Similarity, Compression and Local Steps: Three Pillars of Efficient Communications for Distributed Variational Inequalities

Aleksandr Beznosikov [1,2,3,4] Martin Takáč [5] Alexander Gasnikov [3,2,6]

[1]Innopolis University [2]Skoltech [3]Moscow Institute of Physics and Technology [4]Yandex [5]Mohamed bin Zayed University of Artificial Intelligence [6] IITP RAS

Distributed problem

• **Variational inequality** (VIPs) :

$$\text{Find } z^* \in \mathcal{Z} \text{ such that } \langle F(z^*), z - z^* \rangle \geq 0 \quad \forall z \in \mathcal{Z},$$

where $F : \mathcal{Z} \to \mathbb{R}^d$ is an operator, and $\mathcal{Z} \subseteq \mathbb{R}^d$ is a convex set.

• Training data describing F is **distributed** across n devices/nodes:

$$F(z) = \frac{1}{n}\sum_{i=1}^{n} F_i(z),$$

where $F_i : \mathcal{Z} \to \mathbb{R}^d$ for all $i \in [n] := \{1, 2, \ldots, n\}$.

Examples

• Minimization problem:

$$\min_{z \in \mathcal{Z}} f(z),$$

for which $F(z) := \nabla f(z)$.

• Saddle point problem:

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} g(x_1, x_2),$$

for which $F(z) := (\nabla_x g(x, y), -\nabla_y g(x, y))$.

Communication bottleneck

Fight for efficient communications:

• **Compression** – send small packages.
Examples: QGD, DIANA, MARINA, EF, EF21

• **Local steps** – communicate less often
Examples: Local GD/FedAvg, FedProx, Scaffold, Scaffnew

• **Similarity** – use the closeness of local data
Examples: DANE, DiSCO, SPAG, SONATA

Setting

• Each operator F_i is L-Lipschitz continuous on $\mathcal{Z}$, i.e. for all $u, v \in \mathcal{Z}$ we have

$$\|F_i(u) - F_i(v)\| \leq L\|u - v\|.$$

• The operator F is μ-strongly monotone on $\mathcal{Z}$, i.e. for all $u, v \in \mathcal{Z}$ we have

$$\langle F(u) - F(v), u - v \rangle \geq \mu\|u - v\|^2.$$

Each operator F_i is monotone on $\mathcal{Z}$, i.e. $\mu = 0$.

• The operators $\{F_i\}$ is δ-related in mean on $\mathcal{Z}$. It means that for any j operators $\{F_i - F_j\}$ and for all $u, v \in \mathcal{Z}$ we have

$$\frac{1}{n}\sum_{i=1}^{n} \|F_i(u) - F_j(u) - F_i(v) + F_j(v)\|^2 \leq \delta^2\|u - v\|^2.$$

Hessian similarity: if the data is uniformly distributed between devices then $\delta = \tilde{\mathcal{O}}(L/\sqrt{b})$, where b is the number of local data points on each of the devices.
Minimization: $\|\nabla^2 f_j(z) - \nabla^2 f_i(z)\| \leq \delta$,
SPP: $\|\nabla^2_{xx} f_j(x, y) - \nabla^2_{xx} f_i(x, y)\| \leq \delta$,
$\|\nabla^2_{xy} f_j(x, y) - \nabla^2_{xy} f_i(x, y)\| \leq \delta$, $\|\nabla^2_{yy} f_j(x, y) - \nabla^2_{yy} f_i(x, y)\| \leq \delta$.

Main Contributions

◇ **Three Pillars Algorithm.** We present a new method `Three Pillars Algorithm` that combines three approaches: compression, similarity, and local steps for the efficient solution of distributed VI and SPP.

◇ **The best communication complexity.** We analyze the convergence of the new distributed algorithm for smooth strongly monotone VI and strongly convex-strongly concave SPP under similarity assumption. Our algorithm has better communication complexity than all available competitors.

◇ **Extension with partial participation.** We present a modified version of our first method. Instead of compression, one device is selected to send an uncompressed message.

◇ **Lower bounds.** To demonstrate the optimality of our proposed approaches, we establish lower bounds for the communication complexities.

