Compressed Distributed Gradient Descent: Communication-Efficient Consensus over Networks

Xin Zhang
Joint Work with Dr.Jia Liu and Dr.Zhengyuan Zhu

Dept. of Statistics
Iowa State University

Dec. 08, 2018

Network Consensus Optimization

Wireless Networks

Distributed Machine Learning

Tactical Multi-Agent Swarm

Smart Power Grid

Network Consensus Optimization

- First Order Method:
  - Decentralized Gradient Descent ([Nedic, Ozdaglar'09],[Shi et al.'15], [Yuan et al.'16], [Zeng, Yin'18])
  - Decentralized Stochastic Gradient Descent ([Lian et al.'17], [Jiang et al.'17], [Pu, Nedic'18])
- Second Order Method:
  - Decentralized Newton Method ([Mansoori, Wei'18])
  - Decentralized Quasi Newton Method ([Eisen et al.'17])
- Dual-Primal Method:
  - Decentralized ADMM ([Shi et al.'14], [Ling, Ribeiro'14])

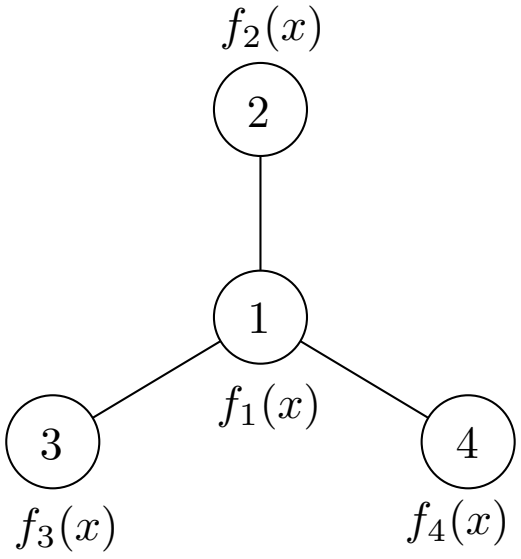
Network Consensus Optimization

- First Order Method:
  - Decentralized Gradient Descent ([Nedic, Ozdaglar'09],[Shi et al.'15], [Yuan et al.'16], [Zeng, Yin'18])
  - Decentralized Stochastic Gradient Descent ([Lian et al.'17], [Jiang et al.'17], [Pu, Nedic'18])
- Second Order Method:
  - Decentralized Newton Method ([Mansoori, Wei'18])
  - Decentralized Quasi Newton Method ([Eisen et al.'17])
- Dual-Primal Method:
  - Decentralized ADMM ([Shi et al.'14], [Ling, Ribeiro'14])

Mainly focus on Decentralized Gradient Descent

Network Consensus Optimization: The Basics

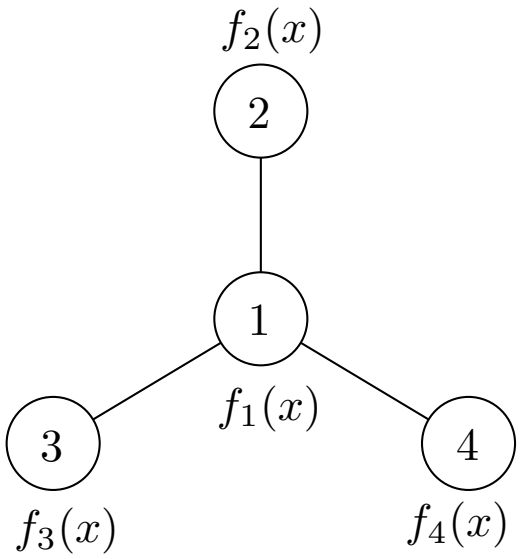
- A network $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, with $|\mathcal{N}| = N$, $|\mathcal{L}| = L$
- Let $x \in \mathbb{R}^P$ be a global state/decision variable
- Each node i can only access some local objective function $f_i(x)$
- Global objective function is: $\sum_{i=1}^{N} f_i(x)$



Network Consensus Optimization: The Basics

- Goal: To solve the following optimization problem distributively & collaboratively

$$\min_{x \in \mathbb{R}^P} f(x) = \min_{x \in \mathbb{R}^P} \sum_{i=1}^{N} f_i(x)$$



