Local Stochastic ADMM for Communication-Efficient Distributed Learning

Chaouki ben Issaid, Anis Elgabli, and Mehdi Bennis
Centre for Wireless Communications (CWC)
University of Oulu, Finland
Email: {chaouki.benissaid, anis.elgabli, mehdi.bennis}@oulu.fi

***Abstract*—In this paper, we propose a communication-efficient alternating direction method of multipliers (ADMM)-based algorithm for solving a distributed learning problem in the stochastic non-convex setting. Our approach runs a few stochastic gradient descent (SGD) steps to solve the local problem at each worker instead of finding the exact/approximate solution as proposed by existing ADMM-based works. By doing so, the proposed framework strikes a good balance between the computation and communication costs. Extensive simulation results show that our algorithm significantly outperforms existing stochastic ADMM in terms of communication-efficiency, notably in the presence of non-independent and identically distributed (non-IID) data.**

***Index Terms*—Communication-efficiency, alternating direction method of multipliers (ADMM), stochastic non-convex distributed optimization.**

I. Introduction

Recently, more attention has been paid to designing distributed learning algorithms in the non-convex setting inspired by the vast success of deep neural networks and the important role non-convex optimization plays in their training. In fact, non-convex loss functions, such as the composition of multiple nonlinear activation functions in deep learning, are often needed by the workers to represent the local data accurately.

Primal methods, including stochastic gradient descent (SGD)-based methods [1], and primal-dual methods, such as the alternating direction method of multipliers (ADMM) [2], [3] are the two most used approaches to solve distributed learning problems. To achieve better generalization performance and avoid the overfitting problem, regularization has been considered in the literature. However, for regularized problems, SGD-bases methods are shown to suffer from the lack of capability in exploiting the problem structure, especially in settings with explicit regularization [4].

In contrast to that, ADMM has been commonly used to solve regularized statistical learning problems, such as the generalized LASSO [5]. Originally, ADMM was introduced to solve the deterministic distributed learning problem in the convex case [2], [3], [6], as well as the non-convex [7]–[9]. Recently, several stochastic and online variants of ADMM have been proposed in [10]–[14]. These variants propose to use a second-order approximation technique to simplify the local problem at each worker. While this procedure significantly reduces the computational cost, it has a major drawback since it requires frequent communication with the parameter server (PS) at every iteration. To alleviate this issue, we propose *local-SADMM*, a communication-efficient ADMM framework for solving the distributed learning problem under the non-convex setting. In fact, local-SADMM reduces the communication frequency by allowing the workers to perform more local work before sharing their models with the PS.

The main contributions of this work are summarized as follows

- We propose a communication efficient algorithm that reduces the number of communication rounds between the PS and the workers compared to existing variants of ADMM for the stochastic non-convex regularized learning problem [12], [13], [15].
- Our proposed algorithm offers computational savings in solving the minimization problem of the primal models at the workers' side since the cost of updating the primal variables reduces to the cost of running a few local SGD steps. This reduces the computational time compared to solving the exact problem using the standard ADMM.
- Extensive simulations are conducted to investigate the performance of our proposed algorithm in the identically and independent distributed (IID) and non-IID case as well its sensitivity with respect to the hyperparameters. Simulations results demonstrate that our proposed approach outperforms significantly the mini-batch stochastic ADMM, STOC-ADMM, introduced in [12] under different settings.

The remainder of this paper is organized as follows. In Section II, we formulate the distributed learning problem. Then, we present our proposed framework, local-SADMM, in Section III. Finally, Section IV validates our proposed algorithm's performance by simulations compared to STOC-ADMM for different datasets while studying the impact of the various hyperparameters on the test accuracy.

Notations: Throughout this paper, $\|\cdot\|_1$ and $\|\cdot\|_2$ denote the ℓ^1-norm and the Euclidean norm of a vector, respectively, and $\langle\cdot,\cdot\rangle$ the inner product of two vectors. The notation ∇f stands for the gradient of the function f, $\mathbb{E}[\cdot]$ denotes the expected value, and $\mathbb{1}_{(\cdot)}$ is the indicator function.

