Privacy-Preserving Distributed Processing Over Networks

Qiongxiu, Li

DOI (link to publication from Publisher):
10.54337/aau443832164

Publication date:
2021

Document Version
Publisher's PDF, also known as Version of record

PRIVACY-PRESERVING DISTRIBUTED PROCESSING OVER NETWORKS

BY
QIONGXIU LI

DISSERTATION SUBMITTED 2021

AALBORG UNIVERSITY
DENMARK

Privacy-Preserving Distributed Processing Over Networks

Ph.D. Dissertation

Qiongxiu Li

Department of Architecture, Design, and Media Technology
Aalborg University
Rendsburggade 14, 9000 Aalborg, Denmark

Dissertation submitted May, 2021

Dissertation submitted: May, 2021

PhD supervisor: Prof. Mads Græsbøll Christensen
Aalborg University

PhD co-supervisor: Prof. Richard Heusdens
Netherlands Defence Academy and Delft University of Technology

PhD committee: Professor Petar Popovski (chairman)
Aalborg University

Professor Marc Moonen
KU Leuven

Professor Antonio G. Marques
King Juan Carlos University

PhD Series: Technical Faculty of IT and Design, Aalborg University

Department: Department of Architecture,
Design and Media Technology

ISSN (online): 2446-1628
ISBN (online): 978-87-7210-950-3

Published by:
Aalborg University Press
Kroghstræde 3
DK – 9220 Aalborg Ø
Phone: +45 99407140
aauf@forlag.aau.dk
forlag.aau.dk

Printed in Denmark by Rosendahls, 2021

Curriculum Vitae

Qiongxiu(Jane) Li

Qiongxiu (Jane) Li was born in Hubei, China, in 1993. She received M. Sc. degree in information and communication engineering from Inha University, Incheon, South Korea, in 2017. She is currently a Ph.D. student at the Audio Analysis Lab, Department of Architecture, Design and Media Technology, Aalborg University, Aalborg, Denmark. Her research interests include distributed signal processing, machine learning, data privacy and security.

Abstract

Privacy has become a primary concern in modern world. Addressing the privacy issue is particularly challenging in the context of distributed processing due to many constraints such as absence of centralized coordination, limited computational resources and inevitable information exchange between different computing units. This thesis discusses on how to conduct signal processing over a network in a distributed manner without violating privacy. In particular, we focus on practical privacy-preserving solutions that are lightweight in terms of communication and computational cost. We first investigate existing privacy-preserving approaches which apply well-established cryptographic techniques into distributed processing tools. Secondly, we propose a new subspace perturbation based method that, instead of applying existing cryptographic techniques, directly exploits the potential of distributed processing tools such as distributed optimization for privacy-preservation. The proposed method is able to alleviate two fundamental limitations in existing approaches: the privacy-accuracy trade-off of differential privacy approaches and expensive communication cost incurred in secret sharing based approaches, respectively. Thirdly, based on the observation that all the above-discussed algorithms use the idea of inserting noise to mask the private data for privacy-preservation, we propose a new information-theoretical metric that is able to relate and compare all of them in a unified framework. Fourthly, we observe that there is typically a trade-off between the communication cost and privacy in noise insertion approaches and propose to address this trade-off, by making use of a quantization scheme in a particular way that the accuracy of the algorithm output is not deteriorated. Finally, continuing with the idea of exploring the potential of existing distributed processing tools for privacy-preservation, we take the first step to investigate the emerging graph signal processing tool and propose a privacy-preserving distributed graph filtering solution using noise insertion. This proposed solution has comparative performance compared with the above proposed subspace perturbation based distributed optimization approaches.

Resumé

Beskyttelse af privat information er blevet en stadig større udfordring i den moderne verden. Det er i særdeleshed vanskeligt at beskytte privat information i distribueret signalbehandling pga. de mange begrænsninger såsom fraværet af central koordination, begrænsede beregningsmæssige ressourcer samt den uundgåelige deling af information mellem forskellige enheder. Denne afhandling søger at svare på, hvordan signalbehandling kan blive udført i et netværk på en distribueret alt imens privat information beskyttes. Der fokuseres her på simple løsninger, der beskytter privat information, som kan bruges i forskellige anvendelser såsom i ressource-begrænsede trådløse sensornetværk. Først undersøges eksisterende informations-beskyttende tilgange, som anvender kendte kryptografiske metoder i værktøjer til distribuerede beregeninger. Dernæst forslås en ny metode baseret på underrums-perturbationer, som, i stedet for at anvende kryptografiske metoder, direkte udnytter potentialet i distribuerede beregningsværktøjer, såsom distribueret optimering, til at beskytte privat information. Den foreslåede underrums-pertuberingsmetode er i stand til at to grundlæggende begrænsninger i eksisterende metoder, nemlig afvegningen mellem beskyttelse of privat information og præcision i differential privacy-metoder og de store kommunikationsomkostninger i secret sharing-metoder. Baseret på oberservationen at alle de nævnte metoder er baseret på indsættelse af støj for at maskere og beskytte privat information forelås en ny informationsteoretisk metrik som tillader sammenligning af de eksisterende metoder. Endvidere observeres det, at der typisk er en afvegning mellem kommunikationsomkostningerne og beskyttelsen af privat information i metoder baseret på indsættelse af støj, men vha. en ny kvantiserigsmetode kan dette undgås. Endeling, i tråd med ideen om at udnytte potentialet i eksisterende distribuerede beregningsbærktøjer til beskyttelse af privat information, undersøges det hvordan privat information kan beskyttes i signalbehandling på grafer, som er et spirende felt, vha. støjindsættelse. Den foreslåede løsning har en ydelelse, der er sammenlignelig med den førnævnte optimerings-metode baseret på underrums-pertubering.

List of Papers

The main body of this thesis consists of the following papers:

[A] **Q. Li**, I. Cascudo, and M. G. Christensen, "Privacy-Preserving Distributed Average Consensus based on Additive Secret Sharing," in *Proc. Eur. Signal Process. Conf.*, pp. 1-5, 2019.

[B] **Q. Li**, and M. G. Christensen, "A Privacy-Preserving Asynchronous Averaging Algorithm based on Shamir's Secret Sharing," in *Proc. Eur. Signal Process. Conf.*, pp. 1-5, 2019.

[C] **Q. Li**, R. Heusdens, and M. G. Christensen, "Convex Optimisation-based Privacy-Preserving Distributed Average Consensus in Wireless Sensor Networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, pp. 5895-5899, 2020.

[D] **Q. Li**, R. Heusdens, and M. G. Christensen, "Convex Optimization-based Privacy-Preserving Distributed Least Squares via Subspace Perturbation," in *Proc. Eur. Signal Process. Conf.*, pp. 2110-2114, 2021.

[E] **Q. Li**, R. Heusdens, and M. G. Christensen, "Privacy-Preserving Distributed Optimization via Subspace Perturbation: A General Framework," in *IEEE Trans. Signal Process.*, vol. 68, pp. 5983 - 5996, 2020.

[F] **Q. Li**, R. Heusdens, and M. G. Christensen, "Communication Efficient Privacy-Preserving Distributed Optimization using Adaptive Quantization," in *Signal Process. Lett.* (submitted), 2021.

[G] **Q. Li**, M. Coutino, G. Leus and M. G. Christensen, "Privacy-Preserving Distributed Graph Filtering," in *Proc. Eur. Signal Process. Conf.*, pp. 2155-2159, 2021.

[H] **Q. Li**, J.S. Gundersen, R. Heusdens, and M. G. Christensen, "Privacy-Preserving Distributed Processing: Metrics, Bounds, and Algorithms," in *IEEE Trans. Inf. Forensics Security.*, 2021.

Contents

Preface

This thesis is submitted to the Technical Faculty of IT and Design at Aalborg University in partial fulfillment of the requirements for the Degree of Doctor of Philosophy. The thesis is constituted by two parts: a summary and a collection of papers. The summary consists of an introduction to the research project privacy-preserving distributed processing over networks, a background chapter introducing fundamentals, an overview of different solutions and the conclusions. After the summary, a collection of papers that have been published or submitted in peer-reviewed conferences or journals will be presented.

The work was carried out from March 2018 to March 2021 in the Audio Analysis Lab at the Department of Architecture, Design, and Media Technology (CREATE) at Aalborg University. First of all, I would like to thank my supervisor Mads Græsbøll Christensen for his guidance and endless support throughout my Ph.D life. He always trusted me and gave me the freedom to do what I am motivated for, such as choosing which research directions to explore. Another person I would like to thank is my co-supervisor Richard Heusdens who has influenced me a lot especially on what standard should a scientist hold and insist. I always admire his idealistic thinking and really appreciate his endless patience and the precious time he spent for discussing every single detail with me. Next, I would like to thank all my co-authors, Ignacio Cascudo, Mario Coutino, Geert Leus and Jaron Skovsted Gundersen, on the wonderful research we have done together. I also thank all my colleagues in Audio Analysis Lab and CAS group in TU Delft, and my friends for all the precious time we spent together. In addition, I would like to thank all members in SECURE project for the interesting discussions and cross-disciplined collaborations we have had together. Finally, I would like to thank my parents and my siblings Qiongyao Li, Qiongling Li and Hongyang Li who always stand in my back and support me through all ups and downs.

Qiongxiu Li
Aalborg University, May 24, 2021

Part I

Summary

1 Introduction

As the modern world is becoming increasingly digitized and interconnected, there has been a huge growth in the availability of information/datasets. In addition, these datasets are often located or collected in different computing units. Therefore, it is of great importance to allow inference over the sheer amount of heterogeneous datasets over different devices. It also brings new challenges as traditional centralized algorithms are too expensive and not practical, which solely rely on the so-called centralized authority/server to first collect data from different units and then process the collected data together at the centralized server. As a consequence, it requires novel tools that are able to collect, store and process the massive amount of datasets in a fully decentralized (or distributed) fashion. Compared to traditional centralized algorithms, distributed processing tools eliminate the dependence of a single centralized coordination by making use of the network nature. More specifically, each unit/node is allowed to carry out certain, usually simple, local computations by communicating only the necessary information with its neighboring nodes. Such distributed processing has many advantages, e.g., (1) it is more flexible and scalable to the number of nodes; (2) it is very commercial-friendly as it does not require an expensive centralized server to take care of all data collection and computations, instead it distributes the computational power and resources to different units; (3) it is more robust to the single point of failure because it is able to function properly even if a few nodes are missing or dropping out. The whole centralized system would, however, not work if the centralized server is being hacked or broken.

With the increasing concern of individual rights for data privacy, the privacy concern has become a key challenge that hinders the wide adaptation of distributed processing tools, or collaborative learning over multiple parties/nodes [1–6]. In particular, due to the fact that the primary computing devices for many people are phones and tablets [7, 8], these devices are embedded with many different sensors such as microphones and cameras. Data processing over these sensors would require data exchange between them and the exchanged data is usually very sensitive regarding individuals' privacy. The data collected from these sensors often contain sensitive personal information such as location, voice recordings and personal profiles. Therefore, such data processing are restricted by privacy concerns as it might cause the loss of pri-

vacy. For example, assume a scenario where a number of hospitals would like to learn a machine learning model to predict cancer at an early stage. A better and more accurate model which has higher predicting accuracy can be achieved, if these hospitals are able to collaborate together to allow learning over all datasets available to them. However, the dataset collected at each hospital is very sensitive as it contains patients' health records etc. Without addressing this privacy concern, such collaborations are not possible, or even not legally allowed according to the regulations like the EU General Data Protection Regulation [9].

To summarize, it is very important to develop privacy-preserving distributed processing tools that are able to conduct signal processing or data learning over networks in a distributed manner without violating privacy. It is quite challenging to develop such tools as it is an interdisciplinary research field that requires knowledge from different fields such as information theory, statistics, cryptography, differential privacy, distributed signal processing, machine learning and more.

1.1 Research questions and structures

In this thesis we focus on studying and proposing new methods to address the privacy issues in distributed processing over networks under different assumptions and constraints. In particular, we attempt to answer the following research questions.

1. In various applications such as wireless sensor networks, there are many practical constrains like low computational power and limited memory. How can we develop effective privacy-preserving distributed processing tools that are lightweight in terms of both communication cost and computational complexity?

2. One typical way to address the privacy issue is to apply well-established cryptographic techniques such as secure multiparty computation (SMPC) and differential privacy (DP), into different distributed processing tools for privacy-preservation. However, these techniques are not originally defined in the context of distributed processing over networks and thereby each of them has its own limitations. An important question that should be addressed is: can we instead develop new privacy-preserving methods that are able to address these limitations other than directly applying these cryptographic tools?

3. Given various privacy-preserving algorithms that are derived from different contexts and with different assumptions, how to choose an appropriate algorithm for a specific problem at hand? That is, how to relate these algorithms to each other and compare them?

Accordingly, we structure the summary of the thesis in the following way. An introduction of privacy-preserving distributed processing over networks is given

first. After that, the necessary background and fundamentals for understanding remaining contents of the thesis will be introduced in Section 2. An overview of the existing cryptographic tools for achieving privacy-preservation will be given in Section 3 and 4. Finally, the conclusions of the thesis will be given in Section 5, including both a paper-wise contribution summary and a discussion of future research directions.

2 Background

In order to facilitate the understanding the fundamental concepts in privacy-preserving distributed processing over networks, we demonstrate a toy example in Fig. 1. We assume three smart speaker companies/institutions, each of which has a local dataset collected from its end users. These companies would like to increase the performance of its own smart speakers by improving the speech recognition accuracy. The goal of them is to cooperate together to learn a global speech recognition model based on all their datasets. However, none of them would like to reveal it own dataset to others as it would lose its end users' data privacy. We remark that there are several important questions to be considered before proposing solutions. (1) How to model the relationship between different parties in the network? As an example shown in the figure, not all companies in the network can talk/communicate with each other directly, e.g., company 2 and 3. How would this be modelled? (2) What does distributed processing mean? How is it defined? (3) What does privacy mean or how is privacy defined?; (4) What if there are some security attacks in the network? For example, if there are some non-trustworthy/ malicious parties in the network, will they compromise the privacy of the other ones? E.g., how to model the case that company 1 and 2 are from the same institution and they pretend to participate in this computation but their goal is to infer the local dataset of company 3 or to manipulate the learning model to be inaccurate for company 3. (5) How strong is the malicious party assumed? Do they have infinite computational power and resources?

In what follows, we will answer the above questions one by one. Section 2.1 explains how to model the network as a graph with a number of nodes and edges, which represent the relationship between different parties. Section 2.2 and 2.3 explain what distributed processing is and defines privacy, respectively. Section 2.4 introduces the so-called adversary model, which is used to simulate different security attacks. Section 2.5 introduces different assumptions for the adversary. Finally, in Section 2.6 we summarize the main requirements that a privacy-preserving distributed solution should satisfy.

2.1 Network setup

A distributed network can be modelled as a graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$, where $\mathcal{N} = \{1, \ldots, n\}$ denotes the set of n nodes/parties/agents and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ denotes

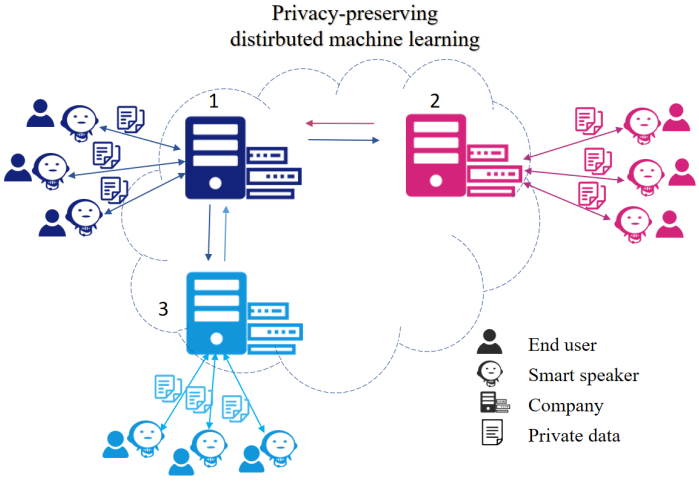


Fig. 1: Example

the set of m edges. Two nodes i, j are connected with each other if there is an edge between them, i.e., $(i, j) \in \mathcal{E}$. Let $\mathcal{N}_i = \{j \mid (i, j) \in \mathcal{E}\}$ denote the set of neighboring nodes of node i and $d_i = |\mathcal{N}_i|$ denotes the degree of node i. Define matrix $B \in \mathbb{R}^{m \times n}$ based on the edge set, for each edge $(i, j) \in \mathcal{E}$ we set

$$B_{i|j} = \begin{cases} 1, & \text{if } (i,j) \in \mathcal{E} \text{ and } i < j, \\ -1, & \text{if } (i,j) \in \mathcal{E} \text{ and } i > j, \\ 0, & \text{if } (i,j) \notin \mathcal{E}. \end{cases} \tag{1}$$

2.2 Distributed processing

Let s_i denote the local data held by node i, we denote $\boldsymbol{s} = [s_1, \ldots, s_n]^\top$ by stacking all local data together, where $(\cdot)^\top$ denotes matrix transpose. Similarly, we denote y_i as the desired output of node i and thus $\boldsymbol{y} = [y_1, \ldots, y_n]^\top$. The main goal of distributed processing over a network is to jointly compute a function, i.e.,

$$f : \mathbb{R}^n \mapsto \mathbb{R}^n, \boldsymbol{y} = f(\boldsymbol{s}), \tag{2}$$

without any centralized coordination. That is, each node i is only allowed to communicate or exchange information with its neighbors $j \in \mathcal{N}_i$. For example, many applications require a distributed data aggregation protocol such as those used in average consensus algorithms [10, 11], graph filtering methods [12, 13], probabilistic inference algorithms [14, 15], gossip algorithms [16, 17] or convex optimization algorithms (ADMM [18], PDMM [19–21]).

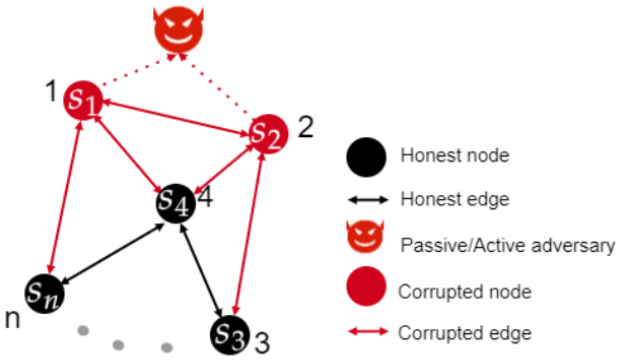


Fig. 2: Passive/Active adversary

2.3 Privacy definition

In this context, we define the private data to be protected as the local data s_i held by each node, which often contains individual's sensitive personal information that can lead to severe privacy leakage like identity, health condition and personal behavior. For example, with voice signals, health condition such as Parkinson's disease [22, 23] can be diagnosed. Power consumption data can reveal the activities of the householders [24]. Sensitive information such as the identity of an individual can be revealed using only an anonymous 10-ride bus ticket [25].

2.4 Adversary model

When considering privacy, the so-called adversary model should be specified as it evaluates the algorithm robustness when dealing with different security attacks. We now introduce a few adversary models which are relevant in this context.

Passive adversary

The passive adversary model is also called a semi-honest or honest-but-curious adversary model. All nodes in the network are classified into two types based on whether they are colluded by the adversary. The colluded nodes are referred to as corrupted nodes and the rest of non-colluded nodes are called honest nodes. See Fig.2 for a toy example. The corrupted nodes will follow the instructions of the algorithm but they can share information together like their own private data and the messages transmitted from and to them. The goal of the passive adversary is to infer the private data of the honest nodes by collecting information from all the corrupted nodes. Note that as long as there is one corrupted node at either end of an edge, all information transmitted along this edge will be revealed to this corrupted node, thereby to the passive

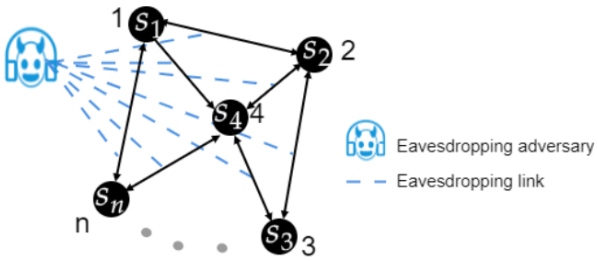


Fig. 3: Eavesdropping adversary

adversary. An algorithm is more robust against a passive adversary if it can tolerate a higher number of corrupted nodes without revealing the private data of the honest nodes.

Active adversary

The active adversary works similarly as the passive adversary by colluding a number of nodes. The main difference is that the actively corrupted nodes are allowed to deviate from the algorithm instructions. For example, they can lie to their neighbors by sharing wrong data with them. Therefore, in addition to infer the private data of the honest nodes, they are able to make more damage like manipulating the result of the algorithm to be incorrect. As a consequence, dealing with the active adversary is more challenging than the above passive adversary. That is, if an algorithm is robust to an active adversary implies that it is also robust to the passive adversary. The reverse is, however, not true.

Eavesdropping adversary

Compared to the passive and active adversaries, the eavesdropping adversary does not work by colluding some nodes in the network but by listening to the messages transmitted between each pair of neighboring nodes. The goal of the eavesdropping adversary is to infer the private data of the nodes by collecting the eavesdropped information from the communication channels, i.e., the edges, between nodes (see Fig.3 for an example). Compared to the above passive adversary, the eavesdropping adversary model has received very little attention in the literature since it can be addressed by assuming securely encrypted channels [26] such that no information can be eavesdropped. However, we argue that the eavesdropping adversary is indeed very relevant for the context of distributed processing. The reason is that, due to its decentralized nature, distributed processing solutions are iterative and thus require the communication channels to be used many times. It is thus too expensive to assume that all communication channels over all iterations are securely encrypted. As a consequence, when designing algorithms it is very important to minimize the expense of secure channel encryption.

Having introduced different adversary models, we note that compared to the active adversary model, the passive adversary model has been more widely investigated in a distributed setting [27]. In addition, for algorithms which consider multiple adversaries models, we assume that these adversaries can cooperate together. For example, if both the passive and eavesdropping adversaries are considered, they may be able to share information together to infer the private data of the honest nodes.

2.5 Security model

The security model is classified into two categories based on the assumption of the adversaries' computation power, i.e., whether it is computationally bounded or not. If the adversary is assumed computationally bounded, the constructed security model is called computational (also called conditional) security. The privacy is guaranteed by assuming that the adversary cannot decrypt the private data efficiently, e.g., in polynomial time. On the other hand, information-theoretical (also called unconditional) security model is constructed if the adversary is assumed to have unbounded computation power but can not infer the private data with the amount of available information.

In general, an information-theoretical security model is preferred over a computational security model in the context of distributed setting as it considers a stronger, i.e., computationally unbounded, adversary and is often more lightweight in both computational complexity and communication costs [28]. Therefore, in this study we will focus on information-theoretic security approaches.

Noise insertion for information theoretical security

A typical way to achieve information theoretical security is to design certain noise insertion methods that are able to obfuscate the private data

$$\forall i \in \mathcal{N} : s_i^r = F_r(s_i, r_i), \tag{3}$$

where F_r denotes the designed noise insertion function and r_i denotes the inserted noise for node i. After the noise insertion, how well is the private data s_i protected depends on how much information can be inferred from knowing s_i^r. Denote s_i and s_i^r as realizations of random variables S_i and S_i^r, respectively. One natural metric to quantify how much information about S_i can be revealed given the knowledge of S_i^r, or vice versa, is mutual information [29].

$$I(S_i; S_i^r) \leq \delta, \tag{4}$$

where δ denotes the threshold of privacy leakage, the smaller it is, the better the privacy is being protected.

A simple yet effective way to design the noise insertion function F_r is to add noise to the private data to mask it, i.e.,

$$\forall i \in \mathcal{N} : s_i^r = s_i + r_i. \tag{5}$$

How to design the inserted noise r_i depends on several factors such as the distribution of the private data. In the coming two sections we will come back to this in more detail.

2.6 Problem formulation

Given a privacy-preserving distributed processing algorithm, denote $\hat{y}_i$ as the algorithm output for node i and denote $\mathcal{V}$ as the set of information collected by the considered adversaries throughout the whole algorithm for the purpose of inferring the private data of the honest nodes. Let $\mathcal{N}_h$ and $\mathcal{N}_c$ denote the set of honest and corrupted nodes, respectively. We conclude that there are two main requirements to be considered when evaluating the performance of a privacy-preserving distributed signal processing algorithm: output correctness and individual privacy.

Output correctness

Each node i would like its estimated output $\hat{y}_i$ to be as close as possible to its desired output y_i. One widely adopted metric for quantifying the output correctness is the squared error $\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2$.

Individual privacy

Throughout the execution of the algorithm, each honest node $i \in \mathcal{N}_h$ would like to protect its private data s_i from being revealed to the adversaries. That is, from an information-theoretic point of view, with the knowledge $\mathcal{V}$ available to the adversaries it is insufficient to infer what is S_i. We will use the mutual information $I(S_i; \mathcal{V})$ to quantify how much information about S_i is revealed given the knowledge of $\mathcal{V}$.

3 Secure multiparty computation based approaches

The main concept of secure multiparty computation (SMPC) [30] is to securely compute a function over multiple parties/nodes in a network without revealing each node's private data to others. Consider a scenario that there are n nodes each with private data $s_i, i \in \mathcal{N}$, they would like to collaborate together to compute a function over their private data, i.e., $f(s_1, s_2, \ldots, s_n)$ but each of them would not like to reveal its own private data. In SMPC, it defines that

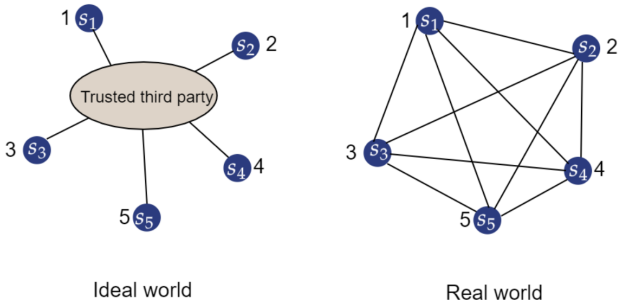


Fig. 4: Ideal world v.s. Real world in SMPC

the perfect solution to solve this problem requires a trusted third party (TTP), which collects the private data from all nodes and then computes the function output, after that send the output to each node. Such scenario is defined as the so-called ideal world (see the left-hand plot in Fig. 4 for a toy example) because it assumes the TTP can not be corrupted, thus the private data is protected and in the meantime each node realizes the goal of knowing the function output. However, in real world applications, a TTP might not be available. Therefore, the goal of SMPC is to design a protocol to replace the TTP, by relying only on the information exchange between all nodes in the network (see the right-hand plot in Fig. 4).

This goal of SMPC fits well with the purpose of privacy-preserving distributed processing approaches. Therefore, many existing approaches make use of techniques from SMPC for privacy-preservation, for example in machine learning applications [31–38]. There are mainly three kinds of privacy-preserving approaches using different SMPC techniques. The first type of techniques is homomorphic encryption (HE) [39–41], which is able to allow all computations in the encrypted domain. As shown in Fig. 5, all private data are first encrypted and then computation is conducted on the encrypted data, after that each node receives the encrypted output and then decrypt it. Overall, HE techniques establish a computational security model for protecting the private data. HE has been applied in various problems such as distributed average consensus [42–44], convex optimization [45–49] and machine learning [11, 50–53]. The second technique is garbled circuits (GC) [54, 55] which allows to compare inputs of two nodes without knowing each node's input. GC is thus adopted in the cases when secure comparison is involved [56]. Finally, the last technique is called secret sharing [41]. Its main idea is to first split each private data into pieces and distribute them to different nodes in the network. The private data is protected because it can only be reconstructed if and only if a large number (specified by a threshold) of nodes are corrupted. Various secret sharing schemes such as additive secret sharing and Shamir's secret sharing have been widely adopted in recent studies [57–62]. Note that among these three types of approaches, both HE and GC based approaches

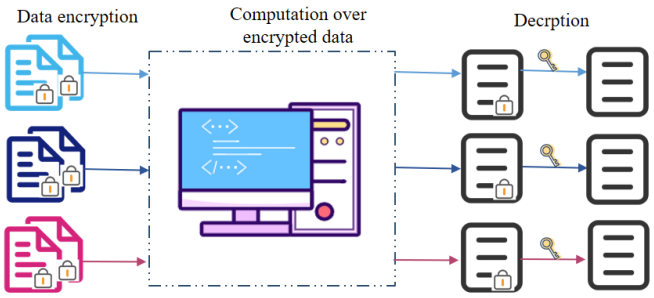


Fig. 5: Homomorphic encryption

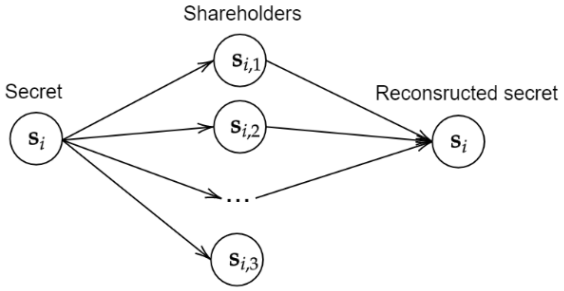


Fig. 6: Secret sharing

are based on computational security model and thus are very computationally demanding. Secret sharing based approaches, on the other hand, are based on an information-theoretic security model which is very lightweight in terms of computational complexity. Since this study mainly focuses on information-theoretic security, in what follows we will introduce more details about secret sharing techniques.

3.1 Fundamentals of secret sharing

Imagine a scenario that a number of scientists cooperate together to work on a confidential project [63], the confidential documents are locked up in a cabinet. The cabinet can only be unlocked if the majority of scientists agree. Secret sharing schemes [30] can provide a perfect solution to this problem.

That is, assume a secret holder i would like to share his or her secret s_i among a group of n nodes. By adopting a secret sharing scheme, it guarantees that the secret s_i can be reconstructed if and only if a sufficient amount of nodes agree to collaborate, otherwise no information regarding the secret s_i will be revealed. A secret sharing scheme mainly comprises two parts, see Fig.6 for a toy example.

- **Share construction**: The secret share algorithm takes both secret s_i and

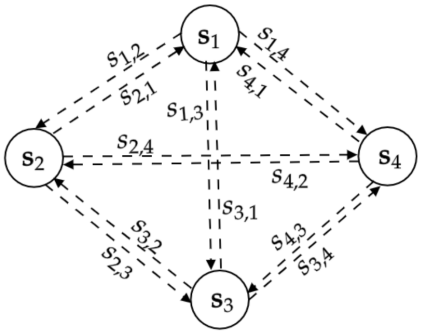


Fig. 7: Secret sharing over a network

some randomness r_i as inputs and then outputs n shares, i.e.,

$$F_{\mathrm{S}}(s_i, r_i) = (s_{i,1}, s_{i,2}, ..., s_{i,n}), \tag{6}$$

where $s_{i,j}$ denotes the j^{th} share of secret s_i.

- **Secret reconstruction**: By collecting a sufficient number (denoted by a threshold t) of shares from some subset of the nodes $\{l_1, \ldots, l_t\}$, the secret s_i can be reconstructed, i.e.,

$$F_{\mathrm{R}}(s_{i,l_1}, s_{i,l_2}, ..., s_{i,l_t}) = s_i, \tag{7}$$

3.2 Secret sharing over a network

Assume the case that each node in the network has a secret, denoted as $s_1, s_2, \ldots, s_n$ and they would like to cooperate together to compute a function $f(s_1, s_2, \ldots, s_n)$ with their secrets as inputs. In this case, each node is both a secret holder and a share receiver in the network. In order to conduct secret sharing over the network, each node will first construct n shares based on its own secret, after that it will send $n-1$ shares to every other node in the network and receive shares from others as well. See Fig. 7 for an example with $n = 4$ nodes in the network. To guarantee that the function output can be computed without revealing each node's secret, a new function should be defined, which takes all the shares as inputs and outputs the same result as the original function, i.e.,

$$f'(s_{1,1}, s_{1,2}, \ldots, s_{2,1}, s_{2,2}, \ldots, s_{n,n}) = f(s_1, s_2, \ldots, s_n).$$

3.3 Privacy-preserving summation as an example

To explain how to apply secret sharing in detail, we will use the privacy-preserving summation as an example. The goal is to compute the sum of all

private data over the network, i.e.,

$$\forall j \in \mathcal{N} : \boldsymbol{y}_j = f(s_1, s_2, \ldots, s_n) = \sum_{i \in \mathcal{N}} s_i, \tag{8}$$

To relate to the idea of using noise insertion to achieve information theoretic security, we now show how to design the noise insertion method using additive secret sharing. Define $\mathbb{Z}_p$ as a group of integers modulo p and p is sufficiently large such that such that $\sum_{i \in \mathcal{N}} s_i < p$. The reason of working with modulo operation is to achieve perfect security, which we will explain later. Let each node i select $n-1$ shares $\{s_{i,j} \in \mathbb{Z}_p\}_{j \in \mathcal{N}, j \neq i}$ uniformly at random and set share $s_{i,i} = (s_i - \sum_{j \in \mathcal{N}, j \neq i} s_{i,j}) \bmod p$. Note that s_i can only be reconstructed if and only if all n shares are given. Based on (5), we can design the noise insertion function as

$$\forall i \in \mathcal{N} : s_i^r = (s_i + r_i) \bmod p \tag{9}$$

where the inserted noise is constructed using the shares:

$$\forall i \in \mathcal{N} : r_i = \big(\sum_{j \in \mathcal{N}_i} (s_{i,j} - s_{j,i}) \big) \bmod p. \tag{10}$$

The sum is guaranteed to be preserved, i.e.,

$$\begin{aligned} \sum_{i \in \mathcal{N}} s_i^r \bmod p &= \big(\sum_{i \in \mathcal{N}} (s_i + r_i) \big) \bmod p \\ &= \sum_{i \in \mathcal{N}} s_i + \big(\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}_i} (s_{i,j} - s_{j,i}) \big) \bmod p \\ &= \sum_{i \in \mathcal{N}} s_i, \end{aligned} \tag{11}$$

since $\big(\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}_i} (s_{i,j} - s_{j,i}) \big) \bmod p = 0$. Note that with the help of modular arithmetic, S_i^r is uniformly distributed on $\mathbb{Z}_p$ and thus statistically independent of S_i (see [30, Section 1.3.1] for further details). Consequently, no information about S_i can be revealed by observing S_i^r, i.e., perfect security is achieved:

$$I(S_i; S_i^r) = 0. \tag{12}$$

3.4 Limitations

By inspecting the example shown in Fig. 7, we note that there are several limitations in adopting secret sharing techniques into distributed signal processing for privacy-preservation.

- Assumption of fully-connected graph: Secret sharing often requires a fully-connected graph topology, i.e., each node is connected with every

other node, to guarantee that the shares can be distributed and the output of the function is correct. However, in practical applications fully-connected graphs may not be available as they does not scale with the number of nodes in the network.

- High communication cost: Due to the fact that secret sharing usually requires to distribute shares over a fully-connected network, the amount of communication times will increase quadratically with the number of nodes in the networks. This makes it impractical for large sensor networks.
- Computationally demanding when dealing with an eavesdropping adversary: Secret sharing approaches usually assume that the communication channels are securely encrypted such that no eavesdropping can be conducted. However, as mentioned before, distributed processing algorithms often require a large number of iterations for converging to the optimum result. This makes it very expensive to assume all communication channels over all iterations are securely encrypted.
- Requirement of honest neighboring nodes when dealing with a passive adversary: It requires each honest node to have at least one honest neighboring node to ensure that at least one share is not known to the passive adversary such that the private data can not reconstructed. That is, secret sharing is not robust to $n-1$ passive corruptions.

One way to deal with above three limitations is to construct several servers in the networks [64, 65], such that each node only needs to send shares to these servers. Therefore, the communication cost will only increase linearly with the number of nodes. The fully-connected graph assumption is also relaxed to the case that only these servers need to form a fully-connected graph. Finally, the expense of secure channel encryption will also be reduced. However, assuming a number of servers in the network might be expensive and sometimes is not available in many applications. Another way to deal with the limitation of graph topology is to establish virtual edges between nodes to form a virtual graph which is fully-connected, however, it require expensive encryption technique to construct the virtual edges and thereby increasing the computational complexity.

4 Differential privacy based approaches

Differential privacy (DP) based approaches, as the name suggested, adopt differential privacy techniques [66–70] to achieve privacy-preservation. The main idea is to, instead of sharing the private data directly, add careful noise to the private data to achieve a noisy version and then share the noisy one to others. The more noise inserted, the better the private data is being protected.

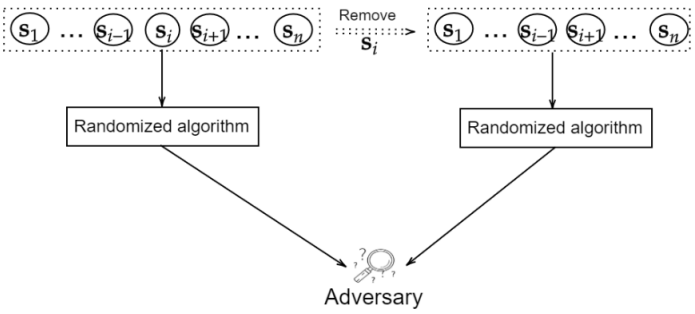


Fig. 8: Differential privacy

Note that DP considers an extreme scenario where even if there is only one honest node in the network, its privacy can still be protected. Such strong privacy guarantee makes DP very popular and has been widely adopted in many applications such as distributed average consensus [71–75], distributed optimization [76–82] and distributed machine learning [83].

4.1 Fundamentals

Consider a scenario where you are asked to participate in a survey which requires you input your sensitive data like political preference. You would be reluctant to participate in this survey as there is a risk to reveal your sensitive data. Differential privacy is a perfect solution to address your concern. In fact, it guarantees your privacy under an extreme case that even though all other participants except you are corrupted by the adversary, your personal data can still be guaranteed to be protected from being revealing to others. That is, based on the two results of the two surveys, one with and one without your participation, the adversary would not be able to infer out what is your private data. In Fig. 8 we show a toy example to illustrate the main idea. From an information-theoretic point of view, it means that the posterior guess of the adversary about your personal data with your participation is only slightly better than the prior guess when your participation is absent. This fact can be mathematically described as follows:

Denote $\boldsymbol{s}^{-i} \in \mathbb{R}^{n-1}$ as a so-called adjacent vector of $\boldsymbol{s}$ by removing s_i from the vector $\boldsymbol{s}$. Let Ω_i denote the range of s_i. Assume a randomized algorithm F which protects its input from being revealed to others. Let $\mathcal{Y}$ denote the range of output of algorithm F . Given $\epsilon \geq 0$, the algorithm F is called ϵ-differentially private if it satisfies, for all pairs of adjacent vectors $\boldsymbol{s}$ and $\boldsymbol{s}^{-i}$, and all sets $\mathcal{Y}_s \subseteq \mathcal{Y}$,

$$\forall s_i \in \Omega_i : \quad \frac{P(F(\boldsymbol{s}) \in \mathcal{Y}_s)}{P(F(\boldsymbol{s}^{-i}) \in \mathcal{Y}_s)} \leq e^{\epsilon}. \tag{13}$$

By inspecting the above equation, we can see that it require for any $s_i \in \Omega_i$,

the posterior guess of the private data where s_i is included in the input is only slightly better (e^ϵ times of) the prior guess when s_i is not included. We note that differential privacy has the following advantages: (1) it does not depend on the distribution of the private data; (2) ϵ measures the privacy in the worst-case scenario as it is the maximum among all input ranges, i.e., $\forall i \in \mathcal{N}, \forall s_i \in \Omega_i$,; (3) compared to secret sharing based approaches, it has no assumption on the graph topology and the communication cost is lightweight; (4) it does not require any honest neighboring node for privacy-preservation, i.e., it is robust to $n-1$ passive corruptions.

4.2 Privacy-preserving summation as an example

To achieve differential privacy, each node can choose to insert independent zero-mean noise using certain distributions such as Laplacian [66]. Hence, the private data s_i of node i is protected even though all other $n-1$ nodes are passively corrupted, as the inserted noise r_i is independent to all other noise and thus is unknown. According to the law of large number, the sum will be preserved if and only if $n = |\mathcal{N}| \to \infty$. That is,

$$\sum_{i \in \mathcal{N}} s_i^r = \sum_{i \in \mathcal{N}} (s_i + r_i) = \sum_{i \in \mathcal{N}} s_i, \tag{14}$$

as $\lim_n \to \infty \sum_{i \in \mathcal{N}} r_i = 0$. Here we can see that as the assumption of $n \to \infty$ is not realistic in practice, so there is always a trade-off between the privacy and the accuracy of the sum result.

4.3 Limitations

The strong privacy guarantee ensured by differential privacy is both a feature and a weak point of itself. Its main limitations are listed below:

- Difficulty in realization: Since differential privacy considers a worst case and privacy should be guaranteed in any situation, for example scenarios for all prior distribution of the private data, such strong privacy guarantee is very difficult, sometimes impossible, to realize in practical applications [84–86].

- Privacy-accuracy trade-off: To realize differential privacy, it often requires to design the noise to obfuscate each private data in a way such that (13) can be satisfied. However, there is a price to pay: the inserted noise often affects the algorithm output to be inaccurate. The more noise inserted, the less privacy leakage the node has, but the output will be more inaccurate. Therefore, there is a fundamental trade-off between privacy and accuracy.

- Sensitivity to noise insertion: Proceeding with the privacy-accuracy trade-off, one follow-up challenge is to design a proper budget for differential privacy, i.e., how to control the ϵ in a way that the algorithm accuracy is not severely compromised. To address this problem we need to investigate how sensitive is the algorithm output in terms of the inserted noise. However, except for very simple problems such as the summation mentioned above, it is very difficult to analytically track the sensitivity.

Many studies are focusing on relaxing the differential privacy to make it more practically achievable [68, 69, 87, 88]. For example [87, Theorem 1] showed that for the right hand side of (13) if we consider to compute the expected value instead of the maximum over all $s_i \in \Omega_i$, it will reduce to the Kullback-Leibler divergence

$$D_{\mathrm{KL}}(P(F(\boldsymbol{s})\|P(F(\boldsymbol{s}^{-i})) \leq \epsilon, \tag{15}$$

and can further be relaxed to the following conditional mutual information:

$$I(S_i; Y|\{S_j\}_{j\in\mathcal{N}\setminus\{i\}}) \leq \epsilon. \tag{16}$$

5 Conclusions

The main body of this thesis consists of a collection of papers: A-H. In what follows we will first summarize each paper's contribution and give discussions on future research directions.

5.1 Contributions

Paper A This paper proposes a privacy-preserving distributed average consensus algorithm using the additive secret sharing technique. The main idea is to add one pre-processing step to obfuscate first each private data and then use the obfuscated data for further processing. Such pre-processing only requires one time additive secret sharing and is conducted only between neighboring nodes, i.e., no fully-connected graph is required. In addition, due to the fact that the average result remains identical if the sum of the private data over the network is preserved, the result of the proposed method is accurate because the sum of all the shares is constructed to be equal as the sum of all private data. As a consequence, the proposed approach is very lightweight and alleviates the privacy-accuracy trade-off incurred in differential privacy approaches. Moreover, the proposed approach is able to protect the private data of a honest node against a passive adversary as long as it has at least one honest neighboring node.

Paper B In addition to the passive adversary model considered in Paper A, we would like to consider a more malicious model for the distributed average consensus application, i.e., the active adversary model which allows the corrupted

nodes to deviate from the instructions of the algorithm instructions. In SMPC, Shamir's secret sharing is a typical technique for addressing the active adversary model. However, it requires a fully-connected graph for share distribution. In order to address this limitation, in this paper we propose to relax such assumption by requiring only clique-based graph. That is, we only assume that each node and at least two of its neighboring nodes can form a fully-connected subgraph, i.e., clique, such that Shamir's secret sharing can be applied. Overall, we propose a new privacy-preserving distributed average consensus algorithm which considers an active adversary model.

Paper C Instead of directly applying existing cryptographic techniques for privacy-preservation, in this paper we propose to directly explore the potential of distributed signal processing tools for protecting privacy. In particular for distributed optimization, we observe that the updating of the so-called primal-dual method of multipliers (PDMM) is conducted in a so-called convergent subspace determined by the graph topology. By exploiting this subspace property, we propose a novel subspace perturbation idea, which achieves a privacy-preserving distributed average consensus solution by inserting noise into the non-convergent subspace, which is orthogonal to the convergent subspace. Due to the orthogonal property, the average result is not affected and in the meantime the privacy is protected by the inserted noise. That is, the proposed solution circumvents the privacy-accuracy trade-off incurred in differential privacy approaches, and it considers two adversary models: passive and eavesdropping. For a passive adversary, each honest node's privacy is guaranteed as long as there is one honest neighboring node. Additionally, it is very lightweight when dealing with the eavesdropping adversary because it requires that only the communication channels in the initialization step are encrypted.

Paper D Continuing with the idea showed in Paper C, in this paper we extend extend this subspace perturbation idea for the application of distributed least squares. As the data involved in distributed least squares are often vectors/metrics but not the scalars, we then extend the information theoretic privacy guarantee in distributed average consensus to the high dimension case. In addition, we verify the proposed idea using not only PDMM but also ADMM, which demonstrates its general applicability. Overall, a subspace based privacy-preserving distributed least squares approach is proposed. It has superior performance compared with SMPC based approaches in terms of computational complexity and communication cost.

Paper E The subspace perturbation idea is not only applicable to both distributed average consensus and distributed least squares, it is indeed generally applicable to all distributed convex optimization problems. In addition, we prove that the subspace property is not only true for PDMM and ADMM but for many distributed optimizers like dual ascent, based on the fact that the

incidence matrix of a connected graph is always rank deficient. Overall, in this paper we summarize things together and propose a general distributed optimization framework using subspace perturbation which is able to protect privacy by inserting noise in the non-convergent subspace.

Paper F The above idea of using noise insertion to achieve privacy-preservation is widely adopted in the literature. However, there is a fundamental trade-off between privacy and communication bandwidth in such techniques. This is because the more noise inserted, the better privacy level can be achieved, which will inevitably increase the entropy of the inserted noise and thereby its bitrate. In this paper, we propose to make use of an adaptive quantization scheme [89, 90] to alleviate such trade-off for distributed optimization. The proposed approach is very advantageous as it does not compromise the accuracy of the optimization output by considering both privacy and quantization. That is, the proposed approach is able to achieve a high privacy level in distributed optimization with very low computational costs without compromising the algorithm accuracy.

Paper G In the above papers we mainly propose privacy-preserving solutions using distributed optimization tools. However, the potential of many other signal processing tools like graph signal processing [91–96], which is an emerging field, has not been fully explored yet. In this paper, we take the first step to propose a novel privacy-preserving distributed graph filtering algorithm by designing a noise insertion method. In order to ensure that the inserted noise does not severely degrade the accuracy of the filtering output, we calibrate the effects of the inserted noise into the target graph filter. The proposed approach is very lightweight in terms of both communication cost and computational complexity as it allows a distributed implementation [13, 97]. However, the approximated graph filtering output might not be as accurate as its non-privacy-preserving counterpart.

Paper H As we discussed above, there are many ways to address the privacy issue in distributed processing over network. One typical way is to integrate existing well-established cryptographic techniques for privacy-preservation, for example the SMPC and differential privacy techniques. Another way is to establish new privacy-preserving solutions by directly exploring the potential of different distributed signal processing tools like distributed optimization and graph signal processing. However, it raises questions like 'which method is the optimum and how are they related to each other?'. In this paper we would like to answer these questions. The main challenge comes from the fact that these methods are developed from different contexts and thus have different metrics. In order to be able to quantify these algorithms in a consistent manner, we propose a mutual information based metric, which is able to relate all metrics in these approaches, to quantify both privacy and utility. By utility we mean how

close is the privacy-preserving output and the non-privacy-preserving output from an information-theoretic point of view. Moreover, we analyze the bounds of both privacy and utility and discover that the lower bound of privacy is the key to link these approaches. We use a concrete example application, distributed average consensus, to verify the effectiveness of the proposed metrics. Finally, we provide some principles to help to choose an appropriate algorithm given a specific problem at hand.

To conclude, the papers answered the research questions listed in Section 1.1. Papers A-G are related to the first question where all the proposed privacy-preserving solutions are lightweight in terms of both communication and computation complexity. Moreover, among them, Papers C-G answer the second research question by proposing novel privacy-preserving solutions without relying existing cryptographic techniques like SMPC and differential privacy, but exploring the potential of distributed processing tools such as distributed optimization and graph signal processing for protecting privacy. In particular, the proposed subspace perturbation-based distributed optimization approach is very general and can be applied in a very broad context for example distributed machine learning. Finally, paper H answers the last research question by explicitly comparing and relating different privacy-preserving solutions including SMPC, differential privacy and subspace perturbation based approaches. Overall we conclude that there is no universally optimum solution for all problems and it depends on the properties of each specific problem and the available assumptions. By analyzing the bounds of privacy and utility, we provide some suggestions and principles on how to design a proper algorithm.

5.2 Future research

Privacy-preserving distributed processing is a relatively new research area, there are many exciting research directions to further explore. In what follows we will briefly mention a few of them.

Hybrid approaches for privacy-preserving distributed processing

In this thesis, we studied and proposed a number of privacy-preserving solutions for different contexts. In paper H we related and compared three different types of approaches together including secure multiparty computation, differential privacy and subspace perturbation algorithms. There we showed that these algorithms are not mutually exclusive and can be combined together. This leads to a potential direction of future research, i.e, develop hybrid approaches by adopting different principles together to achieve a superior performance.

Bounds of privacy-preserving solutions when considering graph topology

In Paper H we derived the lower bound of privacy and explained how to achieve it. There we found that in the case of distributed average consensus, such lower bound can be achieved if and only if the passive adversary does not disconnect the honest nodes, i.e., after removing all corrupted nodes the honest nodes can still form a connected graph. This assumption can only be guaranteed if the graph is fully-connected. However, in practical applications, a fully-connected graph might not be available as it scales up poorly with the number of nodes. Therefore, a new definition of the lower bound of privacy which considers the graph topology is of sufficient interest to be explored.

Quantization effects in SMPC and differential privacy approaches

In paper F we explore the quantization effects of subspace perturbation based distributed optimization approaches to achieve both privacy-preservation and a lightweight communication cost. It would be very interesting to see how to extend similar idea to other privacy-preserving approaches such as SMPC and differential privacy approaches. In other words, to verify whether the adaptive quantization idea is general to resolve the trade-off between privacy and communication cost for all kinds of privacy-preserving distributed processing algorithms.

Explore the potential of graph signal processing for privacy-preservation

From the work presented in Paper G, we can see that there is very little investigation in privacy-preserving graph signal processing. As graph signal processing has proven to be very advantageous in solving large-scale data learning problems by exploiting the inherent structure of the underlying data, it is certainly a promising future research direction to optimize graph signal processing with privacy constraints.

References

[1] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proc. ACM CCS, pp. 603–618*, 2017.

[2] T. Li, A. K. Sahu, A. Talwalkar and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Magazine, vol. 37, no. 3, pp. 50-60,*, 2020.

[3] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *in Proc. IEEE Symp. Secur. Privacy (SP), pp.3–18*, 2017.

[4] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and

federated learning," in *2019 IEEE symposium on security and privacy (SP), pp. 739-753*, 2019.

[5] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *2019 IEEE Symposium on Security and Privacy (SP), pp. 691–706.*, 2019.

[6] S. Yagli, A. Dytso and H.V. Poor, "Information-theoretic bounds on the generalization error and privacy leakage in federated learning," in *IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2020, pp. 1–5.

[7] M. Anderson, *Technology device ownership, 2015*. Pew Research Center, 2015.

[8] J. Poushter and others, "Smartphone ownership and internet usage continues to climb in emerging economies," *Pew Research Center*, vol. 22, pp. 1–44, 2016.

