A Decentralized Local Stochastic Extragradient Approach for Variational Inequalities

Abstract

This study examines distributed stochastic variational inequalities (VIs) within unbounded domains, where the problem data is heterogeneous, meaning it is non-identically distributed and spread across numerous devices. We adopt a broad assumption regarding the computational network, which encompasses fully decentralized computations with dynamic networks and the centralized structures commonly employed in Federated Learning. Additionally, we allow multiple local updates on the workers to reduce how often they communicate. We adapt the stochastic extragradient method to this versatile framework, and conduct theoretical analysis on its convergence rate, specifically in strongly-monotone, monotone, and non-monotone scenarios (given that a Minty solution is available). The rates we provide demonstrate a clear relationship with various network properties like mixing time, the number of iterations, data heterogeneity, variance, the quantity of devices, and other typical parameters. As a particular application, our method and analysis can be used for distributed stochastic saddle-point problems (SPP), such as the training of Deep Generative Adversarial Networks (GANs), which is known to be very difficult when using decentralized training. The experiments we perform for decentralized GANs training demonstrate the efficacy of our proposed approach.

1 Introduction

In extensive machine learning (ML) situations, training data is often split among multiple devices like data centers or mobile devices. Decentralized training methods can produce an ML model with the same accuracy as if all data were on a single server. Moreover, decentralized training has advantages over traditional centralized methods including data ownership, privacy, fault tolerance, and scalability. Federated Learning (FL) is a decentralized learning approach where the training process is managed by a single device or server that communicates with all the participating clients. However, in fully decentralized learning (FD) scenarios, devices only communicate with their neighbors via a communication network with an arbitrary structure. Therefore, decentralized algorithms are valuable when centralized communication is expensive, undesirable, or impossible.

Recently, significant advances have been made in the creation, design, and understanding of decentralized training methods. In particular, aspects such as data heterogeneity, communication efficiency, which includes local updates or compression, and personalization have been explored. However, these advancements have focused on training with single-criterion loss functions, which lead to minimization problems, and are not applicable to more general types of problems. For instance, training Generative Adversarial Networks (GANs) requires the simultaneous competing optimization of the generator and discriminator objectives, which translates to solving a non-convex-non-concave saddle-point problem (SPP). This kind of problem structure makes GANs extremely challenging to train, even in the single-node setting, let alone when training over decentralized datasets.

This study centers around solving decentralized stochastic SPPs and, more broadly, decentralized stochastic Minty variational inequalities (MVIs). In a decentralized stochastic MVI, data is distributed

.

across M or more devices/nodes. Each device m has access to its own local stochastic oracle Fm(z, m) for the local operator Fm(z) := EmDmFm(z, m). The data m in device m follows a distribution Dm, which can vary across devices. The devices are connected via a communication network, allowing two devices to exchange information only if their corresponding nodes are connected by an edge in the network graph. The objective is to find cooperatively a point z* Rn that satisfies the inequality:

$$\sum_{m=1}^{M} \mathbb{E}[F_m(z^*), z - z^*] \geq 0 \tag{1}$$

for all z Rn.

A specific instance of decentralized stochastic MVIs is the decentralized stochastic SPP with local objectives fm(x, y) := EmDm[fm(x, y, m)]:

$$\min_{x \in \mathbb{R}^n} \max_{y \in \mathbb{R}^m} \sum_{m=1}^{M} f_m(x, y) \tag{2}$$

The connection to VI can be seen by setting z = (x, y) and the gradient field F(z) = (xf(x, y), -yf(x, y)). In cases where f(x,y) is convex-concave, the operator F(z) is monotone. However, in the context of GANs training, where x and y are parameters of the generator and discriminator, respectively, the local losses fm(x, y) are generally non-convex-non-concave in x, y, and monotonicity of F cannot be assumed.

In this study, we develop a new algorithm for addressing problems (1) and (2). Because gradient descent-ascent for problem (2) can diverge even in simple convex-concave settings with a single device, we use extragradient updates and combine them with a gossip-type communication protocol on arbitrary, possibly dynamic, network topologies. One challenge arising from communication constraints is a "network error" that stems from the inability of all devices to achieve exact consensus. Therefore, each device uses a local variable, with only approximate consensus among devices achieved through gossip steps. Our method avoids multiple gossip steps per iteration, leading to better practical performance on dynamic networks. It also allows multiple local updates between communication rounds to reduce communication overhead, making it suitable for communication- and privacy-restricted FL or fully decentralized scenarios.

