2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Switch and Conquer: Efficient Algorithms By Switching Stochastic Gradient Oracles For Decentralized Saddle Point Problems

Chhavi Sharma, Vishnu Narayanan and P. Balamurugan

***Abstract*— We consider a class of non-smooth strongly convex-strongly concave saddle point problems in a decentralized setting without a central server. To solve a consensus formulation of problems in this class, we develop an inexact primal dual hybrid gradient (inexact PDHG) procedure that allows generic gradient computation oracles to update the primal and dual variables. We first investigate the performance of inexact PDHG with stochastic variance reduction gradient (SVRG) oracle. Our numerical study uncovers a significant phenomenon of initial conservative progress of iterates of IPDHG with SVRG oracle. To tackle this, we develop a simple and effective switching idea, where a generalized stochastic gradient (GSG) computation oracle is employed to hasten the iterates' progress to a saddle point solution during the initial phase of updates, followed by a switch to the SVRG oracle at an appropriate juncture. The proposed algorithm is named Decentralized Proximal Switching Stochastic Gradient method with Compression (C-DPSSG), and is proven to converge to an ϵ-accurate saddle point solution with linear rate. Apart from delivering highly accurate solutions, our study reveals that utilizing the best convergence phases of GSG and SVRG oracles makes C-DPSSG well suited for obtaining solutions of low/medium accuracy faster, useful for certain applications. Numerical experiments on two benchmark machine learning applications show C-DPSSG's competitive performance which validate our theoretical findings. The codes used in the experiments can be found here.**

I. Introduction

We focus on solving the following saddle point (or min-max) problem in a fully decentralized setting **without a central server**:

$$\min_{x\in\mathbb{R}^{d_x}} \max_{y\in\mathbb{R}^{d_y}} \frac{1}{m}\sum_{i=1}^{m}(f_i(x,y)+g(x)-r(y)), \qquad \text{(SPP)}$$

where $f_i : \mathbb{R}^{d_x}\times\mathbb{R}^{d_y} \to \mathbb{R}$ private to every node $i \in \{1,2,\ldots,m\} =: [m]$ is smooth, strongly convex in primal variable x and strongly concave in dual variable y and $g : \mathbb{R}^{d_x} \to \mathbb{R}$ and $r : \mathbb{R}^{d_y}\to\mathbb{R}$ are proper, convex and potentially non-smooth functions. This class of saddle point problems finds its use in distributionally robust optimization, robust classification and regression applications, AUC maximization problems [24], [32], [34] and multi-agent reinforcement learning [29]. Additionally, saddle point problems appear in the Lagrangian formulations of constrained minimization problems [35], [21]. Decentralized environments are useful for large-scale systems where privacy and other constraints on data sharing (e.g. legal, geographical) prevent the availability of entire data set in a single computing machine (or node). In this work, we consider a decentralized environment where the computing nodes possess similar processing and storage capabilities. The (static) topology of the decentralized environment is represented using an undirected, connected, simple graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = [m]$ denotes the set of m computing nodes and an edge $e_{ij} \in \mathcal{E}$ denotes the fact that nodes $i, j \in \mathcal{V}$ are connected. Also, we assume that the communication is synchronous and at every synchronization step, node i communicates only with its neighbors $\mathcal{N}(i) = \{j \in \mathcal{V} : e_{ij} \in \mathcal{E}\}$.

General stochastic gradient oracle (GSGO) [5], popularly used to solve saddle point problems [19], [30], [34], [3], unfortunately suffers from inherent variance developed due to stochastic gradients used for updating primal and dual variables at every epoch. Despite the availability of stochastic variance reduction gradient oracle (SVRGO) [9], [12], which addresses GSGO's variance issue, GSGO is adopted by practitioners due to its simplicity and fast progress in the initial stage. SVRGO prepares itself from the start to keep the variance under control which affects the crucial initial phase convergence. However, the variance in SVRGO vanishes asymptotically speeding up its progress at the later stages. The fast convergence behavior of GSGO at the initial stage and SVRGO at the later stage respectively, provide inspiration for developing a novel algorithm in this work, where a switch is performed between these stochastic gradient oracles.

Apart from gradient computations, high dimensional parameters are communicated by each node in the decentralized environment with its neighbors, which becomes expensive. Thus in this paper, we aim to develop a primal dual decentralized algorithm which attains efficiency in gradient computations by harnessing the best convergence phases of GSGO and SVRGO, and communication efficiency by using compressed representations [17], [22], [20], [34] of iterates. We summarize below the **contributions** of this work:

1) Inspired by algorithms developed for decentralized minimization problems [20], [15], we design a decentralized inexact primal dual hybrid gradient method with compression (IPDHG) by exploiting the consensus constrained formulation of (SPP).
2) We numerically study the initial behavior of IPDHG with SVRGO and GSGO. To improve the observed initial conservative progress of iterates of IPDHG with SVRGO towards an ϵ-accurate saddle point solution, we propose a Decentralized Proximal Switching Stochastic Gradient method with Compression (C-DPSSG), where a generalized stochastic gradient oracle guides the initial progress of iterates, which switches to SVRGO at an appropriate point during the iterative update process.

Chhavi Sharma, Vishnu Narayanan and P. Balamurugan are with Industrial Engineering and Operations Research (IEOR), IIT Bombay, Mumbai, India-400076. Email: {chhavisharma, vishnu, balamurugan.palaniappan}@iitb.ac.in

C-DPSSG is useful to obtain solutions of low/medium accuracy (where $\epsilon \approx 10^{-4}$) faster, pertinent to certain applications. Using SVRGO at the later iterations of C-DPSSG reduces the variance and hence provides highly accurate solutions in the long run. We further prove that C-DPSSG converges to an ϵ-accurate saddle point solution with linear rate.

3) We conduct experiments on robust binary classification and AUC maximization problems to demonstrate the practical performance of proposed algorithms.

To our knowledge, this is the first work which provides a closer look at the behavior of GSGO and SVRGO in a newly designed IPDHG scheme *with compression* to solve saddle point problems of the form (SPP). Note that a practical improvement in SVRG is studied in [2] using a combination of GSGO and SVRGO for solving *smooth convex minimization problems in a single machine setting*; however we leverage the best performance phases of GSGO and SVRGO to solve *non-smooth saddle point problems in a decentralized environment*. We now present notations useful for subsequent discussion.

Notations: Let $z{=}(x,y){\in}\mathbb{R}^{d_x+d_y}$ denote the pair of primal variable x and dual variable y, $z^\star{=}(x^\star,y^\star)$ denote a saddle point solution of problem (SPP) and $\mathbf{z}^\star = ((z^\star)^\top,\dots,(z^\star)^\top)$. Weights W_{ij} associated with the communication link between a pair of nodes $(i,j) \in \mathcal{V}\times\mathcal{V}$ are collected into a matrix W of size $m\times m$. I_d denotes a $d\times d$ identity matrix, $\mathbf{1}$ denotes a $m\times 1$ column vector of ones and $J = \frac{1}{m}\mathbf{1}\mathbf{1}^\top$ denotes a $m\times m$ matrix of uniform weights equal to $\frac{1}{m}$. $A\otimes B$ denotes the Kronecker product of two matrices A and B. Let $f(x,y) \coloneqq \sum_{i=1}^m f_i(x,y)$. Condition number κ_f of f is defined as L/μ, where L is the smoothness parameter of $f_i(x,y)$ (see Appendix XI in [27]) and $\mu = \min\{\mu_x,\mu_y\}$ (see Assumptions 1-2). Condition number κ_g of communication graph $\mathscr{G}$ is defined as the ratio of largest eigenvalue and second smallest eigenvalue of $I-W$. For a $d\times 1$ vector u and for some $d\times d$ symmetric positive semi-definite (p.s.d) matrix A, we define $\|u\|_A^2 = u^\top Au$.

Paper Organization: We develop and interpret IPDHG algorithm in Section II, followed by a discussion of assumptions (Section III). Early stage behavior of IPDHG with SVRGO and GSGO is explained is Section IV. The proposed C-DPSSG algorithm is presented in Section V. Related work is discussed in Section VI and experimentation details are in Section VII. Due to space constraints, all proofs and additional experiments are deferred to our technical report [27].

