A framework for bilevel optimization that enables stochastic and global variance reduction algorithms

M. Dagréou, P. Ablin, S. Vaiter, T. Moreau

https://arxiv.org/abs/2201.13409

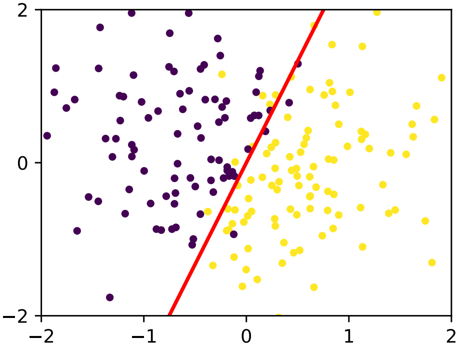
February 9, 2023

Motivating example

Classification problem

Setup:

- Data $(x_i)_{1\leq i\leq n}$ in $\mathbb{R}^p$, target binary $(y_i)_{1\leq i\leq n}$ in $\{-1,1\}$
- Goal: find a parameter $\theta^* \in \mathbb{R}^p$ to predict the class y by $\mathrm{sign}(\langle x, \theta^* \rangle)$



Logistic regression

Logistic loss:

$$G(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i \langle x_i, \theta \rangle))$$

Training:

$$\theta^* \in \underset{\theta \in \mathbb{R}^p}{\arg\min}\, G(\theta)$$

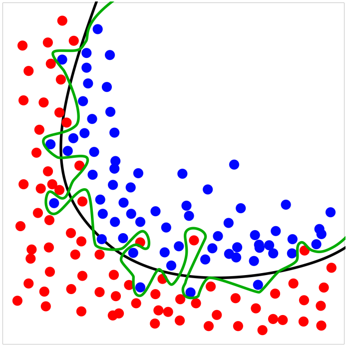
Avoiding overfitting

Regularized logistic loss:

$$G(\theta, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i\langle x_i, \theta\rangle)) + \frac{\lambda}{2}\|\theta\|_2^2$$

Training:

$$\theta^*(\lambda) \in \operatorname*{arg\,min}_{\theta \in \mathbb{R}^p} G(\theta, \lambda)$$



Source: `https://fr.wikipedia.org/wiki/Surapprentissage`

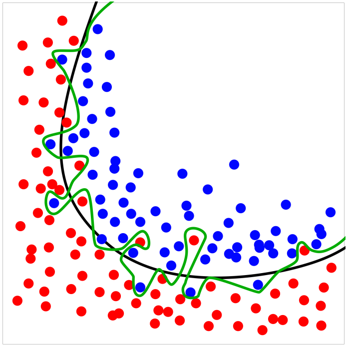
Avoiding overfitting

Regularized logistic loss:

$$G(\theta, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i\langle x_i, \theta\rangle)) + \frac{\lambda}{2}\|\theta\|_2^2$$

Training:

$$\theta^*(\lambda) \in \arg\min_{\theta\in\mathbb{R}^p} G(\theta, \lambda)$$



Source: https://fr.wikipedia.org/wiki/Surapprentissage

How to choose λ?

Grid search

1. Define a grid $\{\lambda_1, \ldots, \lambda_K\}$

Grid search

1. Define a grid $\{\lambda_1, \dots, \lambda_K\}$
2. Train the model for each λ_k to get the parameters $\theta^*(\lambda_1), \dots, \theta^*(\lambda_K)$

Grid search

1. Define a grid $\{\lambda_1, \ldots, \lambda_K\}$
2. Train the model for each λ_k to get the parameters $\theta^*(\lambda_1), \ldots, \theta^*(\lambda_K)$
3. Evaluate the performances on validation samples $(x_i^{\mathrm{val}}, y_i^{\mathrm{val}})_{1 \leq i \leq m}$ *not* used in the training phase by computing

$$F(\theta^*(\lambda_k)) = \frac{1}{m} \sum_{i=1}^{m} \log(1 + \exp(-y_i^{\mathrm{val}} \langle x_i^{\mathrm{val}}, \theta^*(\lambda_k) \rangle)) \ .$$

Grid search

1. Define a grid $\{\lambda_1, \ldots, \lambda_K\}$
2. Train the model for each λ_k to get the parameters $\theta^*(\lambda_1), \ldots, \theta^*(\lambda_K)$
3. Evaluate the performances on validation samples $(x_i^{\text{val}}, y_i^{\text{val}})_{1\leq i\leq m}$ *not* used in the training phase by computing

$$F(\theta^*(\lambda_k)) = \frac{1}{m}\sum_{i=1}^m \log(1+\exp(-y_i^{\text{val}}\langle x_i^{\text{val}}, \theta^*(\lambda_k)\rangle)) \ .$$

4. Keep the value of λ that gives the lowest value of $F(\theta^*(\lambda))$.

Grid search as a bilevel optimization problem

Grid search = "Find λ such that $F(\theta^*(\lambda))$ is the lowest possible."

