Bilevel learning for imaging problems

Joint work with Mohammad Sadegh Salehi, Matthias Ehrhardt (Bath), Subhadip Mukherjee (IIT Kharagpur)

Lindon Roberts, University of Melbourne (lindon.roberts@unimelb.edu.au)
Supported by the Australian Research Council (DE240100006)

Applied Mathematics Seminar, University of Melbourne
11 September 2025

Key references

This talk is based on work in

- M. J. Ehrhardt & LR. Analyzing inexact hypergradients for bilevel learning. *IMA J. Applied Mathematics* (2024).
- M. S. Salehi, S. Mukherjee, LR & M. J. Ehrhardt. An Adaptively Inexact First-Order Method for Bilevel Optimization with Application to Hyperparameter Learning. *SIAM J. Mathematics of Data Science* (2025).
- M. S. Salehi, S. Mukherjee, LR & M. J. Ehrhardt. Bilevel Learning with Inexact Stochastic Gradients. *Scale Space and Variational Methods in Computer Vision* (2025).

Outline

Variational Regularization

We wish to solve inverse problems (here, in imaging) of variational regularization type.

Variational regularization problem

Suppose we have an object of interest $x^* \in \mathcal{X}$, a measurement operator A and some observed data $y^* \approx Ax^*$.

We wish to find x^* given y^* by solving

$$\min_{x \in \mathcal{X}} \mathcal{D}(Ax, y^*) + \mathcal{R}(x),$$

where $\mathcal{D}(y_1, y_2)$ is a measure of distance and $\mathcal{R}(x)$ is a regularizer encouraging solutions of a given type.

[Chambolle & Pock, 2016]

Example: Image denoising

Example (image denoising): given a noisy image y, find a denoised image x by solving

$$\min_x \underbrace{\frac{1}{2}\|x-y\|_2^2}_{\mathcal{D}(x,y)} + \underbrace{\alpha \mathrm{TV}(x)}_{\mathcal{R}(x)}$$

where $\alpha > 0$ and $\mathrm{TV}(x) = \|\nabla x\|_1$ is the total variation of an image (discretized into a sum over pixels).

<u>Goal:</u> find $x \approx y$ with small total variation (approx. piecewise constant).

Example: Image denoising

Example (image denoising): given a noisy image y, find a denoised image x by solving

$$\min_x \underbrace{\frac{1}{2}\|x-y\|_2^2}_{\mathcal{D}(x,y)} + \underbrace{\alpha \mathrm{TV}(x)}_{\mathcal{R}(x)}$$

where $\alpha > 0$ and $\mathrm{TV}(x) = \|\nabla x\|_1$ is the total variation of an image (discretized into a sum over pixels).

<u>Goal:</u> find $x \approx y$ with small total variation (approx. piecewise constant).

We will need to consider a smoothed version of TV to meet our assumptions,

$$\min_x \underbrace{\frac{1}{2}\|x-y\|_2^2}_{\mathcal{D}(x,y)} + \alpha \underbrace{\sum_j \sqrt{\|\nabla x_j\|_2^2 + \nu^2}}_{\approx \mathrm{TV}(x)}$$

Example: Image denoising

$$\min_x \underbrace{\frac{1}{2}\|x - y\|_2^2}_{\mathcal{D}(x,y)} + \alpha \underbrace{\sum_j \sqrt{\|\nabla x_j\|_2^2 + \nu^2}}_{\approx \mathrm{TV}(x)}$$

Issue: the solution depends on regularizer parameters $\alpha, \nu > 0$!

Example: Image denoising

$$\min_{x} \underbrace{\frac{1}{2}\|x - y\|_2^2}_{\mathcal{D}(x,y)} + \alpha \underbrace{\sum_{j} \sqrt{\|\nabla x_j\|_2^2 + \nu^2}}_{\approx \mathrm{TV}(x)}$$

Issue: the solution depends on regularizer parameters $\alpha, \nu > 0$!

Original image

Noisy image

Image source: University of Melbourne

Example: Image denoising

$$\min_x \underbrace{\frac{1}{2}\|x-y\|_2^2}_{\mathcal{D}(x,y)} + \underbrace{\alpha \sum_j \sqrt{\|\nabla x_j\|_2^2 + \nu^2}}_{\approx \mathrm{TV}(x)}$$

Issue: the solution depends on regularizer parameters $\alpha, \nu > 0$!

$\log \alpha = -4$, $\log \nu = -3$
PSNR = 21.7 dB

$\log \alpha = -2$, $\log \nu = -3$
PSNR = 25.1 dB

$\log \alpha = 0$, $\log \nu = -3$
PSNR = 20.6 dB

Example: Image denoising

$$\min_x \underbrace{\frac{1}{2}\|x-y\|_2^2}_{\mathcal{D}(x,y)} + \alpha \underbrace{\sum_j \sqrt{\|\nabla x_j\|_2^2 + \nu^2}}_{\approx \mathrm{TV}(x)}$$

Issue: the solution depends on regularizer parameters $\alpha, \nu > 0$!

$\log \alpha = -2$, $\log \nu = -5$	$\log \alpha = -2$, $\log \nu = -3$	$\log \alpha = -2$, $\log \nu = -1$
PSNR = 24.4 dB	**PSNR = 25.1 dB**	**PSNR = 23.6 dB**

Choosing Parameters

Recovered solution depends strongly on problem parameters (e.g. α, ν)

Question

How to choose good problem parameters?

Choosing Parameters

Recovered solution depends strongly on problem parameters (e.g. α, ν)

Question

How to choose good problem parameters?

- Trial & error
- L-curve criterion
- **Bilevel learning** — data-driven approach

Outline

1. Simple example: image denoising
2. **Bilevel learning**
3. Calculating hypergradients
4. Dynamic linesearch
5. Inexact SGD

Bilevel Learning

Suppose we have training data $(x_1, y_1), \ldots, (x_n, y_n)$ — ground truth and noisy observations.

Attempt to recover x_i from y_i by solving inverse problem with parameters $\theta \in \mathbb{R}^m$:

$$\hat{x}_i(\theta) := \arg\min_x g_i(x, \theta), \qquad \text{e.g. } g_i(x, \theta) = \mathcal{D}(Ax, y_i) + \mathcal{R}(x, \theta).$$

Try to find θ by making $\hat{x}_i(\theta)$ close to x_i

$$\min_\theta \frac{1}{n} \sum_{i=1}^{n} \|\hat{x}_i(\theta) - x_i\|^2 + \mathcal{J}(\theta),$$

with optional (smooth) term $\mathcal{J}(\theta)$ to encourage particular choices of θ.