II. Algorithm Development

Before proceeding to the algorithm development, we first present few terminologies to be used in the remaining part of the paper. Assuming the local copy of (x,y) in i-th node as (x^i,y^i), we collect local primal and dual variables into $\mathbf{x} = \left(x^1,x^2,\dots x^m\right) \in \mathbb{R}^{md_x}$ and $\mathbf{y} = \left(y^1,y^2,\dots y^m\right) \in \mathbb{R}^{md_y}$. Using this notation and following [21], the problem (SPP) can be formulated as:

$$\min_{\mathbf{x}\in\mathbb{R}^{md_x}} \max_{\mathbf{y}\in\mathbb{R}^{md_y}} F(\mathbf{x},\mathbf{y}) + G(\mathbf{x}) - R(\mathbf{y})$$
$$\text{s.t. } (U\otimes I_{d_x})\mathbf{x} = 0, \quad (U\otimes I_{d_y})\mathbf{y} = 0, \tag{1}$$

where $F(\mathbf{x},\mathbf{y}) = \sum_{i=1}^m f_i(x^i,y^i)$, $G(\mathbf{x}) = \sum_{i=1}^m g(x^i)$, $R(\mathbf{y}) = \sum_{i=1}^m r(y^i)$, $U = \sqrt{I_m - W}$ and consensus constraints are present on $\mathbf{x}$ and $\mathbf{y}$. The assumptions on W (to be made later) would imply $I_m - W$ to be symmetric p.s.d. and hence leads to existence of $\sqrt{I_m-W}$. We consider the following Lagrangian function of problem (1):

$$\mathcal{L}(\mathbf{x},\mathbf{y};S^{\mathbf{x}},S^{\mathbf{y}}) = F(\mathbf{x},\mathbf{y}) + G(\mathbf{x}) - R(\mathbf{y}) + \langle S^{\mathbf{x}},(U\otimes I_{d_x})\mathbf{x}\rangle + \langle S^{\mathbf{y}},(U\otimes I_{d_y})\mathbf{y}\rangle, \tag{2}$$

where $S^{\mathbf{x}} \in \mathbb{R}^{md_x}$ and $S^{\mathbf{y}} \in \mathbb{R}^{md_y}$ denote the Lagrange multipliers associated with consensus constraints on variables $\mathbf{x}$ and $\mathbf{y}$ respectively. We prove that solving constrained problem (1) is equivalent to solving the following problem (see Theorem 3 in [27]):

$$\min_{\mathbf{x}\in\mathbb{R}^{md_x},S^{\mathbf{y}}\in\mathbb{R}^{md_y}} \max_{\mathbf{y}\in\mathbb{R}^{md_y},S^{\mathbf{x}}\in\mathbb{R}^{md_x}} \mathcal{L}(\mathbf{x},\mathbf{y};S^{\mathbf{x}},S^{\mathbf{y}}). \tag{3}$$

A similar equivalence is provided in [26] under the assumption of convex compact constraint sets and bounded gradients of $F(\mathbf{x},\mathbf{y})$. On the contrary, we formally show the equivalence using convexity-concavity of $f_i(x,y)$ and using properties of weight matrix W (to be defined in next section). Further, our proof does not require compactness and bounded gradient assumptions.

To solve problem (3), we propose gradient descent ascent parallel updates for the primal-dual variable pair $\mathbf{x}, S^{\mathbf{x}}$ and dual-primal pair $\mathbf{y}, S^{\mathbf{y}}$, illustrated in equations (P1) and (D1). Note that in eq. (P1), $\nu_{t+1}^{\mathbf{x}}$ is found using a prox-linear step involving linearization of $F(\mathbf{x},\mathbf{y})$ with respect to $\mathbf{x}$ and a penalized cost-to-move term $\frac{1}{2s}\|\mathbf{x}-\mathbf{x}_t\|^2$, followed by an ascent step to update the Lagrange dual variable $S^{\mathbf{x}}$. Then $\hat{\mathbf{x}}_{t+1}$ is found using prox-linear step similar to the first step but using the recent $S_{t+1}^{\mathbf{x}}$ to further correct the direction. Finally $\mathbf{x}_{t+1}$ is found by a prox step where $\text{prox}_{sG}(\mathbf{x}) = \arg\min_{\mathbf{u}\in\mathbb{R}^{md_x}} G(\mathbf{u}) + \frac{1}{2s}\|\mathbf{u}-\mathbf{x}\|^2$. Letting $D_t^{\mathbf{x}} = (U\otimes I_{d_x})S_t^{\mathbf{x}}$, and pre-multiplying by $U\otimes I_{d_x}$ in the update step of $S^{\mathbf{x}}$, $S_{t+1}^{\mathbf{x}}$ update reduces to $D_{t+1}^{\mathbf{x}} = D_t^{\mathbf{x}} + \frac{\gamma}{2s}((I_m-W)\otimes I_{d_x})\nu_{t+1}^{\mathbf{x}}$. Now using $\nu_{t+1}^{\mathbf{x}}$ and $D_{t+1}^{\mathbf{x}}$ updates, we can further reduce $\hat{\mathbf{x}}_{t+1}$ update to $\hat{\mathbf{x}}_{t+1} = \nu_{t+1}^{\mathbf{x}} - \frac{\gamma}{2}((I_m-W)\otimes I_{d_x})\nu_{t+1}^{\mathbf{x}}$. Similarly, the updates to $\mathbf{y}, S^{\mathbf{y}}$ can be done using appropriate gradient ascent-descent steps which lead to corresponding equations (D1). Further the update of $D_{t+1}^{\mathbf{y}}$ analogous to $D_{t+1}^{\mathbf{x}}$ update can be obtained by letting $D_t^{\mathbf{y}} = -(U\otimes I_{d_y})S_t^{\mathbf{y}}$.

A similar update process is explored in [15], however for solving *convex minimization problems* only. The dual variable in [15] is simpler since it arises from the Lagrangian formulation of consensus constrained minimization problem and appears only as linear term in the Lagrangian function. However in our work, the Lagrangian function in eq. (2) is not in general linear in the dual variable $\mathbf{y}$ despite the linear terms associated with Lagrange multipliers $S^{\mathbf{x}}, S^{\mathbf{y}}$. Hence updates (P1) and (D1) in our work need to tackle the original primal dual pair $\mathbf{x},\mathbf{y}$ along with the Lagrange multipliers $S^{\mathbf{x}}, S^{\mathbf{y}}$ related to consensus constraints.

Observe that the terms $((I_m-W)\otimes I_{d_x})\nu_{t+1}^{\mathbf{x}}$ and $((I_m-W)\otimes I_{d_y})\nu_{t+1}^{\mathbf{y}}$ respectively in $D_{t+1}^{\mathbf{x}}$ and $D_{t+1}^{\mathbf{y}}$ updates denote the communication of $\nu_{t+1}^{\mathbf{x}}$ and $\nu_{t+1}^{\mathbf{y}}$ across the nodes. Further note that $\nu_{t+1}^{\mathbf{x}}$ and $\nu_{t+1}^{\mathbf{y}}$ need to be communicated

Updates to primal dual pair $\mathbf{x}, S^{\mathbf{x}}$:

$$\left.\begin{aligned}\nu^{\mathbf{x}}_{t+1} &= \mathbf{x}_t - s\nabla_{\mathbf{x}} F(\mathbf{x}_t, \mathbf{y}_t) - s(U \otimes I_{d_x}) S^{\mathbf{x}}_t \\ S^{\mathbf{x}}_{t+1} &= S^{\mathbf{x}}_t + \frac{\gamma}{2s}(U \otimes I_{d_x})\nu^{\mathbf{x}}_{t+1} \\ \hat{\mathbf{x}}_{t+1} &= \mathbf{x}_t - s\nabla_{\mathbf{x}} F(\mathbf{x}_t, \mathbf{y}_t) - s(U \otimes I_{d_x}) S^{\mathbf{x}}_{t+1} \\ \mathbf{x}_{t+1} &= \underset{sG}{\text{prox}}(\hat{\mathbf{x}}_{t+1}).\end{aligned}\right\} \quad \text{(P1)}$$

Updates to dual primal pair $\mathbf{y}, S^{\mathbf{y}}$:

$$\left.\begin{aligned}\nu^{\mathbf{y}}_{t+1} &= \mathbf{y}_t + s\nabla_{\mathbf{y}} F(\mathbf{x}_t, \mathbf{y}_t) - s(U \otimes I_{d_x}) S^{\mathbf{y}}_t \\ S^{\mathbf{y}}_{t+1} &= S^{\mathbf{y}}_t - \frac{\gamma}{2s}(U \otimes I_{d_y})\nu^{\mathbf{y}}_{t+1} \\ \hat{\mathbf{y}}_{t+1} &= \mathbf{y}_t + s\nabla_{\mathbf{y}} F(\mathbf{x}_t, \mathbf{y}_t) - s(U \otimes I_{d_x}) S^{\mathbf{y}}_{t+1} \\ \mathbf{y}_{t+1} &= \underset{sR}{\text{prox}}(\hat{\mathbf{y}}_{t+1}).\end{aligned}\right\} \quad \text{(D1)}$$

only once for updating $D^{\mathbf{x}}_{t+1}, \hat{\mathbf{x}}_{t+1}$ and $D^{\mathbf{y}}_{t+1}, \hat{\mathbf{y}}_{t+1}$ To improve the communication efficiency further, we propose to compress $\nu^{\mathbf{x}}_{t+1}$ and $\nu^{\mathbf{y}}_{t+1}$ using a compression module (COMM procedure [20]) as illustrated in Algorithm 1. Algorithm 2 illustrates the proposed Inexact Primal Dual Hybrid Gradient (IPDHG) method with compression. In the next section, we state assumptions useful for further discussions.

