On the Iteration Complexity of Hypergradient Computation

Riccardo Grazzi

Computational Statistics and Machine Learning, Istituto Italiano di Tecnologia.
Department of Computer Science, University College London.

riccardo.grazzi@iit.it

Joint work with Luca Franceschi, Massimiliano Pontil and Saverio Salzo.

Bilevel Optimization Problem

$$\min_{\lambda \in \Lambda \subseteq \mathbb{R}^n} f(\lambda) := E(w(\lambda), \lambda) \quad \textbf{(upper-level)}$$
$$w(\lambda) := \Phi(w(\lambda), \lambda) \quad \textbf{(lower-level)}$$

- Hyperparameter optimization, meta-learning.
- Graph and recurrent neural networks.

How can we solve this optimization problem?

- Black-box methods (random/grid search, Bayesian optimization, ...).

Bilevel Optimization Problem

$$\min_{\lambda \in \Lambda \subseteq \mathbb{R}^n} f(\lambda) := E(w(\lambda), \lambda) \quad \textbf{(upper-level)}$$
$$w(\lambda) := \Phi(w(\lambda), \lambda) \quad \textbf{(lower-level)}$$

- Hyperparameter optimization, meta-learning.
- Graph and recurrent neural networks.

How can we solve this optimization problem?

- Black-box methods (random/grid search, Bayesian optimization, ...).
- Gradient-based methods exploiting the *hypergradient* $\nabla f(\lambda)$.

Computing the Hypergradient $\nabla f(\lambda)$

Can be really expensive or even impossible to compute Exactly.

Two common approximation strategies are

1. Iterative Differentiation (ITD).
2. Approximate Implicit Differentiation (AID).

Computing the Hypergradient $\nabla f(\lambda)$

Can be really expensive or even impossible to compute Exactly.

Two common approximation strategies are

1. **Iterative Differentiation (ITD)**.
2. **Approximate Implicit Differentiation (AID)**.

Which one is the best?

- Previous works provide mostly qualitative and empirical results.

Can we have quantitative results on the approximation error?

- Yes! If the fixed point map $\Phi(\cdot, \lambda)$ is a **contraction**.

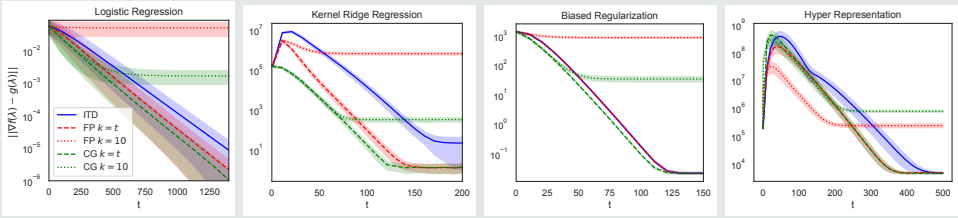
Our Contributions

Upper bounds on the approximation error for both ITD and AID

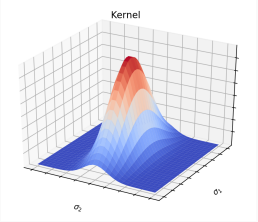
- Both methods achieve **non-asymptotic linear convergence rates**.
- We prove that ITD is generally worse than AID in terms of upper bounds.

Extensive experimental comparison among different AID strategies and ITD

- If $\Phi(\cdot, \lambda)$ is a contraction, the results confirm the theory.
- If $\Phi(\cdot, \lambda)$ is *NOT* a contraction, ITD can be still a reliable strategy.



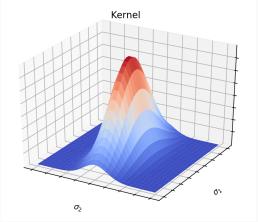
Motivation

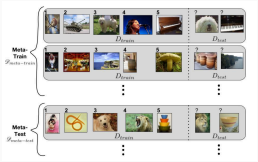


Source: S.Ravi, H. Larochelle (2016).

