Comparison of novel gradient tracking methods for distributed optimization for real world application

Jayesh Yevale and Prerna Rathi

Abstract

In this project, we want to compare the convergence results of two novel gradient tracking method based algorithms for distributed multi agent optimization for big data applications. We considered distributed logistic regression for classification in this study. With regularization, the problem is a smooth, strongly convex. The two algorithms under study are the DSGT and SONATA for undirected graph variant. We use two high dimensional data sets for our study, MNIST and synthetic data. The optimization was run for different network settings and the results have been presented.

Motivation

Distributed optimization has been a trending topic of research in the past few decades. This is majorly due to the latest advancements in the technology of wireless sensors and also the emerging applications in machine learning, etc. Priorly the optimization problems were accounted for using centralized schemes. In these schemes, the data were processed on a single server, and clients were updated. Also, the complete data was accessible from a server. The main reasons that motivate the need for the distributed model are to include: (i) the unavailability of the collected data at the server, (ii) the privacy of the data among clients should be considered, and (iii) the limited power of computation and memory of both clients and servers.

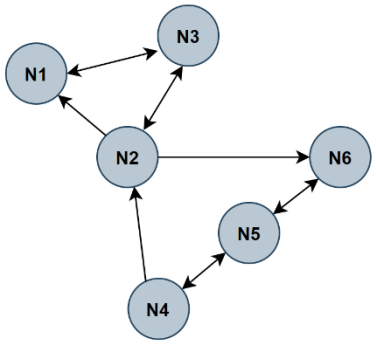

Fig 1: Distributed Network

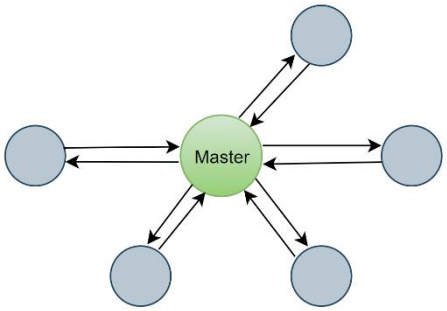

Fig 2: Centralized Network.

For addressing such challenges mentioned above, distributed optimization provides a proper framework. The clients (e.g., data processor, sensor) communicate over a network with their local information for minimizing the global objective function.
In some applications, the data may have a huge sample size or a large number of attributes. The problems associated with distributed models mostly have data with a large number of samples and attributes, often known as big data problems. In these problems, the computation of local gradient becomes expensive, and recently many algorithms are proposed by researchers in the past decade. In this report, our goal is to study the new algorithms in the distributed optimization models and compare their performance to a real-world problem

Introduction

We consider the regularized logistic optimization problem for binary classification applications. In a progressive manner, we show that this problem can be reformulated as a distributed stochastic optimization problem.

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x) \qquad \text{where } f(x) \triangleq \sum_{\ell\in\mathcal{S}} \ln\left(1+\exp\left(-v_\ell u_\ell^T x\right)\right) + \frac{\mu}{2}\|x\|^2$$

For a distributed implementation, let us assume that the dataset $\mathcal{S}$ is distributed among m agents. Let $\mathcal{S}_i$ denote the data locally known by agent i where $\mathcal{S} = \cup_{i=1}^{m\mathcal{S}_i}$. Note that the number of data points may differ among the agents. We let $|\mathcal{S}_i|$ denote the number of data points in the set $\mathcal{S}_i$. We rewrite the preceding loss minimization problem as a distributed regularized logistic regression loss minimization problem as

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad \sum_{i=1}^{m} f_i(x) \qquad \text{where } f_i(x) \triangleq \sum_{\ell\in\mathcal{S}_i} \ln\left(1+\exp\left(-v_\ell u_\ell^T x\right)\right) + \frac{\mu}{2m}\|x\|^2,$$

, where the agents communicate over undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the node set and $\mathcal{E} = \mathcal{N} \times \mathcal{N}$.

Literature Review

Distributed optimization minimizes the local function at each agent, by collecting information from its neighbours at each iteration of the update.

There are several types of methods algorithms developed and studied for network connections and sharing of local states in order to reach an optimum.

There are 3 main categories for distributed optimization approaches [3]-

(i) Consensus-based methods-classical gradient or subgradient steps, gradient tracking methods.