Algorithm 1 Compressed Communication Procedure (COMM) [20]

INPUT: $\nu_{t+1}, H_t, H^w_t, \alpha$
$Q^i_t = Q(\nu^i_{t+1} - H^i_t)$ (compression)
$\hat{\nu}^i_{t+1} = H^i_t + Q^i_t$,
$H^i_{t+1} = (1-\alpha)H^i_t + \alpha\hat{\nu}^i_{t+1}$,
$\hat{\nu}^{i,w}_{t+1} = H^{i,w}_t + \sum_{j=1}^m W_{ij} Q^j_t$, (communicating compressed vectors)
$H^{i,w}_{t+1} = (1-\alpha)H^{i,w}_t + \alpha\hat{\nu}^{i,w}_{t+1}$,
RETURN: $\hat{\nu}^i_{t+1}, \hat{\nu}^{i,w}_{t+1}, H^i_{t+1}, H^{i,w}_{t+1}$ for each node i .

Algorithm 2 Inexact Primal Dual Hybrid Gradient method with compression using stochastic gradient oracle $\mathcal{G}$ (IPDHG)

INPUT: x, y, D^{x}, D^{y}, H^{x}, H^{y}, $\mathsf{H}^{\mathsf{w},\mathsf{x}}$, $\mathsf{H}^{\mathsf{w},\mathsf{y}}$, s, γ_{x}, γ_{y}, α_{x}, α_{y}, $\mathcal{G}$
Compute gradients $\mathcal{G}^{\mathsf{x}}$ and $\mathcal{G}^{\mathsf{y}}$ at (x,y) via oracle $\mathcal{G} = (\mathcal{G}^{\mathsf{x}}, \mathcal{G}^{\mathsf{y}})$
$\nu^{\mathsf{x}} = \mathsf{x} - s\mathcal{G}^{\mathsf{x}} - s\mathsf{D}^{\mathsf{x}}$
$\hat{\nu}^{\mathsf{x}}, \hat{\nu}^{\mathsf{w},\mathsf{x}}, \mathsf{H}^{\mathsf{x}}_{new}, \mathsf{H}^{\mathsf{w},\mathsf{x}}_{new} = \text{COMM}\,(\nu^{\mathsf{x}}, \mathsf{H}^{\mathsf{x}}, \mathsf{H}^{\mathsf{w},\mathsf{x}}, \alpha_{\mathsf{x}})$
$\mathsf{D}^{\mathsf{x}}_{new} = \mathsf{D}^{\mathsf{x}} + \frac{\gamma_{\mathsf{x}}}{2s}(\hat{\nu}^{\mathsf{x}} - \hat{\nu}^{\mathsf{w},\mathsf{x}})$
$\hat{\mathsf{x}} = \nu^{\mathsf{x}} - \frac{\gamma_{\mathsf{x}}}{2}(\hat{\nu}^{\mathsf{x}} - \hat{\nu}^{\mathsf{w},\mathsf{x}})$
$\mathsf{x}_{new} = \text{prox}_{sG}(\hat{\mathsf{x}})$
$\nu^{\mathsf{y}} = \mathsf{y} + s\mathcal{G}^{\mathsf{y}} - s\mathsf{D}^{\mathsf{y}}$
$\hat{\nu}^{\mathsf{y}}, \hat{\nu}^{\mathsf{w},\mathsf{y}}, \mathsf{H}^{\mathsf{y}}_{new}, \mathsf{H}^{\mathsf{w},\mathsf{y}}_{new} = \text{COMM}\,(\nu^{\mathsf{y}}, \mathsf{H}^{\mathsf{y}}, \mathsf{H}^{\mathsf{w},\mathsf{y}}, \alpha_{\mathsf{y}})$
$\mathsf{D}^{\mathsf{y}}_{new} = \mathsf{D}^{\mathsf{y}} + \frac{\gamma_{\mathsf{y}}}{2s}(\hat{\nu}^{\mathsf{y}} - \hat{\nu}^{\mathsf{w},\mathsf{y}})$
$\hat{\mathsf{y}} = \nu^{\mathsf{y}} - \frac{\gamma_{\mathsf{y}}}{2}(\hat{\nu}^{\mathsf{y}} - \hat{\nu}^{\mathsf{w},\mathsf{y}})$
$\mathsf{y}_{new} = \text{prox}_{sR}(\hat{\mathsf{y}})$
RETURN: $\mathsf{x}_{new}, \mathsf{y}_{new}, \mathsf{D}^{\mathsf{x}}_{new}, \mathsf{D}^{\mathsf{y}}_{new}, \mathsf{H}^{\mathsf{x}}_{new}, \mathsf{H}^{\mathsf{y}}_{new}, \mathsf{H}^{\mathsf{w},\mathsf{x}}_{new}, \mathsf{H}^{\mathsf{w},\mathsf{y}}_{new}$

III. Assumptions

We make the following assumptions, which would be useful throughout this work.

Assumption 1. *Each $f_i(\cdot, y)$ is μ_x-strongly convex for every $y \in \mathbb{R}^{d_y}$; hence for any $x_1, x_2 \in \mathbb{R}^{d_x}$ and fixed $y \in \mathbb{R}^{d_y}$, it holds: $f_i(x_1, y) \geq f_i(x_2, y) + \langle \nabla_x f_i(x_2, y), x_1 - x_2\rangle + \frac{\mu_x}{2}\|x_1 - x_2\|^2$.*

Assumption 2. *Each $f_i(x, \cdot)$ is μ_y-strongly concave for every $x \in \mathbb{R}^{d_x}$; hence for any $y_1, y_2 \in \mathbb{R}^{d_y}$ and fixed $x \in \mathbb{R}^{d_x}$, it holds: $f_i(x, y_1) \leq f_i(x, y_2) + \langle \nabla_y f_i(x, y_2), y_1 - y_2\rangle - \frac{\mu_y}{2}\|y_1 - y_2\|^2$.*

Assumption 3. *$g(x)$ and $r(y)$ are proper, convex and possibly non-smooth functions.*

Assumption 4. *The compression operator Q (see Algorithm 1) satisfies the following for every $u \in \mathbb{R}^d$: (i) $Q(u)$ is an unbiased estimate of u: $E[Q(u)] = u$ (ii) $E[\|Q(u) - u\|^2] \leq \delta\|u\|^2$, where the constant $\delta \geq 0$ denotes the amount of compression induced by operator Q and is called a compression factor. When $\delta = 0$, Q achieves no compression.*

Assumption 5. *Weight matrix W is symmetric, row stochastic and $W_{ij} > 0$ if and only if $(i,j) \in \mathcal{E}$ and $W_{ii} > 0$ for all $i \in [m]$. Eigenvalues of W denoted by $\lambda_1, \ldots, \lambda_m$ satisfy: $-1 < \lambda_m \leq \lambda_{m-1} \leq \ldots \leq \lambda_2 < \lambda_1 = 1$.*

Assumption 6. *Assume that each $f_{ij}(x, y)$ is L_{xx} smooth in x, i.e. for every fixed y, $\|\nabla_x f_{ij}(x_1, y) - \nabla_x f_{ij}(x_2, y)\| \leq L_{xx}\|x_1 - x_2\|, \forall x_1, x_2 \in \mathbb{R}^{d_x}$.*

Assumption 7. *Assume that each $f_{ij}(x, y)$ is L_{yy} smooth in y, i.e. for every fixed x, $\|\nabla_y f_{ij}(x, y_1) - \nabla_y f_{ij}(x, y_2)\| \leq L_{yy}\|y_1 - y_2\|, \forall y_1, y_2 \in \mathbb{R}^{d_y}$.*

Assumption 8. *Assume that each $\nabla_x f_{ij}(x, y)$ is L_{xy} Lipschitz in y, i.e. for every fixed x, $\|\nabla_x f_{ij}(x, y_1) - \nabla_x f_{ij}(x, y_2)\| \leq L_{xy}\|y_1 - y_2\|, \forall y_1, y_2 \in \mathbb{R}^{d_y}$.*

Assumption 9. *Assume that each $\nabla_y f_{ij}(x, y)$ is L_{yx} Lipschitz in x, i.e. for every fixed y, $\|\nabla_y f_{ij}(x_1, y) - \nabla_y f_{ij}(x_2, y)\| \leq L_{yx}\|x_1 - x_2\|, \forall x_1, x_2 \in \mathbb{R}^{d_x}$.*

Note that Assumptions 1-3 and Assumptions 5-9 are standard in the study of saddle point problems (e.g. [3], [4], [19], [23]). Assumption 4 is also standard in existing works (e.g. [1], [15], [20]).

