sin$ terms, 10 $\cos$ terms, and a bias, yielding 21 total inner parameters. For the exponential-amplitude Fourier basis used in Section 4.2, we used SGD with learning rates 1e-8 and 1e-2 for the inner and outer parameters, respectively; for the $1/n$ amplitude Fourier basis (discussed below, and used for Figure 12), we used SGD with learning rates 1e-3 and 1e-2 for the inner and outer parameters, respectively.

Approximate Hypergradient Visualization. For the visualization in Figure 7c, the experiment setup is as follows: we have a single original dataset point at xy coordinate $(1, 1)$, and we learn the y-coordinates of two synthetic datapoints, which we initialize at xy-coordinates $(0, 0)$ and $(2, 0)$, respectively. The inner model trained on these two synthetic datapoints is a linear regressor with one weight and one bias, e.g., $y = wx + b$. The outer problem is overparameterized, because there are many valid settings of the learned datapoints such that the inner model trained on those datapoints will fit the point $(1, 1)$. For example, three valid solutions for the learned datapoints are $\{(0, 0), (2, 2)\}$, $\{(0, 1), (2, 1)\}$, and $\{(0, 2), (2, 0)\}$. The set of such valid solutions is visualized in Figure 7c, as well as the gradients using different hypergradient approximations, outer optimization trajectories, and converged outer solutions with each approximation. We focused on Neumann series approximations with different k, and ran the Neumann series at the converged inner parameters in each case.

Conjugate Gradient. Here, we present results using truncated conjugate gradient to approximate the inverse Hessian vector product when using the implicit function theorem to compute the hypergradient. We use the same problem setup as in Section 4.2, where we have a single validation datapoint and 13 learned datapoints. Because our Fourier basis consists of 10 $\sin$ terms, 10 $\cos$ terms, and a bias, we have 21 inner parameters, and using 21 steps of CG is guaranteed to yield the true inverse-Hessian vector product (barring any numerical issues). In Figure 11, we show the effect of using truncated CG iterations on the learned datapoints: we found that while Neumann iterations, truncated unrolling, and proximal optimization all yield nearly identical results, CG produces a very different inductive bias.

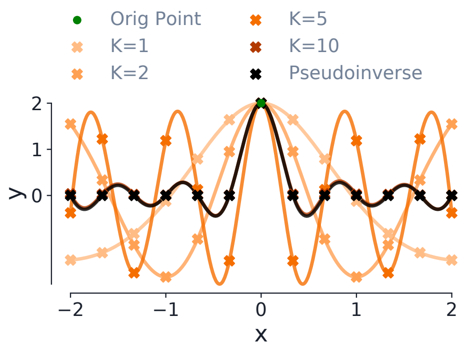
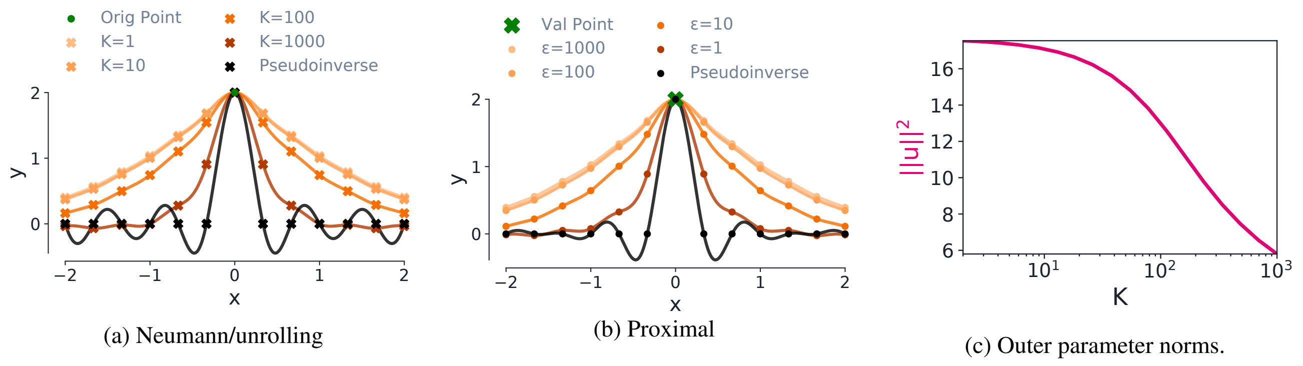


Figure 11: Truncated CG used to compute hypergradients for the anti-distillation task.

