Communication Efficient Distributed Learning with Censored, Quantized, and Generalized Group ADMM

Chaouki ben Issaid
chaouki.benissaid@oulu.fi

Intelligent Connectivity and Networks/Systems (ICON) Group
Faculty of Information Technology and Electrical Engineering
University of Oulu

April 7, 2021

Outline

Relevant Papers

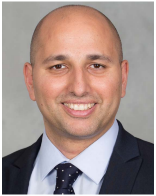
- Anis Elgabli, Jihong Park, Amrit S. Bedi, Mehdi Bennis and Vaneet Aggarwal, GADMM: Fast and Communication Efficient Framework for Distributed Machine Learning, Journal of Machine Learning Research, 2020.
- Anis Elgabli, Jihong Park, Amrit S Bedi, Chaouki Ben Issaid, Mehdi Bennis and Vaneet Aggarwal, Q-GADMM: Quantized Group ADMM for Communication Efficient Decentralized Machine Learning, IEEE Transactions on Communications, 2021.
- Chaouki Ben Issaid, Anis Elgabli, Jihong Park, Mehdi Bennis and Mérouane Debbah, Communication Efficient Distributed Learning with Censored, Quantized, and Generalized Group ADMM, arXiv:2009.06459, 2021.

Collaborators

Anis Elgabli
Postdoctoral Fellow
University of Oulu

Jihong Park
Lecturer
Deakin University

Mehdi Bennis
Associate Professor
University of Oulu



Mérouane Debbah
Director of Lagrange Mathematical and Computing Research Center
Huawei, Paris

Overview

Inroduction and Motivation

Centralized Learning

- For most traditional ML tasks, the training data are usually stored at a single computing unit.

Centralized Learning

- For most traditional ML tasks, the training data are usually stored at a single computing unit.
- When the dataset is **large-scale** and further contains **private information**, it is not feasible to communicate and process the entire dataset at a central location.

Centralized Learning

- For most traditional ML tasks, the training data are usually stored at a single computing unit.
- When the dataset is **large-scale** and further contains **private information**, it is not feasible to communicate and process the entire dataset at a central location.
- Critical point of failure.

Centralized Learning

- For most traditional ML tasks, the training data are usually stored at a single computing unit.
- When the dataset is **large-scale** and further contains **private information**, it is not feasible to communicate and process the entire dataset at a central location.
- Critical point of failure.
- When data samples are available at different devices with limited communication, it is often desirable to seek scalable learning methods that do not require assembling, storing, and processing data at one location.

Centralized Learning

- For most traditional ML tasks, the training data are usually stored at a single computing unit.
- When the dataset is **large-scale** and further contains **private information**, it is not feasible to communicate and process the entire dataset at a central location.
- Critical point of failure.
- When data samples are available at different devices with limited communication, it is often desirable to seek scalable learning methods that do not require assembling, storing, and processing data at one location.

$\rightarrow$ To design distributed and efficient learning algorithms to replace the centralized mode of operation.

Parameter Server Based Distributed Learning

- Data may either be artificially distributed onto a collection of workers, or it may already be physically collected by dispersed nodes/devices (e.g. wearables, wireless sensors, drones, robots or selfdriving automobiles).

Parameter Server Based Distributed Learning

- Data may either be artificially distributed onto a collection of workers, or it may already be physically collected by dispersed nodes/devices (e.g. wearables, wireless sensors, drones, robots or selfdriving automobiles).
- Each node is usually assigned a local computation task

Parameter Server Based Distributed Learning

- Data may either be artificially distributed onto a collection of workers, or it may already be physically collected by dispersed nodes/devices (e.g. wearables, wireless sensors, drones, robots or selfdriving automobiles).
- Each node is usually assigned a local computation task
- Goal: to enable the nodes to converge towards the global minimizer of a central learning model with the help of a central node (parameter server/master).

Parameter Server Based Distributed Learning

- The distributed learning problem is formulated as

$$\textbf{(P1)} \quad \boldsymbol{\Theta}^* := \arg\min_{\boldsymbol{\Theta}} \sum_{n=1}^{N} f_n(\boldsymbol{\Theta}). \tag{1}$$

Parameter Server Based Distributed Learning

- The distributed learning problem is formulated as

$$\textbf{(P1)} \quad \boldsymbol{\Theta}^* := \arg\min_{\boldsymbol{\Theta}} \sum_{n=1}^{N} f_n(\boldsymbol{\Theta}). \tag{1}$$

- N: number of workers.

Parameter Server Based Distributed Learning

- The distributed learning problem is formulated as

$$\textbf{(P1)} \quad \mathbf{\Theta}^* := \arg\min_{\mathbf{\Theta}} \sum_{n=1}^{N} f_n(\mathbf{\Theta}). \tag{1}$$

- N: number of workers.
- $\mathbf{\Theta} \in \mathbb{R}^d$: global model parameter.

