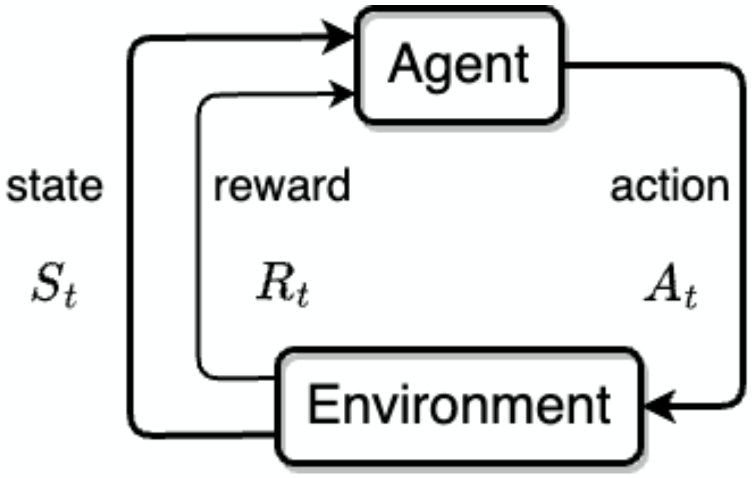
Improved Communication Efficiency in Federated Natural Policy Gradient via ADMM-based Gradient Updates

Guangchen Lan, Han Wang, James Anderson, Christopher Brinton, Vaneet Aggarwal

NeurIPS 2023

lan44@purdue.edu

Reinforcement Learning



trajectories $\tau = (s_0, a_0, r_0, s_1, a_1, r_1 \cdots)$

Challenge:

- High volumes of data (trajectories) are required.

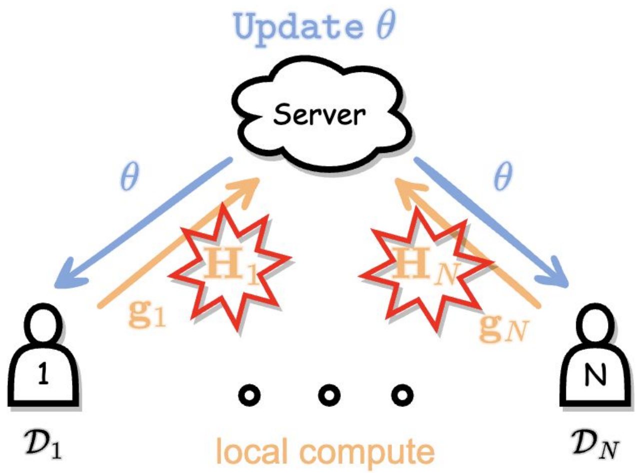
Conventional approach:

Multi-agent solution: Transmit raw **data** collected locally by different agents to a central serve.

- Communication overhead;
- Long delays;
- Privacy and legal issues.

Federated Learning & FedNPG

Federated Natural Policy Gradient (FedNPG)



(a) FedNPG

$$\{\mathbf{H}_i \in \mathbb{R}^{d\times d},\ \mathbf{g}_i \in \mathbb{R}^d\}_{i=1}^N$$

Standard Federated Learning Approach:

- Instead of transmitting raw trajectories, model parameters are transmitted in federated learning.

Challenges in Standard FedNPG:

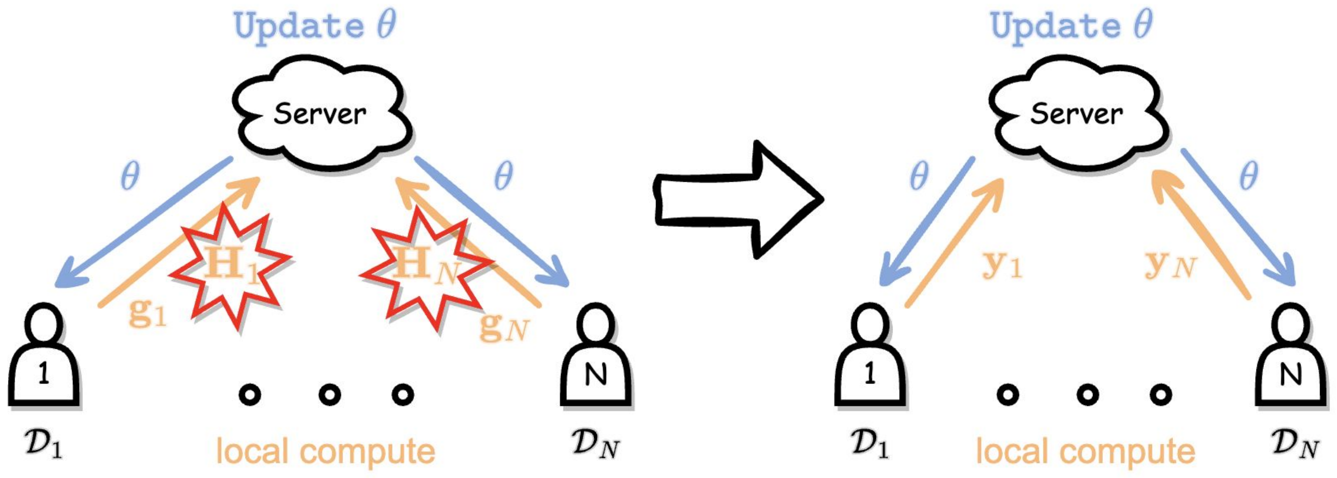
- **Communication overhead**: In NPG methods, the 2nd-order information with size ***O*(*d*²)** needs to be transmitted in each iteration. Thus, it is not scalable.

Q: Can we reduce the communication complexity for 2nd-order FedNPG approach while maintaining performance guarantees?

A: Yes! Use FedNPG-ADMM!

FedNPG-ADMM

Alternating Direction Method of Multipliers (ADMM)



(a) FedNPG (b) FedNPG-ADMM

Table 1: Complexity comparison in each agent.

	NPG	FedNPG	FedNPG-ADMM
Sample complexity	$\mathcal{O}(\frac{1}{(1-\gamma)^6\epsilon^2})$	$\mathcal{O}(\frac{1}{(1-\gamma)^6 N\epsilon^2})$	$\mathcal{O}(\frac{1}{(1-\gamma)^6 N\epsilon^2})$
Communication complexity	-	$\mathcal{O}(\frac{d^2}{(1-\gamma)^2\epsilon})$	$\mathcal{O}(\frac{d}{(1-\gamma)^2\epsilon})$

FedNPG-ADMM

Key Steps

$$\theta \leftarrow \theta + \sqrt{\frac{2N\delta}{(\sum_{i=1}^N \mathbf{g}_i)^\top \underline{(\sum_{i=1}^N \mathbf{H}_i)^{-1} \sum_{i=1}^N \mathbf{g}_i}}} \underline{(\sum_{i=1}^N \mathbf{H}_i)^{-1} \sum_{i=1}^N \mathbf{g}_i}.$$ [1,2]

$$\{\mathbf{H}_i \in \mathbb{R}^{d\times d},\ \mathbf{g}_i \in \mathbb{R}^d\}_{i=1}^N$$

$$(\sum_{i=1}^N \mathbf{H}_i)^{-1} \sum_{i=1}^N \mathbf{g}_i = \arg\min_{\mathbf{y}} \ \frac{1}{2}\mathbf{y}^\top (\sum_{i=1}^N \mathbf{H}_i)\mathbf{y} - \mathbf{y}^\top \sum_{i=1}^N \mathbf{g}_i.$$

Equivalent problems

$$\min_{\mathbf{y},\{\mathbf{y}_i\}_{i=1}^N} \sum_{i=1}^N \left(\frac{1}{2}\mathbf{y}_i^\top \mathbf{H}_i \mathbf{y}_i - \mathbf{y}_i^\top \mathbf{g}_i + \frac{\rho}{2}\|\mathbf{y}_i - \mathbf{y}\|^2\right)$$
$$\text{s.t. } \mathbf{y} = \mathbf{y}_i, \quad i = 1, \cdots, N,$$

$$\mathcal{L}(\mathbf{y}, \{\mathbf{y}_i\}_{i=1}^N, \{\lambda_i\}_{i=1}^N) = \sum_{i=1}^N \left(\frac{1}{2}\mathbf{y}_i^\top \mathbf{H}_i \mathbf{y}_i - \mathbf{y}_i^\top \mathbf{g}_i + \frac{\rho}{2}\|\mathbf{y}_i - \mathbf{y}\|^2 + \langle \lambda_i, \mathbf{y}_i - \mathbf{y}\rangle\right)$$