Grid search as a bilevel optimization problem

Grid search = "Find λ such that $F(\theta^*(\lambda))$ is the lowest possible."

Mathematical formalization: Bilevel optimization problem

$$\begin{cases} \min_\lambda h(\lambda) \triangleq F(\theta^*(\lambda)) \\ \theta^*(\lambda) \in \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \lambda) \end{cases}$$

Grid search with multiple hyperparameters

Regularized logistic loss:

$$G(\theta, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i\langle x_i, \theta\rangle)) + \frac{\lambda}{2}\|\theta\|_2^2$$

Grid search with multiple hyperparameters

Regularized logistic loss:

$$G(\theta, \lambda) = \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-y_i \langle x_i, \theta \rangle)) + \frac{1}{2} \sum_{k=1}^{p} \lambda_k \theta_k^2$$

The number of trials increases exponentially with the dimension of λ.

Grid search with multiple hyperparameters

Regularized logistic loss:

$$G(\theta, \lambda) = \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-y_i \langle x_i, \theta \rangle)) + \frac{1}{2} \sum_{k=1}^{p} \lambda_k \theta_k^2$$

The number of trials increases exponentially with the dimension of λ.

Can we use first-order information in order to minimize $h(\lambda) = F(\theta^*(\lambda))$?

Problem statement

Bilevel optimization in general

Bilevel optimization problem

$$\begin{cases} \min_{\lambda\in\mathbb{R}^d} h(\lambda) \triangleq F(\theta^*(\lambda),\lambda) & \text{Outer problem} \\ \theta^*(\lambda) \in \arg\min_{\theta\in\mathbb{R}^p} G(\theta,\lambda) & \text{Inner problem} \end{cases}$$

Neural Architecture Search

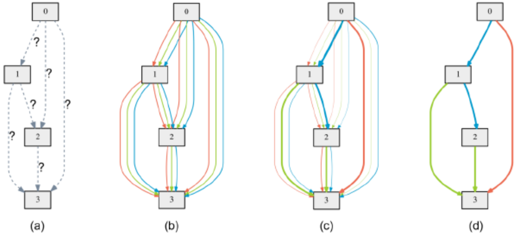
Darts [Liu et al. 2019]:
Differentiable Architecture Search

Goal: Find the best architecture of a Neural Network for a given task

Idea: Parametrize the probability of the architectures by λ

Bilevel formulation:

$$\begin{cases} \min_{\lambda \in \mathbb{R}^d} \mathcal{L}_{\text{val}}(\theta^*(\lambda), \lambda) \\ \theta^*(\lambda) \in \arg\min_{\theta \in \mathbb{R}^p} \mathcal{L}_{\text{train}}(\theta, \lambda) \end{cases}$$

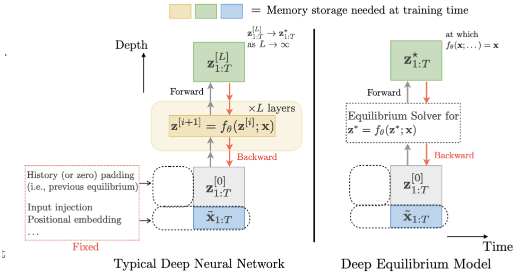


Source: [Liu et al. 2019]

Deep Equilibrium Networks [Bai et al. 2019]

Idea: Replace the forward pass by a root finding problem $g(z, \theta) = 0$
Training a DEQ: Boils down to solve

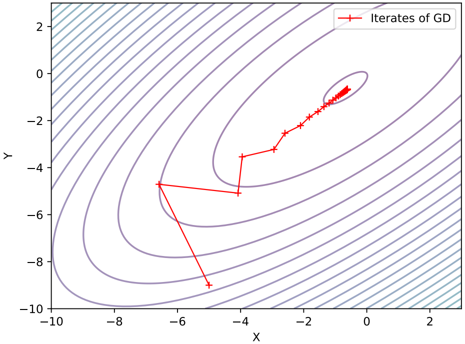
$$\min_{\theta} \mathcal{L}(z^*(\theta)), \quad g(z^*(\theta), \theta) = 0$$



Gradient descent

Gradient descent on *h*:

$$\lambda^{t+1} = \lambda^t - \gamma^t \nabla h(\lambda^t)$$



Gradient of h?

Definition of h:

$$h(\lambda) = F(\theta^*(\lambda), \lambda), \quad \theta^*(\lambda) \in \operatorname*{arg\,min}_{\theta \in \mathbb{R}^p} G(\theta, \lambda)$$

Gradient of h?