II. Problem Formulation

In this paper, we consider a set of N workers communicating with a PS with the aim to learn a global model using only local data. The learning problem can be formulated as the following optimization problem

$$\textbf{(P1)} \quad \min_{\boldsymbol{\Theta}} \sum_{n=1}^{N} f_n(\boldsymbol{\Theta}) + \beta g(\boldsymbol{\Theta}) \tag{1}$$

where $\boldsymbol{\Theta} \in \mathbb{R}^{d\times 1}$ is the global model, the functions $f_n : \mathbb{R}^d \to \mathbb{R}$ are local *differentiable* and possibly *nonconvex* functions, the function g is a *nonsmooth* and *convex* function that regularizes the learning model and helps to generalize better, and $\beta > 0$ is the regularization parameter. Problem **(P1)** can be reformulated as follows

$$\textbf{(P2)} \quad \min_{\boldsymbol{\theta},\boldsymbol{\Theta}} \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n) + \beta g(\boldsymbol{\Theta}) \tag{2}$$

$$\text{s.t. } \boldsymbol{\theta}_n = \boldsymbol{\Theta}, \forall n \in \{1, \cdots, N\}, \tag{3}$$

where $\boldsymbol{\theta}_n$ is the local model at worker n and $\boldsymbol{\theta} = \{\boldsymbol{\theta}_n\}_{n=1}^N$ is the set of local models.

In this case, the augmented Lagrangian of the optimization problem **(P2)** can be written as

$$\mathcal{L}_\rho(\boldsymbol{\theta}, \boldsymbol{\Theta}, \boldsymbol{\lambda}) = \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n) + \beta g(\boldsymbol{\Theta}) + \sum_{n=1}^{N} \langle \lambda_n, \boldsymbol{\theta}_n - \boldsymbol{\Theta} \rangle + \frac{\rho}{2} \sum_{n=1}^{N} \|\boldsymbol{\theta}_n - \boldsymbol{\Theta}\|_2^2, \tag{4}$$

where $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_n\}_{n=1}^N$ is the collection of dual variables, $\boldsymbol{\lambda}_n$ is the dual variable between the worker n and the PS, and the parameter $\rho > 0$ is a constant penalty parameter.

III. Local Stochastic ADMM

For illustration purpose, we start by reviewing the standard ADMM for solving the problem **(P2)**. At iteration $k+1$, the standard ADMM algorithm runs as follow

(1) Every worker n updates its primal variable by solving

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \{f_n(\boldsymbol{\theta}_n) + \langle \boldsymbol{\lambda}_n^k, \boldsymbol{\theta}_n - \boldsymbol{\Theta}^k \rangle + \frac{\rho}{2} \|\boldsymbol{\theta}_n - \boldsymbol{\Theta}^k\|_2^2\}, \tag{5}$$

and sends its updated model to the PS.

(2) The primal variable of the PS is updated as

$$\boldsymbol{\Theta}^{k+1} = \arg\min_{\boldsymbol{\Theta}} \{\beta g(\boldsymbol{\Theta}) + \sum_{n=1}^{N} \langle \boldsymbol{\lambda}_n^k, \boldsymbol{\theta}_n^{k+1} - \boldsymbol{\Theta} \rangle + \frac{\rho}{2} \sum_{n=1}^{N} \|\boldsymbol{\theta}_n^{k+1} - \boldsymbol{\Theta}\|_2^2\} \tag{6}$$

and sends it to the workers.

Algorithm 1 Local Stochatic ADMM (Local-SADMM)