[9] M. Goddard, "The eu general data protection regulation (gdpr): European regulation that has a global impact," *International Journal of Market Research*, vol. 59, no. 6, pp. 703–705, 2017.

[10] L. Xiao, S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Control Lett., vol. 53, no. 1, pp. 65-78*, 2004.

[11] A. Alabdulatif, I. Khalil, A. Y. Zomaya, Z. Tari, and X. Yi,, "Fully homomorphic based privacy-preserving distributed expectation maximization on cloud," *IEEE Trans. Parallel and Distributed Syst.*, vol. 31, no. 11, pp. 2668–2681, 2020.

[12] S. Aliaksei and K. Soummya and M. José MF, "Finite-time distributed consensus through graph filters," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 1080–1084*, 2014.

[13] S. Santiago, M. Antonio G and R. Alejandro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process., vol. 65, no. 15, pp. 4117-4131*, 2017.

[14] C.M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[15] J.Pearl, *Reverend Bayes on inference engines: A distributed hierarchical approach*. Proc. 1982 Am. Assoc. Artificial Intell., pp. 133-136, 1982.

[16] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory, vol. 52, no. 6, pp. 2508–2530*, 2006.

[17] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE, vol. 98, no. 11, pp. 1847–1864*, 2010.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning, vol. 3, no. 1, pp. 1–122*, 2011.

[19] G. Zhang and R. Heusdens, "Bi-alternating direction method of multipliers over graphs," in *Proc. Int. Conf. Acoustics, Speech, Signal Proc.* Brisbane (Australia): IEEE, 2015.

[20] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process., vol. 4, no. 1, pp. 173–187*, 2018.

[21] T. Sherson, R. Heusdens, W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 2, pp 334-347*, 2018.

[22] A. H. Poorjam, Y. P. Raykov, R. Badawy, J. R. Jensen, M. G. Christensen and M.A. Little, "Quality control of voice recordings in remote Parkinson's disease monitoring using the infinite hidden markov model," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 805-809*, 2019.

[23] A. H. Poorjam, M. S. Kavalekalam, L. Shi, Y. P. Raykov, J. R. Jensen, M. A. Little and M. G. Christensen, "Automatic quality control and enhancement for voice-based remote Parkinson's disease detection," *arXiv preprint arXiv:1905.11785*, 2019.

[24] G. Giaconi, D. Gündüz, H. V. Poor, "Privacy-aware smart metering: Progress and challenges," *IEEE Signal Process. Mag., vol. 35, no. 6, pp. 59-78*, 2018.

[25] G. Avoine, L. Calderoni, J. Delvaux, D. Maio, and P. Palmieri, "Passengers information in public transport and privacy: Can anonymous tickets prevent tracking?" *Int. J. Inf. Manage, vol. 34, pp. 682-688*, 2014.

[26] D. Dolev, C. Dwork, O. Waarts, M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach., vol. 40, no. 1, pp. 17-47,*, 1993.

[27] D. Bogdanov, S. Laur, J. Willemson, "Sharemind: A framework for fast privacy-preserving computations," in *Proc. 13th Eur. Symp. Res. Comput. Security: Comput. Security, pp. 192-206,*, 2008.

[28] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Process. Magazine, vol. 30, no. 1, pp. 82–105*, 2013.

[29] T. M. Cover and J. A. Tomas, *Elements of information theory*. John Wiley & Sons, 2012.

[30] R. Cramer, I. B. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

[31] J. Vaidya and C. Clifton, "Privacy-preserving k-means clustering over vertically partitioned data," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 206–215.

[32] S. Jha, L. Kruger, and P. McDaniel, "Privacy preserving clustering," in *European Symposium on Research in Computer Security*. Springer, 2005, pp. 397–417.

[33] G. Jagannathan and R. N. Wright, "Privacy-preserving distributed k-means clustering over arbitrarily partitioned data," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 593–599.

[34] S. Samet, A. Miri, and L. Orozco-Barbosa, "Privacy preserving k-means clustering in multi-party environment." in *SECRYPT*, 2007, pp. 381–385.

[35] C. Su, F. Bao, J. Zhou, T. Takagi, and K. Sakurai, "Privacy-preserving two-party k-means clustering via secure approximation," in *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*, vol. 1. IEEE, 2007, pp. 385–391.

[36] P. Bunn and R. Ostrovsky, "Secure two-party k-means clustering," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 486–497.

[37] M. C. Doganay, T. B. Pedersen, Y. Saygin, E. Savaş, and A. Levi, "Distributed privacy preserving k-means clustering with additive secret sharing," in *Proceedings of the 2008 international workshop on Privacy and anonymity in information society*. ACM, 2008, pp. 3–11.

[38] K. Xing, C. Hu, J. Yu, X. Cheng, and F. Zhang, "Mutual privacy preserving k-means clustering in social participatory sensing," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2066–2076, 2017.

[39] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *STOC, pp.169–178*, 2009.

[40] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT, pp. 223–238*, 1999.

[41] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO, pp. 643–662*. Springer, 2012.

[42] R. Lazzeretti, S. Horn, P. Braca, and P. Willett, "Secure multi-party consensus gossip algorithms," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 7406–7410*, 2014.

[43] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy preserving distributed beamforming based on homomorphic encryption," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2013.

[44] ——, "Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 7005–7009*, 2013.

[45] Z. Xu and Q. Zhu, "Secure and resilient control design for cloud enabled networked control systems," in *Proc. 1st ACM Workshop Cyber-Phys. Syst.-Secur. Privacy., pp. 31–42*, 2015.

[46] N. M. Freris and P. Patrinos, "Distributed computing over encrypted data," in *Proc. IEEE 54th Annu. Allerton Conf. Commun., Control, Comput. pp. 1116–1122*, 2016.

[47] Y. Shoukry et al., "Privacy-aware quadratic optimization using partially homomorphic encryption," in *Proc. IEEE 55th Conf. Decis. Control., pp. 5053–5058*, 2016.

[48] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *INFOCOM*, 2011, pp. 820–828.

[49] C. Zhang, M. Ahmad, and Y. Wang, "ADMM based privacy-preserving decentralized optimization," *IEEE Trans. Inf. Forensics Security, vol. 14, no. 3, pp. 565–580*, 2019.

[50] Y. Aono, T. Hayashi, L. T. Phong, and L. Wang, "Privacy-preserving logistic regression with distributed data sources via homomorphic encryption," *IEICE Trans. Inf. Syst., vol. 99-D, no. 8, pp. 2079–2089*, 2016.

[51] Y. Aono, T. Hayashi, L. Wang, S. Moriai et al., "Privacy-preserving deep learning via additively homomorphic encryption," in *IEEE Trans. Inf. Forensics Security. vol. 13, no. 5, pp. 1333–1345*, 2018.

[52] B. Yang, I. Sato and H. Nakagawa,, "Privacy-preserving em algorithm for clustering on social network," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2012, pp. 542–553.

[53] S. X. Lee, K. L. Leemaqz and G. J. McLachlan, "PPEM: Privacy-preserving em learning for mixture models," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 24, p. e5208, 2019.

[54] A. C. Yao, "Protocols for secure computations," in *FOCS, pp. 160–164*, 1982.

[55] ——, "How to generate and exchange secrets," in *FOCS, pp. 162–167*, 1986.

[56] F. Hanzely, J. Konečnỳ, N. Loizou, P. Richtárik, and D. Grishchenko, "Privacy preserving randomized gossip algorithms," *arXiv preprint arXiv:1706.07636*, 2017.

[57] Q. Li, I. Cascudo, and M. G. Christensen, "Privacy-preserving distributed average consensus based on additive secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

[58] Q. Li and M. G. Christensen, "A privacy-preserving asynchronous averaging algorithm based on shamir's secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

[59] N. Gupta, J. Katz, N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine, vol. 50, no. 1, pp. 9515-9520*, 2017.

[60] N. Gupta, J. Kat and N. Chopra, "Statistical privacy in distributed average consensus on bounded real inputs," in *ACC, pp 1836-1841*, 2019.

[61] K. Tjell and R. Wisniewski, "Privacy preservation in distributed optimization via dual decomposition and ADMM," in *Proc. IEEE 58th Conf. Decis. Control., pp. 7203–7208*, 2020.

[62] V. Chen, V. Pastro and M. Raykova, "Secure computation for machine learning with spdz," *arXiv preprint arXiv:1901.00329*, 2019.

[63] C. Liu, "Introduction to combinatorial mathematics," 1968.

[64] T. Araki, J. Furukawa, Y. Lindell, A. Nof and K. Ohara, "High-throughput semi-honest secure three-party computation with an honest majority," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur., pp. 805-817*, 2016.

[65] P. Mohassel and P. Rindal, "Aby3: A mixed protocol framework for machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur., pp. 35-52*, 2018.

[66] C. Dwork, "Differential privacy," in *ICALP, pp. 1–12*, 2006.

[67] C. Dwork, F. McSherry, K. Nissim, A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory of Cryptography Conf. , pp. 265-284*, 2006.

[68] C. Dwork and G.N. Rothblum, "Concentrated differential privacy," *arXiv preprint arXiv:1603.01887*, 2016.

[69] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Advances in Cryptology—EUROCRYPT, pp. 486–503*, 2006.

[70] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proc. 41st Annu. ACM Symp. Theory Comput., pp. 371-380*, 2009.

[71] M. Kefayati, M. S. Talebi, B. H. Khalajand H. R. Rabiee , "Secure consensus averaging in sensor networks using random offsets," in *Proc. of the IEEE Int. Conf. on Telec., and Malaysia Int. Conf. on Commun., pp. 556–560*, 2007.

[72] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," in *ACM workshop Privacy electron. Soc., pp. 81–90*, 2012.

[73] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *ECC, pp. 760–765*, 2013.

[74] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Automat Contr., vol. 62, no. 2, pp. 753–765*, 2017.

[75] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica, vol. 81, pp. 221–231*, 2017.

[76] Z. Huang, S. Mitra, and N. Vaidya, "Differentially private distributed optimization., pp. 1–10," in *Proc. Int. Conf. Distrib. Comput. Netw*, 2015.

[77] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Trans. Autom. Control., vol. 62, no. 1, pp 50-64*, 2016.

[78] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private distributed convex optimization via functional perturbation," *IEEE Trans. Control Netw. Syst., vol. 5, no. 1, pp 395-408*, 2018.

[79] T. Zhang and Q. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Trans. Inf. Forensics Security, vol. 12, no. 1, pp. 172–187*, 2016.

[80] X. Zhang, M. M. Khalili, and M. Liu, "Recycled ADMM: Improve privacy and accuracy with less computation in distributed algorithms," in *in Proc. 56th Annu. Allerton Conf. Commun., Control, Comput. pp.959–965*, 2018.

[81] ——, "Improving the privacy and accuracy of ADMM-based distributed algorithms," *Proc. Int. Conf. Mach. Lear. pp. 5796–5805*, 2018.

[82] Y. Xiong, J. Xu, K. You, J. Liu and L. Wu, "Privacy preserving distributed online optimization over unbalanced digraphs via subgradient rescaling," *IEEE Trans. Control Netw. Syst.*, 2020.

[83] M. Park, J. Foulds, K. Choudhary and M. Welling, "DP-EM: Differentially private expectation maximization," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 896–904.

[84] M. Gtz, A. Machanavajjhala, G. Wang, X. Xiao, J. Gehrke, "Publishing search logs—a comparative study of privacy guarantees," *IEEE Trans. Knowl. Data. Eng. vol. 24, pp. 520 - 532*, 2011.

[85] A. Haeberlen, B. C. Pierce, A. Narayan, "Differential privacy under fire." in *Proc. 20th USENIX Conf. Security., vol. 33*, 2011.

[86] A. Korolova, K. Kenthapadi, N. Mishra, A. Ntoulas, "Releasing search queries and clicks privately," in *Proc. Int'l Conf. World Wide Web, pp. 171–180*, 2009.

[87] P. Cuff and L. Yu, "Differential privacy as a mutual information constraint," in *Proc. 23rd ACM SIGSAC Conf. Comput. Commun. Secur., pp 43–54*, 2016.

[88] I. Mironov, "Rényi differential privacy," in *Proc. IEEE 30th Comput. Secur. Found. Symp. (CSF), pp. 263-275*, 2017.

[89] D. H. M. Schellekens, T. Sherson, and R. Heusdens, "Quantisation effects in PDMM: A first study for synchronous distributed averaging," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 4237–4241*, 2017.

[90] J. A. G. Jonkman, T. Sherson, and R. Heusdens, "Quantisation effects in distributed optimisation," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 3649–3653*, 2018.

[91] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process., vol. 61, no. 7, pp. 1644-1656,*, 2013.

[92] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Magazine, vol. 30, no. 3, pp. 83-98,*, vol. 30, no. 3, pp. 83–98, 2013.

[93] M. Coutino, E. Isufi, G. Leus, "Advances in distributed graph filtering," *IEEE Trans. Signal Process., vol. 67, no. 9, pp. 2320-2333,*, 2019.

[94] J. Pang, G. Cheung, A. Ortega, O. C. Au, "Optimal graph Laplacian regularization for natural image denoising," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp 2294-2298*, 2015.

[95] SK Narang, A Gadde, A Ortega, "Signal processing techniques for interpolation in graph structured data," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp 5445-5449*, 2013.

[96] S. Segarra, A. G. Marques, G. Leus and A. Ribeiro,, "Reconstruction of graph signals through percolation from seeding nodes," *IEEE Trans. Signal Process., vol. 64, no. 16, pp. 4363-4378,*, vol. 64, no. 16, pp. 4363–4378, 2016.

[97] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *DCOSS, pp 1–8*, 2011.

Part II

Papers

Paper A

Privacy-Preserving Distributed Average Consensus based on Additive Secret Sharing

Qiongxiu Li, Ignacio Cascudo, and Mads Græsbøll Christensen

The paper has been published in the
Proc. Eur. Signal Process. Conf. (EUSIPCO) 2019

The layout has been revised.

Abstract

One major concern of distributed computation in networks is the privacy of the individual nodes. To address this privacy issue in the context of the distributed average consensus problem, we propose a general, yet simple solution that achieves privacy using additive secret sharing, a tool from secure multiparty computation. This method enables each node to reach the consensus accurately and obtains perfect security at the same time. Unlike differential privacy based approaches, there is no trade-off between privacy and accuracy. Moreover, the proposed method is computationally simple compared to other techniques in secure multiparty computation, and it is able to achieve perfect security of any honest node as long as it has one honest neighbour under the honest-but-curious model, without any trusted third party.

1 Introduction

The consensus problem has received a lot of attention from researchers over the past decades since it has many practical uses, such as distributed data fusion [1] and group coordination [2]. To solve the average consensus problem in arbitrary random connected distributed networks (e.g., in wireless sensor networks), many distributed averaging algorithms have been proposed, such as basic average consensus algorithms [3], gossip algorithms [4, 5], ADMM [6] and PDMM [7] algorithms based on convex optimization and graph filter methods [8, 9]. These iterative approaches require to exchange information among participants to compute the average result. However, the information exchange is a cause for concerns with respect to the privacy of the data, as private information may be revealed.

To compute the average in arbitrary random connected distributed networks while preserving the privacy of the data, two categories of algorithms have been proposed. The first type of algorithms [10–14] implements average consensus by modifying the basic average consensus algorithm [3] based on the concept of differential privacy [15]. If there are two databases that differ only in one single element, it is easy to get the information of this element by comparing the query results of two databases. Differential privacy aims at protecting the privacy of this single element by introducing randomness in query results. The underlying idea is to maintain a balance between the individual privacy and output accuracy by inserting noise to obfuscate the function output in a random manner. Many algorithms [10–14] applied this idea to achieve privacy-preserving average consensus with a careful zero-sum noise insertion process. A detailed analysis of the trade-off between maximum information disclosure and estimation accuracy is performed in [16]. However, as proven in [14], exact accuracy and differential privacy cannot be obtained at the same time. Thus, [17] refers to differential privacy methods as

consensus-perturbing algorithms and proposed a new consensus-preserving algorithm that, with the help of a trusted third party, assigns to each node a single noise value the sum of which equals zero. Unfortunately, the trusted third party assumption is not practical in many real-world applications. Another type of algorithms [18–21] applies garbled circuits (GC) [22, 23] and homomorphic encryption (HE) [24, 25] techniques, as known from secure multiparty computation [26], to general gossip algorithms [4] to preserve privacy. Secure multiparty computation allows all nodes in a network to jointly compute a function and keep their inputs private. Two GC based gossip algorithms were proposed in [18] to iteratively compare the state values of two nodes and update the state values with a step-size while keeping each state value secret. However, the computational complexity is big and only asymptotic consensus is achieved. The HE technique was applied in [20, 21] to compute the consensus in the encrypted domain. The initial state value of each node is kept private because only encrypted values are accessed by other nodes. Unfortunately, the computational complexity of the HE technique is big and a trusted third party is also required.

In this paper, we propose a general, yet simple algorithm to solve the privacy-preserving distributed average consensus problem using the principle of additive secret sharing. Note that additive secret sharing has been applied in various applications such as smart grids [27] to address privacy concerns under a strong assumption of network topology (e.g., fully connected). This differs significantly from the proposed algorithm, since we here assume a more practical and general network topology (i.e., arbitrarily connected). In a decentralized network, the average consensus is usually computed by iterative distributed averaging algorithms such as [3–7]. Thus, the question of how to achieve the privacy concern during all iterations is the main challenge.

The proposed approach is lightweight compared to the above mentioned HE and GC approaches in [18–21], as only additions are involved. The underlying idea of additive secret sharing is to replace each initial state value with another obfuscated value by subtracting and adding random numbers. Unlike the differential privacy approaches [10–14], there is no trade-off between privacy and estimated accuracy. The main properties of the proposed approach can be summarized as follows: 1) the proposed approach achieves perfect security and exact accuracy at the same time; 2) it is computationally simple; 3) individual privacy is guaranteed as long as it has one honest neighbour under the honest-but-curious model without any trusted third party; 4) it is convenient since only an additive randomization step is needed; and 5) it is very general since it can be applied in any distributed averaging algorithm.

2 Preliminaries and Problem Setup

2.1 Privacy-preserving distributed average consensus problem

A distributed system composed of a set of nodes can be modelled as an undirected connected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. The node set of the graph is denoted as $\mathcal{N} = \{1, 2, ..., n\}$ and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ denotes the communication links between nodes. The communication of two nodes is enabled if there is one edge connecting two nodes, i.e., $(i, j) \in \mathcal{E}$, and $n_i = \{j | (i, j) \in \mathcal{E}, j \neq i\}$ denotes the neighbours of node i. The initial state value held by node i is denoted as a_i, and the initial state values in the network can be written as a vector $\mathbf{a} = [a_1, a_2, ..., a_n]^T$. The main goal is to address the following two challenges at the same time:

1. Compute the average result of the private values

$$a_{\text{ave}} = \frac{1}{n} \sum_{i=1}^{n} a_i, \tag{A.1}$$

   in a distributed network without having any centralized coordinator, an iterative algorithm is usually adopted.

2. Protect the private value a_i of each node throughout the algorithm execution.

2.2 Privacy concern and adversary model

An important aspect of this work is the definition of privacy. Our goal here is to protect the initial state value a_i of each node, which constitutes the private information, during the execution of the algorithm. The reason is that it may represent an individual's opinion [28] or private information [20, 21].

In this paper, a general honest-but-curious (also called "passive" or "semi-honest") model is considered. It means that all nodes in the network follow the designed protocol, but some of them might be curious about the other nodes' private information. Such curious nodes are said to be passively corrupted, and they can cooperate to share their received information with the aim of inferring other honest nodes' private information, here the initial state values. We assume the worst case situation within this model, where passively corrupted nodes know the following:

- The whole graph topology.
- The initial state values of all passively corrupted nodes.
- The transmitted information over the communication links involving the corrupted nodes.

Thus, the corrupted nodes will know all the information except the information kept by the honest nodes themselves and exchanged between every two honest nodes, as long as it cannot be deduced from the above.

3 Additive secret sharing

A secret sharing scheme is a cryptographic tool that splits a secret into a number of shares, where each node in a group will receive one share. The secret can be reconstructed only if a sufficient number of shares are collected, otherwise no information about the hidden secret will be revealed. General secret sharing schemes usually consist of two parts:

- The secret sharing algorithm takes a secret s as input and some randomness r, and outputs n shares of this secret:

$$F_{\mathrm{S}}(s, r) = (s_1, s_2, ..., s_n). \tag{A.2}$$

- The secret reconstruction function (which technically is a family of functions, one for each subset of shares that can reconstruct the secret) takes the shares of some subset of the nodes $\{\Lambda_1, \ldots, \Lambda_t\}$ as inputs to reconstruct the secret s:

$$F_{\mathrm{R}}(s_{\Lambda_1}, s_{\Lambda_2}, ..., s_{\Lambda_t}) = s, \tag{A.3}$$

where s_i denotes the i^{th} share of secret s. If we have a reconstruction function for any set of at least t shares, but no set with less than t shares provides any information about the secret, then the secret sharing scheme can be referred to as (n, t) threshold secret sharing scheme.

One of the simplest secret sharing schemes is the additive secret sharing where $t = n$. This is defined over an algebraic group F, usually given by the integers $\{0, \ldots, p-1\}$ together with the additive operation modulo p. While p is a prime number in many applications (so that the group is also a finite field), this is not required here. The additive secret sharing scheme is defined as follows: choose $n-1$ integers $r_1, ..., r_{n-1}$ in F uniformly at random. Then the output of the function (A.2) consists of $s_i = r_i$ for $i = 1, \ldots, n-1$ and $s_n = (s - \sum_{i=1}^{n-1} r_i) \bmod p$. Given the full set of n shares, the secret can be reconstructed by

$$s = (\sum_{i=1}^{n} s_i) \bmod p. \tag{A.4}$$

It is easy to see that the secret s cannot be reconstructed even if only one share is missing, and the secret is, in fact, uniformly distributed over the integers within F even though the knowledge of $n-1$ shares is given. Additive secret sharing has the following property, which enables secure computation of additions: if two secrets $s, s' \in F$ are shared among some set of nodes, then the nodes can reconstruct the sum of the secrets, without needing to reconstruct the individual secrets, as follows: each node i locally add the received shares s_i, s'_i and reveal only this sum of shares $h_i = s_i + s'_i \mod p$ to the other nodes.

Then applying the reconstruction function (A.4) to these share sums h_i will give the sum of the original secrets $s + s' = (\sum_{i=1}^{n} h_i) \bmod p$, without revealing anything else. This can be extended to summing an arbitrary number of secrets, and it can be turned into a secure computation protocol to compute the sum of the secrets of a set of n nodes in a fully connected network, where every node first sends shares of its secret among the full set of nodes, and the process described above is used to reconstruct only the sum. This is secure against an arbitrary number of passive corruptions (see [26, Section 1.3.1] for further details).

4 Proposed algorithm

In this section, the details of the proposed algorithm will be described. The algorithm itself is shown in Algorithm 0, where d_i denotes the total number of elements in n_i and T denotes the maximum iteration number, F is the set of integers modulo p for a large enough number p ($p > \sum_{i=1}^{n} a_i$), c denotes the penalty parameter in PDMM algorithm [7].

The first stage of the algorithm is additive randomization, where each node uses additive secret sharing for distributing shares of its private value a_i to its neighbours. We remark that the difference between this use of additive secret sharing and the one described at the end of the previous section is the assumption of the graph topology. The scheme described in the previous section assumes a fully connected network where each node sends shares to all other nodes. However, a fully connected graph scales poorly in the number of connections. In this paper, we assume an arbitrarily connected graph, which is much more practical and scalable in real-life applications. Each node only sends shares to its neighbours and an iterative distributed averaging algorithm is used afterwards. The main goal of additive randomization is to address the privacy challenge in Section 2.1 by replacing the private value a_i of each node with an obfuscated value u_i, which can then be revealed. In Section 5.3 we show exactly how much information this provides to the corrupted nodes. An important observation is that by construction we have that $\sum_{i=1}^{n} a_i = (\sum_{i=1}^{n} u_i) \mod p$.

After additive randomization, we take the obfuscated values u_i as inputs to a distributed averaging algorithm [3–7] to compute the average, which meets the requirements described in Section 2.1. Here we apply the asynchronous PDMM algorithm as an example. After convergence, the primal variable x_i^T for all nodes $i \in \mathcal{N}$ will reach the average of obfuscated values, i.e., $x_i^T = \frac{1}{n} \sum_{i=1}^{n} u_i$.

The last part of the proposed algorithm is to compute the final average result by (A.9) with an assumption of knowing the total number of nodes n.

Algorithm 1 Proposed algorithm

Additive randomization:

1: Each node $i \in \mathcal{N}$ extract d_i random numbers as shares r_i^k with uniform probability in F.

2: Node i sends shares r_i^k to its neighbours $k \in n_i$ and keep the share r_i as.

$$r_i = (a_i - \sum_{k \in n_i} r_i^k) \bmod p. \tag{A.5}$$

3: Node i receives shares r_k^i from its neighbours $k \in n_i$.

4: Node i updates a_i as the obfuscated value

$$u_i = (r_i + \sum_{k \in n_i} r_k^i) \bmod p. \tag{A.6}$$

Distributed averaging (e.g., PDMM):

5: Each node initializes the primal variable x_i^0 and dual variable $\xi_{i|j}^0$ as zeros, $i, j \in \mathcal{N}$.

6: For iteration $t = 1, 2, 3, ..., T$

7: Activate node $i \in \mathcal{N}$ randomly with uniform probability.

8: Node i updates x_i^t and broadcasts to its neighbours

$$x_i^t = \frac{u_i + \sum_{k \in n_i} (c x_k^{t-1} + \xi_{k|i}^{t-1})}{1 + c d_i}. \tag{A.7}$$

9: After receiving x_i^t updates, all neighbouring nodes $k \in n_i$ update the dual variable as

$$\xi_{i|k}^t = -\xi_{k|i}^{t-1} + c(x_i^t - x_k^{t-1}). \tag{A.8}$$

10: Repeat until the primal variable x_i^t converges

Average consensus computation

11: Each node obtains the average as

$$x_{\text{ave}} = \frac{1}{n}(x_i^T \times n \mod p). \tag{A.9}$$

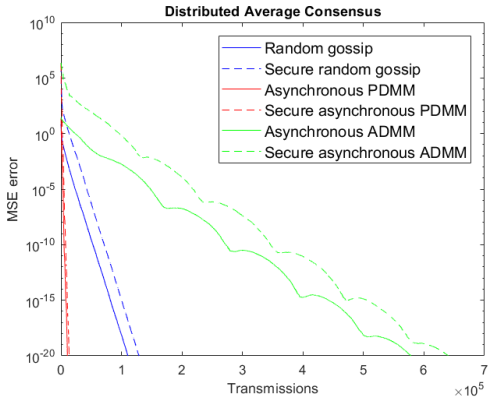


Fig. A.1: Experimental results

The average result x_{ave} of the proposed algorithm is identical to a_{ave} since

$$a_{\text{ave}} = \frac{1}{n}\sum_{i=1}^{n} a_i = \frac{1}{n}((\sum_{i=1}^{n} u_i) \bmod p) = x_{\text{ave}}. \tag{A.10}$$

Concerning data representation, we remark that the values in the additive randomization process should be integers within the modular domain $\{0, ...p-1\}$ due to the additive secret sharing. Floating point numbers can be scaled up as integers and negative numbers can be represented using modular additive inverse. Note that integers are not required afterwards because additive secret sharing scheme is no longer applied in the distributed averaging step, which is also why the division operation can be used in (A.7).

5 Experimental results and analysis

5.1 Experimental results

Simulations are conducted here to investigate the performance of the proposed approach. A random geometric graph [29] with $n = 100$ nodes is simulated and the connectivity of nodes is enabled if their distance is within a radius $\sqrt{\frac{\log n}{n}}$ to have a connected graph with high probability [29]. Based on the same initial state values over the network and additive randomization procedure, the simulation results are demonstrated in Fig. A.1, where the solid blue, green, red lines denote the conventional non-privacy concerned random gossip [4], asynchronous ADMM [6] and PDMM [7] algorithms, respectively, and the related dashed lines represent the proposed secure approaches which add additive randomization before the above mentioned conventional algorithms, and the penalty parameters in both ADMM and PDMM are set as 0.4.

As demonstrated in Fig. A.1, we can see that the estimated accuracy of all the proposed secure approaches is identical to conventional non-secure approaches. The convergence rate of the proposed approaches will be slightly slower than the traditional approaches as the initial mean square error becomes higher after additive randomization. We remark that the extra additive randomization will not affect the convergence speed but only cause higher initial errors.

5.2 Comparisons

A comparison of the proposed approach with existing methods is shown in Table A.1, where β denotes the number of bits needed to represent transmitted message [30]. The same passive adversary model is considered in all approaches. We can see that HE and GC based approaches both require expensive computational function as encryption is involved, and the communication bandwidth is also big since the cipher text after encryption usually require much longer bit lengths than plain text, and a trusted third party is also required in HE. Moreover, the proposed approach is able to achieve identical accuracy and perfect security with same communication bandwidth and computational function as differential privacy approaches. Note that the maximum number of corruptions for algorithms without an accuracy trade-off is $n-2$, because the corrupted nodes can always know the initial state value of the only honest node given the knowledge of the exact consensus result and the initial state values of the corrupted $n-1$ nodes. For differential privacy and GC based approaches, the maximum number of corruptions can be $n-1$, as the average result is inexact. Furthermore, the proposed algorithm can protect the privacy of any honest node only if it has at least one honest neighbour, which is not required in the other approaches, e.g., in differential privacy approaches.

Table A.1: Privacy-preserving distributed average consensus approaches under arbitrary connected graphs

	Proposed	HE [20, 21]	GC [18]	Differential privacy [10–14]
Accuracy	Identical	Identical	Dependent on step size	Degraded with noise
Security	Perfect	Computational	Computational	Differential privacy
Involved function	Linear	Exponential	Exponential	Linear
Trusted Third Party	No	Yes	No	No
Adversary model	Passive	Passive	Passive	Passive
Communication bandwidth per round	$\mathcal{O}(1)$	$\mathcal{O}(\beta)$	$\mathcal{O}(\beta)$	$\mathcal{O}(1)$
Maximum number of corruptions	n-2	n-2	n-1	n-1

5.3 Security guarantee

In this section, we analyze the security of the proposed algorithm in more detail. The statement we will argue is as follows: Let $\mathcal{C} \subseteq \mathcal{N}$ be the subset of passively corrupted nodes, and let $\mathcal{H} = \mathcal{N} \setminus \mathcal{C}$ be the set of honest nodes. If the subgraph $\mathcal{H}$ is connected, then the only information about the honest nodes' initial state values can be learned by the corrupted nodes is $\sum_{k\in\mathcal{H}} a_k$,

but nothing more than that. And we remark that learning this information is logically unavoidable if we have exact accuracy, since this information can always be deduced from the average result and the initial state values of the corrupted nodes:

$$\sum_{k\in\mathcal{H}} a_k = n \times a_{\text{ave}} - \sum_{k\in\mathcal{C}} a_k.$$

This implies the following: The individual privacy of the honest nodes is protected as long as it has some honest neighbours, even in the case where there are only two honest nodes. If the two honest nodes i, j are neighbours, the corrupted nodes do not learn the individual private value a_i and a_j, but only their sum.

The proof is as follows: adopting a pessimistic view, the information set obtained by $\mathcal{C}$, also known as the information view, after reaching consensus is in the worst case the union $V = V_1 \cup V_2 \cup V_3$ of the information sets

$$\begin{cases} V_1 = \{u_k | k \in \mathcal{N}\}, \\ V_2 = \{a_{\text{ave}}, a_k | k \in \mathcal{C}\}, \\ V_3 = \{r_k^m, r_m^k | k \in \mathcal{C}, m \in d_k\} \cup \{r_k, k \in \mathcal{C}\}. \end{cases}$$

Note that all the obfuscated values u_i in the distributed averaging step are included in V_1 as the primal and dual variables are initialized as zeros in (A.7), these values u_i can therefore be considered non-private.

Now suppose a "real" instance

$$I = \{a_k, k \in \mathcal{N}\},$$

has produced the above view V with real initial state values and randomness $r_\ell, \ell \in \mathcal{N}$ and $r_\ell^m, (\ell, m) \in \mathcal{E}$.

Let i,j be two honest nodes which are neighbours of each other. We now produce a "fake" instance

$$I' = \{a'_k, k \in \mathcal{N}\},$$

having the view V' with all $a'_k = a_k, k \in \mathcal{N}, k \neq i, j$ and $a'_j = a_j - d$, $a'_i = a_i + d$ for some d. Note that $\sum_{k\in\mathcal{H}} a_k = \sum_{k\in\mathcal{H}} a'_k$ by setting the randomness as $r'_i = r_i + d$, $r_j^{i\prime} = r_j^i - d$, and leave all other random values r_ℓ, r_ℓ^m unchanged.

Thus, the information view V' produced by the fake instance $I^{'}$ will be exactly the same with V produced by the real instance I, which means that the corrupted nodes cannot distinguish the "real" from the "fake". Since $\mathcal{H}$ is connected, we can repeat the argument to modify the initial state values of $\mathcal{H}$ in any way that we want, as long as this modification does not change the sum of the honest initial state values, and still produce the same view for corrupted nodes. The corrupted nodes can only learn the sum of honest nodes' initial state values $\sum_{k\in\mathcal{H}} a_k$, but no other information. Hence, the proposed algorithm is perfectly secure in the sense of secure computation, as it protects all information that is not implied by the average result and corrupted nodes' initial state values.

We remark that if the subgraph of the honest nodes is not connected, then the corrupted nodes can infer the partial sums of the initial state values held by the connected subsets in $\mathcal{H}$, but nothing else beyond that. In an extreme case, if a honest node has only one honest neighbour, then the leaked information is only the sum of the initial state values held by these two honest nodes, we emphasize that the privacy of the *individual* node is always protected, which is our goal here. Hence the privacy of individual node is guaranteed as long as it has one honest neighbour.

6 Conclusions and future work

In this paper, we have proposed a general and simple solution to address the privacy concern in distributed average consensus problems with the help of the additive secret sharing scheme. An additive randomization step is applied before distributed averaging to replace the initial state value of each node with a non-private obfuscated value for privacy-preserving. The proposed solution outperforms differential privacy based approaches, as it obtains perfect security and accurate consensus at the same time. Moreover, it is computationally less complex compared to HE and GC based approaches. The proposed algorithm is general and can be used with arbitrary distributed averaging algorithms. Moreover, it does not require any trusted third party, and the privacy of each individual honest node is protected as long as it has one honest neighbour. Future work will focus on how to maintain privacy under more challenging adversary models (i.e., active attacks) where the corrupted nodes may not follow the protocol correctly but deviate from it to interfere with the computation result.

References

[1] L. Xiao, S. Boyd, S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," *IPSN*, pp. 63–70, 2005.

[2] J. N. Tsitsiklis, "Problems in decentralized decision making and computation.," Tech. Rep., Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, 1984.

[3] L. Xiao, S. Boyd, "Faster linear iterations for distributed averaging," *Syst. Control Lett.*, vol. 53, no. 1, pp. 65-78, 2004.

[4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[5] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, 2010.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[7] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process.*, vol. 4, no. 1, pp. 173–187, 2018.

[8] S. Aliaksei and K. Soummya and M. José MF, "Finite-time distributed consensus through graph filters," *ICASSP*, pp. 1080–1084, 2014.

[9] S. Santiago and M. Antonio G and R. Alejandro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, 2017.

[10] M. Kefayati, M. S. Talebi, B. H. Khalaj, and H. R. Rabiee, "Secure consensus averaging in sensor networks using random offsets," *Proc. of the IEEE Int. Conf. on Telec., and Malaysia Int. Conf. on Commun.*, pp. 556–560, 2007.

[11] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," *ACM workshop Privacy electron. Soc.*, pp. 81–90, 2012.

[12] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," *ECC*, pp. 760–765, 2013.

[13] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Automat Contr.*, vol. 62, no. 2, pp. 753–765, 2017.

[14] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica*, vol. 81, pp. 221–231, 2017.

[15] C. Dwork, "Differential privacy," *ICALP*, pp. 1–12, 2006.

[16] J. He, L. Cai, C. Zhao, P. Cheng, and X. Guan, "Privacy-preserving average consensus: privacy analysis and optimal algorithm design," *IEEE Trans. Signal Process.*, 2018.

[17] P. Braca, R. Lazzeretti, S. Marano, and V. Matta, "Learning with privacy in consensus + obfuscation," *IEEE signal process. Lett.*, vol. 23, no. 9, pp. 1174–1178, 2016.

[18] F. Hanzely, J. Konečnỳ, N. Loizou, P. Richtárik, and D. Grishchenko, "Privacy preserving randomized gossip algorithms," *arXiv preprint arXiv:1706.07636*, 2017.

[19] R. Lazzeretti, S. Horn, P. Braca, and P. Willett, "Secure multi-party consensus gossip algorithms," *ICASSP*, pp. 7406–7410, 2014.

[20] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy preserving distributed beamforming based on homomorphic encryption," *EUSIPCO*, pp. 1–5, 2013.

[21] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy-preserving distributed speech enhancement forwireless sensor networks by processing in the encrypted domain," *ICASSP*, pp. 7005–7009, 2013.

[22] A. C. Yao, "Protocols for secure computations," *FOCS*, pp. 160–164, 1982.

[23] A. C. Yao, "How to generate and exchange secrets," *FOCS*, pp. 162–167, 1986.

[24] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," *EUROCRYPT*, pp. 223–238, 1999.

[25] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," *Advances in Cryptology–CRYPTO*, pp. 643–662, 2012.

[26] R. Cramer, I. B. Damgrd, and J. B. Nielsen, *Secure multiparty computation and secret sharing*, Cambridge University Press, 2015.

[27] K. Kursawe, G. Danezis, and M. Kohlweiss, "Privacy-friendly aggregation for the smart-grid," in *Privacy Enhancing Technologies*. pp. 175–191, 2011.

[28] M. H. DeGroot, "Reaching a consensus," *J. Am. Statist. Assoc.*, vol. 69, no. 345, pp. 118–121, 1974.

[29] J. Dall and M. Christensen, "Random geometric graphs," *Physical review E*, vol. 66, no. 1, pp. 016121, 2002.

[30] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Process. Magazine*, vol. 30, no. 1, pp. 82–105, 2013.

Paper B

A Privacy-Preserving Asynchronous Averaging Algorithm based on Shamir's Secret Sharing

Qiongxiu Li, Mads Græsbøll Christensen

The paper has been published in the
Proc. Eur. Signal Process. Conf. (EUSIPCO) 2019

The layout has been revised.

Abstract

Average consensus is widely used in information fusion, and it requires information exchange between a set of nodes to achieve an agreement. Unfortunately, the information exchange may disclose the individual's private information, and this raises serious concerns for individual privacy in some applications. Hence, a privacy-preserving asynchronous averaging algorithm is proposed in this paper to maintain the privacy of each individual using Shamir's secret sharing scheme, as known from secure multiparty computation. The proposed algorithm is based on a lightweight cryptographic technique. It gives identical accuracy solution as the non-privacy concerned algorithm and achieves perfect security in clique-based networks without the use of a trusted third party. In each iteration of the algorithm, each individual's privacy in the selected clique is protected under a passive attack where the adversary controls some of the nodes. Finally, it also achieves robustness of up to one third transmission error.

1 Introduction

Consensus has been intensively investigated over the past decades since it is useful to solve problems in information fusion, especially in distributed systems. Distributed average consensus has been adopted in various applications such as group coordination [1] and dynamic load balancing [2]. There are many approaches to iteratively achieve consensus without centralized coordination: average consensus algorithms [3], general-purpose gossip algorithms [4, 5], methods based on convex optimization such as the ADMM [6] and the PDMM [7] algorithms and graph filter methods [8, 9]. All the algorithms above require information exchange between certain entities. However, this exchange may disclose the individual's privacy. In a distributed network, such as a sensor network, the nodes of the network are interested in reaching an agreement but they may also have concerns about protecting the privacy of their data. For example, a group of individuals may want to achieve a common opinion using a consensus algorithm; at the same time, each individual is unwilling to trust the others by revealing his/her own opinion [10]. This makes privacy-preserving in consensus problem a crucial topic to address.

Two types of methods have been deployed to obtain privacy-preserving solutions in distributed average consensus: differential privacy [11] approaches, which try to maintain the maximum accuracy from statistical database queries while minimizing the chances of identifying its records; and secure multiparty computation [12] approaches, which aim at jointly computing a function over the inputs of a set of nodes while keeping their inputs private. The underlying idea in most existing differential privacy algorithms [13–17] is to mask the secret values with zero-sum random noise during the information exchange. This protects privacy without any trusted third party while the average consensus is

still achieved by carefully design the noise insertion process. A statistical analysis of maximum disclosure probability and estimation accuracy is performed in [18]. However, Nozari et al. [15] proved that exact average consensus and differential privacy cannot be achieved simultaneously. Differential privacy based algorithms are thus referred to as consensus perturbing approaches [19]. A new consensus preserving approach was proposed in [19] that guarantees an exact average by employing obfuscation via a single noise sample for each node while ensuring that the allocated noise sum to zero. However, the obfuscation noise samples have to be generated by a trusted third party, something that is not always practical.

Other algorithms [20–23] obtain secure average consensus based on techniques from secure multiparty computation, such as homomorphic encryption (HE) schemes [24, 25] and the garbled circuit (GC) technique [26, 27]. Homomorphic encryption enables computation on the encrypted data. HE was adopted in [22, 23] to guarantee that each node can only access the encrypted values of other nodes. However, HE requires a high computational complexity for encryption and a trusted third party. Two GC based algorithms were proposed in [20] to securely compare the state value of two nodes. However, these are also computational expensive and requires global information beforehand, and only asymptotic consensus is obtained.

In this paper, Shamir's secret sharing scheme, as known from secure multiparty computation, is adopted in a distributed asynchronous averaging algorithm reminiscent of [3] to solve the problem of privacy-preserving distributed average consensus. The main idea is to divide a secret into a number of shares and distribute a share to each node in the network. The secret can be reconstructed if and only if a sufficient amount of shares are collected, otherwise no information of secret will be disclosed. Compared to differential privacy based approaches [13–17], the proposed algorithm is able to achieve perfect security and exact accuracy at the same time. Since only computations on polynomials are involved in Shamir's scheme, it has lower computational complexity compared to encryption approaches such as the HE and GC of [20–23] and no trusted third party is required. Moreover, the proposed method considers both a general passive attack model and a weak active attack model.

2 Preliminaries and Problem Setup

2.1 Privacy-preserving distributed average consensus problem

In a distributed system, we assume an undirect connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ composed by the node set $\mathcal{V} = \{1, 2, ..., n\}$, where $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the set of undirected edges. Every two nodes can communicate with each other if and only if they are connected neighbours, i.e., $(i, j) \in \mathcal{E}$. The neighbourhood of node i is denoted as $d_i = \{j | (i, j) \in \mathcal{E}, j \neq i\}$. Each node i holds an initial state

value $a_i(0)$, which is its private information, and the vector of the initial state values on the network is denoted as $\mathbf{a}(0) = [a_1(0), a_2(0), ..., a_n(0)]^T$. The main goal is to solve the following two challenges at the same time:

1. Compute the average result of the private information

$$a_{ave} = \frac{1}{n} \sum_{i=1}^{n} a_i(0) \tag{B.1}$$

   in a distributed network without having any centralized coordinator, here using an iterative average consensus algorithm.

2. The private information of each node, $a_i(0)$, in the network should be protected during the iterations of the algorithm, hence preserving the privacy of the node.

2.2 Privacy concern and adversary model

The privacy concern addressed in this paper here pertains to the initial state value held by each node in the network, as it may be sensitive and undesirable for each node if this is revealed to others. The adversary models adopted here include a passive and a very weak kind of active attack model that is interesting from a practical point of view. In the passive attack model, also named honest-but-curious model, each node follows the protocol correctly but so-called passively corrupted nodes try to infer the honest nodes' privacy. A number of passively corrupted nodes may cooperate to increase the chance of inferring the other's initial state value by sharing information. In contrast, in an active attack model the corrupted nodes may not follow the defined protocol and attempt to manipulate the computation result by lying about the exchanged information or refuse to act according to the protocol. In that case, we would need to deal with these active attacks, and this can be done in a number of ways: one possibility is to settle for a solution that detects the errors and aborts the process in time; but a more ambitious possibility would be to find a protocol that does not only detect errors but is also able to correct the errors automatically without aborting, a property usually referred to as robustness. In what follows, we consider a weaker model that is sufficient to achieve robustness towards transmission errors.

3 Shamir's Secret Sharing scheme

In this section, a technique from secure multiparty computation called Shamir's secret sharing is introduced. Before exploring the algorithm in detail, we illustrate the concept of secret sharing with the following example [28]: several scientists are working on a secret project where some confidential documents are in a cabinet locked up with a pass-code. The cabinet can be unlocked if

(and only if) half or more of the scientists are present. Shamir's secret sharing scheme [29], first proposed by Shamir in 1979, provides a powerful solution to this problem. The principle of Shamir's secret sharing is Lagrange polynomial interpolation. It is based on the fact that a prior unknown polynomial with degree at most t can be reconstructed if its value at $t+1$ or more points are given, but any number strictly smaller than $t+1$ will give no information about the polynomial in other points.

Shamir's secret sharing is defined as follows. Assume there are n nodes, referred to as $p_1, p_2, \ldots, p_n$. The indices of these nodes are denoted as $\mathcal{N} = \{1, 2, ..., n\}$. Take a finite field F of cardinality more than n, for example we take the field of integers modulo a prime number p with $p > n$. In addition we select an integer $t < n$. In order to share a secret $s \in F$, the dealer (the node who knows the secret) proceeds as follows:

1. **Polynomial construction**: Selects coefficients $\{c_i | i = 1, 2, \ldots, t\}$ uniformly at random in F and constructs the polynomial $f(x) = s + c_1 x + c_2 x^2 + \cdots + c_t x^t \mod p$. Note that the secret is $f(0)$.

2. **Share distribution**: Compute and distribute the secret shares s_i related to p_i as $s_i = f(i) \mod p, i \in \mathcal{N}$ Note that since the dealer is also one of the p_i's, this also includes sending a share to itself. This share will be needed when aggregating the information with the shared secrets from other nodes later on.

3. **Secret reconstruction**: If a set of $t+1$ nodes, indexed by $\Lambda \subseteq \mathcal{N}$ agree to reconstruct the secret, they can use Lagrange interpolation

$$s = \sum_{i \in \Lambda} r_i s_i. \tag{B.2}$$

   where r_i is the Lagrange basis computed by

$$r_i = \prod_{j \in \Lambda \setminus \{i\}} \frac{-j}{i-j}. \tag{B.3}$$

This Shamir's secret sharing scheme divides the secret s into several shares $s_i, i \in \mathcal{N}$ and distributes them to n different nodes, and all shares and the secret are evaluations of a polynomial of degree t. The privacy guarantee of Shamir's scheme is based on interpolation properties, implying that a set of t or less shares gives no more information about the secret than what was known a priori. Moreover, Shamir's scheme is also linear: it allows to "add secrets". If two secrets s and s' are shared, possibly by different dealers, among the same network of users by using polynomials f and f', then the nodes can obtain a sharing of $s + s'$ by simply adding their two shares. This works well because $f + f'$ is still of degree $\leq t$ and $f(i) + f'(i) = (f + f')(i) \mod p$.

Finally Shamir's scheme has certain error correction properties that can be used to detect errors, and in some cases, correct them. This allows to correct

certain types of active malicious behaviour. More precisely it is a robust secret sharing scheme: if $t < n/3$, then given the set of all n shares, if at most t are erroneous then the Berlekamp-Welch algorithm [30] can output the correct secret. This prevents a set of $t < n/3$ nodes to cheat when reconstructing the secret. Following the above description, the correct polynomial constructed by secret holder is $f(x)$, and the received share set is denoted by $\{(i, s_i), i \in \mathcal{N}\}$. Since there might be some inconsistent shares in the share set, instead of directly constructing $f(x)$ based on Lagrange interpolation, we set to find two other polynomials $e(x)$ with degree t and $q(x)$ with degree $2t$ satisfying the following equality

$$q(x) = e(x)f(x),$$

and where in addition $e(x)$ (referred to as error locator polynomial) satisfies that $e(i) = 0$ whenever $f(i) \neq s_i$. Under the two conditions above, $e(x)$ and $q(x)$ satisfy the following system of linear equations, in which the unknowns are the coefficients of e and q:

$$s_i e(i) = q(i), i \in \mathcal{N}. \tag{B.4}$$

If we can solve the system and find e and q, then $f(x)$ can be constructed correctly as $f(x) = q(x)/e(x)$. As there are n equalities available with $3t + 1$ coefficients (the coefficient e_t in $e(x) = e_0 + e_1 x + \ldots + e_t x^t$ can be set as 1) in (B.4), the degree t of the polynomial should be smaller than $n/3$ in order to solve the equation. Thus, the Berlekamp-Welch algorithm allows to correct the secret even in the presence of t invalid shares in secret construction step as long as $t < n/3$. However, it does not prevent malicious behaviour (by even one malicious node) when creating the shares, as this node could create more than $n/3$ errors in the sharing process. This can be detected by the use of verifiable secret sharing [31]. We will not be concerned about this in this paper.

It is important to note that Shamir's secret sharing scheme is only applicable to fully connected graphs due to the fact that each node has to distribute shares to all other nodes. This affects the choice of distributed averaging algorithm and the possibility of graph topology relaxation. We will address these issue in the next section.

4 Proposed Approach

To approach the challenges of having an algorithm that is both distributed and privacy-preserving, we adopt a distributed asynchronous averaging algorithm based on [3] to compute the average iteratively, and Shamir's secret sharing scheme is then applied in each iteration of this algorithm to guarantee that the privacy of each node is protected. The detailed algorithm is described in Algorithm 0.

As previously mentioned, the application of Shamir's secret sharing scheme requires a fully connected graph, something that was also observed in [32]. However, such graphs are not always practical or scalable since they require a huge number of connections. Therefore, we adopt a distributed asynchronous averaging algorithm to relax the network topology requirement: as shown in step 4 of Algorithm 0, Shamir's secret sharing scheme is applied in a fully connected subset of nodes in each iteration. Thereby, we relax the impractical topology requirement, from a fully connected to a clique-based graph, and a preprocessing step named clique detection is added. The clique C_i of node i should satisfy

$$\begin{cases} C_i \subseteq \{d_i \cup i\}, n_i > 2, \\ \forall j,k \in C_i, j \neq k, (j,k) \in \mathcal{E}, \end{cases} \tag{B.5}$$

where n_i denotes the total node number in clique C_i. We can see that C_i need not be unique. The requirement $n_i > 2$ is simply due to the fact that one can always infer the other's initial state value with the final addition result if there are only two nodes [33]. The clique-based graph topology is required to guarantee that each node should have at least two neighbour nodes and all these three nodes are interconnected. In practice, the clique based graph is quite normal in distributed system (e.g., in wireless sensor networks) since the connectivity between certain nodes is typically enabled for nodes within a fixed distance of each other.

Algorithm 2 Proposed approach

Clique selection:

1: For all the nodes $i \in \mathcal{V}$ in the whole network, find all possible cliques C_i satisfy (B.5).

2: **Distributed asynchronous averaging [3]:**

3: Randomly activate one node i with uniform probability.

4: Node i choose one clique C_i and set the polynomial degree t based on adversary model, compute the addition result $y(k) = \sum_{j \in C_i} a_j(k)$ securely in selected clique based on Algorithm 0.

5: Update the node values as $a_j(k+1) = \frac{y(k)}{n_i}, j \in C_i$.

6: Repeat step 3-4 till convergence.