Bilevel Optimization

The bilevel learning problem is:

$$\min_{\theta} \quad F(\theta) := \frac{1}{n}\sum_{i=1}^{n} \|\hat{x}_i(\theta) - x_i\|^2 + \mathcal{J}(\theta),$$
$$\text{s.t.} \quad \hat{x}_i(\theta) := \arg\min_{x} g_i(x, \theta), \quad \forall i = 1, \ldots, n.$$

- If g_i are strongly convex in x and sufficiently smooth in x and θ, then $\hat{x}_i(\theta)$ is well-defined and continuously differentiable.
- Upper-level problem ($\min_\theta F(\theta)$) is a smooth nonconvex optimization problem

Many use cases in data science: learning image regularizers, hyperparameter tuning, data hypercleaning, ...

Bilevel Learning

Difficulty?

Bilevel learning is just a smooth nonconvex problem — where is the challenge?

Difficulty?

Bilevel learning is just a smooth nonconvex problem — where is the challenge?

- Can't evaluate lower-level minimizers $\hat{x}_i(\theta)$ exactly, so can never get exact $F(\theta)$ or $\nabla F(\theta)$ [Kunisch & Pock, 2013; Sherry et al., 2020]
- And more to come...

Bilevel Learning

Difficulty?

Bilevel learning is just a smooth nonconvex problem — where is the challenge?

- Can't evaluate lower-level minimizers $\hat{x}_i(\theta)$ exactly, so can never get exact $F(\theta)$ or $\nabla F(\theta)$ [Kunisch & Pock, 2013; Sherry et al., 2020]
- And more to come...

Key question 1: how can we approximate $\nabla F(\theta)$, and how accurate is this approximation?

Bilevel Learning

Difficulty?

Bilevel learning is just a smooth nonconvex problem — where is the challenge?

- Can't evaluate lower-level minimizers $\hat{x}_i(\theta)$ exactly, so can never get exact $F(\theta)$ or $\nabla F(\theta)$ [Kunisch & Pock, 2013; Sherry et al., 2020]
- And more to come...

Key question 1: how can we approximate $\nabla F(\theta)$, and how accurate is this approximation?

Note: error in $F(\theta)$ approximation is easy to bound from μ-strong convexity, $\|x - \hat{x}_i(\theta)\| \leq \frac{1}{\mu}\|\nabla_x g_i(x, \theta)\|$

Outline

1. Simple example: image denoising
2. Bilevel learning
3. **Calculating hypergradients**
4. Dynamic linesearch
5. Inexact SGD

Hypergradient

Consider the simple bilevel problem:

$$\min_{\theta \in \mathbb{R}^n} \quad F(\theta) := f(x^*(\theta)), \qquad \text{s.t.} \quad x^*(\theta) := \arg\min_{y \in \mathbb{R}^d} g(y, \theta).$$

Theorem (Implicit Function Theorem)

If g sufficiently smooth (in y and θ) and strongly convex in y, then $\theta \mapsto x^*(\theta)$ is continuously differentiable with

$$\nabla x^*(\theta) = -[\partial_{yy} g(x^*(\theta), \theta)]^{-1} \partial_y \partial_\theta g(x^*(\theta), \theta) \in \mathbb{R}^{d \times n}$$

Hypergradient

Consider the simple bilevel problem:

$$\min_{\theta\in\mathbb{R}^n} \quad F(\theta) := f(x^*(\theta)), \qquad \text{s.t.} \quad x^*(\theta) := \arg\min_{y\in\mathbb{R}^d} g(y,\theta).$$

Theorem (Implicit Function Theorem)

If g sufficiently smooth (in y and θ) and strongly convex in y, then $\theta \mapsto x^*(\theta)$ is continuously differentiable with

$$\nabla x^*(\theta) = -[\partial_{yy} g(x^*(\theta),\theta)]^{-1} \partial_y \partial_\theta g(x^*(\theta),\theta) \in \mathbb{R}^{d\times n}$$

This gives us the exact hypergradient

$$\nabla F(\theta) = -[\partial_y \partial_\theta g(x^*(\theta),\theta)]^T [\partial_{yy} g(x^*(\theta),\theta)]^{-1} \nabla f(x^*(\theta))$$

Hypergradient Computation

The exact hypergradient is

$$\nabla F(\theta) = -[\partial_y \partial_\theta g(x^*(\theta), \theta)]^T [\partial_{yy} g(x^*(\theta), \theta)]^{-1} \nabla f(x^*(\theta))$$

Hypergradient Computation

The exact hypergradient is

$$\nabla F(\theta) = -[\partial_y \partial_\theta g(x^*(\theta), \theta)]^T [\partial_{yy} g(x^*(\theta), \theta)]^{-1} \nabla f(x^*(\theta))$$

- We can never evaluate $x^*(\theta)$ exactly (minimizer of g).
- If dimension of y is large, solve linear system inexactly ($\partial_{yy} g$ is SPD so use CG)

Hypergradient Computation

The exact hypergradient is

$$\nabla F(\theta) = -[\partial_y \partial_\theta g(x^*(\theta), \theta)]^T [\partial_{yy} g(x^*(\theta), \theta)]^{-1} \nabla f(x^*(\theta))$$

- We can never evaluate $x^*(\theta)$ exactly (minimizer of g).
- If dimension of y is large, solve linear system inexactly ($\partial_{yy} g$ is SPD so use CG)

Implicit Function Theorem (+ CG) approach:

1. Solve lower-level problem to get x_ε^* such that $\|x_\varepsilon^* - x^*(\theta)\| \leq \varepsilon$
2. Using CG, find $q_{\varepsilon,\delta}$ such that $\|[\partial_{yy} g(x_\varepsilon^*, \theta)] q_{\varepsilon,\delta} - \nabla f(x_\varepsilon^*)\| \leq \delta$.
3. Return hypergradient estimate $h_{\varepsilon,\delta} := -[\partial_y \partial_\theta g(x_\varepsilon^*, \theta)]^T q_{\varepsilon,\delta}$.

Hypergradient Computation

The exact hypergradient is

$$\nabla F(\theta) = -[\partial_y \partial_\theta g(x^*(\theta), \theta)]^T [\partial_{yy} g(x^*(\theta), \theta)]^{-1} \nabla f(x^*(\theta))$$

- We can never evaluate $x^*(\theta)$ exactly (minimizer of g).
- If dimension of y is large, solve linear system inexactly ($\partial_{yy} g$ is SPD so use CG)

Implicit Function Theorem (+ CG) approach:

1. Solve lower-level problem to get x_ε^* such that $\|x_\varepsilon^* - x^*(\theta)\| \le \varepsilon$
2. Using CG, find $q_{\varepsilon,\delta}$ such that $\|[\partial_{yy} g(x_\varepsilon^*, \theta)] q_{\varepsilon,\delta} - \nabla f(x_\varepsilon^*)\| \le \delta$.
3. Return hypergradient estimate $h_{\varepsilon,\delta} := -[\partial_y \partial_\theta g(x_\varepsilon^*, \theta)]^T q_{\varepsilon,\delta}$.

