Acceleration for Compressed Gradient Descent in Distributed and Federated Optimization

Zhize Li
King Abdullah University of Science and Technology (KAUST)
https://zhizeli.github.io

Joint work with Dmitry Kovalev (KAUST), Xun Qian (KAUST) and Peter Richtárik (KAUST)

ICML 2020

King Abdullah University of Science and Technology

ICML | 2020
Thirty-seventh International Conference on Machine Learning

Overview

Problem

Training distributed/federated learning models is typically performed by solving an optimization problem

$$\min_{x\in\mathbb{R}^d} \left\{ P(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x) \right\},$$

$f_i(x)$: loss function associated with data stored on node/device i
$\psi(x)$: regularization term (e.g., ℓ_1 regularizer $\|x\|_1$, ℓ_2 regularizer $\|x\|_2^2$ or indicator function $\mathcal{I}_{\mathcal{C}}(x)$ for some set $\mathcal{C}$)

Examples

$$\min_{x\in\mathbb{R}^d} \left\{ P(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x) \right\}$$

Each node/device i stores m data samples $\{(a_{i,j}, b_{i,j}) \in \mathbb{R}^{d+1}\}_{j=1}^m$

- **Lasso regression**: $f_i(x) = \frac{1}{m}\sum_{j=1}^m (a_{i,j}^T x - b_{i,j})^2, \;\; \psi(x) = \lambda\|x\|_1$
- **Logistic regression**: $f_i(x) = \frac{1}{m}\sum_{j=1}^m \log\left(1 + \exp(-b_{i,j} a_{i,j}^T x)\right)$
- **SVM**: $f_i(x) = \frac{1}{m}\sum_{j=1}^m \max\left(0, 1 - b_{i,j} a_{i,j}^T x\right), \;\; \psi(x) = \frac{\lambda}{2}\|x\|_2^2$

Goal

$$\min_{x\in\mathbb{R}^d}\left\{P(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x)\right\}$$

Goal: find an ϵ-solution (parameters) $\hat{x}$, e.g., $P(\hat{x}) - P(x^*) \le \epsilon$ or $\|\hat{x} - x^*\|_2^2 \le \epsilon$, where $x^* := \arg\min_{x\in\mathbb{R}^d} P(x)$.

Goal

$$\min_{x\in\mathbb{R}^d} \left\{ P(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x) \right\}$$

Goal: find an ϵ-solution (parameters) $\hat{x}$, e.g., $P(\hat{x}) - P(x^*) \le \epsilon$ or $\|\hat{x} - x^*\|_2^2 \le \epsilon$, where $x^* := \arg\min_{x\in\mathbb{R}^d} P(x)$.

For optimization methods:
Bottleneck: communication cost
Common strategy: Compress the communicated messages (lower communication cost in each iteration/communication round) and hope that this will not increase the total number of iterations/comm. rounds.

Related Work

- Several recent work show that the total communication complexity can be improved via compression. See e.g., QSGD [Alistarh et al., 2017], DIANA [Mishchenko et al., 2019], Natural compression [Horváth et al., 2019].

Related Work

- Several recent work show that the total communication complexity can be improved via compression. See e.g., QSGD [Alistarh et al., 2017], DIANA [Mishchenko et al., 2019], Natural compression [Horváth et al., 2019].
- However previous work usually lead to this kind of improvement:

Communication cost per iteration (- -) Iterations (+) $\Rightarrow$ Total (-)

'-' denotes **decrease**, '+' denotes **increase**

Related Work

- Several recent work show that the total communication complexity can be improved via compression. See e.g., QSGD [Alistarh et al., 2017], DIANA [Mishchenko et al., 2019], Natural compression [Horváth et al., 2019].
- However previous work usually lead to this kind of improvement:

Communication cost per iteration (- -) Iterations (+) $\Rightarrow$ Total (-)

'-' denotes **decrease**, '+' denotes **increase**

- In this work, we provide the **first** optimization methods provably **combining** the benefits of gradient **compression** and **acceleration**:

Communication cost per iteration (- -) Iterations (- -) $\Rightarrow$ Total (- - - -)

Single Device Setting

- First, consider the simple **single device (i.e. $n = 1$)** case:

$$\min_{x \in \mathbb{R}^d} f(x),$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth, and convex or μ-strongly convex.

