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

[illegible] (upper-level)

[illegible] (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

$$\mathrm{k\,g\,l}_{\text{سژ گ}^{\mathrm{c}}}\ \gamma\&\ '\,\mathbb{l}\,;\ \dot{}\ \&\text{Đ}\&\ '*\ '\quad \text{(upper-level)}$$

$$\text{Đ}\&\ '\,\mathbb{l}\,;\quad \&\text{Đ}\&\ '*\ '\quad \text{(lower-level)}$$

- 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* $\gamma\&\ '$.

Computing the Hypergradient $\nabla_{\lambda} \Phi$

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 $\nabla_{\theta} \mathcal{L}$

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 $\Phi(\lambda, w)$ 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 $\&\nu^* '$ is a contraction, the results confirm the theory.
- If $\&\nu^* '$ 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).

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

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

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

Motivation

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

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

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

- 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

$$\underset{\text{گ\&ï}}{\text{k g l}}\ \text{Ü}\ '\ \frac{/}{0}\text{ق}\square_{\text{val}}\text{Ð\&}\ '\ \text{L}\ \text{ვ}_{\text{val}}\text{ق}\phi$$

$$\text{Ð\&}\ '\ ;\ \underset{\text{ғ گسگ}^{\text{P}}}{\text{\_pe k g l}}\ \text{ﻝ}\ \&\text{Ð*}\ '\ \text{ﺎ};\ \frac{/}{0}\text{ق}\square\ \text{Ð}\ \text{L}\ \text{ვ}\text{ق}\phi\ )\ \frac{}{0}\text{ق}\text{Ð}\text{ق}\phi\text{ﻝ}$$

Ð\& ' is the **unique fixed point** of the *one step GD* map

$$\&\text{Ð*}\ '\ ;\ \text{Ð}\ \text{L}\quad_{\text{E-}}\&\text{Ð*}\ '$$

If the step size is sufficiently small, &ŭ* ' is also a **contraction**.

The Bilevel Framework

$$\underset{\text{گ}\,\text{ﷺ}^{\text{G}}}{\text{k g l}}\ \gamma\text{\&}\ \text{'}\,\text{ß}\,\text{;}\ \dot{}\ \text{\&Đ\&}\ \text{'*}\ \text{'}\quad \text{(upper-level)}$$

$$\text{Đ\&}\ \text{'}\,\text{ß}\,\text{;}\quad \text{\&Đ\&}\ \text{'*}\ \text{'}\quad \text{(lower-level)}$$

- $\text{Đ\&}\ \text{'}\,\text{گ}\text{ﷺ}^{\delta}$ is often not available in closed form.
- γ is usually non convex and **expensive** or **impossible** to evaluate exactly.

k g l ہمثر گ G ϒ& ' ⌷ ; ˙ &Đ& '* ' (upper-level)

Đ& ' ⌷ ; &Đ& '* ' (lower-level)

- Đ& ' ہمگ δ is often not available in closed form.
- ϒ is usually non convex and **expensive** or **impossible** to evaluate exactly.
- ϒ is even harder to evaluate.

How to Compute the Hypergradient ℊ& '?

Iterative Differentiaton (ITD)

1. Set ɾ ц& ' ; ï and compute,

   for ᵤ ; E· ф µ ʁ

   Ч ɾ ʙ& ' ; &ɾ ʙL ļ& '· '

2. Compute ᵡ ᴍ& ' ; ᵉ&ɾ ᴍ& '· '.

3. Compute ᵡ ᴍ& ' *efficiently* using reverse (RMAD) or forward (FMAD) mode automatic differentiation.

How to Compute the Hypergradient ∇_{λ} ?

Iterative Differentiaton (ITD)

1. Set ϕ_0 ; λ and compute,

   for t ; $\ldots$
   ϕ_t ; $\Phi_t(\phi_{t-1}, \lambda)$

2. Compute $\nabla_{\lambda} L_T(\phi_T, \lambda)$.

3. Compute $\nabla_{\lambda} L_T(\phi_T, \lambda)$ *efficiently* using reverse (RMAD) or forward (FMAD) mode automatic differentiation.

Approximate Implicit Differentiation (AID)

1. Get $\phi^*(\lambda)$ with K **steps** of a lower-level solver.

2. Compute $p \approx \left(\frac{\partial \phi^*}{\partial \lambda}\right)^T \nabla_{\phi} L_T(\phi^*(\lambda), \lambda)$ with β **steps** of a solver for the *linear system*

   $$\left(\frac{\partial \phi^*}{\partial \lambda}\right)^T \left(I - \frac{\partial \Phi}{\partial \phi}(\phi^*(\lambda), \lambda)\right)^T p = \nabla_{\phi} L_T(\phi^*(\lambda), \lambda)$$

3. Compute the approximate gradient as

   $$\nabla_{\lambda} L_T(\phi_T, \lambda) + \alpha \nabla_{\lambda} \Phi(\phi^*(\lambda), \lambda)^T p$$

How to Compute the Hypergradient $\nabla_{\lambda} f(\lambda)$?