(ii) Dual methods- Lagrangian dual equivalent formulations of the problem to obtain a distributed routine.

(iii) Constraint exchange methods-exchange of active constraints.

The focus of this study is going to be on the gradient tracking method which is an improvement over the subgradient method. It tracks the gradient of the overall cost function using a dynamic consensus scheme[3]. These methods provide a faster convergence rate over the subgradient method as it allows the usage of constant step size.

In this study, we compare the two distributed optimization algorithms which are the distributed stochastic gradient tracking (DSGT) method developed by Nedich [1] and the SONATA method developed by Scutari[2].

Assumptions

1. All agents f_i is μ strongly convex and L smooth.
2. For all agents i, the random variable ξ_i is independent of each other, and the
   $E[\nabla f_i(x, \xi_i)|x] = \nabla f_i(x)$
   $E[\| \nabla f_i(x, \xi_i) - \nabla f_i(x) \|^2 |x] \leq \nu^2$, for some $\nu > 0$
3. The weight matrix $\boldsymbol{W}$ is doubly stochastic and all the diagonal elements $w_{ii} > 0$. The graph that we use for connecting the agents to a network can be line, star, complete etc. The non-negative coupling matrix $\boldsymbol{W}$ for the network graph is doubly stochastic. i.e $\boldsymbol{W}1 = 1$ and $1^T \mathbf{W} = 1^T$. In addition, $w_{ii} > 0$ for some $i \in \mathcal{N}$.[1].
4. The graph $\mathcal{G}$ corresponds to communication network between the agents, and the graph is connected and undirected.
5. Agents are cooperative and each agent only has limited information of the whole problem and optimizes the cost locally.

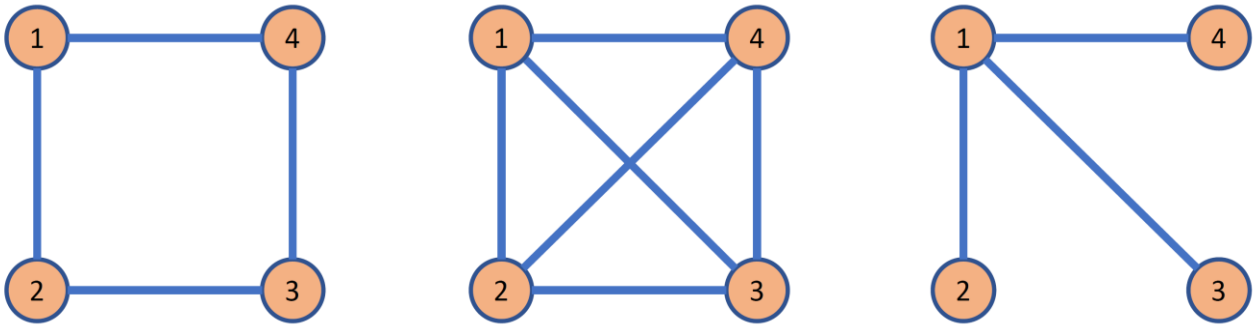

Fig 3: Doubly stochastic undirected network (Ring, Complete and Star Graphs)

Gradient Tracking Algorithms

Notations

Let x^* denotes the unique global optimal solution of the problem.

The $x_{i,k} \in R^n$denotes the local copy of the {decision variable}.

The $y_{i,k} \in R^n$ denotes the local copy which is used to track the average of the gradient mapping of the global objective function.

$$\mathbf{x} := [x_1, x_2, \ldots, x_m]^T \in \mathbb{R}^{m\times n}, \qquad \mathbf{y} := [y_1, y_2, \ldots, y_m]^T \in \mathbb{R}^{m\times n},$$

$$\bar{x} := \frac{1}{m}\mathbf{1}^T\mathbf{x} \in \mathbb{R}^{1\times n}, \qquad \bar{y} := \frac{1}{m}\mathbf{1}^T\mathbf{y} \in \mathbb{R}^{1\times n},$$

where **1** indicates the vector for all entries as 1. We define the total objective function.

