# Federated Optimization Algorithms with Random Reshuffling and Gradient Compression


Abdurakhmon Sadiev[1] Grigory Malinovsky[1] Eduard Gorbunov[2] Igor Sokolov[1] Ahmed Khaled[3] Konstantin Burlachenko[1] Peter Richtárik[1]

[1]KAUST [2]Princeton University [3]MBZUAI


## The Problem

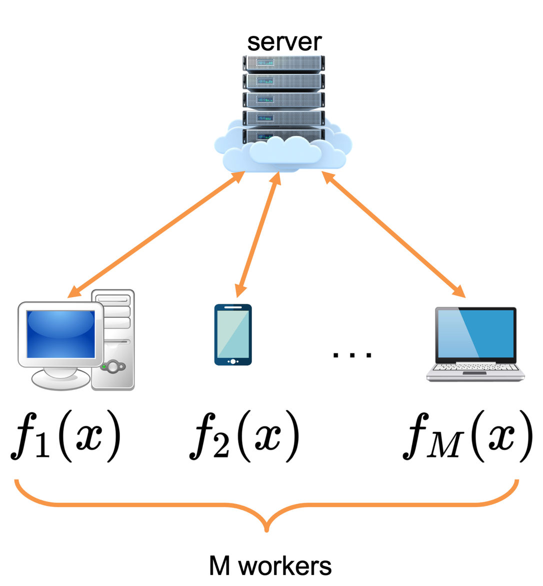
We study the optimization problem of Federated Learning (FL), which has the form

$$\min_{x\in\mathbb{R}^d}\left[f(x) := \frac{1}{M}\sum_{m=1}^{M} f_m(x)\right], \quad (1)$$

where $M$ is the number of clients/devices and each function

$$f_m(x) := \frac{1}{n}\sum_{i=1}^{n} f_m^i(x), \quad (2)$$

represents the loss on client $m$.




### Smoothness & Strong Convexity

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

$$\frac{\mu}{2}\|x-y\|^2 \le f_m^i(x) - f_m^i(y) - \left\langle \nabla f_m^i, x-y\right\rangle$$

$$\|\nabla f_m^i(x) - \nabla f_m^i(y)\| \le L_{i,m}\|x-y\|$$

for all $m \in [M]$ and $i \in [n]$

## Notation

- $[M] := \{1,\ldots,M\}$
- $L_{\max} := \max_{i,m} L_{i,m}$
- $D_h(x,y) := h(x) - h(y) - \langle \nabla h(y), x-y\rangle$
- $\sigma_{rad}^2 := \max_i \left\{\frac{1}{\gamma^2 M}\sum\limits_{m=1}^{M} \mathbb{E} D_{f_m^{\pi^i}}\left(x_\star^i, x_\star\right)\right\}$
- $\zeta_\star^2 := \frac{1}{M}\sum\limits_{m=1}^{M} \|\nabla f_m(x_\star)\|^2$
- $\sigma_\star^2 := \frac{1}{Mn}\sum\limits_{m=1}^{M}\sum_{i=1}^{n} \|\nabla f_m^i(x_\star) - \nabla f_m(x_\star)\|^2$

## Compressed Learning

### Unbiased Compressor

A compression operator is a randomized mapping $\mathcal{Q}:\mathbb{R}^d \to \mathbb{R}^d$ such that for some $\omega > 0$

$$\mathbb{E}\left[\mathcal{Q}(x)\right] = x, \qquad \mathbb{E}\left[\|\mathcal{Q}(x) - x\|^2\right] \le \omega\|x\|^2$$

for all $x \in \mathbb{R}^d$.

- Rand-$K$ sparsification operator is defined via

$$\mathcal{Q}(x) := \frac{d}{k}\sum_{i\in S} x_i e_i,$$

where $S \subseteq [d]$ is a subset of $[d]$ of cardinality $k$ chosen uniformly at random. This is unbiased compressor with $\omega := \frac{d}{k} - 1$.

### Main Goal

Design and analyze communication-efficient algorithms for Federated Learning using compression, random reshuffling, and/or local steps and improving upon existing algorithms both theoretically and practically.