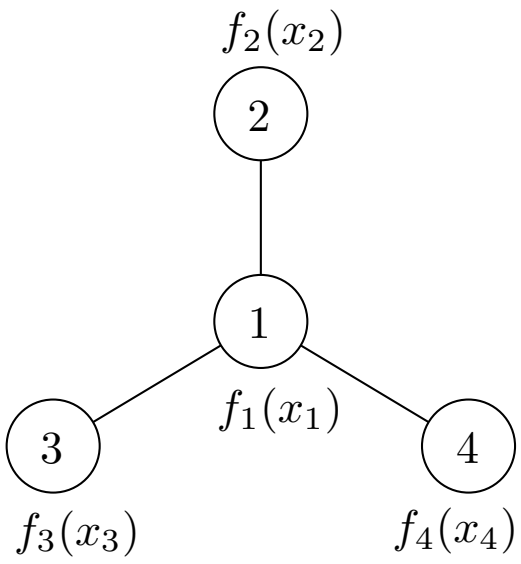
Network Consensus Optimization: The Basics

- Goal: To solve the following optimization problem distributively & collaboratively

$$\min_{x \in \mathbb{R}^P} f(x) = \min_{x \in \mathbb{R}^P} \sum_{i=1}^{N} f_i(x)$$

- Clearly, this problem can be rewritten in a consensus form:

$$\min_{x_i \in \mathbb{R}^P, \forall i} \left\{ \sum_{i=1}^{N} f_i(x_i) \middle| x_i = x_j, \forall (i,j) \in \mathcal{L} \right\}$$



Decentralized Gradient Descent Method (DGD)

- It has been shown in [Nedic, Ozdaglar'09] that

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{subject to} \quad & x_i = x_j, \qquad \forall (i,j) \in \mathcal{L}. \end{aligned}$$

can be rewritten as:

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{subject to} \quad & (\mathbf{W} \otimes \mathbf{I}_P)\mathbf{x} = \mathbf{x}, \end{aligned} \tag{1}$$

where $\mathbf{x} \triangleq [x_1^\top, \ldots, x_n^\top]^\top \in \mathbb{R}^{NP}$, $\mathbf{I}_P$ is the P-dimensional identity matrix, and the operator $\otimes$ is the Kronecker product.

Decentralized Gradient Descent Method (DGD)

In Problem (1), $\mathbf{W} \in \mathbb{R}^{N \times N}$ is called the consensus matrix and satisfies:

- $\mathbf{W}$ is doubly stochastic: $\sum_{i=1}^{N} [\mathbf{W}]_{ij} = \sum_{j=1}^{N} [\mathbf{W}]_{ij} = 1$.

Decentralized Gradient Descent Method (DGD)

In Problem (1), $\mathbf{W} \in \mathbb{R}^{N \times N}$ is called the consensus matrix and satisfies:

- $\mathbf{W}$ is doubly stochastic: $\sum_{i=1}^{N} [\mathbf{W}]_{ij} = \sum_{j=1}^{N} [\mathbf{W}]_{ij} = 1$.
- Sparsity pattern of $\mathbf{W}$ follows network topology: $[\mathbf{W}]_{ij} > 0$ for $\forall\, (i, j) \in \mathcal{L}$ and $[\mathbf{W}]_{ij} = 0$ otherwise.

Decentralized Gradient Descent Method (DGD)

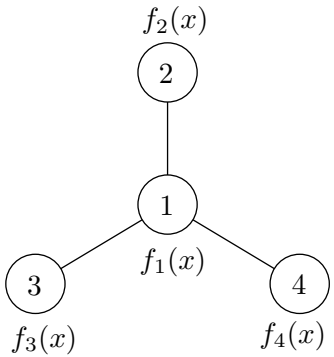
In Problem (1), $\mathbf{W} \in \mathbb{R}^{N \times N}$ is called the consensus matrix and satisfies:

- $\mathbf{W}$ is doubly stochastic: $\sum_{i=1}^{N} [\mathbf{W}]_{ij} = \sum_{j=1}^{N} [\mathbf{W}]_{ij} = 1$.
- Sparsity pattern of $\mathbf{W}$ follows network topology: $[\mathbf{W}]_{ij} > 0$ for $\forall\, (i,j) \in \mathcal{L}$ and $[\mathbf{W}]_{ij} = 0$ otherwise.
- $\mathbf{W}$ is symmetric and hence it has real eigenvalues in $(-1, 1]$ (thus can be sorted); Let β the 2nd-largest eigenvalue in magnitude.