$$f(x) \triangleq \sum_{i=1}^{m} f_i(x), \qquad \mathbf{f}(\mathbf{x}) \triangleq \sum_{i=1}^{m} f_i(x_i)$$

$$f_i(x) \triangleq \mathbb{E}[f_i(x, \xi_i) \mid x]\,.$$

$$\boldsymbol{\xi} := [\xi_1, \xi_2, \ldots, \xi_m]^T \in \mathbb{R}^{m\times d},$$

$$\mathbf{G}(\mathbf{x}, \boldsymbol{\xi}) \triangleq [\nabla f_1(x_1, \xi_1), \ldots, \nabla f_m(x_m, \xi_m)]^T,$$

$$G(\mathbf{x}, \boldsymbol{\xi}) = \frac{1}{m}\sum_{i=1}^{m} f_i(x_i, \xi_i) \in \mathbb{R}^{m\times n}.$$

The following are the recursive bounds considered while studying the paper:

$E[|\overline{x_k} - x^*|^2]$ is a suboptimality metric,

$E[|x_k - 1\overline{x_k}|^2]$ is the consensus violation metric for the vector $x_{i,k}$, and

$E[|y_k - 1\overline{y_k}|^2]$ is the consensus violation metric for the vector $y_{i,k}$.

Distributed Stochastic Gradient Tracking Method

A distributed gradient tracking method was proposed by Shi Pu and Angelica Nedich. In this algorithm, the agent-based auxiliary variables known as the gradient tracker y_i is introduced. This gradient tracker tracks the average gradients of f_i from the neighbouring agents by getting their local information maintaining the privacy and this information is assumed the information accurate. It is shown that using the diminishing step size converges linearly to the optimal.

Algorithm 1 Distributed Stochastic Gradient Tracking Method (DSGT)

Input: Agents choose a doubly stochastic weight matrix $\mathbf{W}$ and set an initial step-size α.
For all $i \in [m]$, agent i chooses a random initial point $x_i^0 \in \mathbb{R}^n$

For all $i \in [m]$, agent i generates a realization of the random variable ξ_i, denoted as ξ_i^0, and evaluates the initial gradient tracker $y_i^0 := \nabla f_i\left(x_i^0, \xi_i^0\right)$

for $k = 0, 1, \ldots,$ **do**

  For all $i \in [m]$, agent i generates a realization of the random variable ξ_i, denoted as ξ_i^{k+1}, and evaluates the local gradient mapping $\nabla f_i\left(x_i^{k+1}, \xi_i^{k+1}\right)$

  For all $i \in [m]$, agent i does the following updates:

    $x_i^{k+1} := \sum_{j=1}^{m} W_{ij}\left(x_j^k - \alpha y_j^k\right)$

    $y_i^{k+1} := \sum_{j=1}^{m} W_{ij} y_j^k + \nabla f_i\left(x_i^{k+1}, \xi_i^{k+1}\right) - \nabla f_i\left(x_i^k, \xi_i^k\right)$

end for

Compact form DSGT:

$$\mathbf{x}_{k+1} = \mathbf{W}(\mathbf{x}_k - \gamma_k \mathbf{y}_k),$$
$$\mathbf{y}_{k+1} = \mathbf{W}\mathbf{y}_k + \mathbf{G}(\mathbf{x}_{k+1}, \boldsymbol{\xi}_{k+1}) - \mathbf{G}(\mathbf{x}_k, \boldsymbol{\xi}_k).$$

SONATA Gradient Tracking Method

The SONATA algorithm (Successive cONvex ApproximaTion Algorithm) is proposed by Ying Sun, Amir Daneshmand, and Gesualdo Scutari. It uses a surrogate function of the agents subproblems. The perturbed (push-sum) consensus mechanism tracks the gradient of *F* locally. These surrogate functions replaces the classical first order approximation methods. Traditionally, the local objective functions were computed and managed by the central server, which was suited for the distributed type of models

earlier. The surrogate methods can also be applied to (approximate) Newton-type subproblem or mirror descent-type methods.

In this report, we use the variant of the SONATA algorithm in undirected graphs. We also reduce the local gradient update to ATC form to $x_i^{k+1/2} = x_i^k + \alpha y_i^k$, for incorporating the unconstrained nature of the problem. This allows us to use the algorithm for a wider application.