IV. Understanding Early Stage Behavior of IPDHG with SVRGO and GSGO

In this section, we first recap SVRG oracle and then draw key observations on the behavior of IPDHG with SVRG oracle. We refer to IPDHG with SVRGO as Decentralized Proximal Stochastic Variance Reduced Gradient algorithm with Compression (C-DPSVRG).

We assume that each local function $f_i(x, y)$ is of the form $\frac{1}{n}\sum_{j=1}^n f_{ij}(x, y)$ where $f_{ij}(x, y)$ represents the loss function at j-th batch of samples at node i. This type of structure can be seen for instance in empirical risk minimization problems [5]. For simplicity, we assume that each node i has same number of batches n. However, our analysis easily extends to different number of batches n_i. Let N_ℓ denote the number of ℓocal samples at each node i. Then number of samples in the function component f_{ij} is determined by the batch size $B = N_\ell/n$. Let $\mathcal{P}_i = \{p_{il} : l \in \{1, 2, \ldots, n\}\}$ denote a probability distribution where p_{il} is the probability with which batch l is sampled at node i. Let $p_{\min} := \min_{i,l} p_{il}$. Without loss of generality we assume that $p_{\min} > 0$, hence each batch is chosen with a positive probability. Inspired from [9], [13], we consider stochastic variance reduced gradient oracle comprising the following steps to compute stochastic gradients in IPDHG:

Stochastic Variance Reduced Gradient Oracle (SVRGO):
(1). Index sampling: Sample $l \in \{1, 2, \ldots, n\} \sim \mathcal{P}_i$ for every node i.
(2). Stochastic gradient computation with variance reduction: For a reference point $\tilde{z}^i = (\tilde{x}^i, \tilde{y}^i)$, compute stochastic gradients at $z^i = (x^i, y^i)$ with respect to x and y as follows:

$$\mathcal{G}^{i,x} = \frac{1}{np_{il}}(\nabla_x f_{il}(z^i) - \nabla_x f_{il}(\tilde{z}^i)) + \nabla_x f_i(\tilde{z}^i),$$

$$\mathcal{G}^{i,y} = \frac{1}{np_{il}}(\nabla_y f_{il}(z^i) - \nabla_y f_{il}(\tilde{z}^i)) + \nabla_y f_i(\tilde{z}^i).$$

(3). Reference point update: Sample $\omega \sim$ Bernoulli(p) and update the reference point $\tilde{z}_i$ as follows:

$$\tilde{x}^i \longleftarrow \omega x^i + (1-\omega)\tilde{x}^i, \quad \tilde{y}^i \longleftarrow \omega y^i + (1-\omega)\tilde{y}^i.$$

As described above, SVRGO evaluates stochastic gradients at current iterate z_t^i and reference point $\tilde{z}_t^i$ as follows:

$$\mathcal{G}_t^{i,x} = \underbrace{\frac{1}{np_{il}}\left(\nabla_x f_{il}(z_t^i) - \nabla_x f_{il}(\tilde{z}_t^i)\right)} + \nabla_x f_i(\tilde{z}_t^i). \quad (4)$$

SVRGO contains an expensive but key component $\nabla_x f_i(\tilde{z}_t^i)$ obtained using full batch gradient evaluation to reduce the variance in stochastic gradients. Until the current iterate z_t^i and reference point $\tilde{z}_t^i$ start converging to saddle point, there is a gradient approximation error captured by first term in (4). Therefore, full batch gradients evaluated at the early iterations are not effective due to large distance between early iterates and saddle point solution. In addition, SVRGO evaluates on an average $2B + pN_\ell$ gradients per iterate. This phenomenon leads to slow convergence of C-DPSVRG in the initial stage with high computational cost. In this work, we propose a remedy to improve the early stage slow convergence of C-DPSVRG using a general stochastic gradient oracle described below:

General Stochastic Gradient Oracle (GSGO):
(1). Index sampling: Sample $l \in \{1, 2, \ldots, n\} \sim \mathcal{P}_i$ for every node i.
(2). Stochastic gradient computation: Compute stochastic gradients at $z^i = (x^i, y^i)$ with respect to x and y as follows:

$$\mathcal{G}^{i,x} = \frac{1}{np_{il}}\nabla_x f_{il}(z^i), \quad \mathcal{G}^{i,y} = \frac{1}{np_{il}}\nabla_y f_{il}(z^i).$$

Stochastic gradient descent ascent (SGDA) [31], [16], [19], [30], [34] which uses GSGO or its variant is the workhorse of several algorithms due to its promising fast convergence in the initial phase of iterate updates along with low computational cost. Though SGDA exhibits slow convergence or saturation behavior asymptotically due to inherent variance in the stochastic gradients, its impressive behavior in initial phase motivates us to exploit GSGO in Algorithm 2 to obtain fast convergence along with low computation cost during the initial iterate updates. Incorporating GSGO in IPDHG scheme pushes the iterates to a region close to the saddle point solution which can potentially make the full batch gradients in SVRGO more effective. Leveraging fast early convergence using GSGO and fast asymptotic convergence using SVRGO, we propose a switching algorithm which uses GSGO in IPDHG for a fixed number of initial iterations and then switches to SVRGO to achieve highly accurate solution. Before discussing the algorithm details, we examine this behavior empirically on robust logistic regression problem (10). In Figure 1, the behavior of C-DPSVRG on (10) at every iterate update is compared with a switching scheme where GSGO is used for first T_i iterate updates followed by SVRGO. We can clearly see that iterates of C-DPSVRG (blue line) make little progress in initial stage whereas the use of GSGO leads to faster progress of iterates towards saddle point solution with much lesser gradient computations. We further observe that larger values of T_i lead to clear

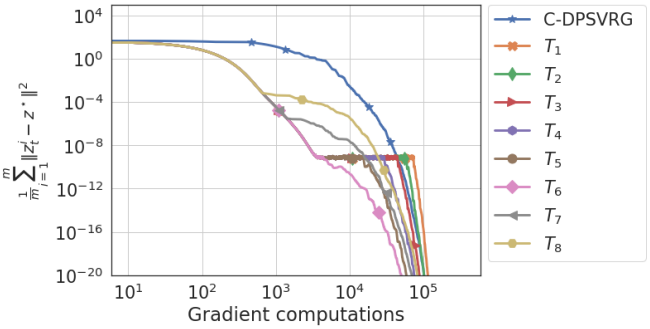

Fig. 1. Iterate convergence behavior of IPDHG using GSGO and SVRGO on robust logistic regression problem. $T_1 > T_2 > T_3 > T_4 > T_5 > T_6 > T_7 > T_8$ are the switching points from GSGO to SVRGO in IPDHG

saturation of progress of iterates due to ineffective GSGO update steps after a while, whereas small values of T_i cause early switching to SVRGO. Thus choosing a right switching point is crucial to harness the effectiveness of both GSGO and SVRGO, which we will discuss in the next section.

V. IPDHG with Switching between Stochastic Gradient Oracles

In the previous section, we have seen the advantage of leveraging GSGO in the initial stage of IPDHG iterate updates. We are now ready to describe our novel switching algorithm to solve (SPP). Starting from initial points x_0, y_0, each node i updates its primal and dual variables using IPDHG with GSGO as illustrated in Steps 5-6 of Algorithm 3. This process is repeated for the first T_0 iterations. After T_0 iterations, each node i switches to SVRGO with reference point $\tilde{z}_{T_0}^i$ initialized to $(x_{T_0}^i, y_{T_0}^i)$ and performs IPDHG updates using SVRGO for the remaining $T - T_0$ iterations. We see that Algorithm 3 requires the knowledge of switching point T_0. To address this, we first analyze the behavior of Algorithm 3 during the first T_0 iterations in the following lemma.