7: End

Algorithm 0 describes a solution to securely compute addition in the selected clique C_i based on the linearity of Shamir's secret sharing. The attack model is defined by parameter $flag$ in the algorithm description. If it is equal to 1, the algorithm is robust to one third errors in share distribution, otherwise only passive attack is considered. Concerning data representation, Shamir's secret sharing schemes works with integer numbers modulo a prime. Thus, a sufficiently large finite field F is selected to represent all the values in the modular domain $[0, p-1]$. Floating point numbers can be encoded as integers by

simply multiplying them with same scale factor and the negative numbers can be represented with modular additive inverse. A rounding operation is needed in step 5 of Algorithm 0 to make sure all the input values in Shamir's secret sharing are integers.

Algorithm 3 Secure addition using Shamir's secret sharing

Polynomial construction:

1: All nodes $i \in C_i$ agree a polynomial degree t based on adversary model (active or passive).
2: Each node p_i randomly choose coefficients $c_i^1, c_i^2, ..., c_i^t$ on F, construct polynomial
$$f_i(x) = a_i + c_i^1 x + c_i^2 x^2 + ... + c_i^t x^t \bmod p.$$
3: **Input sharing:**
4: Each node p_i computes shares $f_i(j)$ and distributes shares $f_i(j)$ to all other nodes $\{j \mid j \in C_i, j \neq i\}$, respectively.
5: Each node p_i receives shares $f_j(i)$ from all other nodes $j \in C_i, j \neq i\}$, respectively.
6: Each node p_i computes sum l_i based on received shares $l_i = \sum_{j=1}^{n} f_j(i)$.
7: Each node p_i broadcasts l_i.
8: **Output construction:**
9: If $flag = 1$ (active attack model)
10: Each node p_i defines $q(x)$ and $e(x)$ (see Section 3).
11: Each node p_i computes $q(x)$ and $e(x)$ based on share set $\{(i, l_i), i \in C_i\}$ with (B.4) and the desired polynomial $f(x)$ is determined by $q(x)/e(x)$.
12: Each node p_i computes the result $y = f(0)$.
13: Else (passive attack model)
14: Each node p_i computes r_i using (B.3).
15: Each node p_i computes the result $y = \sum_{i \in C_i} r_i l_i$.
16: End

5 Analysis

A comprehensive comparison of the proposed approach with existing approaches is shown in Table B.1, where β denotes the number of bits needed to represent encrypted cipher text [34]. We can see that the HE and GC approaches are computationally expensive and require high communication bandwidths, as the cipher texts after encryption usually require much longer bit length than plain texts. With the application of Shamir's secret sharing, the proposed algorithm outperforms differential privacy based approaches by having perfect security and identical accuracy with the non-privacy concerned algorithms [3–7]. Moreover, the involved functions are simpler than HE and GC based approaches and no trusted third party is required. The proposed

approach also addresses a more challenging adversary model than the other approaches. For each iteration, where a node i is activated with n_i nodes in its clique, the proposed approach needs extra communication times compared to the other approaches because of the share distribution process.

Table B.1: Comparisons with existing approaches

	Proposed	HE [22, 23]	GC [20]	Differential privacy [13–17]
Accuracy	Identical	Identical	Dependent on step size	Degraded with noise
Security	Perfect	Computational	Computational	Differential privacy
Attack model	Passive/Active	Passive	Passive	Passive
Involved function	Polynomial	Exponential	Exponential	Linear
Trusted Third Party	No	Yes	No	No
Communication bandwidth per time	$\mathcal{O}(1)$	$\mathcal{O}(\beta)$	$\mathcal{O}(\beta)$	$\mathcal{O}(1)$
Communication times per iteration	$\mathcal{O}(n_i^2)$	$\mathcal{O}(n_i)$	$\mathcal{O}(n_i)$	$\mathcal{O}(n_i)$

5.1 Security analysis under passive and active attack

There is a difference between the privacy concern from a cryptographic point of view and a practical point of view. From the cryptographic point of view, the security definition imposes a very strong demand, namely, that a protocol is only secure if the adversary does not learn more information about the inputs of the honest nodes than what is implied by the output and the inputs of the corrupted nodes. From this point of view, the computation in each clique (see Algorithm 0), when considered in isolation, is information-theoretic (i.e., perfect) secure [29], but the full computation in Algorithm 0 would not be considered secure, since the adversary can learn the partial sums of the honest nodes' initial state values in some cliques, and this is not implied by the average result of the full network and the corrupted nodes' initial state values. However, from a practical point of view, as already stated in Section 2.2, we are trying to protect the individual node's private information, and each individual node's initial state value is not revealed even if the sum of them are known.

For passive attacks, the privacy of the honest node will be protected as long as it has one honest neighbour in its clique. In a weaker model of active attacks, where the nodes act honestly when distributing the shares, but errors (either intentionally or unintentionally) can occur later on, the proposed algorithm can successfully reconstruct the correct result as long as at most one third of the shares are erroneous. This model captures cases such as unintentional errors produced when exchanging information. To the best of our knowledge, this is the first algorithm that obtains robustness against active attack in privacy-preserving distributed average consensus computation with both error detection and correction.

5.2 Security analysis under dynamic participation

One possible concern here is whether a clever combination of the information obtained in successive iterations can help to infer the privacy of the individual honest nodes, similarly to the privacy analysis in a dynamic setting [35] where

nodes may come and leave between executions. This is, however, difficult to analyze. Note that since the inputs of the nodes involved in different iterations are dynamically updated, this is different from the case considered in [35] wherein inputs are static. We remark that it is difficult (without additional knowledge) for a passive adversary to ascertain whether any two iterations are successively related due to the random nature of the node activation and the clique selection in Algorithm 0. It is, however, possible that such situations can occur, and future research should investigate this further.

6 Conclusions and Future Work

In this paper, we proposed a privacy-preserving distributed averaging consensus algorithm based on Shamir's secret sharing to compute the consensus in a distributed manner over a clique based network while protecting the individual privacy. The proposed algorithm is able to achieve both accurate consensus and perfect security at the same time, it does not depend on any trusted third party, and the computational complexity is lightweight. The adoption of Shamir's secret sharing allows to maintain the privacy of each individual node, i.e., as long as the clique selected in an iteration has at least 2 honest nodes. Moreover, robustness against up to one third errors is obtained under an active attack model. A drawback of the proposed approach is the higher communication times required by Shamir's secret sharing compared to, for example, differential privacy based methods. Future work will focus on how to reduce the overall communication times.

References

[1] J. N. Tsitsiklis, "Problems in decentralized decision making and computation.," Tech. Rep., Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, 1984.

[2] G. Cybenko, "Dynamic load balancing for distributed memory multiprocessors," *J. Parallel and Distributed Comput.*, vol. 7, no. 2, pp. 279–301, 1989.

[3] L. Xiao, S. Boyd, "Faster linear iterations for distributed averaging," *Syst. Control Lett.*, vol. 53, no. 1, pp. 65-78, 2004.

[4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[5] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, 2010.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[7] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process.*, vol. 4, no. 1, pp. 173–187, 2018.

[8] S. Aliaksei and K. Soummya and M. José MF, "Finite-time distributed consensus through graph filters," *ICASSP*, pp. 1080–1084, 2014.

[9] S. Santiago and M. Antonio G and R. Alejandro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, 2017.

[10] M. H. DeGroot, "Reaching a consensus," *J. Am. Statist. Assoc.*, vol. 69, no. 345, pp. 118–121, 1974.

[11] C. Dwork, "Differential privacy," *ICALP*, pp. 1–12, 2006.

[12] R. Cramer, I. B. Damgrd, and J. B. Nielsen, "Secure multiparty computation and secret sharing," Cambridge University Press, 2015.

[13] M. Kefayati, M. S. Talebi, B. H. Khalaj, and H. R. Rabiee, "Secure consensus averaging in sensor networks using random offsets," *Proc. of the IEEE Int. Conf. on Telec., and Malaysia Int. Conf. on Commun.*, pp. 556–560, 2007.

[14] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," *ACM workshop Privacy electron. Soc.*, pp. 81–90, 2012.

[15] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica*, vol. 81, pp. 221–231, 2017.

[16] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," *ECC*, pp. 760–765, 2013.

[17] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Automat Contr.*, vol. 62, no. 2, pp. 753–765, 2017.

[18] J. He, L. Cai, C. Zhao, P. Cheng, and X. Guan, "Privacy-preserving average consensus: privacy analysis and optimal algorithm design," *IEEE Trans. Signal Process.*, 2018.

[19] P. Braca, R. Lazzeretti, S. Marano, and V. Matta, "Learning with privacy in consensus + obfuscation," *IEEE signal process. Lett.*, vol. 23, no. 9, pp. 1174–1178, 2016.

[20] F. Hanzely, J. Konečnỳ, N. Loizou, P. Richtárik, and D. Grishchenko, "Privacy preserving randomized gossip algorithms," *arXiv preprint arXiv:1706.07636*, 2017.

[21] R. Lazzeretti, S. Horn, P. Braca, and P. Willett, "Secure multi-party consensus gossip algorithms," *ICASSP*, pp. 7406–7410, 2014.

[22] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy preserving distributed beamforming based on homomorphic encryption," *EUSIPCO*, pp. 1–5, 2013.

[23] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy-preserving distributed speech enhancement forwireless sensor networks by processing in the encrypted domain," *ICASSP*, pp. 7005–7009, 2013.

[24] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," *EUROCRYPT*, pp. 223–238, 1999.

[25] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," *Advances in Cryptology–CRYPTO*, pp. 643–662, 2012.

[26] A. C. Yao, "Protocols for secure computations," *FOCS*, pp. 160–164, 1982.

[27] A. C. Yao, "How to generate and exchange secrets," *FOCS*, pp. 162–167, 1986.

[28] C. Liu, "Introduction to combinatorial mathematics," 1968.

[29] A. Shamir, "How to share a secret," *Comm. Assoc. Comput. Mach.*, vol. 22, no. 11, pp. 612–613, 1979.

[30] L. R. Welch and E. R. Berlekamp, "Error correction for algebraic block codes," Dec. 1986.

[31] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch, "Verifiable secret sharing and achieving simultaneity in the presence of faults," *FOCS*, pp. 383–395, 1985.

[32] S. Goryczka and L. Xiong, "A comprehensive comparison of multiparty secure additions with differential privacy," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 5, pp. 463–477, 2017.

[33] S. C. S. Cheung and T. Nguyen, "Secure multiparty computation between distrusted networks terminals," *EURASIP J. Inf. Security*, vol. 2007, no. 1, pp. 051368, 2007.

[34] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Process. Magazine*, vol. 30, no. 1, pp. 82–105, 2013.

[35] D. Kononchuk, Z. Erkin, J. C. van der Lubbe, and R. L. Lagendijk, "Privacy-preserving user data oriented services for groups with dynamic participation," in *ESORICS*. pp. 418–442, 2013.

Paper C

Convex Optimisation-based Privacy-Preserving Distributed Average Consensus in Wireless Sensor Networks

Qiongxiu Li, Richard Heusdens, and Mads Græsbøll Christensen

The paper has been published in the
Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP), 2020

The layout has been revised.

Abstract

In many applications of wireless sensor networks, it is important that the privacy of the nodes of the network be protected. Therefore, privacy-preserving algorithms have received quite some attention recently. In this paper, we propose a novel convex optimization-based solution to the problem of privacy-preserving distributed average consensus. The proposed method is based on the primal-dual method of multipliers (PDMM), and we show that the introduced dual variables of the PDMM will only converge in a certain subspace determined by the graph topology and will not converge in the orthogonal complement. These properties are exploited to protect the private data from being revealed to others. More specifically, the proposed algorithm is proven to be secure for both passive and eavesdropping adversary models. Finally, the convergence properties and accuracy of the proposed approach are demonstrated by simulations which show that the method is superior to the state-of-the-art.

1 Introduction

Advances in wireless communication technology and embedded microprocessor design have enabled a huge growth of distributed computing systems, including also wireless sensor networks (WSNs). Average consensus, which is an essential building block of such distributed systems, has been intensively investigated for decades, and it has been applied in various fields such as automatic control, signal processing, robotics and optimisation [1]. To solve the average consensus problem in distributed networks, many (iterative) algorithms have been proposed [2–10]. The methods work by iteratively exchanging information between computational units (i.e., nodes/agents), whereby the network eventually reaches a consensus. The data exchange required in these algorithms can lead to privacy problems, as it is becoming clear that there is no real separation between the identity of individuals and their data [11]. Therefore, it is crucial to protect the data held by each node as the private data for being revealed to others.

An algorithm is called secure or privacy-preserving if it is able to protect the private data during the algorithm execution. Existing privacy-preserving distributed average consensus algorithms can be classified into two classes: computationally secure algorithms and information-theoretically secure algorithms. Computational security is defined in terms of computational hardness: secrets cannot be reconstructed efficiently under the condition that so-called malicious adversaries are computationally limited. Computationally secure algorithms [12–16] usually apply techniques from secure multiparty computation [17] such as homomorphic encryption (HE) [18, 19] and garbled circuit (GC) [20, 21], where computations are performed in the encrypted domain. However, these algorithms are computationally demanding and have high a

communication bandwidth. This makes it difficult to apply them in resource constrained applications like WSNs.

In contrast to the aforementioned computationally expensive algorithms, the information-theoretically secure algorithms are quite lightweight by comparison, as they simply insert noise to obfuscate the private data. Moreover, information-theoretic security has a stronger security guarantee than computational security as it is robust against a computationally unlimited adversary. Depending on the amount of information about the private data obtained by the adversary, information-theoretically secure algorithms can be further classified into two classes. The first class contains algorithms using secret sharing, whereby perfect security is achieved [22]. It possesses the strongest security guarantees. No information regarding the private data is revealed as the information obtained by the adversary is statistically independent of the private data. However, it requires prior knowledge about the network. The second class of algorithms achieves a weaker form of security, called ϵ-statistical security, which implies that the information obtained by the adversary is not totally independent of the private data but only results in a slightly better posterior guessing probability than the prior probability. Most ϵ-statistical security algorithms [23–25] adopt differential privacy [26, 27] to obfuscate the private data with independent noise. However, as shown in [25], differential privacy-based approaches cannot obtain the exact average and privacy at the same time. One way to circumvent the trade-off between accuracy and privacy is to guarantee that the inserted noise adds up to zero. Some algorithms [28–30] insert noise having a geometrically decreasing variance over iterations and guarantee that the inserted noise adds up to zero. Some other algorithms [31–33] rely on a trusted third party to obtain the zero-sum property. However, a trusted third party is hard to implement in ad hoc networks including also many WSNs.

As discussed above, the existing information-theoretically secure algorithms have some limitations, such as requiring prior knowledge of the network, the zero-sum property of the inserted noise, or the existence of a trusted third party. To address these limitations, we propose a convex optimisation-based method. To explain the basic concept, we show how it can be applied in the primal-dual method of multipliers (PDMM) [10, 34] which is an iterative algorithm for solving constrained convex optimisation problems. The concept can, however, also be applied to other convex optimisation methods, for example ADMM-based algorithms. As we shall see, the proposed method has a number of attractive properties: 1) the proposed algorithm obtains asymptotically perfect security and requires no trusted party nor prior knowledge about the network; 2) exact consensus and privacy can be obtained simultaneously; 3) the algorithm does not need zero-sum noise insertion but only a proper initialisation of the dual variables; 4) the convergence rate is independent of the privacy level; 5) the algorithm is secure under both passive and eavesdropping adversaries; and 6) the privacy of any honest node is guaranteed as long as it has one honest neighbour.

2 Preliminaries and Problem Definition

In this section, we will define the problem at hand and introduce some important definitions and concepts.

2.1 Distributed average consensus

Let $G = (V, E)$ denote a simple graph, where $V = \{1, 2, ..., n\}$ and $E = \{e_1, \ldots, e_m\} \subseteq V \times V$ denote the set of nodes and edges, respectively. The neighbourhood of node i is denoted as $N_i = \{j \in V \,|\, (i, j) \in E\}$ and the degree of node i is denoted by $d_i = |N_i|$. Finally, let $A \in \mathbb{R}^{n\times n}$ denote the adjacency matrix of the graph defined as $A_{ij} = 1$ if and only if $(i, j) \in E$, and let $B \in \mathbb{R}^{m\times n}$ denote the incidence matrix defined as $B_{li} = B_{i|j} = 1$ if and only if $e_l = (i, j) \in E$ and $i < j$ and $B_{li} = B_{i|j} = -1$ if and only if $e_l = (i, j) \in E$ and $i > j$. Distributed average consensus aims to estimate the average of all the initial state values given by

$$s_{\text{ave}} = n^{-1} \sum_{i\in V} s_i, \tag{C.1}$$

with s_i the initial state value of node i, without any centralised coordination. For simplicity, we will assume that s_i is a scalar but the results can easily be generalised to arbitrary dimensions.

2.2 Privacy concern and adversary model

In this work, the initial state value of each node is the private data to be protected. Most algorithms consider a passive adversary model (also known as the honest-but-curious model) where the instructions of the protocol are followed, but the so-called corrupted nodes might collude and attempt to deduce information about the initial state values of the other honest nodes from the messages they receive. The eavesdropping adversary is usually neglected in existing approaches since eavesdropping can be prevented by using channel encryption [35]. However, channel encryption is computationally expensive. For iterative algorithms where the communication channels between nodes are used many times, channel encryption is, therefore, less attractive. We thus assume that the communication in the network is performed through non-secure channels, except for the communication during the initialisation of the network.

2.3 Problem definition

The goal of privacy-preserving distributed average consensus algorithms is to design a protocol that jointly computes the average of all initial state values while protecting them from being revealed in the process. We thus have the following two requirements which need to be satisfied simultaneously:

1) Correctness: at the end of the algorithm, each node has obtained the average result $s_{\text{ave}} = n^{-1} \sum_{i \in V} s_i$.

2) Individual privacy: throughout the execution of the algorithm, the initial state value held by each honest node is protected against both passive and eavesdropping adversaries.

Some remarks are in order here. The adversary always knows the sum of the initial state values of the honest nodes, as it can be deduced from the average result and the initial states values of the corrupted nodes. Therefore, revealing this sum is unavoidable [17]. Furthermore, for incomplete (i.e., not fully connected) networks, as in the case in many practical networks, the partial sums of the honest nodes in each (connected) subgraph will be revealed as well, something that is also unavoidable for any information-theoretically private protocol [36, 37].

The corrupted nodes aim to infer the initial state value s_i of node i. Let s_i denote a realisation of a random variable S_i having differential entropy $h(S_i)$, assuming it exists[1], and let $g^{(k)}(S_i)$ denote the information sent out at iteration k by node i. We will measure the amount of privacy by

$$I(S_i; g^{(k)}(S_i)) = h(S_i) - h(S_i \,|\, g^{(k)}(S_i)), \tag{C.2}$$

where $I(\cdot\,;\cdot)$ denotes mutual information [38]. Note that $I(S_i; g^{(k)}(S_i)) = 0$ corresponds to perfect security in the sense that $h(S_i \,|\, g^{(k)}(S_i)) = h(S_i)$ so that S_i and $g^{(k)}(S_i)$ are statistically independent, while $I(S_i; g^{(k)}(S_i)) < \epsilon$, where $\epsilon > 0$, corresponds to ϵ-statistical security. Again, having perfect security at every iteration does not necessarily imply that $I(S_i; g^{(k)}(S_i), \ldots, g^{(0)}(S_i)) = 0$ since in the end the adversary is able to compute partial sums of connected subgraphs, but nothing else beyond that.

3 Primal-dual method of multipliers

The proposed approach is based on the primal-dual method of multipliers (PDMM), an instance of Peaceman-Rachford splitting of the extended dual problem (see [34] for details). PDMM can, like ADMM, be used for iteratively solving constrained convex optimisation problems. The PDMM update equations are given by

$$\begin{aligned} x^{(k+1)} &= \arg\min_x \left(f(x) + \lambda^{(k)^T} PCx + \frac{c}{2} \|Cx + PCx^{(k)}\|_2^2 \right), \\ \lambda^{(k+1)} &= P\lambda^{(k)} + c(Cx^{(k+1)} + PCx^{(k)}), \end{aligned} \tag{C.3}$$

where k denotes the iteration index, $x^{(k)} \in \mathbb{R}^n$ is the primal variable, $\lambda^{(k)} \in \mathbb{R}^{2m}$ the dual variable, $f(x)$ the objective function to be minimised, $C \in \mathbb{R}^{2m \times n}$

[1] In the case that S_i is a discrete random variable, the conditions are given in terms of the Shannon entropy $H(S_i)$.

a matrix related to the graph's incidence matrix B, and $P \in \mathbb{R}^{2m\times 2m}$ a symmetric permutation matrix exchanging the first m with the last m rows. The $c > 0$ is a constant controlling the convergence rate. The vector λ contains the dual variables controlling the constraints; for each edge $(i,j) \in E$ there are two node variables $\lambda_{i|j}$ and $\lambda_{j|i}$, one for each node i and j, respectively, where $\lambda(l) = \lambda_{i|j}$ and $C_{li} = B_{i|j}$ if and only if $e_l = (i,j) \in E$ and $i < j$, and $\lambda(l+m) = \lambda_{i|j}$, $C_{(l+m)i} = B_{i|j}$ if and only if $e_l = (i,j) \in E$ and $i > j$. Note that $C + PC = [B^T\ B^T]^T$ and $\forall (i,j) \in E : \lambda_{j|i} = (P\lambda)_{i|j}$.

Consider the update of two successive λ-updates, given by

$$\lambda^{(k+2)} = \lambda^{(k)} + c(Cx^{(k+2)} + 2PCx^{(k+1)} + Cx^{(k)}), \tag{C.4}$$

since $P^2 = I$. Let $H = \mathrm{ran}(C) + \mathrm{ran}(PC)$ where $\mathrm{ran}(\cdot)$ denotes the range, and let Π_H denote the orthogonal projection onto H. By inspection of (C.4), we conclude that every two PDMM updates only affect $\Pi_H\lambda \in H$ and leave $(I - \Pi_H)\lambda \in H^\perp$, $H^\perp = \mathrm{null}(C^T) \cap \mathrm{null}((PC)^T)$ unchanged, where $\mathrm{null}(\cdot)$ denotes the null space. Moreover, by inspecting (C.3), we conclude that the x-update is independent of $(I - \Pi_H)\lambda$ since $\lambda^T(I - \Pi_H)PC = 0$. As a consequence, the component $(I - \Pi_H)\lambda$ will only be permuted every iteration and therefore not converge. We will refer to $\Pi_H\lambda$ and $(I - \Pi_H)\lambda$ as the converging and non-converging component of the dual variable, respectively.

4 Proposed approach

The distributed average consensus problem can be formulated as an optimisation problem where we minimise the objective function

$$f(x) = \frac{1}{2}\|x - s\|_2^2, \tag{C.5}$$

where $s = (s_1, \ldots, s_n)^T$, subject to the constraint that $x_i = x_j$ for all $(i,j) \in E$. The solution is given by $x^* = s_{\mathrm{ave}}(1, \ldots, 1)^T$. That is, all nodes in the network eventually know the average. The PDMM update equation (C.3) for this problem is then given by

$$x^{(k+1)} = (I + cD)^{-1}\left(s + cAx^{(k)} - C^T P\lambda^{(k)}\right), \tag{C.6}$$

where $D = C^TC$ is the degree matrix of the underlying graph and $C^TPC = -A$. The update equations for node i then become

$$x_i^{(k+1)} = \frac{s_i + \sum_{j\in N_i}\left(cx_j^{(k)} - B_{i|j}\lambda_{j|i}^{(k)}\right)}{1 + cd_i}, \tag{C.7}$$

$$\forall j \in N_i : \quad \lambda_{i|j}^{(k+1)} = \lambda_{j|i}^{(k)} + cB_{i|j}\left(x_i^{(k+1)} - x_j^{(k)}\right). \tag{C.8}$$

From (C.8) we can see that the update of the dual variables only depends on $\lambda_{j|i}^{(k)}$, $x_j^{(k)}$ and $x_i^{(k+1)}$, of which $\lambda_{j|i}^{(k)}$ and $x_j^{(k)}$ are already available at node j. Therefore, after broadcasting $x_i^{(k+1)}$, all neighbouring nodes can construct $\lambda_{i|j}^{(k+1)}$ and the dual variables do not need to be transmitted at all, except for the initialisation, as all $\lambda_{j|i}^{(0)}$s need to be known at the first iteration.

As mentioned before, the non-converging component $(I-\Pi_H)\lambda^{(k)}$ will only be permuted every iteration so that

$$\lambda^{(k)} \to \lambda^* + \begin{cases} (I-\Pi_H)\lambda^{(0)}, & k \text{ even}, \\ P\,(I-\Pi_H)\,\lambda^{(0)}, & k \text{ odd}, \end{cases} \tag{C.9}$$

where λ^* is given by

$$\lambda^* = -\begin{pmatrix} C^T \\ (PC)^T \end{pmatrix}^\dagger \begin{pmatrix} \nabla f(x^*) + cC^TCx^* \\ \nabla f(x^*) + cC^TPCx^* \end{pmatrix} + cCx^*, \tag{C.10}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo inverse. As a consequence, if we initialise the dual variable λ in such a way that the non-converging component $(I-\Pi_H)\lambda^{(0)}$ sufficiently obfuscates the initial state value, the primal variables will converge to s_{ave} while the initial state value itself cannot be inferred, assuming there is at least one honest neighbour. We will prove this claim more formally in what follows.

4.1 Correctness

As shown in [34], the primal variable $x^{(k)}$ will converge geometrically to x^* for arbitrary initialisation $x^{(0)}$ and $\lambda^{(0)}$, thereby proving the correctness of the algorithm.

4.2 Individual privacy

We will now proceed to prove that the proposed algorithm protects the individual privacy under both passive and eavesdropping adversaries. As we can see, each node transmits only the primal variable $x_i^{(k+1)}$ to all of its neighbours and does not reveal its initial state value s_i directly. To analyse the privacy properties of the proposed algorithm, let V_c and V_h denote the set of corrupted and honest nodes, respectively. With this, the numerator of (C.7) can be expressed as

$$\begin{aligned} &s_i + \sum_{j\in N_i} \left(cx_j^{(k)} - B_{i|j}\lambda_{j|i}^{(k)} \right) = \\ &\quad s_i + \sum_{j\in N_i} cx_j^{(k)} - \sum_{j\in N_i\cap V_h} B_{i|j}\lambda_{j|i}^{(k)} - \sum_{j\in N_i\cap V_c} B_{i|j}\lambda_{j|i}^{(k)}. \end{aligned} \tag{C.11}$$

At convergence, x^* is known and λ^* can be calculated through (C.10). Hence, by inspection of (C.9) and (C.11), we conclude that the adversary can infer about the initial state value s_i from observing $x_i^{(k+1)}$ is the term given by

$$s_i - \sum_{j \in N_i \cap V_h} B_{i|j} \left(P^k (I - \Pi_H) \lambda^{(0)} \right)_{j|i}, \tag{C.12}$$

and we conclude that, as long as $N_i \cap V_h \neq \emptyset$, we can obfuscate the initial sate value by introducing uncertainty in $(I - \Pi_H)\lambda^{(0)}$.

To quantitatively measure the amount of information carried by $x_i^{(k)}$ about s_i, consider both $x_i^{(k)}$ and s_i as realisations of the random variables $X_i^{(k)}$ and S_i, respectively. We will analyse the mutual information $I(S_i; X_i^{(k)})$ between S_i and $X_i^{(k)}$ for which we need the following result.

Proposition 1. *Let X and Y be independent continuous random variables with $\mathrm{var}(X), \mathrm{var}(Y) < \infty$ and let $Z = X + Y$. Then*

$$\lim_{\mathrm{var}(Y) \to \infty} I(X; Z) = 0,$$

assuming $I(X; Z)$ exists.

Proof. Let $\gamma = 1/(\mathrm{var}(Y))^{\frac{1}{2}}$ and define $Y' = \gamma Y$. Hence, Y' has unit variance. Since mutual information is invariant under scaling, we have $I(X; Z) = I(X; X + Y) = I\left(\gamma X; \gamma X + Y'\right)$. As a consequence, we have

$$\begin{aligned} \lim_{\mathrm{var}(Y) \to \infty} I(X; Z) &= \lim_{\gamma \to 0} I\left(\gamma X; \gamma X + Y'\right) \\ &= I\left(0; Y'\right) = 0. \end{aligned}$$

□

By applying Proposition 1, we can conclude that the mutual information $I(S_i; X_i^{(k)})$ can be made arbitrarily small by increasing the variance of the random variable representing the λ-contribution in (C.12). That is, let $\lambda^{(0)}$ be a realisation of the random variable $\Lambda^{(0)}$. Then we have $I(S_i; X_i^{(k)}) = 0$ if

$$\exists j \in N_i \cap V_h : \mathrm{var}\left(\left((I - \Pi_H) \Lambda^{(0)} \right)_{j|i} \right) \to \infty. \tag{C.13}$$

Hence, the proposed algorithm obtains asymptotically perfect security. A summary of the complete privacy-preserving PDMM algorithm is given in Algorithm 4.

Some remarks are in order here. Firstly, since the dual variables are not transmitted at all, except during initialisation for which we need secure communication, no encryption is needed during the execution of the algorithm. Secondly, a necessary condition for achieving privacy is that $N_i \cap V_h \neq \emptyset$. That is, node i requires at least one honest neighbour. In the case the graph is complete, this means that the algorithm is secure up to $n - 2$ malicious nodes in

Algorithm 4 Privacy-preserving PDMM

1: Each node $i \in V$ initialises its primal and dual variables. The dual variables are initialised with random numbers having sufficiently large variance (depending on the required privacy level), whereas the primal variables can be initialised arbitrarily.
2: Each node $i \in V$ communicates its dual variables $\lambda_{i|j}^{(0)}$ to its neighbour $j \in N_i$ through secure channels [35].
3: **while** $\|x^{(k)} - x^*\|_2 <$ threshold **do**
4: Activate a node uniformly at random, say node i, updates its primal variable $x_i^{(k+1)}$ according to (C.7).
5: Node i broadcasts $x_i^{(k+1)}$ to all of its neighbours $j \in N_i$ (through non-secure channels).
6: After receiving $x_i^{(k+1)}$ by the neighbours, the dual variables $\lambda_{i|j}^{(k+1)}$ are updated using (C.8).
7: **end while**

the network. Thirdly, although we proved that the mutual information is zero under the condition of (C.13), the variance of the dual variables cannot be made infinitely large. Therefore, information about the initial state variables will be leaked upon receiving the primal variables. To have an indication of the amount of leakage in practice, consider the following example of two independent Gaussian distributed random variables X and Y and their sum $Z = X+Y$. The differential entropy of a Gaussian random variable with variance σ^2 is given by $\frac{1}{2}\log(2\pi e\sigma^2)$, so that $I(X;Z) = h(Z) - h(Y) = \frac{1}{2}\log(1+\sigma_X^2/\sigma_Y^2)$. Hence, if we have $\sigma_Y^2/\sigma_X^2 = 100$ (the range of Y is approximately 10 times the range of X), the information leakage is only 0.007 bits. Fourthly, in order to satisfy (C.13), a necessary condition is that $\lambda^{(0)} \cap H^\perp \neq \emptyset$. By inspection of the matrix C, we conclude that the matrix $[C, PC] \in \mathbb{R}^{2m\times 2n}$ can be considered as the incidence matrix of a bipartite graph having $2n$ nodes. As a consequence, we have that $\mathrm{rank}([C, PC]) \leq 2n-1$ and we conclude that $\dim(H) \leq 2n-1$ and thus $H^\perp \neq \emptyset$. Hence, if we randomly initialise $\lambda^{(0)}$, we have $(I - \Pi_H)\lambda^{(0)} \neq 0$ with probability one. Last but not least, the proposed algorithm can also be applied to other convex optimisation methods such as ADMM and related algorithms where the update equations have a similar structure.

5 Experimental results

Now we proceed to evaluate the performance of the proposed algorithm by simulations in terms of the mean square error (MSE) of primal and dual variables as a function of transmission number. We generated a random geometric network with $n = 10$ nodes where two nodes can communicate if their distance is within a radius r satisfying $r^2 = 2\frac{\log n}{n}$, thereby guaranteeing a connected

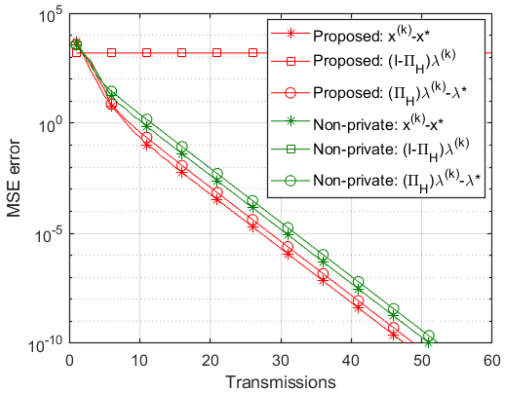


Fig. C.1: Convergence of the primal variable, the converging component and non-converging component of the dual variable in PDMM with two different initialisations.

graph with probability at least $1 - \frac{1}{n^2}$ [39]. For simplicity, we use uniform distribution as an example to demonstrate the results, where the initial state values s_i are uniformly distributed in the interval $[0, 1]$.

Figure C.1 shows the convergence behavior of PDMM for different initialisations. The red lines show the proposed PDMM algorithm in which $x^{(0)}$ is initialised with all zeros and $\lambda^{(0)}$ is randomly initialised with uniformly distributed noise in the interval $[0, 100]$. The green lines show results where the dual variable is initialised in H such that $\lambda^{(0)} \cap H^{\perp} = \emptyset$, which implies that the initial state values are not protected. The star, square, and circle marker show the convergence of $x^{(k)}$, $(I - \Pi_H)\lambda^{(k)}$ and $\Pi_H \lambda^{(k)}$, respectively. We see that for both initialisations $x^{(k)}$ and $\Pi_H \lambda^{(k)}$ converge to the optimal solutions x^* and λ^*, respectively. The magnitude of $(I - \Pi_H)\,\lambda^{(k)}$, on the other hand, does not converge. As a consequence, the proposed algorithm protects the initial state value by obfuscating it with a high-variance non-converging component $(I - \Pi_H)\,\lambda^{(k)}$. Note that the green line with square marker is not visible since $(I - \Pi_H)\,\lambda^{(k)} = 0$ for all k.

Figure C.2 shows a comparison of the proposed PDMM approach with popular state-of-the-art information-theoretically secure algorithms including differential privacy (DP) [25] and the correlated noise insertion approach (CNI) [30], where we compare the effect of adding noise on the convergence rate of the algorithm. We considered three different noise levels: $\Gamma = 0, 10^2$, and 10^4, where Γ denotes the ratio of noise variance to the variance of initial state value. The case $\Gamma = 0$ corresponds to the situation where no noise is added so that the initial state values are not protected. In the other cases we inserted noise having an initial range approximately 10 and 100 times the range of initial state values, therefore we have $\Gamma = 10^2$ and 10^4, respectively. We observe, as expected, that the accuracy of the differential privacy approach (black lines) decreases with increasing noise variance and that for $\Gamma \neq 0$ the algorithm does not converge anymore. That is, with differential privacy, there is a trade-off

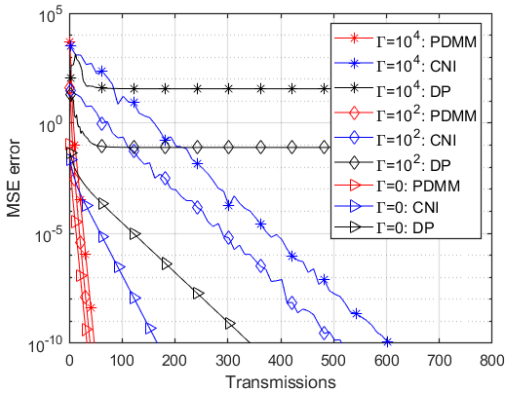


Fig. C.2: Convergence of the proposed PDMM and state-of-the-art algorithms under three different noise levels.

between privacy and accuracy. As for correlated noise insertion (blue lines), high accuracy is obtained in the end (the algorithm is guaranteed to converge) but the convergence rate slows down with increasing noise variance. The convergence rate of the proposed PDMM-based algorithm (red lines), on the other hand, is independent of the noise level since the convergence rate of PDMM depends on the graph topology and not on the initialisation; increasing the noise variance will only result in a higher initial error.

6 Conclusions

In this paper, we proposed a novel lightweight privacy-preserving distributed average consensus algorithm for WSNs based on PDMM, a convex optimisation algorithm. By simply initialising the dual variable with random numbers, the non-converging component of the dual variable will obfuscate the initial state values, thereby protecting them from being revealed. We showed that the proposed algorithm achieves asymptotically perfect security under a passive adversary, where the privacy is guaranteed as long as there is at least one honest neighbour. For an eavesdropping adversary, the proposed algorithm does not require secure channel encryption in the network except for the initialisation step. Compared to existing information-theoretically secure algorithms, the proposed algorithm has no trade-off between accuracy and privacy, and converges at a rate independent of the amount of inserted noise and, thus, of the level of privacy.

References

[1] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *IEEEE Proc., vol. 95, no. 1, pp. 215-233*, 2007.

[2] L. Xiao, S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Control Lett., vol. 53, no. 1, pp. 65-78*, 2004.

[3] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory, vol. 52, no. 6, pp. 2508–2530*, 2006.

[4] A. G. Dimakis, S. Kar, J. M. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE, vol. 98, no. 11, pp. 1847–1864*, 2010.

[5] C.M. Bishop, *Pattern recognition and machine learning*, springer, 2006.

[6] J.Pearl, *Reverend Bayes on inference engines: A distributed hierarchical approach*, Proc. 1982 Am. Assoc. Artificial Intell., pp. 133-136, 1982.

[7] S. Aliaksei and K. Soummya and M. José MF, "Finite-time distributed consensus through graph filters," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 1080–1084*, 2014.

[8] S. Santiago, M. Antonio G and R. Alejandro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process., vol. 65, no. 15, pp. 4117-4131*, 2017.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning, vol. 3, no. 1, pp. 1–122*, 2011.

[10] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process., vol. 4, no. 1, pp. 173–187*, 2018.

[11] D. Sarwate and K. Chaudhuri, "Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data," *IEEE Signal Process. Magazine, vol. 30, no. 5, pp. 86–94*, 2013.

[12] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy preserving distributed beamforming based on homomorphic encryption," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2013.

[13] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 7005–7009*, 2013.

[14] M. H. Ruan, M. Ahmad, Y. Q. Wang, "Secure and privacy-preserving average consensus," in *Proc. Workshop Cyber-Phys. Syst. Secur. Privacy, pp. 123-129*, 2017.

[15] C. Zhang, M. Ahmad, and Y. Wang, "ADMM based privacy-preserving decentralized optimization," *IEEE Trans. Inf. Forensics Security, vol. 14, no. 3, pp. 565–580*, 2019.

[16] F. Hanzely, J. Konečnỳ, N. Loizou, P. Richtárik, and D. Grishchenko, "Privacy preserving randomized gossip algorithms," *arXiv preprint arXiv:1706.07636*, 2017.

[17] R. Cramer, I. B. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*, Cambridge University Press, 2015.

[18] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT, pp. 223–238*, 1999.

[19] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO, pp. 643–662*. Springer, 2012.

[20] A. C. Yao, "Protocols for secure computations," in *FOCS, pp. 160–164*, 1982.

[21] A. C. Yao, "How to generate and exchange secrets," in *FOCS, pp. 162–167*, 1986.

[22] Q. Li, I. Cascudo, and M. G. Christensen, "Privacy-preserving distributed average consensus based on additive secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

[23] M. Kefayati, M. S. Talebi, B. H. Khalajand H. R. Rabiee , "Secure consensus averaging in sensor networks using random offsets," in *Proc. of the IEEE Int. Conf. on Telec., and Malaysia Int. Conf. on Commun., pp. 556–560*, 2007.

[24] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," in *ACM workshop Privacy electron. Soc., pp. 81–90*, 2012.

[25] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica, vol. 81, pp. 221–231*, 2017.

[26] C. Dwork, "Differential privacy," in *ICALP, pp. 1–12*, 2006.

[27] C. Dwork, F. McSherry, K. Nissim, A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory of Cryptography Conf. , pp. 265-284*, 2006.

[28] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *ECC, pp. 760–765*, 2013.

[29] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Automat Contr., vol. 62, no. 2, pp. 753–765*, 2017.

[30] J. He, L. Cai, C. Zhao, P. Cheng, X. Guan, "Privacy-preserving average consensus: privacy analysis and algorithm design," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 1, pp. 127–138*, 2019.

[31] P. Braca, R. Lazzeretti, S. Marano, and V. Matta, "Learning with privacy in consensus + obfuscation," *IEEE signal process. Lett., vol. 23, no. 9, pp. 1174–1178*, 2016.

[32] M. T. Hale, M. Egerstedt, "Differentially private cloud-based multi-agent optimization with constraints," in *Proc. American Control Conf., pp. 1235-1240*, 2015.

[33] M. T. Hale, M. Egerstedt, "Cloud-enabled differentially private multiagent optimization with constraints," *IEEE Trans. Control Netw. Syst., vol. 5, no. 4, pp. 1693–1706*, 2018.

[34] T. Sherson, R. Heusdens, W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 2, pp 334-347*, 2018.

[35] D. Dolev, C. Dwork, O. Waarts, M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach., vol. 40, no. 1, pp. 17-47,*, 1993.

[36] G. Kreitz, M. Dam, and D. Wikstrom, "Practical private information aggregation in large networks," in *In: Aura, T., Järvinen, K., Nyberg, K. (eds.) NordSec. LNCS, vol. 7127, pp. 89-103*, 2010.

[37] Beimel, A, "On private computation in incomplete networks," *Distrib. Comput. 19(3), 237-252*, 2007.

[38] T. M. Cover and J. A. Tomas, *Elements of information theory*, John Wiley & Sons, 2012.

[39] J. Dall and M. Christensen, "Random geometric graphs," *Physical review E, vol. 66, no. 1, pp. 016121*, 2002.

Paper D

Convex optimization-based Privacy-Preserving Distributed Least Squares via Subspace Perturbation

Qiongxiu Li, Richard Heusdens, and Mads Græsbøll Christensen

The paper has been published in the
Proc. Eur. Signal Process. Conf. (EUSIPCO) 2021

The layout has been revised.

Abstract

Over the past decades, privacy-preservation has received considerable attention, not only as a consequence of regulations such as the General Data Protection Regulation in the EU, but also from the fact that people are more concerned about data abuse as the world is becoming increasingly digitized. In this paper we propose a convex optimization-based subspace perturbation approach to solve privacy-preserving distributed least squares problems. Based on the primal-dual method of multipliers, the introduced dual variables will only converge in a subspace determined by the graph topology and do not converge in its orthogonal complement. We, therefore, propose to exploit this property for privacy-preservation by using the non-converging part of the dual variables to perturb the private data, thereby protecting it from being revealed. Moreover, we prove that the proposed approach is secure under both eavesdropping and passive adversaries. Computer simulations are conducted to demonstrate the benefits of the proposed approach through its convergence properties and accuracy.

1 Introduction

In modern systems, such as smart grids and smart internet-of-things, the trend is to have collaborations between different parties. This distributed processing has a number of advantages over centralised processing, like avoiding a single point of failure and being robust against changes in the network topology. Such distributed systems usually require data exchange among the parties. These data, more often than not, contain sensitive information about individual parties/agents. For example, it was shown in [1] that even electricity consumption data can reveal sensitive information about the consumers' privacy such as whether the consumer has illnesses/disabilities or not. To address such privacy issues in distributed processing, in this paper we focus on privacy-preserving distributed least squares as it is a fundamental problem and serves as a building block to many other problems such as robust signal de-noising and linear regression in machine learning.

The privacy issue in distributed processing has been addressed in the literature by either protecting the private data using secure multiparty computation (SMPC) techniques or by perturbing it with noise insertion. SMPC [2] aims to jointly compute a function among a group of parties while keeping each party's input private. Popular SMPC protocols like secret sharing, homomorphic encryption, garbled circuits and hybrid methods have been applied in linear regression problems in machine learning [3–6]. However, these SMPC-based frameworks usually assume either a non-colluding trusted third party (TTP) or a small network with only a few computing parties. Consequently, they are quite far from being applied in large scale networks such as wireless sensor networks and many other applications where a TTP is hard to implement. To alleviate these problems, both distributed computation and SMPC

were employed in [7] for solving the privacy-preserving recursive least squares problems. Unfortunately, it comes at the cost of high communication complexity.

Noise insertion can be an attractive alternative as it is lightweight and usually does not require a TTP. A noise insertion framework for perturbing private data by balancing the privacy level with the output accuracy (referred to as differential privacy (DP) [8]), has been applied in many applications like robust statistic [9], Kalman filtering [10] and distributed average consensus [11], etc. In principle, it can also be applied to the distributed least squares problem. However, as stated in [11], there is an inherent trade-off between privacy and accuracy, and they can not be achieved simultaneously.

To address the above mentioned limitations, we here propose a novel convex optimization-based subspace perturbation approach which protects the private data by adding noise in a particular subspace. We use the primal-dual method of multipliers (PDMM) [12, 13], a distributed algorithm for solving constrained convex optimization problems, to illustrate the main idea of subspace perturbation, but the approach will work with other algorithms, like ADMM, as well. A number of attractive properties of the proposed approach are: 1) it is fundamentally different from the DP approaches as it is able to achieve both privacy and accuracy at the same time; 2) it requires no TTP and has a low computational complexity; 3) it converges at a rate independent of the privacy level and 4)it is secure under both passive and eavesdropping adversaries.

2 Fundamentals and problem Setup

In this section, we will first recall the fundamentals of the distributed least squares and explain the motivation for privacy-preservation. Next, we introduce the so-called adversary models, an essential concept when considering privacy, and then state the problem setup.

2.1 Distributed least squares

Given a distributed network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{1, \dots, n\}$ the set of nodes and $\mathcal{E} = \{e_1, \dots, e_m\}$ the set of edges. The neighbourhood of node i is denoted as $\mathcal{N}_i = \{j | (i,j) \in \mathcal{E}\}$ and $d_i = |\mathcal{N}_i|$. Let $B \in \mathbb{R}^{m \times n}$ denote the incidence matrix defined as $B_{li} = B_{i|j} = 1$ if and only if $e_l = (i,j) \in \mathcal{E}$ and $i < j$, $B_{li} = B_{i|j} = -1$ if and only if $e_l = (i,j) \in \mathcal{E}$ and $i > j$.

The goal of distributed least squares is to find a solution of an overdetermined system (set of equations in which there are more equations than unknowns), where each node only knows part of the equations and is only able to exchange information with its neighbours. Let $Q_i \in \mathbb{R}^{N_i \times u}$, $N_i > u$, denote a matrix containing the input observations of node i. That is, each node i has N_i observations and each observation contains an u-dimensional feature

vector. Moreover, let $y_i \in \mathbb{R}^{N_i}$ denote the decision vector observed by node i. Stacking all the local information such that $Q = [Q_1^T, \ldots, Q_n^T]^T \in \mathbb{R}^{N \times u}$ and $y = [y_1^T, \ldots, y_n^T]^T \in \mathbb{R}^N$ where $N = \sum_{i \in \mathcal{V}} N_i$, the least-squares problem is given by

$$\min_x \frac{1}{2}\|y - Qx\|_2^2.$$

We can formulate the least-squares problem as a distributed linearly-constrained convex optimization problem given by

$$\begin{aligned} \min_{\{x_i\}} f(x) &= \sum_{i \in \mathcal{V}} \frac{1}{2}\|y_i - Q_i x_i\|_2^2 \\ \text{s.t. } & x_i - x_j = 0, \forall (i,j) \in \mathcal{E}, \end{aligned} \tag{D.1}$$

where $x_i \in \mathbb{R}^u$ denotes the local estimated least-squares solution at node i. A number of distributed optimizers (e.g., ADMM, PDMM) has been proposed to solve the above problem by only exchanging information in the local neighbourhood. At every iteration k, each node i updates its local estimate $x_i^{(k)}$ based on a certain local updating function and then sends it to its neighbours. Generally, this local updating function requires local information of node i (that is, Q_i, y_i) to guarantee that $x_i^{(k)}$ converges to the global optimum solution $x^* = \arg\min_x \frac{1}{2}\|y - Qx\|_2^2$.

2.2 Privacy concerns

The local information (input observations Q_i and decision vector y_i) of each node is considered as private data and should be protected from being revealed. This is because it usually contains sensitive private information about individuals. For example, assume a number of hospitals participate in a research project with the aim of obtaining a predictive model by collaboratively learning all the data in their medical data sets. However, releasing this medical data violates the privacy regulation as it contains sensitive information of the patients such as their health conditions and insurance records. As mentioned earlier, at each iteration of the distributed computation, each node will send out the updated $x_i^{(k)}$ where the related updating function usually takes the private data Q_i and y_i as inputs. As a consequence, the updated $x_i^{(k)}$ carries information about the concerned private data and thus revealing it will inevitably cause loss of privacy. Such privacy issues will be investigated and addressed in the rest of the paper.

2.3 Adversary model

The adversary model qualifies the robustness of a privacy-preserving algorithm under security attacks. An adversary usually works by colluding a number of nodes to conduct certain malicious behaviours, such as learning the private

data and manipulating the outputs of the computations. These colluded nodes will be referred to as corrupted nodes and while the others will be referred to as honest nodes. Here we consider two general adversary models that are often encountered in real applications: passive and eavesdropping. In the former case, all nodes follow the instructions of the algorithm but they are curious about knowing the private data held by other honest nodes. The eavesdropping adversary, either internal or external, aims to infer the private data by eavesdropping the communication channels between honest nodes. This adversary has not received much attention in privacy-preserving distributed computation as it is commonly solved by assuming securely encrypted communication channels [14]. Encryption, however, incurs high computational complexity which is particularly cumbersome using iterative algorithms such as the ones we are using here, because communication channels are used many times. In this paper, we alleviate this problem and assume all the communication is done through non-secure channels except for the initialization.

2.4 Privacy-preserving distributed least squares

Combining things together, we conclude that there are two key requirements to be satisfied simultaneously:

1. Output correctness: all nodes are able to obtain the optimum solution $x^* = \arg\min_x \frac{1}{2}\|y - Qx\|_2^2$ when the algorithm converges.
2. Individual privacy: the concerned private data (Q_i, y_i) held by each node is protected from being revealed to others against both passive and eavesdropping adversaries, throughout the whole algorithm execution.

3 Primal-dual method of multipliers

We use PDMM as an example to explain the main idea of subspace perturbation. PDMM, like ADMM, is a distributed optimizer for solving constrained convex optimization problems. As an instance of Peaceman-Rachford splitting of the extended dual problem (see [13] for details), PDMM is characterised by a faster convergence rate compared to ADMM. The update equations of PDMM are given by

$$\begin{aligned} x^{(k+1)} &= \arg\min_x \big(f(x) + \lambda^{(k)^T} PCx + \frac{c}{2}\|Cx + PCx^{(k)}\|_2^2\big), \\ \lambda^{(k+1)} &= P\lambda^{(k)} + c(Cx^{(k+1)} + PCx^{(k)}), \end{aligned} \tag{D.2}$$

where $f(x)$ denotes the objective function to be minimised, k the iteration index, $x^{(k)} \in \mathbb{R}^n$ is the primal variable, $\lambda^{(k)} \in \mathbb{R}^{2m}$ the dual variable, $P \in \mathbb{R}^{2m\times 2m}$ a symmetric permutation matrix which exchanges the first m with the last m rows and $C \in \mathbb{R}^{2m\times n}$ a matrix related to the incidence matrix B. The

constant $c > 0$ controls the convergence rate. The vector λ contains the dual variables for the constraints; there are two dual variables $\lambda_{i|j}$ and $\lambda_{j|i}$, one for each node i and j, for each edge $(i,j) \in E$; where $\lambda(l) = \lambda_{i|j}$ and $C_{li} = B_{i|j}$ if and only if $e_l = (i,j) \in E$ and $i < j$, and $\lambda(l+m) = \lambda_{i|j}$, $C_{(l+m)i} = B_{i|j}$ if and only if $e_l = (i,j) \in E$ and $i > j$. Note that $C + PC = [B^T\, B^T]^T$ and $\forall (i,j) \in E : \lambda_{j|i} = \big(P\lambda\big)_{i|j}$.