◇ **Extension with stochastic local computations.** Motivated by applications where computation is expensive, we present a modification of the main algorithm that emphasizes the use of stochastic operators in local computations.

Table: Summary of the results on the number of transmitted information in different approaches to **communication bottleneck for distributed VI/SPP**.

Method	Approach	Communication complexity
`Extra Gradient`		$\mathcal{O}\left(\frac{L}{\mu}\log\frac{1}{\varepsilon}\right)$
`Local SGDA`	local steps	$\mathcal{O}\left(\frac{L^{2/5}n^{2/5}}{\mu^{2/5}\varepsilon^{1/5}} + \frac{L\zeta}{\mu^{3/2}\sqrt{\varepsilon}}\right)$
`FedAvg-S`	local steps	$\mathcal{O}\left(\frac{L^2}{\mu^2}\log\frac{1}{\varepsilon} + \frac{L\zeta}{\mu^2\sqrt{\varepsilon}}\right)$
`SCAFFOLD-S`	local steps	$\mathcal{O}\left(\frac{L^2}{\mu^2}\log\frac{1}{\varepsilon}\right)$
`SCAFFOLD-Catalyst-S`	local steps	$\mathcal{O}\left(\frac{L}{\mu}\log^2\frac{1}{\varepsilon}\right)$
`ESTVGM`	local steps	$\mathcal{O}\left(\frac{L}{\mu}\log\frac{1}{\varepsilon} + \frac{L\zeta}{\mu^2\sqrt{\varepsilon}}\right)$
`SMMDS`	similarity local steps	$\mathcal{O}\left(\left[1 + \frac{\delta}{\mu}\right]\log\frac{1}{\varepsilon}\right)$
`MASHA`	compression	$\mathcal{O}\left(\frac{L}{\sqrt{n\mu}}\log\frac{1}{\varepsilon}\right)$
`Optimistic MASHA`	similarity compression	$\mathcal{O}\left(\left[\frac{L}{n\mu} + \frac{\delta}{\sqrt{n\mu}}\right]\log\frac{1}{\varepsilon}\right)$
`Accelerated Extra Gradient`	similarity local steps	$\mathcal{O}\left(\left[1 + \frac{\delta}{\mu}\right]\log\frac{1}{\varepsilon}\right)$
Algorithm 1 (this paper)	similarity local steps compression	$\mathcal{O}\left(\left[1 + \frac{\delta}{\sqrt{n\mu}}\right]\log\frac{1}{\varepsilon}\right)$
Algorithm 2 (this paper)	similarity local steps partial participation	$\mathcal{O}\left(\left[1 + \frac{\delta}{\sqrt{n\mu}}\right]\log\frac{1}{\varepsilon}\right)$
Lower bound		$\Omega\left(\left[1 + \frac{\delta}{\mu}\right]\log\frac{1}{\varepsilon}\right)$ (1)
Lower bound (this paper)	similarity local steps compression	$\Omega\left(\left[1 + \frac{\delta}{\sqrt{n\mu}}\right]\log\frac{1}{\varepsilon}\right)$
Lower bound (this paper)	similarity local steps partial participation	$\Omega\left(\left[1 + \frac{\delta}{\sqrt{n\mu}}\right]\log\frac{1}{\varepsilon}\right)$

(1) lower bound is deterministic and does not take into account the possibility of compression and partial participation. *Notation:* ζ = heterogeneity parameter, ε = precision of the solution.

Algorithm

Algorithm 1 `Three Pillars Algorithm`

Parameters: stepsizes γ and η, momentum τ, probability $p \in (0; 1]$, number of local steps H;
Initialization: Choose $z^0 = m^0 = (x^0, y^0) \in \mathcal{Z}$;