Definition of h:

$$h(\lambda) = F(\theta^*(\lambda), \lambda), \quad \theta^*(\lambda) \in \operatorname*{arg\,min}_{\theta \in \mathbb{R}^p} G(\theta, \lambda)$$

Chain rule:

$$\nabla h(\lambda) = \nabla_2 F(\theta^*(\lambda), \lambda) + (\mathrm{d}\theta^*(\lambda))^\top \nabla_1 F(\theta^*(\lambda), \lambda)$$

Implicit differentiation

Optimality condition for $\theta^*(\lambda)$:

$$\nabla_1 G(\theta^*(\lambda), \lambda) = 0$$

Implicit differentiation

Optimality condition for $\theta^*(\lambda)$:

$$\nabla_1 G(\theta^*(\lambda), \lambda) = 0$$

Implicit function theorem:

$$\mathrm{d}\theta^*(\lambda) = -\left[\nabla^2_{11} G(\theta^*(\lambda), \lambda)\right]^{-1} \nabla^2_{12} G(\theta^*(\lambda), \lambda)$$

Implicit gradient in practice

Gradient of h:

$$\nabla h(\lambda) = \nabla_2 F(\theta^*(\lambda), \lambda) - \nabla^2_{21} G(\theta^*(\lambda), \lambda) \left[\nabla^2_{11} G(\theta^*(\lambda), \lambda)\right]^{-1} \nabla_1 F(\theta^*(\lambda), \lambda)$$

Implicit gradient in practice

Gradient of h:

$$\nabla h(\lambda) = \nabla_2 F(\theta^*(\lambda), \lambda) - \nabla^2_{21} G(\theta^*(\lambda), \lambda) \left[\nabla^2_{11} G(\theta^*(\lambda), \lambda)\right]^{-1} \nabla_1 F(\theta^*(\lambda), \lambda)$$

- Need to solve the inner optimization problem

Implicit gradient in practice

Gradient of h**:**

$$\nabla h(\lambda) = \nabla_2 F(\theta^*(\lambda), \lambda) - \nabla^2_{21} G(\theta^*(\lambda), \lambda) \left[\nabla^2_{11} G(\theta^*(\lambda), \lambda)\right]^{-1} \nabla_1 F(\theta^*(\lambda), \lambda)$$

- Need to solve the inner optimization problem
- Need to solve a linear system of size $p \times p$

Empirical Risk minimization

Classical ML setting:

$$F(\theta,\lambda)=\frac{1}{m}\sum_{j=1}^{m}F_j(\theta,\lambda),\quad G(\theta,\lambda)=\frac{1}{n}\sum_{i=1}^{n}G_i(\theta,\lambda)$$

Empirical Risk minimization

Classical ML setting:

$$F(\theta, \lambda) = \frac{1}{m}\sum_{j=1}^{m} F_j(\theta, \lambda), \quad G(\theta, \lambda) = \frac{1}{n}\sum_{i=1}^{n} G_i(\theta, \lambda)$$

Consequence: For large m and n, any single derivative is cumbersome to compute.

Aside: Stochastic optimization for single level problems

Single level problem:

$$\min_{\theta} f(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

Aside: Stochastic optimization for single level problems

Single level problem:

$$\min_{\theta} f(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

First-order stochastic optimization:

$$\theta^{t+1} = \theta^t - \rho^t g^t, \quad \mathbb{E}[g^t|\theta^t] = \nabla f(\theta^t)$$

Aside: Stochastic optimization for single level problems

Single level problem:

$$\min_{\theta} f(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

First-order stochastic optimization:

$$\theta^{t+1} = \theta^t - \rho^t g^t, \quad \mathbb{E}[g^t|\theta^t] = \nabla f(\theta^t)$$

Example: stochastic gradient descent [Robbins and Monro 1951]:

$$\theta^{t+1} = \theta^t - \rho^t \nabla f_i(\theta^t), \quad i \sim \mathcal{U}(\{1, \dots, n\})$$

Bilevel optimization case

$$F(\theta, \lambda) = \frac{1}{m} \sum_{j=1}^{m} F_j(\theta, \lambda), \quad G(\theta, \lambda) = \frac{1}{n} \sum_{i=1}^{n} G_i(\theta, \lambda)$$

Bilevel optimization case

$$F(\theta, \lambda) = \frac{1}{m} \sum_{j=1}^{m} F_j(\theta, \lambda), \quad G(\theta, \lambda) = \frac{1}{n} \sum_{i=1}^{n} G_i(\theta, \lambda)$$

$$\nabla h(\lambda) = \nabla_2 F(\theta^*(\lambda), \lambda) - \nabla^2_{21} G(\theta^*(\lambda), \lambda) \left[\nabla^2_{11} G(\theta^*(\lambda), \lambda)\right]^{-1} \nabla_1 F(\theta^*(\lambda), \lambda)$$

Bilevel optimization case

$$F(\theta,\lambda) = \frac{1}{m}\sum_{j=1}^{m} F_j(\theta,\lambda), \quad G(\theta,\lambda) = \frac{1}{n}\sum_{i=1}^{n} G_i(\theta,\lambda)$$

$$\nabla h(\lambda) = \nabla_2 F(\theta^*(\lambda),\lambda) - \nabla^2_{21} G(\theta^*(\lambda),\lambda)\left[\nabla^2_{11} G(\theta^*(\lambda),\lambda)\right]^{-1}\nabla_1 F(\theta^*(\lambda),\lambda)$$

Problem:

$$\left[\sum_{i=1}^{n} \nabla^2_{11} G_i(\theta^*(\lambda),\lambda)\right]^{-1} \neq \sum_{i=1}^{n}\left[\nabla^2_{11} G_i(\theta^*(\lambda),\lambda)\right]^{-1}$$

Summary

Can we progress in the problem **without**

- computing exactly $\theta^*(\lambda)$ at each iteration?

