Annals of Optimization With Applications

www.anowa.reapress.com

Ann. Optim. Appl. Vol. 1, No. 2 (2025) 110–118.

Paper Type: Original Article

A Dual-Based Distributed Optimization Method on Time-Varying Networks

Elham Monifi[1], Nezam Mahdavi Amiri[1,*]

[1] Department of Mathematics, Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran; elham.monifi@sharif.edu; nezamm@sharif.edu.

Citation:

Received: 14 November 2024 *Revised: 17 January 2025* *Accepted: 19 March 2025*	Monifi, E., & Mahdavi Amiri, N. (2025). A dual-based distributed optimization method on time-varying networks. *Annals of optimization with applications, 1* (2), 110-118.

Abstract

We propose a time-varying dual accelerated gradient method for minimizing the average of n strongly convex and smooth functions over a time-varying network with n nodes. We prove that the Time-Varying Dual Accelerated Gradient Ascent (TV-DAGA) method converges at a linear rate with the time to reach an ε-neighborhood of the solution being of $\mathcal{O}(\ln\frac{1}{\epsilon})$. We test the proposed method on two classes of problems: L_2-regularized least squares and logistic classification problems. For each class, we generate 1000 problems and use the Dolan-Moré performance profiles to compare our obtained results with the ones obtained by several state-of-the-art algorithms to illustrate the efficiency of our method.

Keywords: Distributed learning, Distributed optimization, Time-varying networks.

1 | Introduction

Distributed optimization has become a popular approach for dealing with today's networks having huge datasets in various areas, such as machine learning and sensor networks, to name a few. Datasets in these networks are often produced in a decentralized fashion, and transporting these datasets over a network is usually undesirable either due to security concerns or traffic constraints.

Classical centralized optimization methods often work well for small networks with a central node directly communicating with all the other nodes for the reception of data to solve the optimization problem. But, this often fails as the network becomes large and security or traffic concerns impel nodes to share data only with neighbors. Decentralized methods should be appropriated to deal with such problems; that is, nodes should cooperate in finding an optimal solution for the network. In solving a centralized constrained optimization

Corresponding Author: nezamm@sharif.edu

https://doi.org/10.48314/anowa.v1i2.43

problem, efficiency is measured by the amount of required computations, while in a distributed constrained optimization problem, the amount of data communication is a more decisive element.

Several authors have considered distributed optimization on time-varying undirected graphs. Nedic et al. [1] combined an inexact gradient method with the gradient tracking technique on a time-varying undirected graph having a doubly stochastic communication matrix. They established a polynomial time complexity of their method. Jakovetić et al. [2] considered a time-varying network with each node at iteration k being active with a probability p_k. They established a probabilistic sub-linear convergence rate in the sense of expected distance to the solution. Maros and Jaldén [3] proposed a dual method for an always-connected time-varying undirected graph. They assumed double stochastic communication matrices with a constraint on the spectrum of these matrices. They proved an R-linear convergence rate. They further proposed a more computationally economical algorithm, trading off the convergence rate of the original algorithm [4]. Rogozin et al. [5] proposed a dual gradient method based on Nesterov's idea for time-varying networks with a finite number of changes over time. Wu et al. [6] also proposed a dual gradient method on time-varying networks. Ding et al. [7] considered the privacy of data exchange to be low when the exact information is transmitted. So, to preserve privacy, the addition of noise to transmitted data is inevitable. They discussed the trade-off between accuracy and privacy. They also proved linear convergence rate in a mean sense.