Lemma 1. *Let $\{\mathbf{x}_t\}_t, \{\mathbf{y}_t\}_t$ be the sequences generated by Algorithm 3. Suppose Assumptions 1-9 hold. Then for every $0 \le t \le T_0 - 1$:*

$$E_0[\Phi_{t+1}] \le (\rho_0)^{t+1}\Phi_0 + \frac{2s_0^2(C_x + C_y)}{(1-\rho_0)n^2 p_{\min}}, \quad (5)$$

where E_0 denotes the total expectation when $t \leq T_0 - 1$, Φ_t denotes the distance of the iterates $\mathbf{x}_t, \mathbf{y}_t, D_t^{\mathbf{x}}, D_t^{\mathbf{y}}, H_t^{\mathbf{x}}, H_t^{\mathbf{y}}$ from their respective limit points (described in eq. (123) in [27]), $\rho_0 \in (0,1)$ is a problem dependent parameter defined in eq. (141) in [27] and $C_x = \sum_{i=1}^{m}\sum_{l=1}^{n} \|\nabla_x f_{il}(z^\star)\|^2$, $C_y = \sum_{i=1}^{m}\sum_{l=1}^{n} \|\nabla_y f_{il}(z^\star)\|^2$.

Due to space considerations, the proof of Lemma 1 is provided in Appendix XIV-B [27]. Note that C_x and C_y in (5) appear due to variance in the stochastic gradients. If $\mathcal{G}_{x,t}^i, \mathcal{G}_{y,t}^i$ are set to be $\nabla_x f_i(x_t^i, y_t^i)$ and $\nabla_y f_i(x_t^i, y_t^i)$ respectively in Algorithm 3 for $t \leq T_0 - 1$, the second term in the r.h.s of (5) will be absent. Consequently, using Lemma 1, IPDHG converges to ϵ-accurate saddle point solution with linear rate. Without loss of generality, we assume that C_x and C_y are positive. We can also see that C_x and C_y are bounded because $z^\star$ is unique since each f_i is assumed to be strongly convex in x and strongly concave in y.

Lemma 1 indicates that Algorithm 3 has an error term (second term in the bound in RHS) due to the variance in stochastic gradients accumulated in the course of t iterations. This shows that IPDHG with GSGO returns an approximate solution asymptotically. By choosing large $t = T_0$, the first term in the RHS of (5) can be made sufficiently small. However, there might be wastage of iterations once the iterates converge in the neighborhood of saddle point solution as demonstrated in Figure 1. Further, choosing small T_0 might not exploit the full potential of GSGO in the early stage. To address this situation, we propose to choose T_0 such that there is a sufficient drift from the initial value Φ_0. We introduce a hyperparameter $\epsilon_0 \in (0,1)$ to achieve this and set T_0 such that $\rho_0^{T_0} = \epsilon_0$. This reduces the upper bound of $E_0[\Phi_{T_0}]$ to $\epsilon_0\Phi_0 + \frac{2s_0^2(C_x+C_y)}{(1-\rho_0)n^2 p_{\min}}$. We still have an unanswered question on the choice of ϵ_0. We exploit the convergence behavior of Algorithm 3 for $t \geq T_0$ to find a suitable ϵ_0 and hence the switching point T_0.

A. Determining Switching Point

As discussed earlier, the switching point T_0 depends on a hyperparameter ϵ_0. Using Lemma 1 and the choice of T_0, it is clear that the upper bound on $E_0[\Phi_{T_0}]$ increases linearly with ϵ_0. However, the effect of ϵ_0 on the convergence behavior of Algorithm 3 is not clear after GSGO is switched to SVRGO (i.e. $t \geq T_0$). We investigate this effect in Lemma 2, which further paves the way for determining a suitable value of ϵ_0.

Lemma 2. *Let $\{\mathbf{x}_t\}_t, \{\mathbf{y}_t\}_t$ be the sequences generated by Algorithm 3. Suppose Assumptions 1-9 hold. Then for any $T \geq T_0 + 1$:*

$$E[\tilde{\Phi}_T] \leq C_{\max}\Big(\frac{\epsilon_0\Phi_0}{\rho^{T_0}}\rho^T + \frac{V_e\rho^T}{\epsilon_0}\Big) + \frac{C_1\rho^T}{\epsilon_0}, \tag{6}$$

where $\tilde{\Phi}_T$ denotes the distance of the iterates $\mathbf{x}_T, \tilde{\mathbf{x}}_T, \mathbf{y}_T, \tilde{\mathbf{y}}_T, D_T^{\mathbf{x}}, D_T^{\mathbf{y}}, H_T^{\mathbf{x}}, H_T^{\mathbf{y}}$ from their respective limit points (described in eq. (206) in [27]), $V_e = \frac{2s_0^2(C_x+C_y)}{(1-\rho_0)n^2 p_{\min}}$ and $C_{\max}$, C_1, $\rho \in (0,1)$ are problem dependent constant parameters defined in equations (262), (263) and (205) [27].

Lemma 2 which is proven in Appendix XVI [27], leads to an upper bound on $E[\tilde{\Phi}_T]$ when GSGO switches to SVRGO at T_0 in Algorithm 3. This upper bound is small for sufficiently large T and hence iterates x_T^i, y_T^i are also close to saddle point solution $x^\star, y^\star$ in expectation according to the definition of $\tilde{\Phi}_T$. We recall that T_0 is chosen such that a sufficient progress is obtained from initial value Φ_0. Now using the facts that $T \geq T_0 + 1$, $\rho \in (0,1)$, (6) reduces to

$$E[\tilde{\Phi}_T] \leq C_{\max}(\epsilon_0\Phi_0 + \frac{V_e\rho^T}{\epsilon_0}) + \frac{C_1\rho^T}{\epsilon_0}. \tag{7}$$

Interestingly, the expected value of $\tilde{\Phi}_T$ in (7) is upper bounded by an ϵ_0-dependent quantity which attains its minimum value at $\epsilon_0^\star = \sqrt{\frac{(C_{\max}V_e + C_1)\rho^T}{C_{\max}\Phi_0}}$, where T is the total number of iterations used in Algorithm 3. One natural way is to set T to be the total number of iterations $T(\epsilon)$ required to achieve an ϵ-accurate saddle point solution.

Computing $T(\epsilon)$: By substituting $\epsilon_0 = \epsilon_0^\star$ in (7), we get $E[\tilde{\Phi}_T] \leq 2\sqrt{C_{\max}\Phi_0(C_{\max}V_e + C_1)\rho^T}$. As a consequence, after $T(\epsilon) = \frac{2}{-\log\rho}\log(\frac{2\sqrt{C_{\max}\Phi_0(C_{\max}V_e+C_1)}}{\epsilon})$ iterations, Algorithm 3 returns an ϵ-accurate saddle point solution in expectation. Therefore, we get $\epsilon_0^\star = \frac{\epsilon}{2C_{\max}\Phi_0}$, and hence $T_0 = \lceil \frac{1}{\log\rho_0}\log(\frac{\epsilon}{2C_{\max}\Phi_0})\rceil$. We see that the value of T_0 depends on

$$\begin{aligned}\Phi_0 = {} & M_{x,0}\|\mathbf{x}_0 - (\mathbf{1}\otimes I_{d_x})x^\star\|^2 + M_{y,0}\|\mathbf{y}_0 - (\mathbf{1}\otimes I_{d_y})y^\star\|^2 \\ & + \frac{2s_0^2}{\gamma_{x,0}}\|((I-J)\otimes I_{d_x})\nabla_x F(\mathbf{z}^\star)\|^2_{(I-W)^\dagger} \\ & + \frac{2s_0^2}{\gamma_{y,0}}\|((I-J)\otimes I_{d_y})\nabla_y F(\mathbf{z}^\star)\|^2_{(I-W)^\dagger} \\ & + \sqrt{\delta}\|\mathbf{x}_0 - (\mathbf{1}\otimes I_{d_x})x^\star + \frac{s_0}{m}(\mathbf{1}\otimes I_{d_x})\nabla_x f(z^\star)\|^2 \\ & + \sqrt{\delta}\|\mathbf{y}_0 - (\mathbf{1}\otimes I_{d_y})y^\star - \frac{s_0}{m}(\mathbf{1}\otimes I_{d_y})\nabla_y f(z^\star)\|^2.\end{aligned} \tag{8}$$

It is clear that computing Φ_0 requires knowledge of the saddle point solution $z^\star$ which is not available in practice. We also

Algorithm 3 **D**ecentralized **P**roximal **S**witching **S**tochastic **G**radient method with **C**ompression (C-DPSSG)