Iterative Differentiaton (ITD)

1. Set $\phi_0(\lambda) = \phi_0$ and compute,

$$\text{for } t = 1, \ldots, T \quad \phi_t(\lambda) = \Phi(\phi_{t-1}(\lambda), \lambda)$$

2. Compute $f(\lambda) = L(\phi_T(\lambda))$.
3. Compute $\nabla f(\lambda)$ *efficiently* using reverse (RMAD) or forward (FMAD) mode automatic differentiation.

Approximate Implicit Differentiation (AID)

1. Get $\phi^*(\lambda) = \phi_T(\lambda)$ with T **steps** of a lower-level solver.
2. Compute $p_{T+1}(\lambda) = v$ with β **steps** of a solver for the *linear system*

$$\nabla_1 L(\phi^*(\lambda), \lambda) + v^\top \nabla_1 \Phi(\phi^*(\lambda), \lambda) = 0$$

3. Compute the approximate gradient as

$$\nabla f(\lambda) \approx \nabla_2 L(\phi^*(\lambda), \lambda) + p_{T+1}(\lambda)^\top \nabla_2 \Phi(\phi^*(\lambda), \lambda)$$

Which one is the best?

A First Comparison

ITD	AID
• Ignores the bilevel structure.	• Can use any lower-level solver.
• Cost in time (RMAD): ה&Amqr&ﾚ '''	• Cost in time (ת ; в): ה&Amqr&ﾚ '''.
• Cost in memory (RMAD): ה&вף'.	• Cost in memory: ה&ף'.
• **Can we control ق ﾚ& ' L ﾚ& 'ق?**	• **Can we control ق ,ﾚ& ' L ﾚ& 'ق?**

χ M& ' ; æ & Ŧ M& '. '.

Previous Work on the Approximation Error

ITD

- _pe kgl $\gamma_{m} \frac{\text{IJ} \Omega}{\text{m} \Omega \text{Ü}}$ _pe kgl γ (Franceschi et al., 2018).

AID

- ق ,γ& ' L γ& 'ق $\frac{\text{IJIJ}\Omega}{\text{m} \beta \Omega \text{Ü}}$. (Pedregosa, 2016).
- ق ,γ& ' L γ& 'ق $\frac{\text{IJIJ}\Omega}{\text{m} \beta \Omega \text{Ü}}$. at a **linear rate** in в and ת for meta-learning with biased regularization (Rajeswaran et al., 2019).

χ M& ' ; æ & ƚ M& '· '.

Previous Work on the Approximation Error

ITD

- _pe kgl γ_m ΠΓΩ _pe kgl γ (Franceschi et al., 2018).
- We provide non-asymptotic upper bounds on ق γ_m& ' L γ& 'ق.

AID

- ق ,γ& ' L γ& 'ق ΠΓΩ . (Pedregosa, 2016).
- ق ,γ& ' L γ& 'ق ΠΓΩ . at a **linear rate** in ʙ and ת for meta-learning with biased regularization (Rajeswaran et al., 2019).
- We provide non-asymptotic upper bounds on ق ,γ& ' L γ& 'ق.

χ_M& ' ; æ&ɾ_M& '· '.

Preliminaries

Assumptions

- $\&\mathbb{V}^* \ '$ is a **contraction** with constant $\mathrm{A} \ : \ /$.
- $_{\mathrm{E}} \ \&\mathbb{V}^* \ ', \ _{\phi} \ \&\mathbb{V}^* \ ', \ _{\mathrm{E}}^{\cdot} \ \&\mathbb{V}^* \ '$ and $_{\phi}^{\cdot} \ \&\mathbb{V}^* \ '$ are Lipschitz continuous

№ γ **differentiable** and

$$\text{Ð}\tilde{\text{-}}\& \ ' \ \text{ß} \ ; \ \&\text{ı} \ \text{L} \ _{\mathrm{E}} \ \&\text{Ð}\& \ '^* \ ''\text{LE} \ _{\phi} \ \&\text{Ð}\& \ '^* \ '$$
$$\gamma\& \ ' \ ; \ _{\phi}^{\cdot} \ \&\text{Ð}\& \ '^* \ ' \ ) \ \text{Ð}\tilde{\text{-}}\& \ '\tilde{} \ _{\mathrm{E}}^{\cdot} \ \&\text{Ð}\& \ '^* \ ',$$