Input: $N, \rho, T, f_n(\boldsymbol{\theta}_n)$, $\boldsymbol{\lambda}_n^0 = 0$, $\forall n$, $\boldsymbol{\theta}_{n,0}^0 = \boldsymbol{\Theta}^0 = 0$
for $k = 0, 1, 2, \cdots, K$ **do**
  Each worker n **in parallel:**
    for $t = 0, 1, 2, \cdots, T-1$ **do**
      samples uniformly and independently a mini-batch $\{\xi_{n,i}^k\}_{i=1}^B$ of size B.
      evaluates stochastic gradient using the sampled mini-batch using (12).
      updates local model as in (11).
    end for
    sets its primal variable $\boldsymbol{\theta}_n^{k+1} = \boldsymbol{\theta}_{n,T}^k$.
    sends $\left(\boldsymbol{\theta}_n^{k+1} + \boldsymbol{\lambda}_n^k/\rho\right)$ to the parameter server.
  The parameter server:
    computes its primal variable $\boldsymbol{\Theta}^{k+1}$ via (6).
    sends $\boldsymbol{\Theta}^{k+1}$ to all workers.
  Each worker n **in parallel:**
    updates its dual variables $\boldsymbol{\lambda}_n^{k+1}$ locally via (7).
    sets $\boldsymbol{\theta}_{n,0}^{k+1} = \boldsymbol{\Theta}^{k+1}$.
end for
Output: Global model $\boldsymbol{\Theta}^K$.

(3) The dual variables are updated locally for every worker

$$\boldsymbol{\lambda}_n^{k+1} = \boldsymbol{\lambda}_n^k + \rho(\boldsymbol{\theta}_n^{k+1} - \boldsymbol{\Theta}^{k+1}). \tag{7}$$

When the loss function of the n^{th} worker is of the form $f_n(\boldsymbol{\theta}_n) = \mathbb{E}[f_n(\boldsymbol{\theta}_n, \xi)]$, as is typical in classification tasks using deep neural networks, the standard ADMM scheme is no longer applicable [11]. Note that the random vector ξ follows a fixed but unknown distribution, from which we are able to get a set of IID samples. Most stochastic versions of ADMM [10], [11] attempt to approximate the local loss function by its second-order Taylor expansion and to use an unbiased estimate of the gradient, such as the mini-batch stochastic gradient as in *STOC-ADMM* [12]. We briefly review STOC-ADMM in the next section and highlight its major weakness and how local-SADMM can solve this issue.

A. STOC-ADMM

For a given iteration k, let B be the size of a mini-batch $\mathcal{I}_n^k = \{\xi_{n,i}^k\}_{i=1}^B$ where $\{\xi_{n,i}^k\}_{i=1}^B$ are i.i.d variables. The mini-batch stochastic gradient estimator of worker n is

$$\boldsymbol{g}_n^k = \frac{1}{B} \sum_{i=1}^{B} \nabla f_n(\boldsymbol{\theta}_n^k, \xi_{n,i}^k). \tag{8}$$

We clearly see that $\boldsymbol{g}_n^k$ satisfies $\mathbb{E}\left[\boldsymbol{g}_n^k\right] = \nabla f_n(\boldsymbol{\theta}_n^k)$. With the above definition at hand, the update of the primal variable of

every worker n in the STOC-ADMM algorithm is given by

$$\boldsymbol{\theta}_n^{k+1} = \boldsymbol{\theta}_n^k - \frac{1}{\rho+\eta}\left(\boldsymbol{\lambda}_n^k + \boldsymbol{g}_n^k + \rho\left(\boldsymbol{\theta}_n^k - \boldsymbol{\Theta}^k\right)\right). \quad (9)$$

The update of the global model at the PS side as well as the update of the dual variable of each worker is the same as in (6) and (7), respectively. Note that in the update (9), the function $f_n(\boldsymbol{\theta}_n)$ is replaced by its quadratic approximation so that the primal models of each worker solves the following problem

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \{f_n(\boldsymbol{\theta}_n^k) + \langle \boldsymbol{g}_n^k, \boldsymbol{\theta}_n - \boldsymbol{\theta}_n^k\rangle + \frac{\eta}{2}\|\boldsymbol{\theta}_n - \boldsymbol{\theta}_n^k\|^2 + \langle \boldsymbol{\lambda}_n^k, \boldsymbol{\theta}_n - \boldsymbol{\Theta}^k\rangle + \frac{\rho}{2}\|\boldsymbol{\theta}_n - \boldsymbol{\Theta}^k\|_2^2\}, \quad (10)$$

where $\eta > 0$. Note that while using the Taylor expansion and mini-batch stochastic gradient in the update of the primal models of the workers reduces the computational time significantly, it does not solve the communication bottleneck issue caused by the frequent communication with the PS.