INPUT: $\mathbf{x}_0 = (\mathbf{1}\otimes I_{d_x})x_0, \mathbf{y}_0 = (\mathbf{1}\otimes I_{d_y})y_0, D_0^{\mathbf{x}} = D_0^{\mathbf{y}} = \mathbf{0}, H_0^{\mathbf{x}} = \mathbf{x}_0, H_0^{\mathbf{y}} = \mathbf{y}_0, H_0^{\mathbf{w},\mathbf{x}} = (W\otimes I_{d_x})\mathbf{x}_0, H_0^{\mathbf{w},\mathbf{y}} = (W\otimes I_{d_y})\mathbf{y}_0, s_0 = \frac{np_{\min}}{4\sqrt{2}L\kappa_f}, s = \frac{\mu np_{\min}}{24L^2}$, $\gamma_{x,0}, \gamma_x, \gamma_{y,0}, \gamma_y, \alpha_{x,0}, \alpha_x, \alpha_{y,0}, \alpha_y$ defined in Appendices XIV-A and XV-A [27], switching point T_0.
for $t = 0$ **to** $T-1$ **do**
  Sample $l \in \{1,2,\ldots,n\} \sim \mathcal{P}_i$ for every node i
  if $t \leq T_0 - 1$ **then**
    $\mathcal{G}_t^{i,x} = \frac{1}{np_{il}}\nabla_x f_{il}(z_t^i)$ and $\mathcal{G}_t^{i,y} = \frac{1}{np_{il}}\nabla_y f_{il}(z_t^i)$ for every node i
    $\mathbf{x}_{t+1}, \mathbf{y}_{t+1}, D_{t+1}^{\mathbf{x}}, D_{t+1}^{\mathbf{y}}, H_{t+1}^{\mathbf{x}}, H_{t+1}^{\mathbf{y}}, H_{t+1}^{\mathbf{w},\mathbf{x}}, H_{t+1}^{\mathbf{w},\mathbf{y}}$ =IPDHG$(\mathbf{x}_t, \mathbf{y}_t, D_t^{\mathbf{x}}, D_t^{\mathbf{y}}, H_t^{\mathbf{x}}, H_t^{\mathbf{y}}, H_t^{\mathbf{w},\mathbf{x}}, H_t^{\mathbf{w},\mathbf{y}}, s_0, \gamma_{x,0}, \gamma_{y,0}, \alpha_{x,0}, \alpha_{y,0}, \mathcal{G}_t)$
  else
    $\tilde{\mathbf{x}}_{T_0} = \mathbf{x}_{T_0}$, $\tilde{\mathbf{y}}_{T_0} = \mathbf{y}_{T_0}$
    Compute stochastic gradients $\mathcal{G}_t^{i,x}$ and $\mathcal{G}_t^{i,y}$ using SVRGO for every node i
    $\mathbf{x}_{t+1}, \mathbf{y}_{t+1}, D_{t+1}^{\mathbf{x}}, D_{t+1}^{\mathbf{y}}, H_{t+1}^{\mathbf{x}}, H_{t+1}^{\mathbf{y}}, H_{t+1}^{\mathbf{w},\mathbf{x}}, H_{t+1}^{\mathbf{w},\mathbf{y}}$ =IPDHG$(\mathbf{x}_t, \mathbf{y}_t, D_t^{\mathbf{x}}, D_t^{\mathbf{y}}, H_t^{\mathbf{x}}, H_t^{\mathbf{y}}, H_t^{\mathbf{w},\mathbf{x}}, H_t^{\mathbf{w},\mathbf{y}}, s, \gamma_x, \gamma_y, \alpha_x, \alpha_y, \mathcal{G}_t)$
  end if
end for
RETURN: $\mathbf{x}_T, \mathbf{y}_T$.

emphasize that Φ_0 is a global quantity as it depends on the full gradient information of $f(x,y)$ which is inaccessible to the nodes. We address this issue by proposing a practical version of Algorithm 3 which approximates Φ_0 using local information and without the knowledge of $z^\star$.

B. Practical Approach for Determining Switching Point

To determine the switching point in Algorithm 3, we discuss a practical scheme whose broad idea is illustrated in Figure 2. This scheme consists of the following steps. We first allow IPDHG with GSGO to perform $T_0^{'} = \lceil \frac{\log 2}{-\log \rho_0} \rceil$ iterations to obtain primal iterate $\mathbf{x}_{T_0'}$ and dual iterate $\mathbf{y}_{T_0'}$. These primal dual iterates might saturate to a point in $T_0^{'}$ iterations due to the nature of GSGO. To detect this behavior, each node i computes the distance between the last two successive iterates and gets an approximation of the average quantity $m^{-1}\sum_{i=1}^m \|z^i_{T_0'} - z^i_{T_0'-1}\|^2$ using the accelerated gossip scheme [18]. If the average distance is less than a suitable small threshold for atleast one node, then each node i switches to SVRGO. If a fraction of nodes find the average distance to be within the threshold value, then this information can be spread to the entire network in maximum number of hops of order $\mathcal{O}(m)$. On the other hand, if the average distance is above threshold value, then we consider $\mathbf{z}_{T_0'}$ as a proxy of $\mathbf{z}^\star$ to compute approximation of Φ_0. Using this procedure, the last two terms of Φ_0 depend on global gradients $\frac{1}{m}\sum_{i=1}^m \nabla_x f_i(x^i_{T_0'}, y^i_{T_0'})$ and $\frac{1}{m}\sum_{i=1}^m \nabla_y f_i(x^i_{T_0'}, y^i_{T_0'})$. Each node can now run accelerated gossip scheme on local gradients $\nabla_x f_i(x^i_{T_0'}, y^i_{T_0'})$ and $\nabla_y f_i(x^i_{T_0'}, y^i_{T_0'})$ to achieve approximations $\tilde{\mathcal{G}}^{i,x}_{T_0'}$ and $\tilde{\mathcal{G}}^{i,y}_{T_0'}$ of the global gradients. Finally, each node i approximates Φ_0 by average quantity $\bar{\Phi}^i_0(T_0')$ where $\{\bar{\Phi}^i_0(T_0')\}_{i=1}^m$ are obtained using accelerated gossip scheme on local scalar values $\Phi^i_0(T_0')$ given by:

$$\begin{aligned}\Phi^i_0(T_0') &= M_{x,0}\|x_0 - x^i_{T_0'}\|^2 + M_{y,0}\|y_0 - y^i_{T_0'}\|^2 \\ &+ \frac{2s_0^2}{\gamma_{x,0}}\|\nabla_x f_i(z^i_{T_0'}) - \tilde{\mathcal{G}}^{i,x}_{T_0'}\|^2\lambda_{\max}(I-W)^\dagger \\ &+ \frac{2s_0^2}{\gamma_{y,0}}\|\nabla_y f_i(z^i_{T_0'}) - \tilde{\mathcal{G}}^{i,y}_{T_0'}\|^2\lambda_{\max}(I-W)^\dagger \\ &+ \sqrt{\delta}\|x_0 - x^i_{T_0'} + s_0\tilde{\mathcal{G}}^{i,x}_{T_0'}\|^2 + \sqrt{\delta}\|y_0 - y^i_{T_0'} - s_0\tilde{\mathcal{G}}^{i,y}_{T_0'}\|^2.\end{aligned}$$

Finally, the above process yields the approximated value of switching point as $T_0^i = \lceil \frac{\log \bar{\epsilon}^i_0}{\log \rho_0} \rceil = \lceil \frac{1}{\log \rho_0}\log(\frac{\epsilon}{2C_{\max}\bar{\Phi}^i_0(T_0')}) \rceil$. We note that values $\bar{\Phi}^i_0(T_0')$ are close to each other because they are the outputs of gossip scheme and hence values T_0^i are also similar for all nodes. A concise form of this entire procedure is formally demonstrated in Algorithm 4. It is worth noting that the above practical approach to detect switching point invokes gossip scheme thrice out of which two gossips are performed only on scalar values.

Approximation quality of T_0: From previous discussion, we have the approximated value $T_0^i = \lceil \frac{\log \bar{\epsilon}^i_0}{\log \rho_0} \rceil$. Whenever $\bar{\epsilon}^i_0 \approx \epsilon^\star_0$, T_0^i is a good approximation of T_0. We observe in our empirical study that the values of $\bar{\epsilon}^i_0$ and $\epsilon^\star_0$ are close to each other which in turn implies that T_0^i and T_0 are also close (see Table I in Section VII).

Algorithm 4 Practical way of determining switching point

1: Implement $T_0' = \lceil \frac{\log 2}{-\log \rho_0} \rceil$ iterations of IPDHG with GSGO and obtain $\mathbf{z}_{T_0'}, \mathbf{z}_{T_0'-1}$
2: Each node i gets approximated value $\hat{z}^i_{T_0'}$ of $m^{-1}\|\mathbf{z}_{T_0'} - \mathbf{z}_{T_0'-1}\|^2$ by invoking accelerated gossip [18] on $\{\|z^i_{T_0'} - z^i_{T_0'-1}\|^2\}_{i=1}^m$
3: **if** $\hat{z}^i_{T_0'} >$ threshold for every node i **then**
4: Each node i obtains approximation $\bar{\Phi}^i_0(T_0')$ of global quantity Φ_0 by invoking accelerated gossip on $\{\Phi^i_0(T_0')\}_{i=1}^m$ and computes T_0^i
5: Each node i continues with GSGO for remaining $T_0^i - T_0'$ iterations
6: **else**
7: Switch to SVRGO and continue using SVRGO
8: **end if**

C. Complexity of Algorithm 3

We present the iteration complexity of Algorithm 3 in Theorem 1 below.