The λ-updates of two successive iterations is given by

$$\lambda^{(k+2)} = \lambda^{(k)} + c(Cx^{(k+2)} + 2PCx^{(k+1)} + Cx^{(k)}), \tag{D.3}$$

as $P^2 = I$. Let $H = \mathrm{ran}(C) + \mathrm{ran}(PC)$ and $H^\perp = \mathrm{null}(C^T) \cap \mathrm{null}((PC)^T)$ where $\mathrm{ran}(\cdot)$ and $\mathrm{null}(\cdot)$ denote the range and nullspace, respectively. Note that $[C, PC] \in \mathbb{R}^{2m \times 2n}$ can be viewed as an incidence matrix of a new graph having $2n$ nodes and $2m$ edges. Therefore, we have $\dim(H) \leq 2n - 1$ and thus $H^\perp$ is always non-empty. Let Π_H denote the orthogonal projection onto H. From (D.3) we can see that every two λ-updates only affect $\Pi_H \lambda \in H$ and leave $(I - \Pi_H)\lambda \in H^\perp$ unchanged. As a consequence, the component $(I - \Pi_H)\lambda$ will not converge and only be permuted every iteration. We can thus divide the dual variable $\lambda^{(k)}$ into two parts given by

$$\lambda^{(k)} = \Pi_H \lambda^{(k)} + \begin{cases} (I - \Pi_H)\lambda^{(0)}, & k \text{ even}, \\ P\,(I - \Pi_H)\,\lambda^{(0)}, & k \text{ odd}. \end{cases} \tag{D.4}$$

It is proven in [13] that $\Pi_H \lambda^{(k)}$ converges to the optimum λ^* given by

$$\lambda^* = -\begin{pmatrix} C^T \\ (PC)^T \end{pmatrix}^\dagger \begin{pmatrix} \nabla f(x^*) + cC^T Cx^* \\ \nabla f(x^*) + cC^T PCx^* \end{pmatrix} + cCx^*, \tag{D.5}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo inverse. We thus denote $\Pi_H \lambda$ and $(I - \Pi_H)\lambda$ as the converging and non-converging component of the dual variable, respectively. Similarly, H and $H^\perp$ are referred to as the converging subspace and non-converging subspace of PDMM. It is worthy to mention that this non-converging component $(I - \Pi_H)\lambda$ would not affect the x-update in (D.2) since $\lambda^T (I - \Pi_H) PC = 0$.

4 Proposed approach

Having introduced PDMM, we will now proceed to describe the proposed approach. For the problem at hand, the PDMM updating functions for node i become

$$\begin{aligned} x_i^{(k+1)} &= (Q_i^T Q_i + cd_i I)^{-1} (Q_i^T y_i + \sum_{j \in \mathcal{N}_i} (cx_j^{(k)} - B_{i|j} \lambda_{j|i}^{(k)})) \\ \forall j \in \mathcal{N}_i : \quad \lambda_{i|j}^{(k+1)} &= \lambda_{j|i}^{(k)} + cB_{i|j}\big(x_i^{(k+1)} - x_j^{(k)}\big), \end{aligned} \tag{D.6}$$

whereas the update of dual variable $\lambda_{i|j}^{(k+1)}$ only depends on $\lambda_{j|i}^{(k)}$, $x_j^{(k)}$ and $x_i^{(k+1)}$, of which $\lambda_{j|i}^{(k)}$ and $x_j^{(k)}$ are local information held by node j. Therefore, $x_i^{(k+1)}$ is the only information needs to be transmitted by node i to its neighbours. After broadcasting $x_i^{(k+1)}$, all neighbouring nodes can construct $\lambda_{i|j}^{(k+1)}$ themselves and the dual variables do not need to be transmitted at all, except for the first iteration where the initialized $\lambda_{j|i}^{(0)}$s need to be transmitted.

Since $x_i^{(k+1)}$ is the only revealed information, by inspecting the x-update in (D.6) we can see that $x_i^{(k+1)}$ is dependent of node i's private data Q_i, y_i and the data $x_j^{(k)}, \lambda_{j|i}^{(k)}$ from its neighbours. We therefore propose to initialize the dual variables in a way such that the non-converging component $(I - \Pi_H)\lambda$ sufficiently perturbs the private data Q_i, y_i. Thus the private data cannot be inferred and meanwhile the primal variable will still converge to x^*, as long as there is at least one honest neighbouring node. In what follows we will give a formal proof of this claim.

4.1 Output correctness

As proved in [13], the primal variable $x^{(k+1)}$ is guaranteed to converge to x^* geometrically given arbitrary initialization $x^{(0)}$ and $\lambda^{(0)}$, thereby guaranteeing the output correctness.

4.2 Individual privacy

Now we turn to analyse the individual privacy under both passive and eavesdropping adversaries. Under the passive adversary model, let $\mathcal{V}_c$ and $\mathcal{V}_h$ denote the set of corrupted and honest nodes, respectively. Without loss of generality, assume the passive adversary attempts to infer the private data of honest node $i \in \mathcal{V}_h$. As mentioned earlier, as the only information transmitted from node i after initialization is the primal variable $x_i^{(k+1)}$, the problem thus becomes to analyse how much information about Q_i and y_i would the passive adversary obtain by observing $x_i^{(k+1)}$. Using (D.4) we can express $x_i^{(k+1)}$ as

$$(Q_i^T Q_i + cd_i I)^{-1} \left(\sum_{j \in \mathcal{N}_i \cap \mathcal{V}_h} \left(c x_j^{(k)} - B_{i|j} \left(P^k \Pi_H \lambda^{(k)} \right)_{j|i} \right) \right.$$
$$\left. - \sum_{j \in \mathcal{N}_i \cap \mathcal{V}_h} B_{i|j} \left(P^k (I - \Pi_H) \lambda^{(0)} \right)_{j|i} + Q_i^T y_i + c_p \right), \tag{D.7}$$

where $c_p = \sum_{j \in \mathcal{N}_i \cap \mathcal{V}_c} \left(c x_j^{(k)} - B_{i|j} \lambda_{j|i}^{(k)} \right)$ can be considered constant as it is known by the passive adversary. As $k \to \infty$, x^* will be known and $\Pi_H \lambda^{(k)} \to \lambda^*$ given by (D.5). Thus we conclude that, as long as $\mathcal{N}_i \cap \mathcal{V}_h \neq \emptyset$, we can

perturb the private data by introduce noise in $(I - \Pi_H)\lambda^{(0)}$. More specifically, let $s_i^q = (Q_i^T Q_i + cd_i I)^{-1}$, $s_i^y = Q_i^T y_i$ and $\lambda^{(0)}$ denote realizations of the random variables $\bar{S}_i^q$, $\bar{S}_i^y$ and $\bar{\Lambda}^{(0)}$, respectively. Note that $\bar{\Lambda}^{(0)}$ is independent of both $\bar{S}_i^q$ and $\bar{S}_i^y$ as the initialization of dual variables is independent of the inputs. From (D.7), we can see that the information leakage regarding to Q_i and y_i can be represented by the mutual information [15] $I(\bar{S}_i^q, \bar{X}_i^{(k+1)})$ and $I(\bar{S}_i^y, \bar{X}_i^{(k+1)})$. To analyse both of them we need the following result.

Proposition 2. *Consider the continuous random variables $\{\bar{X}_1, \ldots, \bar{X}_n\}$ having mean and variance $\mu_{\bar{X}_i}$ and $\sigma^2_{\bar{X}_i}$, respectively. Let $\{\bar{Y}_1, \ldots, \bar{Y}_n\}$ be independent random variables independent of $\{\bar{X}_1, \ldots, \bar{X}_n\}$. That is, $I(\bar{X}_i, \bar{Y}_j) = 0$ for all $(i, j) \in \mathcal{V}$. Let $\bar{Z}_i = \bar{X}_i + \bar{Y}_i$ and $\bar{W}_i = \bar{X}_i \bar{Y}_i$, and let $\bar{Z}_i' = \bar{Z}_i / \sigma_{\bar{Z}_i}$ and $\bar{W}_i' = \bar{W}_i / \sigma_{\bar{W}_i}$ be the normalised variables having unit variance. We then have*

$$\lim_{\sigma^2_{\bar{Y}_i} \to \infty} I(\bar{X}_1, \ldots, \bar{X}_n; \bar{Z}_1, \ldots, \bar{Z}_n) = 0,$$
$$\lim_{\sigma^2_{\bar{Y}_i} \to \infty} I(\bar{X}_1, \ldots, \bar{X}_n; \bar{W}_1, \ldots, \bar{W}_n) = 0.$$

Proof.

$$\begin{aligned}
&I(\bar{X}_1, \ldots, \bar{X}_n; \bar{Z}_1, \ldots, \bar{Z}_n) \\
&\quad = h(\bar{Z}_1, \ldots, \bar{Z}_n) - h(\bar{Z}_1, \ldots, \bar{Z}_n | \bar{X}_1, \ldots, X_n) \\
&\quad \overset{\text{(a)}}{=} h(\bar{Z}_1, \ldots, \bar{Z}_n) - h(\bar{Y}_1, \ldots, \bar{Y}_n) \\
&\quad \overset{\text{(b)}}{=} \sum_{i=1}^{n} h\big(\bar{Z}_i | \bar{Z}_1, \ldots, \bar{Z}_{i-1}\big) - \sum_{i=1}^{n} h\big(\bar{Y}_i\big) \\
&\quad \overset{\text{(c)}}{\leq} \sum_{i=1}^{n} h\big(\bar{Z}_i\big) - \sum_{i=1}^{n} h(\bar{Y}_i) \\
&\quad \overset{\text{(d)}}{=} \sum_{i=1}^{n} I(\bar{X}_i; \bar{Z}_i) \\
&\quad \overset{\text{(e)}}{=} \sum_{i=1}^{n} I(\bar{X}_i / \sigma_{\bar{Z}_i}; \bar{Z}_i'),
\end{aligned}$$

where $h(\cdot)$ denotes the differential entropy of the random variable, assuming it exists. Step (a) follows from $h(\bar{Z}_i | \bar{X}_i) = h(\bar{Y}_i)$, (b) follows from the chain rule for differential entropy and the fact that the $\bar{Y}_i$'s are independent random variables, (c) follows from the fact that conditioning decreases entropy, (d) follows from $h\big(\bar{Z}_i\big) - h(\bar{Y}_i) = h(\bar{Z}_i) - h(\bar{Z}_i | \bar{X}_i) = I(\bar{X}_i; \bar{Z}_i)$ and (e) holds as

mutual information is invariant under scaling. As a consequence

$$\lim_{\sigma^2_{\bar{Y}_i}\to\infty}\sum_{i=1}^{n} I(\bar{X}_i;\bar{Z}_i) = \lim_{\sigma_{\bar{Z}_i}\to\infty}\sum_{i=1}^{n} I(\bar{X}_i/\sigma_{\bar{Z}_i};\bar{Z}_i') = \sum_{i=1}^{n} I(0;\bar{Z}_i') = 0.$$

For the case $\bar{W}_i = \bar{X}_i\bar{Y}_i$, we have

$$\begin{aligned} h(\bar{W}_i|\bar{X}_i) &= \int p(\bar{x}_i)h(\bar{W}_i|\bar{X}_i=\bar{x}_i)d\bar{x}_i \\ &= \int p(\bar{x}_i)h(\bar{x}_i\bar{Y}_i|\bar{X}_i=\bar{x}_i)d\bar{x}_i \\ &\overset{\text{(a)}}{=} \int p(\bar{x}_i)h(\bar{Y}_i)d\bar{x}_i = h(\bar{Y}_i), \end{aligned}$$

where (a) holds since the probability measure of the event $\bar{X}_i = 0$ is zero. Hence, the proof of our second claim goes along the same lines as the one presented above, and we conclude that

$$\begin{aligned} &\lim_{\sigma^2_{\bar{Y}_i}\to\infty} I(\bar{X}_1,\ldots,\bar{X}_n;\bar{W}_1,\ldots,\bar{W}_n) \\ &\qquad\leq \lim_{\sigma_{\bar{W}_i}\to\infty}\sum_{i=1}^{n} I(\bar{X}_i/\sigma_{\bar{W}_i};\bar{W}_i') = 0, \end{aligned}$$

thereby proving our claims. □

By applying Proposition 2 to $I(\bar{S}_i^q,\bar{X}_i^{(k+1)})$ and $I(\bar{S}_i^y,\bar{X}_i^{(k+1)})$, we conclude that both mutual information can be made arbitrarily small by increasing the variance of the random variable $(I-\Pi_H)\Lambda^{(0)}$. We thus have both $I(\bar{S}_i^q,\bar{X}_i^{(k+1)})=0$ and $I(\bar{S}_i^y,\bar{X}_i^{(k+1)})=0$ if

$$\exists j\in\mathcal{N}_i\cap\mathcal{V}_h : \operatorname{var}\big(((I-\Pi_H)\Lambda^{(0)})_{j|i}\big)\to\infty. \tag{D.8}$$

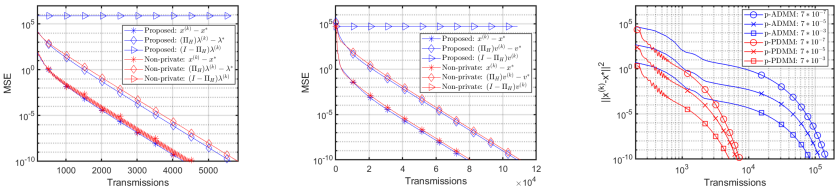
Hence, the proposed approach is able to achieve asymptotically perfect security.

Now we consider an eavesdropping adversary. As we already proved that the transmitted primal variable does not contain information about the private data, the proposed method is also secure against eavesdropping. The communications can therefore be conducted in non-secure channels except for the first iteration where the initialized dual variables $\lambda^{(0)}$ should be communicated through secure channels. The details of the proposed approach are summarised in Algorithm 0.

Several remarks are in place here. Firstly, (D.8) requires $\lambda^{(0)}\cap H^{\perp}\neq\emptyset$. Recall that the non-converging subspace $H^{\perp}$ is non-empty, so that by randomly initializing the dual variables $\lambda^{(0)}$, we have $\lambda^{(0)}\cap H^{\perp}\neq\emptyset$ with probability 1. Secondly, it is important to note that the adversary does not have

Algorithm 5 Privacy-preserving distributed least squares based on PDMM

1: Every node $i \in \mathcal{V}$ initializes its primal variable arbitrarily, and initializes the dual variables with random numbers having sufficiently large variance (specified by the required privacy level).
2: Every node i sends the initialized dual variables $\lambda_{i|j}^{(0)}$ to its neighbours $j \in \mathcal{N}_i$ through securely encrypted channels.
3: **while** $\|x^{(k)} - x^*\|_2 <$ threshold **do**
4: Randomly activate a node, say node i, update its primal variable $x_i^{(k+1)}$ using the x-update in (D.6).
5: Node i broadcasts $x_i^{(k+1)}$ to its neighbours $j \in \mathcal{N}_i$ through non-secure channels.
6: Each neighbour uses $x_i^{(k+1)}$ to update the dual variable $\lambda_{i|j}^{(k+1)}$ based on the λ-update in (D.6).
7: **end while**



(a) (b) (c)

Fig. D.1: Convergence of the primal variable, the converging component and non-converging component of the dual variable for two initializations of (a) PDMM and (b) ADMM. (c) Convergence of the primal variable of the proposed algorithm for ADMM and PDMM for three different privacy levels.

the knowledge of the subspace noise $(I - \Pi_H)\lambda^{(0)}$ as it does not know the converging subspace H, due to the fact that both the total number of nodes and the connectivity between the honest nodes are unknown to the adversary. Thirdly, although we proved that both $I(\bar{S}_i^q, \bar{X}_i^{(k+1)})$ and $I(\bar{S}_i^y, \bar{X}_i^{(k+1)})$ are zero if the inserted noise has infinitely large variance, in practical situation the noise variance will be finite. To quantify the amount of information leakage when dealing with finite variance noise, we consider the simple case of a random variable $\bar{Z} = \bar{X} + \bar{Y}$, where $\bar{X}$ and $\bar{Y}$ are independent Gaussian distributed random variables. For a Gaussian random variable with variance σ^2, the differential entropy is given by $\frac{1}{2}\log(2\pi e\sigma^2)$, so that $I(\bar{X};\bar{Z}) = h(\bar{Z}) - h(\bar{Y}) = \frac{1}{2}\log(1 + \sigma_{\bar{X}}^2/\sigma_{\bar{Y}}^2)$. Hence, the information loss is only 0.007 bits if $\sigma_{\bar{Y}}^2/\sigma_{\bar{X}}^2 = 100$ (the range of $\bar{Y}$ is approximately 10 times the range of $\bar{X}$). Lastly, we note that the proposed approach is also applicable to other distributed optimizers, e.g. ADMM, where the update equations of the dual variables have a similar structure as (D.2) and there also exists a

non-converging subspace. To demonstrate this general applicability, in what follows we will show numerical results for both PDMM and ADMM.

5 Numerical results

We now evaluate the performance of the proposed algorithm by computer simulations. We simulated a random geometric graph with $n = 20$ nodes, and set the wireless transmission radius as $\sqrt{2\frac{\log n}{n}}$ to obtain a connected graph with probability at least $1 - 1/n^2$ [16]. We set $N_i = 20, u = 10$ and generated all the entries of Q and y randomly according to a zero-mean, unit-variance Gaussian distribution.

Fig. D.1a and D.1b show the convergence behaviour of PDMM and ADMM, respectively (mean-squared error versus number of transmissions). The blue lines denote the proposed privacy-preserving approaches (p-PDMM and p-ADMM) where the dual variables are randomly initialized from a Gaussian distribution with variance 1000, while the red lines denote the non-private approaches (n-PDMM and n-ADMM) where the dual variables are initialized within the converging subspace, that is $\lambda^{(0)} \in H$. We can see that both $x^{(k)}$ and $\Pi_H \lambda^{(k)}$ converge to the optimum solution while $\big(I - \Pi_H\big)\lambda^{(k)}$ does not. Note that the lines with red triangle markers are not shown as $\big(I - \Pi_H\big)\lambda^{(k)} = 0$ in this case. Hence, the proposed approach is able to obfuscate the private data while not affecting the output correctness.

To inspect the performance of the proposed approach under different privacy levels, we considered three cases where the variances of the associated dual variables were set at 10, 100, and 1000, which corresponds to an approximated privacy loss of 7×10^{-3}, 7×10^{-5}, and 7×10^{-7} bits, respectively. As shown in Fig. D.1c, for both PDMM and ADMM, the convergence rate is independent of the privacy level (note that the x-axis is on a log scale). This is because the convergence rate of these algorithms only depends on the graph topology and not on the initialization (the initial error does). Therefore, increasing the amount of noise will not affect the convergence rate but only results in a higher initial error.

6 Conclusions

In this paper, we proposed a lightweight yet general convex optimization-based subspace perturbation method to achieve privacy-preserving distributed least squares. In particular, we show that the concerned private data can be protected by inserting noise in a particular subspace determined by the graph topology. The proposed approach is proven secure under both eavesdropping and passive adversaries. More specifically, the individual privacy of any honest node is protected as long as it has one honest neighbour and no securely en-

crypted channels are required except the initialization step. Additionally, it is able to achieve both privacy and accuracy simultaneously, and its convergence rate is independent of the privacy level.

References

[1] G. Giaconi, D. Gündüz, H. V. Poor, "Privacy-aware smart metering: Progress and challenges," *IEEE Signal Process. Mag., vol. 35, no. 6, pp. 59-78*, 2018.

[2] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO, pp. 643–662*. Springer, 2012.

[3] I. Giacomelli, S. Jha, M. Joye, C. D. Page, and K. Yoon, "Privacy-preserving ridge regression with only linearly-homomorphic encryption," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur., pp. 243-261,*, 2018.

[4] A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, and D. Evans, "Privacy-preserving distributed linear regression on high-dimensional data," *in Proc. Priv. Enhancing Technol. no. 4, pp. 345–364*, 2017.

[5] V. Nikolaenko, U.Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, "Privacy-preserving ridge regression on hundreds of millions of records," in *Proc. IEEE Symp. Security Privacy, pp. 334-348*, 2013.

[6] Y.R. Chen, A. Rezapour and W.-G Tzeng, "Privacy-preserving ridge regression on distributed data," *Inf. Sci., vol. 451, pp. 34-49,*, 2018.

[7] K. Tjell, I. Cascudo and R. Wisniewski, "Privacy preserving recursive least squares solutions," in *ECC, pp.3490–3495*, 2019.

[8] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proc. 41st Annu. ACM Symp. Theory Comput., pp. 371-380*, 2009.

[9] D. Sarwate and K. Chaudhuri, "Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data," *IEEE Signal Process. Magazine, vol. 30, no. 5, pp. 86–94*, 2013.

[10] K. H. Degue and J. L. Ny, "On differentially private kalman filtering," in *Proc. IEEE Global Conf. Signal Inf. Process., pp. 487-491*, 2017.

[11] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica, vol. 81, pp. 221–231*, 2017.

[12] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process., vol. 4, no. 1, pp. 173–187*, 2018.

[13] T. Sherson, R. Heusdens, W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 2, pp 334-347*, 2018.

[14] D. Dolev, C. Dwork, O. Waarts, M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach., vol. 40, no. 1, pp. 17-47,*, 1993.

[15] T. M. Cover and J. A. Tomas, *Elements of information theory*, John Wiley & Sons, 2012.

[16] J. Dall and M. Christensen, "Random geometric graphs," *Physical review E, vol. 66, no. 1, pp. 016121*, 2002.

Paper E

Privacy-Preserving Distributed Optimization via Subspace Perturbation: A General Framework

Qiongxiu Li, Richard Heusdens, and Mads Græsbøll Christensen

The paper has been published in the
IEEE/ACM Transactions on Signal Processing, 2020

The layout has been revised.

Abstract

As the modern world becomes increasingly digitized and interconnected, distributed signal processing has proven to be effective in processing its large volume of data. However, a main challenge limiting the broad use of distributed signal processing techniques is the issue of privacy in handling sensitive data. To address this privacy issue, we propose a novel yet general subspace perturbation method for privacy-preserving distributed optimization, which allows each node to obtain the desired solution while protecting its private data. In particular, we show that the dual variable introduced in each distributed optimizer will not converge in a certain subspace determined by the graph topology. Additionally, the optimization variable is ensured to converge to the desired solution, because it is orthogonal to this non-convergent subspace. We therefore propose to insert noise in the non-convergent subspace through the dual variable such that the private data are protected, and the accuracy of the desired solution is completely unaffected. Moreover, the proposed method is shown to be secure under two widely-used adversary models: passive and eavesdropping. Furthermore, we consider several distributed optimizers such as ADMM and PDMM to demonstrate the general applicability of the proposed method. Finally, we test the performance through a set of applications. Numerical tests indicate that the proposed method is superior to existing methods in terms of several parameters like estimated accuracy, privacy level, communication cost and convergence rate.

1 Introduction

In a world of interconnected and digitized systems, new and innovative signal processing tools are needed to take advantage of the sheer scale of information/data. Such systems are often characterized by "big data". Another central aspect of such systems is their distributed nature, in which the data are usually located in different computational units that form a network. In contrast to the traditional centralized systems, in which all the data must be firstly collected from different units and then processed at a central server, distributed signal processing circumvents this limitation by utilizing the network nature. That is, instead of relying on a single centralized coordination, each node/unit is able to collect information from its neighbors and also to conduct computation on a subset of the overall networked data. This distributed processing has many advantages, such as allowing for flexible scalability of the number of nodes and robustness to dynamical changes in the graph topology. Currently, the computational unit/node in distributed systems is usually limited in resources, as tablets and phones become the primary computing devices used by many people [1, 2]. These devices often contain sensors that can use wireless communication to form so-called ad-hoc networks. Therefore, these devices can collaborate in solving problems by sharing computational resources and sensor data. However, the information collected from sensors such as GPS, cameras

and microphones often includes personal data, thus posing a major concern, because such data are private in nature.

There has been a considerable growth of optimization techniques in the field of distributed signal processing, as many traditional signal processing problems in distributed systems can be equivalently formed as convex optimization problems. Owing to the general applicability and flexibility of distributed optimization, optimization has emerged in a wide range of applications such as acoustic signal processing [3, 4], control theory [5] and image processing [6]. Typically, the paradigm of distributed optimization is to separate the global objective function over the network into several local objective functions, which can be solved for each node through exchanging data only with its neighbors. This data exchange is a major concern regarding privacy, because the exchanged data usually contain sensitive information, and traditional distributed optimization schemes do not address this privacy issue. Therefore, how to design a distributed optimizer for processing sensitive data, is a challenge to be overcome in the field.

1.1 Related works

To address the privacy issues in distributed optimization, the literature has mainly used techniques from differential privacy [7, 8] and secure multiparty computation (SMPC) [9]. Differential privacy is one of the most commonly used non-cryptographic techniques for privacy preservation, because it is computationally lightweight, and it also uses a strict privacy metric to quantify that the posterior guess of the private data is only slightly better than the prior (quantified by a small positive number ϵ). This method of protecting private data has been applied in [10–16] through carefully adding noise to the exchanged states or objective functions. However, this noise insertion mechanism involves an inherent trade-off between the privacy and the accuracy of the optimization outputs. Additionally, some approaches [17–19] have applied differential privacy with the help of a trusted third party (TTP) like a server/cloud. However, requiring a TTP for coordination makes the protocol not completely decentralized (i.e., peer-to-peer setting). Consequently, it thus hinders use in many applications such as wireless sensor networks in which centralized coordinations are unavailable.

SMPC, in contrast, has been widely used in distributed processing, because it provides cryptographic techniques to ensure privacy in a distributed network. More specifically, it aims to compute the result of a function of a number of parties' private data while protecting each party's private data from being revealed to others. Examples of how to preserve privacy by using SMPC have been applied in [20–24], in which partially homomorphic encryption (PHE) [25] was used to conduct computations in the encrypted domain. However, PHE requires the assistance of a TTP and thus cannot be directly applied in a fully decentralized setting. Additionally, although PHE is more computationally lightweight than other encryption techniques, such as fully homomorphic encryption [26]

and Yao's garbled circuit [27, 28], it is more computationally demanding than the noise insertion techniques, such as differential privacy, because it relies on the computational hardness assumption. To alleviate the bottleneck of computational complexity, another technique in SMPC, called secret sharing [29], has become a popular alternative for distributed processing, because its computational cost is comparable to that of differential privacy. It has been applied in [30] to preserve privacy by splitting sensitive data into pieces and sending them to the so-called computing parties afterward. However, secret sharing generally is expensive in terms of communication cost, because it requires multiple communication rounds for each splitting process.

1.2 Paper contributions

The main contribution of this paper is that we propose a novel subspace perturbation method, which circumvents the limitations of both the differential privacy and SMPC approaches for distributed signal processing. We propose to insert noise in the subspace such that not only the private data is protected from being revealed to others but also the accuracy of results is not affected. The proposed subspace perturbation method has several attractive properties:

- Compared to differential privacy based approaches, the proposed approach is ensured to converge to the optimum results without compromising privacy. Additionally, it is defined in a completely decentralized setting, because no central aggregator is required.
- In contrast to SMPC based approaches, the proposed approach is efficient in both computation and communication. Because it does not require complex encryption functions (such as those involved in PHE), and it does not have high communication costs (such as those required in the secret sharing approaches).
- The proposed subspace perturbation method is generally applicable to many distributed optimization algorithms like ADMM, PDMM or the dual ascent method.
- The convergence rate of the proposed method is invariant with respect to the amount of inserted noise and thus to the privacy level.

We published preliminary results in [31, 32] where the main concept of subspace perturbation was introduced using PDMM based on a specific application. Here we give a more complete analysis of the proposed subspace perturbation in a broader context, i.e., for all convex problems, and further generalize it into other optimizers such as ADMM and dual ascent.

1.3 Outline and notation

The remainder of this paper is organized as follows: Section 2 reviews distributed convex optimization and some important concepts for privacy preservation. Section 3 defines the problem to be solved and provides qualitative metrics to evaluate the performance. Section 4 introduces the primal-dual method of multipliers (PDMM), explaining its key properties used in the proposed approach. Section 5 introduces the proposed subspace perturbation method based on the PDMM. Section 6 shows the general applicability of the proposed method by considering other types of distributed optimizers, such as ADMM and the dual ascent method. In Section 7 the proposed approach is applied to a wide range of applications including distributed average consensus, distributed least squares and distributed LASSO. Section 8 demonstrates the numerical results for each application and compares the proposed method with existing approaches. Finally, Section 9 concludes the paper.

The following notations are used in this paper. Lowercase letters (x), lowercase boldface letters $(\boldsymbol{x})$, uppercase boldface letters $(\boldsymbol{X})$, overlined uppercase letters $(\bar{X})$ and calligraphic letters $(\mathcal{X})$ denote scalars, vectors, matrices, subspaces and sets, respectively. An uppercase letter (X) denotes the random variable of its lowercase argument, which means that the lowercase letter x is assumed to be a realization of random variable X. $\mathrm{null}\{\cdot\}$ and $\mathrm{span}\{\cdot\}$ denote the nullspace and span of their argument, respectively. $(\boldsymbol{X})^\dagger$ and $(\boldsymbol{X})^\top$ denote the Moore-Penrose pseudo inverse and transpose of $\boldsymbol{X}$, respectively. $\boldsymbol{x}_i$ denotes the i-th entry of the vector $\boldsymbol{x}$, and $\boldsymbol{X}_{ij}$ denotes the (i,j)-th entry of the matrix $\boldsymbol{X}$. $\mathbf{0}$, $\mathbf{1}$ and $\boldsymbol{I}$ denote the vectors with all zeros and all ones, and the identity matrix of appropriate size, respectively.

2 Fundamentals

In this section, we review the fundamentals and some important concepts related to privacy preservation. We first review the distributed convex optimization and highlight its privacy concerns. Then we describe the adversary models that will be addressed later in this paper.

2.1 Distributed convex optimization

A distributed network is usually modeled as a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \{1, 2, ..., n\}$ is the set of nodes, and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of edges. Let $n = |\mathcal{N}|$ and $m = |\mathcal{E}|$ denote the numbers of nodes and edges, respectively. $\mathcal{N}_i = \{j \,|\, (i,j) \in \mathcal{E}\}$ denotes the neighborhood of node i, and $d_i = |\mathcal{N}_i|$ denotes the degree of node i.

Let $\boldsymbol{x}_i \in \mathbb{R}^{u_i}$ denote the local optimization variable at node i. A standard constrained convex optimization problem over the network can then be

expressed as

$$\begin{aligned} \min_{\boldsymbol{x}_i} \quad & \sum_{i\in\mathcal{N}} f_i(\boldsymbol{x}_i) \\ \text{s.t.} \quad & \boldsymbol{B}_{i|j}\boldsymbol{x}_i + \boldsymbol{B}_{j|i}\boldsymbol{x}_j = \boldsymbol{b}_{i,j}\ \forall (i,j)\in\mathcal{E} \end{aligned} \tag{E.1}$$

where $f_i : \mathbb{R}^{u_i} \mapsto \mathbb{R}\cup\{\infty\}$ denote the local objective function at node i, which we assume to be convex for all nodes $i \in \mathcal{N}$. Additionally, let $v_{i,j}$ denote the dimension of constraints at each edge $(i,j) \in \mathcal{E}$, $\boldsymbol{B}_{i|j} \in \mathbb{R}^{v_{i,j}\times u_i}$, $\boldsymbol{b}_{i,j} \in \mathbb{R}^{v_{i,j}}$ are defined for the constraints. Note that we distinct the subscripts $i|j$ and i,j, where the former is a directed identifier that denotes the directed edge from node i to j and the later i,j is an undirected identifier. Stacking all the local information and let $N_n = \sum_{i\in\mathcal{N}} u_i$ and $M_m = \sum_{(i,j)\in\mathcal{E}} v_{i,j}$, we can compactly express (E.1) as

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{B}\boldsymbol{x} = \boldsymbol{b} \end{aligned}\ , \tag{E.2}$$

where $f : \mathbb{R}^{N_n} \mapsto \mathbb{R}\cup\{\infty\}$, $\boldsymbol{x} \in \mathbb{R}^{N_n}$, $\boldsymbol{B} \in \mathbb{R}^{M_m\times N_n}$, $\boldsymbol{b} \in \mathbb{R}^{M_m}$. For simplicity, we assume the dimension of $\boldsymbol{x}_i$ of all nodes are the same and set it as u, i.e., $u = u_i, \forall i \in \mathcal{N}$, all $\boldsymbol{B}_{i|j}$ be square matrices, i.e., $v_{i,j} = u_i = u, \forall (i,j) \in \mathcal{E}$, and the constraints be zeros, i.e., $\boldsymbol{b} = \boldsymbol{0}$. We thus have $M_m = m \times u$ and $N_n = n \times u$. We further define matrix $\boldsymbol{B}$ based on the incidence matrix of the graph: $\boldsymbol{B}_{i|j} = \boldsymbol{I}$, $\boldsymbol{B}_{j|i} = -\boldsymbol{I}$ if and only if $(i,j) \in \mathcal{E}$ and $i < j$ and $\boldsymbol{B}_{i|j} = -\boldsymbol{I}$, $\boldsymbol{B}_{j|i} = \boldsymbol{I}$ if and only if $(i,j) \in \mathcal{E}$ and $i > j$. Note that $\boldsymbol{B}$ reduces to the graph incidence matrix if $u = 1$.

To solve the above problem without any centralized coordination, several distributed optimizers have been proposed, such as ADMM [33] and PDMM [34, 35], to iteratively solve the problem by communicating only with the local neighborhood. That is, at each iteration (denoted by index k), each node i updates its optimization variable $\boldsymbol{x}_i^{(k)}$ by exchanging data only with its neighbors. The goal of distributed optimizers is to design certain updating functions to ensure that $\boldsymbol{x}_i^{(k)} \to \boldsymbol{x}_i^*$, where $\boldsymbol{x}_i^*$ denotes the optimum solution for node i.

2.2 Privacy concerns

As mentioned in the introduction, the sensor data captured by an individual's device are usually private in nature. For example, health conditions like Parkinson's disease can be detected by voice signals [36, 37], and activities of householders can be revealed by power consumption data [38]. In the context of distributed optimization, such private information regarding each node i is contained in its local objective function $f_i(\boldsymbol{x}_i)$ [12]. Recall that after each iteration, node i will send the updated optimization variable $\boldsymbol{x}_i^{(k+1)}$ to all of its neighbors. Since this variable is related to $f_i(\boldsymbol{x}_i)$, the revealed $\boldsymbol{x}_i^{(k+1)}$ leaks information about $f_i(\boldsymbol{x}_i)$, e.g., its subgradient $\partial f_i(\boldsymbol{x}_i)$, thereby violating privacy. This privacy concern, however, has not been addressed in existing distributed

optimizers. Therefore, in this paper, we attempt to investigate this privacy issue and propose a general solution to achieve privacy-preserving distributed optimization.

2.3 Adversary model

When designing a privacy-preserving algorithm, it is important to determine the adversary model that qualifies its robustness under various types of security attack. By colluding with a number of nodes, the adversary aims to conduct certain malicious behaviors, such as learning private data or manipulating the function result to be incorrect. These colluding and non-colluding nodes are referred to as corrupted nodes and honest nodes, respectively. Most of the literature has considered only a passive (also called honest-but-curious or semi-honest) adversary, where the corrupted nodes are assumed to follow the instructions of the designed protocol, but are curious about the honest nodes' private data. Another common adversary is the external eavesdropping adversary, which is assumed to infer the private data of the honest nodes by eavesdropping all the communication channels in the network. The eavesdropping adversary in the context of privacy-preserving distributed optimization is relatively unexplored. In fact, many SMPC based approaches, such as secret sharing [30, 39, 40], assume that all messages are transmitted through securely encrypted channels [41], such that the communication channels cannot be eavesdropped upon. However, channel encryption is computationally demanding and is therefore very expensive for iterative algorithms, such as those used here, because they require use of communication channels between nodes many times. In this paper, we design the privacy-preserving distributed optimizers in an efficient way, such that the channel encryption needs to be used only once.

3 Problem definition

Given the above-mentioned fundamentals, we thus conclude that the goal of privacy-preserving distributed convex optimization is to jointly optimize the constrained convex problem while protecting the private data of each node from being revealed under defined adversary models. More specifically, there are two requirements that should be satisfied simultaneously:

1) Output correctness: at the end of the algorithm, each node i obtains its optimum solution $\boldsymbol{x}_i^*$ and its correct function result $f_i(\boldsymbol{x}_i^*)$, which implies that the global function result $f(\boldsymbol{x}^*)$ has been also achieved.

2) Individual privacy: throughout the execution of the algorithm, the private data, i.e., the information regarding $f_i(\boldsymbol{x}_i)$, held by each honest node should be protected against both passive and eavesdropping adversaries; except for the information that can be directly inferred from the

knowledge of the function output and the private data of the corrupted nodes (in Section 7 we will explain this in detail).

To quantify the above requirements, two metrics must be defined.

3.1 Output correctness metric

For each node i, achieving the optimum solution $\boldsymbol{x}_i^*$ implies obtaining the correct function output $f_i(\boldsymbol{x}_i^*)$ as well. To measure the output correctness for the whole network in terms of the amount of communication, we thus use the mean squared error $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_2^2$ over all the nodes as a function of number of transmission: one transmission denotes that one message package is transmitted from one node to another.

3.2 Individual privacy metric

In the literature, information-theoretic measures like mutual information and ϵ-differential privacy are often adopted as the privacy metric (see [42] for details). In this paper, we deploy mutual information as the metric for quantifying the individual privacy. The reason of choosing mutual information over ϵ-differential privacy is because ϵ-differential privacy corresponds to a worst-case metric and the worst-case privacy leakage can in practice be quite far from the typical leakage of the average user [43]. Notably, mutual information and ϵ-differential privacy are closely related with each other and it is shown in [44] that Bayesian mutual information is a relaxation of ϵ-differential privacy.

Given a continuous random variable X with a probability density function f_X, the differential entropy of X is defined as $h(X) = -\int f(x) \log f(x) dx$. Let Y be another random variable, the conditional entropy $h(X|Y)$ quantifies the uncertainty of X after knowing Y. The mutual information $I(X;Y)$ [45] measures the amount of information learned about X by observing Y, or vice versa, which is given by[1]

$$I(X;Y) = h(X) - h(X|Y). \tag{E.3}$$

3.3 Lower bound of information leakage

When defining the individual privacy, we explicitly exclude the information that can be deduced from the function output and the private data of the corrupted nodes, because each node will eventually obtain its function output from the algorithm, and in some cases, this output may contain certain information regarding the private data held by the individual honest node. To explain this scenario more explicitly, take the distributed average consensus as an example. Let $\mathcal{N}_c$ and $\mathcal{N}_h$ denote the set of corrupted and honest nodes,

[1] In the case of discrete random variables, the condition is expressed in terms of the Shannon entropy $H(\cdot)$

respectively. A group of n people would like to compute their average salary, denoted by s_{ave}, while keeping each person's salary $\boldsymbol{s}_i$ unknown to the others. If the average result is accurate, the salary sum of the honest people can always be computed by $\sum_{i\in\mathcal{N}_h} \boldsymbol{s}_i = n \times s_{\text{ave}} - \sum_{i\in\mathcal{N}_c} \boldsymbol{s}_i$ assuming the adversary knows n, regardless of the underlying algorithms. With the mutual information metric, the salary sum will leak $I(S_i; \sum_{j\in\mathcal{N}_h} S_j)$ amount of information about the salary of the honest node i. Provided that this information leakage is unavoidable, we therefore refer to it as the lower bound of information leakage. We now give a definition of perfect security in the context of distributed processing.

Definition 1. *(Perfect privacy-preserving algorithms.) Given a specific application, let $\delta \geq 0$ denote its lower bound of information leakage. A privacy-preserving algorithm is considered perfect (or achieves perfect security) as long as it reveals no more information than this lower bound δ.*

4 Primal-dual method of multipliers

To introduce the main idea of subspace perturbation, we first use PDMM as an example and then generalize it to other distributed optimizers in Section 6, like ADMM or dual ascent. The main reasons for choosing PDMM over other optimizers are its general applicability and its broadcasting property (see Section 4.2) which allows for simplification of the individual privacy analysis. In this section, we first provide a review of the fundamentals of the PDMM and introduce its main properties, which will be used later in the proposed approach.

4.1 Fundamentals

PDMM is an instance of Peaceman-Rachford splitting of the extended dual problem (refer to [35] for details). It is an alternative distributed optimization tool to ADMM for solving constrained convex optimization problems and is often characterized by a faster convergence rate [34]. For the distributed optimization problem stated in (E.1), the extended augmented Lagrangian of PDMM is given by

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + (\boldsymbol{P}\boldsymbol{\lambda}^{(k)})^\top \boldsymbol{C}\boldsymbol{x} + \frac{c}{2}\|\boldsymbol{C}\boldsymbol{x} + \boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(k)}\|_2^2, \tag{E.4}$$

and the updating equations of PDMM are given by

$$\boldsymbol{x}^{(k+1)} = \arg\min_{\boldsymbol{x}} L\left(\boldsymbol{x}, \boldsymbol{x}^{(k)}, \boldsymbol{\lambda}^{(k)}\right), \tag{E.5}$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{P}\boldsymbol{\lambda}^{(k)} + c(\boldsymbol{C}\boldsymbol{x}^{(k+1)} + \boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(k)}), \tag{E.6}$$

where $\boldsymbol{P} \in \mathbb{R}^{2M_m \times 2M_m}$ is a symmetric permutation matrix exchanging the first M_m with the last M_m rows of the matrix it applies, $c > 0$ is a constant controlling the convergence rate. $\boldsymbol{\lambda}^{(k)} \in \mathbb{R}^{2M_m}$ denotes the dual variable at iteration k, introduced for controlling the constraints. Each edge $(i,j) \in \mathcal{E}$ is related to two dual variables $\boldsymbol{\lambda}_{i|j}, \boldsymbol{\lambda}_{j|i} \in \mathbb{R}^u$, controlled by node i and j, respectively. Additionally, $\boldsymbol{C} \in \mathbb{R}^{2M_m \times N_n}$ is a matrix related to $\boldsymbol{B}$: $\boldsymbol{C} = [\boldsymbol{B}_+^\top \ \boldsymbol{B}_-^\top]^\top$, where $\boldsymbol{B}_+$ and $\boldsymbol{B}_-$ are the matrices containing only the positive and negative entries of $\boldsymbol{B}$, respectively. Of note, $\boldsymbol{C}+\boldsymbol{P}\boldsymbol{C} = [\boldsymbol{B}^\top \ \boldsymbol{B}^\top]^\top$ and $\forall (i,j) \in \mathcal{E} : \boldsymbol{\lambda}_{j|i} = (\boldsymbol{P}\boldsymbol{\lambda})_{i|j}$.

4.2 Broadcast PDMM

On the basis of (E.5), the local updating function at each node i is given by

$$\boldsymbol{x}_i^{(k+1)} = \arg\min_{\boldsymbol{x}_i} \left(f_i(\boldsymbol{x}_i) + \sum_{j \in \mathcal{N}_i} \boldsymbol{\lambda}_{j|i}^{(k)\top} \boldsymbol{B}_{i|j} \boldsymbol{x}_i + \frac{c}{2} \sum_{j \in \mathcal{N}_i} \| \boldsymbol{B}_{i|j} \boldsymbol{x}_i + \boldsymbol{B}_{j|i} \boldsymbol{x}_j^{(k)} \|_2^2 \right) \tag{E.7}$$

$$\forall j \in \mathcal{N}_i : \boldsymbol{\lambda}_{i|j}^{(k+1)} = \boldsymbol{\lambda}_{j|i}^{(k)} + c\big(\boldsymbol{B}_{i|j} \boldsymbol{x}_i^{(k+1)} + \boldsymbol{B}_{j|i} \boldsymbol{x}_j^{(k)}\big). \tag{E.8}$$

We can see that updating $\boldsymbol{\lambda}_{i|j}^{(k+1)}$ requires only $\boldsymbol{\lambda}_{j|i}^{(k)}, \boldsymbol{x}_j^{(k)}$ and $\boldsymbol{x}_i^{(k+1)}$, of which $\boldsymbol{\lambda}_{j|i}^{(k)}$ and $\boldsymbol{x}_j^{(k)}$ are already available at node j. Thus, node i needs to broadcast only $\boldsymbol{x}_i^{(k+1)}$ after which the neighboring nodes can update $\boldsymbol{\lambda}_{i|j}^{(k+1)}$ themselves. As a consequence, the dual variables do not need to be transmitted at all, except for the initialization step.

More specifically, each node i, regarding each iteration k, has the following knowledge:

$$\{\boldsymbol{x}_i^{(k)}, \boldsymbol{\lambda}_{i|j}^{(k)}\}_{j \in \mathcal{N}_i} \cup \{\boldsymbol{x}_j^{(k)}, \boldsymbol{\lambda}_{j|i}^{(k)}\}_{j \in \mathcal{N}_i}, \tag{E.9}$$

where the first term represents the local variables of node i, and the latter is the variables of its neighbors that are related to node i. Note that the optimization variables $\{\boldsymbol{x}_j^{(k)}\}_{j \in \mathcal{N}_i}$ are sent by the neighbouring nodes during the (iterative) optimization process whereas the dual variables $\{\boldsymbol{\lambda}_{j|i}^{(k)}\}_{j \in \mathcal{N}_i}$ are computed and kept locally at node i, except for the initialization step ($k = 0$) where these variables need to be exchanged through securely encrypted channels.

4.3 Convergence of dual variables

Consider two successive $\boldsymbol{\lambda}$-updates in (E.6), in which we have

$$\boldsymbol{\lambda}^{(k+2)} = \boldsymbol{\lambda}^{(k)} + c(\boldsymbol{C}\boldsymbol{x}^{(k+2)} + 2\boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(k+1)} + \boldsymbol{C}\boldsymbol{x}^{(k)}), \tag{E.10}$$

as $\boldsymbol{P}^2 = \boldsymbol{I}$. Let $\bar{H}_p = \text{span}(\boldsymbol{C}) + \text{span}(\boldsymbol{PC})$ and $\bar{H}_p^{\perp} = \text{null}(\boldsymbol{C}^\top) \cap \text{null}((\boldsymbol{PC})^\top)$. Denote $\Pi_{\bar{H}_p}$ as the orthogonal projection onto $\bar{H}_p$. From (E.10), we conclude that every two $\boldsymbol{\lambda}$-updates affect only $\Pi_{\bar{H}_p}\boldsymbol{\lambda} \in \bar{H}_p$, and $(\boldsymbol{I} - \Pi_{\bar{H}_p})\boldsymbol{\lambda} \in \bar{H}_p^{\perp}$ remains the same. Moreover, as shown in [35], $\left(\boldsymbol{I} - \Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}$ will only be permuted in each iteration and $\Pi_{\bar{H}_p}\boldsymbol{\lambda}$ will eventually converge to $\boldsymbol{\lambda}^*$ given by

$$\boldsymbol{\lambda}^* = -\begin{pmatrix} \boldsymbol{C}^\top \\ (\boldsymbol{PC})^\top \end{pmatrix}^{\dagger} \begin{pmatrix} \partial f(\boldsymbol{x}^*) + c\boldsymbol{C}^\top \boldsymbol{C}\boldsymbol{x}^* \\ \partial f(\boldsymbol{x}^*) + c\boldsymbol{C}^\top \boldsymbol{PC}\boldsymbol{x}^* \end{pmatrix} + c\boldsymbol{C}\boldsymbol{x}^*. \tag{E.11}$$

We thus can separate the dual variable into two parts:

$$\begin{aligned} \boldsymbol{\lambda}^{(k)} &= \Pi_{\bar{H}_p}\boldsymbol{\lambda}^{(k)} + (\boldsymbol{I} - \Pi_{\bar{H}_p})\boldsymbol{\lambda}^{(k)}, \\ &\to \boldsymbol{\lambda}^* + \boldsymbol{P}^k \left(\boldsymbol{I} - \Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}^{(0)} \end{aligned} \tag{E.12}$$

since $\boldsymbol{P}^2 = \boldsymbol{I}$. Below, $\bar{H}_p$ and $\bar{H}_p^{\perp}$ are respectively referred to as the convergent subspace and non-convergent subspace associated with PDMM, and similarly $\Pi_{\bar{H}_p}\boldsymbol{\lambda}$ and $\left(\boldsymbol{I} - \Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}$ are called the convergent and non-convergent component of the dual variable, respectively.

5 Proposed approach using PDMM

Having introduced the PDMM algorithm, we now introduce the proposed approach. To achieve a computationally and communicationally efficient solution for privacy preservation, one of the most used techniques is obfuscation by inserting noise, such as those used in differential privacy based approaches. However, inserting noise usually compromises the function accuracy, because the updates are disturbed by noise. To alleviate this trade-off, we propose to insert noise in the non-convergent subspace only so that the accuracy of the optimization solution is not affected (see also Remark 4), thus achieving both privacy and accuracy at the same time. The proposed noise insertion method is referred to as subspace perturbation. Below, we first present some information-theoretic results regarding privacy, after which we explain the proposed subspace perturbation in detail and prove that it satisfies both the output correctness and individual privacy requirements stated in Section 3.

5.1 Privacy preservation using noise insertion

We first present the following information theoretic results regarding privacy, which serve as fundamentals for the proposed approach.

Proposition 3. *Let $\{X_i\}_{i=1,\dots,n}$ denote a set of continuous random variables with mean and variance μ_{X_i} and $\sigma^2_{X_i}$, respectively, assuming they exist. Let $\{Y_i\}_{i=1,\dots,n}$ be a set of mutually independent random variables, i.e., $I(Y_i, Y_j) =$*

$0, i \neq j$, *which is independent of* $\{X_i\}_{i=1,\ldots,n}$, *i.e.,* $I(X_i, Y_j) = 0$ *for all* $i, j \in \mathcal{N}$. *Let* $Z_i = X_i + Y_i$ *and let* $Z_i' = Z_i/\sigma_{Z_i}$ *be the normalized random variable with unit variance. We have*

$$\lim_{\sigma^2_{Y_i} \to \infty} I(X_1, \ldots, X_n; Z_1, \ldots, Z_n) = 0, \tag{E.13}$$

Proof. See Appendix 10.1. □

Proposition 3 states that, if the lower bound of information leakage $\delta = 0$, we have perfect security if the variance of the inserted noise goes to infinity. That is, the system is asymptotically optimal. In the context of distributed signal processing, however, δ is usually positive as the optimum solution is often an aggregation of the local information of all nodes. In these cases, perfect security can be achieved through finite noise insertion. The following result gives a lower bound on the noise variance for guaranteeing perfect security.

Proposition 4. *Consider two independent random variables* X *and* Y *with variance* σ_X^2 *and* σ_Y^2, *where* X *denotes the private data and* Y *denotes the inserted noise for protecting* X. *Let* $Z = X + Y$. *Given* $\delta > 0$, *if we choose to insert Gaussian noise, we can obtain*

$$I(X; Z) \leq \delta$$

as long as

$$\sigma_Y^2 \geq \frac{\sigma_X^2}{2^{2\delta} - 1}, \tag{E.14}$$

Proof. See Appendix 10.2. □

Proposition 4 provides a simple way to set the noise variance for achieving perfect security. As an example, if $\delta = 7 \times 10^{-2}$ bits, then perfect security can be guaranteed by setting the variance of the inserted Gaussian noise to be 10 times that of the variance of the private data.

5.2 Subspace perturbation

We first give the following assumption.

Assumption 1. *The communication channels in the network are securely encrypted when transmitting the initialized dual variable* $\boldsymbol{\lambda}^{(0)}$.