B. Local-SADMM

To solve the communication bottleneck issue that the current proposed stochastic ADMM variants (e.g. STOC-ADMM) suffer from, we introduce our proposed stochastic version of ADMM coined local-SADMM, a communication-efficient algorithm for solving **(P2)**. In our algorithm, we propose to run a few number of local SGD steps in order to have a solution to the local problem described in (5).

In the remainder, we denote by K the number of communication rounds between the PS and the workers and T the number of local SGD steps. Given a learning rate $\alpha > 0$, the update of the primal model, at iteration $k+1$, is given by

$$\boldsymbol{\theta}_{n,t+1}^k = \boldsymbol{\theta}_{n,t}^k - \alpha\left(\boldsymbol{g}_{n,t}^k + \boldsymbol{\lambda}_n^k + \rho(\boldsymbol{\theta}_{n,t}^k - \boldsymbol{\Theta}^k)\right), \quad (11)$$

where $\boldsymbol{\theta}_{n,t}^k$ is the primal model of the n^{th} worker after running k global iterations and t local iterations. Moreover, the mini-batch stochastic gradient $\boldsymbol{g}_{n,t}^k$ is defined as

$$\boldsymbol{g}_{n,t}^k = \frac{1}{B}\sum_{i=1}^{B} \nabla f_n(\boldsymbol{\theta}_{n,t}^k, \xi_{n,i}^k). \quad (12)$$

Under local-SADMM, the computation cost per local iteration reduces to computing the mini-batch stochastic gradient $\boldsymbol{g}_{n,t}^k$, which is light-weight. Note that STOC-ADMM can be seen a special case if we set $\alpha = 1/(\rho+\eta)$ and $T = 1$.

After running T local iterations, the new model that each worker uploads to the PS is $\boldsymbol{\theta}_n^{k+1} = \boldsymbol{\theta}_{n,T}^k$. Instead of starting the local iterations from the previous model, *i.e.* $\boldsymbol{\theta}_{n,0}^k = \boldsymbol{\theta}_n^k$, we rather start from the previous global model, *i.e.* $\boldsymbol{\theta}_{n,0}^k = \boldsymbol{\Theta}^k$. By doing so, not only local-SADMM reduces the number of communication rounds but also has a lower computational cost compared to standard ADMM. Furthermore, it is worth mentioning that local-SADMM contributes to reducing the variance compared to the STOC-ADMM. The detailed steps of local-SADMM are summarized in Algorithm 1.

IV. Numerical Results

In this section, we validate the performance of our proposed algorithm, local-SADMM, compared to STOC-ADMM while studying the impact of certain hyperparameters (the number of local iterations T, the penalty parameter ρ, and the mini-batch size B) on the performance of these algorithms when $g(\cdot) = \|\cdot\|_1$. In this case, the update of the PS can be explicitly derived as in (13), where $\boldsymbol{\Theta}_i^{k+1}$ is the i^{th} component of the PS model at iteration $k+1$ and $\text{sgn}(\cdot)$ is the sign function.

A. Experimental Setup

We consider two experiments

- Digits classification using MNIST dataset. The MNIST dataset consists of hand-written $0-9$ digit images. The aim is to predict to which class each image belongs,
- Regression on sinusoid data [16]. Points from a sinusoidal wave with amplitude A and phase ϕ are generated randomly and distributed across the workers. The objective of the collaborative training is to train each worker to accurately produce sinusoidal output y when the input $x \in [-\pi, \pi]$.

In all of our experiments, we plot the average test accuracy and one standard error shaded area based on five runs. For each task, we use the same hyperparameters, detailed in Table I unless otherwise stated in the text. We have used a multi-layer perceptron neural network with two hidden layers having 128 and 64 neurons, respectively, for the MNIST experiment and having 64 and 32 neurons, respectively, for the sinusoid experiment. The activation function used in the hidden layers is the rectified linear unit. The loss function used in the sinusoid experiment is the mean squared error, while we use the cross-entropy loss in the MNIST experiment.