We extend the ideas of the dual gradient ascent and Nesterov's accelerated gradient ascent methods to time-varying graphs. Using a time-varying model, we handle network topology changes, including link failure and latency. We establish that our proposed Time-Varying Dual Accelerated Gradient Ascent (TV-DAGA) method is linearly convergent on time-varying and always connected networks. Our work is an extension of the work of [8] and is close to those of [5] and [9]. However, the authors of [8] and [9] considered time-invariant networks, and the authors of [5] considered a time-varying network with a finite number of changes. Here, we consider time-varying graphs with infinitely many changes over time. Then, in a comparative investigation, we use two performance profiles and compare the performance of our methods with the ones due to some state-of-the-art methods, namely, DIGing [1], FDGM [6], Eco-PANDA [4] and PANDA [3], on two classes of optimization problems: L_2-regularized least squares and logistic classification problems. First, we randomly generate 1000 problem instances with $40 < n < 80$ nodes. Then, corresponding to each class, we first compare the performance of the methods by the Dolan-Moré performance profiles using the performance measure as the error after 100 iterations. Then, we compare the methods again using the Dolan-Moré performance profiles with the performance measure being the number of iterations to reach either a relative accuracy of $\epsilon = 10^{-10}$ or an absolute accuracy of $\delta = 10^{-35}$. We will call this (ε,δ)-accuracy. We show that TV-DAGA outperforms all the other algorithms in terms of accuracy after 100 iterations, as well as in terms of number of iterations to reach an (ε,δ)-accuracy.

2 | Time-Varying Dual Accelerated Gradient Ascent Algorithm

Here, we consider the minimization of the sum of n functions over a time-varying network with n nodes. We model the time-varying network with the graph sequence $\{G_k\}_{k=0}^{\infty}$, where $G_k = (V, E_k)$ is a simple connected undirected graph, with $V = \{1,2,\cdots,n\}$ being the set of nodes and E_k being the set of edges at time k. Each node i has access to a private function $f_i(\theta)$ which is assumed to be α-strongly convex, with its gradient being β-Lipschitz. Our goal is to find the minimizer of the average of these n local functions, that is, to solve the problem.

$$\min_{\theta\in\mathbb{R}^d} f(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta). \tag{1}$$

We assume that nodes upgrade their values by communicating with their neighbors. Moreover, the f_i are private objectives, and nodes cannot share their objective values with neighbors. However, they can share their primal and dual variables and gradient values with neighbors. We make the following assumptions for *Problem (1).*

Assumption 1. Each function f_i is α −strongly convex and β −smooth, that is, there exist $\beta \geq \alpha > 0$ such that for all $x, y \in \mathbb{R}^d$, we have that $\alpha \parallel x - y \parallel_2^2 \leq 2(f_i(x) - f_i(y) - \langle \triangledown f_i(y), x - y\rangle) \leq \beta \parallel x - y \parallel_2^2$.

Note that in *Assumption 1*, for α-strong and β-smooth objective functions, the existence of a unique solution θ^* is guaranteed. The following assumption briefly explains the regulations under which the nodes compute and communicate.

Assumption 2. Each node i has a computing unit which can compute f_i, the Fenchel conjugate f_i^*, the gradients $\triangledown f_i$ and $\triangledown f_i^*$. Moreover, each node has a communicating unit that can distinguish its neighbors at time k and exchange primal, dual, and gradient values with the neighbors.

We also choose a Laplacian-based model for the communication matrix because the eigenvalues of the graph Laplacian matrix provide information about graph connectedness and the number of graph components in a disconnected graph. Moreover, the smallest nonzero eigenvalue of the Laplacian matrix, called the eigengap, plays a vital role in estimating a graph-dependent step size. Our communication model is introduced having the following assumption.

Assumption 3. The communication model is represented by the Laplacian matrix sequence $\{W_k\}_{k=0}^{\infty}$ as follows:

$$[W_k]_{ij} = \{\begin{matrix} -1, & (i,j) \in E_k, \\ \deg(i), & i = j, \\ 0, & \text{otherwise}, \end{matrix} \tag{2}$$

where $\deg(i)$ denotes the degree of node i, that is the number of connected nodes to i at a time k.

We also define another quantity based on the graph spectrum, which is highly used in our analysis.

Definition 1 ([8]). For any integer $k \geq 0$, the quantity $\tau_k = \lambda_2(W_k)/\lambda_n(W_k)$ is called the normalized eigengap of the connected graph G_k with the Laplacian matrix W_k . We also define $\tau = \inf_{k\geq 0}\tau_k$.