Main Contribution

Theorem $\flat^{2}\dot{I}P\ \bar{a}\mathbb{W}\mathbb{W}\omega\mathbb{W}\ \cdot\ \S\S\bar{a}\mathbb{W}\ \varepsilon\omega\cdot\ \hbar\breve{a}\square$

$$\text{ق}\ \ \gamma_{\text{ᵯ}}\&\ \ '\ \text{L}\ \ \ \gamma\&\ \ '\text{ق}\ \text{ب}\ \text{ا}\mathbb{u}_{\text{E}}\&\ \ '\ )\ \frac{\mathbb{u}_{\phi}\&\ \ '}{\text{A}}\text{ʙ}\ )\ \mathbb{u}_{\text{o}}\&\ \ '\text{ا}\text{A}^{\text{ᵯ}}*$$

Theorem $\flat\ \ ^{2}P\ \bar{a}\mathbb{W}\mathbb{W}\omega\mathbb{W}\ \cdot\ \S\S\bar{a}\mathbb{W}\ \varepsilon\omega\cdot\ \hbar\breve{a}\square$

Let $\text{D}_{\text{ᵯ}}\&\ \ '\mathbb{I};\ \&\text{ɾ}\ \text{L}\ \ _{\text{E}}\ \ \&\text{Đ}_{\text{ᵯ}}\&\ \ '*\ \ '\tilde{}\ '\text{LE}\ \ _{\dot{\text{E}}}\ \ \&\text{Đ}_{\text{ᵯ}}\&\ \ '*\ \ '$ and assume that

- $\text{ق}\text{Đ}_{\text{ᵯ}}\&\ \ '\ \text{L}\ \text{Đ}\&\ \ '\text{ق}\ \text{ب}\ \ \ \&\text{ʙ}'\text{ق}\text{Đ}\&\ \ '\text{ق},$
- $\text{ق}\text{D}_{\text{ᵯ}\ \beta}\&\ \ '\ \text{L}\ \text{D}_{\text{ᵯ}}\&\ \ '\text{ق}\ \text{ب}\ \ \ \&\text{ᴨ}'\text{ق}\text{D}_{\text{ᵯ}}\&\ \ '\text{ق}.$

Then,

$$\text{ق}\ \ ,\gamma\&\ \ '\ \text{L}\ \ \ \gamma\&\ \ '\text{ق}\ \text{ب}\ \text{ا}\mathbb{u}_{\text{E}}\&\ \ '\ )\ \frac{\mathbb{u}_{\phi}\&\ \ '}{/\ \text{L}\ \text{A}}\text{ا}\ \ \ \&\text{ʙ}'\ )\ \mathbb{u}_{\text{o}}\&\ \ '\ \ \ \&\text{ᴨ}',$$

Main Contribution (Part 2)

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

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

Theorem [illegible]

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

(FP) [illegible]

Moreover, when [illegible] is symmetric,

(CG) [illegible]

where [illegible].

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

From our analysis:

- ITD, CG and FP **converge linearly** (in в and т) to ɣ& '.
- FP bound : ITD bound for every в, when т ; в.
- CG bound : FP bound for т big enough when E &Ð_в& '* ' is symmetric.

So… Which method has the best approximation error?

From our analysis:

- ITD, CG and FP **converge linearly** (in в and т) to ɣ& '.
- FP bound : ITD bound for every в, when т ; в.
- CG bound : FP bound for т big enough when E &Ð$_{в}$& '* ' is symmetric.

Is this true also for the actual error in practice?

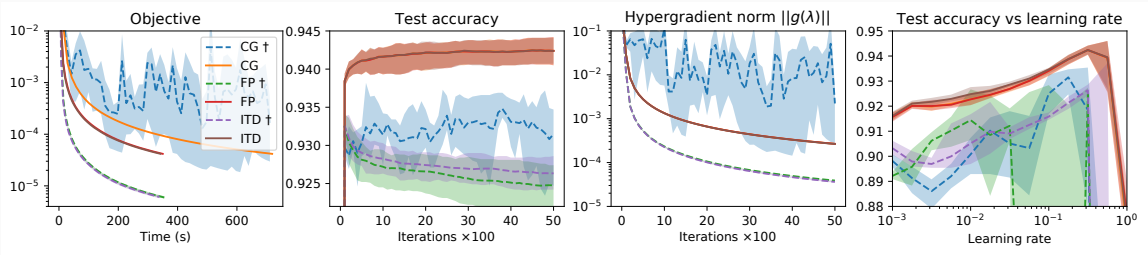
What happens when &ʊ* ' is not a contraction?

Hypergradient Approximation on Synthetic Data

Hypergradient approximation errors (mean/std on randomly drawn values of). ɀ& ' is equal to ɣ& ' for ITD and to ɣ& ' for CG and FP. In all settings &ɓ* ' is a contraction and &Ð* ' 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)



- When α 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 $\&\wr^{*}\ '$ is a contraction, the results confirm the theory.
- If $\&\wr^{*}\ '$ 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.