Because of the broadcasting property of the PDMM, after transmission of the initialized dual variables, the updated optimization variable $\boldsymbol{x}_i^{(k+1)}$ is the only

information transmitted in the network at each iteration. Based on (E.7), $\boldsymbol{x}_i^{(k+1)}$ is computed by [2]

$$\mathbf{0} \in \partial f_i(\boldsymbol{x}_i^{(k+1)}) + \sum_{j\in\mathcal{N}_i} \boldsymbol{B}_{i|j}\boldsymbol{\lambda}_{j|i}^{(k)} + c\sum_{j\in\mathcal{N}_i}(\boldsymbol{x}_i^{(k+1)} - \boldsymbol{x}_j^{(k)}). \tag{E.15}$$

We can see that the information about the local objective function $f_i(\boldsymbol{x}_i^{(k+1)})$ is contained in the subgradient $\partial f_i(\boldsymbol{x}_i^{(k+1)})$. The main goal of privacy preservation thus becomes minimizing the information loss about $\partial f_i(\boldsymbol{x}_i^{(k+1)})$ given the information known by the adversary. To do so, we first need to analyze how much knowledge the adversary knows regarding (E.15). Based on the defined adversary models, there are two ways for the adversary to collect information. The first way is to eavesdrop the communication channels. By doing so, the adversary is able to collect, at every iteration $k \geq 0$, all optimization variables $\{\boldsymbol{x}_i^{(k)}\}_{i\in\mathcal{N}}$ in the network. The dual variables $\{\boldsymbol{\lambda}^{(k)}\}$, on the other hand, cannot be eavesdropped because they are only transmitted during the initialization phase ($k = 0$) through securely encrypted channels (see Assumption 1). The second way is to collect the information of all corrupted nodes (a passive adversary model), which is given by (E.9) for every $i \in \mathcal{N}_c$. Combining the above information together, we conclude that the adversary has the following knowledge regarding $\boldsymbol{x}$ and $\boldsymbol{\lambda}$:

$$\{\boldsymbol{x}_i^{(k)}\}_{i\in\mathcal{N}} \cup \{\boldsymbol{\lambda}_{i|j}^{(k)}\}_{(i,j)\in\mathcal{E}_c}, \tag{E.16}$$

where $\mathcal{E}_c = \{(i,j) : (i,j) \in \mathcal{E}, (i,j) \notin \mathcal{N}_h \times \mathcal{N}_h\}$ denotes the corrupted edge set. Let $\mathcal{N}_{i,c} = \mathcal{N}_i \cap \mathcal{N}_c$ and $\mathcal{N}_{i,h} = \mathcal{N}_i \cap \mathcal{N}_h$ denote the corrupted and honest neighborhood of node i, respectively. By inspecting (E.15), we can see that with the knowledge (E.16), the adversary is able to compute both $c\sum_{j\in\mathcal{N}_i}(\boldsymbol{x}_i^{(k+1)} - \boldsymbol{x}_j^{(k)})$ and the partial sum contributed by the corrupted neighborhood of node i, i.e., $\sum_{j\in\mathcal{N}_{i,c}} \boldsymbol{B}_{i|j}\boldsymbol{\lambda}_{j|i}^{(k)}$. Therefore, after deducing the known terms from (E.15), what the adversary observes is

$$\partial f_i(\boldsymbol{x}_i^{(k+1)}) + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\boldsymbol{\lambda}_{j|i}^{(k)}. \tag{E.17}$$

In order to achieve perfect security (see Definition 1), the information loss at every iteration should not exceed δ, i.e.,

$$I\left(\partial f_i(X_i^{(k+1)})\,;\, \partial f_i(X_i^{(k+1)}) + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\Lambda_{j|i}^{(k)}\right) \leq \delta. \tag{E.18}$$

Note that the lower bound δ is given by the application. The only freedom we have, in order to obtain perfect security, is to control the variance of $\sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\Lambda_{j|i}^{(k)}$.

[2] Note that $\boldsymbol{B}_{i|j}^\top = \boldsymbol{B}_{i|j}$, $\boldsymbol{B}_{i|j}\boldsymbol{B}_{j|i} = -\boldsymbol{I}$, and $\boldsymbol{B}_{i|j}\boldsymbol{B}_{i|j} = \boldsymbol{I}$.

Using (E.12), we can express (E.17) as

$$\partial f_i(\boldsymbol{x}_i^{(k+1)}) + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\left(\Pi_{\bar{H}_p}\boldsymbol{\lambda}^{(k)}\right)_{j|i} + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\left(\boldsymbol{P}^k\left(\boldsymbol{I}-\Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}^{(0)}\right)_{j|i}, \tag{E.19}$$

from which we conclude that the variance of the convergent term $\sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\left(\Pi_{\bar{H}_p}\boldsymbol{\lambda}^{(k)}\right)_{j|i}$ can not be manipulated to be large as it will always converge since $\Pi_{\bar{H}_p}\boldsymbol{\lambda}^{(k)} \to \boldsymbol{\lambda}^*$. On the contrary, the variance of the non-convergent term $\sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\left(\boldsymbol{P}^k\left(\boldsymbol{I}-\Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}^{(0)}\right)_{j|i}$ can be made arbitrarily large as it only depends on the initialization of the dual variable. As a consequence, we propose to exploit this non-convergent term to guarantee perfect security. That is, given an arbitrary $\delta > 0$, we can adjust the variance of $\sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\left(\boldsymbol{P}^k\left(\boldsymbol{I}-\Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}^{(0)}\right)_{j|i}$ such that (E.18) is satisfied. In particular, by applying Proposition 4 to the problem at hand, we conclude that a sufficient condition to guarantee perfect security (E.18) is to initialize the dual variable with Gaussian distributed noise with

$$\exists j \in \mathcal{N}_{i,h} : \operatorname{var}\left(((\boldsymbol{I}-\Pi_{\bar{H}_p})\Lambda^{(0)})_{j|i}\right) \geq \frac{\operatorname{var}(\partial f_i(X_i^{(k+1)}))}{2^{2\delta}-1}. \tag{E.20}$$

By inspecting the above condition, we have the following remarks.

Remark 1. *($\left(\boldsymbol{I}-\Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}^{(0)} \neq \mathbf{0}$ can be realized by randomly initializing $\boldsymbol{\lambda}^{(0)}$). Of note, $[\boldsymbol{C}\ \boldsymbol{PC}] \in \mathbb{R}^{2M_m\times 2N_n}$ can be viewed as a new graph incidence matrix with $2N_n$ nodes and $2M_m$ edges (see (c) in Fig. E.1 for an example); we thus have $\dim(\bar{H}_p) \leq 2N_n - 1$, and $\bar{H}_p^{\perp}$ is non-empty. For a connected graph with the number of edges not less than the number of nodes (i.e., $M_m \geq N_n$), we conclude that a randomly initialized $\boldsymbol{\lambda}^{(0)} \in \mathbb{R}^{2M_m}$ will achieve $\left(\boldsymbol{I}-\Pi_{\bar{H}_p}\right)\boldsymbol{\lambda}^{(0)} \neq \mathbf{0}$ with probability 1.*

Remark 2. *(The privacy is still guaranteed if the adversary has full knowledge of the subspace $\bar{H}_p$). If the network topology, i.e., $\boldsymbol{B}$, is known to the adversary, it is able to construct the subspace $\bar{H}_p$ by using $\boldsymbol{C}$ and $\boldsymbol{PC}$. However, knowing $\bar{H}_p$ will not compromise the privacy because the term $\sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\boldsymbol{\lambda}_{j|i}^{(k)}$ in* (E.17) *can not be determined by the adversary. The reason is that as long as $\boldsymbol{\lambda}^{(0)} \notin \bar{H}_p$, the adversary is not able to reconstruct the dual variables transmitted between the honest nodes.*

Remark 3. *(The privacy will not be compromised even though the adversary collects information over iterations). Since the updates are only conducted in the*

convergent subspace, even if the adversary collects information over iterations, it can only know the difference $\boldsymbol{\lambda}^{(k+2)} - \boldsymbol{\lambda}^{(k)} \in \bar{H}_p$. *With the knowledge of the dual variables associated with the corrupted nodes only, again the adversary is not able to determine the dual variables related to the honest nodes. Hence,* $\sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\boldsymbol{\lambda}_{j|i}^{(k)}$ *in* (E.17) *can not be determined and the privacy is thus guaranteed.*

Remark 4. *(No trade-off between privacy and accuracy). No matter how much noise is inserted in the non-convergent subspace, the convergence of the optimization variable* $\boldsymbol{x} \to \boldsymbol{x}^*$ *is guaranteed. By inspecting* (E.4)*, we can see that the* $\boldsymbol{x}$*-update is independent of* $(\boldsymbol{I} - \Pi_{\bar{H}_p})\boldsymbol{\lambda}$ *as* $\boldsymbol{\lambda}^\top \left(\boldsymbol{I} - \Pi_{\bar{H}_p}\right) \boldsymbol{P}\boldsymbol{C} = 0$.

Details of the proposed approach using PDMM are shown in algorithm 0. And the analysis of both output correctness and individual privacy is summarized below.

Output correctness

As proven in [35], with strictly convex $f(\boldsymbol{x}_i)$, the optimization variable $\boldsymbol{x}_i^{(k)}$ of each node i of the PDMM is guaranteed to converge geometrically (linearly on a logarithmic scale) to the optimum solution $\boldsymbol{x}_i^*$, regardless of the initialization of both $\boldsymbol{x}^{(0)}$ and $\boldsymbol{\lambda}^{(0)}$, thereby ensuring the correctness. Moreover, for convex functions that are not strictly convex, a slightly modified version called averaged PDMM (see Section 7.3 for an example) can be used to guarantee the convergence.

Individual privacy

From (E.20), we conclude that the proposed algorithm is able to achieve perfect security, against both passive and eavesdropping adversaries as long as the honest node has at least one honest neighbor, i.e., $\mathcal{N}_{i,h} \neq \emptyset$ and Assumption 1 holds.

Overall, the proposed subspace perturbation approach is able to achieve perfect security without compromising the accuracy.

5.3 Discussions

In Remark 4 we mentioned that the proposed approach has no trade-off between privacy and accuracy. When considering practical signal processing tasks that require quantization, the above claim still holds if the quantizer has a fixed resolution (cell width), but there will be a trade-off between privacy and bit rate as an increase in variance will require a higher bit rate. On the other hand, there will be a trade-off between privacy and accuracy if the quantizer has a fixed bit rate, since increasing the noise will end up with a lower resolution. These quantization related trade-offs, however, exist in all approaches

Algorithm 6 Privacy-preserving distributed optimization via subspace perturbation using PDMM

1: Each node $i \in \mathcal{N}$ initializes its optimization variable $\boldsymbol{x}_i^{(0)}$ arbitrarily, and its dual variables $\boldsymbol{\lambda}_{i|j}^{(0)}, j \in \mathcal{N}_i$ are randomly initialized with a certain distribution with large variance (specified by the required privacy level).
2: Node i broadcasts $\boldsymbol{x}_i^{(0)}$ and sends the initialized $\{\boldsymbol{\lambda}_{i|j}^{(0)}\}$ to its neighbor j through secure channels [41].
3: **while** $\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^*\|_2 <$ threshold **do**
4: Activated a node uniformly at random, e.g., node i, updates $\boldsymbol{x}_i^{(k+1)}$ using (E.5).
5: Node i broadcasts $\boldsymbol{x}_i^{(k+1)}$ to its neighbors $j \in \mathcal{N}_i$ (through non-secure channels).
6: After receiving $\boldsymbol{x}_i^{(k+1)}$, each neighbor $j \in \mathcal{N}_i$ updates $\boldsymbol{\lambda}_{i|j}^{(k+1)}$ using (E.6).
7: **end while**

using noise insertion for example the differential privacy approaches. One way to circumvent these trade-offs is to adopt a fixed-rate quantizer that change the cell width adaptively during the iterations [46, 47].

In connection to the above quantization effects, we note that the lower bound of information leakage is very important in reducing these trade-offs, i.e., minimizing the communication bandwidth (bit rate) or the error in the algorithm output, because it helps to specify the minimum noise variance that needs to be added for perfect security; it is not necessary to set the noise variance to be large if the lower bound is not low enough.

6 Proposed approach using other optimizers

In this section, we demonstrate the general applicability of the proposed subspace perturbation method. In fact, the proposed method can be generally applied to any distributed optimizer if the introduced dual variables converge only in a subspace (i.e., there is a non-empty nullspace), which is indeed usually true, because these optimizers often work in a subspace determined by the incidence matrix of a graph. To substantiate this claim, we will show that the subspace perturbation also applies to ADMM and the dual ascent method. We then illustrate their differences by linking the convergent subspaces to their graph topologies.

6.1 ADMM

Given a standard distributed optimization problem stated in (E.1), the augmented Lagrangian function of ADMM [33] is given by

$$L(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{z}) = f(\boldsymbol{x}) + \boldsymbol{v}^\top(\boldsymbol{M}\boldsymbol{x} + \boldsymbol{W}\boldsymbol{z}) + \frac{c}{2}\|\boldsymbol{M}\boldsymbol{x} + \boldsymbol{W}\boldsymbol{z}\|^2, \tag{E.21}$$

where $\boldsymbol{M} \in \mathbb{R}^{2M_m \times N_n}$, like PDMM, is a matrix related to the graph incidence matrix, and $\boldsymbol{M} = \left[\boldsymbol{B}_+^\top \; -\boldsymbol{B}_-^\top\right]^\top$, $\boldsymbol{W} = \left[-\boldsymbol{I}^\top \; -\boldsymbol{I}^\top\right]^\top \in \mathbb{R}^{2M_m \times M_m}$. $\boldsymbol{v} \in \mathbb{R}^{2M_m}$ and $\boldsymbol{z} \in \mathbb{R}^{2M_m}$ are the introduced dual variables and auxiliary variable for constraints, respectively. The updating functions of ADMM are given by

$$\boldsymbol{x}^{(k+1)} = \arg\min_{\boldsymbol{x}} L\left(\boldsymbol{x}, \boldsymbol{z}^{(k)}, \boldsymbol{v}^{(k)}\right) \tag{E.22}$$

$$\boldsymbol{z}^{(k+1)} = \arg\min_{\boldsymbol{z}} L\left(\boldsymbol{x}^{(k+1)}, \boldsymbol{z}, \boldsymbol{v}^{(k)}\right) \tag{E.23}$$

$$\boldsymbol{v}^{(k+1)} = \boldsymbol{v}^{(k)} + c\left(\boldsymbol{M}\boldsymbol{x}^{(k+1)} + \boldsymbol{W}\boldsymbol{z}^{(k+1)}\right). \tag{E.24}$$

By inspecting the $\boldsymbol{v}$-update in (E.24), we can see that it has a similar structure to that of the $\boldsymbol{\lambda}$-update in (E.10) in PDMM. Let $\bar{H}_a = \mathrm{span}(\boldsymbol{M}) + \mathrm{span}(\boldsymbol{W})$ and the matrix $[\boldsymbol{M} \; \boldsymbol{W}]$ can also be seen as an incidence matrix of an extended bipartite graph (see (b) in Fig. E.1 for an example). Therefore, we have $\dim(\bar{H}_a) \leq M_m + N_n - 1$ and every $\boldsymbol{v}$-update only effects $\left(\Pi_{\bar{H}_a}\right)\boldsymbol{v} \in \bar{H}_a$ and leaves $(\boldsymbol{I} - \Pi_{\bar{H}_a})\boldsymbol{v} \in \bar{H}_a^\perp, \bar{H}_a^\perp = \mathrm{null}(\boldsymbol{M}^\top) \cap \mathrm{null}(\boldsymbol{W}^\top)$, unchanged. In addition to this, similar as (E.15) in PDMM, the local optimization variable $\boldsymbol{x}_i^{(k+1)}$ of node i is computed by

$$\boldsymbol{0} \in \partial f_i(\boldsymbol{x}_i^{(k+1)}) + \sum_{j \in \mathcal{N}_i} \boldsymbol{v}_{i|j}^{(k)} + c \sum_{j \in \mathcal{N}_i} (\boldsymbol{x}_i^{(k+1)} - \boldsymbol{z}_{i,j}^{(k)}). \tag{E.25}$$

Note that ADMM is not a broadcasting protocol, i.e., it requires pairwise communications for the auxiliary variable. The individual privacy is thus dependent on both $\boldsymbol{x}$ and $\boldsymbol{z}$. To simplify the analysis, we remark that revealing the auxiliary variable $\boldsymbol{z}$ will not disclose more information than revealing $\boldsymbol{x}$ by the data processing inequality [45]. As a consequence, it is sufficient to analyze the individual privacy through (E.25). As for the knowledge of the adversary, we note that, in addition to the optimization variable and the dual variable, all the auxiliary variables $\{\boldsymbol{z}_{i,j}^{(k)}\}_{(i,j)\in\mathcal{E}}$ are assumed to be known by the adversary as they can be eavesdropped. We then conclude that after deducing all the known terms, i.e., $\sum_{j \in \mathcal{N}_{i,c}} \boldsymbol{v}_{i|j}^{(k)}$ and $c\sum_{j \in \mathcal{N}_i}(\boldsymbol{x}_i^{(k+1)} - \boldsymbol{z}_{i,j}^{(k)})$, from (E.25), what

the adversary observes is

$$\begin{aligned}&\partial f_i(\boldsymbol{x}_i^{(k+1)}) + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{v}_{i|j}^{(k)} \\ &= \partial f_i(\boldsymbol{x}_i^{(k+1)}) + \sum_{j\in\mathcal{N}_{i,h}} \left(\Pi_{\bar{H}_a}\boldsymbol{v}^{(k)}\right)_{i|j} \\ &+ \sum_{j\in\mathcal{N}_{i,h}} \left((\boldsymbol{I}-\Pi_{\bar{H}_a})\boldsymbol{v}^{(0)}\right)_{i|j}. \end{aligned} \tag{E.26}$$

The proof for both the output correctness and individual privacy using ADMM follows a similar structure as that of PDMM. The sufficient condition for perfect security using ADMM becomes

$$\exists j \in \mathcal{N}_{i,h} : \mathrm{var}\left(((\boldsymbol{I}-\Pi_{\bar{H}_a})\boldsymbol{v}^{(0)})_{i|j}\right) \geq \frac{\mathrm{var}(\partial f_i(X_i^{(k+1)}))}{2^{2\delta}-1}. \tag{E.27}$$

We thus conclude that perfect security can be achieved using ADMM via subspace perturbation.

Of note, there are variations in ADMM. For example, [48] showed that the auxiliary variable can be eliminated and the dual variable update can be simplified by proper initialization; it requires that the dual variable should be initialized properly such that it is in the column space of $[(\boldsymbol{B})^\top\ (-\boldsymbol{B})^\top]^\top$. However, such initialization ensures $(\boldsymbol{I}-\Pi_{\bar{H}_a})\boldsymbol{v}^{(0)} = \boldsymbol{0}$ and thus there is no subspace noise for protecting the private data. Instead, we need to randomly initialize the dual variable $\boldsymbol{v}^{(0)}$ such that (E.27) can be satisfied.

6.2 Dual ascent method

The Lagrangian of the dual ascent method for solving (E.1) is given by

$$L(\boldsymbol{x},\boldsymbol{u}) = f(\boldsymbol{x}) + \boldsymbol{u}^\top\boldsymbol{B}\boldsymbol{x}, \tag{E.28}$$

where $\boldsymbol{u}\in\mathbb{R}^{M_m}$ is the introduced dual variable. The updating function is given by

$$\boldsymbol{x}^{(k+1)} = \arg\min_{\boldsymbol{x}} L\left(\boldsymbol{x},\boldsymbol{u}^{(k)}\right) \tag{E.29}$$

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + t^{(k)}\boldsymbol{B}\boldsymbol{x}^{(k+1)}, \tag{E.30}$$

where $t^{(k)}$ denotes the step size at iteration k. Likewise, the $\boldsymbol{u}$-update in (E.30) has a similar structure as the $\boldsymbol{\lambda}$-update of PDMM and the $\boldsymbol{v}$-update of ADMM. Here the convergent subspace is $\bar{H}_d = \mathrm{span}(\boldsymbol{B})$ and $\boldsymbol{B}$ is also rank deficient as it is related to the graph incidence matrix. Hence, we conclude that the proposed subspace perturbation method also works for the dual ascent method.

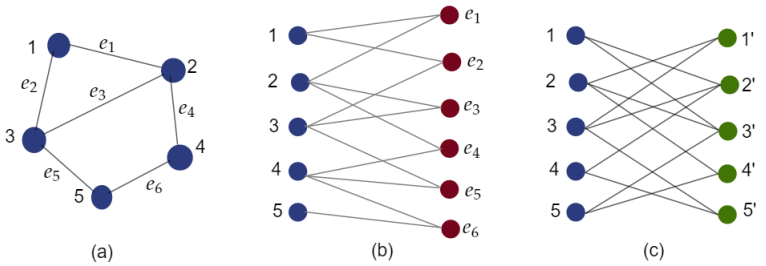


Fig. E.1: An example of graph topologies associated with dual ascent, ADMM and PDMM with $u = 1$: (a) A graph with $n = 5$ nodes and $m = 6$ edges. (b) The bipartite graph constructed by ADMM with $n + m$ nodes and $2m$ edges. (c) The bipartite graph constructed by PDMM with $2n$ nodes and $2m$ edges.

6.3 Linking graph topologies and subspaces

Thus far, we have shown that the dual variable updates of PDMM, ADMM and the dual ascent method are dependent only on their corresponding subspaces: $\bar{H}_p = \text{span}(\boldsymbol{C}) + \text{span}(\boldsymbol{PC})$, $\bar{H}_a = \text{span}(\boldsymbol{M}) + \text{span}(\boldsymbol{W})$ and $\bar{H}_d = \text{span}(\boldsymbol{B})$. As mentioned before, each of the matrices $[\boldsymbol{C}\ \boldsymbol{PC}]$, $[\boldsymbol{M}\ \boldsymbol{W}]$ and $\boldsymbol{B}$ can be seen as an incidence matrix of a graph, therefore they all have a non-empty left nullspace for subspace perturbation as long as $m \geq n$ (Remark 1). To examine the appearance of these constructed graphs, in Fig. E.1 we give an example of these graphs and provide illustrative insights into the differences among these optimizers.

7 Applications

To demonstrate the potential of the proposed approach to be used in a wide range of applications, we now present the application of the proposed subspace perturbation to three fundamental problems: distributed average consensus, distributed least squares and distributed LASSO, because they serve as building blocks for many other signal processing tasks, such as denoising [49], interpolation [50], machine learning [51, 52] and compressed sensing [53, 54]. We first introduce the application and then perform the individual privacy analysis. We will continue using PDMM to introduce the details, but the numerical results of using all the discussed optimizers will be presented in the next section.

7.1 Privacy-preserving distributed average consensus

The optimization problem setup (E.1) for distributed average consensus becomes

$$\begin{aligned} \min_{\boldsymbol{x}_i} \quad & \sum_{i\in\mathcal{N}} \frac{1}{2}\|\boldsymbol{x}_i - \boldsymbol{s}_i\|_2^2 \\ \text{s.t.} \quad & \boldsymbol{x}_i = \boldsymbol{x}_j, \forall (i,j) \in \mathcal{E}, \end{aligned} \tag{E.31}$$

where $\boldsymbol{s}_i$ denotes the initial state value held by node i. The optimum solution for each optimization variable is $\boldsymbol{x}_i^* = n^{-1}\sum_{i\in\mathcal{N}} \boldsymbol{s}_i$ and $\boldsymbol{x}^* = (\boldsymbol{x}_1^*, \ldots, \boldsymbol{x}_n^*)^\top$. Privacy-preserving distributed average consensus is widely investigated in the literature [55–63] and it aims estimate the average of all the nodes' initial state values over a network and keep each node's initial state value unknown to others. Such privacy-preserving solutions are highly desired in practice. For example, in face recognition applications, computing mean faces is usually required, thus prompting privacy concerns. Here, a group of people may cooperate to compute the mean face, but none of them would like to reveal their own facial images during the computation.

Individual privacy

Note that in distributed average consensus, the requirement of protecting the initial state value $\boldsymbol{s}_i$ is equivalent to protecting $\partial f_i(\boldsymbol{x}_i^{(k+1)}) = \boldsymbol{x}_i^{(k+1)} - \boldsymbol{s}_i$, since the optimization variable $\boldsymbol{x}_i^{(k+1)}$ is known to the adversary and can thus be seen as a constant. To comply with existing approaches, in what follows we will analyze the privacy in terms of the initial state value $\boldsymbol{s}_i$. Substitute $\partial f_i(\boldsymbol{x}_i^{(k+1)}) = \boldsymbol{x}_i^{(k+1)} - \boldsymbol{s}_i$ in (E.17) and remove the known term $\boldsymbol{x}_i^{(k+1)}$, we then have

$$-\boldsymbol{s}_i + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\boldsymbol{\lambda}_{j|i}^{(k)}. \tag{E.32}$$

As shown in [31, 55], the only revealed information here would be the partial sums of all the honest components (connected subgraphs consist of honest nodes only) after removal of all the corrupted nodes. Let $\mathcal{H}$ denote the node set of the component that the honest node i belongs to, the lower bound of information leakage for node i is thus given by $I(S_i; \sum_{j\in\mathcal{H}} S_j) = \delta$. We conclude that, given δ, the proposed approach is able to achieve perfect security, i.e.,

$$I\left(S_i\,;\, S_i + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{B}_{i|j}\Lambda_{j|i}^{(k)}\right) \leq \delta, \tag{E.33}$$

by satisfying (E.20) in which $\mathrm{var}(\partial f_i(X_i^{(k+1)}))$ is replaced by $\mathrm{var}(S_i)$.

7.2 Privacy-preserving distributed least squares

Privacy-preserving distributed least squares aims to find a solution for a linear system (here we consider an overdetermined system in which there are more equations than unknowns) over a network in which each node has only partial knowledge of the system and is only able to communicate with its neighbors, and in the meantime the local information held by each node should not be revealed to others. More specifically, here the local information of node i means both the input observations, denoted by $\boldsymbol{Q}_i \in \mathbb{R}^{p_i \times u}$, $p_i > u$ and decision vector, denoted by $\boldsymbol{y}_i \in \mathbb{R}^{p_i}$. That is, each node i has p_i observations, and each contains an u-dimensional feature vector. Collecting all the local information, we thus have $\boldsymbol{Q} = [\boldsymbol{Q}_1^\top, \ldots, \boldsymbol{Q}_n^\top]^\top \in \mathbb{R}^{P_n \times u}$ and $\boldsymbol{y} = [\boldsymbol{y}_1^\top, \ldots, \boldsymbol{y}_n^\top]^\top \in \mathbb{R}^{P_n}$, where $P_n = \sum_{i \in \mathcal{N}} p_i$.

The least-squares problem in a distributed network can be formulated as a distributed linearly constrained convex optimization problem, and the problem setup in (E.1) becomes

$$\begin{array}{ll} \min\limits_{\boldsymbol{x}_i} & \sum_{i \in \mathcal{N}} \frac{1}{2} \|\boldsymbol{y}_i - \boldsymbol{Q}_i \boldsymbol{x}_i\|_2^2 \\ \text{s.t.} & \boldsymbol{x}_i = \boldsymbol{x}_j, \forall (i,j) \in \mathcal{E}, \end{array} \tag{E.34}$$

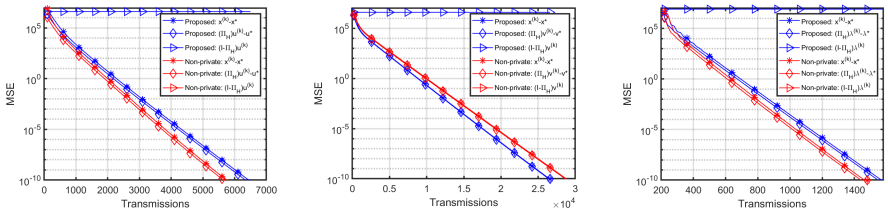
where the optimum solution is given by $\boldsymbol{x}_i^* = (\boldsymbol{Q}^\top \boldsymbol{Q})^{-1} \boldsymbol{Q}^\top \boldsymbol{y} \in \mathbb{R}^u, \forall i \in \mathcal{N}$.

Individual privacy

Note that the local information $\boldsymbol{y}_i$ and $\boldsymbol{Q}_i$ usually contain users' sensitive information [24]. Take the distributed linear regression as an example, which is widely used in the field of machine learning, and consider the case that several hospitals want to collaboratively learn a predictive model by exploring all the data in their datasets. However, such collaborations are limited because they must comply with policies such as the general data protection regulation (GDPR) and because individual patients/customers may feel uncomfortable with revealing their private information to others, such as insurance data and health condition. In this context, since $\partial f_i(\boldsymbol{x}_i^{(k+1)}) = \boldsymbol{Q}_i^\top (\boldsymbol{Q}_i \boldsymbol{x}_i^{(k+1)} - \boldsymbol{y}_i)$ contains sensitive information regarding the local information $\boldsymbol{Q}_i$ and $\boldsymbol{y}_i$ of node i, it is thus important to protect it from being revealed. We can see that at the end each node obtains the optimum solution $\forall i \in \mathcal{N} : \boldsymbol{x}_i^* = (\boldsymbol{Q}^\top \boldsymbol{Q})^{-1} \boldsymbol{Q}^\top \boldsymbol{y}$. The lower bound δ is thus the amount of information learned about $\partial f_i(\boldsymbol{x}_i)$ of honest node $i \in \mathcal{N}_h$ by knowing $\boldsymbol{x}_i^*$ given the knowledge of the corrupted nodes, i.e., $\{f_i(\boldsymbol{x}_i)\}_{i \in \mathcal{N}_c}$. Hence, the propose approach is able to achieve privacy-preserving distributed least squares, i.e., perfect security (E.18) is guaranteed as long as (E.20) is satisfied.

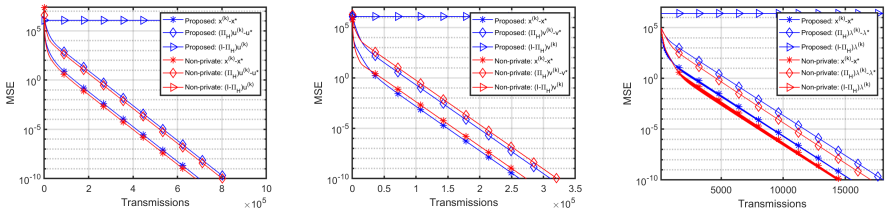
7.3 Privacy-preserving distributed LASSO

Privacy-preserving distributed LASSO aims to securely find a sparse solution when solving an underdetermined system (in which the number of equations



(a) (a) **(b)** (b) **(c)** (c)

Fig. E.2: Distributed average consensus with two different initializations of the dual variable with a variance of 10^6: convergence of the optimization variable, the convergent and non-convergent component of the dual variable, using (a) dual ascent, (b) ADMM and (c) PDMM.



(a) (a) **(b)** (b) **(c)** (c)

Fig. E.3: Distributed least squares with two different initializations of the dual variable with a variance of 10^6: convergence of the optimization variable, the convergent and non-convergent component of the dual variable of (a) dual ascent, (b) ADMM and (c) PDMM.

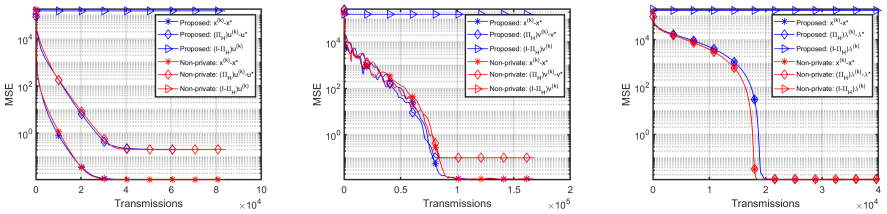
is less than number of unknowns). We thus have a network similar to the previous least squares section but with the dimension $P_n < u$ to ensure an underdetermined system. The distributed LASSO is formulated as a ℓ_1-regularized distributed least squares problem given by

$$\begin{aligned} &\min_{\boldsymbol{x}_i} \sum_{i\in\mathcal{N}} \left(\frac{1}{2}\|\boldsymbol{y}_i - \boldsymbol{Q}_i\boldsymbol{x}_i\|_2^2 + \alpha|\boldsymbol{x}_i| \right) \\ &\text{s.t. } \boldsymbol{x}_i = \boldsymbol{x}_j, \forall (i,j) \in \mathcal{E} \end{aligned} \tag{E.35}$$

where α the constant controlling the sparsity of the solution. Because the objective function is convex but not strictly convex, we use averaged PDMM to ensure convergence, the $\boldsymbol{x}$-updating function remains the same, and the $\boldsymbol{\lambda}$-updating function in (E.6) is replaced with a weighted average by

$$\begin{aligned} \boldsymbol{\lambda}^{(k+1)} &= \theta(\boldsymbol{\lambda}^{(k)} + c\boldsymbol{C}(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)})) \\ &+ (1-\theta)\left(\boldsymbol{P}\boldsymbol{\lambda}^{(k)} + c(\boldsymbol{C}\boldsymbol{x}^{(k+1)} + \boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(k)})\right), \end{aligned} \tag{E.36}$$

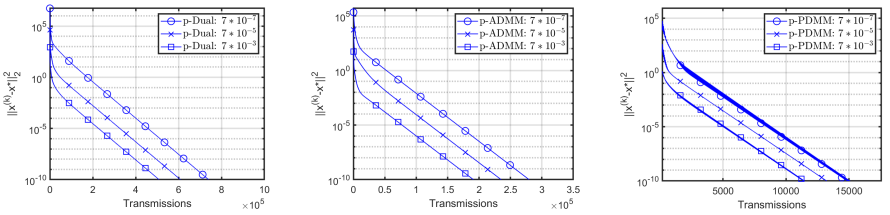
where $0 < \theta < 1$ is the constant controlling the average weight. The output correctness is ensured by simply replacing the equation (E.6) in step 6 of algorithm 0 with the above equation. The rest analysis follows similarly as the



(a) (a) **(b)** (b) **(c)** (c)

Fig. E.4: Distributed LASSO with two different initializations of the dual variable with a variance of 10^6: convergence of the optimization variable, the convergent and non-convergent component of the dual variable of (a) dual ascent, (b) ADMM and (c) PDMM.

example for distributed least squares described above. Hence, with (E.20), we are able to achieve perfect security in distributed LASSO.



(a) (a) **(b)** (b) **(c)** (c)

Fig. E.5: Convergence of the optimization variable in terms of three different privacy levels, i.e., approximately 7×10^{-3}, 7×10^{-5} and 7×10^{-7} bits, for (a) proposed dual ascent (p-Dual), (b) proposed ADMM (p-ADMM) and (c) proposed PDMM (p-PDMM) in a distributed least squares application.

8 Numerical results

In this section, several numerical tests[3] are conducted to demonstrate both the generally applicability and the benefits of the proposed subspace perturbation in terms of several important parameters including accuracy, convergence rate, communication cost and privacy level.

We simulated a distributed network by generating a random graph with $n = 20$ nodes and the communication radius r was set at $r^2 = 2\frac{\log n}{n}$ so ensure that the graph is connected with high probability [64]. For simplicity, all the local data regarding $f(\boldsymbol{x}_i)$ in each application, i.e., $\boldsymbol{s}_i$ in distributed average consensus, $\boldsymbol{Q}_i$ and $\boldsymbol{y}_i$ in distributed least squares and LASSO, are randomly

[3]The code for reproducing these results is available at https://github.com/qiongxiu/PrivacyOptimizationSubspace

generated from a Gaussian distribution with unit variance, and the optimization variables are initialized with zeros. Additionally, we initialize all the dual variables with a Gaussian distribution with different variances.

8.1 General applicability

In Fig. E.2, E.3 and E.4, we compare the convergence behavior of the proposed subspace perturbation methods (blue lines) with traditional non-private approaches (red lines) by using three distributed optimizers in three applications: distributed average consensus, least squares and LASSO. More specifically, the blue lines indicate that the dual variables are randomly initialized with a variance of 10^6, such that the non-convergent component (blue line with triangle markers) can protect the private data, whereas the red lines mean that the dual variables are initialized within the convergent subspace, and the private data are therefore not protected, because the non-convergent component is zero (the red lines with triangle markers are not shown in the plots). We can see that the proposed approach has no effect on the accuracy, because all the optimization variables converge to the same optimum solution as the non-private counterparts. Overall, we can conclude that

1. the proposed approach is able to achieve both accuracy and privacy simultaneously;
2. it is able to solve a variety of convex problems;
3. it is generally applicable to a broad range of distributed convex optimization methods.

8.2 Privacy level-invariant convergence rate

Another important aspect to quantify the performance is the influence on the convergence rate when considering privacy. Because the convergence rates of the discussed distributed optimizers depend only on the underlying graph topologies rather than the initializations, initializing the dual variables with greater variance will therefore not change the convergence rate; and it will only result in only a larger offset in the initial error. To validate these results, in Fig. E.5 we show the convergence behavior of the proposed approaches in the distributed least squares problem under three different privacy levels: $\delta = 7 \times 10^{-3}$, 7×10^{-5}, and 7×10^{-7} bits. In order to achieve perfect security, based on Proposition 4, the variance of each dual variable is set as $10^2, 10^4$ and 10^6, respectively. As expected, in all optimizers, the convergence rate remains identical regardless of the privacy level.

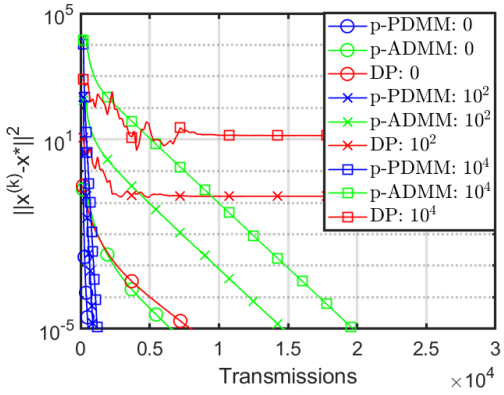


Fig. E.6: Performance comparison: convergence behavior under three different noise variances using the proposed p-PDMM, p-ADMM and an existing differential privacy (DP) approach.

8.3 Comparison with differential privacy

To demonstrate the benefits of the proposed method, in Fig. E.6, we compare the proposed approaches (both p-PDMM and p-ADMM) with a differential privacy approach [60] by using the distributed average consensus application. We consider three cases in which the variance of each dual variable is set as $0, 10^2, 10^4$. We can see that the accuracy of the differential privacy approach decreases with increasing privacy level. Hence, there is a trade-off between privacy and accuracy. Additionally, the convergence rates of differential privacy approaches will also be affected when increasing the level of privacy, because noise is inserted at every iteration, and a higher privacy level will also result in a slower convergence rate.

8.4 Information loss over the iterative process

To visualize how the information loss behaves during the iterative process, we perform 10^5 Monte Carlo simulations and estimate the normalized mutual information (NMI) $\frac{I(S_i;X_i^{(k)})}{I(S_i;S_i)}$ of distributed average consensus using the non-parametric entropy estimation toolbox (npeet) [65]. In Fig. E.7, we show the estimated normalized mutual information of the proposed p-PDMM with a noise variance of 10^6 and traditional non-private PDMM, in which the dual variables are initialized with zeros. Since both approaches converge to the same average result $\boldsymbol{x}_i^* = n^{-1}\sum_{i\in\mathcal{N}} \boldsymbol{s}_i$, they ultimately have the same information loss $I(S_i; X_i^*)$, which corresponds to the lower bound of information leakage under the condition that there is no passively corrupted nodes. As expected, the information loss of the proposed p-PDMM never exceeds the lower bound; hence, the proposed approach achieves perfect security. However, the n-PDMM reveals all the information about s_i at the first iteration as no noise is inserted;

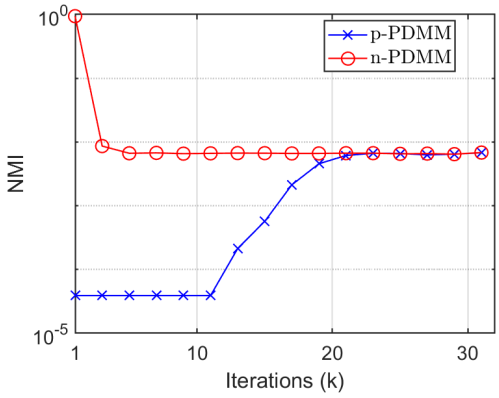


Fig. E.7: Normalized mutual information of an arbitrary node i (i.e., $\frac{I(S_i;X_i^{(k)})}{I(S_i;S_i)}$) using the proposed p-PDMM and non-private PDMM (n-PDMM) for each iteration.

thus the privacy is not protected at all.

9 Conclusions

In this paper, a novel and general subspace perturbation method was proposed for privacy-preserving distributed optimization. As a noise insertion approach, this method is more practical than SMPC based approaches in terms of both computation and communication costs. Additionally, by inserting noise in subspace, it circumvents the trade-off between privacy and accuracy in traditional noise insertion approaches such as differential privacy. Moreover, the proposed method guarantees perfect security and is generally applicable to various optimizers and all convex problems. Furthermore, we consider both passive and eavesdropping adversary models; in which the private data of each honest node are protected as long as the node has one honest neighbor, and only secure channel encryption in the initialization is required.

10 Appendix

10.1 Proof of Proposition 3

Proof.

$$\begin{aligned}
&I(X_1,\ldots,X_n;Z_1,\ldots,Z_n)\\
&\quad= h(Z_1,\ldots,Z_n) - h(Z_1,\ldots,Z_n|X_1,\ldots,X_n)\\
&\quad\overset{\text{(a)}}{=} h(Z_1,\ldots,Z_n) - h(Y_1,\ldots,Y_n)\\
&\quad\overset{\text{(b)}}{=} \sum_{i=1}^{n} h\big(Z_i|Z_1,\ldots,Z_{i-1}\big) - \sum_{i=1}^{n} h\big(Y_i\big)\\
&\quad\overset{\text{(c)}}{\leq} \sum_{i=1}^{n} h\big(Z_i\big) - \sum_{i=1}^{n} h(Y_i)\\
&\quad\overset{\text{(d)}}{=} \sum_{i=1}^{n} I(X_i;Z_i)\\
&\quad\overset{\text{(e)}}{=} \sum_{i=1}^{n} I(X_i/\sigma_{Z_i};Z_i').
\end{aligned}$$

Step (a) holds, as $h(Z_i|X_i) = h(Y_i)$, (b) holds from the chain rule of differential entropy and the condition that the Y_i's are independent random variables, (c) holds, as conditioning decreases entropy, (d) holds, as $h\big(Z_i\big) - h(Y_i) = h(Z_i) - h(Z_i|X_i) = I(X_i;Z_i)$, and (e) holds from the fact that mutual information is scaling invariant. As a consequence,

$$\begin{aligned}
\lim_{\sigma^2_{Y_i}\to\infty} \sum_{i=1}^{n} I(X_i;Z_i) &= \lim_{\sigma_{Z_i}\to\infty} \sum_{i=1}^{n} I(X_i/\sigma_{Z_i};Z_i')\\
&= \sum_{i=1}^{n} I(0;Z_i') = 0,
\end{aligned}$$

thereby proving our claims. □

10.2 Proof of Proposition 4

Proof. As X and Y are independent, we have $\sigma_Z^2 = \sigma_X^2 + \sigma_Y^2$. For a Gaussian random variable with variance σ^2, the differential entropy is given by $\frac{1}{2}\log(2\pi e\sigma^2)$, so that

$$\begin{aligned}
\delta = I(X;Z) &= h(Z) - h(Z|X)\\
&= h(Z) - h(Y)\\
&\overset{\text{(a)}}{\leq} \frac{1}{2}\log(2\pi e\sigma_Z^2) - \frac{1}{2}\log(2\pi e\sigma_Y^2)\\
&= \frac{1}{2}\log(1+\sigma_X^2/\sigma_Y^2),
\end{aligned} \tag{E.37}$$

where (a) holds, because the maximum entropy of a random variable with fixed variance is achieved by a Gaussian distribution; and equality holds if X is also Gaussian distributed. □

Acknowledgement

We thank the anonymous reviewers for their helpful comments to improve this manuscript. In particular the reviewer who suggested to directly define the subgradient as the private data helped to simplify and clarify this manuscript.

References

[1] M. Anderson, *Technology device ownership, 2015*, Pew Research Center, 2015.

[2] J. Poushter and others, "Smartphone ownership and internet usage continues to climb in emerging economies," *Pew Research Center*, vol. 22, pp. 1–44, 2016.

[3] T. Sherson, W. B. Kleijn, R. Heusdens, "A distributed algorithm for robust lcmv beamforming," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 101-105*, 2016.

[4] A. I. Koutrouvelis, T. W. Sherson, R. Heusdens, and R. C. Hendriks, "A low-cost robust distributed linearly constrained beamformer for wireless acoustic sensor networks with arbitrary topology," *IEEE/ACM Trans. Audio, Speech, Lang. Process. vol. 26, no. 8 , pp. 1434–1448*, 2018.

[5] M. Zhu, S. Martínez, "On distributed convex optimization under inequality and equality constraints," *IEEE Trans. Autom. Control., vol. 57, no. 1, pp. 151–164*, 2011.

[6] M. Zibulevsky, M. Elad, "L1-L2 optimization in signal and image processing," *IEEE Signal Process. Mag., vol. 27, no. 3, pp. 76–88*, 2010.

[7] C. Dwork, "Differential privacy," in *ICALP, pp. 1–12*, 2006.

[8] C. Dwork, F. McSherry, K. Nissim, A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory of Cryptography Conf. , pp. 265-284*, 2006.

[9] R. Cramer, I. B. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*, Cambridge University Press, 2015.

[10] Z. Huang, S. Mitra, and N. Vaidya, "Differentially private distributed optimization., pp. 1–10," in *Proc. Int. Conf. Distrib. Comput. Netw*, 2015.

[11] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Trans. Autom. Control., vol. 62, no. 1, pp 50-64*, 2016.

[12] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private distributed convex optimization via functional perturbation," *IEEE Trans. Control Netw. Syst., vol. 5, no. 1, pp 395-408*, 2018.

[13] T. Zhang and Q. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Trans. Inf. Forensics Security, vol. 12, no. 1, pp. 172–187*, 2016.

[14] X. Zhang, M. M. Khalili, and M. Liu, "Recycled ADMM: Improve privacy and accuracy with less computation in distributed algorithms," in *in Proc. 56th Annu. Allerton Conf. Commun., Control, Comput. pp.959–965*, 2018.

[15] X. Zhang, M. M. Khalili, and M. Liu, "Improving the privacy and accuracy of ADMM-based distributed algorithms," *Proc. Int. Conf. Mach. Lear. pp. 5796–5805*, 2018.

[16] Y. Xiong, J. Xu, K. You, J. Liu and L. Wu, "Privacy preserving distributed online optimization over unbalanced digraphs via subgradient rescaling," *IEEE Trans. Control Netw. Syst.*, 2020.

[17] Z. Huang, R. Hu, Y. Gong, and E. Chan-Tin, "DP-ADMM: ADMM-based distributed learning with differential privacy," *IEEE Trans. Inf. Forensics Security., vol. 15, pp. 1002–1012*, 2019.

[18] M. T. Hale, M. Egerstedt, "Differentially private cloud-based multi-agent optimization with constraints," in *Proc. American Control Conf., pp. 1235-1240*, 2015.

[19] M. T. Hale, M. Egerstedt, "Cloud-enabled differentially private multiagent optimization with constraints," *IEEE Trans. Control Netw. Syst., vol. 5, no. 4, pp. 1693–1706*, 2018.

[20] Z. Xu and Q. Zhu, "Secure and resilient control design for cloud enabled networked control systems," in *Proc. 1st ACM Workshop Cyber-Phys. Syst.-Secur. Privacy., pp. 31–42*, 2015.

[21] N. M. Freris and P. Patrinos, "Distributed computing over encrypted data," in *Proc. IEEE 54th Annu. Allerton Conf. Commun., Control, Comput. pp. 1116–1122,* 2016.

[22] Y. Shoukry et al., "Privacy-aware quadratic optimization using partially homomorphic encryption," in *Proc. IEEE 55th Conf. Decis. Control., pp. 5053–5058*, 2016.

[23] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *INFOCOM*, 2011, pp. 820–828.

[24] C. Zhang, M. Ahmad, and Y. Wang, "ADMM based privacy-preserving decentralized optimization," *IEEE Trans. Inf. Forensics Security, vol. 14, no. 3, pp. 565–580*, 2019.

[25] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT, pp. 223–238*, 1999.

[26] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *STOC, pp.169–178*, 2009.

[27] A. C. Yao, "Protocols for secure computations," in *FOCS, pp. 160–164*, 1982.

[28] A. C. Yao, "How to generate and exchange secrets," in *FOCS, pp. 162–167*, 1986.

[29] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO, pp. 643–662*. Springer, 2012.

[30] K. Tjell and R. Wisniewski, "Privacy preservation in distributed optimization via dual decomposition and ADMM," in *Proc. IEEE 58th Conf. Decis. Control., pp. 7203–7208*, 2020.

[31] Q. Li, R. Heusdens and M. G. Christensen, "Convex optimisation-based privacy-preserving distributed average consensus in wireless sensor networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 5895-5899*, 2020.

[32] Q. Li, R. Heusdens and M. G. Christensen, "Convex optimization-based privacy-preserving distributed least squares via subspace perturbation," in *Proc. Eur. Signal Process. Conf., to appear*, 2020.

[33] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning, vol. 3, no. 1, pp. 1–122*, 2011.

[34] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process., vol. 4, no. 1, pp. 173–187*, 2018.

[35] T. Sherson, R. Heusdens, W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 2, pp 334-347*, 2018.

[36] A. H. Poorjam, Y. P. Raykov, R. Badawy, J. R. Jensen, M. G. Christensen and M.A. Little, "Quality control of voice recordings in remote parkinson's disease monitoring using the infinite hidden markov model," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 805-809*, 2019.

[37] A. H. Poorjam, M. S. Kavalekalam, L. Shi, Y. P. Raykov, J. R. Jensen, M. A. Little and M. G. Christensen, "Automatic quality control and enhancement for voice-based remote parkinson's disease detection," *arXiv preprint arXiv:1905.11785*, 2019.

[38] G. Giaconi, D. Gündüz, H. V. Poor, "Privacy-aware smart metering: Progress and challenges," *IEEE Signal Process. Mag., vol. 35, no. 6, pp. 59-78*, 2018.

[39] Q. Li and M. G. Christensen, "A privacy-preserving asynchronous averaging algorithm based on shamir's secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

[40] K. Tjell, I. Cascudo and R. Wisniewski, "Privacy preserving recursive least squares solutions," in *Proc. Eur. Control Conf., pp.3490–3495*, 2019.

[41] D. Dolev, C. Dwork, O. Waarts, M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach., vol. 40, no. 1, pp. 17-47,*, 1993.

[42] I. Wagner and D. Eckhoff, "Technical privacy metrics: a systematic survey," *ACM Comput. Surv., vol. 51, no. 3, pp. 1–38*, 2018.

[43] M. Lopuhaä-Zwakenberg, B. Škorić and N. Li, "Information-theoretic metrics for local differential privacy protocols," *arXiv preprint arXiv:1910.07826*, 2019.

[44] P. Cuff and L. Yu, "Differential privacy as a mutual information constraint," in *Proc. 23rd ACM SIGSAC Conf. Comput. Commun. Secur., pp 43–54*, 2016.

[45] T. M. Cover and J. A. Tomas, *Elements of information theory*, John Wiley & Sons, 2012.

[46] D. H. M. Schellekens, T. Sherson, and R. Heusdens, "Quantisation effects in PDMM: A first study for synchronous distributed averaging," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 4237–4241*, 2017.

[47] J. A. G. Jonkman, T. Sherson, and R. Heusdens, "Quantisation effects in distributed optimisation," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 3649–3653*, 2018.

[48] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Trans. Signal Processing, vol. 62, no. 7, pp. 1750–1761*, 2014.