A very standard distributed reformulation of *Problem (1)* is to assign a variable θ_i to each node i and rewrite *Eq. (1)* as follows:

$$\min_{\theta_1=\theta_2=\cdots=\theta_n} \frac{1}{n}\sum_{i=1}^{n} f_i(\theta_i). \tag{3}$$

Since *Problem (1)* has a unique solution θ^* and the f_i are all strongly convex, the reformulated *Problem (3)* has also a unique solution $\theta_1^* = \cdots = \theta_n^* = \theta^*$. Now, let us define $\Theta = [\theta_1, \cdots, \theta_n] \in \mathbb{R}^{d\times n}$ and $F(\Theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta_i)$. To include the network communication model in the kth iteration, we replace the above constraint with $\Theta W_k = 0$. Note that W_k is the Laplacian of a connected graph, and using Kirchoff's law and graph connectedness assumption, we conclude that $\theta_1 = \cdots = \theta_n$ if and only if $\Theta W_k = 0$. Now, we move one step further and replace the constraint $\Theta W_k = 0$ by its equivalent $\Theta\sqrt{W_k} = 0$. This equivalence can be easily verified through the spectral decomposition of W_k. We use the above equivalence and build a sequence of subproblems:

$$\min_{\Theta\sqrt{W_k}=0} F(\Theta), k = 0,1. \tag{4}$$

Each subproblem starts with an initial value Θ_k, and using a strictly decreasing method, we compute Θ_{k+1} such that $F(\Theta_{k+1}) < F(\Theta_k)$. The value Θ_{k+1} is used as the initial value for the next subproblem. This way, we construct a strictly decreasing sequence $\{F(\Theta_k)\}_{k=0}^{\infty}$, which is convergent to a unique value due to the lower-boundedness of F. Also, Θ_k converges to the unique solution of *Eq. (3)* due to the strong convexity of F and the equivalence of *Eq. (3)* with all the given *Subproblems (4)*.

The dual of *Problem (5)* is

$$\max_{\Lambda\in\mathbb{R}^{d\times n}} - F^*(\Lambda\sqrt{W_k}), k = 0,1,\cdots. \tag{5}$$

The dual gradient method for solving this problem computes the following iterates:

$$\Lambda_{k+1} = \Lambda_k - \eta_k \nabla F^*(\Lambda_k\sqrt{W_k})\sqrt{W_k}. \tag{6}$$

Let us consider the change of variables $X_k = \Lambda_k\sqrt{W_k}$ to obtain $\Lambda_{k+1}\sqrt{W_k} = X_k - \eta_k \nabla F^*(X_k)W_k$. This relation inspires us to set the left-hand-side to be the next value and write $X_{k+1} = X_k - \eta_k \nabla F^*(X_k)W_k$. We will prove that the iterates $\{X_k\}_{k=0}^{\infty}$ produce a dual objective sequence $\{F^*(X_k)\}_{k=0}^{\infty}$, which is strictly increasing and, since it is upper bounded by $F(\Theta^*)$, converges to the unique solution of *Eq. (1)*.

The acceleration idea has been introduced by Nesterov [10]. Scaman et al. [8] used this idea and introduced algorithms for time-invariant graphs. Here, we use the acceleration idea and give an accelerated method to deal with connected time-varying graphs. We provide the TV-DAGA method as *Algorithm 1*.

Algorithm 1 is an extension of the dual accelerated gradient method for a time-invariant network considered in [8]. We will establish that *Algorithm 1* is convergent if the graph sequence $\{G_k\}_{k=0}^{\infty}$ is always connected.

Theorem 1. If the graph sequence $\{G_k\}_{k=0}^{\infty}$ is always connected, and *Assumptions 1-3* hold, then *Algorithm 1* converges R-linearly to the unique solution of *Problem (3),* that is, $\Theta^* = [\theta^*,\cdots,\theta^*]$, with θ^* being the unique solution of *Eq. (1)*. Moreover, the time to reach an ϵ −neighborhood of the solution is $\mathcal{O}(\frac{1}{\ln\frac{1}{c}}\ln\frac{M}{\epsilon})$, where $c = 1 - \tau\sqrt{\frac{\alpha}{\beta}}$ and M is a constant depending on the initial values.

We provided a detailed discussion and the proof of *Theorem 1* in [11].