Single Device Setting

- First, consider the simple **single device (i.e. $n = 1$)** case:

$$\min_{x \in \mathbb{R}^d} f(x),$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth, and convex or μ-strongly convex.

- f is L-smooth or has L-Lipschitz continuous gradient (for $L > 0$) if

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \tag{1}$$

and μ-strongly convex (for $\mu \geq 0$) if

$$f(x) - f(y) - \langle \nabla f(y), x - y \rangle \geq \frac{\mu}{2}\|x - y\|^2 \tag{2}$$

for all $x, y \in \mathbb{R}^d$. The $\mu = 0$ case reduces to the standard convexity.

Compressed Gradient Descent (CGD)

- Problem: $\min_{x\in\mathbb{R}^d} f(x)$

1) Given initial point x^0, step-size η
2) CGD update: $x^{k+1} = x^k - \eta\mathcal{C}(\nabla f(x^k))$, for $k \geq 0$

Compressed Gradient Descent (CGD)

- Problem: $\min_{x \in \mathbb{R}^d} f(x)$

1) Given initial point x^0, step-size η
2) CGD update: $x^{k+1} = x^k - \eta \mathcal{C}(\nabla f(x^k))$, for $k \geq 0$

Definition (Compression operator)

A randomized map $\mathcal{C} : \mathbb{R}^d \mapsto \mathbb{R}^d$ is an ω-compression operator if

$$\mathbb{E}[\mathcal{C}(x)] = x, \quad \mathbb{E}[\|\mathcal{C}(x) - x\|^2] \leq \omega \|x\|^2, \quad \forall x \in \mathbb{R}^d. \tag{3}$$

In particular, no compression ($\mathcal{C}(x) \equiv x$) implies $\omega = 0$.

Note that Condition (3) is satisfied by many practical compressions, e.g., random-k sparsification, (p, s)-quantization.

Accelerated Compressed Gradient Descent (ACGD)

Inspired by Nesterov's accelerated gradient descent (AGD) [Nesterov, 2004] and FISTA [Beck and Teboulle, 2009], here we propose the first accelerated compressed gradient descent (ACGD) method.

Our ACGD update:

1) $\boldsymbol{x^k = \alpha_k y^k + (1-\alpha_k) z^k}$

2) $y^{k+1} = x^k - \eta_k \mathcal{C}(\nabla f(x^k))$

3) $\boldsymbol{z^{k+1} = \beta_k \big(\theta_k z^k + (1-\theta_k) x^k\big) + (1-\beta_k)\big(\gamma_k y^{k+1} + (1-\gamma_k) y^k\big)}$

Convergence Results in Single Device Setting

Table: Convergence results (Iterations) for the single device ($n = 1$) case $\min_{x\in\mathbb{R}^d} f(x)$

Algorithm	μ-strongly convex f	convex f
Compressed Gradient Descent (CGD [Khirirat et al., 2018])	$O\left((1+\omega)\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$	$O\left((1+\omega)\frac{L}{\epsilon}\right)$
ACGD (this paper)	$O\left((1+\omega)\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$	$O\left((1+\omega)\sqrt{\frac{L}{\epsilon}}\right)$

- If no compression (i.e., $\omega = 0$): CGD recovers the results of vanilla (uncompressed) GD, i.e., $O(\frac{L}{\mu}\log\frac{1}{\epsilon})$ and $O(\frac{L}{\epsilon})$.

