# MARINA: Faster Non-Convex Distributed Learning with Compression


Eduard Gorbunov(MIPT) Konstantin Burlachenko(KAUST) Zhize Li (KAUST)
Peter Richtárik(KAUST)

Moscow Institute of Physics and Technologies and King Abdullah University of Science and Technology


## Problem setup

We consider **distributed optimization problems** in the following form:

$$\min_{x\in\mathbb{R}^d} f(x) := \frac{1}{n}\sum_{i=1}^{n} f_i(x)\,, \qquad (1)$$

- $n$ is the number of devices or workers
- $d$ is dimension of the optimization variable
- $f_i\colon \mathbb{R}^d \to \mathbb{R}$ is a differentiable loss accessible by worker $i$. It's gradient is a Lipschitz continuous.
- In the paper, we consider two cases:
  - $f_i(x) = \mathbb{E}_{i\sim D_i}[f_i(x)]$
  - $f_i(x) = \frac{1}{m}\sum_{j=1}^{m} f_{ij}(x)$
- The goal: find $\hat{x}$, such that $\mathbb{E}\left[\|\nabla f(\hat{x})\|^2\right] \le \varepsilon^2$

## Communication Bottleneck

In distributed training and federated learning, model updates have to be exchanged pretty often. Due to the size of the communicated messages for commonly considered deep learning models, this represents significant bottleneck of the whole optimization procedure. There are several ways how reduce the amount of data that has to be transmitted:

- Change topology of the network
- Do more work on each worker
- *Communication compression*

One can find a detailed summary of the most popular compression operators in (Safaryan et al., 2020; Beznosikov et al., 2020). In our work we use unbiased compressors.

## Unbiased Compression

A randomized mapping $\mathcal{C}\colon \mathbb{R}^d \to \mathbb{R}^d$ is an *unbiased compression operator (unbiased compressor)* if there exists $\omega \ge 0$ such that

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

## MARINA and VR-MARINA

**Input:**
Unbiased compressor $\mathcal{Q}$, starting point $x^0$, stepsize $\gamma$, probability $p \in (0,1]$, number of iterations $K$.

**Algorithms:**

**MARINA**

Master samples $c_k \sim \mathrm{Be}(p)$
Master broadcasts to all workers $g_k$
Workers in parallel: $x^{k+1} = x^k - \gamma g^k$
Workers compute local gradient estimator if $c_k = 1$

$$g_i^{k+1} = \nabla f_i(x^{k+1})$$

Workers compute local gradient estimator $c_k = 0$

$$g_i^{k+1} = g^k + \mathcal{Q}\left(\nabla f_i(x^{k+1}) - \nabla f_i(x^k)\right)$$

On Master $g^{k+1} = \frac{1}{n}\sum_{i=1}^{n} g_i^{k+1}$

**VR-MARINA**

Master samples $c_k \sim \mathrm{Be}(p)$
Master broadcasts to all workers $g_k$
Workers in parallel: $x^{k+1} = x^k - \gamma g^k$
Workers compute local gradient estimator if $c_k = 1$

$$g_i^{k+1} = \nabla f_i(x^{k+1})$$

Workers compute local gradient estimator $c_k = 0$

$$g_i^{k+1} = g^k + \mathcal{Q}\left(\frac{1}{b'}\sum_{j\in I_{i,k}}\left(\nabla f_{ij}(x^{k+1}) - \nabla f_{ij}(x^k)\right)\right)$$

On Master $g^{k+1} = \frac{1}{n}\sum_{i=1}^{n} g_i^{k+1}$

**Communication complexity (Corollaries 2.1, 3.2):**

$$K = \mathcal{O}\left(\frac{(1+w)\sqrt{1/n}}{\varepsilon^2}\right) \qquad K = \mathcal{O}\left(\frac{(1+\max\{w, \sqrt{(1+w)m}\})\sqrt{1/n}}{\varepsilon^2}\right), b' = 1$$

## Comparisons

To the best of our knowledge, the communication complexity bounds we prove for MARINA are strictly superior to those of all previous first order methods for non-convex optimization with the goal funding $\varepsilon$ stationary point including:

- Quantized Gradient Descend (analyzed by Khaled, et al, 2020) requires $\mathcal{O}\left(\frac{1+w}{\varepsilon^4 n}\right)$ rounds.
- DIANA (introduced by Mishchenko et al., 2019) requires $\mathcal{O}\left(\frac{(1+w)\sqrt{w/n}}{\varepsilon^2}\right)$ rounds.
- For another comparisons please check a paper.

## Contributions

1. A new distributed method supporting communication compression with a complete theory for all meta-parameters.
2. Significant improvement in the complexity bounds compate to the previous state of the art methods.
3. Numerical experiments has been implemented with a multi-node distributed execution in MPI4PY.