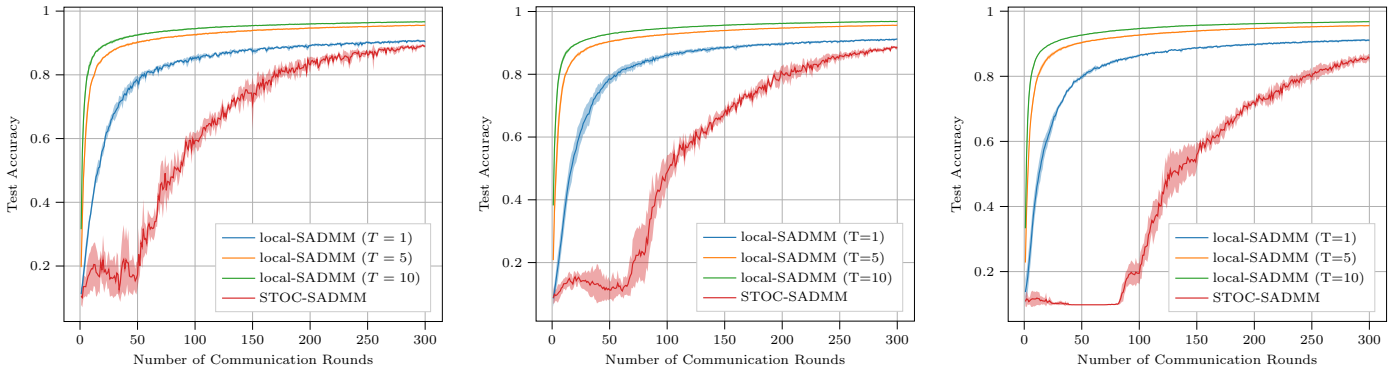
TABLE I: Parameters used in the numerical experiments.

Parameter	Sinusoid	MNIST
Learning rate (α)	0.01	0.01
Batch size (B)	100	10
Number of workers (N)	10	50
Number of local iterations (T)	10	10
Penalty parameter (ρ)	0.1	1
Regularization parameter (β)	0.1	0.01

B. Experimental Results

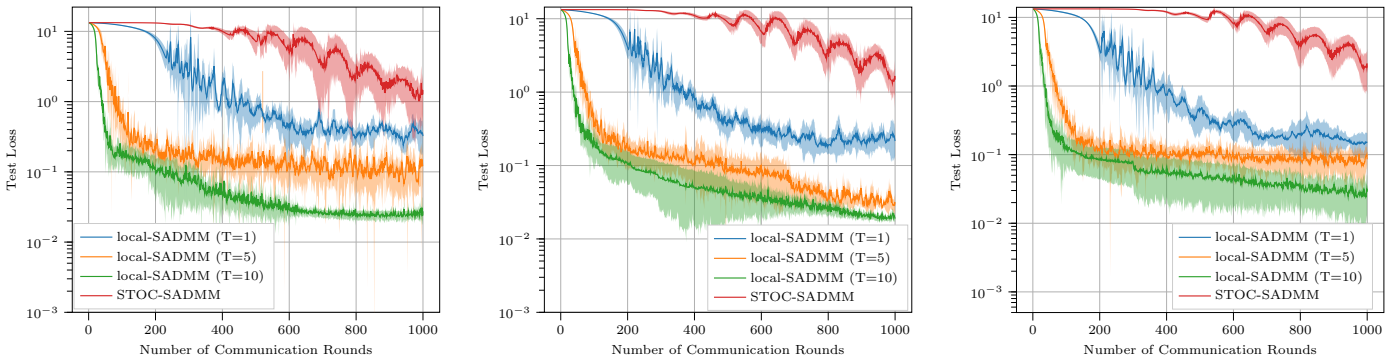
1) IID Case: In this case, the number of samples are uniformly distributed across N workers for both datasets.