## Algorithms & Communication complexities

### Q-RR

**Input:** $x_0$ – starting point, $\gamma > 0$ – stepsize
**for** $t = 0,1,\ldots,T-1$ **do**
  Receive $x_t$ from the server
  $x_{t,m}^0 = x_t$
  Sample random permutation of $[n]$: $\pi_m = (\pi_m^0,\ldots,\pi_m^{n-1})$
  **for** $i = 0,1,\ldots,n-1$ **do**
    **for** $m = 1,\ldots,M$ in parallel **do**
      Receive $x_t^i$ from the server
      Compute and send $\mathcal{Q}\left(\nabla f_m^{\pi_m^i}(x_t^i)\right)$
    $x_t^{i+1} = x_t^i - \gamma\frac{1}{M}\sum_{m=1}^{M}\mathcal{Q}\left(\nabla f_m^{\pi_m^i}(x_t^i)\right)$
    Send $x_t^{i+1}$ to the workers
  $x_{t+1} = x_t^n$
**Output:** $x_T$

Q-RR **[NEW]**:

$$\widetilde{\mathcal{O}}\left(\left(1+\frac{\omega}{M}\right)\frac{L_{\max}}{\widetilde{\mu}} + \frac{\omega\left(\zeta_\star^2+\sigma_\star^2\right)}{M\widetilde{\mu}^2\varepsilon} + \frac{\sigma_{rad}}{\sqrt{\widetilde{\mu}^3\varepsilon}}\right)$$

QSGD **[1]**:

$$\widetilde{\mathcal{O}}\left(\left(1+\frac{\omega}{M}\right)\frac{L_{\max}}{\mu} + \frac{\left(\omega\zeta_\star^2 + (1+\omega)\sigma_\star^2\right)}{M\mu^2\varepsilon}\right).$$

**Strengths:**
- Easy to implement;
- Memory friendly (does not require storing any additional vectors).

**Weaknesses:**
- Q-RR has no theoretical advantages over QSGD unless $\omega$ is very small.

### DIANA-RR

**Input:** $x_0$ – starting point, $\{h_{0,m}^i\}_{m,i=1,1}^{M,n}$ – initial shift-vectors, $\gamma > 0$ – stepsize, $\alpha > 0$ – stepsize for learning the shifts
**for** $t = 0,1,\ldots,T-1$ **do**
  Receive $x_t$ from the server
  $x_{t,m}^0 = x_t$
  Sample random permutation of $[n]$: $\pi_m = (\pi_m^0,\ldots,\pi_m^{n-1})$
  **for** $i = 0,1,\ldots,n-1$ **do**
    **for** $m = 1,2,\ldots,M$ in parallel **do**
      Receive $x_t^i$ from the server
      Compute and send $\mathcal{Q}\left(\nabla f_m^{\pi_m^i}(x_t^i) - h_{t,m}^{\pi_m^i}\right)$
      $\hat{g}_{t,m}^{\pi_m^i} = h_{t,m}^{\pi_m^i} + \mathcal{Q}\left(\nabla f_m^{\pi_m^i}(x_{t,m}^i) - h_{t,m}^{\pi_m^i}\right)$
      $h_{t+1,m}^{\pi_m^i} = h_{t,m}^{\pi_m^i} + \alpha\mathcal{Q}\left(\nabla f_m^{\pi_m^i}(x_{t,m}^i) - h_{t,m}^{\pi_m^i}\right)$
    $x_t^{i+1} = x_t^i - \gamma\frac{1}{M}\sum_{m=1}^{M}\hat{g}_{t,m}^{\pi_m^i}$
    Send $x_t^{i+1}$ to the workers
  $x_{t+1} = x_t^n$
**Output:** $x_T$

DIANA-RR **[NEW]**:

$$\widetilde{\mathcal{O}}\left(n(1+\omega) + \left(1+\frac{\omega}{M}\right)\frac{L_{\max}}{\widetilde{\mu}} + \frac{\sigma_{rad}}{\sqrt{\varepsilon\widetilde{\mu}^3}}\right)$$

DIANA **[1]**:

$$\widetilde{\mathcal{O}}\left(\left(1+\frac{\omega}{M}\right)\frac{L_{\max}}{\mu} + \frac{(1+\omega)\sigma_\star^2}{M\mu^2\varepsilon}\right).$$

**Strengths:**
- Unlike Q-RR, DIANA-RR does not have a $\widetilde{\mathcal{O}}\left(1/\varepsilon\right)$ term;
- Overall complexity of DIANA-RR improves over DIANA, since $\mathcal{O}\left(\sigma_{rad}/\sqrt{\varepsilon\widetilde{\mu}^3}\right)$ has a better dependence on $\varepsilon$ than $\mathcal{O}\left((1+\omega)\sigma_\star^2/(M\mu^2\varepsilon)\right)$.