An Alternate Set of Fourier Basis Functions. In Section 4.2, we used a Fourier basis in which the lower-frequency terms had exponentially larger amplitudes than the high-frequency terms. Figure 12 presents results using an alternative feature function: $\phi(x) = a_0 + \sum_{n=1}^{N} \left(a_n \left(\frac{1}{n}\right) \cos(nx) + b_n \left(\frac{1}{n}\right) \sin(nx)\right)$.

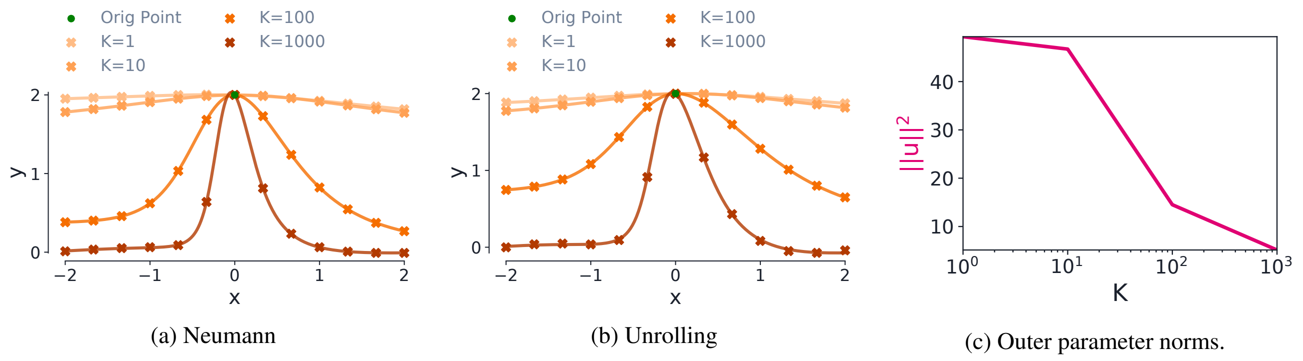
Using an MLP. Finally, we show that similar conclusions hold when training a multi-layer perceptron (MLP) on the anti-distillation task. We used a 2-layer MLP with 10 hidden units per layer and ReLU activations. We used SGD with learning rate 0.01 for both the inner and outer parameters. Figure 13 shows the learned datapoints and model fits resulting from



(a) Neumann/unrolling (b) Proximal (c) Outer parameter norms.

Figure 12: Fourier-basis 1D linear regression, where the outer objective is overparameterized. We learn 13 synthetic datapoints such that a regressor trained on those points will fit a single "validation" datapoint, shown by the green X at $(0, 2)$. The synthetic datapoints are initialized at linearly-spaced x-coordinates, with y-coordinate 0, and we only learn the targets $\mathbf{y}$. In the Fourier basis we use, lower frequency components have larger amplitudes. Here, we use the $1/n$ amplitude scheme.

running several different steps of Neumann iterations or unrolling, as well as the norms of the inner and outer parameters as a function of K. For the Neumann experiments, we first optimize the MLP for 5000 steps to reach approximate convergence of the inner problem, before running K Neumann iterations—the MLP is re-trained from scratch for 5000 steps for each outer parameter update (e.g., cold-started).



(a) Neumann (b) Unrolling (c) Outer parameter norms.

Figure 13: 1D linear regression with an overparameterized outer objective, where we train an MLP rather than performing Fourier-basis function regression. Note that here we cannot analytically compute the proximal solution as in the linear regression case, so we only include results for full-unrolls with truncated Neumann approximations of the inverse Hessian, and alternating gradient descent with various numbers of unroll steps.