Theorem (Pedregosa (2016); Zucchet & Sacramento (2022))

If ε is sufficiently small, then $\|h_{\varepsilon,\delta} - \nabla F(\theta)\| = \mathcal{O}(\varepsilon + \delta)$.

Iterative AD

An alternative approach for calculating $\nabla F(\theta)$ is to use automatic differentiation (AD).

Iterative AD

An alternative approach for calculating $\nabla F(\theta)$ is to use automatic differentiation (AD).

Given some algorithm for approximating $x^*(\theta) := \arg\min_{y\in\mathbb{R}^d} g(y,\theta)$, we can apply AD to that algorithm to compute $\nabla F(\theta) = \nabla x^*(\theta)^T \nabla f(x^*(\theta))$. [Christianson (1994)]

Iterative AD

An alternative approach for calculating $\nabla F(\theta)$ is to use automatic differentiation (AD).

Given some algorithm for approximating $x^*(\theta) := \arg\min_{y\in\mathbb{R}^d} g(y,\theta)$, we can apply AD to that algorithm to compute $\nabla F(\theta) = \nabla x^*(\theta)^T \nabla f(x^*(\theta))$. [Christianson (1994)]

For example, run K iterations of gradient descent with fixed stepsize starting from $x^{(0)}$:

$$x^{(k+1)} = x^{(k)} - \alpha \partial_y g(x^{(k)}, \theta), \qquad k = 0, \ldots, K-1$$

Iterative AD

An alternative approach for calculating $\nabla F(\theta)$ is to use automatic differentiation (AD).

Given some algorithm for approximating $x^*(\theta) := \arg\min_{y\in\mathbb{R}^d} g(y,\theta)$, we can apply AD to that algorithm to compute $\nabla F(\theta) = \nabla x^*(\theta)^T \nabla f(x^*(\theta))$. [Christianson (1994)]

For example, run K iterations of gradient descent with fixed stepsize starting from $x^{(0)}$:

$$x^{(k+1)} = x^{(k)} - \alpha \partial_y g(x^{(k)}, \theta), \qquad k = 0, \ldots, K-1$$

Reverse mode AD on this iteration uses the chain rule to compute $\partial_\theta x^{(k)}$ recursively:

$$\partial_\theta x^{(k+1)} = \partial_\theta x^{(k)} - \alpha [\partial_{yy} g(x^{(k)}, \theta)] \partial_\theta x^{(k)} - \alpha \partial_\theta \partial_y g(x^{(k)}, \theta)$$

with $\partial_\theta x^{(0)} = 0$.

Iterative AD

An alternative approach for calculating $\nabla F(\theta)$ is to use automatic differentiation (AD).

Given some algorithm for approximating $x^*(\theta) := \arg\min_{y\in\mathbb{R}^d} g(y,\theta)$, we can apply AD to that algorithm to compute $\nabla F(\theta) = \nabla x^*(\theta)^T \nabla f(x^*(\theta))$. [Christianson (1994)]

For example, run K iterations of gradient descent with fixed stepsize starting from $x^{(0)}$:

$$x^{(k+1)} = x^{(k)} - \alpha \partial_y g(x^{(k)}, \theta), \qquad k = 0, \ldots, K-1$$

Reverse mode AD on this iteration uses the chain rule to compute $\partial_\theta x^{(k)}$ recursively:

$$\partial_\theta x^{(k+1)} = \partial_\theta x^{(k)} - \alpha[\partial_{yy} g(x^{(k)}, \theta)]\partial_\theta x^{(k)} - \alpha \partial_\theta \partial_y g(x^{(k)}, \theta)$$

with $\partial_\theta x^{(0)} = 0$.

With care, computing $[\partial_\theta x^{(k)}]^T v$ for any vector v (e.g. $\nabla f(x^{(K)}) \approx \nabla f(x^*(\theta))$) can be done with one extra loop (in the reverse direction, $k = K-1, \ldots, 0$).

Iterative AD

We are solving the lower-level problem with GD ($x^{(K)} \approx x^*(\theta)$):

$$x^{(k+1)} = x^{(k)} - \alpha \partial_y g(x^{(k)}, \theta), \qquad k = 0, \ldots, K-1,$$

Since we are solving a smooth, strongly convex problem, if α is small enough then $\|x^{(K)} - x^*(\theta)\| \leq \lambda^K \|x^{(0)} - x^*(\theta)\|$ for some $\lambda < 1$.

Iterative AD

We are solving the lower-level problem with GD ($x^{(K)} \approx x^*(\theta)$):

$$x^{(k+1)} = x^{(k)} - \alpha \partial_y g(x^{(k)}, \theta), \qquad k = 0, \ldots, K-1,$$

Since we are solving a smooth, strongly convex problem, if α is small enough then $\|x^{(K)} - x^*(\theta)\| \leq \lambda^K \|x^{(0)} - x^*(\theta)\|$ for some $\lambda < 1$.

The corresponding AD iteration returns $h^{(K)} \approx \nabla F(\theta)$ after iterating

$$h^{(k+1)} = h^{(k)} - \alpha [\partial_y \partial_\theta g(x^{(K-k-1)}, \theta)]^T \widetilde{x}^{(K-k)},$$
$$\widetilde{x}^{(K-k-1)} = \widetilde{x}^{(K-k)} - \alpha [\partial_{yy} g(x^{(K-k-1)}, \theta)] \widetilde{x}^{(K-k)}.$$

Iterative AD

We are solving the lower-level problem with GD ($x^{(K)} \approx x^*(\theta)$):

$$x^{(k+1)} = x^{(k)} - \alpha \partial_y g(x^{(k)}, \theta), \qquad k = 0, \ldots, K-1,$$

Since we are solving a smooth, strongly convex problem, if α is small enough then $\|x^{(K)} - x^*(\theta)\| \le \lambda^K \|x^{(0)} - x^*(\theta)\|$ for some $\lambda < 1$.