Decentralized Gradient Descent Method (DGD)

In Problem (1), $\mathbf{W} \in \mathbb{R}^{N \times N}$ is called the consensus matrix and satisfies:

- $\mathbf{W}$ is doubly stochastic: $\sum_{i=1}^{N} [\mathbf{W}]_{ij} = \sum_{j=1}^{N} [\mathbf{W}]_{ij} = 1$.
- Sparsity pattern of $\mathbf{W}$ follows network topology: $[\mathbf{W}]_{ij} > 0$ for $\forall\, (i,j) \in \mathcal{L}$ and $[\mathbf{W}]_{ij} = 0$ otherwise.
- $\mathbf{W}$ is symmetric and hence it has real eigenvalues in $(-1, 1]$ (thus can be sorted); Let β the 2nd-largest eigenvalue in magnitude.



$$\mathbf{W} = \begin{bmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 3/4 & 0 & 0 \\ 1/4 & 0 & 3/4 & 0 \\ 1/4 & 0 & 0 & 3/4 \end{bmatrix}$$

Decentralized Gradient Descent [Nedic, Ozdaglar'09]

The DGD algorithm:

1. Initialization: Let $k = 1$. Choose initial values for $x_{i,1}$ and step-size α_1.

Decentralized Gradient Descent [Nedic, Ozdaglar'09]

The DGD algorithm:

1. Initialization: Let $k = 1$. Choose initial values for $x_{i,1}$ and step-size α_1.
2. In k-th iteration: Each node sends its local copy to its neighbors.

Decentralized Gradient Descent [Nedic, Ozdaglar'09]

The DGD algorithm:

1. Initialization: Let $k = 1$. Choose initial values for $x_{i,1}$ and step-size α_1.
2. In k-th iteration: Each node sends its local copy to its neighbors.
3. Upon reception of all local copies from its neighbors, each node updates its local copy:

$$x_{i,k+1} = \underbrace{\sum\nolimits_{j\in\mathcal{N}_i} [\mathbf{W}]_{ij} x_{j,k}}_{\text{Consensus step}} - \underbrace{\alpha_k \nabla f_i(x_{i,k})}_{\text{Gradient step}},$$

where $\mathcal{N}_i \triangleq \{j \in \mathcal{N} : (i,j) \in \mathcal{L}\}$.

Decentralized Gradient Descent [Nedic, Ozdaglar'09]

Pros:

- Implementation Simplicity
- Elegant networking interpretation
- Convergence speed same as traditional gradient descent

Decentralized Gradient Descent [Nedic, Ozdaglar'09]

Pros:

- Implementation Simplicity
- Elegant networking interpretation
- Convergence speed same as traditional gradient descent

Cons:

- Performance not satisfactory for solving high-dimensional consensus optimization problems over network with low communication speed
  - Large amount of information exchange
  - Communication bottlenecks
  - Sharing info between nodes is time-consuming

Related Work on Communication-Efficient Methods

- Second-Order Approach (using second-order Hessian information)
  - [Jordan, Lee, Yang'16], [Wang, Kolar, Srebro, Zhang'16]
  - Key Idea: Treating local hessian as global hessian;
  - Pros: Faster convergence (hence less info exchanges);
  - Cons: High per-iteration complexity (matrix inversion, $\Omega(d^2 \log d)$ for d-dimensional problems);

Related Work on Communication-Efficient Methods

- Second-Order Approach (using second-order Hessian information)
  - [Jordan, Lee, Yang'16], [Wang, Kolar, Srebro, Zhang'16]
  - Key Idea: Treating local hessian as global hessian;
  - Pros: Faster convergence (hence less info exchanges);
  - Cons: High per-iteration complexity (matrix inversion, $\Omega(d^2 \log d)$ for d-dimensional problems);
- First-Order Approach (using only first-order gradient information)
  - [Berahas et al. '17]: DGD^t
  - [Tang et al. '18]: Quantized extrapolation (high complexity in determining step-size)