[1] Rajeswaran, A., Lowrey, K., Todorov, E.V., Kakade, S.M.: Towards generalization and simplicity in continuous control. NeurIPS (2017)
[2] Kakade, S.M.: A natural policy gradient. NeurIPS (2001)

FedNPG-ADMM

Algorithm Details

Algorithm 1 FedNPG-ADMM

Input: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$; Number of timesteps T; Penalty constant ρ; Step size η; Initial $\theta_0 \in \mathbb{R}^d$, $\mathbf{y}^0 \in \mathbb{R}^d$, $\{\mathbf{y}_i^0 = \mathbf{y}^0\}_{i=1}^N$, $\{\lambda_i \in \mathbb{R}^d\}_{i=1}^N$.

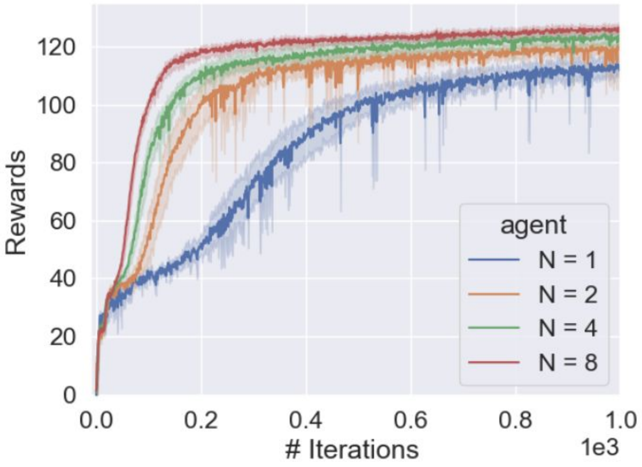
for $k = 1, \cdots, K$ **do**

- ▷ Server broadcast
- Broadcast $\mathbf{y}^{k-1}$ and θ^{k-1} to N agents.
- ▷ Agent update
- **for** each agent $i \in \{N\}$ **do in parallel**
  - $\lambda_i \leftarrow \lambda_i + \rho(\mathbf{y}_i^{k-1} - \mathbf{y}^{k-1})$
  - $\mathbf{g}_i^k \leftarrow \frac{1}{|\mathcal{D}_i|} \sum_{\tau \in \mathcal{D}_i} \sum_{t=0}^{T} \left(\nabla_{\theta^{k-1}} \log \pi_{\theta^{k-1}}(a_t|s_t)\right) \widehat{A}_{\pi_{\theta^{k-1}}}(s_t, a_t)$
  - $\mathbf{y}_i^k \leftarrow (\mathbf{H}_i^k + \rho \mathbf{I})^{-1}(\mathbf{g}_i^k - \lambda_i + \rho \mathbf{y}^{k-1})$
  - Transmit $\mathbf{y}_i^k \in \mathbb{R}^d$ and $\mathbf{g}_i^k \in \mathbb{R}^d$ to the server.
- **end for**
- ▷ Server update
- $\mathbf{y}^k \leftarrow \frac{1}{N} \sum_{i=1}^N \mathbf{y}_i^k$
- $\theta^k \leftarrow \theta^{k-1} + \eta \sqrt{\frac{2N\delta}{(\sum_{i=1}^N \mathbf{g}_i^k)^\top \mathbf{y}^k}} \cdot \mathbf{y}^k$