Summary

Can we progress in the problem **without**

- computing exactly $\theta^*(\lambda)$ at each iteration?
- solving exactly $\left[\nabla^2_{11} G(\theta^*(\lambda), \lambda)\right]^{-1} \nabla_1 F(\theta^*(\lambda), \lambda)$ at each iteration?

Summary

Can we progress in the problem **without**

- computing exactly $\theta^*(\lambda)$ at each iteration?
- solving exactly $\left[\nabla^2_{11} G(\theta^*(\lambda), \lambda)\right]^{-1} \nabla_1 F(\theta^*(\lambda), \lambda)$ at each iteration?
- using all the samples at each iteration?

Related work

General algorithm

for $t = 1, \dots, T$ **do**

1. Take for θ^t an approximation of $\theta^*(\lambda^t)$
2. Take for v^t an approximation of $\left[\nabla^2_{11} G(\theta^t, \lambda^t)\right]^{-1} \nabla_1 F(\theta^t, \lambda^t)$
3. Set
$$p^t = \underbrace{\nabla_2 F(\theta^t, \lambda^t) - \nabla^2_{12} G(\theta^t, \lambda^t) v^t}_{\approx \nabla h(\lambda^t)}$$
4. Update the outer variable
$$\lambda^{t+1} = \lambda^t - \gamma^t p^t$$

Two loops algorithms

Two loops [Ghadimi et al. 2018]: $\theta^*(\lambda^t)$ is approximated by output of K steps of SGD:

$$\theta^{t,k+1} = \theta^{t,k} - \rho^t \nabla_1 G_i(\theta^{t,k}, \lambda^t)$$

Warm start strategy [Ji et al. 2021, Arbel and Mairal 2022]: Initialize the inner SGD by the previous iterate θ^{t-1}.

What about the linear system?

Approximate $v^t = [\nabla^2_{11} G(\theta^t, \lambda^t)]^{-1} \nabla_1 F(\theta^t, \lambda^t)$ with:

- Neumann approximations [Ghadimi et al. 2018, Ji et al. 2021]:

$$\left[\nabla^2_{11} G(\theta^t, \lambda^t)\right]^{-1} = \eta \sum_{q=0}^{+\infty} \left(I - \eta \nabla^2_{11} G(\theta^t, \lambda^t)\right)^q$$

What about the linear system?

Approximate $v^t = \left[\nabla^2_{11} G(\theta^t, \lambda^t)\right]^{-1} \nabla_1 F(\theta^t, \lambda^t)$ with:

- Neumann approximations [Ghadimi et al. 2018, Ji et al. 2021]:

$$\left[\nabla^2_{11} G(\theta^t, \lambda^t)\right]^{-1} \approx \eta \sum_{q=0}^{Q} \left(I - \eta \nabla^2_{11} G(\theta^t, \lambda^t)\right)^q$$

What about the linear system?

Approximate $v^t = \left[\nabla^2_{11} G(\theta^t, \lambda^t)\right]^{-1} \nabla_1 F(\theta^t, \lambda^t)$ with:

- Neumann approximations [Ghadimi et al. 2018, Ji et al. 2021]:

$$\left[\nabla^2_{11} G(\theta^t, \lambda^t)\right]^{-1} \approx \eta \sum_{q=0}^{Q} \prod_{k=0}^{q} \left(I - \eta \nabla^2_{11} G_{i_k}(\theta^t, \lambda^t)\right)^{\cancel{q}}$$

What about the linear system?

Approximate $v^t = [\nabla^2_{11} G(\theta^t, \lambda^t)]^{-1} \nabla_1 F(\theta^t, \lambda^t)$ with:

- Neumann approximations [Ghadimi et al. 2018, Ji et al. 2021]:

$$v^t \approx \eta \sum_{q=0}^{Q} \prod_{k=0}^{q} \left(I - \eta \nabla^2_{11} G_{i_k}(\theta^t, \lambda^t)\right) \nabla_1 F_j(\theta^t, \lambda^t)$$

What about the linear system?