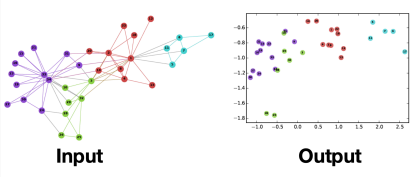
- Hyperparameter optimization (learn the kernel/regularization, ...).
- Meta-learning (MAML, L2LOpt, ...).

Motivation





Source: S.Ravi, H. Larochelle (2016).

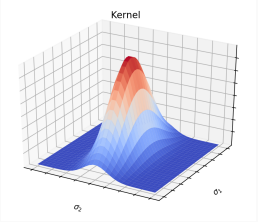


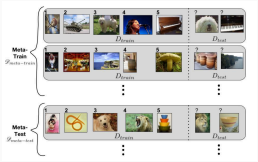
Source: snap.stanford.edu/proj/embeddings-www

- Hyperparameter optimization (learn the kernel/regularization, ...).
- Meta-learning (MAML, L2LOpt, ...).

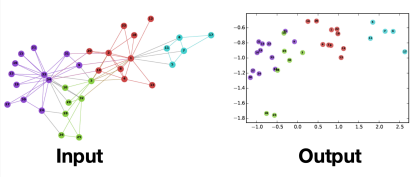
- Graph Neural Networks.
- Some Recurrent Models.
- Deep Equilibrium Models.

Motivation





Source: S.Ravi, H. Larochelle (2016).



Source: snap.stanford.edu/proj/embeddings-www

- Hyperparameter optimization (learn the kernel/regularization, ...).
- Meta-learning (MAML, L2LOpt, ...).

- Graph Neural Networks.
- Some Recurrent Models.
- Deep Equilibrium Models.

All can be cast into the same bilevel framework where at the lower-level we seek for the solution to a parametric fixed point equation.

Example: Optimizing the Regularization Hyperparameter in Ridge Regression

$$\min_{\lambda \in (0,\infty)} \frac{1}{2}\|X_{\text{val}} w(\lambda) - y_{\text{val}}\|_2^2$$

$$w(\lambda) = \arg\min_{w \in \mathbb{R}^d} \left\{ \ell(w, \lambda) := \frac{1}{2}\|Xw - y\|_2^2 + \frac{\lambda}{2}\|w\|_2^2 \right\}$$

$w(\lambda)$ is the **unique fixed point** of the *one step GD* map

$$\Phi(w, \lambda) = w - \alpha \nabla_1 \ell(w, \lambda)$$

If the step size α is sufficiently small, $\Phi(\cdot, \lambda)$ is also a **contraction**.

The Bilevel Framework

$$\min_{\lambda \in \Lambda \subseteq \mathbb{R}^n} f(\lambda) := E(w(\lambda), \lambda) \quad \textbf{(upper-level)}$$

$$w(\lambda) := \Phi(w(\lambda), \lambda) \quad \textbf{(lower-level)}$$

- $w(\lambda) \in \mathbb{R}^d$ is often not available in closed form.
- f is usually non convex and **expensive** or **impossible** to evaluate exactly.

The Bilevel Framework

$$\min_{\lambda \in \Lambda \subseteq \mathbb{R}^n} f(\lambda) := E(w(\lambda), \lambda) \quad \textbf{(upper-level)}$$

$$w(\lambda) := \Phi(w(\lambda), \lambda) \quad \textbf{(lower-level)}$$

- $w(\lambda) \in \mathbb{R}^d$ is often not available in closed form.
- f is usually non convex and **expensive** or **impossible** to evaluate exactly.
- ∇f is even harder to evaluate.

How to Compute the Hypergradient $\nabla f(\lambda)$?

Iterative Differentiaton (ITD)

1. Set $w_0(\lambda) = 0$ and compute,

$$\begin{array}{l} \text{for } i = 1, 2, \dots t \\ \quad w_i(\lambda) = \Phi(w_{i-1}(\lambda), \lambda). \end{array}$$

2. Compute $f_t(\lambda) = E(w_t(\lambda), \lambda)$.
3. Compute $\nabla f_t(\lambda)$ *efficiently* using reverse (RMAD) or forward (FMAD) mode automatic differentiation.

How to Compute the Hypergradient $\nabla f(\lambda)$?