Our Contributions:

1. We have created an algorithm that uses extragradient updates to tackle distributed stochastic MVIs, and consequently distributed stochastic SPPs, with heterogeneous data. This framework offers a flexible communication protocol that supports centralized settings like Federated Learning, fully decentralized configurations, local steps in both centralized and decentralized setups, and dynamic network topologies.

2. Using this general communication protocol, we have demonstrated the convergence of our algorithm in three MVI settings, namely where the operator is strongly-monotone, monotone, or non-monotone (assuming a Minty condition is met). The rates of convergence depend explicitly on several problem parameters, such as network characteristics, data heterogeneity, data variance, number of devices, and other relevant factors. These theoretical results translate directly to the corresponding SPP settings (strongly-convex-strongly-concave, convex-concave, and non-convex-non-concave under the Minty condition). All theoretical results are valid when using heterogeneous data, and allow quantifying how factors like data heterogeneity, noise in the data, and network characteristics influence convergence rate. We have also shown that for decentralized settings, our results are novel for time-varying graphs and the three different monotonicity settings.

3. We have verified our theoretical results through numerical experiments and demonstrated the effectiveness of our strategy in practice. Specifically, we have trained a DCGAN architecture on the CIFAR-10 dataset.

2 Related Work

Research on MVIs dates back to at least 1962, and has been continued in recent works. VIs are used in diverse applications: image denoising, game theory and optimal control, robust optimization, and non-smooth optimization using smooth reformulations. In ML, MVIs and SPPs arise in GANs training, reinforcement learning, and adversarial training.

The extragradient method (EGM) was first introduced and later expanded to include deterministic problems and stochastic problems with bounded variance. However, if the stochastic noise is not uniformly bounded, EGM can diverge.

3 Algorithm

This section details our proposed algorithm (Algorithm 1) based on two main concepts: (i) the extragradient step (as seen in classical methods for VIs), and (ii) gossip averaging (used in decentralized optimization and diffusion strategies in distributed learning). Instead of using gradient descent, as in similar algorithms, ours uses the extragradient method. It is designed for VIs and SPPs. It also includes local steps between communication rounds, supports dynamic networks, and comes with non-asymptotic theoretical convergence guarantees.

Each step of Algorithm 1 has two phases. The local phase (lines 4–6) involves a step of the stochastic extragradient method at each node using only local data. Nodes make an extrapolation step "to look into the future" and then update using the operator value at the "future" point. Next is the communication phase (line 7), during which nodes share local iterates with their neighbors Nm in the communication network graph for each iteration k. Averaging is done using weights w k m,i, which are matrix Wk elements called the mixing matrix.

Definition 2.1 (Mixing matrix). A matrix W [0; 1]M×M is a mixing matrix if it satisfies: 1) W is symmetric, 2) W is doubly stochastic (W1 = 1, 1TW = 1T, where 1 is the vector of all ones), 3) W is aligned with the network: wij 0 if and only if i = j or the edge (i, j) is in the communication network graph.

Reasonable choices of mixing matrices include Wk = IM Lk /max(Lk), where Lk is the Laplacian matrix of the network graph at step k and IM is the identity matrix, or by using local rules based on the degrees of the neighboring nodes. Our setting offers great flexibility because the communication graph's topology can change between iterations. The matrix Wk, which encodes the current network, also changes. This is encoded in line 2, where Wk is generated using a rule Wk that can vary. Examples include the deterministic choice of a matrix sequence Wk or sampling from a dynamic probability distribution on matrices. Local steps without communication can be encoded with a diagonal matrix Wk.

```
Algorithm 1 Extra Step Time-Varying Gossip Method
parameters: stepsize  > 0, {Wk}k0 - rules or distributions for mixing matrix in iteration k.
initialize: z0  Z, m : z0 m = z0
1: for k = 0, 1, 2, . . . do
2: Sample matrix Wk from Wk
3: for each node m do
4: Generate independently mk+1/3  Dm
5:   zk+1/3 m = zk m  Fm(zk m, mk+1/3 )
6: Generate independently mk+2/3  Dm
7:    zk+1 =  Wk m,i zk+1/3
8:    zk+1/3
end for
9: end for
```