In Fig. 1, we start by plotting the test accuracy as a function of the number of communication rounds for different number of workers $N \in \{25, 50, 100\}$ and for different number of local iterations $T \in \{1, 5, 10\}$. Irrespective of the number of workers considered, we can see that local-SADMM significantly outperforms STOC-ADMM. Note that the parameter η used in STOC-ADMM is chosen so that the term $1/(\eta+\rho)$ matches the learning



(a) $N = 25$ (b) $N = 50$ (c) $N = 100$

Fig. 1: Test accuracy in the MNIST experiment for different number of workers.



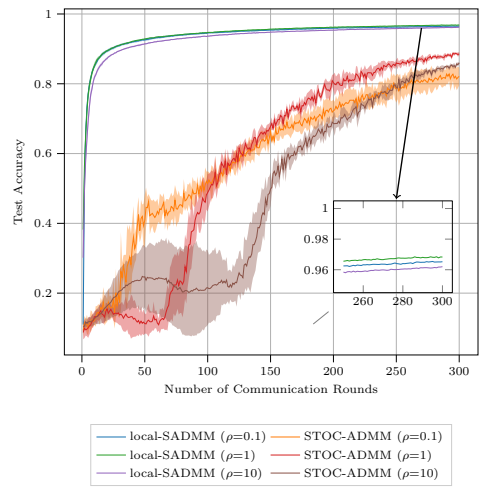
(a) $N = 10$ (b) $N = 15$ (c) $N = 20$

Fig. 2: Test loss in the sinusoid regression experiment for different number of workers.

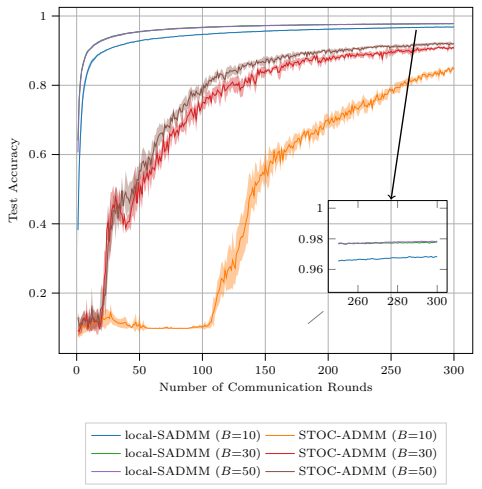
rate α used in local S-ADMM for fair comparison. We can clearly see that local-SADMM strikes a good balance between communication-efficiency and achieving high test accuracy. In fact, for $N = 25$, the local-SADMM test accuracy increases achieving around 90%, 95.5% and 96.5%, for $T \in \{1, 5, 10\}$ respectively, while STOC-ADMM achieves 88% test accuracy. In terms of communication-efficiency, the gap in performance is even clearer. Note that with 100 communications rounds, local-SADMM can still achieve around 85%, 92% and 93% for $T \in \{1, 5, 10\}$ respectively, compared to only 60% for STOC-ADMM. A similar gap in performance is observed for other choices of the number of workers $N \in \{50, 100\}$.

In Fig. 2, we plot the test loss as a function of the number of communication rounds for different number of workers $N \in \{10, 15, 20\}$ and for different number of local iterations $T \in \{1, 5, 10\}$. Similar to the results obtained in the classification problem, we observe that, irrespective of the number of workers and local iterations, local-SADMM significantly outperforms STOC-ADMM. From Fig. 1 and Fig. 2, we conclude that local-SADMM achieves better communication-efficiency and better performance compared to STOC-ADMM.

In Fig. 3, we study the impact of ρ and B on the performance of both algorithms for the MNIST experiment. We plot the test accuracy in terms of the number of communication rounds for different values of $\rho \in \{0.1, 1, 10\}$ in Fig. 3a and of $B \in \{10, 30, 50\}$ in Fig. 3b. We can see from Fig. 3a that local-SADMM offers a significant communication efficiency gain compared to STOC-ADMM for the different values of ρ. In fact, for $\rho = 1$, STOC-ADMM requires 200 communication rounds to reach 80% test accuracy while local-SADMM achieves the same test accuracy using only 10 communication rounds. Furthermore, local-SADMM seems to have a more stable performance while varying ρ compared to STOC-ADMM. Similarly, from Fig. 3b, we observe that, as the batch size increases, the performance of both algorithms improves. However, as noted earlier, our proposed algorithm achieves significant communication efficiency gains compared to STOC-ADMM and it is less sensitive to the batch size B. In fact, increasing B from 30 to 50 barely changes the performance of local-SADMM. Finally, from Fig. 3, we notice that running a few local SGD steps to solve the local minimization problem reduces the variance of local-SADMM estimate compared to STOC-ADMM.