Algorithm 1. TV-DAGA algorithm.

```
Data:   $Y_0 = 0, V_0 = 0, f_1, \cdots, f_n$, Maxiter, $\delta, \epsilon$;
Result:   $\Theta^* \in \mathbb{R}^{d\times n}$;
Set  $k = 0$;
Repeat:
  Read $W_k$ ;
  Compute  $\tau_k = \lambda_2(W_k)/\lambda_n(W_k)$;
  Compute  $\alpha_k = \tau_k\sqrt{\alpha/\beta}$,   $\mu_k = 1/(1 + \alpha_k)$;
  Set  $\Theta_k = \nabla F^*(X_k)$;
  Set $X_k = V_k + \mu_k(Y_k - V_k)$;
  Set $V_{k+1} = (1 - \alpha_k)V_k + \alpha_k X_k - \frac{\beta\alpha_k}{\lambda_n(W_k)}\Theta_k W_k$;
  Set $Y_{k+1} = X_k - \frac{\alpha\tau_k}{\lambda_n(W_k)}\Theta_k W_k$;
Until:
  $|f_i(\theta_{i,k}) - f_i(\theta_{i,k-1})| \leq \delta + \epsilon|f_i(\theta_{i,k})|$, for all i, or,
          $k = $ Maxiter, or,
          $|\sum_{i=1}^{n} \nabla f_i(\theta_{i,k})| \leq \delta$.
```

3 | Numerical Experiments

Here, the performances of *Algorithm 1*, DIGing [1], FDGM [6], PANDA [3], and Eco-PANDA [4] are compared on distributed L_2 −regularized least squares and logistic classification problems. The implementations in this section are done on a MacOS Ventura 13.5.1 with 8GB RAM in a Python 3.8 environment. However, practically, the implementations are to be made on a network.

3.1 | L2-Regularized Regression Problems

In L_2-regularized regression, we aim at solving the optimization problem

$$\min_{\theta\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n}(a_i^T\theta - b_i)^2 + c \parallel \theta \parallel^2, \tag{7}$$

where $c > 0$ is a regularization parameter. We set c = 0.2 and d = 20. We also pick all the a_i and b_i entries from a uniform distribution in the interval $[0.1,1]$. The entries of the graph sequence $\{G_k\}_{k=0}^{\infty}$ are chosen uniformly from the sample space of n-node-graphs with $m = 5 \times n$ edges. In our algorithm, we work with the Laplacian matrix of the graph sequence entries. However, the other algorithms need a sequence of symmetric doubly stochastic communication matrices. So, we use the sequence of Metropolis weight matrices for them (see [1]).

We first implement our method and other state-of-the-art methods on a network with n = 100 nodes and m = 500 time-varying edges. We plot the error value as a function of k. The result is shown in *Fig. 1a*, Comparing TV-DAGA and PANDA, we see that they decrease with the same slope in early iterations, but eventually, TV-DAGA converges to a more accurate solution. On the other hand, comparing TV-DAGA with FDGM, we see that they converge with the same accuracy to the optimal solution, but FDGM decreases slower than TV-DAGA in early iterations. We also observe that TV-DAGA outperforms DIGing and Eco-PANDA in terms of accuracy and speed of convergence.

Next, we randomly generate problem set P with 1000 problems with 50 ≤ n ≤ 80 nodes and $m = 5 \times n$ randomly changing edges. Then, we plot the Dolan-Moré performance profiles for accuracy after 100 iterations in *Fig. 1b*; we see that TV-DAGA solves all the problems with the least error after 100 iterations, while PANDA solves all problems with errors at most 40 times the least error value. Next, we use problem set P and produce the Dolan-Moré performance profiles, with the performance measure being the number of iterations to reach within $(10^{-10}\ ,10^{-35})$-accuracy. The result is shown in *Fig. 1c*. We observe that TV-DAGA reaches this accuracy with a minimum number of iterations for all the problems, while FDGM reaches this accuracy for all problems with a number of iterations between 1.2 and 1.4 times the number of iterations due to TV-DAGA.