To ensure consensus between nodes, the mixing properties of the matrix sequence Wk must satisfy the following assumption:

Assumption 2.2 (Expected Consensus Rate). There exists a constant p (0, 1] and an integer 1 such that, after K iterations, for all matrices Z Rd×M and all integers l 0, . . . , K/ ,

$$\mathbb{E}_W \left[||ZW_{l\tau} - \bar{Z}||_F^2 \right] \leq (1-p)||Z - \bar{Z}||_F^2 \tag{3}$$

where Wl = W(l+1)1 ...Wl, we use the matrix notation Z = [z1, ..., zM] with z = (1/M)m=1M zm, and the expectation EW is over distributions of W and indices t l,...,(l+1) - 1.

This assumption guarantees that the consensus between nodes improves by a factor of 1-p after every gossip steps. Some matrices Wk can be the identity matrix (local steps only).

4 Setting and Assumptions

This section outlines the assumptions used to analyze the proposed algorithm:

Assumption 3.1 (Lipschitzness). For every m, the operator Fm(z) is Lipschitz with a constant L, meaning that:

$$||F_m(z_1) - F_m(z_2)|| \leq L||z_1 - z_2||, \forall z_1, z_2 \tag{4}$$

This is a common assumption used when analyzing all the methods in Table 1.

Assumption 3.2. We consider three scenarios for the operator F: (SM) Strong monotonicity, (M) Monotonicity, and (NM) Non-monotonicity under the Minty condition:

(SM) Strong monotonicity. For some > 0 and for all z1, z2, we have:

$$(F(z_1) - F(z_2), z_1 - z_2) \geq \mu||z_1 - z_2||^2 \tag{5}$$

(M) Monotonicity. For all z1, z2, we have:

$$(F(z_1) - F(z_2), z_1 - z_2) \geq 0 \tag{6}$$

(NM) Non-monotonicity (Minty). There exists z such that, for all z,

$$(F(z), z - z^*) \geq 0 \tag{7}$$

Assumptions (SM), (M), and (L) are widely used in the literature. Assumption (NM), often called Minty or Variational Stability, has recently been used as a non-monotonicity variant, particularly in GANs training.

Assumption 3.3 (Bounded noise). Fm(z,) is unbiased and has bounded variance. This means, for all z:

$$\mathbb{E}[F_m(z, \xi)] = F_m(z), \quad \mathbb{E}[||F_m(z, \xi) - F_m(z)||^2] \leq \sigma^2 \tag{8}$$

The final assumption pertains to the variability of local operators compared to their mean, which is called D-heterogeneity, and is commonly used when analyzing local-step algorithms.

Assumption 3.4 (D-heterogeneity). The values of the local operator have bounded variability:

$$||F_m(z) - \bar{F}(z)|| \leq D \tag{9}$$

5 Main Results

This section presents convergence rates for our proposed method under different settings defined by Assumption 3.2. We introduce the notation z = (1/M)m=1M zk for the average iterates and Z = (1/K)k=0K-1 z for the averaged sequence, i.e., ergodic average. We denote = $(^2/M + D^2), whichistheconsensuserror.$

Theorem 4.1 (Main theorem). Let Assumptions 2.2 and 3.1-3.4 hold, and the sequence z generated by Algorithm 1 runs for K > 0 iterations. Then:

- Strongly-monotone case: under Assumption 3.2 (SM) with = /L^2, $itholdsthat$:$\mathbb{E}[||\bar{z}_K - z^*||^2] \leq \left(1 - \frac{\mu}{2L}\right)^K ||z_0 - z^*||^2 + \frac{\gamma L^2 \Delta}{\mu}$(10)

Monotone case: under Assumption 3.2 (M), for any convex compact C with z0,z C and Q = maxz,z'C ||z - z'|| < Qc, with = O(min1/(K^0.5L), $(1/L)(p/), itholdsthat$:

$$\sup_{z \in C} \mathbb{E}[(F(\bar{z}_K), \bar{z}_K - z)] \leq \frac{L^2 Q_c^2}{K} + (L\sqrt{Q_c \Delta} + \Delta)\sqrt{\frac{Q}{\sqrt{K}}} \tag{11}$$