The corresponding AD iteration returns $h^{(K)} \approx \nabla F(\theta)$ after iterating

$$h^{(k+1)} = h^{(k)} - \alpha [\partial_y \partial_\theta g(x^{(K-k-1)}, \theta)]^T \widetilde{x}^{(K-k)},$$
$$\widetilde{x}^{(K-k-1)} = \widetilde{x}^{(K-k)} - \alpha [\partial_{yy} g(x^{(K-k-1)}, \theta)] \widetilde{x}^{(K-k)}.$$

Theorem (Mehmood & Ochs (2020))

The reverse mode AD hypergradient $h^{(K)}$ satisfies $\|h^{(K)} - \nabla F_K\| = \mathcal{O}(K\lambda^K)$, where

$$\nabla F_K := -[\partial_y \partial_\theta g(x^{(K)}, \theta)]^T [\partial_{yy} g(x^{(K)}, \theta)]^{-1} \nabla f(x^{(K)}).$$

Iterative AD

We can get a better iteration using inexact AD: evaluate all second derivatives at the best estimate $x^{(K)}$.

$$h^{(k+1)} = h^{(k)} - \alpha[\partial_y \partial_\theta g(x^{(K)}, \theta)]^T \widetilde{x}^{(K-k)},$$
$$\widetilde{x}^{(K-k-1)} = \widetilde{x}^{(K-k)} - \alpha[\partial_{yy} g(x^{(K)}, \theta)]\widetilde{x}^{(K-k)}.$$

Iterative AD

We can get a better iteration using inexact AD: evaluate all second derivatives at the best estimate $x^{(K)}$.

$$h^{(k+1)} = h^{(k)} - \alpha[\partial_y \partial_\theta g(x^{(K)}, \theta)]^T \widetilde{x}^{(K-k)},$$
$$\widetilde{x}^{(K-k-1)} = \widetilde{x}^{(K-k)} - \alpha[\partial_{yy} g(x^{(K)}, \theta)]\widetilde{x}^{(K-k)}.$$

Theorem (Mehmood & Ochs (2020))

The inexact AD hypergradient $h^{(K)}$ satisfies $\|h^{(K)} - \nabla F_K\| = \mathcal{O}(\lambda^K)$.

Iterative AD

We can get a better iteration using inexact AD: evaluate all second derivatives at the best estimate $x^{(K)}$.

$$h^{(k+1)} = h^{(k)} - \alpha[\partial_y \partial_\theta g(x^{(K)}, \theta)]^T \widetilde{x}^{(K-k)},$$
$$\widetilde{x}^{(K-k-1)} = \widetilde{x}^{(K-k)} - \alpha[\partial_{yy} g(x^{(K)}, \theta)]\widetilde{x}^{(K-k)}.$$

Theorem (Mehmood & Ochs (2020))

The inexact AD hypergradient $h^{(K)}$ satisfies $\|h^{(K)} - \nabla F_K\| = \mathcal{O}(\lambda^K)$.

Note: Similar results hold using heavy ball (Polyak) momentum instead of GD.

IFT vs. inexact AD

It turns out that the two hypergradient estimation procedures (IFT and inexact AD) are the same thing!

Theorem (Ehrhardt & LR (2024))

Inexact AD is exactly equivalent to using K iterations of GD with stepsize α to solve the symmetric positive definite linear system

$$[\partial_{yy} g(x^{(K)}, \theta)]q = \nabla f(x^{(K)}) \quad \iff \quad \min_q \frac{1}{2} q^T [\partial_{yy} g] q - \nabla f(x^{(K)})^T q,$$

starting from $q^{(0)} = 0$, and returning $-[\partial_y \partial_\theta g(x^{(K)}, \theta)]^T q^{(K)}$.

IFT vs. inexact AD

It turns out that the two hypergradient estimation procedures (IFT and inexact AD) are the same thing!

Theorem (Ehrhardt & LR (2024))

Inexact AD is exactly equivalent to using K iterations of GD with stepsize α to solve the symmetric positive definite linear system

$$[\partial_{yy} g(x^{(K)}, \theta)]q = \nabla f(x^{(K)}) \quad \Longleftrightarrow \quad \min_q \frac{1}{2} q^T [\partial_{yy} g] q - \nabla f(x^{(K)})^T q,$$

starting from $q^{(0)} = 0$, and returning $-[\partial_y \partial_\theta g(x^{(K)}, \theta)]^T q^{(K)}$.

So inexact AD is exactly an IFT method in disguise!

IFT vs. inexact AD

It turns out that the two hypergradient estimation procedures (IFT and inexact AD) are the same thing!

Theorem (Ehrhardt & LR (2024))

Inexact AD is exactly equivalent to using K iterations of GD with stepsize α to solve the symmetric positive definite linear system

$$[\partial_{yy} g(x^{(K)}, \theta)]q = \nabla f(x^{(K)}) \quad \iff \quad \min_q \frac{1}{2} q^T [\partial_{yy} g] q - \nabla f(x^{(K)})^T q,$$

starting from $q^{(0)} = 0$, and returning $-[\partial_y \partial_\theta g(x^{(K)}, \theta)]^T q^{(K)}$.

So inexact AD is exactly an IFT method in disguise!

An equivalent result holds for inexact AD using heavy ball momentum instead of GD.

Unified Framework

This motivates a general hypergradient approximation framework:

1. Solve the lower-level problem to get x_ε^* such that $\|x_\varepsilon^* - x^*\| \leq \varepsilon$
2. Find $q_{\varepsilon,\delta}$ such that $\|[\partial_{yy} g(x_\varepsilon^*, \theta)] q_{\varepsilon,\delta} - \nabla f(x_\varepsilon^*)\| \leq \delta$.
3. Return hypergradient estimate $h_{\varepsilon,\delta} := -[\partial_y \partial_\theta g(x_\varepsilon^*, \theta)]^T q_{\varepsilon,\delta}$.

This is IFT+CG, but any algorithm can be used in the first two steps, including inexact AD (and they don't have to be the same)

Unified Framework

This motivates a general hypergradient approximation framework:

1. Solve the lower-level problem to get x_ε^* such that $\|x_\varepsilon^* - x^*\| \leq \varepsilon$
2. Find $q_{\varepsilon,\delta}$ such that $\|[\partial_{yy} g(x_\varepsilon^*, \theta)] q_{\varepsilon,\delta} - \nabla f(x_\varepsilon^*)\| \leq \delta$.
3. Return hypergradient estimate $h_{\varepsilon,\delta} := -[\partial_y \partial_\theta g(x_\varepsilon^*, \theta)]^T q_{\varepsilon,\delta}$.