Convergence Results in Single Device Setting

Table: Convergence results (Iterations) for the single device ($n = 1$) case $\min_{x\in\mathbb{R}^d} f(x)$

Algorithm	μ-strongly convex f	convex f
Compressed Gradient Descent (CGD [Khirirat et al., 2018])	$O\left((1+\omega)\frac{L}{\mu}\log\frac{1}{\epsilon}\right)$	$O\left((1+\omega)\frac{L}{\epsilon}\right)$
ACGD (this paper)	$O\left((1+\omega)\sqrt{\frac{L}{\mu}}\log\frac{1}{\epsilon}\right)$	$O\left((1+\omega)\sqrt{\frac{L}{\epsilon}}\right)$

- If no compression (i.e., $\omega = 0$): CGD recovers the results of vanilla (uncompressed) GD, i.e., $O(\frac{L}{\mu}\log\frac{1}{\epsilon})$ and $O(\frac{L}{\epsilon})$.
- If compression parameter $\omega \le O(\sqrt{\frac{L}{\mu}})$ or $O(\sqrt{\frac{L}{\epsilon}})$:

Our ACGD enjoys the benefits of **compression** and **acceleration**, i.e., both the **communication cost per iteration** (compression) and the **total number of iterations** (acceleration) are smaller than that of GD.

Recall the Discussion in Related Work

- Previous work usually lead to this kind of improvement:

Communication cost per iteration (- -) Iterations (+) ⇒ Total (-)

'-' denotes **decrease**, '+' denotes **increase**

- In this work, we provide the **first** optimization methods provably **combining** the benefits of gradient **compression** and **acceleration**:

Communication cost per iteration (- -) Iterations (- -) ⇒ Total (- - - -)

Distributed Setting

Now, we consider the general distributed setting with n devices/nodes:

$$\min_{x\in\mathbb{R}^d}\left\{P(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x)\right\}.$$

The presence of multiple nodes ($n > 1$) and of the regularizer ψ poses additional challenges.

Distributed Setting

Now, we consider the general distributed setting with n devices/nodes:

$$\min_{x\in\mathbb{R}^d} \left\{ P(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x) \right\}.$$

The presence of multiple nodes ($n > 1$) and of the regularizer ψ poses additional challenges.

We propose a distributed variant of ACGD (called ADIANA) which can be seen as an accelerated version of DIANA [Mishchenko et al., 2019].

Accelerated DIANA (ADIANA)

Main update of our ADIANA:

1) $x^k = \theta_1 z^k + \theta_2 w^k + (1 - \theta_1 - \theta_2) y^k$

2i) all devices/nodes/machines **compress shifted local gradient** $\boldsymbol{\mathcal{C}_i^k(\nabla f_i(x^k) - h_i^k)}$ in parallel and send to the server

2ii) update local shift $\boldsymbol{h_i^{k+1} = h_i^k + \alpha \mathcal{C}_i^k(\nabla f_i(w^k) - h_i^k)}$

3) Aggregate received compressed gradient information

$$\boldsymbol{g^k = \frac{1}{n} \sum_{i=1}^{n} \mathcal{C}_i^k(\nabla f_i(x^k) - h_i^k) + h^k}$$

4) Perform a proximal update step $\boldsymbol{y^{k+1} = \mathrm{prox}_{\eta\psi}(x^k - \eta g^k)}$

5) $z^{k+1} = \beta z^k + (1 - \beta) x^k + \frac{\gamma}{\eta}(y^{k+1} - x^k)$

Convergence Results in Distributed Setting

Table: Convergence results (Iterations) for the general distributed case with n devices (the result in the case $n < \omega$ can be found in Table 2 of our paper)

Algorithm	In the case $\boldsymbol{n} \geq \boldsymbol{\omega}$ (lots of devices or low compression)
Distributed CGD (DIANA [Mishchenko et al., 2019])	$O\left(\left(\omega + \frac{L}{\mu}\right) \log \frac{1}{\epsilon}\right)$
ADIANA (this paper)	$O\left(\left(\omega + \sqrt{\frac{L}{\mu}} + \sqrt{\sqrt{\frac{\omega}{n}} \frac{\omega L}{\mu}}\right) \log \frac{1}{\epsilon}\right)$