Theorem 1. *Let $\{\mathbf{x}_t\}_t, \{\mathbf{y}_t\}_t$ be the sequences generated by Algorithm 3. Suppose Assumptions 1-9 hold. Then iteration complexity of Algorithm 3 for achieving ϵ-accurate saddle point solution in expectation is*

$$\mathcal{O}(\max\{\frac{\sqrt{\delta}(1+\delta)\kappa_g\kappa_f^2}{np_{\min}}, (1+\delta)\kappa_g, \frac{(1+\delta)\kappa_f^2}{np_{\min}}, \frac{2}{p}\} \times \log(\frac{2\sqrt{C_{\max}\Phi_0(C_{\max}V_e + C_1)}}{\epsilon})). \tag{9}$$

The proof of Theorem 1 requires significant technical background to be developed. Unfortunately, due to space constraints, we are unable to discuss relevant background details here and hence provide the proof in Appendix XVI of technical report [27]. Theorem 1 indicates that Algorithm 3 converges to ϵ-accurate saddle point solution with linear rate. Further, the complexity in Theorem 1 depends on compression factor as $\mathcal{O}(\max\{\sqrt{\delta}(1+\delta), 1+\delta\})$. Without compression ($\delta = 0$), the iteration complexity reduces to $\mathcal{O}(\max\{\kappa_g, \frac{\kappa_f^2}{np_{\min}}, \frac{2}{p}\}\log(\frac{2\sqrt{C_{\max}\Phi_0(C_{\max}V_e+C_1)}}{\epsilon}))$. The communication complexity of Algorithm 3 has a term similar to iteration complexity along with an additional number of communications required in gossip scheme at $T_0^{'}$-th iteration. In terms of gradient computations, Algorithm 3 requires $(2B + pN_\ell)T(\epsilon) - (B + pN_\ell)T_0$ gradient computations.

VI. Related Work

A distributed saddle point algorithm with Laplacian averaging (DSPAwLA) is proposed in [21] to solve non-smooth convex-concave saddle point problems. An extragradient method with gradient tracking (GT-EG) [23] is shown to converge with linear rates for strongly convex-strongly

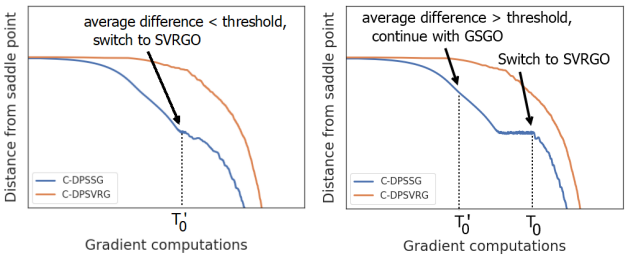

Fig. 2. Broad illustration of proposed practical method to determine switching point. Average difference = $m^{-1}\|\mathbf{z}_{T_0'} - \mathbf{z}_{T_0'-1}\|^2$

concave problems, under a positive lower bound assumption on the gradient difference norm. A distributed Min-Max data similarity (MMDS) algorithm under a suitable data similarity assumption is proposed in [4] which requires solving an inner saddle point problem at every iteration. Another work [25] has designed algorithms for smooth saddle point problems with bilinear structure. We emphasize that [21], [23], [4], [25] are based on non-compressed communications and full batch gradient computations which limit their applicability to large scale problems.

Multiple works [19], [3], [30], [12], [6], [8] have developed algorithms using stochastic gradients, albeit *without compression* for solving decentralized saddle point problems. Note that [28] has designed two different compression based algorithms using GSGO's variant and SVRGO respectively for general stochastic setting and finite sum setting. Figure 3 below helps positioning our work in context of existing methods.

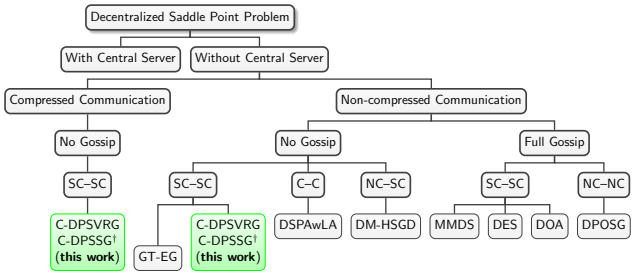

Fig. 3. Our work in comparison to existing art. GT-EG: [23], DSPAwLA: [21], DMHSGD: [30], MMDS: [4], DES: [3], DOA: [12], DPOSG: [19]. C-C, SC-SC, NC-NC, NC-SC denote respectively convex-concave, strongly convex-strongly concave, nonconvex-nonconcave, nonconvex-strongly concave. †: C-DPSSG uses gossip only for deciding switching point, not for iterate updates.

VII. Numerical Experiments

We investigate[1] the performance of proposed algorithms on robust logistic regression and AUC maximization. We rely on binary classification datasets a4a and ijcnn1 from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/. We consider a 2d torus topology of 20 nodes in all our experiments. Additional experiments for ijcnn1, phishing and sido data are presented in our technical report [27] (see Appendices XIX- XX). The performance of proposed algorithms on ring topology, convergence behavior with number of nodes and bits used for compression are also presented in [27]. We use an unbiased b-bits quantization operator $Q_\infty(\cdot)$ [20] in all the experiments.

A. Robust Logistic Regression

We consider robust logistic regression problem

$$\min_{x\in\mathcal{X}}\max_{y\in\mathcal{Y}}\frac{1}{N}\sum_{i=1}^{N}\log(1+\exp(-b_i x^\top(a_i+y)))+\frac{\lambda}{2}\|x\|_2^2-\frac{\beta}{2}\|y\|_2^2, \tag{10}$$

over a binary classification data set $\mathcal{D}=\{(a_i,b_i)\}_{i=1}^N$. The constraint sets $\mathcal{X}$ and $\mathcal{Y}$ are ℓ_2 balls of radius 100 and 1 respectively. We set number of bits $b=4$ in quantization operator $Q_\infty(x)$ and $\lambda=\beta=10$.

Switching Point: For C-DPSSG, we take threshold value to be 10^{-8} and implement 20 iterations of accelerated gossip to decide the switch to SVRGO. It turns out that C-DPSSG switches to SVRGO after performing $T_0^{'}$ iterations with GSGO because the gap between two consecutive iterates gets saturated in these many iterations.

Observations: Switching method C-DPSSG converges faster than C-DPSVRG and other baseline methods as demonstrated in Figure 4. DPOSG and DM-HSGD converge only to a neighborhood of the saddle point solution and start oscillating after a number of iterations. Note that MMDS has poor performance because it is based on full batch gradient computations and multiple calls of gossip scheme at every iterate.

B. AUC maximization

We evaluate the effectiveness of proposed algorithms on area under receiver operating characteristic curve (AUC) maximization [33] formulated as:

$$\min_{x,u,v}\max_{y}\frac{1}{N}\sum_{i=1}^{N}F(x,u,v,y;a_i,b_i)+\frac{\lambda}{2}\|x\|_2^2, \tag{11}$$

where $F(x,u,v,y;a_i,b_i)=(1-q)(a_i^\top x-u)^2\delta_{[b_i=1]}+q(a_i^\top x-v)^2\delta_{[b_i=-1]}-q(1-q)y^2+2(1+y)\left(qa_i^\top x\delta_{[b_i=-1]}-(1-q)a_i^\top x\delta_{[b_i=1]}\right)$, the fraction of positive samples is given by q. We set $\lambda=10^{-5}$ in (11) and consider constraint sets as ℓ_2 ball of radius 100 and 200 respectively on primal and dual variables.

Observations: We observe that C-DPSSG switches to SVRGO after T_0^i iterations. The AUC plots on training set in Figure 4 show that C-DPSSG achieves higher AUC value faster in terms of gradient computations, communications and bits transmitted. These observations suggest that switching scheme is beneficial over purely SVRGO based scheme for obtaining high AUC value as it saves time and gradient computations in the crucial early stage.

Data	Φ_0	$\bar{\Phi}_0^i(T_0')$	$\epsilon_0^\star$	$\bar{\epsilon}_0^i$	T_0	T_0^i	T_0'
a4a	47.4	8.6	5.3×10^{-11}	2.9×10^{-10}	195315	181248	5720
ijcnn1	23.8	3.9	10^{-10}	6.3×10^{-10}	50947	46960	1536

TABLE I

Values of $\bar{\Phi}_0^i(T_0')$, $\bar{\epsilon}_0^i$ T_0^i (observed same for all nodes) obtained from Algorithm (4) and $\epsilon_0^\star, T_0, T_0'$ on AUC maximization.