Algorithm 2 SONATA over undirected graphs

Input: Agents choose a doubly stochastic weight matrix $\mathbf{W}$ and set an initial step-size α. For all $i \in [m]$, agent i chooses a random initial point $x_i^0 \in \mathbb{R}^n$, and evaluates initial gradient tracker $y_i^0 := \nabla f_i\left(x_i^0\right)$.

for $k = 0, 1, \ldots,$ **do**

For all $i \in [m]$, agent i performs a local gradient update.

$$\hat{x}_i^k = \underset{x_i \in \mathcal{K}}{\operatorname{argmin}}\ \tilde{f}_i(x_i, \hat{x}_i^k) + \left(y_i^k - \nabla f_i\left(x_i^k\right)\right)^T \left(x_i - x_i^k\right)$$

$$x_i^{k+1/2} = x_i^k + \alpha d_i^k, \quad \text{with } d_i^k \triangleq \hat{x}_i^k - x_i^k$$

For all $i \in [m]$, agent i does the following for consensus updates:

$$x_i^{k+1} := \sum_{j=1}^{m} W_{ij}\left(x_i^{k+1/2}\right)$$

$$y_i^{k+1} := \sum_{j=1}^{m} W_{ij}\left(y_j^k + \nabla f_i\left(x_i^{k+1}\right) - \nabla f_i\left(x_i^k\right)\right)$$

end for

Numerical Experiments

For the experiments, we consider the regularized logistic regression loss minimization problem presented above. We consider a dataset denoted by $\mathcal{D} \triangleq \{(u_j, v_j) \in R^n \times \{-1, +1\} \mid j \in \mathcal{S}\}$, where $\mathcal{S} \triangleq \{1, \ldots, s\}$ denotes the index set. Let $\mathcal{S}_i$ denote the index set of the data locally known by agent i, where $\cup_{i=1}^m \mathcal{S}_i = \mathcal{S}$. The problem can be formulated as $\min \sum_{i=1}^m f_i(x)$ where we define local functions f_i as

$$f_i(x) \triangleq \frac{1}{|\mathcal{S}|} \sum_{j \in \mathcal{S}_i} \ln\left(1 + \exp\left(-v_j u_j^T x\right)\right) + \frac{\mu}{2m}|x|^2,$$

where $u_j \in R^n$ and $v_j \in \{-1,1\}$ for $j \in \mathcal{S}_i$ which denotes the binary value of the j^{th} data label.

Simulations

For this experiment, we implemented and compared the DSGT and SONATA variant over undirected graphs. We performed the experiments on two data sets with m agents. MNIST and Synthetic data set for $m = 10$, each is used. The MNIST data set consists of 10,000 labels and 784 attributes. The Synthetic data set is a Gaussian distribution with a mean of 5 and a standard deviation of 0.5 with 10,000 labels and 10,000 attributes.

We have used complete and the ring graphs for the communication among the agents. Similar parameters have been considered for different data sets, for appropriate comparison. After running the algorithms for various step sizes (α), we use $\gamma = 1e + 1, \Gamma = 1e + 4, \mu = 1e - 2$. The batch size for computing gradient from each agent $\epsilon = 1e + 1$ for both data sets. As the DSGT algorithm is stochastic, we considered having 10 sample paths in our implementations.

Both the algorithms were run for 100 function evaluations. As DSGT involves stochastic gradient updates the number of function evaluations in each iteration is less for DSGT than SONATA. Therefore it would be unfair to compare convergence results against no of iterations. So, we are using number of function evaluations as a comparison metric.

Challenges and Solutions:

Step-size: Choosing the initial step size to for both algorithms for comparable results was an arduous task. We decided to use diminishing step-size (α) for descent guarantee. α is given by γ /(γ+k), where value of γ was chosen after trial and error for both datasets and k is the number of iterations.

Variant of SONATA: The SONATA algorithm was introduced for time-varying digraphs, and was defined for a constrained set of x. The variant for this surrogate optimization SONATA algorithm was used for the required static and undirected case. For this, we updated the local gradient step update referred to in the NEXT (In-network nonconvex optimization) algorithm in convex setting which was introduced by P. Lorenzo (2016) [5].

Sample paths: In order to capture the stochasticity of the DSGT algorithm, we have considered sample paths for tracing the functions and consensus. We use the averages of the sample paths initially in our plots.