Convergence Results in Distributed Setting

Table: Convergence results (Iterations) for the general distributed case with n devices (the result in the case $n < \omega$ can be found in Table 2 of our paper)

Algorithm	In the case $n \geq \omega$ (lots of devices or low compression)
Distributed CGD (DIANA [Mishchenko et al., 2019])	$O\left(\left(\omega + \frac{L}{\mu}\right) \log \frac{1}{\epsilon}\right)$
ADIANA (this paper)	$O\left(\left(\omega + \sqrt{\frac{L}{\mu}} + \sqrt{\sqrt{\frac{\omega}{n}} \frac{\omega L}{\mu}}\right) \log \frac{1}{\epsilon}\right)$

- Note that $\omega + \frac{L}{\mu} \geq 2\sqrt{\frac{\omega L}{\mu}}$ and $\sqrt{\frac{\omega}{n}} \leq 1$.

Convergence Results in Distributed Setting

Table: Convergence results (Iterations) for the general distributed case with n devices (the result in the case $n < \omega$ can be found in Table 2 of our paper)

Algorithm	In the case $\boldsymbol{n \geq \omega}$ (lots of devices or low compression)
Distributed CGD (DIANA [Mishchenko et al., 2019])	$O\left(\left(\omega + \frac{L}{\mu}\right) \log \frac{1}{\epsilon}\right)$
ADIANA (this paper)	$O\left(\left(\omega + \sqrt{\frac{L}{\mu}} + \sqrt{\sqrt{\frac{\omega}{n}} \frac{\omega L}{\mu}}\right) \log \frac{1}{\epsilon}\right)$

- Note that $\omega + \frac{L}{\mu} \geq 2\sqrt{\frac{\omega L}{\mu}}$ and $\sqrt{\frac{\omega}{n}} \leq 1$.
- If compression parameter $\omega \leq O\left(\min\left\{\sqrt{\frac{L}{\mu}}, n^{\frac{1}{3}}\right\}\right)$: Our ADIANA enjoys the benefits of **compression** and **acceleration**, i.e., **lower communication cost per iteration** (compression) and **fewer total number of iterations** (acceleration) $\sqrt{\frac{L}{\mu}} \log \frac{1}{\epsilon}$ vs. $\frac{L}{\mu} \log \frac{1}{\epsilon}$.

Experiments

We demonstrate the performance of our accelerated distributed method **ADIANA and previous methods** with **different compression operators** on the **regularized logistic regression** problem,

$$\min_{x\in\mathbb{R}^d}\left\{\frac{1}{n}\sum_{i=1}^{n}\log\left(1+\exp(-b_i a_i^T x)\right)+\frac{\lambda}{2}\|x\|^2\right\} \tag{4}$$

Experiments

We demonstrate the performance of our accelerated distributed method **ADIANA and previous methods** with **different compression operators** on the **regularized logistic regression** problem,

$$\min_{x \in \mathbb{R}^d} \left\{ \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + \exp(-b_i a_i^T x)\right) + \frac{\lambda}{2} \|x\|^2 \right\} \tag{4}$$

Compression operators: We adopt three compression operators: **random sparsification** (see e.g. [Stich et al., 2018]), **random dithering** (see e.g. [Alistarh et al., 2017]), and **natural compression** (see e.g. [Horváth et al., 2019]).