1: **for** $k = 0, 1, \ldots, K-1$ **do**
2: Server takes $u_0^k = z^k$;
3: **for** $t = 0, 1, \ldots, H-1$ **do**
4: Server computes $u_{t+1/2}^k = \text{proj}_{\mathcal{Z}}[u_t^k - \eta(F_1(u_t^k) - F_1(m^k) + F(m^k) + \frac{1}{\gamma}(u_t^k - z^k - \tau(m^k - z^k)))]$;
5: Server updates $u_{t+1}^k = \text{proj}_{\mathcal{Z}}[u_t^k - \eta(F_1(u_{t+1/2}^k) - F_1(m^k) + F(m^k) + \frac{1}{\gamma}(u_{t+1/2}^k - z^k - \tau(m^k - z^k)))]$;
6: **end for**
7: Server broadcasts u_H^k and $F_1(u_H^k)$ to devices;
8: Devices in parallel compute $Q_i(F_i(m^k) - F_1(m^k) - F_i(u_H^k) + F_1(u_H^k))$ and send to server;
9: Server updates $z^{k+1} = \text{proj}_{\mathcal{Z}}[u_H^k + \gamma \cdot \frac{1}{n}\sum_{i=1}^n Q_i(F_i(m^k) - F_1(m^k) - F_i(u_H^k) + F_1(u_H^k))]$;
10: Server updates $m^{k+1} = \begin{cases} z^k, & \text{with probability } p, \\ m^k, & \text{with probability } 1-p, \end{cases}$;
11: **if** $m^{k+1} = z^k$ **then**
12: Server broadcasts m^{k+1} to devices;
13: Devices in parallel compute $F_i(m^k)$ and send to server;
14: Server computes $F(m^{k+1}) = \frac{1}{n}\sum_{i=1}^n F_i(m^{k+1})$;
15: **end if**
16: **end for**

• **Local problem.** We can rewrite the original distributed problem as a composite problem of two terms: $\frac{1}{n}\sum_{i=1}^n f_i(x) = [f_1(x)] + [\frac{1}{n}\sum_{i=1}^n f_i(x) - f_1(x)]$. The proximal method works well for composite problems with proximal-friendly functions (e.g, simple ℓ_2 regularizer) where the calculation of the proximal operator is for free, but, in the general case, the proximal operator is calculated by an auxiliary method with another iteration loop inside the main algorithm. The key idea in to use $f_1(x)$ for the proximal operator. This implies the need of additional solving $\arg\min_x\{f_1(x) + \lambda\|x - x_s\|\}$, where x_s is some point. Note that no commutation is needed to calculate the proximal operator as to solution needs access only to f_1 and can be solved locally. To extend the idea into VI , we use the following (see the loop in line 3):

$$\text{Find } \hat{u}^k \in \mathcal{Z} \text{ such that } \langle G(\hat{u}^k), z - \hat{u}^k \rangle \geq 0, \quad \forall z \in \mathcal{Z}$$

with $G(z) = F_1(z) + \frac{1}{\gamma}(z - v^k)$ and $v^k = z^k + \tau(m^k - z^k) - \gamma \cdot (F(m^k) - F_1(m^k))$.

• **Basic outer method.** We need a basic method for solving the composite VI. We selects the `Tseng's` method for this task.

• **Local method.** Since the subproblem in the inner loop from line 3 in Algorithm is a VI with a strongly monotone and Lipschitz continuous operator, then it can be solved using the `Extra Gradient` method.

• **Use of compression.** In order to use compression in the method, an additional technique is also required. Following, we take the idea of basing on the variance reduction technique. We introduce a reference sequence $\{m^k\}_{k\geq 0}$. At point m^k, we need to know the full values of operator F. When m^k is updated (line 10), we transfer the full operators without compression (lines 12-14). If probability p is small, m^k is rarely modified and hence condition from line 11 is satisfied with low probability.

• **Compression operators.** We use special compression to take into account similarity.

Permutation compressors

For $d \geq n$. Assume that $d \geq n$ and $d = qn$, where $q \geq 1$ is an integer. Let $\pi = (\pi_1, \ldots, \pi_d)$ be a random permutation of $\{1, \ldots, d\}$. Then for all $u \in \mathbb{R}^d$ and each $i \in \{1, 2, \ldots, n\}$ we define $Q_i(u) = n \cdot \sum_{j=q(i-1)+1}^{qi} u_{\pi_j} e_{\pi_j}$.