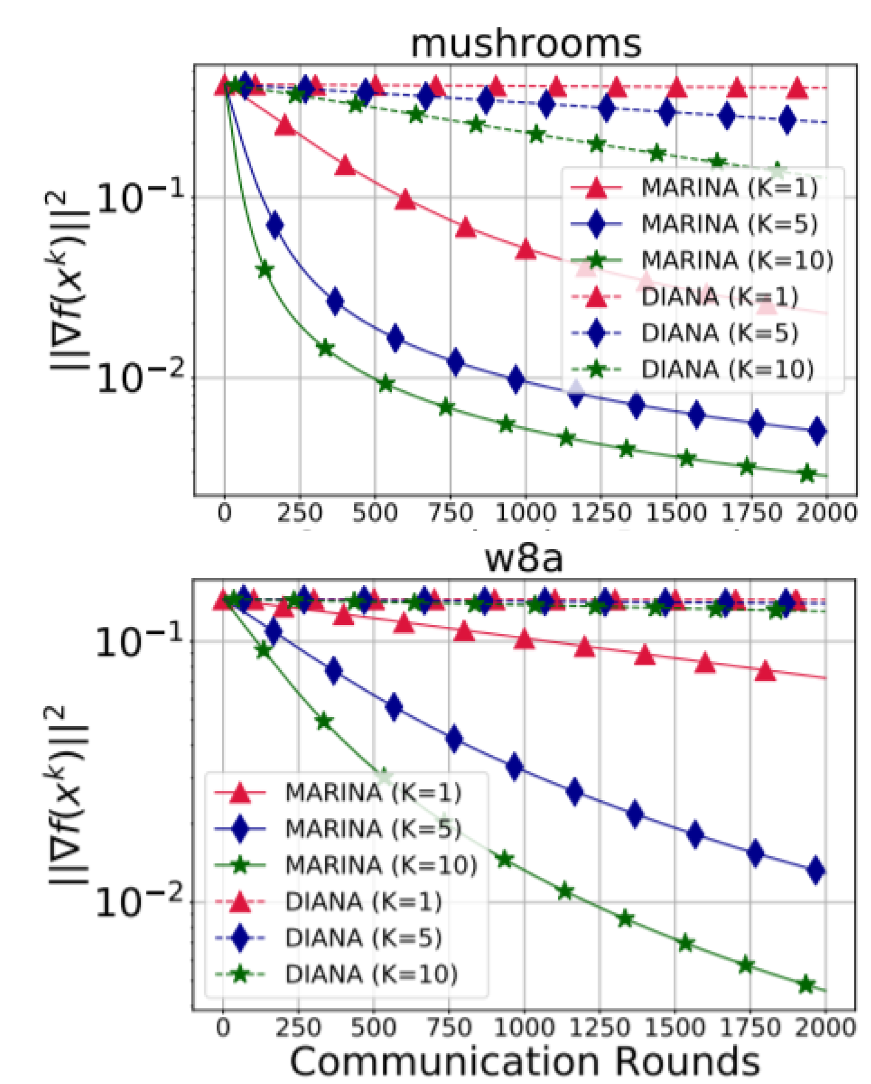
Experimental non-convex problem setup:

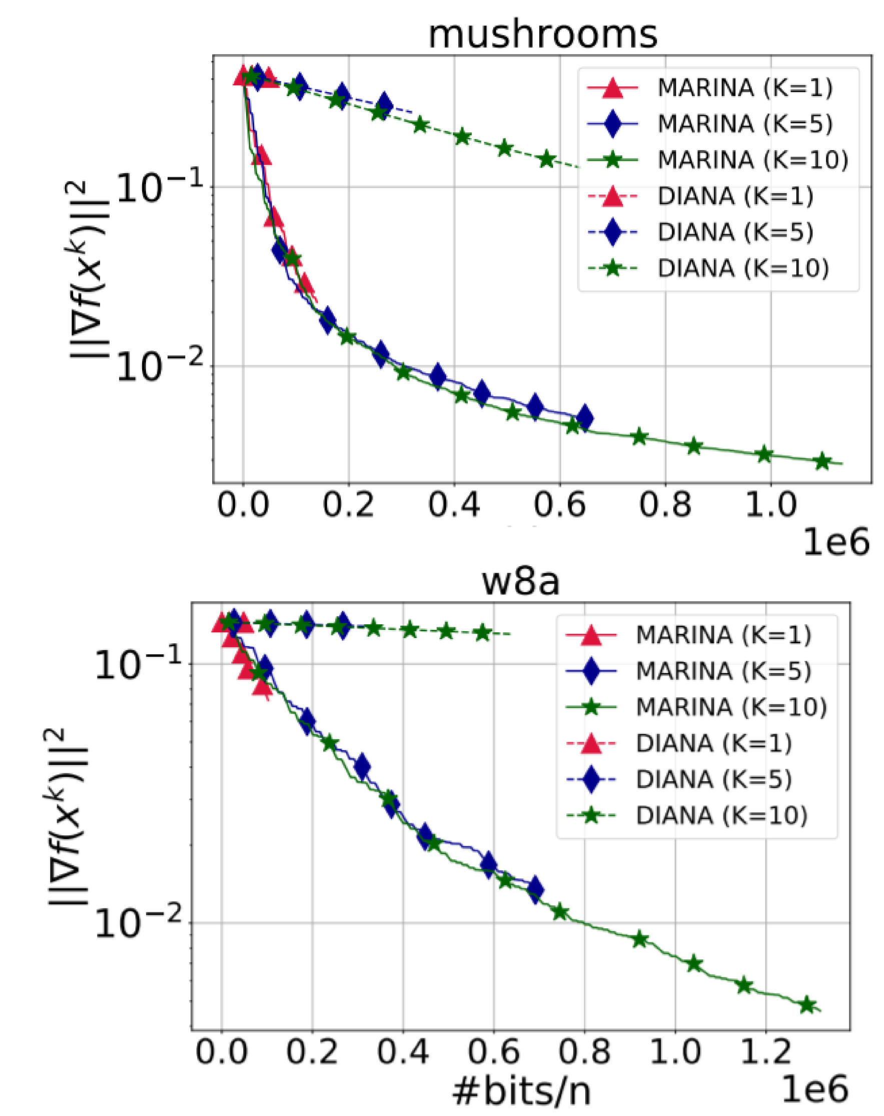
$$\min_{x\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} l(a_i^T x, b_i) + \frac{\lambda}{2}\|x\|^2$$

$$l(\tau, s) = \left(1 - \frac{1}{1+\exp(-\tau s)}\right)^2$$

Right now we are carrying experiments on CNNs.

## Experiments




## Reference to the paper

- Paper: https://arxiv.org/abs/2102.07845