Related Work on Communication-Efficient Methods

- Second-Order Approach (using second-order Hessian information)
  - [Jordan, Lee, Yang'16], [Wang, Kolar, Srebro, Zhang'16]
  - Key Idea: Treating local hessian as global hessian;
  - Pros: Faster convergence (hence less info exchanges);
  - Cons: High per-iteration complexity (matrix inversion, $\Omega(d^2 \log d)$ for d-dimensional problems);
- First-Order Approach (using only first-order gradient information)
  - [Berahas et al. '17]: DGD^t
  - [Tang et al. '18]: Quantized extrapolation (high complexity in determining step-size)

All existing works suffer high complexity

A Natural Idea for Reducing Communication Cost

- Message Conpression
  - Idea: Instead of exact message(e.g. gradient, iterates), transmitting some imprecise but 'simple' message;
  - Method: Unbiased Stochastic Compression;
  - Related Work:
    - Ternary compression operator [Wen et al. NIPS '17]
    - Quantized compression operator [Alistarh et al. NIPS'17]
    - Sparsification operator [Wangni et al.NIPS'18]
  - Most of the works are based on Master/Slave system, could this technique improve network optimization algorithm?

Unbiased Stochastic Compression Operator

Definition 1 (Unbiased Stochastic Compression Operator)

A stochastic compression operator $C(\cdot)$ is unbiased if it satisfies $C(z) = z + \epsilon_z$, with $\mathbb{E}[\epsilon_z] = 0$ and $\mathbb{E}[\epsilon_z^2] \leq \sigma^2, \forall\, z$.

Unbiased Stochastic Compression Operator

Definition 1 (Unbiased Stochastic Compression Operator)

A stochastic compression operator $C(\cdot)$ is unbiased if it satisfies $C(z) = z + \epsilon_z$, with $\mathbb{E}[\epsilon_z] = 0$ and $\mathbb{E}[\epsilon_z^2] \leq \sigma^2, \forall\, z$.

- Quantized compression operator [Alistarh et al. NIPS'17]

$$[C(\mathbf{z})]_k = \begin{cases} \lfloor z_k \rfloor + 1, \text{with probability } p_k, \\ \lfloor z_k \rfloor, \text{with probability } (1 - p_k). \end{cases}$$

where $p_k = z_k - \lfloor z_k \rfloor$

Unbiased Stochastic Compression Operator

Definition 1 (Unbiased Stochastic Compression Operator)

A stochastic compression operator $C(\cdot)$ is unbiased if it satisfies $C(z) = z + \epsilon_z$, with $\mathbb{E}[\epsilon_z] = 0$ and $\mathbb{E}[\epsilon_z^2] \leq \sigma^2, \forall\, z$.

- Quantized compression operator [Alistarh et al. NIPS'17]

$$[C(\mathbf{z})]_k = \begin{cases} \lfloor z_k \rfloor + 1, \text{with probability } p_k, \\ \lfloor z_k \rfloor, \text{with probability } (1 - p_k). \end{cases}$$

where $p_k = z_k - \lfloor z_k \rfloor$

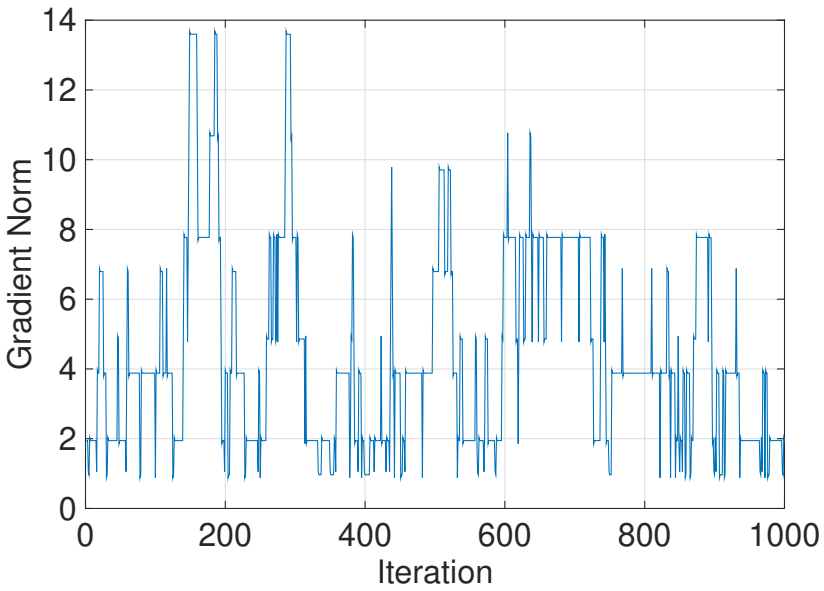
  - $\mathbb{E}[C(\mathbf{z})]_k = (\lfloor z_k \rfloor + 1) \times p_k + \lfloor z_k \rfloor \times (1 - p_k) = \lfloor z_k \rfloor + p_k = z_k$
  - $\text{Var}[C(\mathbf{z})]_k = (\lfloor z_k \rfloor + 1 - z_k)^2 \times p_k + (\lfloor z_k \rfloor - z_k)^2 \times (1 - p_k) \leq \frac{1}{4}$
  - stochastically mapping precise floating point number to inexact integer number;