Confidence Intervals: The averages of the sample paths were flawed when used for higher dimensions and a large number of total gradient updates. In order to overcome this issue, we used the seaborn library in Python to plot the CIs for the plots. This resulted in proper graphs for the mentioned algorithms.

Analysis and Results

Both algorithms exhibit descent property for the logistic regression problem.
It is observed that consensus convergence is achieved for both the synthetic and real-world dataset MNIST under various settings (ring and complete graphs).
Linear convergence is observed for both the datasets and settings. This is shown in the figures below
To ensure the repeatability of results, we ran the algorithms for several randomly generated initial points. We found similar function values and consensus convergence for each run.

Algorithms\ Datasets	MNIST dataset	Synthetic dataset
DSGT (Complete)	[2.736e+2,2.742e+2]	[1.214e+3,1.215e+3]
DSGT (Ring)	[2.734e+2, 2.738e+2]	[1.208e+3,1.209e+3]
SONATA (Complete)	2.425e+2	1.213e+3
SONATA (Ring)	2.424e+2	1.212e+3

Table 1: Objective function comparison of DSGT vs. SONATA for 90% CI's for $m = 10$.

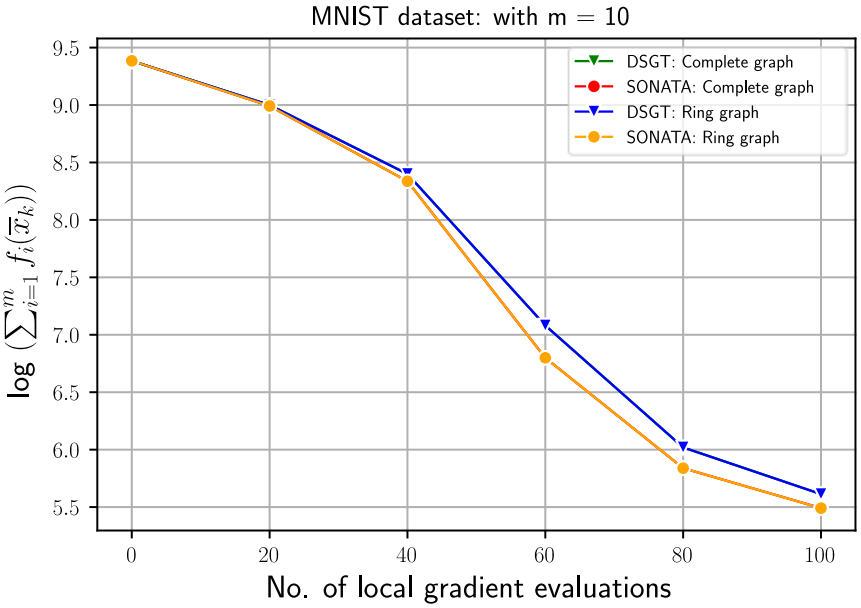

Fig 4: Comparison of objective values of DSGT vs. SONATA for MNIST dataset

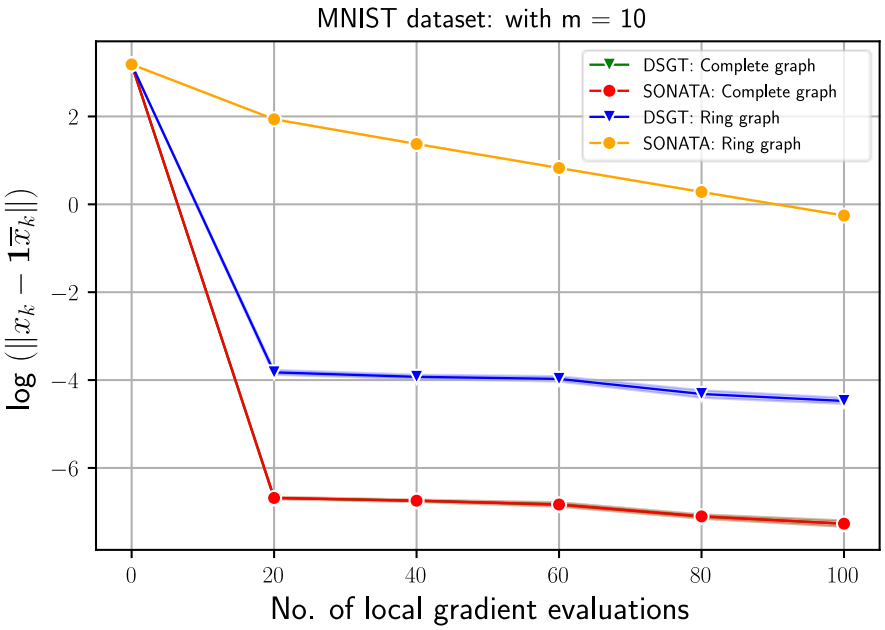

Fig 5: Comparison of consensus values of DSGT vs. SONATA for MNIST dataset

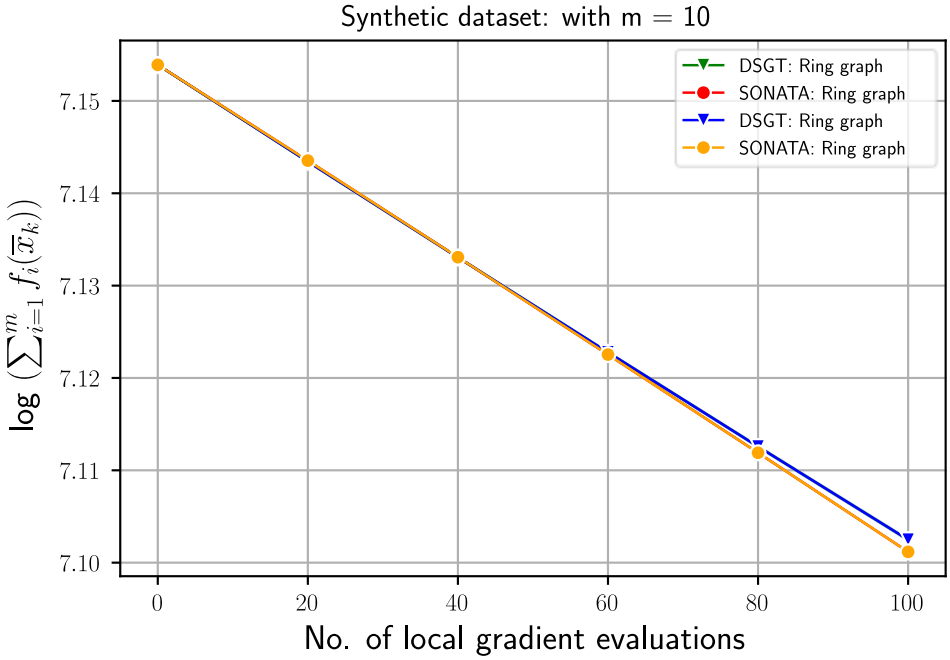

Fig 6: Comparison of objective values of DSGT vs. SONATA for Synthetic dataset

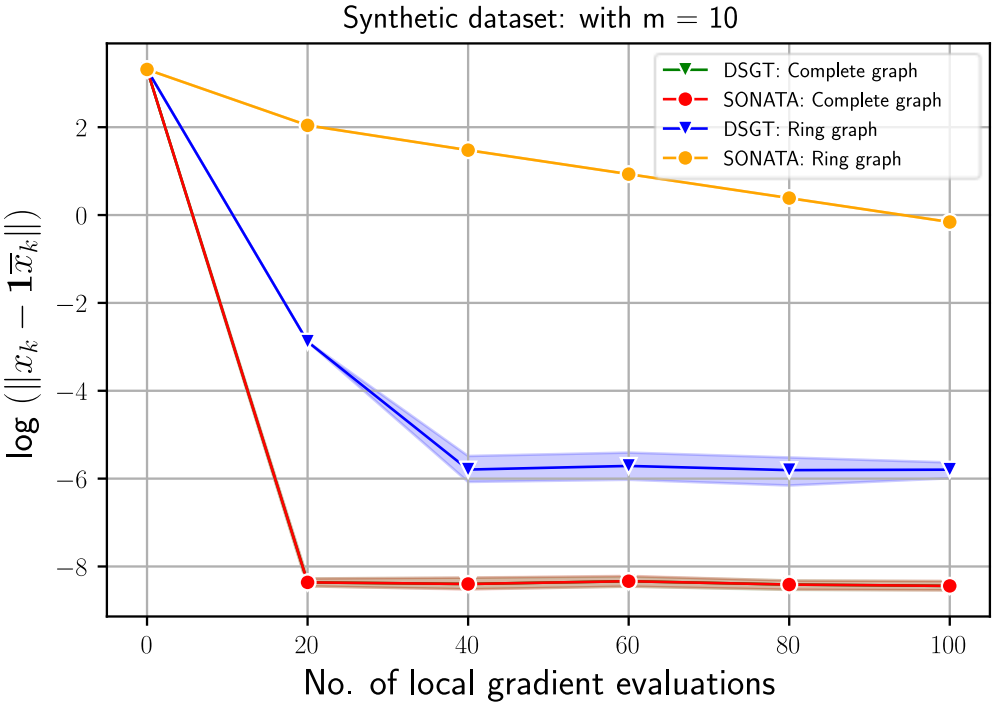

Fig 7: Comparison of consensus values of DSGT vs. SONATA for Synthetic dataset

