ICLR

SGDA with shuffling: faster convergence for nonconvex-PŁ minimax optimization

Hanseul Cho & Chulhee Yun

ICLR 2023

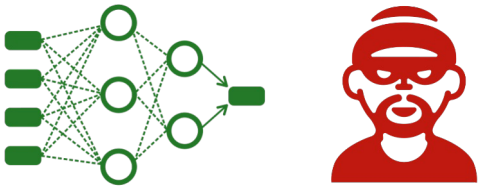
KAIST AI
Kim Jaechul Graduate School
OptiML
Optimization and
Machine Learning Laboratory

Finite-sum minimax optimization problem

- Minimax problem for an objective function $f(x;y)$ with n components:

$$\min_x \max_y f(x;y) := \frac{1}{n}\sum_{i=1}^{n} f_i(x;y)$$

- Examples:



  - Adversarial training: **Classifier** v.s. **Adversary**
  - AUC maximization, imitation learning, matrix game, etc.

Image by Guilherme Simoes from https://thenounproject.com/icon/robber-480223/
and by David Christensen from https://thenounproject.com/icon/neural-network-3339036/

SGDA : Stochastic Gradient Descent-Ascent

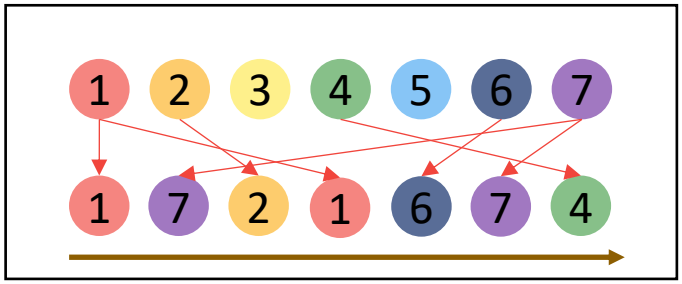
Initialize x_0, y_0
For $t := 0, \dots, T-1$:
Choose $i(t) \in \{1, \dots, n\}$
$x_{t+1} = x_t - \alpha \nabla_1 f_{i(t)}(x_t; y_t)$
$y_{t+1} = y_t + \beta \nabla_2 f_{i(t)}(x_t; y_t)$

How to choose the indices?

- Update x in direction of $f_{i(t)}$ decreases.
- Update y in direction of $f_{i(t)}$ increases.
- $\alpha, \beta > 0$: step sizes

How to choose $i(t)$ (1): with-replacement

- Uniformly randomly sample every iteration.
- **With-replacement** & independent sampling.



- (With-replacement) SGDA:

For $t := 1, \dots, T$ **do**:

Sample $i(t)$ **from** Uniform$(\{1, \dots, n\})$

$$x_t = x_{t-1} - \alpha \nabla_1 f_{i(t)}(x_{t-1}; y_{t-1})$$

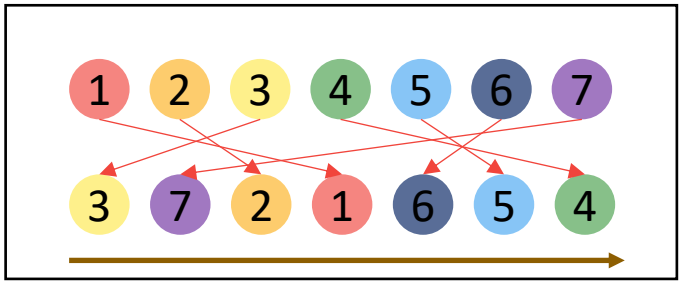
$$y_t = y_{t-1} + \beta \nabla_2 f_{i(t)}(x_{t-1}; y_{t-1})$$

👍 Easy to analyze, $\because \nabla f_{i(t)}(\cdot,\cdot)$ is an '**unbiased** estimator' of $\nabla f(\cdot,\cdot)$.

👎 Far from the practical implementation.

How to choose $i(t)$ (2): without-replacement

- Shuffle the order of indices $\{1, \dots, n\}$.
- Sequentially access them in the order.



- **Without**-replacement SGDA:

For $k := 1, \dots, K$ **do**:

Shuffle $(1, \dots, n) \to (i_1, i_2, \dots, i_n)$

For $j := 1, \dots, n$ **do**:

$$x_j^k = x_{j-1}^k - \alpha \nabla_1 f_{i_j}\left(x_{j-1}^k;\, y_{j-1}^k\right)$$

$$y_j^k = y_{j-1}^k + \beta \nabla_2 f_{i_j}\left(x_{j-1}^k;\, y_{j-1}^k\right)$$

Update $\left(x_0^{k+1}; y_0^{k+1}\right) \leftarrow \left(x_n^k; y_n^k\right)$

epoch

👎 Difficult to analyze. ∵ **biasness**: $\mathbb{E}_{i_j}\left[\nabla f_{i_j}\left(x_{j-1}^k;\, y_{j-1}^k\right)\right] \neq \nabla f\left(x_{j-1}^k;\, y_{j-1}^k\right)$ $(j \geq 2)$

👍 Practically widely used (epoch-based optimizers).