[49] J. Pang, G. Cheung, A. Ortega, O. C. Au, "Optimal graph Laplacian regularization for natural image denoising," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp 2294-2298*, 2015.

[50] SK Narang, A Gadde, A Ortega, "Signal processing techniques for interpolation in graph structured data," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp 5445-5449*, 2013.

[51] Tibshirani, Robert, "Regression shrinkage and selection via the lasso," *J. Royal Statistical Soc. B*, vol. 58, no. 1, pp. 267–288, 1996.

[52] J. Wright, A. Yang, A. Ganesh, S. Sastry, Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, 2008.

[53] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *IEEE Trans. Inf. Theory., vol. 52, no. 12, pp. 5406–5425*, 2006.

[54] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory., vol. 52, no. 4, pp. 1289–1306*, 2006.

[55] Q. Li, I. Cascudo, and M. G. Christensen, "Privacy-preserving distributed average consensus based on additive secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

[56] N. Gupta, J. Katz, N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine, vol. 50, no. 1, pp. 9515-9520*, 2017.

[57] N. Gupta, J. Kat and N. Chopra, "Statistical privacy in distributed average consensus on bounded real inputs," in *ACC, pp 1836-1841*, 2019.

[58] M. Kefayati, M. S. Talebi, B. H. Khalajand H. R. Rabiee , "Secure consensus averaging in sensor networks using random offsets," in *Proc. of the IEEE Int. Conf. on Telec., and Malaysia Int. Conf. on Commun., pp. 556–560*, 2007.

[59] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," in *ACM workshop Privacy electron. Soc., pp. 81–90*, 2012.

[60] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica, vol. 81, pp. 221–231*, 2017.

[61] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *ECC, pp. 760–765*, 2013.

[62] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Automat Contr., vol. 62, no. 2, pp. 753–765*, 2017.

[63] J. He, L. Cai, C. Zhao, P. Cheng, X. Guan, "Privacy-preserving average consensus: privacy analysis and algorithm design," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 1, pp. 127–138*, 2019.

[64] J. Dall and M. Christensen, "Random geometric graphs," *Physical review E, vol. 66, no. 1, pp. 016121*, 2002.

[65] G. Ver Steeg, "Non-parametric entropy estimation toolbox (npeet)," *https://github.com/gregversteeg/NPEET*, 2000.

Paper F

Communication Efficient Privacy-Preserving Distributed Optimization using Adaptive Quantization

Qiongxiu Li, Richard Heusdens, and Mads Græsbøll Christensen

The paper has been submitted in the
Signal Processing Letter, 2021

The layout has been revised.

Abstract

Privacy issues and communication cost are both major concerns in distributed optimization. There is often a trade-off between them because the encryption methods required in privacy-preservation often incur expensive communication bandwidth. To address this issue, we, in this paper, propose a quantization-based approach to achieve both communication efficient and privacy-preserving solutions in the context of distributed optimization. By deploying an adaptive quantization scheme, we allow each node in the network to achieve its optimum solution with a low communication cost while keeping its private data unrevealed. Additionally, the proposed approach is general and can be applied in various distributed optimization methods, such as the primal-dual method of multipliers (PDMM) and the alternating direction method of multipliers (ADMM). Moveover, we consider two widely used adversary models: honest-but-curious and eavesdropping. Finally, we investigate the properties of the proposed approach using different applications and demonstrate its superior performance in terms of several parameters including accuracy, privacy, and communication cost.

1 Introduction

With the emergence of interconnected or networked systems, distributed optimization is widely used to process its massive amount of data. As the primary computation units in these distributed networks are often personal devices, such as mobile phones and tablets [1, 2], the underlying networked data are private in nature. Furthermore, the available computational resources are also limited by the hardware and energy consumption. As a consequence, novel distributed optimization tools are required that are able to address the privacy concern in a way that is efficient in terms of communication and computational resources.

Existing approaches mostly address the above challenges only partially. To achieve computationally lightweight solutions, noise insertion approaches, which add noise to obfuscate the private data, are widely used in the literature. These methods can be broadly classified into three classes. The first one is the class of differentially private distributed optimization approaches [3–9]. A drawback of these algorithms is that they compromise the algorithm accuracy, as they have an inherent trade-off between privacy and accuracy. The second class is that of secret-sharing based distributed optimization approaches [10, 11] which deploy secret sharing to prevent privacy leakage, a technique used in secure multiparty computation [12, 13]. This, however, comes with additional communication costs as secret sharing requires extra communication for transmitting the shares. The third class is the class of subspace perturbation based distributed optimization approaches [14–16] which, by inserting noise in a subspace determined by the graph topology, alleviates the privacy-accuracy trade-off without severely increasing the communication costs. Note

that when considering the communication cost, aside from the number of times the communication channel is used, there is another critical parameter, namely the communication bandwidth or the corresponding bit-rate. The communication bandwidth is often omitted in the privacy related approaches by assuming infinite precision. However, there is often a fundamental trade-off between privacy and communication bandwidth in noise insertion type methods. The reason for this is that a higher privacy level usually requires a larger amount of noise insertion, which, in turn, increases the noise entropy and thereby the bit-rate.

In this paper we propose to exploit an adaptive quantization scheme to mitigate the aforementioned trade-off between privacy and communication bandwidth. We show that the accuracy of the proposed approach is not compromised by considering both privacy and quantization. To the best of our knowledge, this is the first approach which provides information theoretical privacy guarantee for distributed optimization with quantization.

We use the following notations throughout this paper. Lowercase letters x denote scalars, lowercase boldface letters $\boldsymbol{x}$ denote vectors, uppercase boldface letters $\boldsymbol{X}$ denote matrices. $\boldsymbol{x}_i$ and $\boldsymbol{X}_{ij}$ denote the i-th and (i,j)-th entry of the vector $\boldsymbol{x}$ and the matrix $\boldsymbol{X}$, respectively. Denote calligraphic letters $\mathcal{X}$ as sets and uppercase letters X denote random variables having realizations x.

2 Fundamentals

2.1 Distributed optimization over networks

We model a distributed network as a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ where $\mathcal{N} = \{1, 2, ..., n\}$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of edges. Moreover, let $\mathcal{N}_i = \{j \,|\, (i,j) \in \mathcal{E}\}$ denote the set of neighboring nodes and $d_i = |\mathcal{N}_i|$ the degree (number of neighboring nodes) of node i. A standard distributed convex optimization problem with constraints over the network can be formulated as

$$\begin{aligned} &\min_{\boldsymbol{x}_i, \forall i \in \mathcal{N}} \sum_{i \in \mathcal{N}} f_i(\boldsymbol{x}_i) \\ &\text{s.t. } \forall (i,j) \in \mathcal{E} : \boldsymbol{B}_{i|j}\boldsymbol{x}_i + \boldsymbol{B}_{j|i}\boldsymbol{x}_j = \boldsymbol{b}_{i,j} \end{aligned} \tag{F.1}$$

where $\boldsymbol{x}_i \in \mathbb{R}^u$ denotes the optimization variable of node i, $f_i : \mathbb{R}^u \mapsto \mathbb{R} \cup \{\infty\}$ denotes the local objective function which is assumed to be convex, $\boldsymbol{B}_{i|j}$ and $\boldsymbol{B}_{j|i}$ are edge-related matrices (weights) and $\boldsymbol{b}_{i,j} \in \mathbb{R}^u$ denotes the constraint imposed at edge $(i,j) \in \mathcal{E}$. For simplicity, we assume $u = 1$ and $\boldsymbol{b}_{i,j} = \boldsymbol{0}$, but the results can easily be generalized to arbitrary dimension and cases where $\boldsymbol{b}_{i,j} \neq \boldsymbol{0}$. With this, $\boldsymbol{B}_{i|j}$ and $\boldsymbol{B}_{j|i}$ are related to entries of the incidence matrix $\boldsymbol{B} \in \mathbb{R}^{m \times n}, m = |\mathcal{E}|$, of the graph: $\boldsymbol{B}_{li} = \boldsymbol{B}_{i|j} = 1$, $\boldsymbol{B}_{lj} = \boldsymbol{B}_{j|i} = -1$ if and only if $e_l = (i,j) \in \mathcal{E}$ and $i < j$,

2.2 Distributed optimizers

To solve the problem in (F.1) in a decentralized manner, i.e., where each node is only allowed to exchange information with its neighboring nodes, a number of distributed, iterative optimizers have been proposed, including ADMM [17] and PDMM [18–20]. It has been shown using monotone operator theory and operator splitting techniques that ADMM and PDMM are closely related [20] (see [21] for details on monotone operator theory). For both methods, the update equations at iteration $t = 0, 1, \ldots$ are given by

$$\boldsymbol{x}_i^{(t+1)} = \arg\min_{\boldsymbol{x}_i} \big(f_i(\boldsymbol{x}_i) + \sum_{j \in \mathcal{N}_i} \boldsymbol{z}_{i|j}^{(t)} \boldsymbol{B}_{i|j} \boldsymbol{x}_i + \frac{c d_i}{2} \boldsymbol{x}_i^2 \big)$$

$$\forall j \in \mathcal{N}_i : \boldsymbol{z}_{j|i}^{(t+1)} = \theta \boldsymbol{z}_{j|i}^{(t)} + (1-\theta)(\boldsymbol{z}_{i|j}^{(t)} + 2c \boldsymbol{B}_{j|i} \boldsymbol{x}_i^{(t+1)}),$$

where c is a constant for controlling the convergence rate. Each $e_l = (i, j) \in \mathcal{E}$ is associated to two auxiliary variables $\boldsymbol{z}_l = \boldsymbol{z}_{i|j}$ and $\boldsymbol{z}_{l+m} = \boldsymbol{z}_{j|i}$. Stacking all auxiliary variables together we have $\boldsymbol{z} \in \mathbb{R}^{2m}$. $\theta \in [0, 1)$ is a constant for controlling the averaging weight of Peaceman-Rachford splitting. For example, $\theta = 0$ corresponds to PDMM and $\theta = 0.5$ corresponds to ADMM. For simplicity, in what follows we will use $\theta = 0$, i.e., PDMM, as an example to explain the main idea but the conclusions hold for all $\theta \in [0, 1)$.

3 Problem definition

3.1 Privacy concern and adversary models

In distributed optimization, sensitive personal information is often embedded in each node's local objective function $f_i(\boldsymbol{x}_i)$. Therefore, $f_i(\boldsymbol{x}_i)$ is considered as private data to be protected from being revealed in the process of solving the optimization problem. To analyze the privacy an "adversary model" must be specified. The purpose of such a model is to quantify the system robustness in dealing with different security attacks. In addition to the well-known external eavesdropping adversary that attacks the system by listening to the messages transmitted along the communication links, there is another commonly considered adversary model in distributed networks, namely the passive (also called honest-but-curious) adversary. It controls a number of so-called corrupted nodes who are assumed to follow the algorithm instructions but can collect information together. The passive adversary will then use the collected information to infer the private data of the non-corrupted nodes, which we will refer to as honest nodes.

3.2 Main requirements and related metrics

Putting things together, we now state the main requirements that communication efficient privacy-preserving distributed optimization should satisfy and

introduce the related metrics.

1. **Output correctness**: each node i should obtain the optimal solution to (F.1), denoted by $\boldsymbol{x}_i^*$, at the end of the algorithm. The overall mean square error (MSE) is used to quantify the output correctness, i.e., $\left\|\boldsymbol{x}^{(t)} - \boldsymbol{x}^*\right\|^2$.

2. **Individual privacy**: each node's private information, embedded in $f_i(\boldsymbol{x}_i)$, should be protected under both eavesdropping and passive adversaries throughout the algorithm. In this paper mutual information [22] is adopted to quantify the individual privacy as it has been widely accepted in the literature and is closely related to other metrics like differential privacy [23–26].

3. **Communication cost**: the algorithm should have low communication cost. This cost can be quantified using the total number of bits required for transmitting all messages incurred in the algorithm.

4 Proposed approach

As previously mentioned, noise insertion approaches achieve privacy by inserting noise to obfuscate the private data first and then share the obfuscated data. We will elaborate on this with the following result.

Proposition 5. *(Privacy guarantee using noise insertion) Let R and S be continuous random variables with variance $\sigma_R^2, \sigma_S^2 < \infty$, denoting inserted noise and private data, respectively, and assume that R and S are statistically independent. Given an arbitrarily small $\delta > 0$, there exists $\beta > 0$ such that for $\sigma_R^2 \geq \beta$*

$$I(S; S + R) \leq \delta. \tag{F.2}$$

In the case that the noise R is Gaussian distributed, we have

$$\beta = \frac{\sigma_S^2}{2^{2\delta} - 1}. \tag{F.3}$$

Proof. See [24, Proof of Proposition 1]. □

Hence, the more noise is inserted, the higher privacy level can be obtained. However, more noise will inevitably increase the noise entropy and thus requires a higher communication bandwidth (i.e., bit-rate). In this paper, we propose to use the adaptive quantization scheme of [27, 28] to circumvent this.

For fixed point iterations, the difference of every two successive iterations will converge to zero (i.e, it is a Cauchy sequence), which implies that the

entropy of the difference of successive iterations will decrease to zero as the number of iteration increases. Motivated by this, the idea behind the adaptive quantization scheme of [27, 28] is to quantize the difference of two successive iterations with an adaptive cell-width decreasing with increasing iterations. By doing so, low data rate transmission between nodes can be achieved without compromising the accuracy of the algorithm.

We now proceed to describe the details of the adaptive quantization scheme. Let $\hat{\boldsymbol{z}}$ denote the quantized version of $\boldsymbol{z}$. At iteration $t \in \mathcal{T} = \{0, 1, \ldots, T\}$, each node i first updates the local variables $\boldsymbol{x}_i^{(t+1)}$ and $\boldsymbol{z}_{j|i}^{(t+1)}$ using the quantized $\hat{\boldsymbol{z}}_{i|j}^{(t)}$ from the previous iteration, i.e.,

$$\boldsymbol{x}_i^{(t+1)} = \arg\min_{\boldsymbol{x}_i} \big(f_i(\boldsymbol{x}_i) + \sum_{j \in \mathcal{N}_i} \hat{\boldsymbol{z}}_{i|j}^{(t)} \boldsymbol{B}_{i|j} \boldsymbol{x}_i + \frac{cd_i}{2} \boldsymbol{x}_i^2 \big) \tag{F.4}$$

$$\forall j \in \mathcal{N}_i : \boldsymbol{z}_{j|i}^{(t+1)} = \hat{\boldsymbol{z}}_{i|j}^{(t)} + 2c\boldsymbol{B}_{j|i} \boldsymbol{x}_i^{(t+1)} \tag{F.5}$$

After that, instead of sending out the unquantized $\boldsymbol{z}_{j|i}^{(t+1)}$ to node j directly, node i shares the quantized $\hat{\boldsymbol{z}}_{j|i}^{(t+1)}$. In order to do so, we first define the difference variable $\boldsymbol{v}$ as

$$\boldsymbol{v}^{(t+1)} \triangleq \boldsymbol{z}^{(t+1)} - \hat{\boldsymbol{z}}^{(t)}. \tag{F.6}$$

Let $Q(\cdot)$ denote the quantization operation, we have

$$\hat{\boldsymbol{v}}^{(t+1)} = Q(\boldsymbol{z}^{(t+1)} - \hat{\boldsymbol{z}}^{(t)}) = \boldsymbol{z}^{(t+1)} - \hat{\boldsymbol{z}}^{(t)} + \boldsymbol{n}_{q,v^{(t+1)}}, \tag{F.7}$$

where $\boldsymbol{n}_{q,v^{(t+1)}}$ denotes the noise introduced by quantizing $\boldsymbol{v}^{(t+1)}$. The adopted quantizer has a geometrically decreasing cell-width $\Delta^{(t)} = \gamma^t \Delta^{(0)}$ with initial cell-width $\Delta^{(0)}$ and rate of growth $\gamma \in (0, 1)$. After computing $\hat{\boldsymbol{v}}^{(t+1)}$, the quantized value, $\hat{\boldsymbol{z}}^{(t+1)}$, is obtained by

$$\hat{\boldsymbol{z}}^{(t+1)} = \boldsymbol{z}^{(t+1)} + \boldsymbol{n}_{q,v^{(t+1)}} \tag{F.8}$$

$$= \hat{\boldsymbol{z}}^{(t)} + \hat{\boldsymbol{v}}^{(t+1)} \tag{F.9}$$

$$= \boldsymbol{z}^{(0)} + \sum_{\tau=1}^{t+1} \hat{\boldsymbol{v}}^{(\tau)}. \tag{F.10}$$

Note that we assume that the initialized $\boldsymbol{z}^{(0)}$ is not quantized, i.e., $\boldsymbol{z}^{(0)}$ is assumed to be transmitted with very high precision (see the following privacy analysis for details).

4.1 Privacy analysis

By inspection of (F.4), the updated $\boldsymbol{x}_i^{(t+1)}$ satisfies

$$\partial f_i(\boldsymbol{x}_i^{(t+1)}) + \sum_{j \in \mathcal{N}_i} \boldsymbol{B}_{i|j} \hat{\boldsymbol{z}}_{i|j}^{(t)} + cd_i \boldsymbol{x}_i^{(t+1)} = 0, \tag{F.11}$$

Algorithm 7 Proposed appproach

[1]
For each node $i \in \mathcal{N}$, initialize $\{\boldsymbol{z}_{j|i}^{(0)}\}_{j\in\mathcal{N}_i}$ based on the desired privacy level (where $\boldsymbol{x}_i^{(0)}$ can be initialized arbitrarily)
Input : $\boldsymbol{x}_i^{(0)}$, $\{\boldsymbol{z}_{j|i}^{(0)}\}_{j\in\mathcal{N}_i}$
Output: $\boldsymbol{x}_i^*$
if $\|\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_i^*\|^2 <$ *threshold* **then**
  Receive $\{\hat{\boldsymbol{v}}_{i|j}^{(t)}\}_{j\in\mathcal{N}_i}$ from all neighbors through non-secure channels (if $t = 0$, receive $\boldsymbol{z}_{i|j}^{(0)}$ through secure channels), update $\{\hat{\boldsymbol{z}}_{i|j}^{(t)}\}_{j\in\mathcal{N}_i}$ using (F.9).
  $\boldsymbol{x}_i^{(t+1)} \leftarrow$ (F.4), $\{\boldsymbol{z}_{j|i}^{(t+1)}\}_{j\in\mathcal{N}_i} \leftarrow$ (F.5), $\{\boldsymbol{v}_{j|i}^{(t+1)}\}_{j\in\mathcal{N}_i} \leftarrow$ (F.6). Quantize $\{\boldsymbol{v}_{j|i}^{(t+1)}\}_{j\in\mathcal{N}_i} \rightarrow \{\hat{\boldsymbol{v}}_{j|i}^{(t+1)}\}_{j\in\mathcal{N}_i}$.
  Send $\hat{\boldsymbol{v}}_{j|i}^{(t+1)}$ to each neighbour $j \in \mathcal{N}_i$.
end

which shows that the private data is only contained in the subgradient $\partial f_i(\boldsymbol{x}_i^{(t+1)})$. As a consequence, the goal of the privacy analysis is to see what information regarding $\partial f_i(\boldsymbol{x}_i^{(t+1)})$ is revealed during the iterations.

For simplicity, assume $\boldsymbol{B}_{i|j} = 1$ for all $j \in \mathcal{N}_i$, and thus $\boldsymbol{B}_{j|i} = -1$. Denote by $\mathcal{N}_{i,c} = \mathcal{N}_i \cap \mathcal{N}_c$ and $\mathcal{N}_{i,h} = \mathcal{N}_i \cap \mathcal{N}_h$ the set of the corrupted and honest neighbors of node i, respectively, and assume that $\mathcal{N}_{i,c} \neq \emptyset$. For node $k \in \mathcal{N}_{i,c}$, using (F.5) and (F.8), we can express the left-hand side of (F.11) as

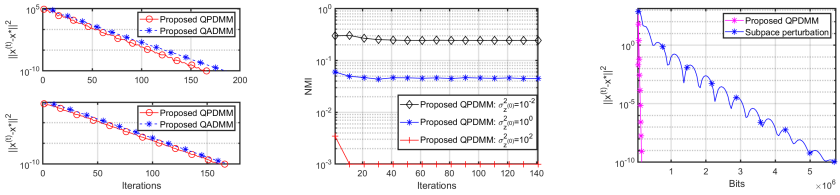
$$\partial f_i(\boldsymbol{x}_i^{(t+1)}) + \sum_{j\in\mathcal{N}_i} \hat{\boldsymbol{z}}_{i|j}^{(t)} - \frac{d_i(\hat{\boldsymbol{z}}_{k|i}^{(t+1)} - \boldsymbol{n}_{q,v_{k|i}^{(t+1)}} - \hat{\boldsymbol{z}}_{i|k}^{(t)}}{2}). \tag{F.12}$$

To quantify the amount of information about the private data $\partial f_i(\boldsymbol{x}_i^{(t+1)})$ learned by the adversaries, we must first inspect what information is available to them. We first consider the passive adversary. As it can collect all the information available to the corrupted nodes, it has the following knowledge:

$$\{\boldsymbol{x}_i^{(t)}\}_{i\in\mathcal{N}_c, t\in\mathcal{T}} \cup \{\boldsymbol{z}_{i|j}^{(0)}, \hat{\boldsymbol{v}}_{i|j}^{(t+1)}\}_{(i,j)\in\mathcal{E}_c, t\in\mathcal{T}},$$

where $\mathcal{E}_c = \{(i,j) \in \mathcal{E}, (i,j) \notin \mathcal{N}_h \times \mathcal{N}_h\}$ denotes the set of corrupted edges. With the above knowledge, the passive adversary is able to compute both $\sum_{j\in\mathcal{N}_{i,c}} \hat{\boldsymbol{z}}_{i|j}^{(t)}$ using (F.10) and $\frac{1}{2}d_i(\hat{\boldsymbol{z}}_{k|i}^{(t+1)} - \hat{\boldsymbol{z}}_{i|k}^{(t)})$ in (F.12). After computing these known terms, the unknown terms in (F.12) are given by

$$\{\partial f_i(\boldsymbol{x}_i^{(t+1)}) + \sum_{j\in\mathcal{N}_{i,h}} \hat{\boldsymbol{z}}_{i|j}^{(t)} + \frac{d_i}{2}\boldsymbol{n}_{q,v_{k|i}^{(t+1)}}\}_{k\in\mathcal{N}_{i,c}}. \tag{F.13}$$



(a) Output correctness **(b)** Individual Privacy **(c)** Communication cost

Fig. F.1: (a) Output correctness (MSE) in terms of iteration numbers using the proposed privacy-preserving quantized PDMM and ADMM algorithm (QPDMM and QADMM, respectively) for the distributed average consensus (top) and distributed least-squares problem (bottom), (b) individual privacy (normalized mutual information (NMI)) of QPDMM in terms of iteration numbers for three different noise levels $\sigma^2_{z^{(0)}}$ of the auxiliary variables, and (c) the communication cost (bits) comparison of the proposed QPDMM algorithm and the existing subspace perturbation approach [16] under the same noise level $\sigma^2_{z^{(0)}} = 10^4$. Fig. (b) and (c) only show results for the distributed average consensus problem.

Next, we consider the eavesdropping adversary. In the literature eavesdropping is usually tackled by assuming that the communication channels are securely encrypted [29] such that no eavesdropping can be conducted. This assumption is particularly expensive to realize in distributed optimization, as a large number of iterations is often required. To address this problem, we propose that no channel encryption is used, except for transmitting $\boldsymbol{z}^{(0)}$ during the initialization. As a consequence, the eavesdropping adversary can listen to all transmitted messages after initialization, i.e., $\{\hat{\boldsymbol{v}}_{i|j}^{(t+1)}\}_{(i,j)\in\mathcal{E},t\in\mathcal{T}}$ but it does not have knowledge about $\boldsymbol{z}_{i|j}^{(0)}$. Based on (F.10), we can, therefore, deduce $\sum_{\tau=1}^{t}\hat{\boldsymbol{v}}_{i|j}^{(\tau)}$ from $\hat{\boldsymbol{z}}_{i|j}^{(t)}$ in (F.13) as it is known to the eavesdropping adversary. Consequently, we conclude that all what the passive and eavesdropping adversaries observe about the honest node i is given by

$$\{\partial f_i(\boldsymbol{x}_i^{(t+1)}) + \sum_{j\in\mathcal{N}_{i,h}} \boldsymbol{z}_{i|j}^{(0)} - \frac{d_i}{2}\boldsymbol{n}_{q,v_{k|i}^{(t+1)}}\}_{k\in\mathcal{N}_{i,c}} \tag{F.14}$$

where the last term $\{\boldsymbol{n}_{q,v_{k|i}^{(t+1)}}\}_{j\in\mathcal{N}_{i,c}}$ will converge to the all-zero vector as the iterations proceed. By applying Proposition 5 to (F.14), the term $\sum_{j\in\mathcal{N}_{i,h}}\boldsymbol{z}_{i|j}^{(0)}$ can be seen as noise, which can be made arbitrarily large at the initialization step to protect the private data $\partial f_i(\boldsymbol{x}_i^{(t+1)})$ from being revealed. Therefore, arbitrarily small information leakage regarding $\partial f_i(\boldsymbol{x}_i^{(t+1)})$ can be achieved at every iteration.

4.2 Individual privacy guarantee

We conclude that the conditions to guarantee the privacy of the data of honest node $i \in \mathcal{N}_h$ for both eavesdropping and passive adversaries are given by:

- There is at least one honest neighbor. That is, $\mathcal{N}_{i,h} \neq \emptyset$.
- The communication channels are encrypted in the initialization phase when transmitting $\boldsymbol{z}^{(0)}$.

4.3 Output correctness and communication cost

In [28] it has been shown that if the sequence $\{\boldsymbol{n}_{q,v^{(t)}}\}_{t\in\mathbb{T}}$ is finitely summable, then Douglas-Rachford splitting will convergence to a fixed point $\boldsymbol{x}^*$ which is the solution to (F.1). However, due to the quantization, the communication cost of the proposed approach is substantially reduced. The details of the proposed algorithm are summarized in Algorithm 7.

5 Numerical results

To demonstrate the desirable properties of the proposed approach, we simulated a geometric network with $n = 30$ nodes where every two nodes are allowed to transmit messages if their distance is within a radius of $\sqrt{\frac{\log(n)}{n}}$, as this condition ensures that the corresponding graph is connected with high probability [30]. For the proposed quantized approach, a one-bit (mid-rise) quantizer is used with step-size $\Delta^{(t)}$, which means that we only transmit the signs of the $\boldsymbol{z}_{i|j}$s which will be reconstructed at the receiver by $\pm\Delta^{(t)}/2$. In all experiments, we randomly draw the private data from a zero-mean Gaussian distribution with unit variance. In addition, we set $c = \gamma = 0.9$ and the auxiliary variable $\boldsymbol{z}^{(0)}$ is initialized with zero-mean Gaussian distributed noise having a variance $\sigma^2_{z^{(0)}} = {\Delta^{(0)}}^2$, where $\Delta^{(0)}$ is the initial quantization step-size.

We demonstrate the performance in terms of the three requirements mentioned in Section 3.2:

(1) **Output correctness**: Fig.F.1 (a) shows simulation results for both ADMM and PDMM for two applications, distributed average consensus (top plot) and distributed least squares (bottom plot). We see that both PDMM and ADMM $\boldsymbol{x}^{(t)}$ converge to the optimum $\boldsymbol{x}^*$. Hence, the proposed approaches satisfy the output correctness requirement, i.e., accuracy is not compromised by considering both quantization and privacy.

(2) **Individual privacy**: Fig.F.1 (b) shows the individual privacy over iterations, i.e., the information loss in (F.10), of the proposed approach when applied to the distributed average consensus problem. We can see that the larger $\sigma^2_{z^{(0)}}$, the less individual privacy is revealed. Hence, the desired individual privacy can be guaranteed by the proposed approach.

(3) **Communication cost**: Fig. F.1 (c) demonstrates that the proposed approach circumvents the trade-off between privacy and communication cost incurred in privacy-preserving distributed optimization, e.g., the subspace perturbation approach [16], in which we compare their communication costs for

distributed average consensus under the same noise level, thus also the privacy level. The communication cost of the latter algorithm is given by $T(2m)b$, where T is the total number of iterations, $2m$ is the total amount of messages transmitted at each iteration (d_i per node) and $b = 64$ the number of bits needed to represent each message (MATLAB double precision floating-point format). Note that $b = 1$ for the proposed algorithm. As expected, the proposed approach significantly reduces the communication costs without compromising both accuracy and privacy.

One remark is placed here. Other than the subspace perturbation approach, there are also other noise insertion approaches like secret sharing [10, 11, 31–33] and differential privacy algorithms [3–9]. In future work it would be interesting to investigate how quantization affects their performances in terms of privacy and accuracy.

6 Conclusion

In this paper, we proposed a novel yet general communication efficient privacy-preserving distributed optimization approach using adaptive quantization. By adopting an adaptive quantizer that dynamically decreases its cell-width for each iteration, we are able to alleviate the trade-off between privacy and communication cost without compromising the algorithm accuracy. The algorithm is able to protect privacy of any honest node against the passive adversary requiring only one honest neighboring node. Moreover, the proposed method is computationally very lightweight in its way of dealing with an eavesdropping adversary as no secure encryption is needed, except for in the initialization step. Numerical results were conducted, which confirm the desirable properties of the proposed approach in terms of accuracy, privacy and communication cost.

References

[1] M. Anderson, *Technology device ownership, 2015*, Pew Research Center, 2015.

[2] J. Poushter and others, "Smartphone ownership and internet usage continues to climb in emerging economies," *Pew Research Center*, vol. 22, pp. 1–44, 2016.

[3] Z. Huang, S. Mitra, and N. Vaidya, "Differentially private distributed optimization., pp. 1–10," in *Proc. Int. Conf. Distrib. Comput. Netw*, 2015.

[4] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Trans. Autom. Control., vol. 62, no. 1, pp 50-64*, 2016.

[5] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private distributed convex optimization via functional perturbation," *IEEE Trans. Control Netw. Syst., vol. 5, no. 1, pp 395-408*, 2018.

[6] T. Zhang and Q. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Trans. Inf. Forensics Security, vol. 12, no. 1, pp. 172–187*, 2016.

[7] X. Zhang, M. M. Khalili, and M. Liu, "Recycled ADMM: Improve privacy and accuracy with less computation in distributed algorithms," in *in Proc. 56th Annu. Allerton Conf. Commun., Control, Comput. pp.959–965*, 2018.

[8] X. Zhang, M. M. Khalili, and M. Liu, "Improving the privacy and accuracy of ADMM-based distributed algorithms," *Proc. Int. Conf. Mach. Lear. pp. 5796–5805*, 2018.

[9] Y. Xiong, J. Xu, K. You, J. Liu and L. Wu, "Privacy preserving distributed online optimization over unbalanced digraphs via subgradient rescaling," *IEEE Trans. Control Netw. Syst.*, 2020.

[10] K. Tjell and R. Wisniewski, "Privacy preservation in distributed optimization via dual decomposition and ADMM," in *Proc. IEEE 58th Conf. Decis. Control., pp. 7203–7208*, 2020.

[11] K. Tjell, I. Cascudo and R. Wisniewski, "Privacy preserving recursive least squares solutions," in *Proc. Eur. Control Conf., pp.3490–3495*, 2019.

[12] R. Cramer, I. B. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*, Cambridge University Press, 2015.

[13] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO, pp. 643–662*. Springer, 2012.

[14] Q. Li, R. Heusdens and M. G. Christensen, "Convex optimisation-based privacy-preserving distributed average consensus in wireless sensor networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 5895-5899*, 2020.

[15] Q. Li, R. Heusdens and M. G. Christensen, "Convex optimization-based privacy-preserving distributed least squares via subspace perturbation," in *Proc. Eur. Signal Process. Conf., pp. 2110-2114*, 2021.

[16] Q. Li, R. Heusdens and M. G. Christensen, "Privacy-preserving distributed optimization via subspace perturbation: A general framework," in *IEEE Trans. Signal Process., vol. 68, pp. 5983 - 5996*, 2020.

[17] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning, vol. 3, no. 1, pp. 1–122*, 2011.

[18] G. Zhang and R. Heusdens, "Bi-alternating direction method of multipliers over graphs," in *Proc. Int. Conf. Acoustics, Speech, Signal Proc.*, Brisbane (Australia), 2015, IEEE.

[19] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process., vol. 4, no. 1, pp. 173–187*, 2018.

[20] T. Sherson, R. Heusdens, W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 2, pp 334-347*, 2018.

[21] E. Ryu, S. P. Boyd, "Primer on monotone operator methods," *Appl. Comput. Math., vol. 15, no. 1, pp. 3-43,*, 2016.

[22] T. M. Cover and J. A. Tomas, *Elements of information theory*, John Wiley & Sons, 2012.

[23] M. Lopuhaä-Zwakenberg, B. Škorić and N. Li, "Information-theoretic metrics for local differential privacy protocols," *arXiv preprint arXiv:1910.07826*, 2019.

[24] Q. Li, J. S. Gundersen, R. Heusdens and M. G. Christensen, "Privacy-preserving distributed processing: Metrics, bounds, and algorithms," in *IEEE Trans. Inf. Forensics Security.*, 2021.

[25] P. Cuff and L. Yu, "Differential privacy as a mutual information constraint," in *Proc. 23rd ACM SIGSAC Conf. Comput. Commun. Secur., pp 43–54*, 2016.

[26] Q. Li, M. Coutino, G. Leus and M. G. Christensen, "Privacy-preserving distributed graph filtering," in *Proc. Eur. Signal Process. Conf., pp. 2155-2159*, 2021.

[27] D. H. M. Schellekens, T. Sherson, and R. Heusdens, "Quantisation effects in PDMM: A first study for synchronous distributed averaging," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 4237–4241*, 2017.

[28] J. A. G. Jonkman, T. Sherson, and R. Heusdens, "Quantisation effects in distributed optimisation," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 3649–3653*, 2018.

[29] D. Dolev, C. Dwork, O. Waarts, M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach., vol. 40, no. 1, pp. 17-47,*, 1993.

[30] J. Dall and M. Christensen, "Random geometric graphs," *Physical review E, vol. 66, no. 1, pp. 016121*, 2002.

[31] N. Gupta, J. Katz, N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine, vol. 50, no. 1, pp. 9515-9520*, 2017.

[32] N. Gupta, J. Kat and N. Chopra, "Statistical privacy in distributed average consensus on bounded real inputs," in *ACC, pp 1836-1841*, 2019.

[33] Q. Li, I. Cascudo, and M. G. Christensen, "Privacy-preserving distributed average consensus based on additive secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

Paper G

Privacy-Preserving Distributed Graph Filtering

Qiongxiu Li, Mario Coutino, Geert Leus, Mads Græsbøll Christensen

The paper has been published in the
Proc. Eur. Signal Process. Conf. (EUSIPCO) 2021

The layout has been revised.

Abstract

With an increasingly interconnected and digitized world, distributed signal processing and graph signal processing have been proposed to process its big amount of data. However, privacy has become one of the biggest challenges holding back the widespread adoption of these tools for processing sensitive data. As a step towards a solution, we demonstrate the privacy-preserving capabilities of variants of the so-called distributed graph filters. Such implementations allow each node to compute a desired linear transformation of the networked data while protecting its own private data. In particular, the proposed approach eliminates the risk of possible privacy abuse by ensuring that the private data is only available to its owner. Moreover, it preserves the distributed implementation and keeps the same communication and computational cost as its non-secure counterparts. Furthermore, we show that this computational model is secure under both passive and eavesdropping adversary models. Finally, its performance is demonstrated by numerical tests and it is shown to be a valid and competitive privacy-preserving alternative to traditional distributed optimization techniques.

1 Introduction

Modern systems routinely gather large-scale data from different individuals and then draw inferences from these data. To this end, graph signal processing (GSP) has been put forth and proven effective for processing large amounts of networked data by exploiting their inherent structural information [1]. However, to process these networked data in a distributed manner, data exchange among different nodes is required. As the underlying data usually contains sensitive information about each individual node/agent in the network, the data exchanges may raise privacy concerns for example: 1) insecure communication channels which expose the data to eavesdroppers; 2) the trust issues that appear when individuals agree to participate in a distributed computation but are reluctant to reveal their own data to others. In fact, as shown in [2], the identities of individuals and their data are inseparable. For example, with only an anonymous 10-ride bus ticket, the identity of a specific bus passenger can be revealed [3]. Therefore, developing efficient techniques for processing large volumes of data in a privacy-friendly manner is nowadays required, posing new challenges that need to be overcome.

The insecure communication channel concern is usually resolved by assuming securely encrypted channels [4]. The trust issue, on the other hand, is particularly challenging as inferences on the exchanged data are required and classical channel encryption is insufficient. Existing privacy-preserving algorithms for solving such trust issues can be broadly classified into two classes based on established security models: computational and information-theoretical. The first type comprises computationally secure algorithms which ensure privacy

through the assumption of computational hardness; that is, the malicious adversary is assumed to be computationally limited thus the secrets cannot be reconstructed efficiently. These algorithms usually adopt popular techniques like homomorphic encryption (HE) [5] and garbled circuit (GC) [6] from secure multiparty computation (SMPC) [7] to achieve privacy-preserving data aggregation [8, 9]. The privacy of the nodes/agents is protected as all the data are first encrypted and then the computations are conducted in the encrypted domain. However, these techniques are usually computationally demanding, and are thus hard to apply in practice.

On the other hand, the information-theoretical security model addresses the privacy issues through an information theory point of view. Distinctly from the computational hardness assumption, it assumes a computationally unlimited adversary and that privacy is achieved only if the information obtained by the adversary just leads to a slightly better (or the same) posterior guess of the private data compared to the prior. Information-theoretical security is usually achieved by obfuscating the private data through noise insertion, it is thus computationally lightweight and has been used in various fields through different noise insertion methods, e.g., zero-sum noise insertion for distributed average consensus [10–12]; differentially private Kalman filtering [13]; secret sharing based recursive least squares [14]; subspace noise insertion using distributed optimization [15, 16]. However, these algorithms suffer from a heavy communication overhead as a large number of iterations is required for convergence.

As a first step towards providing both computational and communication efficient privacy-preserving networked data processing, in this paper, we focus on addressing the privacy issues in the context of distributed graph filtering, the building block of GSP. As the conventional distributed graph filtering contains mainly two parts: offline learning conducted by a trusted third party (TTP) and online distributed computation by network nodes, we, therefore, propose a complete framework to avoid possible privacy abuse in both the offline and online steps. In the offline step, the TTP receives only the encryption seeds from the nodes and not their private data; while in the online step, an information-theoretical security model is achieved through noise insertion, which protects the private data of each node from being revealed to others in the network.

2 Distributed Processing Over Networks

Consider an N-dimensional signal $\boldsymbol{x}$ residing on a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{N} = \{1, \ldots, N\}$ denotes the set of N nodes and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ denotes the set of M edges. Further, let $\mathcal{N}_i = \{j | (i,j) \in \mathcal{E}\}$ denote the neighbourhood of the i-th node. The goal of distributed processing over networks is to compute a transformation, $\mathcal{H}$, of the networked data, $\boldsymbol{x}$, i.e.,

$$\boldsymbol{y} = \mathcal{H}(\boldsymbol{x}), \tag{G.1}$$

in a distributed manner; that is, only employing local data exchanges, i.e., data exchanges among neighboring nodes.

Although $\mathcal{H}$ can take many forms, e.g., optimization problems [17], linear or non-linear transforms [18], here, we focus on transformations that are linear, or that can be properly approximated by a linear transformation, i.e.,

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} \tag{G.2}$$

such as the consensus operation, i.e., $\boldsymbol{H} = \mathbf{1}\mathbf{1}^\top$, which are pervasive in typical network processing tasks, e.g., denoising [19], interpolation [20]. A popular way to implement (or approximate) $\boldsymbol{H}$ is to express it as a K-th order polynomial of a matrix representation of $\mathcal{G}$, i.e.,

$$\boldsymbol{H}_{\mathrm{c}} \triangleq \sum_{k=0}^{K} \phi_k \boldsymbol{S}^k, \tag{G.3}$$

where $\{\phi_k \in \mathbb{R}\}_{k=0}^{K}$ are the filter coefficients; and $\boldsymbol{S}$ is the so-called graph shift operator in GSP [1]. Common choices of $\boldsymbol{S}$ are the weighted graph adjacency matrix $\boldsymbol{W}$ and the graph Laplacian matrix $\boldsymbol{L}$. By construction, $\boldsymbol{S}$ is an $N \times N$-symmetric matrix for undirected graphs and carries the notion of frequency in the graph setting through its eigen decomposition [1, 21]. The matrix polynomial in (G.3) is typically referred to as a classical (node-invariant) FIR graph filters. Note that by making use of the local structure of $\boldsymbol{S}$, the computation of (G.3) can be implemented in a distributed way [22, 23], where each node can locally compute the k-th shift of $\boldsymbol{x}$ by exchanging its previous shifted versions within its neighbourhood, i.e., $\boldsymbol{S}^k\boldsymbol{x} = \boldsymbol{S}(\boldsymbol{S}^{k-1}\boldsymbol{x})$. This distributed implementation is particularly beneficial in saving the communication overhead and computational complexity because a FIR graph filter of order K incurs in a cost of $\mathscr{O}(MK)$.

Though the computational and communication cost scales linearly with K, a large K is usually required to obtain a high approximation accuracy. To alleviate this issue, the constrained edge-variant (CEV) FIR graph filters [24] were proposed to approximate the desired graph filter with a lower order K by endowing it with more degrees of freedom, i.e.,

$$\boldsymbol{H}_{\mathrm{cev}} \triangleq \sum_{k=0}^{K} \boldsymbol{\Phi}_k \boldsymbol{S}^k, \tag{G.4}$$

where the coefficient matrices $\{\boldsymbol{\Phi}_k \in \mathbb{R}^{N\times N}\}_{k=0}^{K}$ denote the weights that each node assigns to its edges at shift k, and they share the support with $\boldsymbol{S} + \boldsymbol{I}_N$, where $\boldsymbol{I}_N$ is the $N \times N$-identity matrix. It is easy to see that (G.4) reduces to the classical FIR graph filter if $\boldsymbol{\Phi}_k = \phi_k \boldsymbol{I}_N$. And it also reduces to the node-variant FIR graph filter [23] if $\{\boldsymbol{\Phi}_k\}_{k=0}^{K}$ are diagonal matrices. Note that while the CEV FIR filters (G.4) still enjoy a distributed implementation and the communication and computational cost remains $\mathscr{O}(MK)$, they allow for a richer family of linear operators that can be approximated as they are not restricted to linear operators that commute with the graph shift operator.

3 Privacy-Preserving Processing

In this section, we first highlight the privacy concerns of distributed data processing using current FIR graph filters and introduce the adversary models. The latter is an important issue when designing privacy-preserving protocols. We then formally state the problem addressed in this work.

3.1 Privacy concern

Generally, privacy is associated with individuals and their personal data. Thus, we identify the privacy concern in graph filtering is to protect the input graph signal x_i of each node from being revealed. This is motivated by the fact that this data usually contains sensitive information about the individual like health condition or political views. Unfortunately, this privacy concern is not addressed in current distributed FIR graph filters, since the private data x_i is propagated through the network and exposed to neighbors and eavesdroppers.

3.2 Adversary model

To evaluate the robustness of a system under different security attacks, we consider the so-call adversary model. An adversary can be both external or internal to the network, and aims to conduct certain malicious activities such as inferring the private data by controlling a number of nodes. These controlled nodes are referred to as corrupted nodes and the others are called non-corrupted or honest nodes. In this paper we will address two widely-used adversary models: eavesdropping and passive. The eavesdropping adversary attempts to infer the private data by eavesdropping the communication channels in the network. The passive adversary, on the other hand, assumes that all the nodes follow the protocol instructions and aims also to infer the private data of honest nodes through the information collected by all the corrupted nodes.

3.3 Problem formulation

Putting things together, we conclude that a privacy-preserving distributed graph filter should be able to protect private data while computing the filter output. To this end, we formulate the problem as follows; *given a linear transform $\boldsymbol{H}$, design an encryption function $E(\boldsymbol{x})$ and related operator $\boldsymbol{H}^e$ that satisfy the following two requirements simultaneously:*

- *Output correctness*: all nodes should be able to achieve the correct filtering output, i.e.,

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} = \boldsymbol{H}^e\boldsymbol{x}^e, \tag{G.5}$$

  where $\boldsymbol{x}^e = E(\boldsymbol{x})$ denotes the encrypted input data.

- *Individual privacy*: after encryption, each node uses the encrypted data x_i^e as its input. In addition, the information theoretic security criterion

$$I(X_i; X_i^e) < \epsilon_i, \forall i \in \mathcal{N}, \tag{G.6}$$

must be met. Here, $I(\cdot)$ denotes mutual information [25]; x_i^e and x_i are assumed to be a realization of random variables X_i^e and X_i, respectively, and ϵ_i is the desired privacy level of node i. Hence, (G.6) guarantees that the encrypted x_i^e reveals asymptotically no information about the private data x_i,

4 Proposed approach

Before moving to our proposed method, let us first consider the following straightforward design for both $E(\boldsymbol{x})$ and $\boldsymbol{H}^e$. The privacy of each node can be preserved by considering the following rendition of the edge-variant graph filter [c.f. (G.4)], i.e.,

$$\boldsymbol{H}_{\text{rev}} = \sum_{k=0}^{K} \boldsymbol{S}^k \boldsymbol{\Phi}_k. \tag{G.7}$$

Here, each node exploits the graph filter coefficients as encryption seeds to mask its private data and sends the masked data to its neighbours. Note that (G.4) and (G.7) are equivalent when $\boldsymbol{\Phi}_k \boldsymbol{S} = \boldsymbol{S}\boldsymbol{\Phi}_k \ \forall k$ and that the graph filter (G.7) can also be implemented distributively. However, there is a price to pay: instead of having a communication cost $\mathcal{O}(KM)$, the implementation now has an overall communication cost of $\mathcal{O}(K^2M)$. Unfortunately, for this case, although the above-mentioned requirements can be achieved, there seems to be no free lunch and communication complexity has to be sacrificed.

To alleviate the communication overhead, we now proceed to introduce another choice for both $E(\boldsymbol{x})$ and $\boldsymbol{H}^e$ meeting the above-mentioned requirements. We further analyze their performance in two widely-used adversary models.

4.1 Encryption function design

One typical way to maintain privacy is to obfuscate the private data by inserting noise since it is computationally lightweight. We, therefore, propose to design the encryption function using multiplicative noises as

$$E(\boldsymbol{x}) = \text{diag}(\boldsymbol{e})\boldsymbol{x}, \tag{G.8}$$

where $\boldsymbol{e} = [e_1, \ldots, e_N]^T \in \mathbb{R}^N$ is the encryption vector. Thus the encrypted data of node i is given by $x_i^e = e_i x_i$. Let e_i denote a realization of random vari-

able E_i having differential entropy $h(E_i)$, assuming it exists[1]. The concerned individual privacy for node i is thus $I(X_i; X_iE_i) = h(X_i) - h(X_i|X_iE_i)$. To guarantee privacy, we need the following result.

Proposition 6. *Consider X and Y as independent continuous random variables with mean $\mu(X), \mu(Y) < \infty$ and variance $0 < \mathrm{var}(X) < \mathrm{var}(Y) < \infty$, and let $Z = XY$. Then*

$$\lim_{\mathrm{var(Y)}\to\infty} I(X; Z) = 0,$$

assuming $I(X; Z)$ exists.

Proof. As X is independent with Y, we have $\mathrm{var}(Z) = \mathrm{var}(X)\mathrm{var}(Y) + \mathrm{var}(X)\mu^2(Y) + \mathrm{var}(Y)\mu^2(X)$. Let $\gamma = 1/(\mathrm{var}(Z))^{\frac{1}{2}}$ and define $Z' = \gamma Z$. Hence, Z' has unit variance. We have $I(X; Z) = I(\gamma X; Z')$ as mutual information is scale-invariant. Thus,

$$\lim_{\mathrm{var(Y)}\to\infty} I(X; Z) = \lim_{\gamma\to 0} I(\gamma X; Z') = I(0; Z') = 0. \qquad \square$$

So, applying Proposition 6 to the problem at hand, we have

$$\lim_{\mathrm{var}(E_i)\to\infty} I(X_i; X_iE_i) = 0. \tag{G.9}$$

We conclude that each node is able to achieve arbitrary small information loss by increasing the variance of its encryption seed. Hence, the privacy requirement in (G.6) is satisfied.

4.2 Encrypted operator design

To fullfil the output correctness requirement, we should design the encrypted operator $\boldsymbol{H}^e$ satisfying $\boldsymbol{H}^e = \boldsymbol{H}\,\mathrm{diag}(\boldsymbol{e})^{-1}$. To this end, we then further approximate the desired encrypted operator $\boldsymbol{H}^e$ using standard distributed graph filters. Here, we use the constrained edge-variant graph filter as an example, which in turn leads to the following convex optimization problem

$$\begin{aligned} \min_{\{\boldsymbol{\Phi}_k\}} \quad & \left\| \boldsymbol{H}^e - \left(\sum_{k=0}^{K} \boldsymbol{\Phi}_k \boldsymbol{S}^k \right) \right\|_F^2 \\ \text{s.t.} \quad & \mathrm{supp}\,\{\boldsymbol{\Phi}_k\} = \mathrm{supp}\{\boldsymbol{S} + \boldsymbol{I}_N\}\ \forall k \in [K], \end{aligned} \tag{G.10}$$

where $\|\cdot\|_F$ is Frobenius norm and $\mathrm{supp}\{\cdot\}$ denotes the support of its argument. To solve the above-mentioned problem, we assume a TTP (cloud/server) to learn the filter coefficients offline. In order to avoid possible data abuse, we have the following assumptions which minimize the amount of information available to each node while still guaranteeing the distributed implementation, i.e.,

[1] If E_i is a discrete random variable, the conditions are given in terms of the Shannon entropy $H(E_i)$.

Algorithm 8 Privacy-preserving distributed FIR graph filters

1: **Offline learning** (secure channels)
2: Each node $i \in \mathcal{N}$ chooses its encryption seed $\boldsymbol{e}_i$ based on its desired privacy level ϵ_i, and sends it to the TTP.
3: The TTP first computes $\boldsymbol{H}^e = \boldsymbol{H}\,\mathrm{diag}(\boldsymbol{e})^{-1}$ and solves the problem (G.10), and then sends the corresponding information $[\boldsymbol{S}]_{i,j\in\mathcal{N}_i\cup\{i\}}, \{[\boldsymbol{\Phi}_k]_{i,j\in\mathcal{N}_i\cup\{i\}}\}_{k=0}^K$ to node i.
4: **Online distributed computation** (non-secure channels)
5: Each node i initializes $x_i^{(0)} = x_i^e$.
6: **while** $k = 0, \ldots, K$ **do**
7: Collect $x_j^{(k)}$ from all neighbours $j \in \mathcal{N}_i$
8: Store locally $z_i^{(k)} = \sum_{j\in\mathcal{N}_i\cup\{i\}} [\boldsymbol{\Phi}_k]_{ij} x_j^{(k)}$
9: Compute $x_i^{(k+1)} = \sum_{j\in\mathcal{N}_i\cup\{i\}} [\boldsymbol{S}]_{ij} x_j^{(k)}$
10: Send $x_i^{(k+1)}$ to all neighbours $j \in \mathcal{N}_i$
11: **end while**
12: Set $y_i = \sum_{k=0}^K z_i^{(k)}$

Assumption 2. *(Knowledge of the TTP) The TTP has the knowledge of the desired data transformation $\boldsymbol{H}$, the encryption seeds $\boldsymbol{e}$, the graph shift operator $\boldsymbol{S}$ and the filter coefficients $\{\boldsymbol{\Phi}_k\}_{k=0}^K$.*

Assumption 3. *(Knowledge of the nodes) Each node i only has knowledge of its private data x_i, and its corresponding entries of both $\boldsymbol{S}$ and filter coefficients, i.e., $[\boldsymbol{S}]_{i,j\in\mathcal{N}_i\cup\{i\}}, \{[\boldsymbol{\Phi}_k]_{i,j\in\mathcal{N}_i\cup\{i\}}\}_{k=0}^K$.*

Note that Assumptions 3 is motivated similarly as the distributed signal processing [17, 26] that each node only has local knowledge of its neighbourhood. That is, each node does not have the knowledge of the desired data transformation $\boldsymbol{H}$. For example, for the distributed average consensus application, i.e., $\boldsymbol{H} = \frac{1}{N}\mathbf{1}\mathbf{1}^T$, all the nodes would like to reach an agreement over the network but they do not know the size of the network, i.e., N. In the distributed recursive least squares application [14], each node only has its local observations but not the full knowledge of the whole system. After the offline learning, the filter output can be computed distributedly. It is worth to note that the communication and computational cost remains $\mathcal{O}(KM)$. The proposed approach is summarized in Algorithm 0.