3.2 | Logistic Classification

In logistic classification, we aim to solve the optimization problem:

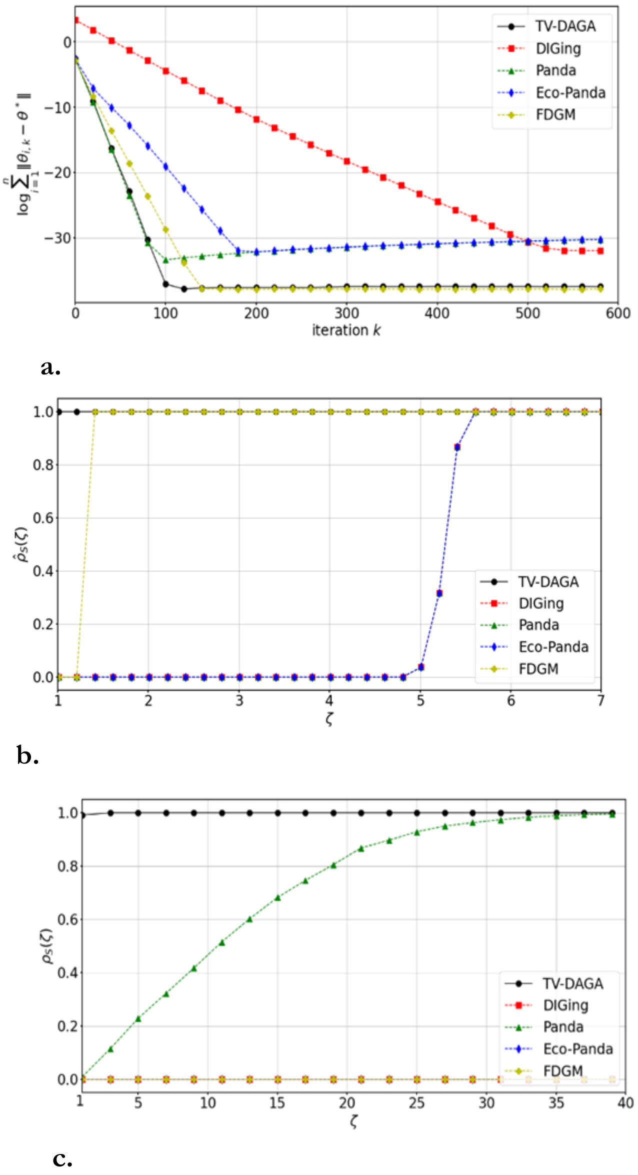
$$\min_{\theta\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n}\frac{1}{J}\sum_{j=1}^{J}\ln(1 + e^{-b_{i,j}a_{i,j}^T\theta}) + c \parallel \theta \parallel^2, \tag{8}$$

where $a_{i,j} \in \mathbb{R}^d$ is a data point with an assigned value $b_{i,j} \in \{-1,1\}$. Our goal is to use $n \times J$ data points and their labels to learn the coefficients $\theta \in \mathbb{R}^d$ of a linear classifier through solving the optimization *Problem (8)*. We assume that the network has n nodes, and each node has J data points. In our implementation, we set c = 0.2, d = 20, and J =10. We also pick all the $a_{i,j}$ entries from a uniform distribution in the interval [0.1, 1] and all the $b_{i,j}$ randomly from the set $\{-1,1\}$. The entries of the graph sequence $\{G_k\}_{k=0}^{\infty}$ are chosen uniformly from the sample space of n-node-graphs with $m = 5 \times n$ edges. In *Fig. 2a*, we compare the performance of our method with other methods over a network with n = 50 nodes. Here, we cannot analytically compute θ^* and so the performance measure is the norm of the sum of gradients that is to converge to zero. As seen, TV-DAGA outperforms all the other methods after a few iterations. We also randomly generate problem set P with 1000 problems with 50 ≤ n ≤ 80 nodes and plot the Dolan-Moré performance profiles for accuracy after 100 iterations. The result is shown in *Fig. 2b*. As we see, TV-DAGA solves all the problems with the least error after 100 iterations, while FDGM, after 100 iterations, solves all problems with errors at most 80 times the error due to TV-DAGA. We also use problem set P and produce the Dolan-Moré performance profiles, with the performance measure being the number of iterations to reach within a $(10^{-10},10^{-35})$-

accuracy. The result is shown in *Fig. 2c*. Again, we see that TV-DAGA on all problems reaches the desired accuracy with the least number of iterations, while FDGM reaches the desired accuracy on all problems with a number of iterations being between 1.2 to 1.6 than the number of iterations due to TV-DAGA.