Under the assumption that for all k, ||z$_k$||$Qwith = O(min1/KL, p/), wehave$:

$$\sup_{z \in C} \mathbb{E}[(F(\bar{z}), \bar{z} - z)] \leq O(\frac{LQ^2}{K}) + O(\frac{L\Delta Q}{\sqrt{K}}) \tag{12}$$

Non-monotone case: under Assumption 3.2 (NM) and if ||z*||$Qwith = O(min1/KL, p/)$,$||z - z^*||^2 \leq \frac{LQ^2}{K} + \frac{L^2\Delta}{\mu} + \frac{LQ}{K^{1/4}}$(13)Under the additional assumption that, for all k, ||z$_k$||$Q, wehavethat\mathbb{E}[||\bar{z}_K - z^*||^2] \leq \frac{LQ^2}{K} + \frac{L^2\Delta Q}{K^{1/4}}$(14)

The proof of the theorem can be found in the supplementary materials, where the dependence of rates on the stepsize before optimal selection are given. In contrast to other analyses, our analysis addresses the fact that problem (1) has no feasible bounded set, which is important for analysis in both monotone and non-monotone settings. Furthermore, our algorithm includes a communication step that introduces a bias in the oracle, which needs to be analyzed over unbounded feasible sets. We overcome this by bounding the bias, and proving the boundedness in expectation of the sequence of iterates for both monotone and non-monotone cases. We also analyze stochastic extragradient method with biased oracles on unbounded domains which has not been done before. We achieve this under a general Assumption 2.2, with time varying graphs and all three monotonicity settings.

The convergence rates explicitly depend on the network, characterized by mixing time and mixing factor p, and on data heterogeneity D, which appear only as the quantity , the variance 2, Lipschitz constant L, strong monotonicity parameter , and the number of nodes M. These results help us determine how data heterogeneity, noise, and network characteristics influence convergence. This opens meta-optimization opportunities to design networks and set parameters such as M, , and p to improve convergence.

The convergence results presented in the theorem have a similar multi-term structure. The first term is from the deterministic case and mirrors existing methods for smooth VIs in a non-distributed setting. The second term is stochastic and is also standard for the non-distributed setting. The leading stochastic term is proportional to 2/M, decreasing with the number of nodes. Other terms represent a consensus error, due to imperfect communication between nodes. In all the cases this does not worsen the convergence, because dependence on K is no worse than the stochastic term.

Theorem 4.1 is given for a fixed iteration budget K, and corresponding stepsizes that depend on K, which is standard in literature. We also offer a procedure that allows extending the result to all-time convergence without a priori fixed K, by restarting the algorithm after K iterations, which are doubled each time.

In the strongly monotone case, our rate is slightly better than other results. The other methods' stepsize is limited as p/(L2), slowing convergence. For decentralized settings, our rate is worse, probably because Assumption 2.2 is more general, but our algorithm is more practical because it avoids multiple gossip steps per iteration and works with time-varying topologies. In the monotone case, we use the Gap function as a measure of suboptimality. And in the non-monotone setting we are able to obtain convergence up to a certain accuracy. It is important to note that we use assumptions about iterates that we can obtain only when they are generated by the algorithm. We manage to obtain corresponding results that can be used for establishing that the algorithm behaves nicely under certain initial conditions. The experimental section will demonstrate these theoretical findings.

6 Experiments

Here we present two experiments to validate the performance of Algorithm 1. Section 5.1 verifies the obtained convergence guarantees on two examples, a strongly-monotone and a monotone bilinear problem. Section 5.2 uses a non-monotone case with a GAN training application. Full details about the experimental setup are available in the supplementary material.

6.1 Verifying Theoretical Convergence Rate

This experiment aims to determine whether Algorithm 1's actual performance matches our theoretical rate from Theorem 4.1.

We consider a distributed bilinear saddle point problem (SPP) with the objective functions:

$$f_m(x, y) = a\|x\|^2 + b\langle y, C_m x\rangle,$$

where $x, y, C_m \in \mathbb{R}^n$, and a, b are real numbers.