**Weaknesses:**
- It can be memory expensive to maintain $\left\{h_{t,m}^i\right\}_{m\in[M], i\in[n]}$ shifts.

### Q-NASTYA

**Input:** $x_0$ – starting point, $\gamma > 0$ – local stepsize, $\eta > 0$ – global stepsize
**for** $t = 0,1,\ldots,T-1$ **do**
  **for** $m = 1,\ldots,M$ in parallel **do**
    Receive $x_t$ from the server
    $x_{t,m}^0 = x_t$
    Sample random permutation of $[n]$: $\pi_m = (\pi_m^0,\ldots,\pi_m^{n-1})$
    **for** $i = 0,1,\ldots,n-1$ **do**
      $x_{t,m}^{i+1} = x_{t,m}^i - \gamma\nabla f_m^{\pi_m^i}(x_{t,m}^i)$
    $g_{t,m} = \frac{1}{\gamma n}\left(x_t - x_{t,m}^n\right)$
    Send $\mathcal{Q}_t(g_{t,m})$ to the server
  $g_t = \frac{1}{M}\sum_{m=1}^{M}\mathcal{Q}_t(g_{t,m})$
  $x_{t+1} = x_t - \eta g_t$
  Send $x_{t+1}$ to the workers
$x_T = x_T^n$
**Output:** $x_T$

Q-NASTYA **[NEW]**:

$$\widetilde{\mathcal{O}}\left(\frac{L_{\max}}{\mu}\left(1+\frac{\omega}{M}\right) + \frac{\omega}{M}\frac{\zeta_\star^2}{\varepsilon\mu^3} + \sqrt{\frac{L_{\max}}{\varepsilon\mu^3}}\sqrt{\zeta_\star^2 + \frac{\sigma_\star^2}{n}}\right)$$

FedPAQ **[2]**:

$$\widetilde{\mathcal{O}}\left(\frac{L_{\max}}{\mu}\left(1+\frac{\omega}{M}\right) + \frac{\omega}{M}\frac{\sigma^2}{\mu^2\varepsilon} + \frac{\sigma^2}{M\mu^2\varepsilon}\right).$$

**Strengths:**
- Unlike FedCOM [4], Q-NASTYA provably works in a fully heterogeneous regime;
- Unlike FedPAQ, analysis of Q-NASTYA does not rely on the bounded variance assumption;
- Unlike FedCRR [3], Q-NASTYA converges for any $\omega \ge 0$;
- If $\omega$ is small, complexity of Q-NASTYA is superior to FedPAQ.

**Weaknesses:**
- In the big $\omega$ regime, Q-NASTYA has the same $\widetilde{\mathcal{O}}\left(1/\varepsilon\right)$ dependence as FedPAQ.

### DIANA-NASTYA

**Input:** $x_0$ – starting point, $\{h_{0,m}\}_{m=1}^{M}$ – initial shift-vectors, $\gamma > 0$ – local stepsize, $\eta > 0$ – global stepsize, $\alpha > 0$ – stepsize for learning the shifts
**for** $t = 0,1,\ldots,T-1$ **do**
  **for** $m = 1,\ldots,M$ in parallel **do**
    Receive $x_t$ from the server
    $x_{t,m}^0 = x_t$
    Sample random permutation of $[n]$: $\pi_m = (\pi_m^0,\ldots,\pi_m^{n-1})$
    **for** $i = 0,1,\ldots,n-1$ **do**
      $x_{t,m}^{i+1} = x_{t,m}^i - \gamma\nabla f_m^{\pi_m^i}(x_{t,m}^i)$
      $g_{t,m} = \frac{1}{\gamma n}\left(x_t - x_{t,m}^n\right)$
      Send $\mathcal{Q}_t\left(g_{t,m} - h_{t,m}\right)$ to the server
      $h_{t+1,m} = h_{t,m} + \alpha\mathcal{Q}_t\left(g_{t,m} - h_{t,m}\right)$
      $\hat{g}_{t,m} = h_{t,m} + \mathcal{Q}_t\left(g_{t,m} - h_{t,m}\right)$
    $h_{t+1} = h_t + \frac{\alpha}{M}\sum_{m=1}^{M}\mathcal{Q}_t\left(g_{t,m} - h_{t,m}\right)$
    $\hat{g}_t = h_t + \frac{1}{M}\sum_{m=1}^{M}\mathcal{Q}_t\left(g_{t,m} - h_{t,m}\right)$
    $x_{t+1} = x_t - \eta\hat{g}_t$
**Output:** $x_T$

DIANA-NASTYA **[NEW]**:

$$\widetilde{\mathcal{O}}\left(\omega + \frac{L_{\max}}{\mu}\left(1+\frac{\omega}{M}\right) + \sqrt{\frac{L_{\max}}{\varepsilon\mu^3}}\sqrt{\zeta_\star^2 + \frac{\sigma_\star^2}{n}}\right)$$