This is IFT+CG, but any algorithm can be used in the first two steps, including inexact AD (and they don't have to be the same)

Theorem (Ehrhardt & LR (2024))

We have $\|h_{\varepsilon,\delta} - \nabla F(\theta)\| = \mathcal{O}(\varepsilon + \delta + \varepsilon^2 + \delta\varepsilon)$. *Holds for any* $\varepsilon > 0$ *(new!).*

Unified Framework

This motivates a general hypergradient approximation framework:

1. Solve the lower-level problem to get x_ε^* such that $\|x_\varepsilon^* - x^*\| \leq \varepsilon$
2. Find $q_{\varepsilon,\delta}$ such that $\|[\partial_{yy} g(x_\varepsilon^*, \theta)] q_{\varepsilon,\delta} - \nabla f(x_\varepsilon^*)\| \leq \delta$.
3. Return hypergradient estimate $h_{\varepsilon,\delta} := -[\partial_y \partial_\theta g(x_\varepsilon^*, \theta)]^T q_{\varepsilon,\delta}$.

This is IFT+CG, but any algorithm can be used in the first two steps, including inexact AD (and they don't have to be the same)

Theorem (Ehrhardt & LR (2024))

We have $\|h_{\varepsilon,\delta} - \nabla F(\theta)\| = \mathcal{O}(\varepsilon + \delta + \varepsilon^2 + \delta\varepsilon)$. *Holds for any* $\varepsilon > 0$ *(new!).*

Important improvement: the constants in the error bound are computable.

Outline

1. Simple example: image denoising
2. Bilevel learning
3. Calculating hypergradients
4. **Dynamic linesearch**
5. Inexact SGD

Bilevel Learning

Difficulty?

Bilevel learning is just a smooth nonconvex problem — where is the challenge?

Bilevel Learning

Difficulty?

Bilevel learning is just a smooth nonconvex problem — where is the challenge?

- Can't evaluate lower-level minimizers $\hat{x}_i(\theta)$ exactly, so can never get exact $F(\theta)$ or $\nabla F(\theta)$ [Kunisch & Pock, 2013; Sherry et al., 2020]
- But can evaluate F and ∇F to arbitrary accuracy (with significant computational cost) [Berahas et al., 2021; Cao et al., 2024]
- Potentially large scale in upper-level problem
  - Many ML people looking at SGD-type methods at both levels simultaneously
    e.g. [Grazzi et al., 2021; Ji et al., 2021; Kwon et al., 2023]

Difficulty?

Bilevel learning is just a smooth nonconvex problem — where is the challenge?

- Can't evaluate lower-level minimizers $\hat{x}_i(\theta)$ exactly, so can never get exact $F(\theta)$ or $\nabla F(\theta)$ [Kunisch & Pock, 2013; Sherry et al., 2020]
- But can evaluate F and ∇F to arbitrary accuracy (with significant computational cost) [Berahas et al., 2021; Cao et al., 2024]
- Potentially large scale in upper-level problem
  - Many ML people looking at SGD-type methods at both levels simultaneously e.g. [Grazzi et al., 2021; Ji et al., 2021; Kwon et al., 2023]

Key question 2: how to choose a good evaluation accuracy to get (i) guaranteed convergence, (ii) without requiring hyperparameter tuning, (iii) at a reasonable computational cost?

Algorithm for Bilevel Learning

We aim to solve the bilevel learning problem

$$\begin{aligned}
\min_{\theta} \quad & F(\theta) := \frac{1}{n}\sum_{i=1}^{n} \|\hat{x}_i(\theta) - x_i\|^2 + \mathcal{J}(\theta), \\
\text{s.t.} \quad & \hat{x}_i(\theta) := \arg\min_{x} g_i(x,\theta), \quad \forall i = 1, \ldots, n.
\end{aligned}$$

Algorithm for Bilevel Learning

We aim to solve the bilevel learning problem

$$\min_{\theta} \quad F(\theta) := \frac{1}{n}\sum_{i=1}^{n} \|\hat{x}_i(\theta) - x_i\|^2 + \mathcal{J}(\theta),$$
$$\text{s.t.} \quad \hat{x}_i(\theta) := \arg\min_{x} g_i(x,\theta), \quad \forall i = 1,\ldots,n.$$

With our inexact hypergradient computation and taking $\mathcal{J} = 0$, this looks like a single-level problem of the form

$$\min_{\theta} F(\theta) := f(\hat{x}(\theta))$$

where $F(\theta)$ and $\nabla F(\theta)$ can never be computed exactly, but can be computed to arbitrary accuracy (with higher computational costs for higher accuracy).

Inexact Linesearch

A simple algorithm that requires no hyperparameter tuning is gradient descent with linesearch:

$$\theta_{k+1} = \theta_k - \alpha_k \nabla F(\theta_k),$$

with $\alpha_k > 0$ chosen to be ensure that $F(\theta_{k+1}) \leq F(\theta_k) - \alpha_k \|\nabla F(\theta_k)\|^2$ (and α_k not too small).

Inexact Linesearch

A simple algorithm that requires no hyperparameter tuning is gradient descent with linesearch:

$$\theta_{k+1} = \theta_k - \alpha_k \nabla F(\theta_k),$$

with $\alpha_k > 0$ chosen to be ensure that $F(\theta_{k+1}) \leq F(\theta_k) - \alpha_k \|\nabla F(\theta_k)\|^2$ (and α_k not too small).

To handle inexactness, there are two key issues to resolve:

- Given $z_k \approx \nabla F(\theta_k)$ can we ensure $-z_k$ is a descent direction ($-z_k^T \nabla F(\theta_k) < 0$)?
- If no sufficient decrease (with inexact $F(\theta)$ evaluations), should we shrink stepsize or improve accuracy in F (or ∇F)?

Inexact Linesearch

A simple algorithm that requires no hyperparameter tuning is gradient descent with linesearch:

$$\theta_{k+1} = \theta_k - \alpha_k \nabla F(\theta_k),$$

with $\alpha_k > 0$ chosen to be ensure that $F(\theta_{k+1}) \leq F(\theta_k) - \alpha_k \|\nabla F(\theta_k)\|^2$ (and α_k not too small).

To handle inexactness, there are two key issues to resolve:

- Given $z_k \approx \nabla F(\theta_k)$ can we ensure $-z_k$ is a descent direction ($-z_k^T \nabla F(\theta_k) < 0$)?
- If no sufficient decrease (with inexact $F(\theta)$ evaluations), should we shrink stepsize or improve accuracy in F (or ∇F)?

To be practical, we don't want to make accuracy in F or ∇F unnecessarily high (but don't want to lose convergence guarantees either).