4 | The Canadian Institute for Advanced Research-10 Dataset

The Canadian Institute for Advanced Research (CIFAR-10) dataset contains 60000 colored images in 10 classes. Each image is represented by a vector of length 3072 and labeled by an integer number in [0,9]. For our implementation, since our classifier is binary, we pick all the vectors labeled by numbers 1 or 2. We relabel the selected vectors by -1 and 1, respectively. So, we have 10000 data points in $\mathbb{R}^{3072}$ with labels belonging to $\{-1,1\}$. We assume that our network has $n = 50$ nodes, and each node possesses $J = 200$ data points. In *Fig. 3*, we see the log plot for the norm of the sum of gradients at iteration k. We see that TV-DAGA outperforms all the other methods in terms of speed and accuracy.

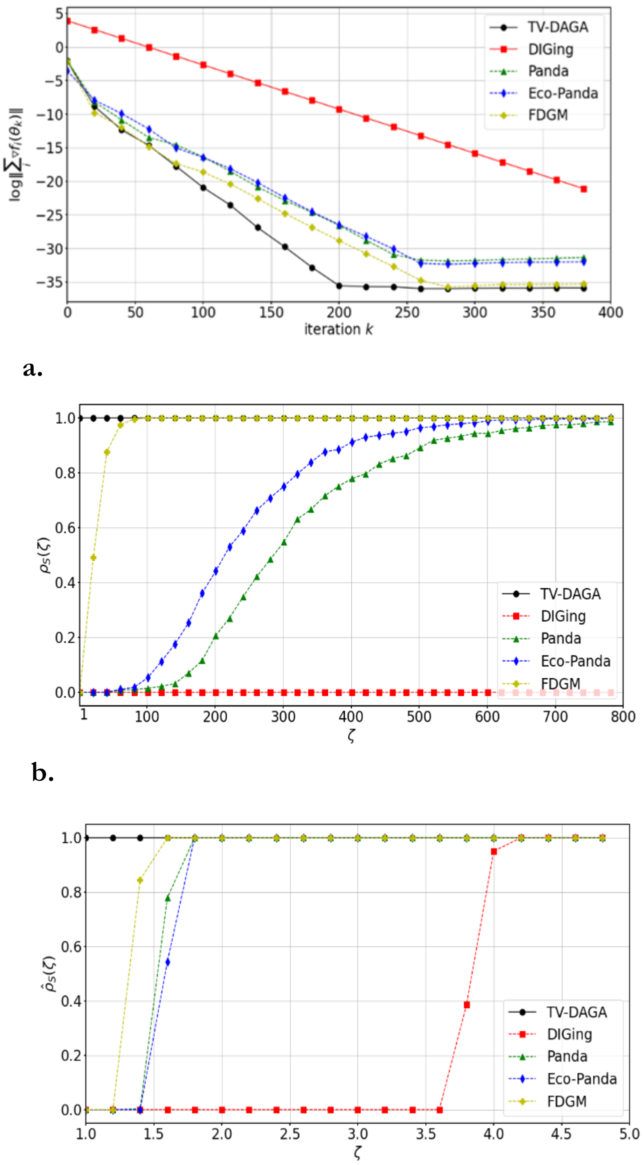


a.

b.

c.

Fig. 1. L2-regularized regression problem on; a. a single problem with $n = 100$, b. and, c. 1000 problems with $50 \leq n \leq 80$ nodes.



a.

b.

c.

Fig. 2. Logistic classification problem on; a. a single problem with $\mathbf{n} = 50$, b. and, c. 1000 problems with $50 \leq \mathbf{n} \leq 80$ nodes.

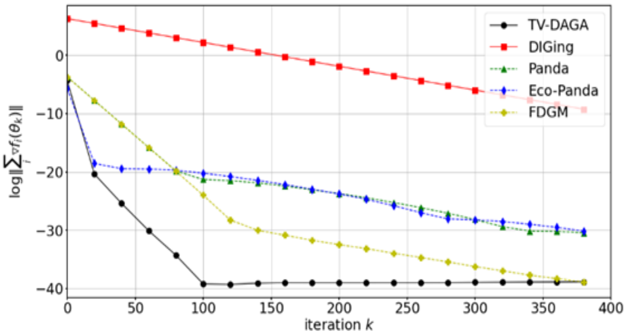


Fig. 3. Logistic classification problem on a network with n = 50 nodes and J = 200 data points per node, with data points being from the CIFAR-10 dataset.