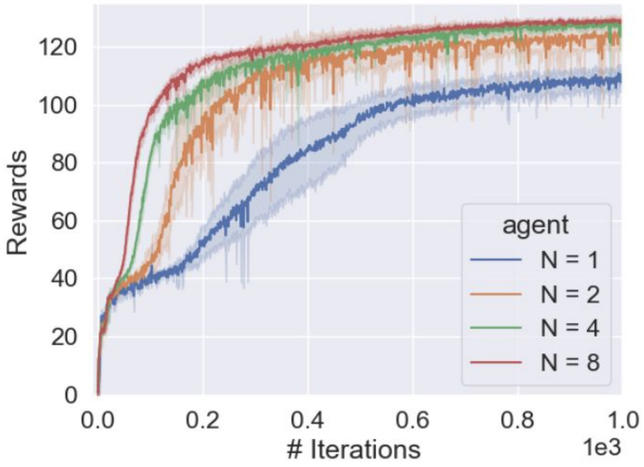
end for

Output: θ^K

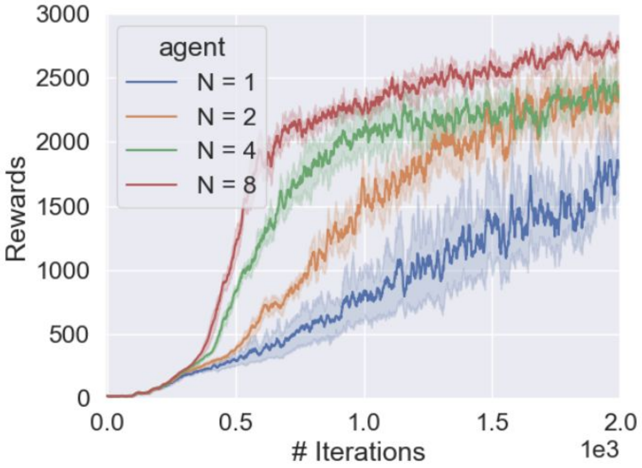
Performances wrt #agents



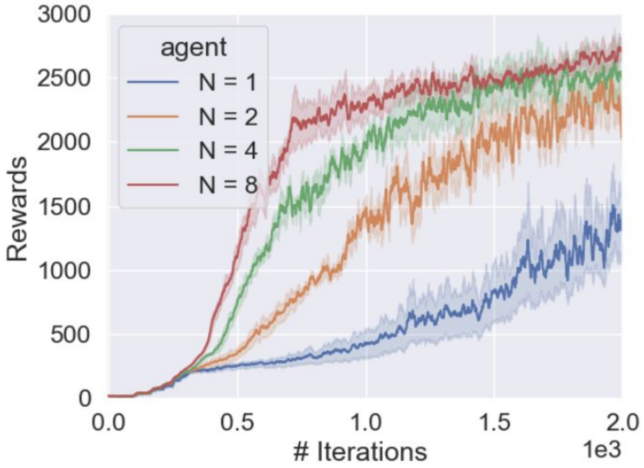
(a) FedNPG (Swimmer-v4)



(b) FedNPG-ADMM (Swimmer-v4)

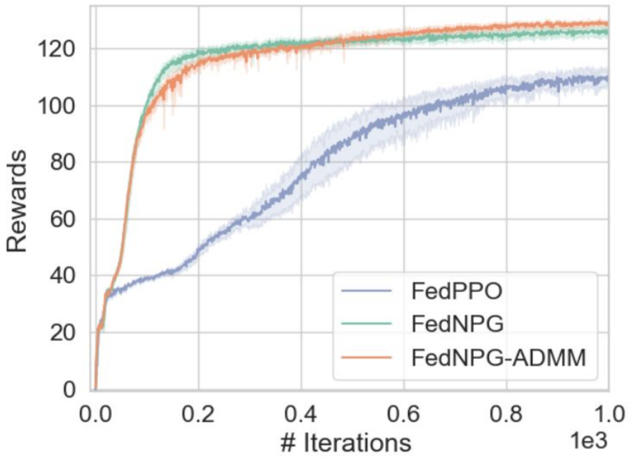


(c) FedNPG (Hopper-v4)

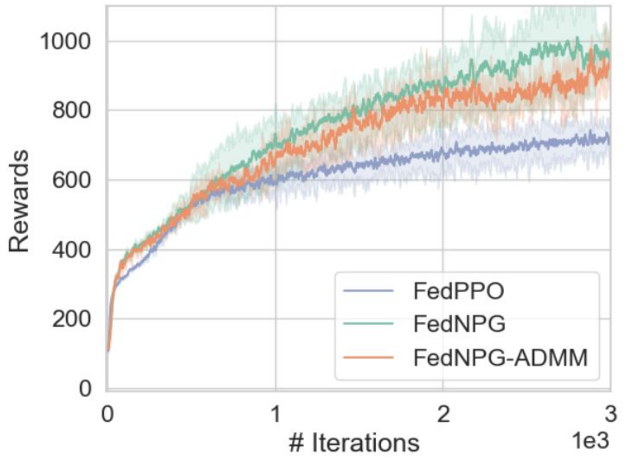


(d) FedNPG-ADMM (Hopper-v4)