Iterative Differentiaton (ITD)

1. Set $w_0(\lambda) = 0$ and compute,

$$\begin{aligned}&\text{for } i = 1, 2, \dots t\\ &\quad w_i(\lambda) = \Phi(w_{i-1}(\lambda), \lambda).\end{aligned}$$

2. Compute $f_t(\lambda) = E(w_t(\lambda), \lambda)$.
3. Compute $\nabla f_t(\lambda)$ *efficiently* using reverse (RMAD) or forward (FMAD) mode automatic differentiation.

Approximate Implicit Differentiation (AID)

1. Get $w_t(\lambda)$ with t **steps** of a lower-level solver.
2. Compute $v_{t,k}(\lambda)$ with k **steps** of a solver for the *linear system*

$$(I - \partial_1 \Phi(w_t(\lambda), \lambda)^\top) v = \nabla_1 E(w_t(\lambda), \lambda).$$

3. Compute the approximate gradient as

$$\begin{aligned}\hat{\nabla} f(\lambda) :=& \nabla_2 E(w_t(\lambda), \lambda)\\ &+ \partial_2 \Phi(w_t(\lambda), \lambda)^\top v_{t,k}(\lambda).\end{aligned}$$

How to Compute the Hypergradient $\nabla f(\lambda)$?

Iterative Differentiaton (ITD)

1. Set $w_0(\lambda) = 0$ and compute,

$$\begin{aligned}&\text{for } i = 1, 2, \dots t\\ &\quad w_i(\lambda) = \Phi(w_{i-1}(\lambda), \lambda).\end{aligned}$$

2. Compute $f_t(\lambda) = E(w_t(\lambda), \lambda)$.
3. Compute $\nabla f_t(\lambda)$ *efficiently* using reverse (RMAD) or forward (FMAD) mode automatic differentiation.

Approximate Implicit Differentiation (AID)

1. Get $w_t(\lambda)$ with t **steps** of a lower-level solver.
2. Compute $v_{t,k}(\lambda)$ with k **steps** of a solver for the *linear system*

$$(I - \partial_1 \Phi(w_t(\lambda), \lambda)^\top) v = \nabla_1 E(w_t(\lambda), \lambda).$$

3. Compute the approximate gradient as

$$\begin{aligned}\hat{\nabla} f(\lambda) :=& \nabla_2 E(w_t(\lambda), \lambda)\\ &+ \partial_2 \Phi(w_t(\lambda), \lambda)^\top v_{t,k}(\lambda).\end{aligned}$$

Which one is the best?

A First Comparison

ITD

- Ignores the bilevel structure.
- Cost in time (RMAD): $O(\mathrm{Cost}(f_t(\lambda)))$
- Cost in memory (RMAD): $O(td)$.
- **Can we control $\|\nabla f_t(\lambda) - \nabla f(\lambda)\|$?**

AID

- Can use any lower-level solver.
- Cost in time ($k = t$): $O(\mathrm{Cost}(f_t(\lambda)))$.
- Cost in memory: $O(d)$.
- **Can we control $\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\|$?**

$f_t(\lambda) = E(w_t(\lambda), \lambda)$.

Previous Work on the Approximation Error

ITD

- $\arg\min f_t \xrightarrow[t\to\infty]{} \arg\min f$ (Franceschi et al., 2018).

AID

- $\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\| \xrightarrow[t,k\to\infty]{} 0$ (Pedregosa, 2016).
- $\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\| \xrightarrow[t,k\to\infty]{} 0$ at a **linear rate** in t and k for meta-learning with biased regularization (Rajeswaran et al., 2019).

$f_t(\lambda) = E(w_t(\lambda), \lambda)$.

Previous Work on the Approximation Error

ITD

- $\arg\min f_t \xrightarrow[t\to\infty]{} \arg\min f$ (Franceschi et al., 2018).
- We provide non-asymptotic upper bounds on $\|\nabla f_t(\lambda) - \nabla f(\lambda)\|$.

AID

- $\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\| \xrightarrow[t,k\to\infty]{} 0$ (Pedregosa, 2016).
- $\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\| \xrightarrow[t,k\to\infty]{} 0$ at a **linear rate** in t and k for meta-learning with biased regularization (Rajeswaran et al., 2019).
- We provide non-asymptotic upper bounds on $\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\|$.