Table I reports the true values $\Phi_0, \epsilon_0^\star, T_0$ and the approximate values $\bar{\Phi}_0^i(T_0'), \bar{\epsilon}_0^i, T_0^i$ in AUC maximization. We notice that $\epsilon_0^\star$ and $\bar{\epsilon}_0^i$ are close to each other and hence the difference between T_0 and T_0^i is also small.

VIII. Conclusion

This work presents C-DPSSG, a technique that leverages the best phases of GSGO and SVRGO in a decentralized setting with compression, by performing a switch between them. The proposed algorithm offers practical advantages for efficiently obtaining low, medium and highly accurate solutions. Adapting the algorithm to cases where some constants are unknown in the problem setup would be an interesting direction to explore in future.

[1]All codes are available at https://github.com/chhavisharma123/C-DPSSG-CDC2023

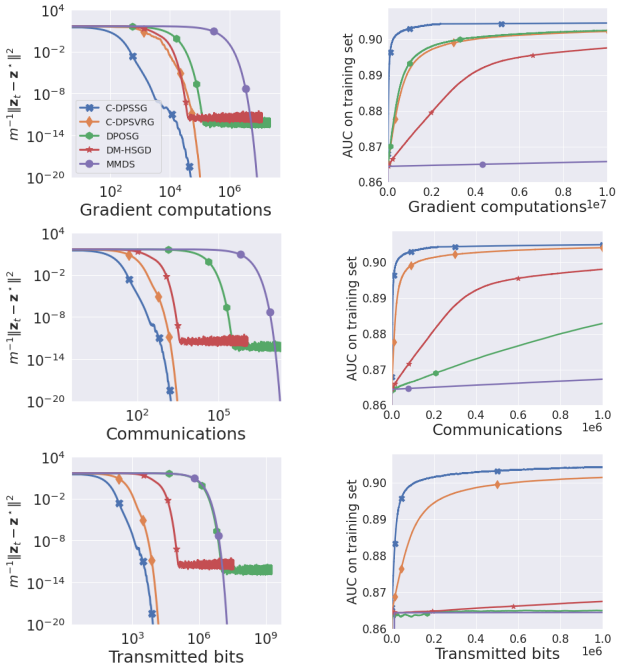

Fig. 4. Left panel: Convergence behavior of iterates to saddle point for robust logistic regression. Right panel: AUC value on training set for AUC maximization problem.

REFERENCES

[1] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, 2017.

[2] Reza Babanezhad Harikandeh, Mohamed Osama Ahmed, Alim Virani, Mark Schmidt, Jakub Konečnỳ, and Scott Sallinen. Stopwasting my gradients: Practical svrg. *Advances in Neural Information Processing Systems*, 28, 2015.

[3] Aleksandr Beznosikov, Valentin Samokhin, and Alexander Gasnikov. Distributed saddle-point problems: Lower bounds, optimal algorithms and federated gans. *arXiv preprint arXiv:2010.13112*, 2020.

[4] Aleksandr Beznosikov, Gesualdo Scutari, Alexander Rogozin, and Alexander Gasnikov. Distributed saddle-point problems under similarity. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[5] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186, Heidelberg, 2010. Physica-Verlag HD.

[6] Lesi Chen, Haishan Ye, and Luo Luo. A simple and efficient stochastic algorithm for decentralized nonconvex-strongly-concave minimax optimization. *arXiv preprint arXiv:2212.02387*, 2022.

[7] Stephen H Friedberg, Arnold J Insel, and Lawrence E Spence. *Linear algebra*. Pearson Higher Ed, 2003.

[8] Hongchang Gao. Decentralized stochastic gradient descent ascent for finite-sum minimax problems. *arXiv preprint arXiv:2212.02724*, 2022.

[9] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, 2013.

[10] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pages 3478–3487. PMLR, 2019.

[11] Galina M Korpelevich. The extragradient method for finding saddle points and other problems. *Matecon*, 12:747–756, 1976.

[12] Dmitry Kovalev, Aleksandr Beznosikov, Abdurakhmon Sadiev, Michael Igorevich Persiianov, Peter Richtárik, and Alexander Gasnikov. Optimal algorithms for decentralized stochastic variational inequalities. In *Advances in Neural Information Processing Systems*, 2022.

[13] Dmitry Kovalev, Samuel Horváth, and Peter Richtárik. Don't jump through hoops and remove those loops: Svrg and katyusha are better without the outer loop. In *Algorithmic Learning Theory*, pages 451–467. PMLR, 2020.

[14] Guanghui Lan, Soomin Lee, and Yi Zhou. Communication-efficient algorithms for decentralized and stochastic optimization. *Mathematical Programming*, 180(1):237–284, 2020.

[15] Yao Li, Xiaorui Liu, Jiliang Tang, Ming Yan, and Kun Yuan. Decentralized composite optimization with compression. *arXiv preprint arXiv:2108.04448*, 2021.

[16] Tianyi Lin, Chi Jin, and Michael Jordan. On gradient descent ascent for nonconvex-concave minimax problems. In *International Conference on Machine Learning*, pages 6083–6093. PMLR, 2020.

[17] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*, 2017.

[18] Ji Liu and A Stephen Morse. Accelerated linear iterations for distributed averaging. *Annual Reviews in Control*, 35(2):160–165, 2011.

[19] Mingrui Liu, Wei Zhang, Youssef Mroueh, Xiaodong Cui, Jerret Ross, Tianbao Yang, and Payel Das. A decentralized parallel algorithm for training generative adversarial nets, 2020.

[20] Xiaorui Liu, Yao Li, Rongrong Wang, Jiliang Tang, and Ming Yan. Linear convergent decentralized optimization with compression. In *International Conference on Learning Representations*, 2021.

[21] David Mateos-Núñez and Jorge Cortès. Distributed saddle-point subgradient algorithms with laplacian averaging. *IEEE Transactions On Automatic Control*, 62(6), 2017.

[22] Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč, and Peter Richtárik. Distributed learning with compressed gradient differences, 2019.

[23] Soham Mukherjee and Mrityunjoy Chakraborty. A decentralized algorithm for large scale min-max problems. In *59th IEEE Conference on Decision and Control (CDC)*, 2020.

[24] Balamurugan Palaniappan and Francis Bach. Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[25] Muhammad I Qureshi and Usman A Khan. Distributed saddle point problems for strongly concave-convex functions. *arXiv preprint arXiv:2202.05812*, 2022.

[26] Alexander Rogozin, Aleksandr Beznosikov, Darina Dvinskikh, Dmitry Kovalev, Pavel Dvurechensky, and Alexander Gasnikov. Decentralized distributed optimization for saddle point problems. *arXiv preprint arXiv:2102.07758*, 2021.

[27] Chhavi Sharma, Vishnu Narayanan, and P. Balamurugan. Switch and conquer: Efficient algorithms by switching stochastic gradient oracles for decentralized saddle point problems. Technical report available at arXiv link.

[28] Chhavi Sharma, Vishnu Narayanan, and P Balamurugan. Stochastic gradient methods with compressed communication for decentralized saddle point problems. *NeurIPS Workshop on Federated Learning: Recent Advances and New Challenge*, 2022.

[29] Hoi-To Wai, Zhuoran Yang, Zhaoran Wang, and Mingyi Hong. Multi-agent reinforcement learning via double averaging primal-dual optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

[30] Wenhan Xian, Feihu Huang, Yanfu Zhang, and Heng Huang. A faster decentralized algorithm for nonconvex minimax problems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[31] Yan Yan, Yi Xu, Qihang Lin, Wei Liu, and Tianbao Yang. Optimal epoch stochastic gradient descent ascent methods for min-max optimization. In *Advances in Neural Information Processing Systems*, 2020.

[32] Yan Yan, Yi Xu, Qihang Lin, Lijun Zhang, and Tianbao Yang. Stochastic primal-dual algorithms with faster convergence than $o(1/\sqrt{t})$ for problems without bilinear structure. *CoRR*, abs/1904.10112, 2019.

[33] Yiming Ying, Longyin Wen, and Siwei Lyu. Stochastic online auc maximization. *Advances in neural information processing systems*, 29, 2016.

[34] Matteo Zecchin, Marios Kountouris, and David Gesbert. Communication-efficient distributionally robust decentralized learning. *Transactions on Machine Learning Research*, 2022.

[35] Minghui Zhu and Sonia Martinez. On distributed convex optimization under inequality and equality constraints. *IEEE Transactions on Automatic Control*, 57(1):151–164, 2011.