How to choose $i(t)$ (2-1): random reshuffling (RR)

- Shuffle the order of indices $\{1, \dots, n\}$, uniformly randomly.
- Sequentially access them in the order.

- SGDA with Random Reshuffling (**SGDA-RR**):

For $k := 1, \dots, K$ **do**:

Sample σ **from** Uniform$(\mathbb{S}_n)$

For $j := 1, \dots, n$ **do**:

$$x_j^k = x_{j-1}^k - \alpha \nabla_1 f_{\sigma(j)}\left(x_{j-1}^k;\, y_{j-1}^k\right)$$

$$y_j^k = y_{j-1}^k + \beta \nabla_2 f_{\sigma(j)}\left(x_{j-1}^k;\, y_{j-1}^k\right)$$

Update $\left(x_0^{k+1}; y_0^{k+1}\right) \leftarrow \left(x_n^k; y_n^k\right)$

epoch

*$\mathbb{S}_n$: the set of all permutations of {1, …, n}

👎 Difficult to analyze. ∵ **biasness**: $\mathbb{E}_\sigma\left[\nabla f_{\sigma(j)}\left(x_{j-1}^k;\, y_{j-1}^k\right)\right] \neq \nabla f\left(x_{j-1}^k;\, y_{j-1}^k\right)$ $(j \geq 2)$

👍 Practically widely used (epoch-based optimizers).

SGDA-RR > With-repl. SGDA?

- In minimization problems: SGD-RR > with-repl. SGD
  - … as long as K is large enough ($K \gtrsim \kappa$)[1]
  - Because without-replacement sampling has smaller variance than with-replacement sampling.
- In general, the convergence theory of minimization algorithms does NOT trivially extends to that of minimax algorithms.
  - *e.g.,* full-batch GDA may **diverge** even on convex-concave objective with a unique Nash equilibrium ($\min_x \max_y xy$).
- Existing analyses on SGDA usually assume *independent* component sampling.
- **Our question:**

Does **SGDA** with **random reshuffling (RR)** provably converge fast,

even on smooth nonconvex-nonconcave objective f with PŁ structures?

[1]Kwangjun Ahn, **Chulhee Yun**, and Suvrit Sra. SGD with shuffling: optimal rates without component convexity and large epoch requirements. NeurIPS 2020.

Assumptions for theoretical analysis (simplified)

A1	Each component f_i is L-smooth.	$\|\nabla f_i(x;y) - \nabla f_i(x';y')\| \le L\|(x;y) - (x';y')\|$
A2	Variance of component gradients is uniformly bounded.	$\frac{1}{n}\sum_{i=1}^{n}\|\nabla_j f_i(x,y) - \nabla_j f(x,y)\|^2 \le \boldsymbol{\tau^2}$
A3	"Primal function" is well-defined.	$\boldsymbol{\Phi(x) \coloneqq \max_y f(x;y)} \ge \Phi^* > -\infty$
A4	For each x, $-f(x;\cdot)$ is $\boldsymbol{\mu_2}$**-PŁ.**	$\|\nabla_2 f(x;y)\|^2 \ge 2\boldsymbol{\mu_2}(\Phi(x) - f(x;y))$
A5	The primal function $\boldsymbol{\Phi(\cdot)}$ is $\boldsymbol{\mu_1}$**-PŁ.**	$\|\nabla\, \Phi(x)\|^2 \ge 2\boldsymbol{\mu_1}(\Phi(x) - \Phi^*)$

📌 Two settings: **"Nonconvex-PŁ"**(A1...A4) ⊃ **"Primal-PŁ-PŁ"**(A1...A5)

📌 **All assumptions do NOT imply any convexity in x NOR concavity in y.**

Results

Comparison: **gradient complexity** of with-repl. SGDA v.s. SGDA-RR

	Algorithm	Nonconvex-PŁ	Primal-PŁ-PŁ
Previous art[2,3,4]	With-replacement SGDA	$\mathcal{O}(\varepsilon^{-4})$	$\mathcal{O}(\varepsilon^{-2})$
Ours	**SGDA-RR**	$\mathcal{O}(\varepsilon^{-3})$ **(Theorem 1)**	$\widetilde{\mathcal{O}}(\varepsilon^{-1})$ **(Theorem 2)**

[Theorem 1]