H. Algorithms

Algorithm 1 Cold-Start BLO

```
1: Input: $\alpha$: inner LR, $\beta$: outer LR
2:     $T$: number of inner optimization steps
3:     $K$: number of Neumann series terms
4: while $||\hat{\nabla}_{\mathbf{u}} F(\mathbf{u}, \mathbf{w})||^2 > 0$, iteration $t$ do
5:     $\hat{\nabla}_{\mathbf{u}} F \leftarrow \text{HypergradApprox}(\mathbf{u}_t, \mathbf{w}_0, T, K)$
6:     $\mathbf{u}_{t+1} \leftarrow \mathbf{u}_t - \beta \hat{\nabla}_{\mathbf{u}} F$
7:
8: end while
9: return $(\mathbf{u}_t, \text{Optimize}(\mathbf{u}_t, \mathbf{w}_0, T))$
```

Algorithm 2 Warm-start BLO

```
1: Input: $\alpha$: inner LR, $\beta$: outer LR
2:     $T$: number of inner optimization steps
3:     $K$: number of Neumann series terms
4: while $||\hat{\nabla}_{\mathbf{u}} F(\mathbf{u}, \mathbf{w})||^2 > 0$, iteration $t$ do
5:     $\hat{\nabla}_{\mathbf{u}} F \leftarrow \text{HypergradApprox}(\mathbf{u}_t, \mathbf{w}_t, T, K)$
6:     $\mathbf{u}_{t+1} \leftarrow \mathbf{u}_t - \beta \hat{\nabla}_{\mathbf{u}} F$
7:     $\mathbf{w}_{t+1} \leftarrow \text{Optimize}(\mathbf{u}_{t+1}, \mathbf{w}_t, T)$
8: end while
9: return $(\mathbf{u}_t, \mathbf{w}_t)$
```

Figure 14: **Algorithms for cold-start and warm-start bilevel optimization.** We highlight the key differences in cyan for cold-start and orange for warm-start.

Algorithm 3 NeumannHypergrad

```
1: Input: $\alpha$: inner LR, $\beta$: outer LR
2:     $T$: inner steps
3:     $K$: Neumann terms
4: $\hat{\mathbf{w}}_T^* \leftarrow \text{Optimize}(\mathbf{u}_t, \mathbf{w}_t, T)$
5: $\frac{d\hat{\mathbf{w}}_T^*}{d\mathbf{u}} \leftarrow \text{BRJ-Approx}(\hat{\mathbf{w}}_T^*, \mathbf{u}_t, K)$
6: $\hat{\nabla}_{\mathbf{u}} F \leftarrow \frac{\partial F}{\partial \mathbf{u}} + \left(\frac{d\hat{\mathbf{w}}_T^*}{d\mathbf{u}}\right)^\top \frac{\partial F(\mathbf{u}, \hat{\mathbf{w}}_T^*)}{\partial \hat{\mathbf{w}}_T^*}$
7: return $\hat{\nabla}_{\mathbf{u}} F$
```

Algorithm 4 $\text{Optimize}(\mathbf{u}, \mathbf{w}, T)$

```
1: Input: $\gamma$: learning rate
2:     $T$: unroll steps
3:     $\mathbf{w}$: initialization
4: for $t = 1, \ldots, T$ do
5:     $\mathbf{w} \leftarrow \mathbf{w} - \gamma \nabla_{\mathbf{w}} f(\mathbf{u}, \mathbf{w})$
6: end for
7: return $\mathbf{w}$
```

Algorithm 5 BRJ-Approx

```
1: return
   $\alpha \sum_{j=0}^{K} \left(I - \alpha \frac{\partial^2 f(\mathbf{u}, \mathbf{w})}{\partial \mathbf{w} \partial \mathbf{w}^\top}\right)^j$
```

Figure 15: Helper functions to compute the implicit hypergradient using the Neumann series to approximate the inverse inner loss Hessian.

Algorithm 6 IterativeDiffHypergrad

```
1: Input: $\alpha$: inner LR, $\beta$: outer LR
2:     $T$: inner steps
3: $\hat{\mathbf{w}}_T^* \leftarrow \text{Optimize}(\mathbf{u}_t, \mathbf{w}_t, T)$
4: $\hat{F} \leftarrow F(\mathbf{u}, \hat{\mathbf{w}}_T^*)$
5: Compute $\nabla_{\mathbf{u}} \hat{F}$ using auto-diff.
```