Approximate $v^t = \left[\nabla^2_{11} G(\theta^t, \lambda^t)\right]^{-1} \nabla_1 F(\theta^t, \lambda^t)$ with:

- Neumann approximations [Ghadimi et al. 2018, Ji et al. 2021]:

$$v^t \approx \eta \sum_{q=0}^{Q} \prod_{k=0}^{q} \left(I - \eta \nabla^2_{11} G_{i_k}(\theta^t, \lambda^t)\right) \nabla_1 F_j(\theta^t, \lambda^t)$$

- Stochastic Gradient Descent [Grazzi et al. 2021] since

$$v^t \in \arg\min_{v \in \mathbb{R}^p} \frac{1}{2} \langle \nabla^2_{11} G(\theta^t, \lambda^t) v, v \rangle + \langle \nabla_1 F(\theta^t, \lambda^t), v \rangle$$

One loop algorithms

Alternate steps in θ and λ [Hong et al. 2020, Yang et al. 2021]:

$$\theta^{t+1} = \theta^t - \rho^t \nabla_1 G_i(\theta^t, \lambda^t) \quad \text{SGD step}$$

$$v^{t+1} = \eta \sum_{q=1}^{Q} \prod_{k=0}^{q} \left(I - \eta \nabla^2_{11} G_{i_k}(\theta^{t+1}, \lambda^t)\right) \nabla_1 F_j(\theta^{t+1}, \lambda^t) \quad \text{Neumann approximation}$$

$$\lambda^{t+1} = \lambda^t - \gamma^t \underbrace{(\nabla_2 F_j(\theta^{t+1}, \lambda^t) - \nabla^2_{21} G_i(\theta^{t+1}, \lambda^t) v^{t+1})}_{\approx \nabla h(\lambda^t)}$$

A new framework for stochastic bilevel optimization

Main idea

Three variables to maintain:

- $\theta \to$ inner optimization problem
- $v \to$ linear system
- $\lambda \to$ outer optimization problem

Idea: evolve in θ, v and λ at the same time following well chosen directions.

Motivation of the framework

Directions:

$$D_\theta(\theta, v, \lambda) = \nabla_1 G(\theta, \lambda) \quad \text{gradient step toward } \theta^*(\lambda)$$

Motivation of the framework

Directions:

$$D_\theta(\theta, v, \lambda) = \nabla_1 G(\theta, \lambda) \quad \text{gradient step toward } \theta^*(\lambda)$$

$$D_v(\theta, v, \lambda) = \nabla^2_{11} G(\theta, \lambda) v + \nabla_1 F(\theta, \lambda)$$

$$\text{gradient step toward } -\left[\nabla^2_{11} G(\theta, \lambda)\right]^{-1} \nabla_1 F(\theta, \lambda)$$

Motivation of the framework

Directions:

$$D_\theta(\theta, v, \lambda) = \nabla_1 G(\theta, \lambda) \quad \text{gradient step toward } \theta^*(\lambda)$$

$$D_v(\theta, v, \lambda) = \nabla^2_{11} G(\theta, \lambda) v + \nabla_1 F(\theta, \lambda)$$

$$\text{gradient step toward } -\left[\nabla^2_{11} G(\theta, \lambda)\right]^{-1} \nabla_1 F(\theta, \lambda)$$

$$D_\lambda(\theta, v, \lambda) = \nabla^2_{21} G(\theta, \lambda) v + \nabla_2 F(\theta, \lambda)$$

$$\text{gradient step toward } \lambda^*$$

Motivation of the framework

Directions:

$$D_\theta(\theta, v, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \nabla_1 G_i(\theta, \lambda)$$

$$D_v(\theta, v, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \nabla^2_{11} G_i(\theta, \lambda)v + \frac{1}{m}\sum_{j=1}^{m} \nabla_1 F_j(\theta, \lambda)$$

$$D_\lambda(\theta, v, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \nabla^2_{21} G_i(\theta, \lambda)v + \frac{1}{m}\sum_{j=1}^{m} \nabla_2 F_j(\theta, \lambda)$$

Proposed framework

for $t = 1, \dots, T$ **do**

1. Update θ

$$\theta^{t+1} = \theta^t - \rho^t D_\theta^t$$

2. Update v

$$v^{t+1} = v^t - \rho^t D_v^t$$

3. Update λ

$$\lambda^{t+1} = \lambda^t - \gamma^t D_\lambda^t$$

with D_θ^t, D_v^t, D_λ^t stochastic estimators of $D_\theta(\theta^t, v^t, \lambda^t)$, $D_v(\theta^t, v^t, \lambda^t)$ and $D_\lambda(\theta^t, v^t, \lambda^t)$.