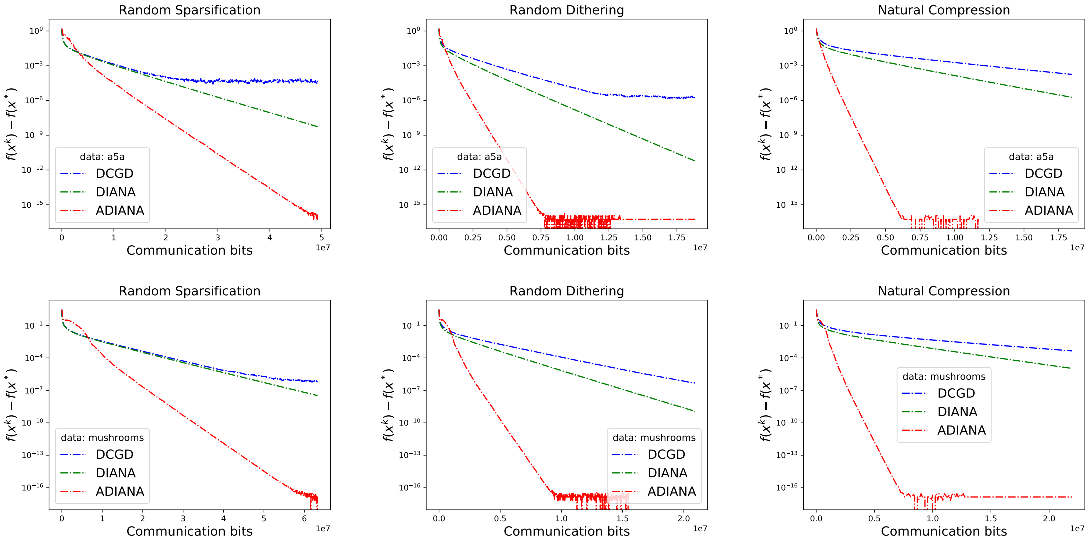

Figure: The communication complexity of **three different methods** for **three different compression operators** on `a5a` (top) and `mushrooms` (bottom) datasets.

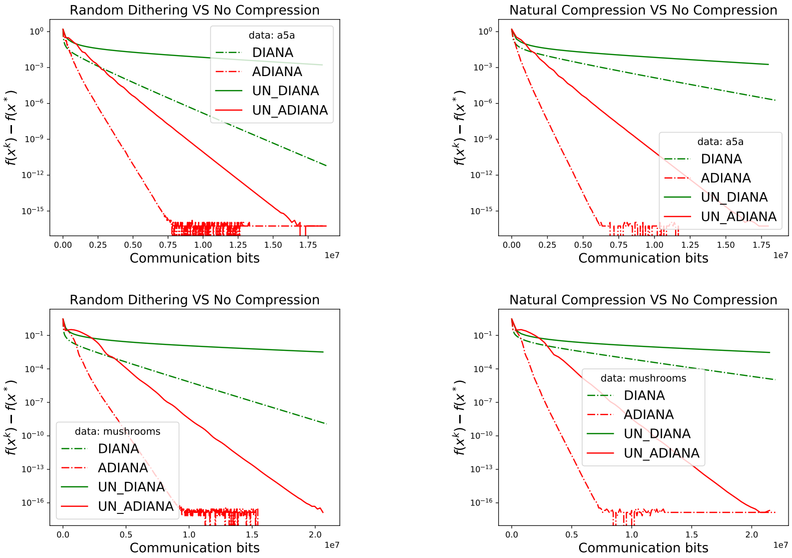

Figure: The communication complexity of **DIANA and ADIANA with and without compression** on a5a (top) and `mushrooms` (bottom) datasets.

Conclusion

- We provide the first **accelerated compressed gradient descent** methods (ACGD ($n = 1$) and ADIANA (general $n > 1$)) which combine the benefits of **compression** and **acceleration**.

- The experimental results validate our theoretical results and confirm the practical superiority of our accelerated methods.

Thanks!

Zhize Li

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, pages 1709–1720, 2017.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Samuel Horváth, Chen-Yu Ho, Ľudovít Horváth, Atal Narayan Sahu, Marco Canini, and Peter Richtárik. Natural compression for distributed deep learning. *arXiv preprint arXiv:1905.10988*, 2019.

Sarit Khirirat, Hamid Reza Feyzmahdavian, and Mikael Johansson. Distributed learning with compressed gradients. *arXiv preprint arXiv:1806.06573*, 2018.

Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč, and Peter Richtárik. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

Yurii Nesterov. *Introductory lectures on convex optimization: a basic course*. Kluwer Academic Publishers, 2004.

Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with memory. In *Advances in Neural Information Processing Systems*, pages 4447–4458, 2018.