4.3 Privacy analysis under adversary models

We now analyze the individual privacy concern under both an eavesdropping and a passive adversary. For an eavesdropping adversary, we assume that the communication required in offline learning is conducted through secure channels. That is, the channels should be securely encrypted [4] when transmitting

the encryption seeds to the TTP and receiving the associated filter coefficients and graph shift operator. As a consequence, the secure channel encryption cost is $2N$. The online distributed computation step, as it is an iterative process, we remark that $\forall k > 0 : X_i \rightarrow X^{(0)} \rightarrow X^{(k)}$ forms a Markov chain where vector $X^{(k)} = [X_1^{(k)}, \ldots, X_N^{(k)}]^\top$ denotes all random variables over the network at iteration k. By the data processing inequality [25] we have

$$I(X_i; X^{(0)}) \geq I(X_i; X^{(k)}),\ \forall k > 0, \tag{G.11}$$

which implies that all the information transmitted in the online step will not reveal additional information. As a consequence, the online step is secure against eavesdropping adversaries without requiring secure channel encryption at all. For the case of a passive adversary, recall Assumptions 2-3, we can see that the corrupted nodes cannot reconstruct the encryption seeds $\boldsymbol{e}$ as $\boldsymbol{H}$ is only known to the TTP. We then conclude that the private data of an honest node is guaranteed even if all other nodes are corrupted (assuming the TTP is not corrupted).

5 Numerical results

We now proceed to present the performance of the proposed approach for approximating user-provided frequency responses and compare it with the distributed optimization technique. We randomly generate a community graph using the GSP toolbox [27] with $N = 40$ nodes and choose the normalized Laplacian as the graph shift operator $\boldsymbol{S}$. Here, we consider as maximum filter order $K = 15$. To show the performance of approximating different frequency responses, we consider two cases commonly used in the GSP community:

1. the exponential kernel
$$h(\lambda) := e^{-\gamma(\lambda-\mu)^2},$$
where γ denotes the spectrum decaying factor and μ is the central parameter;

2. the ideal low pass filter
$$h(\lambda) = \begin{cases} 1 & 0 \leq \lambda \leq \lambda_{\mathrm{c}} \\ 0 & \text{otherwise} \end{cases},$$
where λ_{c} denotes the cutoff frequency.

In Fig. G.1, we demonstrate the normalized approximation error in terms of Frobenius norm between the desired, $\boldsymbol{H}$, and the fitted, $\boldsymbol{H}_{\text{fit}}$, frequency response, i.e., $\|\boldsymbol{H} - \boldsymbol{H}_{\text{fit}}\|_F^2 / \|\boldsymbol{H}\|_F^2$, of the proposed privacy-preserving CEV (p-CEV) filter with the non-private CEV (n-CEV) filter for both scenarios. As we can see, both filters saturate at an order of $K = 7$ but the proposed p-CEV has

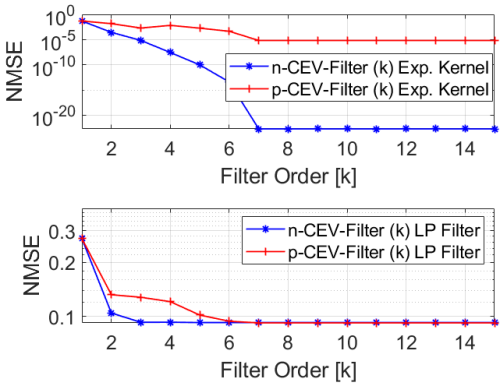


Fig. G.1: Approximation error comparison between the p-CEV and the n-CEV graph filter for different orders. (Top) Results for the exponential kernel. (Bottom) Results for an ideal low-pass filter

a higher error floor for the case of the exponential kernel. And the proposed p-CEV requires a few orders more to achieve the same error floor as the n-CEV for approximating the ideal low pass filter. Overall, we conclude that the proposed approach is able to approximate the desired frequency responses and keeps its private data protected.

In Fig. G.2, we compare the normalized error as a function of iteration number between desired filter output $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}$ and the approximated one $\boldsymbol{y}_{\text{fit}} = \boldsymbol{H}_{\text{fit}}\boldsymbol{x}$ of the proposed p-CEV graph filter and the privacy-preserving alternative using distributed optimization in the average consensus application. In particular, we choose to compare with the privacy-preserving primal-dual method of multipliers (p-PDMM) proposed in [15] as it also achieves information-theoretical security by noise insertion. In both algorithms, we set the noise variance as 100 times the variance of the associated private data, thereby guaranteeing a similar amount of noise perturbation. As we can see, the noise insertion will lead to a high initial error for p-PDMM therefore requiring more iterations to converge. To provide insights into the question on *what is the difference between GSP and distributed optimization* from the privacy-preserving perspective, we compare these two approaches in terms of several important parameters in Table G.1:

1. Both approaches obtain the information-theoretical security model and consider the same adversary models: passive and eavesdropping.

2. Perfect approximation is not possible and thus the proposed approach is not as accurate as the p-PDMM, while the price to pay for the accuracy is the communication cost. The iteration number T of the p-PDMM is usually far larger than the filter order K of the proposed approach, i.e., $T \gg K$.

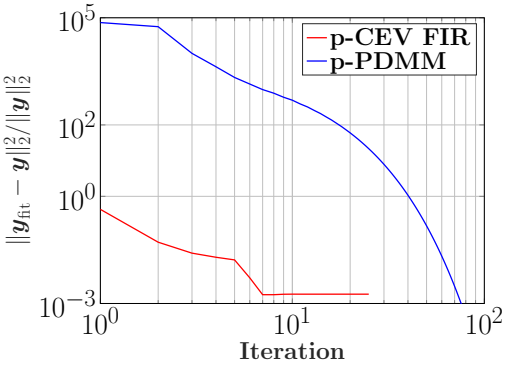


Fig. G.2: Normalized output error versus the number of iterations of the proposed p-CEV filter and the p-PDMM for the average consensus problem

3. Both approaches require secure channels in the initialization step to guarantee the inserted noise cannot be eavesdropped, but the complexity of the proposed approach is usually lower, i.e., $M > N$.

4. The proposed approach assumes that the trusted third party is not corrupted to guarantee the privacy of each node while the p-PDMM requires each honest node has one honest neighbour.

We conclude that the proposed approach is beneficial in solving distributed signal processing tasks in terms of both communication and computational cost. More specifically, the proposed approach is preferred if the required accuracy is not very strict, while the p-PDMM is more attractive if the system requires strictly accurate solution, but with a price of a higher communication cost.

Table G.1: Comparison with distributed optimization

	Proposed	p-PDMM
Security model	Information-theoretic	Information-theoretic
Adversary model	Passive/Eavesdropping	Passive/Eavesdropping
Accuracy	Approximate	Accurate
Communication complexity	$\mathcal{O}(MK)$	$\mathcal{O}(MT)$
Secure channels	$\mathcal{O}(N)$	$\mathcal{O}(M)$
Honest neighbour	No	Yes
Trusted Third Party	Yes	No

6 Conclusions

In this paper, we proposed a communication and computationally efficient solution to achieve privacy-preserving distributed graph filters which allow each

node to compute its desired output and protect its own private data simultaneously. To protect the privacy, each node first inserts noise to obfuscate its private data and sends the obfuscated data to its neighborhood. Numerical tests demonstrated that the proposed approach is able to approximate some desired graph frequency responses with a small filter order and that it is a competitive alternative compared to the privacy-preserving distributed optimization approach in the distributed average consensus problem.

References

[1] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process., vol. 61, no. 7, pp. 1644-1656,*, 2013.

[2] D. Sarwate and K. Chaudhuri, "Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data," *IEEE Signal Process. Magazine, vol. 30, no. 5, pp. 86–94*, 2013.

[3] G. Avoine, L. Calderoni, J. Delvaux, D. Maio, and P. Palmieri, "Passengers information in public transport and privacy: Can anonymous tickets prevent tracking?," *Int. J. Inf. Manage, vol. 34, pp. 682-688*, 2014.

[4] D. Dolev, C. Dwork, O. Waarts, M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach., vol. 40, no. 1, pp. 17-47,*, 1993.

[5] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO, pp. 643–662*. Springer, 2012.

[6] A. C. Yao, "Protocols for secure computations," in *FOCS, pp. 160–164*, 1982.

[7] R. Cramer, I. B. Damgrd, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*, Cambridge University Press, 2015.

[8] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain," in *ICASSP, pp. 7005–7009*, 2013.

[9] M. H. Ruan, M. Ahmad, Y. Q. Wang, "Secure and privacy-preserving average consensus," in *Proc. Workshop Cyber-Phys. Syst. Secur. Privacy, pp. 123-129*, 2017.

[10] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Automat Contr., vol. 62, no. 2, pp. 753–765*, 2017.

[11] J. He, L. Cai, C. Zhao, P. Cheng, X. Guan, "Privacy-preserving average consensus: privacy analysis and algorithm design," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 1, pp. 127–138*, 2019.

[12] N. Gupta, J. Kat and N. Chopra, "Statistical privacy in distributed average consensus on bounded real inputs," in *ACC, pp 1836-1841*, 2019.

[13] K. H. Degue and J. L. Ny, "On differentially private kalman filtering," in *Proc. IEEE Global Conf. Signal Inf. Process., pp. 487-491*, 2017.

[14] K. Tjell, I. Cascudo and R. Wisniewski, "Privacy preserving recursive least squares solutions," in *ECC, pp.3490–3495*, 2019.

[15] Q. Li, R. Heusdens and M. G. Christensen, "Convex optimisation-based privacy-preserving distributed average consensus in wireless sensor networks," in *ICASSP, pp. 5895-5899*, 2020.

[16] Q. Li, R. Heusdens and M. G. Christensen, "Privacy-preserving distributed optimization via subspace perturbation: a general framework," in *arXiv preprint arXiv: 2004.13999*, 2020.

[17] T. Sherson, W. B. Kleijn, R. Heusdens, "A distributed algorithm for robust lcmv beamforming," in *ICASSP, pp. 101-105*, 2016.

[18] Lopes, Cassio G and Sayed, Ali H, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *Trans. Signal Process., vol. 56, no. 7, pp. 3122-3136,*, 2008.

[19] J. Pang, G. Cheung, A. Ortega, O. C. Au, "Optimal graph laplacian regularization for natural image denoising," in *ICASSP, pp 2294-2298*, 2015.

[20] SK Narang, A Gadde, A Ortega, "Signal processing techniques for interpolation in graph structured data," in *ICASSP, pp 5445-5449*, 2013.

[21] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Magazine, vol. 30, no. 3, pp. 83-98,*, vol. 30, no. 3, pp. 83–98, 2013.

[22] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *DCOSS, pp 1–8*, 2011.

[23] S. Santiago, M. Antonio G and R. Alejandro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process., vol. 65, no. 15, pp. 4117-4131*, 2017.

[24] M. Coutino, E. Isufi, G. Leus, "Advances in distributed graph filtering," *IEEE Trans. Signal Process., vol. 67, no. 9, pp. 2320-2333,*, 2019.

[25] T. M. Cover and J. A. Tomas, *Elements of information theory*, John Wiley & Sons, 2012.

[26] T. Sherson, R. Heusdens, W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 2, pp 334-347*, 2018.

[27] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *ArXiv e-prints*, 2014.

Paper H

Privacy-Preserving Distributed Processing: Metrics, Bounds and Algorithms

Qiongxiu Li, Jaron Skovsted Gundersen, Richard Heusdens, and Mads Græsbøll Christensen

The paper has been published in the
IEEE Transactions on Information Forensics and Security, 2021

The layout has been revised.

Abstract

Privacy-preserving distributed processing has recently attracted considerable attention. It aims to design solutions for conducting signal processing tasks over networks in a decentralized fashion without violating privacy. Many existing algorithms can be adopted to solve this problem such as differential privacy, secure multiparty computation, and the recently proposed distributed optimization based subspace perturbation algorithms. However, since each of them is derived from a different context and has different metrics and assumptions, it is hard to choose or design an appropriate algorithm in the context of distributed processing. In order to address this problem, we first propose general mutual information based information-theoretical metrics that are able to compare and relate these existing algorithms in terms of two key aspects: output utility and individual privacy. We consider two widely-used adversary models, the passive and eavesdropping adversary. Moreover, we derive a lower bound on individual privacy which helps to understand the nature of the problem and provides insights on which algorithm is preferred given different conditions. To validate the above claims, we investigate a concrete example and compare a number of state-of-the-art approaches in terms of the concerned aspects using not only theoretical analysis but also numerical validation. Finally, we discuss and provide principles for designing appropriate algorithms for different applications.

1 Introduction

Big data is accompanied by big challenges. Currently, data are collected and simultaneously stored on various local devices, such as phones, tablets and wearable devices [1, 2]. In these cases, three critical challenges exist in processing such large amounts of data: (1) the emerging demand for distributed signal processing tools, as these devices are distributed in nature and often rely on wireless communication to form a network that allows devices to cooperate for solving a problem; (2) the requirement for both computational and communication efficient solutions, due to the fact that these devices are usually resource-constrained, for example in wireless sensor networks; and (3) privacy concerns, as sensors from these devices, such as GPS and cameras, usually contain sensitive personal information. Consequently, having efficient privacy-preserving distributed processing solutions, which are able to address the privacy concerns, is highly important and usually requires interdisciplinary research across fields such as distributed signal processing, information theory and cryptography.

There are two primary types of security models: (1) computational security, in which the adversary is assumed to be computationally bounded such that it cannot decrypt a secret efficiently (i.e., in polynomial time) and (2) information-theoretic security, in which the adversary is assumed to be computationally unbounded but does not have sufficient information for inferring the

secret. In this paper we focus on information-theoretic security since it assumes a stronger adversary and is more efficient in terms of both communication and computational demands [3].

1.1 Related works

Many information-theoretic approaches have been proposed for addressing privacy issues in various distributed processing problems like distributed average consensus [4–16], distributed least squares [17, 18], distributed optimization [19–27] and distributed graph filtering [28]. These approaches can be broadly classified into three classes. The first two classes combine distributed signal processing with commonly used cryptographic tools, such as secure multiparty computation (SMPC) [29, 30], and privacy primitives, such as differential privacy (DP) [31, 32], respectively. The third class directly explores the potential of existing distributed signal processing tools for privacy preservation, such as distributed optimization based subspace perturbation (DOSP) [7, 18, 27]. Among these approaches, SMPC aims to securely compute a function over a number of parties' private data without revealing it. DP, on the other hand, is defined to add noise to ensure that the posterior guess relating to the private data is only slightly better (quantified by the parameter ϵ) than the prior guess. DOSP protects the private data by inserting noise in a specific subspace depending on the graph topology.

Even though all the above mentioned algorithms can in principle be applied in distributed processing, it is still very challenging to design an appropriate algorithm given a specific application at hand. For example, whether choosing one single algorithm is good enough or if we should combine them to have a hybrid approach. The main difficulty comes from the fact that the metrics of these approaches are different and are defined based on different motivations and contexts. There are cases where these approaches are mutually exclusive. For example, it has been shown that, in distributed average consensus applications, the exact average result and differential privacy cannot be achieved simultaneously [10]. This implies that a DOSP or a perfect SMPC protocol, which guarantees accurate results, can never be differentially private in distributed average consensus. Another issue is that the privacy defined by these approaches might not be the same as the individual privacy defined in the context of distributed processing. For example, a perfect SMPC protocol does not necessarily guarantee that no private information is revealed (see Section 4.1). In addition, a perfect DP based approach ($\epsilon = 0$) also does not guarantee that no private information is revealed if the private data are correlated [33] (see Section 4.2). Therefore, it is highly desired to have general metrics that are able to compare and relate these algorithms in a consistent fashion, so that appropriate privacy-preserving distributed algorithms can be designed based on their performance and underlying assumptions.

In addition to the above mentioned challenges in algorithm design, another challenge lies in how to analyze the algorithm performance in a distributed

setting. Due to the fact that distributed processing algorithms are usually iterative, it is complex to analytically track the privacy analysis over the iterations.

1.2 Paper contributions

In this paper, we attempt to solve the above mentioned problems. The main contributions of this paper can be summarized as follows:

- To the best of our knowledge, this is the first paper proposing formal and general information-theoretic metrics for quantifying privacy-preserving distributed processing algorithms in terms of output utility and individual privacy. Additionally, we prove that existing well-known metrics in SMPC and DP can be considered special cases of the proposed metrics under certain assumptions/conditions. Moreover, by analyzing the lower bound on individual privacy which provides insights on the nature of a problem, we give suggestions and discuss principles on how to design appropriate algorithms.
- We demonstrate how to analyze, quantify, compare, and understand the nature of a number of existing privacy-preserving distributed processing algorithms including DP, SMPC and DOSP.

1.3 Outline and notation

This paper is organized as follows. Section 2 introduces fundamentals and states the problem to be solved. Section 3 introduces the proposed metrics. Section 4 relates the well-known SMPC and DP to the proposed metrics. Sections 5 and 6 describe a concrete example of distributed average consensus. The former section defines the problem and shows that traditional approaches leak privacy, while the latter section first presents a theoretical result for achieving privacy-preservation and then analyzes existing privacy-preserving distributed average consensus algorithms using the proposed metrics. Numerical validations are given in Section 7. Section 8 gives suggestions on algorithm design and Section 9 concludes the paper.

The following notations are used in this paper. We will use lowercase letters (x) for scalars, lowercase boldface letters $(\boldsymbol{x})$ for vectors, uppercase boldface letters $(\boldsymbol{X})$ for matrices, overlined uppercase letters $(\bar{X})$ for subspaces, calligraphic letters $(\mathcal{X})$ for arbitrary sets and $|\cdot|$ for the cardinality of a set. Uppercase letters (X) denote random variables having realizations x. $\mathrm{span}\{\cdot\}$ and $\mathrm{null}\{\cdot\}$ denote the span and nullspace of their argument, respectively. $(\boldsymbol{X})^\top$ denotes the transpose of $\boldsymbol{X}$. x_i denotes the i-th entry of the vector $\boldsymbol{x}$ and $\boldsymbol{X}_{ij}$ denotes the (i,j)-th entry of the matrix $\boldsymbol{X}$. $\mathbf{0}$, $\mathbf{1}$ and $\boldsymbol{I}$ denote the vectors with all zeros and all ones, and the identity matrix of appropriate size, respectively.

2 Preliminaries

In this section, we first introduce the problem setup and the adversary models. After that we summarize the key aspects to be considered when evaluating an algorithm.

2.1 Privacy-preserving distributed processing over networks

A network can be modelled as a graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$ where $\mathcal{N} = \{1, \ldots, n\}$ denotes the set of n nodes and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ denotes the set of m (undirected) edges. Note that node i and j can communicate with each other only if there is an edge between them, i.e., $(i, j) \in \mathcal{E}$. Let $\mathcal{N}_i = \{j \mid (i, j) \in \mathcal{E}\}$ denote the neighborhood of node i and $d_i = |\mathcal{N}_i|$, called the degree of node i. Assume each node i has private data s_i and let $\boldsymbol{s} = [s_1, \ldots, s_n]^\top$. Note that for simplicity, s_i is assumed to be scalar but the results can easily be generalized to arbitrary dimensions.

The goal of privacy-preserving distributed processing over a network is to compute a function

$$f : \mathbb{R}^n \mapsto \mathbb{R}^n, \boldsymbol{y} = f(\boldsymbol{s}), \tag{H.1}$$

in a distributed manner without revealing each node's private data s_i to other nodes, where y_i denotes the desired output of node i. By a distributed manner we mean that only data exchange between neighboring nodes is allowed.

2.2 Adversary models

Adversary models are used to evaluate the robustness of the system under different security attacks. In this paper, we consider two types of adversary models: the passive and eavesdropping model.

Passive adversary

The passive adversary model is a typical model to be addressed in distributed networks [34]. It works by colluding a number of nodes to infer the private data of the other nodes. These colluding nodes are referred to as corrupted nodes, and the others are called honest nodes. The corrupted nodes are assumed to follow the algorithm instructions (called the protocol) but will share information together to infer the private data of the honest nodes. We call an edge in the graph corrupted when there is one corrupted node at its ends, see Fig. H.1 for a toy example. Hence, all the messages transmitted along such an edge will be known to the passive adversary. In the following, we will denote $\mathcal{N}_c$ and $\mathcal{N}_h$ as the set of corrupted nodes and honest nodes, respectively. Additionally, we will denote $\mathcal{E}_c = \{(i, j) \in \mathcal{E} : (i, j) \notin \mathcal{N}_h \times \mathcal{N}_h\}$ as the set of corrupted edges. An algorithm is more robust if it can tolerate more corrupted nodes without revealing the private data of the honest nodes.

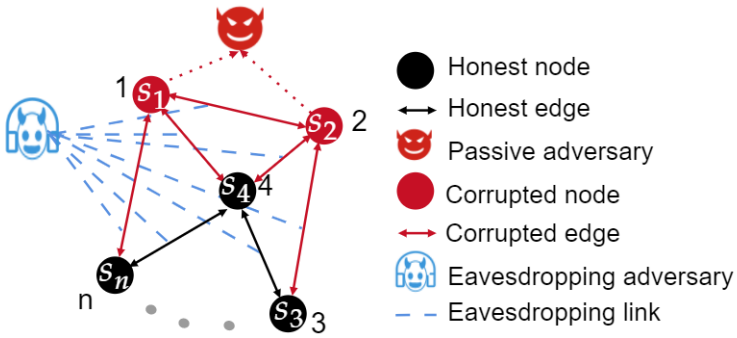


Fig. H.1: System setup and adversary models.

Eavesdropping adversary

The eavesdropping adversary, on the other hand, is assumed to listen to all communication channels, i.e., edges, between nodes with the purpose of inferring the private data. This model is relatively unexplored in the context of privacy-preserving distributed processing. The main reason is that many SMPC based approaches, such as those based on secret sharing [17, 19, 35], assume that all messages are transmitted through securely encrypted channels [36] so that the transmitted messages cannot be eavesdropped. However, channel encryption is computationally demanding for iterative approaches like the distributed processing algorithms considered here, since the channels are used many times before the algorithm converges. As a consequence, the cost for channel encryption is also an important factor to be considered when designing privacy-preserving algorithms.

Throughout this paper we will assume that these two adversaries cooperate. That is, they will share information together to increase the chance of inferring the private data of the honest nodes.

2.3 Key aspects for algorithm evaluation

We will evaluate the performance of privacy-preserving distributed processing algorithms in terms of the following two aspects: output utility and individual privacy.

Output utility

Let $\hat{\boldsymbol{y}} \in \mathbb{R}^n$ denote the estimated output of a privacy-preserving distributed processing algorithm. For each node i, the output utility should measure how close the estimate $\hat{y}_i$ is to its desired output y_i.

Individual privacy

Based on the definition of the adversary models, the corrupted nodes are willing to share their private data to the passive adversary. Therefore, privacy is only relevant for the honest nodes. The individual privacy of honest node $i \in \mathcal{N}_h$ should measure how much information regarding its private data s_i is revealed to the adversaries, both passive and eavesdropping, given all the information available to them.

In next section we will introduce the proposed metrics for quantifying the output utility and individual privacy.

3 Proposed metrics

In this section we will introduce the proposed metrics. We first motivate why we adopt mutual information for defining these metrics and then give details on how to quantify both the output utility and individual privacy stated above.

3.1 Motivation of using mutual information

To quantify the privacy for information-theoretic approaches, a natural language is to use information theory. For an overview of information-theoretic metrics the reader is referred to [37]. In the context of privacy-preserving distributed processing, two types of metrics are widely adopted: mutual information and ϵ-DP (their definitions will be given later in Section 3.2 and 4.2, respectively). The reasons for choosing mutual information over ϵ-DP are:

(1) ϵ-DP is very difficult to realize in practice as it is a worst-case metric that provides strong privacy assurance in any situation, e.g., for all prior distributions of the private data [38–40]. Mutual information is easier to implement in practice as it can be seen as a relaxed version of ϵ-DP [41].

(2) The privacy measured by ϵ-DP only reflects the privacy in the worst-case scenario which can be very far from the typical privacy of the average users; mutual information, on the other hand, is more preferred in quantifying the privacy of the average users [42].

(3) ϵ-DP has problems in working with correlated data [33].

To quantify the output utility, we also adopt mutual information as the metric because it has been widely used in the literature [43, 44].

3.2 Definition of mutual information

Let X denote a continuous random variable with probability density function $f_X(x)$ and differential entropy $h(X) = -\int f_X(x) \log f_X(x) dx$, assuming it exists. Given a random variable Y, the conditional entropy $h(X|Y)$ quantifies how much uncertainty is remained in X after knowing Y. The mutual information $I(X;Y)$ [45] measures the dependence between X and Y. It quantifies

how much information can be learned about X after knowing Y, or vice versa, which is given by[1]

$$I(X;Y) = h(X) - h(X|Y). \tag{H.2}$$

3.3 Output utility u_i

We quantify the output utility as:

$$\forall i \in \mathcal{N} : u_i = I(Y_i; \hat{Y}_i). \tag{H.3}$$

Hence $0 \leq u_i \leq I(Y_i; Y_i)$ where $u_i = I(Y_i; Y_i)$ implies prefect output utility.

3.4 Individual privacy ρ_i

Let $\mathcal{V}$ denote the set of random variables containing all the information collected by the adversaries throughout the whole algorithm. The individual privacy of honest node i quantifies the amount of information about the private data s_i learned by the adversaries, which we define as

$$\forall i \in \mathcal{N}_h : \rho_i = I(S_i, \mathcal{V}), \tag{H.4}$$

and we conclude that $0 \leq \rho_i \leq I(S_i; S_i)$. The smaller ρ_i, the more private the data is. Given the definition of the adversary models, we conclude that the adversaries always have knowledge of the private data $\{s_j\}_{j\in\mathcal{N}_c}$ and estimated outputs $\{\hat{y}_j\}_{j\in\mathcal{N}_c}$, regardless of the algorithm used. Therefore, we conclude that $\{S_j, \hat{Y}_j\}_{j\in\mathcal{N}_c} \subseteq \mathcal{V}$ which give rise to the following lower bound.

lower bound on individual privacy

The individual privacy ρ_i is lower bounded by

$$\rho_{i,\min} = I(S_i; \{S_j, \hat{Y}_j\}_{j\in\mathcal{N}_c}). \tag{H.5}$$

Hence, we have $\rho_{i,\min} \leq \rho_i \leq I(S_i; S_i)$.

There are two more parameters to consider regarding the individual privacy, namely the maximum number of corrupted nodes, giving information about the robustness of the algorithm, and the cost for channel encryption.

Maximum number of corrupted nodes under a passive adversary

The maximum number of corrupted nodes allowed in the network under a passive adversary will be denoted by $k_i \in \{0, \ldots, n-1\}$. That is, the algorithm is guaranteed to achieve individual privacy ρ_i for honest node i if there are at most k_i corrupted nodes in the network.

[1] For the case of discrete random variables, the condition is given in terms of the Shannon entropy $H(\cdot)$

Cost for channel encryption under an eavesdropping adversary

Let $\mathcal{T} = \{0, \ldots, T\}$, where T is the maximum number of iterations. The cost $c_i \in \mathcal{T}$ indicates how many iterations require channel encryption to guarantee individual privacy ρ_i.

We propose a new definition of perfect individual privacy in the context of distributed processing. Intuitively, perfect individual privacy means $\rho_i = 0$. However, due to the fact that in many cases the lower bound $\rho_{i,\min} > 0$, it is in general impossible to achieve zero individual privacy. In addition, we assume $\rho_{i,\min} \neq I(S_i; S_i)$, otherwise there is no privacy at all. We have the following definition of perfect individual privacy.

Definition 1. *(Perfect individual privacy in the context of privacy-preserving distributed processing.) Given $\rho_{i,\min}$, $0 \leq \rho_{i,\min} < I(S_i; S_i)$, a privacy-preserving algorithm achieves perfect individual privacy if and only if $\rho_i = \rho_{i,\min}$.*

4 Linking the proposed metrics to SMPC and DP

In this section, we will show that the well-known SMPC and DP can be considered special cases of the proposed metrics based on different setups or assumptions.

4.1 Secure multiparty computation

An important concept in SMPC is the definition of an ideal world, in which a trusted third party (TTP) is assumed to be available. A TTP first collects all private data from the nodes and computes the output $\boldsymbol{y} = f(\boldsymbol{s})$ after which the outputs y_i are transmitted to each and every node. This scenario is considered secure since a TTP is assumed to be non-corrupted. However, there is a distinction between security and privacy. In the ideal scenario, each node obtains its desired output y_i directly from the TTP. As a consequence, the set of random variables containing the information collected by the adversaries is given by $\mathcal{V} = \{S_j, Y_j\}_{j \in \mathcal{N}_c}$. Therefore, the individual privacy in the ideal world is given by

$$\forall i \in \mathcal{N}_h : \rho_{i,\text{ideal}} = I(S_i; \{S_j, Y_j\}_{j \in \mathcal{N}_c}). \tag{H.6}$$

Apparently, $\rho_{i,\text{ideal}}$ is not necessarily zero and it depends on several factors such as the output function and whether the private data are correlated or not.

The motivation for using SMPC comes from the fact that in practice a third party might not be available or trustworthy. The goal of SMPC is thus to design a protocol that can replace a TTP, i.e., simulates an ideal world. To do so, SMPC has to exchange information between nodes in the network and could, therefore, reveal some information about the private data. Let $\rho_{i,\text{smpc}}$ denote the individual privacy when using SMPC. An SMPC protocol is considered to

be perfect when (1) it achieves perfect output utility and (2) the adversaries do not learn more about each honest node's private data than what will be revealed in an ideal world. That is, SMPC is perfect if

$$\begin{aligned} &\forall i \in \mathcal{N} : u_i = I(Y_i; Y_i), \\ &\forall i \in \mathcal{N}_h : \rho_{i,\text{smpc}} = \rho_{i,\text{ideal}}. \end{aligned} \tag{H.7}$$

As mentioned before, there is a distinction between security and privacy. As an example in which an SMPC protocol is perfect according to (H.7) but reveals maximum individual privacy, i.e., $\rho_{i,\text{smpc}} = I(S_i; S_i)$, consider the situation in which $\boldsymbol{y}$ is a permuted version of the private data $\boldsymbol{s}$. That is, $y_i = s_{i-1 \bmod n}$. Assume that node $i+1$ is corrupted. Using (H.6) we conclude that $\rho_{i,\text{ideal}} = I(S_i; \{S_{i+1}, Y_{i+1} = S_i\} = I(S_i; S_i)$. As $\rho_{i,\text{ideal}}$ is already maximum, any SMPC protocol giving perfect output utility will be considered perfect as $\rho_{i,\text{smpc}} = I(S_i; S_i) = \rho_{i,\text{ideal}}$. Hence, (H.7) is satisfied but there is no privacy at all.

We remark that $\rho_{i,\text{smpc}}$ and $\rho_{i,\text{ideal}}$ in SMPC correspond to the individual privacy ρ_i and its lower bound $\rho_{i,\min}$ under the condition of achieving full output utility in the proposed metrics, respectively. In the above example, in order to achieve meaningful individual privacy $\rho_i < I(S_i; S_i)$, we have to compromise the output utility to decrease the lower bound $\rho_{i,\min}$. That is, perfect output utility and individual privacy are not achievable simultaneously in this example.

4.2 Differential privacy

DP assumes an extreme scenario in which all nodes in the network are corrupted ($k_i = n-1$) except for node i [31, 32]. Let $\boldsymbol{s}^{-i} \in \mathbb{R}^{n-1}$ be a so-called adjacent vector of $\boldsymbol{s}$, obtained by excluding the private data s_i from $\boldsymbol{s}$. Denote Ω_i as the range of s_i. Let $\hat{F}$ be a randomized algorithm that protects the privacy of its input and $\mathcal{Y}$ denotes its output range. Given $\epsilon \geq 0$, algorithm $\hat{F}$ achieves ϵ-DP if for any pair of adjacent vectors $\boldsymbol{s}$ and $\boldsymbol{s}^{-i}$, and for all sets $\mathcal{Y}_s \subseteq \mathcal{Y}$, we have

$$\forall s_i \in \Omega_i : \quad \frac{P(\hat{F}(\boldsymbol{s}) \in \mathcal{Y}_s)}{P(\hat{F}(\boldsymbol{s}^{-i}) \in \mathcal{Y}_s)} \leq e^{\epsilon}. \tag{H.8}$$

It has been shown [41, Theorem 1] that by relaxing the right-hand side of (H.8) to an expected value rather than a statement about all $s_i \in \Omega_i$, (H.8) is related to the Kullback-Leibler divergence and can be further relaxed to the following conditional mutual information (also called mutual information differential privacy):

$$I(S_i; Y | \{S_j\}_{j \in \mathcal{N} \setminus \{i\}}) \leq \epsilon. \tag{H.9}$$

The upper bound ϵ in (H.9) can be interpreted as the difference of the posterior and prior individual privacy. The prior individual privacy, in which the adversaries have the knowledge of $\boldsymbol{s}^{-i}$ and the related output $y' = \hat{F}(\boldsymbol{s}^{-i})$, can be

quantified as

$$\begin{aligned}\rho_{i,\text{prior}} &= I(S_i; \{S_j\}_{j\in\mathcal{N}\setminus\{i\}}, Y') \\ &= I(S_i; \{S_j\}_{j\in\mathcal{N}\setminus\{i\}}),\end{aligned} \tag{H.10}$$

where the last equality holds because Y' is redundant information as $\{S_j\}_{j\in\mathcal{N}\setminus\{i\}}$ can determine Y'. The posterior individual privacy on the other hand, where the adversaries have the knowledge of the algorithm output $y = \hat{F}(\boldsymbol{s})$, is given by

$$\rho_{i,\text{post}} = I(S_i; \{S_j\}_{j\in\mathcal{N}\setminus\{i\}}, Y). \tag{H.11}$$

Based on the definition of conditional mutual information, we can rewrite (H.9) as

$$\begin{aligned}\epsilon &\geq I(S_i; \{S_j\}_{j\in\mathcal{N}\setminus\{i\}}, Y) - I(S_i; \{S_j\}_{j\in\mathcal{N}\setminus\{i\}}) \\ &= \rho_{i,\text{post}} - \rho_{i,\text{prior}},\end{aligned} \tag{H.12}$$

showing the interpretation mentioned above.

We can see that the above $\rho_{i,\text{post}}$ and $\rho_{i,\text{prior}}$ are related to the individual privacy ρ_i and its lower bound $\rho_{i,\min}$, respectively, in the context of distributed processing when we assume that there are $k_i = n - 1$ corrupted nodes. Again, similar to SMPC, $\epsilon = 0$ does not imply zero individual privacy but only means that no additional information is leaked.

4.3 Proposed metrics for SMPC and DP

We end this section by concluding that both the SMPC and DP metrics can be considered as special cases of the proposed metrics under certain assumptions/requirements. For example, a privacy-preserving distributed processing algorithm can be considered as a perfect SMPC protocol if $u_i = I(Y_i; Y_i)$ and $\rho_i = \rho_{i,\min}$, and as an ϵ-DP protocol if $u_i = I(Y_i; \hat{Y}_i), \rho_i \leq \epsilon + \rho_{i,\min}$, and $k_i = n - 1$.

5 Example I: Distributed average consensus

To demonstrate the benefits using the proposed metrics, we use the distributed average consensus as a canonical example. The two main reasons for choosing this problem are that it has general applicability in many signal processing tasks, such as denoising [46] and interpolation [47], and that its privacy-preserving solutions have been widely investigated in the literature [4–16].

In this section, we first define the problem. After that, we introduce traditional distributed average consensus approaches and show that they are not privacy-preserving; maximum individual privacy is revealed as $\forall i \in \mathcal{N}_h : \rho_i = I(S_i; S_i)$.

5.1 Problem definition

The goal of the distributed average consensus algorithm is to compute the global average of all the private data over the network, i.e.,

$$\boldsymbol{y} = s_{\text{ave}}\mathbf{1}, \tag{H.13}$$

where $s_{\text{ave}} = n^{-1}\sum_{i\in\mathcal{N}} s_i$. Hence, we have that $\boldsymbol{y} = n^{-1}\mathbf{1}\mathbf{1}^\top \boldsymbol{s}$. As the nodes in the network can only communicate with the neighboring nodes, the solution is obtained iteratively. Many distributed average consensus algorithms have been proposed to achieve this goal. Below, we introduce two types of approaches that serve as baselines for the coming sections.

Before describing the details, we will make the following assumptions.

Assumption 4. *The private data are statistically independent, i.e.,* $\forall i, j \in \mathcal{N}, i \neq j : I(S_i; S_j) = 0$.

Assumption 5. *The passive adversary has knowledge of the number of nodes* n *in the network and the degree* d_i *of all nodes.*

Let $\mathcal{N}_{i,c} = \mathcal{N}_i \cap \mathcal{N}_c$ and $\mathcal{N}_{i,h} = \mathcal{N}_i \cap \mathcal{N}_h$ denote the set of corrupted and honest neighbors of node i, respectively. In order to consider the worst-case scenario in which all information transmitted by honest nodes is known to the passive adversary, we have the following additional assumption.

Assumption 6. *Every honest node has a non-empty corrupted neighborhood, i.e.,* $\forall i \in \mathcal{N}_h : \mathcal{N}_{i,c} \neq \emptyset$.

5.2 Distributed linear iteration approaches

Distributed average consensus can be obtained by applying, at every iteration $t \in \mathcal{T}$ a linear transformation $\boldsymbol{W} \in \mathcal{W}$ where

$$\mathcal{W} = \left\{\boldsymbol{W} \in \mathbb{R}^{n\times n} \mid \boldsymbol{W}_{ij} = 0 \text{ if } (i,j) \notin \mathcal{E} \text{ and } i \neq j\right\}, \tag{H.14}$$

such that the state vector $\boldsymbol{x}$ is updated as

$$\boldsymbol{x}^{(t+1)} = \boldsymbol{W}\boldsymbol{x}^{(t)}, \tag{H.15}$$

and it is initialized with the private data, i.e.,

$$\boldsymbol{x}^{(0)} = \boldsymbol{s}. \tag{H.16}$$

The structure of $\boldsymbol{W}$ reflects the connectivity of the network[2]. In order to correctly compute the average, that is, $\boldsymbol{x}^{(t)} \to \boldsymbol{y} = n^{-1}\mathbf{1}\mathbf{1}^\top\boldsymbol{s}$ as $t \to \infty$, necessary and sufficient conditions for $\boldsymbol{W}$ are given by (i) $\mathbf{1}^\top\boldsymbol{W} = \mathbf{1}^\top$, (ii) $\boldsymbol{W}\mathbf{1} = \mathbf{1}$,

[2]For simplicity, we assume that $\boldsymbol{W}$ is constant for every iteration, which corresponds to a synchronous implementation of the algorithm. In the case of an asynchronous implementation, the transformation depends on which node will update. The results shown here are easily generalized to asynchronous systems by working with expected values.

(iii) $\alpha\left(\boldsymbol{W}-\frac{\mathbf{1}\mathbf{1}^{\top}}{n}\right)<1$, where $\alpha(\cdot)$ denotes the spectral radius [48].

Individual privacy: By inspecting (H.15), we can see that each node i needs to send its state values $x_i^{(t)}$ to all of its neighbours for updating $\{x_j^{(t+1)}\}_{j\in\mathcal{N}_i}$. Hence, we have $X_i^{(0)}=S_i\in\mathcal{V}$ and we conclude that

$$\rho_i=I(S_i,\mathcal{V})\geq I(S_i,X_i^{(0)})=I(S_i,S_i). \tag{H.17}$$

The algorithm is not private in the sense that it reveals all private information.

5.3 Distributed optimization approaches

The average consensus problem can also be stated as a linear-constrained convex optimization problem given by

$$\begin{aligned}\min_{x_i}\quad & \sum_{i\in\mathcal{N}}\frac{1}{2}\|x_i-s_i\|_2^2\\ \text{s.t.}\quad & \forall(i,j)\in\mathcal{E}:\quad x_i=x_j.\end{aligned} \tag{H.18}$$

Many distributed optimizers have been proposed to solve the above problem, such as ADMM [49] and PDMM [50, 51]. Here, we provide an example using PDMM. The corresponding (extended) augmented Lagrangian function is given by:

$$\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{s}\|_2^2+(\boldsymbol{P}\boldsymbol{\lambda}^{(t)})^T\boldsymbol{C}\boldsymbol{x}+\frac{c}{2}\|\boldsymbol{C}\boldsymbol{x}+\boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(t)}\|_2^2, \tag{H.19}$$

and the updating equations are

$$\boldsymbol{x}^{(t+1)}=\left(\boldsymbol{I}+c\boldsymbol{C}^{\top}\boldsymbol{C}\right)^{-1}\left(\boldsymbol{s}-c\boldsymbol{C}^{\top}\boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(t)}-\boldsymbol{C}^{\top}\boldsymbol{P}\boldsymbol{\lambda}^{(t)}\right), \tag{H.20}$$

$$\boldsymbol{\lambda}^{(t+1)}=\boldsymbol{P}\boldsymbol{\lambda}^{(t)}+c(\boldsymbol{C}\boldsymbol{x}^{(t+1)}+\boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(t)}), \tag{H.21}$$

where $c>0$ is a constant for controlling the convergence rate and $\boldsymbol{\lambda}\in\mathbb{R}^{2m}$ is a dual variable. Let the subscript $i|j$ be a directed identifier that denotes the directed edge from node i to j. We first denote $\boldsymbol{B}\in\mathbb{R}^{m\times n}$ as the graph incidence matrix defined as $\boldsymbol{B}_{li}=1$, $\boldsymbol{B}_{lj}=-1$ if and only if $(i,j)\in\mathcal{E}$ and $i<j$. Denote $e_l=(i,j)\in\mathcal{E}$, where $l\in\{1,\ldots,m\}$, as the l-th edge. The dual variable $\boldsymbol{\lambda}$ is defined as $\lambda_l=\lambda_{i|j}$ and $\lambda_{l+m}=\lambda_{j|i}$. Hence, with PDMM, each edge is associated with two dual variables, $\lambda_{i|j}$ and $\lambda_{j|i}$. The matrix $\boldsymbol{C}\in\mathbb{R}^{2m\times n}$ is related to the graph incidence matrix and defined as $\boldsymbol{C}_{li}=\boldsymbol{B}_{i|j}=1$ and $\boldsymbol{C}_{(l+m)j}=\boldsymbol{B}_{j|i}=-1$ if and only if $i<j$. Of note, $\boldsymbol{P}\in\mathbb{R}^{2m\times 2m}$ denotes a symmetric permutation matrix exchanging the first m with the last m rows. Thus, $\forall(i,j)\in\mathcal{E}:\lambda_{j|i}=(\boldsymbol{P}\boldsymbol{\lambda})_{i|j}$. and $\boldsymbol{C}+\boldsymbol{P}\boldsymbol{C}=[\boldsymbol{B}^{\top}\ \boldsymbol{B}^{\top}]^{\top}$.

The local updating functions for each node become

$$x_i^{(t+1)} = \frac{s_i + \sum_{j\in\mathcal{N}_i}\left(cx_j^{(t)} - \boldsymbol{B}_{i|j}\lambda_{j|i}^{(t)}\right)}{1+cd_i}, \tag{H.22}$$

$$\lambda_{i|j}^{(t+1)} = \lambda_{j|i}^{(t)} + c\boldsymbol{B}_{i|j}\left(x_i^{(t+1)} - x_j^{(t)}\right). \tag{H.23}$$

It has been shown that $\boldsymbol{x}^{(t)}$ converges geometrically (linearly on a logarithmic scale) to the global optimum $\boldsymbol{x}^* = s_{\text{ave}}\mathbf{1}$, given arbitrary initialization of both $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ [50].

Individual privacy: Note that traditional distributed optimization algorithms generally initialize both $\boldsymbol{x}^{(0)}$ and $\boldsymbol{\lambda}^{(0)}$ with all zeros as it gives the smallest initial error resulting in the smallest number of iterations to converge. As a consequence, by inspecting (H.22) we have

$$x_i^{(1)} = \frac{s_i}{1+cd_i}. \tag{H.24}$$

As the constant c is globally known to all nodes and the degree d_i is known to the adversaries based on Assumption 5, the private data s_i can be reconstructed by the adversaries from $x_i^{(1)}$. Since $X_i^{(1)} \in \mathcal{V}$ we conclude that

$$\rho_i = I(S_i, \mathcal{V}) \geq I(S_i, X_i^{(1)}) = I(S_i, S_i). \tag{H.25}$$

Based on (H.17) and (H.25), we conclude that traditional distributed average consensus algorithms, including distributed linear iteration and distributed optimization algorithms, are not privacy-preserving at all; they reveal all private data.

6 Example II: Privacy-preserving distributed average consensus

From the previous section, we can see that the reason why the traditional distributed average consensus algorithms are not privacy-preserving is because the private data, either itself or a scaled version, is directly sent to the neighboring nodes during the data exchange step. As a consequence, one way to protect privacy is to not exchange the private data directly, but to first insert noise to obtain an obfuscated version of it and then exchange the obfuscated data with the neighboring nodes. In what follows, we will first present an information-theoretic result regarding noise insertion to achieve privacy-preservation. After that, we will introduce existing privacy-preserving distributed average consensus approaches and quantify their performances using the proposed metrics.

6.1 Noise insertion for privacy preservation

Proposition 7. *(Arbitrary small information loss can be achieved through noise insertion.) Let private data s and inserted noise r denote realizations of independent random variables S and R with variance $\sigma_S^2, \sigma_R^2 < \infty$, respectively. Let $Z = S + R$. Given arbitrary small $\delta > 0$, there exists $\beta > 0$ such that for $\sigma_R^2 \geq \beta$*

$$I(S; Z) \leq \delta. \tag{H.26}$$

In the case of Gaussian distributed noise, we have

$$\beta = \frac{\sigma_S^2}{2^{2\delta} - 1}. \tag{H.27}$$

Proof. See Appendix 13. □

Proposition 7 shows that the mutual information $I(S; Z)$, where Z is a noisy version of S obtained by adding independent noise, can be made arbitrarily small by making the noise variance sufficiently large.

Based on the design of the noise insertion process, we will classify existing approaches into two classes: zero-sum noise insertion and subspace noise insertion. We first introduce the former case.

The main idea of zero-sum noise insertion comes from the nature of the distributed average consensus. Let r_i denote the noise added by node i to its private data s_i. The estimated output is then given by

$$\hat{y}_i = \frac{1}{n} \sum_{j \in \mathcal{N}} (s_j + r_j) = s_{\text{ave}} + \frac{1}{n} \sum_{j \in \mathcal{N}} r_j. \tag{H.28}$$

Clearly, if the sum of all inserted noise is zero, perfect output utility will be achieved as $\hat{y}_i = s_{\text{ave}} = y_i$ in that case. Next we will proceed to introduce two different approaches, including DP and SMPC, which aim to insert zero-sum noise in a distributed manner.

6.2 Statistical zero-sum noise insertion using DP

DP-based approaches [8–10] mostly apply zero-mean noise insertion to achieve zero-sum in a statistical sense. That is, according to the law of the large numbers, the average of a large number of noise realizations should be close to the expected value, which is zero in this case, and will tend to become closer to the expected value as more realizations are involved. As a consequence, these algorithms only obtain asymptotically perfect output utility as $n \to \infty$. Variants exist in designing the noise insertion process, but here we will focus on one simple example to illustrate the main idea, which was proposed in [8]

and [10]. Each node i initializes its state value by adding zero-mean noise r_i to its private data. That is, the state value initialization (H.16) becomes

$$\forall i \in \mathcal{N} : x_i^{(0)} = s_i + r_i, \tag{H.29}$$

and then arbitrary distributed average consensus algorithms (e.g., linear iterations [48] or distributed optimization [49–51]) can be adopted to compute the average.

Output utility analysis

Assume that all inserted noise are realizations of independent and identically distributed random variables with zero-mean and variance σ^2. Denote $r_{\text{tot}} = \sum_{i\in\mathcal{N}} r_i$ and $r_{\text{ave}} = r_{\text{tot}}/n$ as the sum of all inserted noise realizations and its average, respectively. As a consequence, R_{tot} and R_{ave} are also zero-mean, and their variances are $n\sigma^2$ and σ^2/n, respectively. Based on (H.28) the output utility of node i is

$$\forall i \in \mathcal{N} : \; u_i = I(Y_i; Y_i + R_{\text{ave}}). \tag{H.30}$$

Indeed, as mention before, we obtain perfect output utility only when $n \to \infty$ since $\lim_{n\to\infty} R_{\text{ave}} = 0$.

Individual privacy analysis

DP based approaches do not require any channel encryption and assume $n-1$ corrupted nodes, i.e., $\mathcal{N}_c = \mathcal{N} \setminus \{i\}$. Collecting all state random variables $X_i^{(t)}$ in the vector $X^{(t)} = [X_1^{(t)}, \ldots, X_n^{(t)}]^\top$, we conclude that all information seen by the adversaries throughout the algorithm is

$$\begin{aligned} \mathcal{V} &= \{\hat{Y}_j, S_j, R_j, X^{(t)}\}_{j\in\mathcal{N}_c, t\in\mathcal{T}} \\ &= \{S_j, R_j, X^{(t)}\}_{j\in\mathcal{N}_c, t\in\mathcal{T}}, \end{aligned} \tag{H.31}$$

since $\hat{Y}_j = X_j^{(T)}$. Note that we assume that all messages $\{X^{(t)}\}_{t\in\mathcal{T}}$ transmitted through the communication channels can be eavesdropped and are thus known to the adversaries. We see that computing $I(S_i; \mathcal{V})$ requires to analyze the information flow over the whole iterative process. This imposes challenges as keeping track of information loss throughout all iterations is difficult. We can, however, simplify the privacy analysis through the following result.