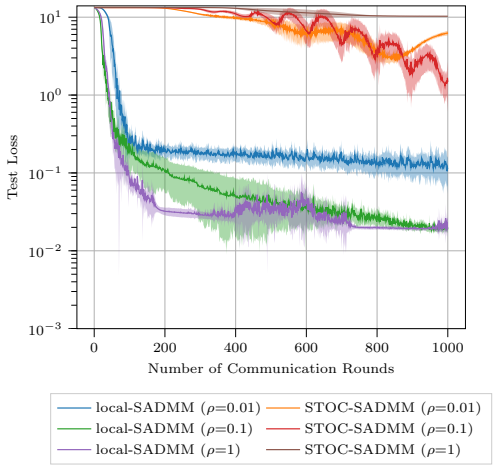


(a) Impact of the penalty parameter ρ.

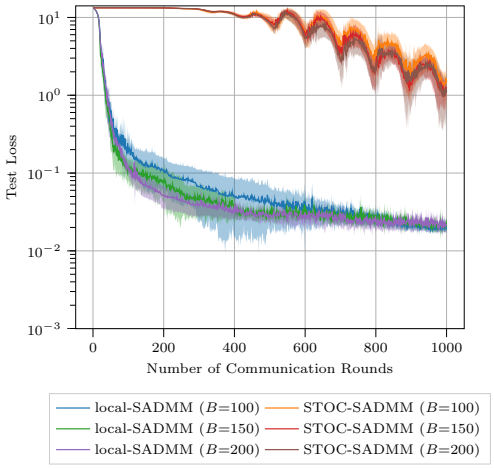


(b) Impact of the mini-batch size B.

Fig. 3: Impact of hyperparameters on the test accuracy in the MNIST experiment.



(a) Impact of the penalty parameter ρ.



(b) Impact of the mini-batch size B.

Fig. 4: Impact of hyperparameters on the test loss in the sinusoid experiment.

Similarly for the sinusoid regression experiment, we plot the loss in terms of the number of communication rounds for both algorithms for different values of $\rho \in \{0.1, 1, 10\}$ in Fig. 4a and of $B \in \{10, 30, 50\}$ in Fig. 4b. The observed results confirm our findings regarding the classification experiment and show the advantage of local-SADMM over STOC-ADMM for all choices of ρ and B.

2) Non-IID Case: In practice, the local data distribution can usually vary greatly between across the different workers, hence, the importance of investigating the performance in the non-IID setting. In this case, we follow the same procedure of artificially distributing the MNIST data over workers as in [1] to get a pathological non-IID partition of the data. By doing so, we make sure that the local dataset of the majority of the workers will only contain examples of two digits. We plot the test accuracy achieved by both algorithms in terms of number of communication rounds, for $N = 100$ in Fig. 5. The gap in the performance between the two algorithms is more evident in the non-IID case. In fact, while STOC-ADMM barely achieves a 50% test accuracy, local-SADMM test accuracy reaches 90% with around 50 communication rounds when $T = 10$.

$$\boldsymbol{\Theta}_i^{k+1} = \frac{1}{\rho N}\left(\left|\sum_{n=1}^{N}\boldsymbol{\lambda}_{n,i}^{k} + \rho\sum_{n=1}^{N}\boldsymbol{\theta}_{n,i}^{k+1}\right| - \beta\right) \times \operatorname{sgn}\left(\left|\sum_{n=1}^{N}\boldsymbol{\lambda}_{n,i}^{k} + \rho\sum_{n=1}^{N}\boldsymbol{\theta}_{n,i}^{k+1}\right|\right) \times \mathbb{1}_{\left(\left|\sum_{n=1}^{N}\boldsymbol{\lambda}_{n,i}^{k} + \rho\sum_{n=1}^{N}\boldsymbol{\theta}_{n,i}^{k+1}\right| > \beta\right)}. \tag{13}$$