• **Summary.** We use `Tseng's` method for the composite VI, inaccurately computing the resolvent (proximal operator) with the `Extra Gradient` method. We use a variance reduction technique for `Tseng's` method, remade into a compression technique in distributed settings, and choose a special compression for similarity setting.

Convergence

Theorem

Let $\{z^k\}_{k\geq 0}$ denote the iterates of the algorithm with PermK compressors for solving the distributed VI problem, which satisfies assumptions. Then, if we choose the stepsizes $\gamma = \tilde{\mathcal{O}}(\min\{\frac{p}{\mu}, \frac{\sqrt{p}}{\delta}, \frac{H}{L}\})$, $\eta = \mathcal{O}((L + \frac{1}{\gamma})^{-1})$ and the momentum $\tau = p$ then we have the convergence guarantee

$$\mathbb{E}[\|z^K - z^*\|^2] \leq 2\left(1 - \frac{\gamma\mu}{2}\right)^K \|z^0 - z^*\|^2.$$

• **Optimal choice of p.** In Algorithm, one mandatory communication round with compression occurs (line 8) and possibly one more (without compression) with probability p (line 13). Permutation compressor compress by a factor of n; hence each iteration requires $\mathcal{O}\left(\frac{1}{n} + p\right)$ data transfers from devices to the server on average. The optimal choice of p is $\frac{1}{n}$, as stated in the corollary.

Corollary

Under the conditions of Theorem, the following number of the outer iterations is needed to achieve the accuracy ε (in terms of $\mathbb{E}[\|z - z^*\|^2] \lesssim \varepsilon$) by Algorithm with $p = \frac{1}{n}$:

$$\mathcal{O}\left(\left[n + \frac{\delta\sqrt{n}}{\mu} + \frac{L}{\mu H}\right]\log\frac{\|z^0 - z^*\|^2}{\varepsilon}\right).$$

• **Optimal choice of H.** It is clear from the corollary that the number of local H steps can improve the number of communications, but there is a limit beyond which more iterations are not useful. Taking $H = \lceil\frac{L}{\delta\sqrt{n}}\rceil$ is optimal.

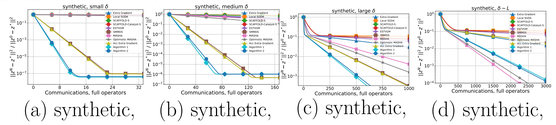
• **"Break" lower bounds.** H from the previous paragraph yields estimates of $\tilde{\mathcal{O}}(n + \delta\sqrt{n}/\mu)$ and $\tilde{\mathcal{O}}(1 + \delta/\sqrt{n}\mu)$ for the number of communications and transmitted information, respectively. These results are better than any existing methods and even superior to lower bounds for deterministic methods.

Experiments

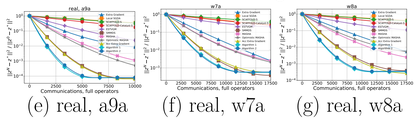
We conduct experiments on the robust linear regression problem. This problem is defined as

$$\min_{w\in\mathbb{R}^d} \max_{\|r_i\|\leq D} \frac{1}{2N}\sum_{i=1}^{N}(w^T(x_i + r_i) - y_i)^2 + \frac{\lambda}{2}\|w\|^2 - \frac{\beta}{2}\|r\|^2,$$

where w are model weights, $\{x_i, y_i\}_{i=1}^N$ is the training dataset and r is artificially added noise. We consider a network with $n = 25$ devices and two types of datasets: synthetic and real. Synthetic data allows us to control the factor δ, which measures statistical similarity of functions over different nodes. Our algorithms are compared with methods from Table.



(a) synthetic, small δ (b) synthetic, medium δ (c) synthetic, large δ (d) synthetic, $\delta \sim L$



(e) real, a9a (f) real, w7a (g) real, w8a

Figure: Comparison of state-of-the-art methods for distributed VI. The comparison is made on synthetic datasets with small, medium, and large δ, as well as the real datasets a9a, w7a, w8a from LibSVM. The x-axis denotes the number of full operators transmitted by one of the devices.