Inexact Gradient Calculation

- Given ϵ and δ, calculate inexact lower-level minimiser $x_\epsilon \approx \hat{x}(\theta)$ and inexact gradient $z_k \approx \nabla F(\theta_k)$ (using CG with residual tolerance δ)
- Calculate computable upper bound ω for $\|z_k - \nabla F(\theta_k)\|$
- If $\omega \leq (1-\eta)\|z_k\|$, then use $-z_k$ (guaranteed descent direction)
- Otherwise, decrease ϵ and δ by a constant factor and start again

Inexact Gradient

Inexact Gradient Calculation

- Given ϵ and δ, calculate inexact lower-level minimiser $x_\epsilon \approx \hat{x}(\theta)$ and inexact gradient $z_k \approx \nabla F(\theta_k)$ (using CG with residual tolerance δ)
- Calculate computable upper bound ω for $\|z_k - \nabla F(\theta_k)\|$
- If $\omega \leq (1-\eta)\|z_k\|$, then use $-z_k$ (guaranteed descent direction)
- Otherwise, decrease ϵ and δ by a constant factor and start again

Theorem (Salehi et al., 2025)

If $\|\nabla F(\theta_k)\| \neq 0$, then $-z_k$ is a descent direction for all sufficiently small ϵ and δ.

i.e. Gradient calculation terminates in finite time.

Sufficient Decrease Condition

Inexact sufficient decrease condition

- Given $\hat{\theta} = \theta_k - \alpha_k z_k$, compute $x_\epsilon(\theta_k) \approx \hat{x}(\theta_k)$ and $x_\epsilon(\hat{\theta}) \approx \hat{x}(\hat{\theta})$ with accuracy ϵ
- Compute approximate objective values $\tilde{F}(\theta_k)$ and $\tilde{F}(\hat{\theta})$
- Inexact sufficient decrease condition is (e.g. for L-smooth and convex f):

$$\tilde{F}(\hat{\theta}) \leq \tilde{F}(\theta_k) - c\alpha_k \|z_k\|^2 - \|\nabla f(x_\epsilon(\hat{\theta}))\|\epsilon - \|\nabla f(x_\epsilon(\theta_k))\|\epsilon - \frac{1}{2}L\epsilon^2$$

Sufficient Decrease Condition

Inexact sufficient decrease condition

- Given $\hat{\theta} = \theta_k - \alpha_k z_k$, compute $x_\epsilon(\theta_k) \approx \hat{x}(\theta_k)$ and $x_\epsilon(\hat{\theta}) \approx \hat{x}(\hat{\theta})$ with accuracy ϵ
- Compute approximate objective values $\tilde{F}(\theta_k)$ and $\tilde{F}(\hat{\theta})$
- Inexact sufficient decrease condition is (e.g. for L-smooth and convex f):

$$\tilde{F}(\hat{\theta}) \leq \tilde{F}(\theta_k) - c\alpha_k \|z_k\|^2 - \|\nabla f(x_\epsilon(\hat{\theta}))\|\epsilon - \|\nabla f(x_\epsilon(\theta_k))\|\epsilon - \frac{1}{2}L\epsilon^2$$

Theorem (Salehi et al., 2025)

- *If inexact sufficient decrease condition holds, then* $F(\hat{\theta}) \leq F(\theta_k) - c\alpha_k \|z_k\|^2$.

Sufficient Decrease Condition

Inexact sufficient decrease condition

- Given $\hat{\theta} = \theta_k - \alpha_k z_k$, compute $x_\epsilon(\theta_k) \approx \hat{x}(\theta_k)$ and $x_\epsilon(\hat{\theta}) \approx \hat{x}(\hat{\theta})$ with accuracy ϵ
- Compute approximate objective values $\tilde{F}(\theta_k)$ and $\tilde{F}(\hat{\theta})$
- Inexact sufficient decrease condition is (e.g. for L-smooth and convex f):

$$\tilde{F}(\hat{\theta}) \le \tilde{F}(\theta_k) - c\alpha_k \|z_k\|^2 - \|\nabla f(x_\epsilon(\hat{\theta}))\|\epsilon - \|\nabla f(x_\epsilon(\theta_k))\|\epsilon - \frac{1}{2}L\epsilon^2$$

Theorem (Salehi et al., 2025)

- *If inexact sufficient decrease condition holds, then* $F(\hat{\theta}) \le F(\theta_k) - c\alpha_k\|z_k\|^2$.
- *For any* ϵ*, inexact sufficient decrease condition holds for all* $\alpha_k \in [\alpha_{\min}(\epsilon), \alpha_{\max}(\epsilon)]$

Sufficient Decrease Condition

Inexact sufficient decrease condition

- Given $\hat{\theta} = \theta_k - \alpha_k z_k$, compute $x_\epsilon(\theta_k) \approx \hat{x}(\theta_k)$ and $x_\epsilon(\hat{\theta}) \approx \hat{x}(\hat{\theta})$ with accuracy ϵ
- Compute approximate objective values $\tilde{F}(\theta_k)$ and $\tilde{F}(\hat{\theta})$
- Inexact sufficient decrease condition is (e.g. for L-smooth and convex f):

$$\tilde{F}(\hat{\theta}) \leq \tilde{F}(\theta_k) - c\alpha_k \|z_k\|^2 - \|\nabla f(x_\epsilon(\hat{\theta}))\|\epsilon - \|\nabla f(x_\epsilon(\theta_k))\|\epsilon - \frac{1}{2}L\epsilon^2$$

Theorem (Salehi et al., 2025)

- *If inexact sufficient decrease condition holds, then* $F(\hat{\theta}) \leq F(\theta_k) - c\alpha_k \|z_k\|^2$.
- *For any* ϵ, *inexact sufficient decrease condition holds for all* $\alpha_k \in [\alpha_{\min}(\epsilon), \alpha_{\max}(\epsilon)]$
- *As* $\epsilon \to 0$, *we have* $[\alpha_{\min}(\epsilon), \alpha_{\max}(\epsilon)] \to [0, \alpha_{\max}]$ *for some* $\alpha_{\max} > 0$

Inexact Backtracking

Method of Adaptive Inexact Descent (MAID) (single iteration k)

1: **for** $J_{\text{max}} = J_0, J_0+1, J_0+2, \ldots$ **do**
2: Compute inexact gradient z_k (possibly reducing ϵ and δ)
3: **for** $j = 0, \ldots, J_{\text{max}} - 1$ **do**
4: If sufficient decrease with stepsize $\alpha_k = \alpha\rho^j$, go to line 8
5: **end for**
6: Reduce ϵ and δ by constant factor *(backtracking failed, need higher accuracy)*
7: **end for**
8: Set $\theta_{k+1} = \theta_k - \alpha_k z_k$ *(successful linesearch)*
9: Increase ϵ and δ by constant factor for next iteration

Inexact Backtracking

Method of Adaptive Inexact Descent (MAID) (single iteration k)

```
1: for J_max = J_0, J_0 + 1, J_0 + 2, ... do
2:     Compute inexact gradient z_k (possibly reducing ε and δ)
3:     for j = 0, ..., J_max − 1 do
4:         If sufficient decrease with stepsize α_k = αρ^j, go to line 8
5:     end for
6:     Reduce ε and δ by constant factor (backtracking failed, need higher accuracy)
7: end for
8: Set θ_{k+1} = θ_k − α_k z_k  (successful linesearch)
9: Increase ε and δ by constant factor for next iteration
```

Theorem (Salehi et al., 2025)

At each iteration k, successful linesearch occurs in finite time. Hence $\|\nabla F(\theta_k)\| \to 0$.