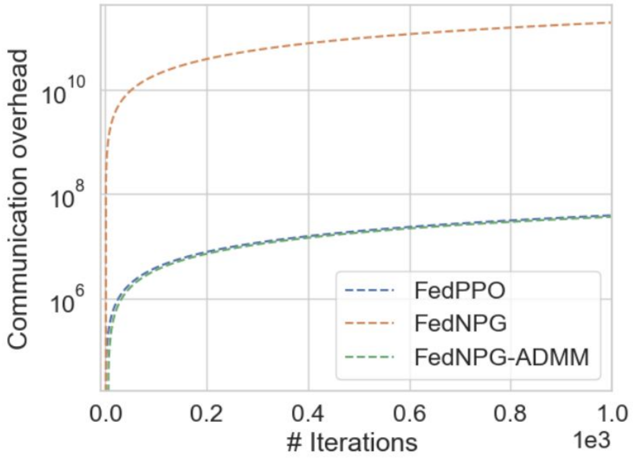
Performance Comparison



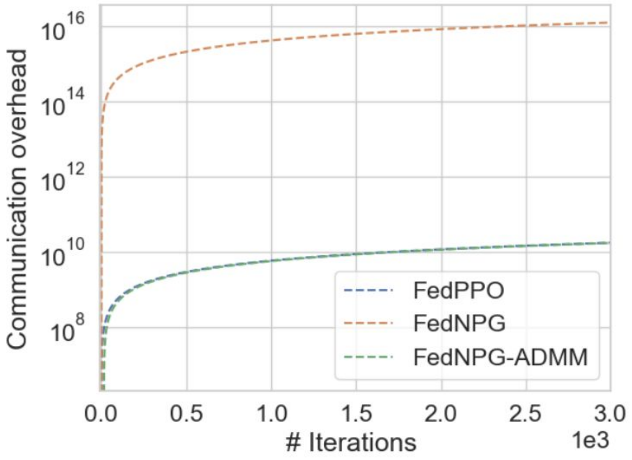
(a) Swimmer-v4 Rewards



(b) Humanoid-v4 Rewards



(c) Swimmer-v4 Overhead



(d) Humanoid-v4 Overhead

Performances with Agent Selection

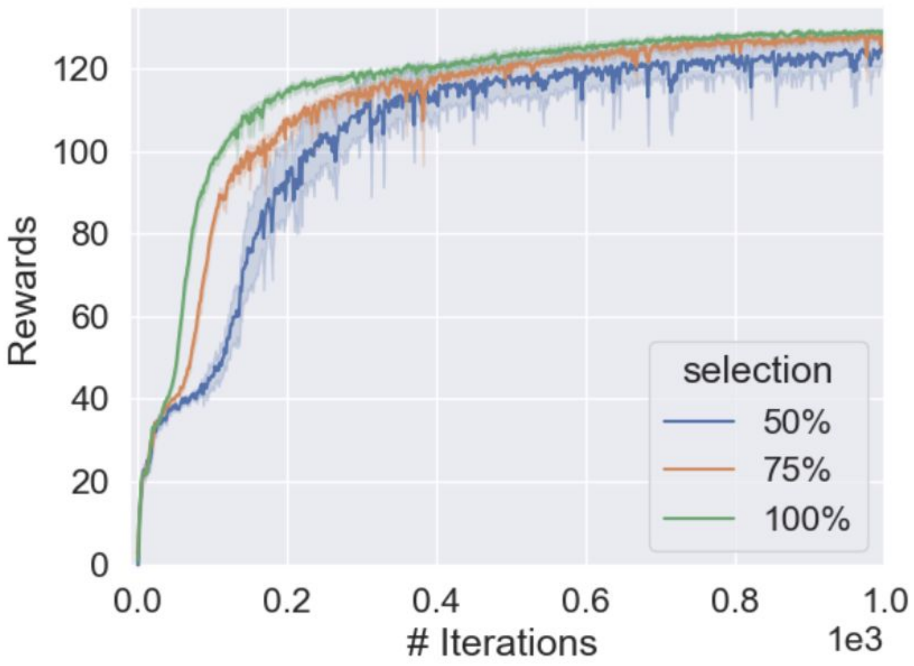


Figure 4: Reward performances of FedNPG-ADMM on the Swimmer-v4 task with agent selection. In each iteration, the server randomly selects 100%, 75%, and 50% of agents for the aggregation.

Thank You

PURDUE UNIVERSITY

Elmore Family School of Electrical and Computer Engineering