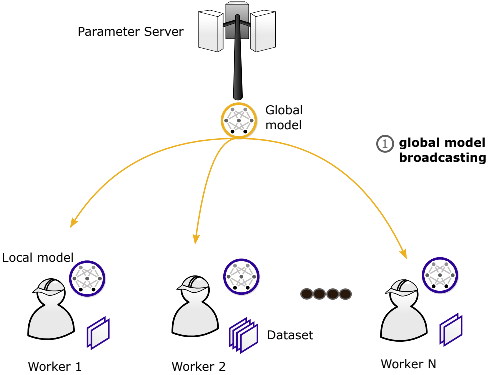
Parameter Server Based Distributed Learning

- The distributed learning problem is formulated as

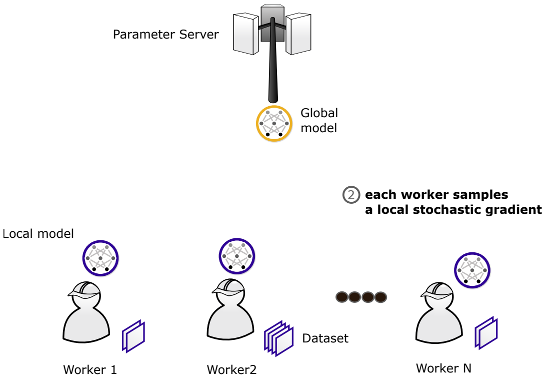
$$\textbf{(P1)} \quad \boldsymbol{\Theta}^* := \arg\min_{\boldsymbol{\Theta}} \sum_{n=1}^{N} f_n(\boldsymbol{\Theta}). \tag{1}$$

- N: number of workers.
- $\boldsymbol{\Theta} \in \mathbb{R}^d$: global model parameter.
- $f_n : \mathbb{R}^d \to \mathbb{R}$: local cost function of worker n.

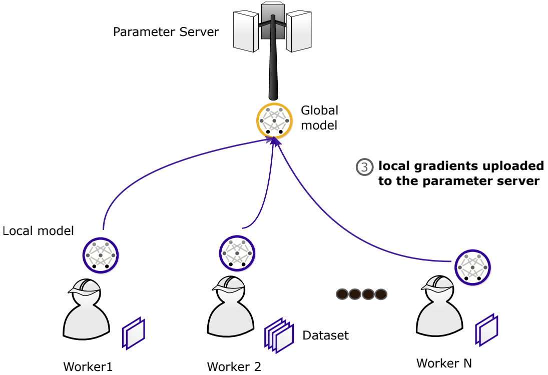
First Example: Mini-Batch SGD



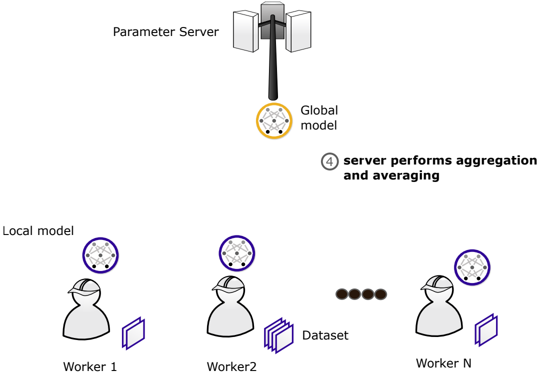
First Example: Mini-Batch SGD



First Example: Mini-Batch SGD



First Example: Mini-Batch SGD



Second Example: Parameter Server ADMM

Constrained formulation of **(P1)**:

$$\min_{\boldsymbol{\Theta},\{\boldsymbol{\theta}_n\}_{n=1}^N} \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n) \tag{2}$$

$$\text{s.t. } \boldsymbol{\theta}_n = \boldsymbol{\Theta}, \forall n \in \{1, \ldots, N\}.$$

Second Example: Parameter Server ADMM

Constrained formulation of **(P1)**:

$$\min_{\boldsymbol{\Theta},\{\boldsymbol{\theta}_n\}_{n=1}^N} \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n) \tag{2}$$

$$\text{s.t. } \boldsymbol{\theta}_n = \boldsymbol{\Theta}, \forall n \in \{1, \ldots, N\}.$$

1. Worker n solves the local problem to get its primal variable

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \{ f_n(\boldsymbol{\theta}_n) + \langle \boldsymbol{\lambda}_n^k, \boldsymbol{\theta}_n - \boldsymbol{\Theta}^k \rangle + \frac{\rho}{2} \|\boldsymbol{\theta}_n - \boldsymbol{\Theta}^k\|^2 \}. \tag{3}$$

Second Example: Parameter Server ADMM

Constrained formulation of **(P1)**:

$$\min_{\boldsymbol{\Theta}, \{\boldsymbol{\theta}_n\}_{n=1}^N} \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n) \tag{2}$$
$$\text{s.t. } \boldsymbol{\theta}_n = \boldsymbol{\Theta}, \forall n \in \{1, \ldots, N\}.$$

1. Worker n solves the local problem to get its primal variable

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \{ f_n(\boldsymbol{\theta}_n) + \langle \boldsymbol{\lambda}_n^k, \boldsymbol{\theta}_n - \boldsymbol{\Theta}^k \rangle + \frac{\rho}{2} \|\boldsymbol{\theta}_n - \boldsymbol{\Theta}^k\|^2 \}. \tag{3}$$