Quadratic Problem

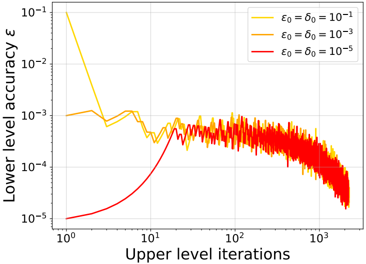
Simple linear least-squares problem (closed form for true solution):

$$\min_{\theta} \; f(\theta) := \|A_1 \hat{x}(\theta) - b_1\|^2 \qquad \text{s.t. } \hat{x}(\theta) = \arg\min_{x} g(x, \theta) := \|A_2 x + A_3 \theta - b_2\|^2$$

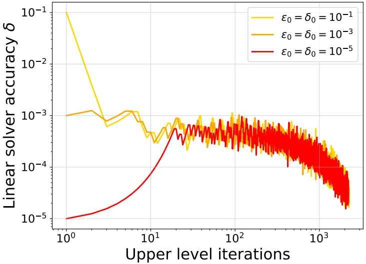
Simple linear least-squares problem (closed form for true solution):

$$\min_{\theta} \; f(\theta) := \|A_1 \hat{x}(\theta) - b_1\|^2 \qquad \text{s.t.} \; \hat{x}(\theta) = \arg\min_{x} g(x, \theta) := \|A_2 x + A_3 \theta - b_2\|^2$$

Do hyperparameters (initial accuracies ϵ and δ) matter?

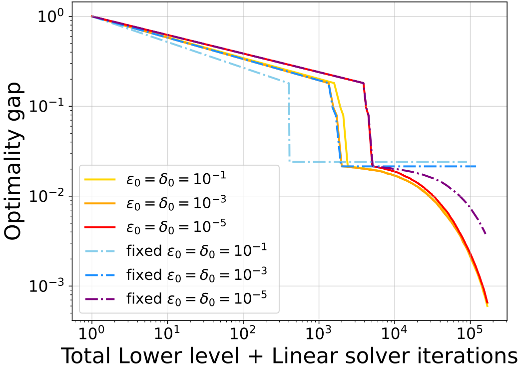


Final ϵ at each iteration



Final δ at each iteration

Dynamic accuracy is better than fixed accuracy



Optimality gap vs. computational work (lower-level + CG iterations)

Field of Experts Image Denoising

$$\min_{\theta} \ f(\theta) := \frac{1}{N}\sum_{i=1}^{N} \|\hat{x}_i(\theta) - x_i^*\|^2,$$

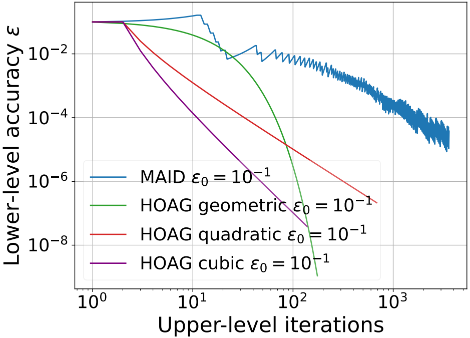
$$\text{s.t.} \ \ \hat{x}_i(\theta) = \arg\min_{x} g_i(x,\theta) := \frac{1}{2}\|x - y_i\|^2 + \sum_{k=1}^{K} \beta_k(\theta)\|c_k(\theta) * x\|_{k,\theta} + \frac{\mu}{2}\|x\|^2.$$

Learn $K = 30$ filters $c_k(\theta)$, smoothed ℓ_1-norms $\|\cdot\|_{k,\theta}$ and weights $\beta_k(\theta)$ to reconstruct noisy 2D images (≈ 1500 hyperparameters θ).

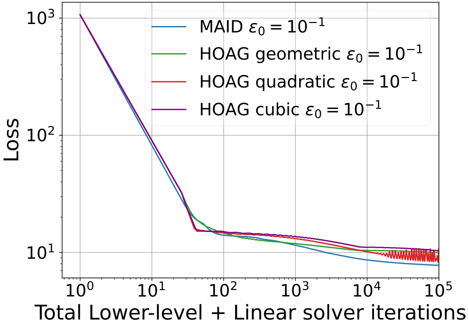
Using $N = 25$ training images (x_i^*, y_i) of size 96×96 pixels.

Field of Experts Denoising

Compare MAID against tuned HOAG (fixed accuracy schedule) [Pedregosa, 2016]



Accuracy ϵ at each iteration



Loss vs. computational work

Field of Experts Denoising

Apply learned filters on new test image

True image

Noisy (PSNR 20.3dB)

MAID (PSNR 29.7dB)

HOAG best (PSNR 28.8dB)

(Palladian Bridge, Bath, UK)

Outline

1. Simple example: image denoising
2. Bilevel learning
3. Calculating hypergradients
4. Dynamic linesearch
5. **Inexact SGD**

Inexact SGD

$$\min_{\theta} \quad F(\theta) := \frac{1}{n}\sum_{i=1}^{n} \|\hat{x}_i(\theta) - x_i\|^2,$$
$$\text{s.t.} \quad \hat{x}_i(\theta) := \arg\min_{x} g_i(x,\theta), \quad \forall i = 1,\ldots,n.$$

So far, we have assumed that n (number of examples) is small enough that we can compute the full (inexact) hypergradient at every iteration. But what if n is large?

Inexact SGD

$$\min_{\theta} \quad F(\theta) := \frac{1}{n}\sum_{i=1}^{n} \|\hat{x}_i(\theta) - x_i\|^2,$$
$$\text{s.t.} \quad \hat{x}_i(\theta) := \arg\min_{x} g_i(x, \theta), \quad \forall i = 1, \ldots, n.$$

So far, we have assumed that n (number of examples) is small enough that we can compute the full (inexact) hypergradient at every iteration. But what if n is large?

This commonly arises in ML, and the solution is to randomly subsample the data at every iteration (stochastic gradient descent).

Inexact SGD

$$\min_{\theta} \quad F(\theta) := \frac{1}{n}\sum_{i=1}^{n} \|\hat{x}_i(\theta) - x_i\|^2,$$

$$\text{s.t.} \quad \hat{x}_i(\theta) := \arg\min_{x} g_i(x, \theta), \quad \forall i = 1, \ldots, n.$$

So far, we have assumed that n (number of examples) is small enough that we can compute the full (inexact) hypergradient at every iteration. But what if n is large?