FedCRR-VR **[3]**:

$$\widetilde{\mathcal{O}}\left(\frac{(\omega+1)\left(1-\frac{1}{\kappa}\right)^n}{\left(1-\left(1-\frac{1}{\kappa}\right)^n\right)^2} + \frac{\sqrt{\kappa}\left(\zeta_\star + \sigma_\star\right)}{\mu\sqrt{\varepsilon}}\right)$$

**Strengths:**
- The complexity of DIANA-NASTYA is superior to both FedPAQ and Q-NASTYA;
- If $\kappa := \frac{L_{\max}}{\mu} \gg 1$, complexity of DIANA-NASTYA is better than for FedCRR-VR.

**Weaknesses:**
- Each worker $i$ has to maintain an additional vector state $h_{t,m}$, which causes an additional memory cost.

## Experiments

**Binary classification via Logistic regression.**

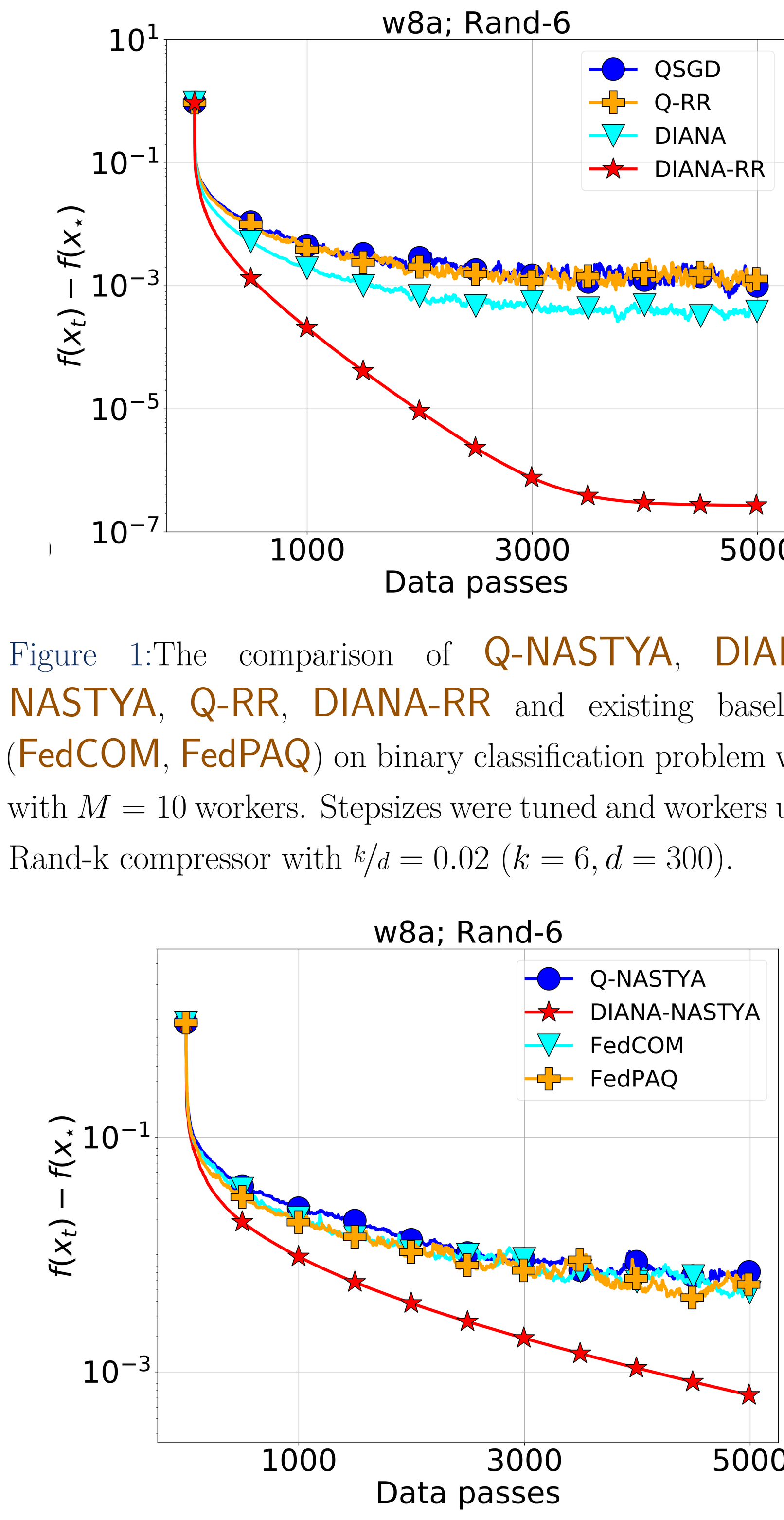



Figure 1: The comparison of Q-NASTYA, DIANA-NASTYA, Q-RR, DIANA-RR and existing baselines (FedCOM, FedPAQ) on binary classification problem with with $M = 10$ workers. Stepsizes were tuned and workers used Rand-k compressor with $k/d = 0.02$ ($k = 6, d = 300$).