Lemma 1. *(Information release of successive iterations.)*

$$I(S_i; X^{(0)}, \ldots, X^{(T)}) = I(S_i; X^{(0)}).$$

Proof. The sequence $S_i \to X^{(0)} \to X^{(t)}$ forms a Markov chain in that order. As a consequence, by the chain rule of mutual information, we have

$$\begin{aligned} I(S_i; X^{(0)}, \ldots, X^{(T)}) &= \sum_{t=0}^{T} I(S_i; X^{(t)} | X^{(t-1)}, \ldots, X^{(0)}) \\ &= I(S_i; X^{(0)}). \end{aligned}$$ □

Lemma 1 states that it is sufficient to analyze the privacy leakage of the initial state vector only as successive iterations will not reveal additional information about the private data. Given this result, we conclude that

$$\begin{aligned} I(S_i; \mathcal{V}) &= I(S_i; \{S_j, R_j, X^{(0)}\}_{j \in \mathcal{N}_c}) \\ &\overset{\text{(a)}}{=} I(S_i; X_i^{(0)}) \\ &\quad + I(S_i; \{S_j, R_j, X_j^{(0)}\}_{j \in \mathcal{N}_c} | X_i^{(0)}) \\ &\overset{\text{(b)}}{=} I(S_i; X_i^{(0)}), \end{aligned} \tag{H.32}$$

where (a) follows from the chain rule of mutual information, and (b) holds as $\{S_j, R_j, X_j^{(0)}\}_{j \in \mathcal{N}_c}$ is independent of both S_i and $X_i^{(0)}$. The individual privacy thus becomes

$$\rho_i = I(S_i; X_i^{(0)}) = I(S_i; S_i + R_i). \tag{H.33}$$

Lower bound analysis. The lower bound on individual privacy is given by

$$\begin{aligned} \rho_{i,\min} &= I(S_i; \{\hat{Y}_j, S_j\}_{j \in \mathcal{N}_c}) \\ &\overset{\text{(a)}}{=} I(S_i; \sum_{j \in \mathcal{N}} S_j + R_{\text{tot}}, \{S_j\}_{j \in \mathcal{N}_c}) \\ &= I(S_i; S_i + R_{\text{tot}}, \{S_j\}_{j \in \mathcal{N}_c}) \\ &\overset{\text{(b)}}{=} I(S_i; S_i + R_{\text{tot}}), \end{aligned} \tag{H.34}$$

where (a) follows from (H.28) and the fact that n is known to the adversaries (Assumption 5) and (b) from the fact that $\{S_j\}_{j \in \mathcal{N}_c}$ is independent of $S_i + R_{\text{tot}}$. By inspection of (H.33) and (H.34) we conclude that for $n > 1$ we have $\rho_{i,\min} < \rho_i$, except for $r_i = 0$, so that DP does not achieve perfect individual privacy for the average consensus problem.

Maximum number of corrupted nodes and cost for channel encryption. Since $\mathcal{N}_c = \mathcal{N} \setminus \{i\}$, we have $k_i = |\mathcal{N}_c| = n - 1$ being the maximum number of corrupted nodes. As no channel encryption is needed, we have $c_i = 0$.

Summarizing, with the proposed metrics, DP-based approaches achieve

$$\begin{aligned} &u_i = I(Y_i; Y_i + R_{\text{ave}}), \\ &\rho_i = I(S_i; S_i + R_i), \\ &\rho_{i,\min} = I(S_i; S_i + R_{\text{tot}}), \\ &k_i = n - 1, \\ &c_i = 0. \end{aligned} \tag{H.35}$$

We have the following remark.

Remark 5. *(In the distributed average consensus, DP always has a trade-off between the output utility and individual privacy.) As both output utility* (H.30) *and individual privacy* (H.33) *are dependent on the inserted noise, we conclude, using Proposition 7, that*

$$\sigma^2 \to \infty \quad \Rightarrow \quad u_i = 0, \rho_i = 0, \tag{H.36}$$

$$\sigma^2 = 0 \quad \Rightarrow \quad u_i = I(Y_i; Y_i),\ \rho_i = I(S_i; S_i). \tag{H.37}$$

Hence DP has a trade-off between privacy and utility. Of note, the conclusion that DP based approaches cannot achieve perfect full utility has been shown before in [10]. Here, we provide a simpler proof in terms of mutual information.

6.3 Exact zero-sum noise insertion using SMPC

Unlike DP based approaches, which have a privacy-utility trade-off, SMPC based approaches can obtain full utility without compromising privacy. However, there is no "free lunch"; the price to be paid is that the robustness over $n-1$ corrupted nodes is no longer achievable. Existing SMPC based approaches [4–6] have applied additive secret sharing [30] to construct exact zero-sum noise through coordinated noise insertion. To do so, at the initialization phase, each node i first sends each neighbor $j \in \mathcal{N}_i$ a random number r_i^j and receives a random number r_j^i from each of its neighbors. After that node i constructs its noise realization as

$$r_i = \sum_{j \in \mathcal{N}_i} r_{i|j}, \tag{H.38}$$

where

$$r_{i|j} = r_j^i - r_i^j. \tag{H.39}$$

Of note, all the random numbers $\{r_i^j\}_{(i,j)\in\mathcal{E}}$ are independent of each other. After constructing the noise realizations, similar as DP based approaches, each node initializes its state value using (H.29) after which an arbitrary distributed average consensus algorithm can be used.

Output utility analysis

In SMPC the noise is constructed such that it sums to zero:

$$\sum_{i\in\mathcal{N}} r_i = \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}_i} r_{i|j} = \sum_{(i,j)\in\mathcal{E}} \left(r_{i|j} + r_{j|i}\right) = 0, \tag{H.40}$$

as $r_{i|j} = -r_{j|i}$ by (H.39). Full utility is thus obtained as $\hat{y}_i = y_i$:

$$\forall i \in \mathcal{N}:\ u_i = I(Y_i; Y_i). \tag{H.41}$$

Individual privacy analysis

SMPC based approaches assume that the communication channels are not securely encrypted except for transmitting the random numbers $\{r_i^j\}_{(i,j)\in\mathcal{E}}$ (initialization phase). As a consequence, all information that the adversaries see throughout the algorithm is given by

$$\begin{aligned}\mathcal{V} &= \{\{Y_j, S_j\}_{j\in\mathcal{N}_c}, \{R_i^j\}_{(i,j)\in\mathcal{E}_c}, \{X^{(t)}\}_{t\in\mathcal{T}}\} \\ &= \{\{S_j\}_{j\in\mathcal{N}_c}, \{R_i^j\}_{(i,j)\in\mathcal{E}_c}, \{X^{(t)}\}_{t\in\mathcal{T}}\},\end{aligned} \tag{H.42}$$

since $Y_j = X_j^{(T)}$ and $X^{(t)}$ is known by Assumption 6.

Let $\mathcal{G}_h \subseteq \mathcal{G}$ denote the graph obtained by removing all corrupted nodes from $\mathcal{G}$. Moreover, let $\mathcal{G}_h = \cup_q \mathcal{C}_q$, where $\mathcal{C}_q$ is a component or connected subgraph of $\mathcal{G}_h$. The set of nodes in $\mathcal{C}_q$ is denoted by $\mathcal{N}_{h_q}$ so that $\mathcal{N}_h = \cup_q \mathcal{N}_{h_q}$. We have the following result which simplifies the individual privacy analysis.

Proposition 8.

$$\forall i \in \mathcal{N}_{h_q} : I(S_i; \mathcal{V}) = I(S_i; \{S_j + \sum_{k\in\mathcal{N}_{j,h}} R_{j|k}\}_{j\in\mathcal{N}_{h_q}}).$$

Proof. See Appendix 14. □

We conclude from Proposition 8 that node i should have at least one honest neighbor. If not, S_i will be revealed as in that case $\mathcal{N}_{h_q} = \{i\}$ and $\mathcal{N}_{j,h} = \emptyset$. Moreover, the adversaries can compute the partial sum of the private data in each component $\mathcal{C}_q$ since

$$\sum_{j\in\mathcal{N}_{h_q}} (S_j + \sum_{k\in\mathcal{N}_{j,h}} R_{j|k}) = \sum_{j\in\mathcal{N}_{h_q}} S_j, \tag{H.43}$$

as $R_{j|k} = -R_{k|j}$. Since this partial sum can always be determined regardless of the amount of noise insertion, we have

$$\rho_i = I(S_i; \mathcal{V}) \geq I(S_i; \sum_{j\in\mathcal{N}_{h_q}} S_j). \tag{H.44}$$

We have equality in (H.44) when the partial sum (H.43) is all the adversaries know and no additional information can be inferred from the individual noisy observations. That is, we have equality if $\forall j \in \mathcal{N}_{h_q} : I(S_i; S_j + \sum_{k\in\mathcal{N}_{j,h}} R_{j|k}) = 0$, which can, by Proposition 7, be achieved asymptotically by adding independent noise to the private data. Therefore, the privacy level SMPC based approaches can achieve is given by

$$\rho_i = I(S_i; \sum_{j\in\mathcal{N}_{h_q}} S_j). \tag{H.45}$$

Lower bound analysis. With perfect output utility, the lower bound (H.5) becomes

$$\begin{aligned}\rho_{i,\min} &= I(S_i; \{Y_j, S_j\}_{j\in\mathcal{N}_c}) \\ &\overset{\text{(a)}}{=} I(S_i; \sum_{j\in\mathcal{N}} S_j, \{S_j\}_{j\in\mathcal{N}_c}) \\ &\overset{\text{(b)}}{=} I(S_i; \sum_{j\in\mathcal{N}_h} S_j, \{S_j\}_{j\in\mathcal{N}_c}) \\ &\overset{\text{(c)}}{=} I(S_i; \sum_{j\in\mathcal{N}_h} S_j),\end{aligned} \tag{H.46}$$

where (a) holds as $\forall j \in \mathcal{N} : y_j = n^{-1}\sum_{j\in\mathcal{N}} S_j$ and n is known by Assumption 5, (b) holds as $\sum_{j\in\mathcal{N}} S_j, \{S_j\}_{j\in\mathcal{N}_c}$ can be determined by $\sum_{j\in\mathcal{N}_h} S_j, \{S_j\}_{j\in\mathcal{N}_c}$ as $S_j, j \in \mathcal{N}_c$, are known to the adversaries, and (c) holds as $\{S_j\}_{j\in\mathcal{N}_c}$ is independent of both S_i and $\sum_{j\in\mathcal{N}_h} S_j$ by Assumption 4.

Maximum number of corrupted nodes and cost for channel encryption. As mentioned before, to guarantee the individual privacy $\rho_i < I(S_i; S_i)$, node i should have at least one honest neighbor, i.e., $\mathcal{N}_{i,h} \neq \emptyset$. The maximum number of corrupted nodes is therefore $k_i = d_i - 1$ and only depends on the degree d_i. For a fully connected graph we have $k_i = n - 2$. The amount of channel encryption is $c_i = 1$ as only the communication channels in the initialization phase need to be securely encrypted.

In conclusion, with the proposed metrics, SMPC based approaches achieve

$$\begin{aligned}&u_i = I(Y_i; Y_i), \\ &\rho_i = I(S_i; \textstyle\sum_{j\in\mathcal{N}_{h_q}} S_j), \\ &\rho_{i,\min} = I(S_i; \textstyle\sum_{j\in\mathcal{N}_h} S_j), \\ &k_i = d_i - 1, \\ &c_i = 1.\end{aligned} \tag{H.47}$$

We can see that u_i is independent of ρ_i. Hence, SMPC has no trade-off between privacy and utility in distributed average consensus. Hence, we have the following remark.

Remark 6. *(Conditions for achieving perfect individual privacy and perfect output utility using the SMPC based approaches in the distributed average consensus.) By inspection of* (H.45) *and* (H.46)*, if $\mathcal{G}_h$ is connected and $|\mathcal{N}_h| \geq 2$, we have only one component so that $\mathcal{N}_{h_q} = \mathcal{N}_h$ and thus $\rho_i = \rho_{i,\min}$; the algorithm achieves both perfect individual privacy (Definition 1) and perfect output utility.*

The main limitation of the above zero-sum noise insertion approaches is that it is hard to be generalized to problems other than distributed average consensus. To mitigate this problem, recently subspace noise-insertion based algorithms have been proposed which are able to solve more general (convex) optimization problems. In the next subsection we will introduce such an

approach referred to as distributed optimization based subspace perturbation (DOSP).

6.4 Subspace noise insertion using DOSP

The DOSP algorithm [7, 27] differentiates from the DP and SMPC based approaches in the sense that it can ensure full output utility without compromising privacy and does not require coordinated noise insertion. In particular, DOSP does not introduce zero-sum noise but exploits the fact that the dual variables, if properly initialized, can obfuscate the private data throughout the algorithm. As a consequence, in order to analyze privacy, we have to consider the convergence behavior of the dual variable $\boldsymbol{\lambda}$.

To do so, consider two successive $\boldsymbol{\lambda}$-update in (H.21). We have

$$\boldsymbol{\lambda}^{(t+2)} = \boldsymbol{\lambda}^{(t)} + c(\boldsymbol{C}\boldsymbol{x}^{(t+2)} + 2\boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(t+1)} + \boldsymbol{C}\boldsymbol{x}^{(t)}), \tag{H.48}$$

as $\boldsymbol{P}^2 = \boldsymbol{I}$. Let $\bar{H} = \mathrm{span}(\boldsymbol{C}) + \mathrm{span}(\boldsymbol{P}\boldsymbol{C})$ and $\bar{H}^{\perp} = \mathrm{null}(\boldsymbol{C}^{\top}) \cap \mathrm{null}((\boldsymbol{P}\boldsymbol{C})^{\top})$. We can see that every two $\boldsymbol{\lambda}$-updates affect only $\Pi_{\bar{H}}\boldsymbol{\lambda} \in \bar{H}$ where $\Pi_{\bar{H}}$ denotes the orthogonal projection onto $\bar{H}$. As shown in [27], the dual variable $\boldsymbol{\lambda}^{(t)}$ composites of two parts: a so-called convergent component $\Pi_{\bar{H}}\boldsymbol{\lambda}^{(t)}$ which will converge to a fixed point $\boldsymbol{\lambda}^*$, and a so-called non-convergent component $(\boldsymbol{I} - \Pi_{\bar{H}})\boldsymbol{\lambda}^{(t)} = \boldsymbol{P}^t\left(\boldsymbol{I} - \Pi_{\bar{H}}\right)\boldsymbol{\lambda}^{(0)}$ which will not converge ($\boldsymbol{P}^t = \boldsymbol{P}$ for t odd and $\boldsymbol{P}^t = \boldsymbol{I}$ for t even) and only depends on the initialization $\boldsymbol{\lambda}^{(0)}$.

By inspecting (H.22), the noise for protecting s_i of honest node i is constructed as

$$\begin{aligned}\forall t \in \mathcal{T} : r_i^{(t)} &= \sum_{j\in\mathcal{N}_i} (\boldsymbol{B}_{i|j}\lambda_{j|i}^{(t)}) \\ &= \sum_{j\in\mathcal{N}_{i,c}} (\boldsymbol{B}_{i|j}\lambda_{j|i}^{(t)}) + \sum_{j\in\mathcal{N}_{i,h}} (\boldsymbol{B}_{i|j}\lambda_{j|i}^{(t)}),\end{aligned} \tag{H.49}$$

where the dual variables $\{\lambda_{j|i}^{(t)}\}_{j\in\mathcal{N}_{i,c}}$ of the corrupted neighbors are known to the adversaries. As a consequence, only $\sum_{j\in\mathcal{N}_{i,h}}(\boldsymbol{B}_{i|j}\lambda_{j|i}^{(t)})$ is unknown to the adversaries. Separating the convergent and non-convergent component of $\boldsymbol{\lambda}^{(t)}$, we have

$$\begin{aligned}\sum_{j\in\mathcal{N}_{i,h}} (\boldsymbol{B}_{i|j}\lambda_{j|i}^{(t)}) &= \sum_{j\in\mathcal{N}_{i,h}} (\boldsymbol{B}_{i|j}(\Pi_{\bar{H}}\boldsymbol{\lambda}^{(t)})_{j|i}) \\ &+ \sum_{j\in\mathcal{N}_{i,h}} \left(\boldsymbol{B}_{i|j}(\boldsymbol{P}^t(\boldsymbol{I} - \Pi_{\bar{H}})\boldsymbol{\lambda}^{(0)})_{j|i}\right).\end{aligned} \tag{H.50}$$

The main idea of subspace noise insertion is to exploit the non-convergent component of the dual variables as subspace noise for guaranteeing the privacy. That is, $\sum_{j\in\mathcal{N}_{i,h}}\left(\boldsymbol{B}_{i|j}(\boldsymbol{P}^t(\boldsymbol{I} - \Pi_{\bar{H}})\boldsymbol{\lambda}^{(0)})_{j|i}\right)$ protects the private data s_i

from being revealed to others. By controlling $\boldsymbol{\lambda}^{(0)}$, the variance of the above subspace noise can be made arbitrarily large so that, by Proposition 7, we can achieve an arbitrarily small information loss.

Before discussing how to implement the subspace noise, we first state the following remark.

Remark 7. *(There is always a non-empty subspace for noise insertion as long as $m \geq n$.) Since $[\boldsymbol{C}\ \boldsymbol{PC}] \in \mathbb{R}^{2m\times 2n}$ can be viewed as a new graph incidence matrix with $2n$ nodes and $2m$ edges [27], we thus have $\dim(\bar{H}) \leq 2n-1$, and $\bar{H}^{\perp}$ is non-empty if $m \geq n$.*

In DOSP, each node only needs to randomly initialize its own dual variables $\{\lambda_{i|j}^{(0)}\}_{j\in\mathcal{N}_i}$ as in that case we have $(\boldsymbol{I}-\Pi_{\bar{H}})\boldsymbol{\lambda}^{(0)} \neq \mathbf{0}$ with probability 1 as long as $m \geq n$. Hence, DOSP does not require any coordination between nodes for noise construction. In the remainder of this section we will investigate the output utility and individual privacy of DOSP.

Output utility analysis

As mentioned before, $\boldsymbol{x}^{(t)}$ converges geometrically to the global optimum $\boldsymbol{x}^* = s_{\text{ave}}\mathbf{1}$, given arbitrary initialization of both $\boldsymbol{x}$ and $\boldsymbol{\lambda}$, even though $\boldsymbol{\lambda}^{(t)}$ does not necessarily converge. Indeed, by inspection of (H.20), we see that the non-converging component of $\boldsymbol{\lambda}^{(t)}$ does not affect the $\boldsymbol{x}$-update since

$$\boldsymbol{C}^{\top}\boldsymbol{P}\left(\boldsymbol{I}-\Pi_{\bar{H}}\right)\boldsymbol{\lambda}^{(t)} = \left(\boldsymbol{PC}\right)^{\top}\left(\boldsymbol{I}-\Pi_{\bar{H}}\right)\boldsymbol{\lambda}^{(t)} = 0. \tag{H.51}$$

Hence, DOSP achieves perfect output utility.

Individual privacy analysis

Similar as the above SMPC based approaches, DOSP assumes that the communication channels are not securely encrypted except for the initialization phase where the initialized $\lambda_{i|j}^{(0)}$ are transmitted to all neighboring nodes. Therefore, the information collected by the adversaries throughout the course of the algorithm is given by

$$\begin{aligned}\mathcal{V} &= \{\{Y_j, S_j\}_{j\in\mathcal{N}_c}, \{\Lambda_{i|j}^{(0)}, X^{(t)}\}_{(i,j)\in\mathcal{E}_c, t\in\mathcal{T}}\} \\ &= \{\{S_j\}_{j\in\mathcal{N}_c}, \{\Lambda_{i|j}^{(0)}, X^{(t)}\}_{(i,j)\in\mathcal{E}_c, t\in\mathcal{T}}\},\end{aligned} \tag{H.52}$$

since $Y_j = X_j^{(T)}$. Note that all the $\{\Lambda_{i|j}^{(t)}\}_{(i,j)\in\mathcal{E}_c, t>0}$ are not included here because they are not transmitted through the network, and they can be determined by $\{X^{(t)}\}_{t\in\mathcal{T}}$ and $\{\Lambda_{i|j}^{(0)}\}_{(i,j)\in\mathcal{E}_c}$ from (H.21). We have the following result which simplifies the privacy analysis of DOSP.

Proposition 9.

$$I(S_i;\mathcal{V}) = I(S_i;\{S_j - \sum_{k\in\mathcal{N}_{j,h}} \boldsymbol{B}_{j|k}\Lambda_{k|j}^{(t)}\}_{j\in\mathcal{N}_h,t=0,1} |\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda_{i|j}^{(0)}\}_{(i,j)\in\mathcal{E}_c}). \tag{H.53}$$

Proof. See Appendix 15. □

We note that, similar to the SMPC based approach, the partial sum $\sum_{j\in\mathcal{N}_{h_q}} S_j$ can be computed by the adversaries. Indeed, the partial sum can be constructed as

$$\sum_{j\in\mathcal{N}_{h_q}} S_j = \frac{1}{2}\left(\sum_{t=0,1}\sum_{j\in\mathcal{N}_{h_q}}\left(S_j - \sum_{k\in\mathcal{N}_{j,h}} \boldsymbol{B}_{j|k}\Lambda_{k|j}^{(t)}\right) + \sum_{t=0,1}\sum_{j\in\mathcal{N}_{h_q}}\sum_{k\in\mathcal{N}_{j,h}} \boldsymbol{B}_{j|k}\Lambda_{k|j}^{(t)}\right). \tag{H.54}$$

The first term of the right-hand side of (H.54) is the addition of terms that are known by the adversaries, as shown by (H.53). Let $\mathcal{E}_{h_q} = \{(i,j) \in \mathcal{E} : (i,j) \in \mathcal{N}_{h_q} \times \mathcal{N}_{h_q}\}$ denote the set of all edges between the honest nodes in component $\mathcal{C}_q$. With this, the second term of (H.54) can be expressed as

$$\begin{aligned}
&\sum_{t=0,1}\sum_{j\in\mathcal{N}_{h_q}}\sum_{k\in\mathcal{N}_{j,h}} \boldsymbol{B}_{j|k}\Lambda_{k|j}^{(t)} \\
&= \sum_{t=0,1}\sum_{(i,j)\in\mathcal{E}_{h_q}} \left(\boldsymbol{B}_{j|k}\Lambda_{k|j}^{(t)} + \boldsymbol{B}_{k|j}\Lambda_{j|k}^{(t)}\right) \\
&= \sum_{t=0,1}\sum_{(i,j)\in\mathcal{E}_{h_q}} \boldsymbol{B}_{j|k}\left(\Lambda_{k|j}^{(t)} - \Lambda_{j|k}^{(t)}\right) \\
&= \sum_{(i,j)\in\mathcal{E}_{h_q}} \boldsymbol{B}_{j|k}\left(\left(\Lambda_{k|j}^{(1)} - \Lambda_{j|k}^{(0)}\right) - \left(\Lambda_{j|k}^{(1)} - \Lambda_{k|j}^{(0)}\right)\right),
\end{aligned}$$

which can be determined by the adversaries since, by inspection of (H.23), the difference $\Lambda_{i|j}^{(1)} - \Lambda_{j|i}^{(0)}$ only depends on $x_i^{(1)}$ and $x_j^{(0)}$, all of which are known by the adversaries (based on (H.52)).

As the partial sum can be computed, the analysis of DOSP follows along the same line as the one presented for SMPC and we conclude that the performance indicators for DOSP, as measured by the proposed metrics, are also given by (H.47). In addition, Remark 6 also holds for DOSP.

6.5 Comparisons of existing approaches

In Table H.1 we summarize the performances of the discussed DP, SMPC and DOSP approaches for distributed average consensus. We can see that SMPC

and DOSP achieve exactly the same performances, except the fact that SMPC requires coordination between nodes to construct zero-sum noise. Moreover, DP is robust against $n-1$ corrupted nodes and does not require channel encryption at all but suffers from a privacy-utility trade-off. On the other hand, SMPC and DOSP do not have privacy-utility trade-off but are only robust to $d_i - 1$ corrupted nodes and require channel encryption for the first iteration.

7 Numerical results

In this section we compare DP, SMPC and DOSP using computer simulations. The comparisons are conducted in terms of (1) convergence behavior and (2) utility/privacy behavior. Their metrics are given below.

- Convergence behavior: mean square error to measure the distance between the state value $\boldsymbol{x}^{(t)}$ and the desired average result $\boldsymbol{x}^* = s_{\text{ave}}\mathbf{1}$ for each iteration t, i.e., $\|\boldsymbol{x}^{(t)} - \boldsymbol{x}^*\|^2$.
- Privacy/utility behavior: normalized mutual information (NMI)[3] to measure the information-theoretical performances, i.e., $u_i/I(Y_i;Y_i)$ for the output utility, $\rho_i/I(S_i;S_i)$ for the individual privacy and $\rho_{i,\min}/I(S_i;S_i)$ for the lower bound on individual privacy.

We simulated a geometrical graph with $n = 10$ nodes, and set the radius as $r^2 = 2\frac{\log n}{n}$ to ensure a connected graph with high probability [52]. For simplicity, all private data have a zero-mean unit variance Gaussian distribution, and all the noise used in the DP, SMPC and DOSP approaches follow a zero-mean Gaussian distribution with variance σ^2.

7.1 Convergence behavior

In Fig. H.2 we present the convergence behavior of the algorithms under different amounts of noise insertion, i.e., different noise variances. We can see that all algorithms achieve the correct average value in the absence of noise,

[3]Since the experiments are done using discrete data, the mutual information $I(X;X)$ is bounded by $H(X) < \infty$.

Table H.1: Comparisons of existing information-theoretic solutions for the distributed average consensus

	DP [8–10]	SMPC [4–6]	DOSP [7, 27]
Adversary models	Passive, Eavesdropping		
Coordinated noise insertion	No	Yes	No
Output utility	$u_i = I(Y_i; Y_i + R_{\text{ave}})$	$u_i = I(Y_i; Y_i)$	
Individual privacy	$\rho_i = I(S_i; S_i + R_i)$	$\rho_i = I(S_i; \sum_{j\in\mathcal{N}_{h_q}} S_j)$	
Lower bound on individual privacy	$\rho_{i,\min} = I(S_i; S_i + R_{\text{tot}})$	$\rho_{i,\min} = I(S_i; \sum_{j\in\mathcal{N}_h} S_j)$	
Maximum number of corrupted nodes	$k_i = n-1$ out of n	$k_i = d_i - 1$ out of d_i	
Cost of channel encryption	$c_i = 0$	$c_i = 1$	

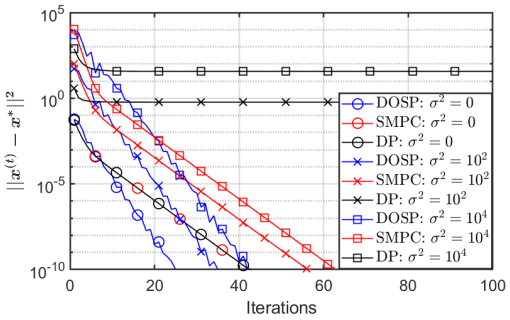
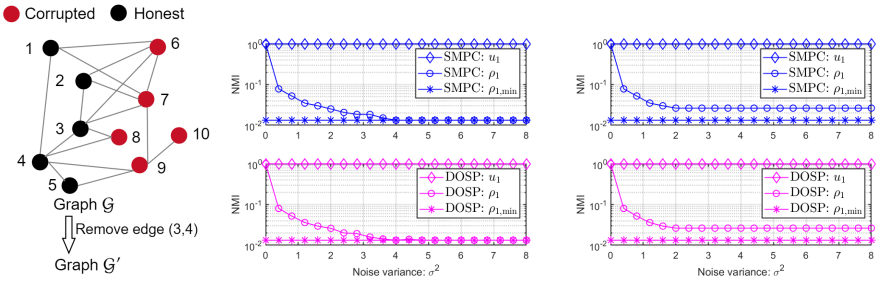


Fig. H.2: Convergence behaviors of DOSP, SMPC and DP based approaches under three different amounts of noise insertion.



(a) (a) **(b)** (b) **(c)** (c)

Fig. H.3: (a) Two sample graphs in which $\mathcal{G}'$ and $\mathcal{G}$ differ in only one edge. Normalized mutual information of output utility, individual privacy, and its lower bound for honest node 1 in terms of the amount of noise insertion by using SMPC and DOSP approaches under (b) graph $\mathcal{G}$ and (c) graph $\mathcal{G}'$.

i.e., $\sigma^2 = 0$. For nonzero noise variance, however, only the DOSP and SMPC based approaches achieve the correct average value, regardless of the amount of noise inserted, whereas the accuracy of the DP based approach is compromised by increasing the amount of noise insertion.

7.2 Utility and privacy

To validate the output utility, individual privacy, and its lower bound, we ran 10^4 Monte Carlo simulations and used the non-parametric entropy estimation toolbox (npeet) [53] to estimate the normalized mutual information.

Privacy-utility results of the DOSP and SMPC based approaches under different graph topologies

As shown in Table H.1, the performances of SMPC and DOSP are dependent on the number of corrupted nodes in the neighborhood and the graph topology.

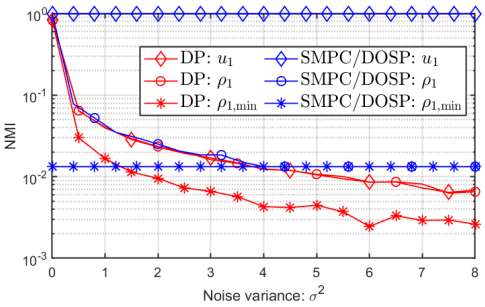


Fig. H.4: NMI of output utility, individual privacy, and its lower bound for honest node i in terms of the amount of noise insertion using DP, SMPC and DOSP approaches.

Note that we do not consider DP here because its performance is not dependent on graph topology as it assumes $n-1$ corrupted nodes. To demonstrate the effects of graph topology, Fig. H.3(a) shows a graph $\mathcal{G}$ satisfying Assumption 6; i.e., every honest node is connected to at least one corrupted node. In addition, we consider the graph $\mathcal{G}'$ which is obtained from $\mathcal{G}$ by removing edge $(3,4)$. The main difference between graph $\mathcal{G}$ and $\mathcal{G}'$ is that, after removing all corrupted nodes, in the former all the honest nodes are connected and in the latter they are separated in two connected subgraphs. The privacy-utility results of the DOSP and SMPC based approaches over graph $\mathcal{G}$ and $\mathcal{G}'$ are shown in Fig. H.3(b) and H.3(c), respectively. We validate the following theoretical results regarding utility and privacy:

- SMPC and DOSP both ensure full utility regardless of the amount of noise, and thus the privacy level;
- The optimum individual privacy of node $i \in \mathcal{C}_q$ is only related to the partial sum of the private data in subgraph $\mathcal{C}_q$, i.e, $\rho_i = I(S_i; \sum_{j\in\mathcal{N}_{h_q}} S_j)$;
- For graph $\mathcal{G}$ both approaches are able to obtain perfect individual privacy, i.e., the result in Remark 6 is validated.

Privacy-utility comparisons of the DP, SMPC and DOSP approaches

In Fig. H.4 we compare DP, SMPC and DOSP in terms of the amount of noise insertion using graph $\mathcal{G}$. We show the performance of SMPC and DOSP together because they have identical performances as shown in Fig. H.3(b). Fig. H.4 shows that, in contrast to SMPC and DOSP which guarantee perfect output utility and a fixed individual privacy, DP can achieve a lower individual privacy by increasing the noise variance. However, the price to pay is a deterioration of output utility, validating the fact that DP trades-off privacy versus utility.

8 Suggestions for algorithm design

We now provide some suggestions on how to design appropriate privacy-preserving algorithms for different applications. Typical ways to design a privacy-preserving solution are (1) choose one of the off-the-shelf tools such as DP, SMPC or DOSP; (2) combine them to obtain a hybrid approach. We concluded that the performances indicator of privacy-preserving distributed processing algorithms were bounded by $u_i \leq I(Y_i; Y_i)$ (perfect output utility), $I(S_i; S_i) > \rho_i \geq \rho_{i,\min}$ (perfect individual privacy), $k_i \leq n-1$ (maximum number of corrupted nodes being), and $c_i \geq 0$ (minimum (zero) cost for channel encryption). To provide insight on *when it is possible to achieve these optimum performances simultaneously*, we have the following result.

Remark 8. *(For any application satisfies $I(S_i; \{S_j, Y_j\}_{j\in\mathcal{N}\setminus\{i\}}) = I(S_i; S_i)$, it is impossible to protect privacy under the conditions of both perfect output utility and $k_i = n-1$ being the maximum number of corrupted nodes.) The reason is simply because the lower bound under such conditions $\rho_{i,\min} = I(S_i; \{S_j, Y_j\}_{j\in\mathcal{N}\setminus\{i\}}) = I(S_i; S_i)$ is already the maximum; there is no privacy at all. An immediate implication of this result is that a SMPC/DOSP, which achieves perfect output utility, can never be differentially private for such applications. In other words, DP and SMPC/DOSP are mutually exclusive for such applications.*

One conclusion for algorithm design can be drawn from the above result: given an application at hand, the first thing to do is to compute the lower bound under the condition of perfect output utility and $k_i = n-1$, i.e., $\rho_{i,\min} = I(S_i; \{S_j, Y_j\}_{j\in\mathcal{N}\setminus\{i\}})$. Based on this lower bound, we then classify applications into two classes and give related suggestions on how to design algorithms.

8.1 Applications for which $\rho_{i,\min} = I(S_i; S_i)$

One example of such applications is the distributed average consensus. For applications where $\rho_{i,\min} = I(S_i; S_i)$ (Remark 8), we should be aware that it is impossible to design privacy-preserving algorithms with all optimum performances. Therefore, we have to prioritize different performances, compromise one to achieve another. Here are some suggestions for algorithm designs:

1. If the application is in an extreme distrust scenario, i.e., $k_i = n-1$ is required, then adopt DP based approaches. But be aware that there is a trade-off between privacy and utility.

2. If the application is very sensitive in terms of the accuracy of function output, e.g., perfect output utility is a must, then both SMPC and DOSP are options. But be aware that $k_i < n-1$ and that the individual privacy depends on the graph topology.

8.2 Applications for which $\rho_{i,\min} < I(S_i; S_i)$

One such example is the application where the objective function is a function of the ℓ_1-norm, like $f(\boldsymbol{s}) = \sum_{i\in\mathcal{N}} |s_i|\mathbf{1}$. For applications where $\rho_{i,\min} < I(S_i; S_i)$, we have the following suggestions:

1. If $\rho_{i,\min}$ is tolerable, it is possible to achieve perfect individual privacy $\rho_i = \rho_{i,\min}$ under the condition of both perfect output utility and $k_i = n - 1$. Try to use either SMPC or DOSP to achieve such optimum performances.

2. If the above cannot be achieved, one option is to compromise the requirement of $k_i = n - 1$, i.e., decrease k_i, and try to use SMPC or DOSP to obtain both perfect individual privacy and perfect output utility only.

3. If $\rho_{i,\min}$ is not tolerable, one option is to combine SMPC or DOSP with DP to decrease this lower bound by compromising the output utility.

9 Conclusions

In this paper, we first proposed information-theoretic metrics for quantifying the algorithm performance in terms of output utility and individual privacy. The proposed metrics are general and can reduce to well-known frameworks including SMPC and DP under certain conditions. We derived several theoretical results in terms of mutual information. We explicitly analyzed, compared and related the state-of-the-art algorithms including DP, SMPC and DOSP for the distributed average consensus problem, and validated the theoretical results by computer simulations. Given the lower bound on individual privacy, we gave suggestions on how to design privacy-preserving algorithms given different conditions/assumptions.

10 Appendix

10.1 Proof of Proposition 7

Proof. As the private data S is independent of the noise R, we have $\sigma_Z^2 = \sigma_S^2 + \sigma_R^2$. Let $\gamma = 1/\sigma_Z$ and define $Z' = \gamma Z$ as the normalized random variable with unit variance. Since mutual information is invariant under scaling, we

have

$$\begin{aligned}\lim_{\sigma_R^2\to\infty} I(S;Z) &= \lim_{\sigma_R^2\to\infty} I(\gamma S;\gamma Z)\\ &= \lim_{\gamma\to 0} I(\gamma S;Z')\\ &= I(0;Z')\\ &= 0.\end{aligned}$$

Hence we conclude that given arbitrary small $\delta > 0$, there exists $\beta > 0$ such that for $\sigma_R^2 \geq \beta$ we have $I(S;Z) \leq \delta$. In the case of Gaussian distributed noise, we find

$$\begin{aligned}I(S;Z) &= h(Z) - h(Z|S)\\ &= h(Z) - h(R)\\ &\overset{\text{(a)}}{=} h(Z) - \frac{1}{2}\log(2\pi e\sigma_R^2)\\ &\overset{\text{(b)}}{\leq} \frac{1}{2}\log(2\pi e\sigma_Z^2) - \frac{1}{2}\log(2\pi e\sigma_R^2)\\ &= \frac{1}{2}\log(1+\sigma_S^2/\sigma_R^2),\end{aligned}$$

where (a) holds as the differential entropy of a Gaussian random variable with variance σ^2 is given by $\frac{1}{2}\log(2\pi e\sigma^2)$, and (b) holds because the maximum entropy of a random variable with fixed variance is achieved by a Gaussian distribution. Hence

$$\delta = \frac{1}{2}\log(1+\sigma_S^2/\sigma_R^2) \quad \Leftrightarrow \quad \sigma_R^2 = \frac{\sigma_S^2}{2^{2\delta}-1} = \beta. \qquad \square$$

10.2 Proof of Proposition 8

Proof.

$$
\begin{aligned}
I(S_i;\mathcal{V}) &= I(S_i;\{S_j\}_{j\in\mathcal{N}_c},\{R_i^j\}_{(i,j)\in\mathcal{E}_c},\{X^{(t)}\}_{t\in\mathcal{T}}) \\
&\overset{\text{(a)}}{=} I(S_i;\{S_j\}_{j\in\mathcal{N}_c},\{R_i^j\}_{(i,j)\in\mathcal{E}_c},X^{(0)}) \\
&\overset{\text{(b)}}{=} I(S_i;\{S_j\}_{j\in\mathcal{N}_c},\{R_i^j\}_{(i,j)\in\mathcal{E}_c},\{X_j^{(0)}\}_{j\in\mathcal{N}_h}) \\
&\overset{\text{(c)}}{=} I(S_i;\{R_i^j\}_{(i,j)\in\mathcal{E}_c},\{X_j^{(0)}\}_{j\in\mathcal{N}_h}) \\
&\overset{\text{(d)}}{=} I(S_i;\{R_i^j\}_{(i,j)\in\mathcal{E}_c},\{S_j+\sum_{k\in\mathcal{N}_j}R_{j|k}\}_{j\in\mathcal{N}_h}) \\
&\overset{\text{(e)}}{=} I(S_i;\{R_i^j\}_{(i,j)\in\mathcal{E}_c},\{S_j+\sum_{k\in\mathcal{N}_{j,h}}R_{j|k}\}_{j\in\mathcal{N}_h}) \\
&\overset{\text{(f)}}{=} I(S_i;\{S_j+\sum_{k\in\mathcal{N}_{j,h}}R_{j|k}\}_{j\in\mathcal{N}_h}) \\
&\overset{\text{(g)}}{=} I(S_i;\{S_j+\sum_{k\in\mathcal{N}_{j,h}}R_{j|k}\}_{j\in\mathcal{N}_{h_q}}),
\end{aligned}
$$

where (a) holds by Lemma 1, as $\forall t\geq 1: S_i\to X^{(0)}\to X^{(t)}$ forms a Markov chain; (b) holds, as $\{X_j^{(0)}\}_{j\in\mathcal{N}_c}$ can be determined from $\{S_j\}_{j\in\mathcal{N}_c},\{R_i^j\}_{(i,j)\in\mathcal{E}_c}$ using (H.29), (H.39) and (H.38); (c) holds because $\{S_j\}_{j\in\mathcal{N}_c}$ is independent of $\{R_i^j\}_{(i,j)\in\mathcal{E}_c},\{X_j^{(0)}\}_{j\in\mathcal{N}_h}$ and S_i; (d) holds by representing $\{X_j^{(0)}\}_{j\in\mathcal{N}_h}$ by using (H.29) and (H.38); (e) follows as $\{\sum_{k\in\mathcal{N}_{j,c}}R_{j|k}\}_{j\in\mathcal{N}_h}$ can be determined from $\{R_i^j\}_{(i,j)\in\mathcal{E}_c}$ by using (H.39); (f) holds as $\{R_i^j\}_{(i,j)\in\mathcal{E}_c}$ is independent of both S_i and $\{S_j+\sum_{k\in\mathcal{N}_{j,h}}R_{j|k}\}_{j\in\mathcal{N}_h}$; and (g) holds as $\{S_j+\sum_{k\in\mathcal{N}_{j,h}}R_{j|k}\}_{j\in\mathcal{N}_h\setminus\mathcal{N}_{h_q}}$ is independent of both S_i and $\{S_j+\sum_{k\in\mathcal{N}_{j,h}}R_{j|k}\}_{j\in\mathcal{N}_{h_q}}$. □

10.3 Proof of equation (H.53)

Proof. By combining (H.48) and two successive $\boldsymbol{x}$-updates (H.20), it can be shown that

$$
\begin{aligned}
\boldsymbol{x}^{(t+1)}-\boldsymbol{x}^{(t-1)} &= \left(\boldsymbol{I}+c\boldsymbol{C}^\top\boldsymbol{C}\right)^{-1} \\
&\quad\left(-2c\boldsymbol{C}^\top\boldsymbol{P}\boldsymbol{C}\boldsymbol{x}^{(t)}-2c\boldsymbol{C}^\top\boldsymbol{C}\boldsymbol{x}^{(t-1)}\right). \qquad \text{(H.55)}
\end{aligned}
$$

We have

$$
\begin{aligned}
I(S_i;\mathcal{V}) &= I(S_i;\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c},\{X^{(t)}\}_{t\in\mathcal{T}}) \\
&\overset{\text{(a)}}{=} I(S_i;\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c},\{X^{(1)},X^{(2)}\}) \\
&\overset{\text{(b)}}{=} I(S_i;\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c},\{X^{(1)}_j,X^{(2)}_j\}_{j\in\mathcal{N}_h}) \\
&\overset{\text{(c)}}{=} I(S_i;\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c} \\
&\qquad ,\{S_j-\sum_{k\in\mathcal{N}_{j,h}}\boldsymbol{B}_{j|k}\Lambda^{(t)}_{k|j}\}_{j\in\mathcal{N}_h,t=0,1}) \\
&\overset{\text{(d)}}{=} I(S_i;\{S_j-\sum_{k\in\mathcal{N}_{j,h}}\boldsymbol{B}_{j|k}\Lambda^{(t)}_{k|j}\}_{j\in\mathcal{N}_h,t=0,1} \\
&\qquad |\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c})
\end{aligned}
$$

where (a) holds, as all $\{X^{(t)}\}_{t>2}$ can be determined by $X^{(1)}$ and $X^{(2)}$ using (H.55) (note that we omit $X^{(0)}$ by assuming $\boldsymbol{x}$ is initialized with all zeros); (b) holds, as $\{X^{(1)}_j\}_{j\in\mathcal{N}_c}$ can be constructed by $\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c}$; and similarly $\{X^{(2)}_j\}_{j\in\mathcal{N}_c}$ can be constructed by using $\{S_j\}_{j\in\mathcal{N}_c},X^{(1)},\{\Lambda^{(1)}_{i|j}\}_{(i,j)\in\mathcal{E}_c}$ based on (H.22), in which the last set can be determined using $\{S_j\}_{j\in\mathcal{N}_c},\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c}$; (c) also follows from (H.22); and (d) follows from the definition of conditional mutual information and S_i being independent of both $\{S_j\}_{j\in\mathcal{N}_c}$ and $\{\Lambda^{(0)}_{i|j}\}_{(i,j)\in\mathcal{E}_c}$. □

References

[1] M. Anderson, *Technology device ownership, 2015*, Pew Research Center, 2015.

[2] J. Poushter and others, "Smartphone ownership and internet usage continues to climb in emerging economies," *Pew Research Center*, vol. 22, pp. 1–44, 2016.

[3] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Process. Magazine, vol. 30, no. 1, pp. 82–105*, 2013.

[4] Q. Li, I. Cascudo, and M. G. Christensen, "Privacy-preserving distributed average consensus based on additive secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

[5] N. Gupta, J. Katz, N. Chopra, "Privacy in distributed average consensus," *IFAC-PapersOnLine, vol. 50, no. 1, pp. 9515-9520*, 2017.

[6] N. Gupta, J. Kat and N. Chopra, "Statistical privacy in distributed average consensus on bounded real inputs," in *ACC, pp 1836-1841*, 2019.

[7] Q. Li, R. Heusdens and M. G. Christensen, "Convex optimisation-based privacy-preserving distributed average consensus in wireless sensor networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp. 5895-5899*, 2020.

[8] M. Kefayati, M. S. Talebi, B. H. Khalajand H. R. Rabiee , "Secure consensus averaging in sensor networks using random offsets," in *Proc. of the IEEE Int. Conf. on Telec., and Malaysia Int. Conf. on Commun., pp. 556–560*, 2007.

[9] Z. Huang, S. Mitra, and G. Dullerud, "Differentially private iterative synchronous consensus," in *ACM workshop Privacy electron. Soc., pp. 81–90*, 2012.

[10] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private average consensus: Obstructions, trade-offs, and optimal algorithm design," *Automatica, vol. 81, pp. 221–231*, 2017.

[11] N. E. Manitara and C. N. Hadjicostis, "Privacy-preserving asymptotic average consensus," in *ECC, pp. 760–765*, 2013.

[12] Y. Mo and R. M. Murray, "Privacy preserving average consensus," *IEEE Trans. Automat Contr., vol. 62, no. 2, pp. 753–765*, 2017.

[13] J. He, L. Cai, C. Zhao, P. Cheng, X. Guan, "Privacy-preserving average consensus: privacy analysis and algorithm design," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 1, pp. 127–138*, 2019.

[14] P. Braca, R. Lazzeretti, S. Marano, and V. Matta, "Learning with privacy in consensus + obfuscation," *IEEE signal process. Lett., vol. 23, no. 9, pp. 1174–1178*, 2016.

[15] M. T. Hale, M. Egerstedt, "Differentially private cloud-based multi-agent optimization with constraints," in *Proc. American Control Conf., pp. 1235-1240*, 2015.

[16] M. T. Hale, M. Egerstedt, "Cloud-enabled differentially private multiagent optimization with constraints," *IEEE Trans. Control Netw. Syst., vol. 5, no. 4, pp. 1693–1706*, 2018.

[17] K. Tjell and R. Wisniewski, "Privacy preservation in distributed optimization via dual decomposition and ADMM," in *Proc. IEEE 58th Conf. Decis. Control., pp. 7203–7208*, 2020.

[18] Q. Li, R. Heusdens and M. G. Christensen, "Convex optimization-based privacy-preserving distributed least squares via subspace perturbation," in *Proc. Eur. Signal Process. Conf., to appear*, 2020.

[19] K. Tjell, I. Cascudo and R. Wisniewski, "Privacy preserving recursive least squares solutions," in *Proc. Eur. Control Conf., pp.3490–3495*, 2019.

[20] Z. Huang, S. Mitra, and N. Vaidya, "Differentially private distributed optimization., pp. 1–10," in *Proc. Int. Conf. Distrib. Comput. Netw*, 2015.

[21] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Trans. Autom. Control., vol. 62, no. 1, pp 50-64*, 2016.

[22] E. Nozari, P. Tallapragada, and J. Cortés, "Differentially private distributed convex optimization via functional perturbation," *IEEE Trans. Control Netw. Syst., vol. 5, no. 1, pp 395-408*, 2018.

[23] T. Zhang and Q. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Trans. Inf. Forensics Security, vol. 12, no. 1, pp. 172–187*, 2016.

[24] X. Zhang, M. M. Khalili, and M. Liu, "Recycled ADMM: Improve privacy and accuracy with less computation in distributed algorithms," in *in Proc. 56th Annu. Allerton Conf. Commun., Control, Comput. pp.959–965*, 2018.

[25] X. Zhang, M. M. Khalili, and M. Liu, "Improving the privacy and accuracy of ADMM-based distributed algorithms," *Proc. Int. Conf. Mach. Lear. pp. 5796–5805*, 2018.

[26] Y. Xiong, J. Xu, K. You, J. Liu and L. Wu, "Privacy preserving distributed online optimization over unbalanced digraphs via subgradient rescaling," *IEEE Trans. Control Netw. Syst.*, 2020.

[27] Q. Li, R. Heusdens and M. G. Christensen, "Privacy-preserving distributed optimization via subspace perturbation: A general framework," in *IEEE Trans. Signal Process., vol. 68, pp. 5983 - 5996*, 2020.

[28] Q. Li, M. Coutino, G. Leus and M. G. Christensen, "Privacy-preserving distributed graph filtering," in *Proc. Eur. Signal Process. Conf., to appear*, 2020.

[29] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO, pp. 643–662*. Springer, 2012.

[30] R. Cramer, I. B. Damgård, and J. B. Nielsen, *Secure Multiparty Computation and Secret Sharing*, Cambridge University Press, 2015.

[31] C. Dwork, "Differential privacy," in *ICALP, pp. 1–12*, 2006.

[32] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proc. 41st Annu. ACM Symp. Theory Comput., pp. 371-380*, 2009.

[33] D. Kifer and A. Machanavajjhala, "No free lunch in data privacy," in *SIGMOD, pp. 193–204*, 2011.

[34] D. Bogdanov, S. Laur, J. Willemson, "Sharemind: A framework for fast privacy-preserving computations," in *Proc. 13th Eur. Symp. Res. Comput. Security: Comput. Security, pp. 192-206,*, 2008.

[35] Q. Li and M. G. Christensen, "A privacy-preserving asynchronous averaging algorithm based on shamir's secret sharing," in *Proc. Eur. Signal Process. Conf., pp. 1-5*, 2019.

[36] D. Dolev, C. Dwork, O. Waarts, M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach., vol. 40, no. 1, pp. 17-47,*, 1993.

[37] I. Wagner and D. Eckhoff, "Technical privacy metrics: a systematic survey," *ACM Comput. Surv., vol. 51, no. 3, pp. 1–38*, 2018.

[38] M. Gtz, A. Machanavajjhala, G. Wang, X. Xiao, J. Gehrke, "Publishing search logs—a comparative study of privacy guarantees," *IEEE Trans. Knowledge and Data Eng., vol. 24, no. 3, pp. 520–532*, 2011.

[39] A. Haeberlen, B. C. Pierce, A. Narayan, "Differential privacy under fire.," in *Proc. 20th USENIX Conf. Security., vol. 33*, 2011.

[40] A. Korolova, K. Kenthapadi, N. Mishra, A. Ntoulas, "Releasing search queries and clicks privately," in *Proc. Int'l Conf. World Wide Web, pp. 171–180*, 2009.

[41] P. Cuff and L. Yu, "Differential privacy as a mutual information constraint," in *Proc. 23rd ACM SIGSAC Conf. Comput. Commun. Secur., pp 43–54*, 2016.

[42] M. Lopuhaä-Zwakenberg, B. Škorić and N. Li, "Information-theoretic metrics for local differential privacy protocols," *arXiv preprint arXiv:1910.07826*, 2019.

[43] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Proc. IEEE Annu. Symp. Found. Comput. Sci., pp. 429–438*, 2013.

[44] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," in *NIPS., pp. 2879–2887*, 2014.

[45] T. M. Cover and J. A. Tomas, *Elements of information theory*, John Wiley & Sons, 2012.

[46] J. Pang, G. Cheung, A. Ortega, O. C. Au, "Optimal graph Laplacian regularization for natural image denoising," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp 2294-2298*, 2015.

[47] SK Narang, A Gadde, A Ortega, "Signal processing techniques for interpolation in graph structured data," in *Proc. Int. Conf. Acoust., Speech, Signal Process., pp 5445-5449*, 2013.

[48] A. Olshevsky and J. Tsitsiklis, "Convergence speed in distributed consensus and averaging," *SIAM J. Control Optim., vol. 48, no. 1, pp. 33–55*, 2009.

[49] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning, vol. 3, no. 1, pp. 1–122*, 2011.

[50] T. Sherson, R. Heusdens, W. B. Kleijn, "Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory," *IEEE Trans. Signal Inf. Process. Netw., vol. 5, no. 2, pp 334-347*, 2018.

[51] G. Zhang and R. Heusdens, "Distributed optimization using the primal-dual method of multipliers," *IEEE Trans. Signal Process., vol. 4, no. 1, pp. 173–187*, 2018.

[52] J. Dall and M. Christensen, "Random geometric graphs," *Physical review E, vol. 66, no. 1, pp. 016121*, 2002.

[53] G. Ver Steeg, "Non-parametric entropy estimation toolbox (npeet)," *https://github.com/gregversteeg/NPEET*, 2000.

ISSN (online): 2446-1628
ISBN (online): 978-87-7210-950-3

AALBORG UNIVERSITY PRESS