2. Parameter server updates its model

$$\boldsymbol{\Theta}^{k+1} = \frac{1}{N} \sum_{n=1}^{N} \left(\boldsymbol{\theta}_n^{k+1} + \frac{1}{\rho} \boldsymbol{\lambda}_n^k \right). \tag{4}$$

Second Example: Parameter Server ADMM

Constrained formulation of **(P1)**:

$$\min_{\boldsymbol{\Theta},\{\boldsymbol{\theta}_n\}_{n=1}^N} \sum_{n=1}^{N} f_n(\boldsymbol{\theta}_n) \tag{2}$$
$$\text{s.t. } \boldsymbol{\theta}_n = \boldsymbol{\Theta}, \forall n \in \{1,\dots,N\}.$$

1. Worker n solves the local problem to get its primal variable

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \{f_n(\boldsymbol{\theta}_n) + \langle \boldsymbol{\lambda}_n^k, \boldsymbol{\theta}_n - \boldsymbol{\Theta}^k \rangle + \frac{\rho}{2}\|\boldsymbol{\theta}_n - \boldsymbol{\Theta}^k\|^2\}. \tag{3}$$

2. Parameter server updates its model

$$\boldsymbol{\Theta}^{k+1} = \frac{1}{N}\sum_{n=1}^{N}\left(\boldsymbol{\theta}_n^{k+1} + \frac{1}{\rho}\boldsymbol{\lambda}_n^k\right). \tag{4}$$

3. Worker n updates its dual variable

$$\boldsymbol{\lambda}_n^{k+1} = \boldsymbol{\lambda}_n^k + \rho\left(\boldsymbol{\theta}_n^{k+1} - \boldsymbol{\Theta}^{k+1}\right). \tag{5}$$

Towards Communication-Efficient Approaches

- As the number of workers increases, the computational cost reduces but the communication cost increases.
- In parameter server applications, **communication is the bottleneck**.

Towards Communication-Efficient Approaches

- As the number of workers increases, the computational cost reduces but the communication cost increases.
- In parameter server applications, **communication is the bottleneck**.

Towards Communication-Efficient Approaches

- As the number of workers increases, the computational cost reduces but the communication cost increases.
- In parameter server applications, **communication is the bottleneck**.

> - To reduce the frequency of communication by performing more work locally on each worker (e.g. FedAvg).
> - To reduce the number of workers communicating each round.
> - To change the topology of the network $\rightarrow$ decentralized setting.
> - To send less information to reduce communication cost (e.g. quantization).

Towards Communication-Efficient Approaches

- As the number of workers increases, the computational cost reduces but the communication cost increases.
- In parameter server applications, **communication is the bottleneck**.

- To reduce the frequency of communication by performing more work locally on each worker (e.g. FedAvg).
- To reduce the number of workers communicating each round.
- To change the topology of the network $\rightarrow$ decentralized setting.
- To send less information to reduce communication cost (e.g. quantization).

This talk

Why Decentralized Learning?

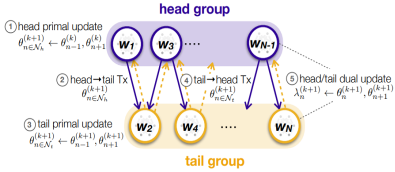
- Parameter server based architectures are not ideal for on-device intelligence settings for various reasons
  1. Transmission of information between the central node and the devices can be expensive especially when communication is conducted via multi-hop relays or when the devices are moving.
  2. Privacy and secrecy considerations where individual nodes may be reluctant to share information with a remote center.
  3. Efficiency depends on the slowest worker $\rightarrow$ as the number of workers increases, the uplink communication resources become the bottleneck.
- Alternative: extract information in a decentralized manner to avoid high latency and communication costs (number of active IoT devices is expected to be over 75 billion by 2025[1]).

[1] "Internet of things-number of connected devices worldwide 2015-2025", 2016.

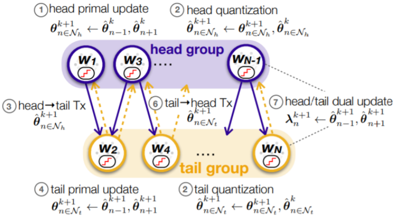
Censored and Quantized Generalized Group ADMM

Group ADMM

- A connected network wherein a set $\mathcal{V}$ of N workers aim to reach a consensus around a solution of the global optimization problem.



(a) GADMM [1]



(b) QGADMM [2]

[1] A. Elgabli, J. Park, AS Bedi, M. Bennis, and V. Aggarwal, "GADMM: Fast and Communication Efficient Framework for Distributed Machine Learning", JMLR, 21 (76), pp 1-39, 2020.

[2] A. Elgabli, J. Park, A. S. Bedi, C. Ben Issaid, M. Bennis and V. Aggarwal, "Q-GADMM: Quantized Group ADMM for Communication Efficient Decentralized Machine Learning," in IEEE TCOM, vol. 69, no. 1, pp. 164-181, Jan. 2021.