Suppose $f(x,y)$ is L-smooth and μ_2-PŁ in y. Let $\kappa_2 = \frac{L}{\mu_2}$. With appropriate choices of step sizes $\alpha > 0$ and $\beta = \Theta(\kappa^2)\alpha$, SGDA-RR satisfies $\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\nabla\Phi(x_k)\|^2 \le \varepsilon^2$ in $T = nK$ component gradient evaluations, such that

$$T \le \mathcal{O}\left(\frac{\kappa_2^2 L}{\varepsilon^2}\max\left\{n, \frac{\kappa_2\tau\sqrt{n}}{\varepsilon}\right\}\right).$$

*$\Phi(x) \coloneqq \max_y f(x;y)$. : 'Primal function'

Results

Comparison: **gradient complexity** of with-repl. SGDA v.s. SGDA-RR

	Algorithm	Nonconvex-PŁ	Primal-PŁ-PŁ
Previous art[2,3,4]	With-replacement SGDA	$\mathcal{O}(\varepsilon^{-4})$	$\mathcal{O}(\varepsilon^{-2})$
Ours	**SGDA-RR**	$\boldsymbol{\mathcal{O}(\varepsilon^{-3})}$ **(Theorem 1)**	$\boldsymbol{\tilde{\mathcal{O}}(\varepsilon^{-1})}$ **(Theorem 2)**

[Theorem 1]

Suppose $f(x,y)$ is L-smooth and μ_2-PŁ in y, and $\Phi(\cdot)$ is μ_1-PŁ. Let $\kappa_1 = \frac{L}{\mu_1}$ and $\kappa_2 = \frac{L}{\mu_2}$.

With appropriate choices of step sizes $\alpha > 0$ and $\beta = \Theta(\kappa^2)\alpha$, SGDA-RR satisfies $\mathbb{E}[V(x_n^K, y_n^K)] \leq \varepsilon^2$ in $T = nK$ component gradient evaluations, such that

$$T \leq \tilde{\mathcal{O}}\left(\kappa_1\kappa_2^2 \max\left\{n, \frac{\kappa_2\tau\sqrt{n}}{\varepsilon\sqrt{\mu_1}}\right\}\right).$$

*$\Phi(x) \coloneqq \max_y f(x;y)$. : 'Primal function'

$V(x,y) = 4(\Phi(x) - \Phi^*) + \big(\Phi(x) - f(x;y)\big)$: 'Potential function'. ($V(x,y) = 0 \Leftrightarrow$ minimax optimal)

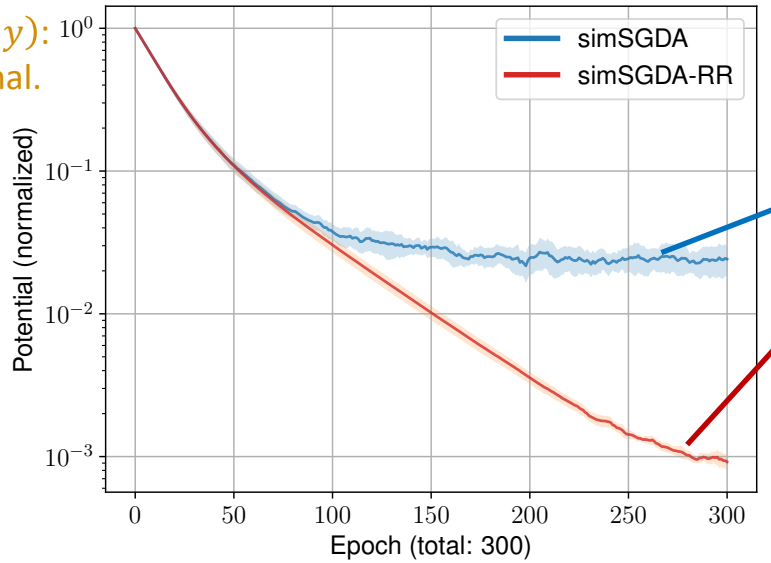
Experimental results

- A toy finite-sum quadratic game satisfying **Primal-PŁ-PŁ** setting:

$$\min_x \max_y f(x;y) = \frac{1}{2}x^\top Ax + x^\top By - \frac{1}{2}y^\top Cy = \frac{1}{n}\sum_{i=1}^n f_i(x;y),$$

$$\text{s.t.} \quad f_i(x;y) = \frac{1}{2}x^\top A_i x + x^\top B_i y - \frac{1}{2}y^\top C_i y + u_i^\top x - v_i^\top y$$

Convergence criterion $V(x,y)$:
Zero value ⇔ minimax optimal.



SGDA (with-replacement)

SGDA-RR : Faster!

Thank you!

- Please check out our paper to see more discussions, experiments, and proofs.

OpenReview
(paper)

Hanseul Cho
jhs4015@kaist.ac.kr

Chulhee Yun
chulhee.yun@kaist.ac.kr

KAIST AI
Kim Jaechul Graduate School

OptiML
Optimization and
Machine Learning Laboratory