**Image classification via `ResNet-18`.**

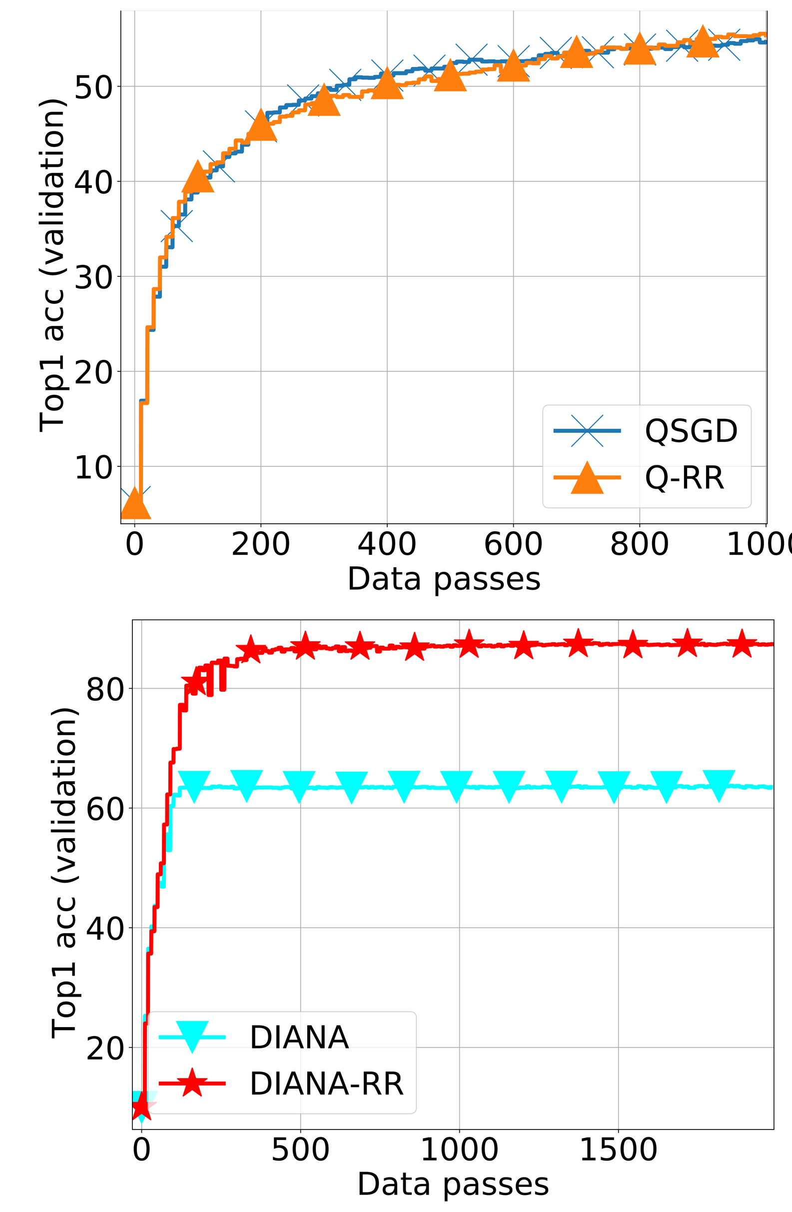



Figure 2: The comparison of Q-RR, QSGD, DIANA, and DIANA-RR on the task of training `ResNet-18` on `CIFAR-10` with $M = 10$ workers. Stepsizes were tuned and workers used Rand-k compressor with $k/d = 0.05$.

## References


[1] Gorbunov, et al. 'A Unified Theory of SGD: Variance Reduction, Sampling, Quantization and Coordinate Descent.' AISTATS, 2020.

[2] Reisizadeh, et al. 'Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization.' AISTATS, 2020.

[3] Malinovsky, et al. 'Federated random reshuffling with compression and variance reduction.' arXiv, 2022.

[4] Haddadpour, et al. 'Federated learning with compression: Unified analysis and sharp guarantees.' AISTATS, 2021.


arXiv GitHub