5 | Conclusion

We proposed a time-varying dual accelerated gradient method for finding the minimum of the average of n strongly convex and smooth functions over a time-varying connected network. We proved that our method is R-linearly convergent if the network is always connected. We applied our method to 1000 randomly generated L_2-regularized least squares problems and 1000 randomly generated classification problems. In all cases, we observed that our algorithm outperformed all the other algorithms in terms of accuracy after 100 iterations as well as in terms of number of iterations to reach a specified combined absolute-relative accuracy. We also showed the out-performance of our method on classification problems with a real dataset named CIFAR-10.

Acknowledgments

We thank the Sharif University of technology for supporting this work.

Conflict of Interest

The authors declare no competing interests.

Data Availability

The datasets generated and analyzed during this study are available from the corresponding author upon reasonable request.

Funding

No funding was received for conducting this study.

References

[1] Nedic, A., Olshevsky, A., & Shi, W. (2017). Achieving geometric convergence for distributed optimization over time-varying graphs. *SIAM journal on optimization*, *27*(4), 2597–2633. https://doi.org/10.1137/16M1084316

[2] Jakovetić, D., Bajović, D., Krejić, N., & Jerinkić, N. K. (2016). Distributed gradient methods with variable number of working nodes. *IEEE transactions on signal processing*, *64*(15), 4080–4095. https://doi.org/10.1109/TSP.2016.2560133

[3] Maros, M., & Jaldén, J. (2018). Panda: A dual linearly converging method for distributed optimization over time-varying undirected graphs. *2018 IEEE conference on decision and control (CDC)* (pp. 6520–6525). IEEE. https://doi.org/10.1109/CDC.2018.8619626

[4] Maros, M., & Jaldén, J. (2019). ECO-panda: A computationally economic, geometrically converging dual optimization method on time-varying undirected graphs. *ICASSP 2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 5257–5261). IEEE. https://doi.org/10.1109/ICASSP.2019.8683797

[5] Rogozin, A., Uribe, C. A., Gasnikov, A. V, Malkovsky, N., & Nedić, A. (2019). Optimal distributed convex optimization on slowly time-varying graphs. *IEEE transactions on control of network systems*, *7*(2), 829–841. https://doi.org/10.1109/TCNS.2019.2949439

[6] Wu, X., & Lu, J. (2019). Fenchel dual gradient methods for distributed convex optimization over time-varying networks. *IEEE transactions on automatic control*, *64*(11), 4629–4636. https://doi.org/10.1109/TAC.2019.2901829

[7] Ding, T., Zhu, S., He, J., Chen, C., & Guan, X. (2021). Differentially private distributed optimization via state and direction perturbation in multiagent systems. *IEEE transactions on automatic control*, *67*(2), 722–737. https://doi.org/10.1109/TAC.2021.3059427

[8] Scaman, K., Bach, F., Bubeck, S., Lee, Y. T., & Massoulié, L. (2017). Optimal algorithms for smooth and strongly convex distributed optimization in networks. *34th international conference on machine learning,* (pp. 3027–3036). PMLR. https://doi.org/10.48550/arXiv.1702.08704

[9] Uribe, C. A., Lee, S., Gasnikov, A., & Nedić, A. (2020). A dual approach for optimal algorithms in distributed optimization over networks. *2020 information theory and applications workshop (ITA)* (pp. 1–37). IEEE. https://doi.org/10.1109/ITA50056.2020.9244951

[10] Nesterov, Y. (1983). A method for solving the convex programming problem with convergence rate o (1/k2). *Soviet mathematics doklady,* 27*(2),* 372-376. https://cir.nii.ac.jp/crid/1370862715914709505

[11] Monifi, E., & Mahdavi-Amiri, N. (2022). Time-varying dual accelerated gradient ascent: A fast network optimization algorithm. *Journal of parallel and distributed computing*, *165*, 130–141. https://doi.org/10.1016/j.jpdc.2022.03.014