SOBA (StOchastic Bilevel Algorithm) directions

Pick $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, m\}$ and take

$$
\begin{aligned}
D_\theta^t &= \nabla_1 G_i(\theta^t, \lambda^t) \\
D_v^t &= \nabla_{11}^2 G_i(\theta^t, \lambda^t) v^t + \nabla_1 F_j(\theta^t, \lambda^t) \\
D_\lambda^t &= \nabla_{21}^2 G_i(\theta^t, \lambda^t) v^t + \nabla_2 F_j(\theta^t, \lambda^t)
\end{aligned}
$$

SOBA (StOchastic Bilevel Algorithm) directions

$$\mathbb{E}_{i,j}[D_\theta^t] = \frac{1}{n}\sum_{i=1}^{n} \nabla_1 G_i(\theta^t, \lambda^t) = D_\theta(\theta^t, v^t, \lambda^t)$$

$$\mathbb{E}_{i,j}[D_v^t] = \frac{1}{n}\sum_{i=1}^{n} \nabla_{11}^2 G_i(\theta^t, \lambda^t) v^t + \frac{1}{m}\sum_{j=1}^{m} \nabla_1 F_j(\theta^t, \lambda^t) = D_v(\theta^t, v^t, \lambda^t)$$

$$\mathbb{E}_{i,j}[D_\lambda^t] = \frac{1}{n}\sum_{i=1}^{n} \nabla_{21}^2 G_i(\theta^t, \lambda^t) v^t + \frac{1}{m}\sum_{j=1}^{m} \nabla_2 F_j(\theta^t, \lambda^t) = D_\lambda(\theta^t, v^t, \lambda^t)$$

Theoretical guarantees of SOBA

Theorem (Convergence of SOBA)

Under some regularity assumptions on F and G, then for decreasing step sizes that verify $\rho^t = \alpha t^{-\frac{1}{2}}$ and $\gamma^t = \beta t^{-\frac{1}{2}}$ for some $\alpha, \beta > 0$, the iterates $(\lambda^t)_{1\leq t\leq T}$ of SOBA verify

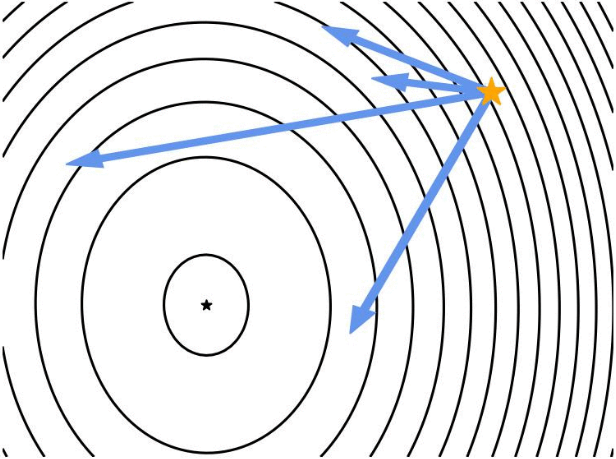
$$\inf_{t\leq T} \mathbb{E}[\|\nabla h(\lambda^t)\|^2] = \mathcal{O}(\log(T)T^{-\frac{1}{2}}) \ .$$

Same convergence rate as SGD for non-convex single level problems![1]

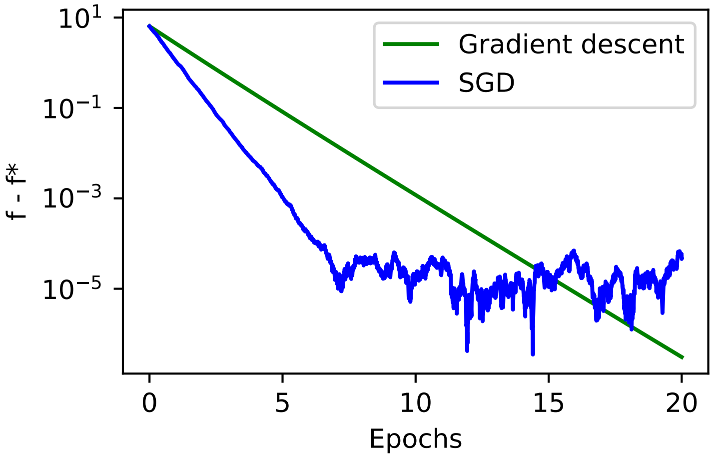
[1] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming, *SIAM Journal on Optimization*, 2013

Toward variance reduction methods

SGD, large step



Toward variance reduction methods



Aside: SAGA for single level problems [Defazio et al. 2014]

Single level problem:

$$\min_{\theta \in \mathbb{R}^p} f(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

Aside: SAGA for single level problems [Defazio et al. 2014]

Single level problem:

$$\min_{\theta \in \mathbb{R}^p} f(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

Initialisation: Compute and store $m[i] = \nabla f_i(\theta^0)$ for any $i \in \{1, \dots, n\}$ and $S[m] = \frac{1}{n} \sum_{i=1}^{n} m[i]$.