Problem Formulation

- The connections among workers are represented as an undirected communication graph $\mathcal{G}$ having the set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ of edges.

Problem Formulation

- The connections among workers are represented as an undirected communication graph $\mathcal{G}$ having the set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ of edges.
- The set of neighbors of worker n is defined as $\mathcal{N}_n = \{m | (n, m) \in \mathcal{E}\}$ whose cardinality is $|\mathcal{N}_n| = d_n$.

Problem Formulation

- The connections among workers are represented as an undirected communication graph $\mathcal{G}$ having the set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ of edges.
- The set of neighbors of worker n is defined as $\mathcal{N}_n = \{m | (n, m) \in \mathcal{E}\}$ whose cardinality is $|\mathcal{N}_n| = d_n$.
- We assume that the communication graph $\mathcal{G}$ is **bipartite** and **connected**.
- In this case, workers are divided into two groups: a *head group* $\mathcal{H}$, and a *tail group* $\mathcal{T}$. Each head worker in $\mathcal{H}$ can only communicate with tail workers in $\mathcal{T}$, and vice versa.

Problem Formulation

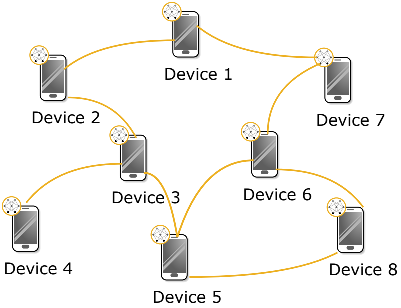
- The connections among workers are represented as an undirected communication graph $\mathcal{G}$ having the set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ of edges.
- The set of neighbors of worker n is defined as $\mathcal{N}_n = \{m | (n, m) \in \mathcal{E}\}$ whose cardinality is $|\mathcal{N}_n| = d_n$.
- We assume that the communication graph $\mathcal{G}$ is **bipartite** and **connected**.
- In this case, workers are divided into two groups: a *head group* $\mathcal{H}$, and a *tail group* $\mathcal{T}$. Each head worker in $\mathcal{H}$ can only communicate with tail workers in $\mathcal{T}$, and vice versa.
- **(P1)** is equivalent to the following problem

$$\textbf{(P2)} \quad \min_{\{\boldsymbol{\theta}_n\}_{n=1}^N} \sum_{n=1}^N f_n(\boldsymbol{\theta}_n) \tag{6}$$

$$\text{s.t. } \boldsymbol{\theta}_n = \boldsymbol{\theta}_m, \forall (n, m) \in \mathcal{E},$$

Examples

- In wireless sensor networks, neighboring nodes can be devices that are within the range of radio broadcasting.
- In smart phone networks, the neighbors can be devices that are within the same local area network.



Schematic Illustration

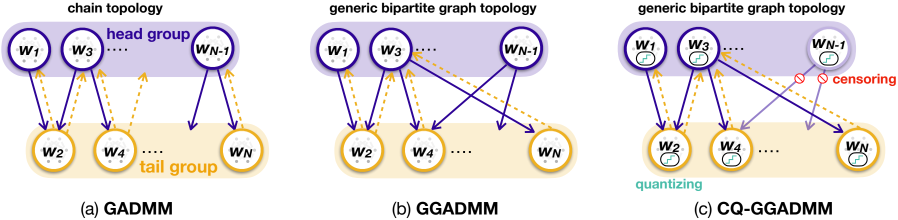


Figure 1: (a) *group ADMM (GADMM)*, the baseline algorithm under a chain topology, compared to our proposed (b) *generalized GADMM (GGADMM)* under a generic bipartite graph topology, and (c) *censored-and-quantized GGADMM (CQ-GGADMM)* that additionally applies link censoring for negligible updates after quantization.

CQ-GGADMM in a Nutshell

CQ-GGADMM exploits two key principles to improve the communication efficiency

CQ-GGADMM in a Nutshell

CQ-GGADMM exploits two key principles to improve the communication efficiency

1. To reduce **the communication payload size per each link** by applying a heterogeneous stochastic quantization scheme that decreases the number of bits to represent each model parameter,

CQ-GGADMM in a Nutshell

CQ-GGADMM exploits two key principles to improve the communication efficiency

1. To reduce **the communication payload size per each link** by applying a heterogeneous stochastic quantization scheme that decreases the number of bits to represent each model parameter,
2. To reduce **the number of communication links per round** by exploiting a censoring approach that allows to exchange model parameters only when the updated quantized model is sufficiently changed from the previous quantized model.

Key Assumptions

Key Assumptions

- **A1.** The communication graph $\mathcal{G}$ is bipartite and connected.

Key Assumptions

- **A1.** The communication graph $\mathcal{G}$ is bipartite and connected.
- **A2.** There exists an optimal solution set to **(P1)** which has at least one finite element.

Key Assumptions

- **A1.** The communication graph $\mathcal{G}$ is bipartite and connected.
- **A2.** There exists an optimal solution set to **(P1)** which has at least one finite element.
- **A3.** The local cost functions f_n are strongly convex with parameter $\mu_n > 0$.