We observe that for complete graph structure for both DSGT and SONATA algorithms, there is no significant difference in the function values as the information sharing is maximum. In the case of ring graph, the information sharing impacts the performance of the consensus metric of y_i. The objective function converges and the performance is significantly similar. Due to the accumulation of stochasticity errors in the DSGT algorithm, the SONATA algorithm supersedes in obtaining a better minima for the objective for both the settings.

In the Figures, for the proper comparison between the chosen algorithms, we compared these algorithms over the total number of local gradient evaluations. The DSGT algorithm updates the agents over a stochastic gradient which in turn reduces the gradient computational efforts drastically. Whereas in the SONATA algorithm variant over the undirected graph, we use the complete gradient of the samples at each agent and share the local copy of the decision variables and surrogate variables. These local gradient evaluations are the number of total samples used in each gradient step. The highlighted areas in the plots in Figures, which represents 90% confidence intervals. We choose the total of 10 sample paths for both MNIST and Synthetic data sets.

Conclusion and Future Work

The report analysis and compares convergence results for the optimization logistic regression problem in a distributed setting for two novel gradient tracking algorithms. The DSGT algorithm which computes stochastic gradients and the SONATA algorithm is chosen for the study.

Both the algorithms exhibit gradient descent property at each agent and also achieves sub-optimality and consensus convergence under different network settings.

We observed that SONATA results are better for objective values compared to SONATA for both network settings, given that it evaluates the entire gradient for the function.

In the future, we can implement the comparison of algorithms on different problems and datasets, also we can theoretically advance the convergence rate analysis for constant stepsize. For example, Least Square Regression and Support Vector Machines can also be studied.

References

[1] Pu, Shi, and Angelia Nedić. "Distributed stochastic gradient tracking methods." *Mathematical Programming* 187, no. 1 (2021): 409-457.

[2] Sun, Ying, Gesualdo Scutari, and Amir Daneshmand. "Distributed Optimization Based on Gradient Tracking Revisited: Enhancing Convergence Rate via Surrogation." *SIAM Journal on Optimization* 32, no. 2 (2022): 354-385.

[3] Notarstefano, Giuseppe, Ivano Notarnicola, and Andrea Camisa. "Distributed optimization for smart cyber-physical networks." *arXiv preprint arXiv:1906.10760* (2019).

[4] Yang, Tao, Xinlei Yi, Junfeng Wu, Ye Yuan, Di Wu, Ziyang Meng, Yiguang Hong, Hong Wang, Zongli Lin, and Karl H. Johansson. "A survey of distributed optimization." *Annual Reviews in Control* 47 (2019): 278-305.

[5] Di Lorenzo, Paolo, and Gesualdo Scutari. "Next: In-network nonconvex optimization." *IEEE Transactions on Signal and Information Processing over Networks* 2, no. 2 (2016): 120-136.

[6] Yousefian, Farzad, Jayesh Yevale, and Harshal D. Kaushik. "Distributed Randomized Block Stochastic Gradient Tracking Method." *arXiv preprint arXiv:2110.06575* (2021).