This commonly arises in ML, and the solution is to randomly subsample the data at every iteration (stochastic gradient descent). Defining random weights $(w_1, \ldots, w_n)$, we get

$$\min_{\theta} \quad F(\theta) := \mathbb{E}_w[F_w(\theta)], \qquad \text{where} \qquad F_w(\theta) := \sum_{i=1}^{n} w_i \|\hat{x}_i(\theta) - x_i\|^2$$

(e.g. $w_i = 1/n_{\text{sample}}$ if example i is sampled, else $w_i = 0$)

Inexact SGD

Since we can only approximate $\nabla F_w(\theta)$ to arbitrary accuracy, we get an inexact SGD iteration:

$$\theta_{k+1} = \theta_k - \alpha z_{w_k}(\theta_k),$$

where $z_w(\theta) \approx \nabla F_w(\theta)$ to some desired accuracy, $\|z_w(\theta) - \nabla F_w(\theta)\| \leq \mathcal{O}(\epsilon)$.

This is a form of SGD with biased stochastic gradients.

Inexact SGD

Since we can only approximate $\nabla F_w(\theta)$ to arbitrary accuracy, we get an inexact SGD iteration:

$$\theta_{k+1} = \theta_k - \alpha z_{w_k}(\theta_k),$$

where $z_w(\theta) \approx \nabla F_w(\theta)$ to some desired accuracy, $\|z_w(\theta) - \nabla F_w(\theta)\| \leq \mathcal{O}(\epsilon)$.

This is a form of SGD with biased stochastic gradients.

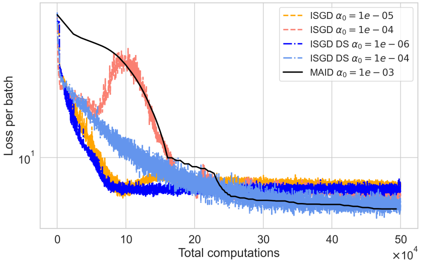
Existing convergence theory for biased SGD gives us convergence to a neighborhood of a solution, provided the stepsize is small enough (requires tuning!). [Demidovich et al., 2023]

Theorem (Salehi et al., 2025)

If all F_w are smooth with Lipschitz continuous gradients and bounded below, and $\alpha = \mathcal{O}(\epsilon^2)$, then $\mathbb{E}[\|\nabla F(\theta_k)\|^2] \leq \mathcal{O}(\epsilon^2)$ after at most $\mathcal{O}(\epsilon^{-4})$ iterations.

Example Results

Applying MAID and ISGD to a field of experts denoising problem with $n = 1024$ training images, we get:



Loss vs. computational effort

Beneficial to do subsampling in the large data regime, but requires hyperparameter tuning.

Conclusions & Future Work

Conclusions

- Bilevel learning provides a structured hyperparameter tuning method
- New link between AD and implicit function theorem hypergradient estimation
- New linesearch method balances accuracy and computational efficiency
- Speed up performance on large datasets with inexact SGD

Future Work

- Theory for inexact SGD with decreasing stepsizes (fixed accuracy)
- Inexact SGD with flexible/dynamic stepsize and accuracy regimes

References i

A. S. Berahas, L. Cao, and K. Scheinberg, *Global convergence rate analysis of a generic line search algorithm with noise*, SIAM Journal on Optimization, 31 (2021), pp. 1489–1518.

L. Cao, A. S. Berahas, and K. Scheinberg, *First- and second-order high probability complexity bounds for trust-region methods with noisy oracles*, Mathematical Programming, 207 (2024), pp. 55–106.

A. Chambolle and T. Pock, *An introduction to continuous optimization for imaging*, Acta Numerica, 25 (2016), pp. 161–319.

B. Christianson, *Reverse accumulation and attractive fixed points*, Optimization Methods and Software, 3 (1994), pp. 311–326.

Y. Demidovich, G. Malinovsky, I. Sokolov, and P. Richtárik, *A guide through the zoo of biased SGD*, in 37th Conference on Neural Information Processing Systems (NeurIPS 2023), New Orleans, USA, 2023.

M. J. Ehrhardt and L. Roberts, *Analyzing inexact hypergradients for bilevel learning*, IMA Journal of Applied Mathematics, 89 (2024), pp. 254–278.

References ii

R. Grazzi, M. Pontil, and S. Salzo, *Convergence properties of stochastic hypergradients*, in Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, vol. 130, 2021, pp. 3826–3834.

K. Ji, J. Yang, and Y. Liang, *Bilevel optimization for machine learning: Algorithm design and convergence analysis*, in Proceedings of the 38th International Conference on Machine Learning, 2021, pp. 4882–4892.

K. Kunisch and T. Pock, *A Bilevel Optimization Approach for Parameter Learning in Variational Models*, SIAM Journal on Imaging Sciences, 6 (2013), pp. 938–983.

J. Kwon, D. Kwon, S. Wright, and R. Nowak, *A fully first-order method for stochastic bilevel optimization*, 40th International Conference on Machine Learning, ICML 2023, (2023).

S. Mehmood and P. Ochs, *Automatic differentiation of some first-order methods in parametric optimization*, in Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS), Palermo, Italy, 2020.

F. Pedregosa, *Hyperparameter optimization with approximate gradient*, in Proceedings of the 33rd International Conference on Machine Learning, New York, 2016.

References iii

M. S. Salehi, S. Mukherjee, L. Roberts, and M. J. Ehrhardt, *An adaptively inexact first-order method for bilevel optimization with application to hyperparameter learning*, SIAM Journal on Mathematics of Data Science, 7 (2025), pp. 906–936.

———, *Bilevel learning with inexact stochastic gradients*, in Scale Space and Variational Methods in Computer Vision, T. A. Bubba, R. Gaburro, S. Gazzola, K. Papafitsoros, M. Pereyra, and C.-B. Schönlieb, eds., vol. 15667, Springer Nature Switzerland, Cham, 2025, pp. 347–359.

F. Sherry, M. Benning, J. C. De los Reyes, M. J. Graves, G. Maierhofer, G. Williams, C.-B. Schonlieb, and M. J. Ehrhardt, *Learning the sampling pattern for MRI*, IEEE Transactions on Medical Imaging, 39 (2020), pp. 4310–4321.

N. Zucchet and J. Sacramento, *Beyond backpropagation: Bilevel optimization through implicit differentiation and equilibrium propagation*, Neural Computation, 34 (2022), pp. 2309–2346.