Key Assumptions

- **A1.** The communication graph $\mathcal{G}$ is bipartite and connected.
- **A2.** There exists an optimal solution set to **(P1)** which has at least one finite element.
- **A3.** The local cost functions f_n are strongly convex with parameter $\mu_n > 0$.
- **A4.** The local cost functions f_n have L_n-Lipschitz continuous gradient ($L_n > 0$).

Notations

- $\hat{\mathcal{Q}}_n$: the quantized version of the primal model at worker n

Notations

- $\hat{\mathcal{Q}}_n$: the quantized version of the primal model at worker n
- $\hat{\theta}_n$: the sequence exchanged between worker n and its neighbours

Notations

- $\hat{\mathcal{Q}}_n$: the quantized version of the primal model at worker n
- $\hat{\boldsymbol{\theta}}_n$: the sequence exchanged between worker n and its neighbours
- $\tau_0 \xi^k$: the censoring sequence where $\tau_0 > 0$ and $\xi \in (0, 1)$

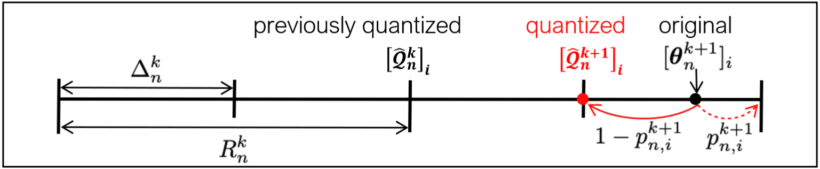
Notations

- $\hat{\mathcal{Q}}_n$: the quantized version of the primal model at worker n
- $\hat{\theta}_n$: the sequence exchanged between worker n and its neighbours
- $\tau_0 \xi^k$: the censoring sequence where $\tau_0 > 0$ and $\xi \in (0,1)$

Communication-censoring Strategy

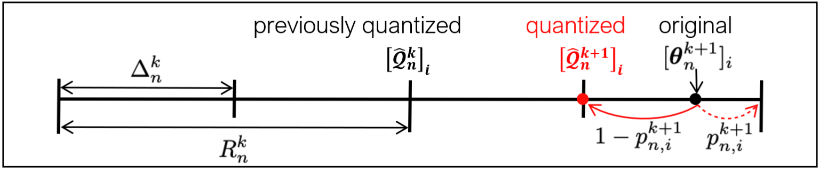
$$\hat{\theta}_n^{k+1} = \begin{cases} \hat{\mathcal{Q}}_n^{k+1}, & \text{if } \|\hat{\theta}_n^k - \hat{\mathcal{Q}}_n^{k+1}\| \geq \tau_0 \xi^{k+1} \\ \hat{\theta}_n^k, & \text{otherwise} \end{cases} \tag{7}$$

Heterogeneous Stochastic Quantization



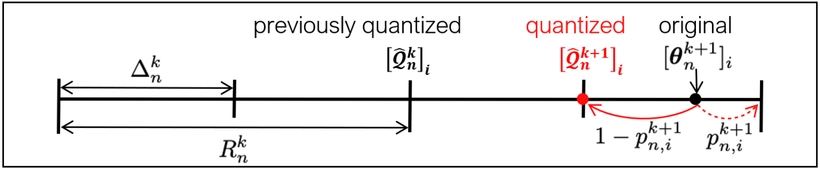
- At iteration k, each worker n quantizes the difference between the current model and the previous quantized model vector: $\boldsymbol{\theta}_n^k - \hat{\mathcal{Q}}_n^{k-1} = Q_n(\boldsymbol{\theta}_n^k, \hat{\mathcal{Q}}_n^{k-1})$.

Heterogeneous Stochastic Quantization



- At iteration k, each worker n quantizes the difference between the current model and the previous quantized model vector: $\boldsymbol{\theta}_n^k - \hat{\mathcal{Q}}_n^{k-1} = Q_n(\boldsymbol{\theta}_n^k, \hat{\mathcal{Q}}_n^{k-1})$.
- $Q_n(\cdot)$ is a stochastic quantization operator that depends on
  - $p_{n,i}^k$: quantization probability for the i^{th} component of each model vector,
  - b_n^k: number of bits used for representing each model vector dimension.

Heterogeneous Stochastic Quantization



- At iteration k, each worker n quantizes the difference between the current model and the previous quantized model vector: $\boldsymbol{\theta}_n^k - \hat{\mathcal{Q}}_n^{k-1} = Q_n(\boldsymbol{\theta}_n^k, \hat{\mathcal{Q}}_n^{k-1})$.
- $Q_n(\cdot)$ is a stochastic quantization operator that depends on
  - $p_{n,i}^k$: quantization probability for the i^{th} component of each model vector,
  - b_n^k: number of bits used for representing each model vector dimension.
- $\Delta_n^k = 2R_n^k/(2^{b_n^k} - 1)$: quantization step size where $2R_n^k$ is the quantization range.