$f_t(\lambda) = E(w_t(\lambda), \lambda)$.

Assumptions

- $\Phi(\cdot, \lambda)$ is a **contraction** with constant $q_\lambda < 1$.
- $\partial_1\Phi(\cdot, \lambda)$, $\partial_2\Phi(\cdot, \lambda)$, $\nabla_1 E(\cdot, \lambda)$ and $\nabla_2 E(\cdot, \lambda)$ are Lipschitz continuous

$\implies$ f **differentiable** and

$$w'(\lambda) := (I - \partial_1\Phi(w(\lambda), \lambda))^{-1}\partial_2\Phi(w(\lambda), \lambda)$$
$$\nabla f(\lambda) = \nabla_2 E(w(\lambda), \lambda) + w'(\lambda)^\top \nabla_1 E(w(\lambda), \lambda).$$

Main Contribution

Theorem *(ITD error upper bound)*

$$\|\nabla f_t(\lambda) - \nabla f(\lambda)\| \leq \Big(c_1(\lambda) + \frac{c_2(\lambda)}{q_\lambda} t + c_3(\lambda)\Big) q_\lambda^t,$$

Theorem *(AID error upper bound)*

Let $v_t(\lambda) := (I - \partial_1 \Phi(w_t(\lambda), \lambda)^\top)^{-1} \nabla_1 E(w_t(\lambda), \lambda)$ and assume that

- $\|w_t(\lambda) - w(\lambda)\| \leq \rho_\lambda(t) \|w(\lambda)\|$,
- $\|v_{t,k}(\lambda) - v_t(\lambda)\| \leq \sigma_\lambda(k) \|v_t(\lambda)\|$.

Then,

$$\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\| \leq \Big(c_1(\lambda) + \frac{c_2(\lambda)}{1 - q_\lambda}\Big) \rho_\lambda(t) + c_3(\lambda) \sigma_\lambda(k).$$

Main Contribution (Part 2)

Efficient solvers for the **linear system** in AID:

- **fixed point method (FP)**
- **conjugate gradient (CG)**

Theorem *(CG and FP error upper bounds)*

Assume that the lower-level problem is solved as in ITD. Then

(FP) $$\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\| \leq \Big(c_1(\lambda) + c_2(\lambda)\frac{1-q_\lambda^k}{1-q_\lambda}\Big) q_\lambda^t + c_3(\lambda) q_\lambda^k.$$

Moreover, when $\partial_1 \Phi(w_t(\lambda), \lambda)$ is symmetric,

(CG) $$\|\hat{\nabla} f(\lambda) - \nabla f(\lambda)\| \leq \Big(c_1(\lambda) + \frac{c_2(\lambda)}{1-q_\lambda}\Big) q_\lambda^t + c_3(\lambda)\hat{c}(\lambda) p_\lambda^k,$$

where $p_\lambda < q_\lambda$.

So... Which method has the best approximation error?

From our analysis:

- ITD, CG and FP **converge linearly** (in t and k) to $\nabla f(\lambda)$.
- FP bound $<$ ITD bound for every t, when $k = t$.
- CG bound $<$ FP bound for k big enough when $\partial_1 \Phi(w_t(\lambda), \lambda)$ is symmetric.

So... Which method has the best approximation error?

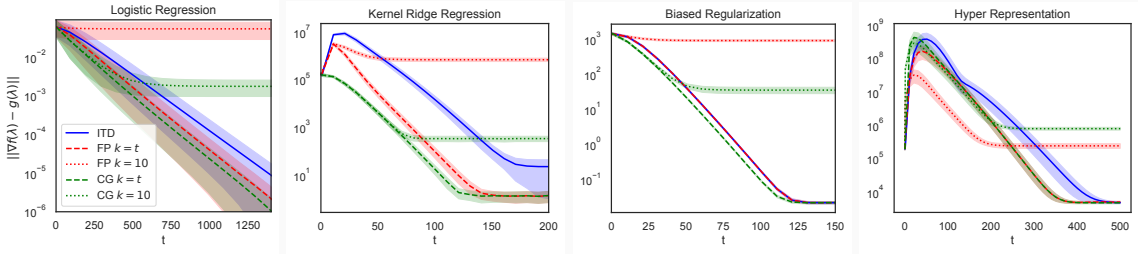
From our analysis:

- ITD, CG and FP **converge linearly** (in t and k) to $\nabla f(\lambda)$.
- FP bound $<$ ITD bound for every t, when $k = t$.
- CG bound $<$ FP bound for k big enough when $\partial_1 \Phi(w_t(\lambda), \lambda)$ is symmetric.

Is this true also for the actual error in practice?

What happens when $\Phi(\cdot, \lambda)$ is not a contraction?

Hypergradient Approximation on Synthetic Data

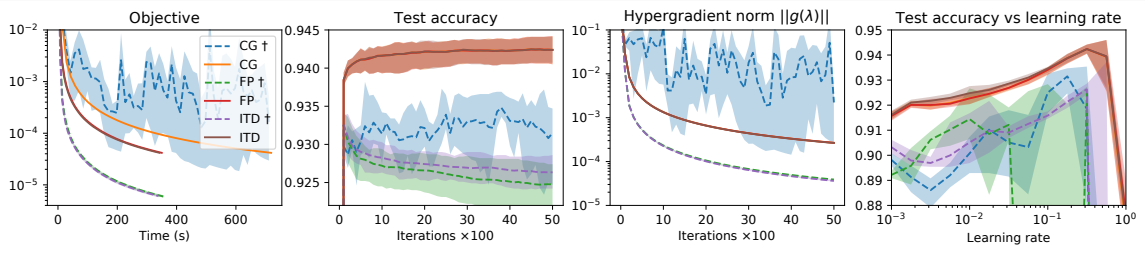


Hypergradient approximation errors (mean/std on randomly drawn values of λ). $g(\lambda)$ is equal to $\nabla f_t(\lambda)$ for ITD and to $\hat{\nabla} f(\lambda)$ for CG and FP. In all settings $\Phi(\cdot, \lambda)$ is a contraction and $\partial_1 \Phi(w, \lambda)$ is symmetric.

- After a while the error decreases linearly for all methods.
- Methods with lower error bounds have lower error on average.

Equilibrium Models [1] on MNIST (Proof of Concept)

$$\min_{\gamma=(A,B,c),\theta} \sum_{i=1}^{n} \mathrm{CE}(w_i(\gamma)^\top \theta, y_i), \qquad w_i(\gamma) = \phi_i(w_i(\gamma), \gamma) = \tanh(Aw_i(\gamma) + Bx_i + c)$$



$\phi_i(\cdot, \gamma)$ *NOT* a contraction for † methods.

- When $\phi_i(\cdot, \gamma)$ is a contraction all the methods perform similarly.
- ITD is the most stable when the contraction assumption is not satisfied.

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. "Deep equilibrium models". In Advances in Neural Information Processing Systems. 2019, pp. 688–699.

Conclusions

We studied the **iteration complexity** of two strategies used to approximate the *hypergradient* in bilevel problems: **iterative differentiation (ITD)** and **approximate implicit differentiation (AID)**.

We proved non-asymptotic upper bounds on the approximation error

- CG, FP and ITD converge linearly to the exact *hypergradient*.
- ITD is generally worse than AID in terms of upper bounds.

We conducted experiments comparing ITD and AID

- If $\Phi(\cdot, \lambda)$ is a contraction, the results confirm the theory.
- If $\Phi(\cdot, \lambda)$ is *NOT* a contraction, ITD can be still a reliable strategy.

Thank you for the attention

CODE (PyTorch): https://github.com/prolearner/hypertorch

References

Franceschi, L., Frasconi, P., Salzo, S., Grazzi, R., and Pontil, M. (2018). Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, pages 1563–1572.

Pedregosa, F. (2016). Hyperparameter optimization with approximate gradient. In *International Conference on Machine Learning*, pages 737–746.

Rajeswaran, A., Finn, C., Kakade, S. M., and Levine, S. (2019). Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, pages 113–124.