Aside: SAGA for single level problems [Defazio et al. 2014]

Single level problem:

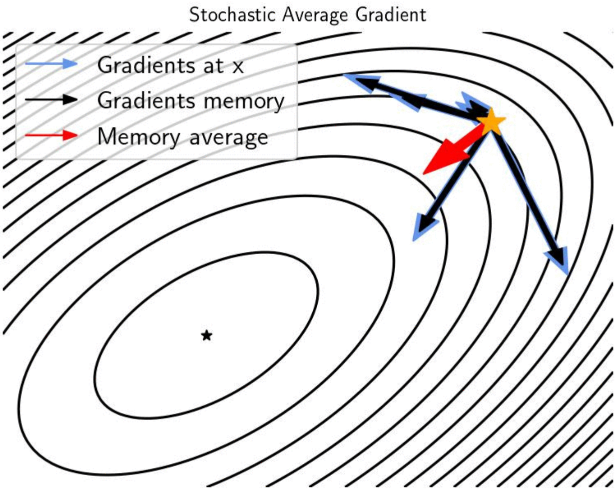
$$\min_{\theta\in\mathbb{R}^p} f(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$$

Initialisation: Compute and store $m[i] = \nabla f_i(\theta^0)$ for any $i \in \{1, \dots, n\}$ and $S[m] = \frac{1}{n}\sum_{i=1}^{n} m[i]$.

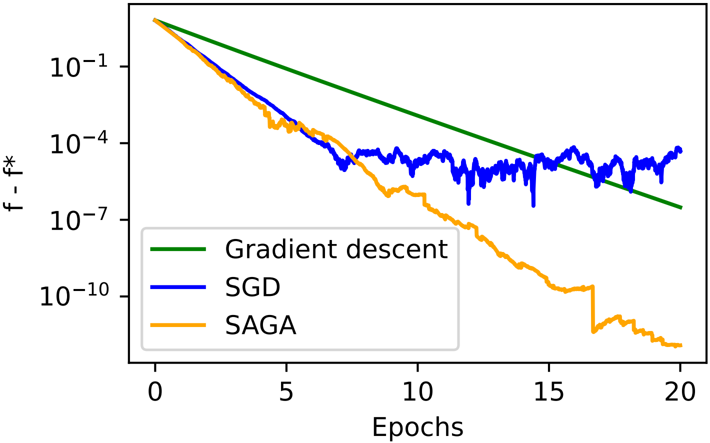
At iteration t:

1. Pick $i \in \{1, \dots, n\}$
2. Update θ
$$\theta^{t+1} = \theta^t - \rho(\nabla f_i(\theta^t) \underbrace{-m[i] + S[m]}_{\text{variance reduction}})$$
3. Update the memory
$$m[i] \leftarrow \nabla f_i(\theta^t)$$

Aside: SAGA for single level problems



Aside: SAGA for single level problems



Bilevel case: SABA (Stochastic Average Bilevel Algorithm)

To estimate

$$\begin{aligned}
D_\theta(\theta^t, v^t, \lambda^t) &= \nabla_1 G(\theta^t, \lambda^t) \\
D_v(\theta^t, v^t, \lambda^t) &= \nabla^2_{11} G(\theta^t, \lambda^t) v^t + \nabla_1 F(\theta^t, \lambda^t) \\
D_\lambda(\theta^t, v^t, \lambda^t) &= \nabla^2_{21} G(\theta^t, \lambda^t) v^t + \nabla_2 F(\theta^t, \lambda^t)
\end{aligned}$$

we have 5 quantities to estimate on the principle of SAGA:

$$\nabla_1 G(\theta^t, \lambda^t), \quad \nabla_1 F(\theta^t, \lambda^t), \quad \nabla_2 F(\theta^t, \lambda^t)$$
$$\nabla^2_{12} G(\theta^t, \lambda^t) v^t, \quad \nabla^2_{11} G(\theta^t, \lambda^t) v^t$$

D_θ^t, D_v^t and D_λ^t given using these estimates = **SABA directions**

Theoretical guarantees

Theorem (Convergence of SABA)

Under some regularity assumptions on F and G, with constant and small enough step sizes, the iterates $(\lambda^t)_{1\leq t\leq T}$ of SABA verify

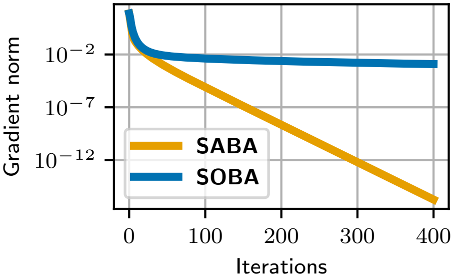
$$\frac{1}{T}\sum_{i=1}^{T}\mathbb{E}[\|\nabla h(\lambda^t)\|^2] = \mathcal{O}((n+m)^{\frac{2}{3}}T^{-1}) \ .$$

Same convergence rate as SAGA for non-convex single level problems![2]

[2] S. J. Reddi, S. Sra, B. Póczos and A. Smola, Fast incremental method for smooth nonconvex optimization, In *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016

Remarks

- We match the convergence rate of gradient descent
- SABA converges with fixed step sizes
- Faster than SOBA



Complexity

Number of calls to oracle to get an ϵ-stationary solution.