Heterogeneous Stochastic Quantization

- Non-increasing step size: $\Delta_n^k \leq \omega \Delta_n^{k-1}$, $\omega \in (0, 1]$.

Heterogeneous Stochastic Quantization

- Non-increasing step size: $\Delta_n^k \leq \omega \Delta_n^{k-1}$, $\omega \in (0, 1]$.
- To ensure the above, b_n^k should satisfy

$$b_n^k \geq \left\lceil \log_2 \left(1 + (2^{b_n^{k-1}} - 1) R_n^k / (\omega R_n^{k-1}) \right) \right\rceil . \tag{8}$$

Heterogeneous Stochastic Quantization

- Non-increasing step size: $\Delta_n^k \leq \omega \Delta_n^{k-1}$, $\omega \in (0, 1]$.
- To ensure the above, b_n^k should satisfy

$$b_n^k \geq \left\lceil \log_2 \left(1 + (2^{b_n^{k-1}} - 1) R_n^k / (\omega R_n^{k-1}) \right) \right\rceil . \tag{8}$$

- Define the function $q_n(\cdot)$ as

$$[q_n(\boldsymbol{\theta}_n^k)]_i = \begin{cases} \lceil [c_n(\boldsymbol{\theta}_n^k)]_i \rceil & \text{with probability } p_{n,i}^k, \\ \lfloor [c_n(\boldsymbol{\theta}_n^k)]_i \rfloor & \text{with probability } 1 - p_{n,i}^k, \end{cases} \tag{9}$$

where $[c_n(\boldsymbol{\theta}_n^k)]_i = \frac{1}{\Delta_n^k} \left([\boldsymbol{\theta}_n^k]_i - [\hat{\boldsymbol{\mathcal{Q}}}_n^{k-1}]_i + R_n^k \right)$.

Heterogeneous Stochastic Quantization

- Non-increasing step size: $\Delta_n^k \leq \omega \Delta_n^{k-1}$, $\omega \in (0, 1]$.
- To ensure the above, b_n^k should satisfy

$$b_n^k \geq \left\lceil \log_2 \left(1 + (2^{b_n^{k-1}} - 1) R_n^k / (\omega R_n^{k-1})\right) \right\rceil . \tag{8}$$

- Define the function $q_n(\cdot)$ as

$$[q_n(\boldsymbol{\theta}_n^k)]_i = \begin{cases} \lceil [c_n(\boldsymbol{\theta}_n^k)]_i \rceil & \text{with probability } p_{n,i}^k, \\ \lfloor [c_n(\boldsymbol{\theta}_n^k)]_i \rfloor & \text{with probability } 1 - p_{n,i}^k, \end{cases} \tag{9}$$

where $[c_n(\boldsymbol{\theta}_n^k)]_i = \frac{1}{\Delta_n^k} \left([\boldsymbol{\theta}_n^k]_i - [\hat{\boldsymbol{\mathcal{Q}}}_n^{k-1}]_i + R_n^k \right)$.

- At the neighbours of worker n: $\hat{\boldsymbol{\mathcal{Q}}}_n^k = \hat{\boldsymbol{\mathcal{Q}}}_n^{k-1} + \Delta_n^k q_n(\boldsymbol{\theta}_n^k) - R_n^k \mathbf{1}$.

CQ-GGADMM Steps

1. Primal variables for head workers are found using

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \; f_n(\boldsymbol{\theta}_n) + \langle \boldsymbol{\theta}_n, \boldsymbol{\alpha}_n^k - \rho \sum_{m \in \mathcal{N}_n} \hat{\boldsymbol{\theta}}_m^k \rangle + \frac{\rho}{2} d_n \|\boldsymbol{\theta}_n\|^2. \quad (10)$$

CQ-GGADMM Steps

1. Primal variables for head workers are found using

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \; f_n(\boldsymbol{\theta}_n) + \langle \boldsymbol{\theta}_n, \boldsymbol{\alpha}_n^k - \rho \sum_{m \in \mathcal{N}_n} \hat{\boldsymbol{\theta}}_m^k \rangle + \frac{\rho}{2} d_n \|\boldsymbol{\theta}_n\|^2. \tag{10}$$

2. Primal variables update for tail workers is done as follow

$$\boldsymbol{\theta}_m^{k+1} = \arg\min_{\boldsymbol{\theta}_m} \; f_m(\boldsymbol{\theta}_m) + \langle \boldsymbol{\theta}_m, \boldsymbol{\alpha}_m^k - \rho \sum_{n \in \mathcal{N}_m} \hat{\boldsymbol{\theta}}_n^{k+1} \rangle + \frac{\rho}{2} d_m \|\boldsymbol{\theta}_m\|^2. \tag{11}$$

CQ-GGADMM Steps

1. Primal variables for head workers are found using

$$\boldsymbol{\theta}_n^{k+1} = \arg\min_{\boldsymbol{\theta}_n} \; f_n(\boldsymbol{\theta}_n) + \langle \boldsymbol{\theta}_n, \boldsymbol{\alpha}_n^k - \rho \sum_{m\in\mathcal{N}_n} \hat{\boldsymbol{\theta}}_m^k \rangle + \frac{\rho}{2} d_n \|\boldsymbol{\theta}_n\|^2. \tag{10}$$