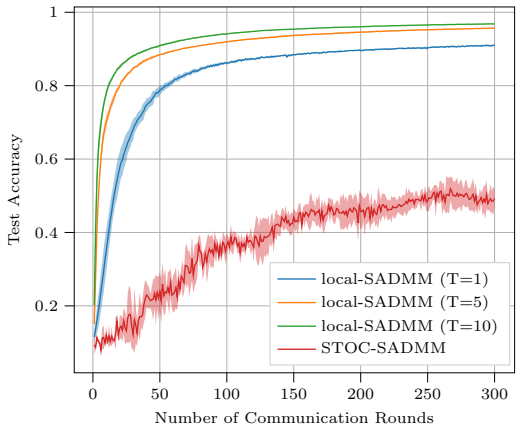


Fig. 5: Test accuracy in the non-IID setting.

V. Conclusions

In this work, we proposed local-SADMM, a communication-efficient ADMM-based framework that solves the distributed learning problem when the local functions are non-convex such as the case when training DNNs. By running a few local SGD steps locally, our approach significantly reduces the number of communication rounds between the devices and the PS. Extensive numerical simulations show that local-SADMM outperforms STOC-ADMM for solving the learning problem in terms of the number of communication rounds. The performance gap is more evident in the practical case when the data is non-IID.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[2] M. Hong and Z.-Q. Luo, "On the linear convergence of the alternating direction method of multipliers," *Mathematical Programming*, vol. 162, no. 1-2, pp. 165–199, 2017.

[3] C. B. Issaid, A. Elgabli, J. Park, M. Bennis, and M. Debbah, "Communication efficient distributed learning with censored, quantized, and generalized group ADMM," *arXiv preprint arXiv:2009.06459*, 2020.

[4] L. Xiao, "Dual averaging methods for regularized stochastic learning and online optimization," *Journal of Machine Learning Research*, vol. 11, no. 88, pp. 2543–2596, 2010.

[5] Y. Xu, M. Liu, Q. Lin, and T. Yang, "ADMM without a fixed penalty parameter: Faster convergence with new adaptive penalization," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 1267–1277.

[6] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus ADMM," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, 2014.

[7] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 337–364, 2016.

[8] M. Hong, D. Hajinezhad, and M.-M. Zhao, "Prox-PDA: The proximal primal-dual algorithm for fast distributed nonconvex optimization and learning over networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1529–1538.

[9] X. Wang, J. Yan, B. Jin, and W. Li, "Distributed and parallel ADMM for structured nonconvex optimization problem," *IEEE transactions on cybernetics*, 2019.

[10] H. Ouyang, N. He, L. Tran, and A. Gray, "Stochastic alternating direction method of multipliers," in *International Conference on Machine Learning*. PMLR, 2013, pp. 80–88.

[11] S. Azadi and S. Sra, "Towards an optimal stochastic alternating direction method of multipliers," in *International Conference on Machine Learning*, 2014, pp. 620–628.

[12] F. Huang and S. Chen, "Mini-batch stochastic ADMMs for nonconvex nonsmooth optimization," *arXiv preprint arXiv:1802.03284*, 2018.

[13] F. Huang, S. Chen, and H. Huang, "Faster stochastic alternating direction method of multipliers for nonconvex optimization," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2839–2848.

[14] H. Chen, Y. Ye, M. Xiao, M. Skoglund, and H. V. Poor, "Coded stochastic ADMM for decentralized consensus optimization with edge computing," *IEEE Internet of Things Journal*, 2021.

[15] F. Huang, S. Chen, and Z. Lu, "Stochastic alternating direction method of multipliers with variance reduction for nonconvex optimization," *arXiv preprint arXiv:1610.02758*, 2016.

[16] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1126–1135.