This setup satisfies the assumptions with constants:

$$\mu = a, \quad L = a^2 + b^2, \quad D = \max_m \|C_m\|.$$

The network uses $M = 20$ nodes with uniform averaging weights. The dimension is $n = 5$, $b = 1$, $D \approx 3$, and $\tau = 1$. The p value is approximately 0.288.

To obtain stochastic gradients, unbiased Gaussian noise with variance σ^2 is added.

Convergence Behaviour. The convergence of Algorithm 1 with a fixed stepsize in both the strongly-monotone (a = 1) and monotone (a = 0) settings. In the strongly monotone setting we observe linear convergence up to an error floor determined by the noise and problem parameters. The monotone case converges more slowly, but is still linear up to a level. This is expected for bilinear problems. We see that when a constant stepsize is used in stochastic optimization algorithms, convergence is usually limited to a certain neighborhood, see Theorem 2 in a previous study. Theorem 4.1 also reflects this; convergence with zero error requires a diminishing stepsize. In the supplementary material, we also validate with decreasing stepsize.

We verify the dependence on the heterogeneity parameter D and set the noise $\sigma^2 = 0$. Based on the theory, we expect that the error when $\sigma = 0$ scales as $\mathcal{O}(D^2 K^{-2})$. We conduct experiments by setting $b = 1$ and $a = 1$, and measuring how many iterations are needed for

$$\left\| \frac{1}{M} \sum_m z_k - z^* \right\| < \epsilon,$$

while varying D. The step size is tuned for every experiment.

The number of iterations scale as $K \approx \epsilon^{-4}$, confirming that the error depends on K as $\mathcal{O}(K^{-1/2})$. The middle plot shows that iterations scale proportionally to D ($D \approx K$). Lastly, we see the number of iterations to reach $\epsilon = 0.01$ while varying the graph parameter p, and observe $D \approx p \cdot K$. This means that experiments confirm the $\mathcal{O}\left(\frac{1}{p} DK^2\right)$ term in the convergence rate.

6.2 Training GANs

Our method allows for combining communication graph topologies and local steps during distributed learning. This section explores our method on GANs training. In Section A.1, we discuss the relevance of our theoretical results to GANs training.

Data and model. We use the CIFAR-10 dataset which includes 60,000 images across 10 classes. We increase the dataset four times by adding transformations and noise, and simulate a distributed set up using 16 nodes on two GPUs with Ray. We create heterogeneity by splitting the dataset into 16 subsets where a major class makes up 20% of the data and the rest is split uniformly between all the other classes. We use the DCGAN architecture, conditioned by class labels, similar to a previous paper. We use Adam as the optimizer. We make one local Adam step and one gossip averaging step with time-varying matrices Wk, similarly to Algorithm 1.

Settings. We compare the following topologies, with respective matrices Wk:

- Full. A full graph is used at the end of each epoch; otherwise, local steps are taken. This leads to 120 communication rounds per epoch.
- Local. A full graph is used every five epochs; otherwise, local steps are taken. This means 24 communication rounds per epoch on average.

- Clusters. At the end of each epoch, clique clusters of size 4 are formed randomly (4 cliques in total). This results in 24 communication rounds per epoch.

The first topology has a 5x larger communication budget.

The learning rate is 0.002 for both generator and discriminator. The rest of the parameters are in the supplementary material.

7 Results

The methods reach a similar convergence in terms of local epochs and produced similar images. The Local and Cluster topologies perform much better in terms of communication, with the Cluster topology slightly outperforming the Local.

8 Conclusion

We have developed an effective algorithm to solve decentralized stochastic MVIs and SPPs, assuming a highly flexible network topology and communication constraints. This method represents the first decentralized extragradient approach that supports local steps for dynamic network topologies. We theoretically demonstrated the convergence rate of the algorithm for SM, M, and NM cases. In numerical experiments, we validated that the dependency on the data heterogeneity parameter D is tight in the SM case and impossible to improve in general. By training DCGAN in a decentralized manner, we showed our method's effectiveness for practical DL tasks. Future work could extend these algorithms to infinite-dimensional problems.