2. Primal variables update for tail workers is done as follow

$$\boldsymbol{\theta}_m^{k+1} = \arg\min_{\boldsymbol{\theta}_m} \; f_m(\boldsymbol{\theta}_m) + \langle \boldsymbol{\theta}_m, \boldsymbol{\alpha}_m^k - \rho \sum_{n\in\mathcal{N}_m} \hat{\boldsymbol{\theta}}_n^{k+1} \rangle + \frac{\rho}{2} d_m \|\boldsymbol{\theta}_m\|^2. \tag{11}$$

3. Dual variable of each worker is updated locally

$$\boldsymbol{\alpha}_n^{k+1} = \boldsymbol{\alpha}_n^k + \rho \sum_{m\in\mathcal{N}_n} (\hat{\boldsymbol{\theta}}_n^{k+1} - \hat{\boldsymbol{\theta}}_m^{k+1}), \; \forall n \in \mathcal{V}. \tag{12}$$

where $\boldsymbol{\alpha}_n = \sum_{m\in\mathcal{N}_n} \boldsymbol{\lambda}_{n,m}$

Main Theorem

Convergence of CQ-GGADMM

Suppose that assumptions **A1-A4** hold and the dual variable $\boldsymbol{\alpha}$ is initialized such that $\boldsymbol{\alpha}^0$ lies in the column space of the oriented incidence matrix $\boldsymbol{M}_{-}$.
For sufficiently small ρ, the sequence of iterates of CQ-GGADMM converges linearly with a rate $(1+\delta_2)/2$ where $\delta_2 = \max\{(1+\kappa)^{-1}, \psi^2\}$ and $\psi = \max\{\xi, \omega\}$.

Main Theorem

Convergence of CQ-GGADMM

Suppose that assumptions **A1-A4** hold and the dual variable $\boldsymbol{\alpha}$ is initialized such that $\boldsymbol{\alpha}^0$ lies in the column space of the oriented incidence matrix $\boldsymbol{M}_-$.
For sufficiently small ρ, the sequence of iterates of CQ-GGADMM converges linearly with a rate $(1+\delta_2)/2$ where $\delta_2 = \max\{(1+\kappa)^{-1}, \psi^2\}$ and $\psi = \max\{\xi, \omega\}$.

The constant κ depends on:

- the network topology through the maximum and minimum non-zero singular values of $\boldsymbol{M}_-$,
- the properties of the local objective functions ($\mu = \min_{1\le n\le N} \mu_n$ and $L = \max_{1\le n\le N} L_n$),
- the penalty parameter ρ.

Numerical Results

Simulation Settings

Simulation Settings

- **Datasets:** (*i*) linear regression (Body Fat dataset (Dua and Graff, 2017)), and (*ii*) logistic regression (Derm dataset (Dua and Graff, 2017)).

Simulation Settings

- **Datasets:** (*i*) linear regression (Body Fat dataset (Dua and Graff, 2017)), and (*ii*) logistic regression (Derm dataset (Dua and Graff, 2017)).
- **Graph generation:** We generate randomly a network consisting of N workers with a connectivity ratio p (the actual number of edges divided by the number of edges for a fully connected graph) (Shi et al., 2014).

Simulation Settings

- **Datasets:** (*i*) linear regression (Body Fat dataset (Dua and Graff, 2017)), and (*ii*) logistic regression (Derm dataset (Dua and Graff, 2017)).
- **Graph generation:** We generate randomly a network consisting of N workers with a connectivity ratio p (the actual number of edges divided by the number of edges for a fully connected graph) (Shi et al., 2014).
- **Communication energy:**
  - Consumed energy defined as $E = P\tau$.
  - Transmission power: $P = \tau D^2 N_0 B_n (2^{R/B_n} - 1)$.
  - Upload/download transmission time: $\tau = 1ms$.
  - Transmission rate: $R = (32d/1ms) bits/sec$.
  - Power spectral density: $N_0 = 10^{-6} W/Hz$.

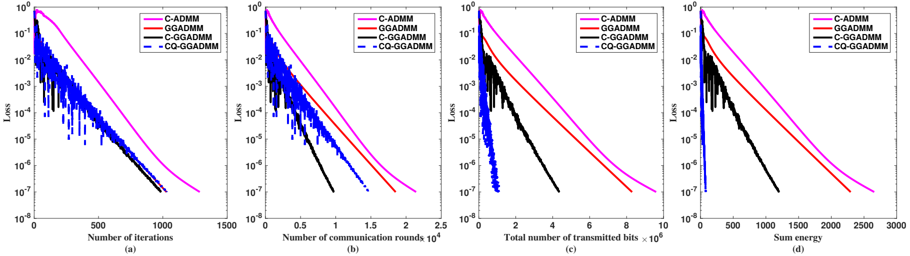


Figure 2: *Linear regression* results on real dataset showing: (a) loss w.r.t. # iterations; (b) loss w.r.t. # communication rounds; (c) loss w.r.t. # transmitted bits; (d) energy efficiency (loss w.r.t. total energy), the number of workers is 18.