BSA	amIGO	stocBiO	TTSA	MRBO	SUSTAIN	**SOBA**	**SABA**
$\mathcal{O}(\epsilon^{-3})$	$\mathcal{O}(\epsilon^{-2})$	$\tilde{\mathcal{O}}(\epsilon^{-2})$	$\tilde{\mathcal{O}}(\epsilon^{-5/2})$	$\tilde{\mathcal{O}}(\epsilon^{-3/2})$	$\mathcal{O}(\epsilon^{-3/2})$	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(\epsilon^{-1})$

SABA achieves SOTA complexity

Numerical experiments

Hyperparameter selection on ℓ^2 regularized logistic regression

Setting:

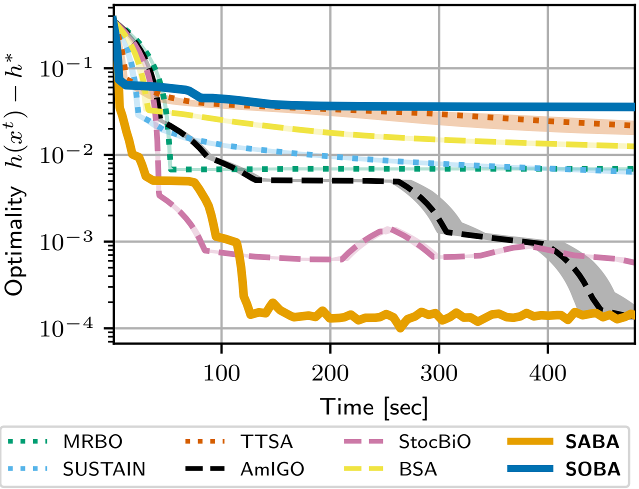
- Task: binary classification
- IJCNN1 dataset: 49 990 training samples, 91 701 validation samples, 22 features
- Training loss:

$$G(\theta, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i \langle x_i, \theta \rangle) + \frac{1}{2}\sum_{k=1}^{p} e^{\lambda_k}\theta_k^2$$

- Validation loss: logistic loss

$$F(\theta, \lambda) = \frac{1}{m}\sum_{j=1}^{m} \log(1 + \exp(-y_i^{\mathrm{val}} \langle x_i^{\mathrm{val}}, \theta \rangle)$$

Hyperparameter selection on ℓ^2 regularized logistic regression



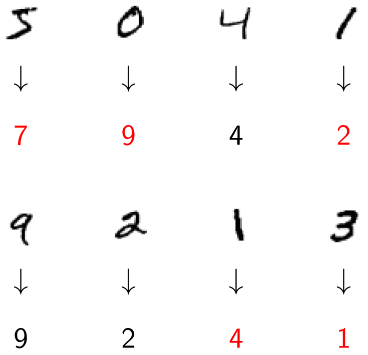
Data hyper-cleaning

Setting:

- Training samples with corrupted labels
- Dataset: MNIST
- Idea: Give more weight to uncorrupted data:

$$G(\theta, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \sigma(\lambda_i)\ell(\theta x_i, y_i) + C_r\|\theta\|^2$$

with $\sigma(\lambda_i) \in [0, 1]$.



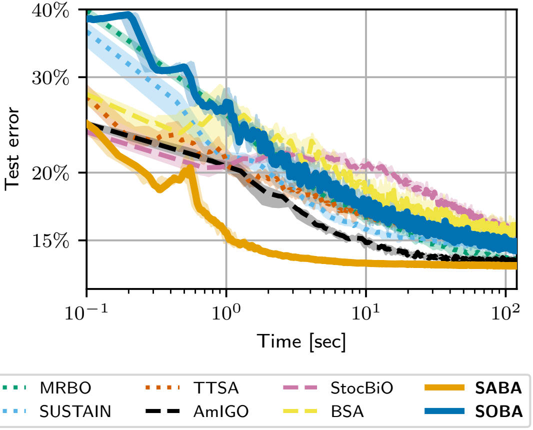
Data hyper-cleaning

Setting:

- We have a validation set with correct labels
- We can use bilevel optimization to tune λ:

$$\begin{cases} \min_{\lambda \in \mathbb{R}^n} F(\theta^*(\lambda), \lambda) = \frac{1}{m} \sum_{j=1}^m \ell(\theta^*(\lambda) x_j^{\mathrm{val}}, y_j^{\mathrm{val}}) \\ \theta^*(\lambda) \in \arg\min_{\theta \in \mathbb{R}^p} G(\theta, \lambda) = \frac{1}{n} \sum_{i=1}^n \sigma(\lambda_i) \ell(\theta x_i, y_i) + C_r \|\theta\|^2 \end{cases}$$

Data hyper-cleaning



Conclusion

Take home message

- It is possible to adapt any kind of single level stochastic optimizer to our framework.
- As in single level optimization, variance reduction allows to get convergence rate that matches rates of full batch gradient descent.

https://arxiv.org/abs/2201.13409