Logistic Regression

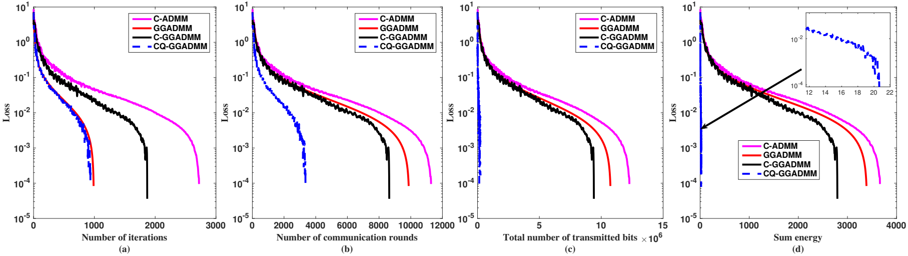
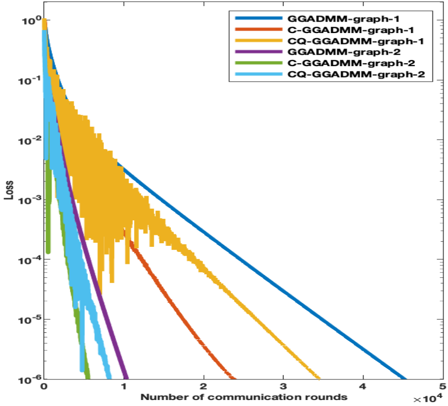


Figure 3: *Logistic regression* results on real dataset showing: (a) loss w.r.t. # iterations; (b) loss w.r.t. # communication rounds; (c) loss w.r.t. # transmitted bits; (d) energy efficiency (loss w.r.t. total energy), the number of workers is 18.

Impact of the Network Graph Density



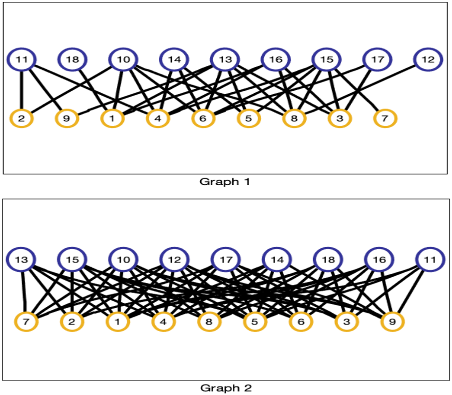


Figure 4: *Effect of the graph density on the performance of the algorithms*: loss w.r.t. # communication rounds (left), Graph 1: Sparse graph (right top); Graph 2: dense graph (right bottom). The number of workers is 18, and the task is linear regression on real dataset.

Conclusion and Future Work

- We proposed CQ-GGADMM, a communication-efficient primal-dual approach, that exploits **spatial sparsity** (GGADMM), **temporal sparsity** (censoring), and **payload size reduction** (quantization).

Conclusion and Future Work

- We proposed CQ-GGADMM, a communication-efficient primal-dual approach, that exploits **spatial sparsity** (GGADMM), **temporal sparsity** (censoring), and **payload size reduction** (quantization).
- Some potential extensions:

Conclusion and Future Work

- We proposed CQ-GGADMM, a communication-efficient primal-dual approach, that exploits **spatial sparsity** (GGADMM), **temporal sparsity** (censoring), and **payload size reduction** (quantization).
- Some potential extensions:
  - Stochastic non-convex setting (e.g. DNN).

Conclusion and Future Work

- We proposed CQ-GGADMM, a communication-efficient primal-dual approach, that exploits **spatial sparsity** (GGADMM), **temporal sparsity** (censoring), and **payload size reduction** (quantization).
- Some potential extensions:
  - Stochastic non-convex setting (e.g. DNN).
  - Wireless connectivity modeling of GGADMM.

Conclusion and Future Work

- We proposed CQ-GGADMM, a communication-efficient primal-dual approach, that exploits **spatial sparsity** (GGADMM), **temporal sparsity** (censoring), and **payload size reduction** (quantization).
- Some potential extensions:
  - Stochastic non-convex setting (e.g. DNN).
  - Wireless connectivity modeling of GGADMM.
  - Time-varying topology.

Conclusion and Future Work

- We proposed CQ-GGADMM, a communication-efficient primal-dual approach, that exploits **spatial sparsity** (GGADMM), **temporal sparsity** (censoring), and **payload size reduction** (quantization).
- Some potential extensions:
  - Stochastic non-convex setting (e.g. DNN).
  - Wireless connectivity modeling of GGADMM.
  - Time-varying topology.
  - Communication and computation efficiency: running few local SGD instead of solving the exact local problem.

Thank you!

- Chaouki Ben Issaid, Anis Elgabli, Jihong Park, Mehdi Bennis and Mérouane Debbah, Communication Efficient Distributed Learning with Censored, Quantized, and Generalized Group ADMM, arXiv:2009.06459, 2021.