Fundamental Questions

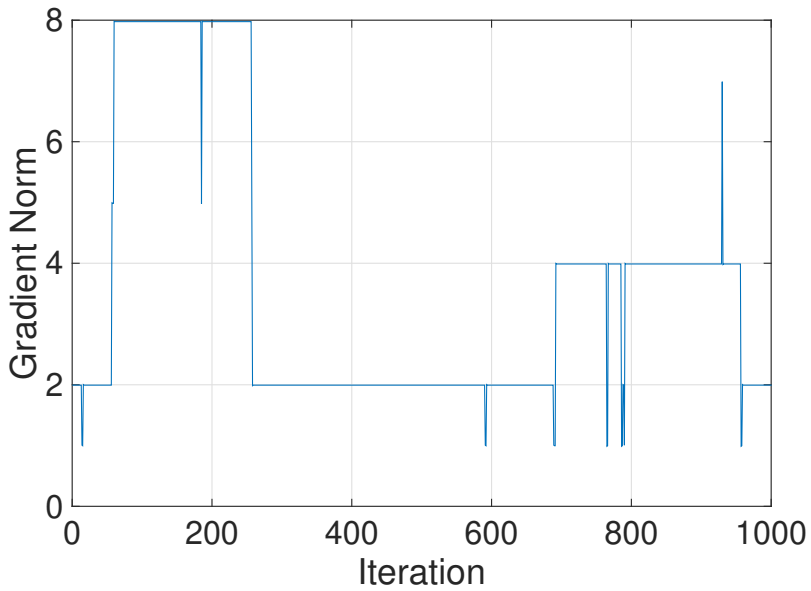
1. Will DGD with straightforward compressed info still converge?
2. If answer to (1) is "No," could we fix and make it work w/ compressed info?
3. If answer to (2) Is "Yes," how fast does the new algorithm converge?

DGD w/ Straightforward Compressed Info Fails

- The node sends compressed message $C(x_i)$ to its neighbors.



(a) $\alpha = 0.001$.



(b) $\alpha = 0.001/\sqrt{k}$.

Figure 1: Simulation results for DGD with quantized compression operator for a 2-node network, for which DGD fails to converge after 1000 iterations.

DGD w/ Straightforward Compressed Info Fails

$$
\begin{aligned}
x_{i,k+1} &= \sum\nolimits_{j\in\mathcal{N}_i} [\mathbf{W}]_{ij} C(x_{j,k}) - \alpha_k \nabla f_i(x_{i,k}) \\
&= \underbrace{\sum\nolimits_{j=\mathcal{N}_i} [\mathbf{W}]_{ij} x_{j,k} - \alpha_k \nabla f_i(x_{i,k})}_{\text{Exact DGD}} + \underbrace{\sum\nolimits_{j\in\mathcal{N}_i} [\mathbf{W}]_{ij} \epsilon_{x_{j,k}}}_{\text{Accumulated noise term}}
\end{aligned}
$$

- The accumulated noise term is similar to the noise term in SGD;
- In SGD, diminishing step-size/variance reduction to guarantee convergence;
- Variance reduction strategy might be helpful.

Open Fundamental Questions

- Will DGD with straightforward compressed info still converge?
- If answer to (1) is "No," could we fix and make it work w/ compressed info?
- If answer to (2) Is "Yes," how fast does the new algorithm converge?

Open Fundamental Questions

- Will DGD with straightforward compressed info still converge?
- If answer to (1) is "No," could we fix and make it work w/ compressed info?
- If answer to (2) Is "Yes," how fast does the new algorithm converge?

Our Solution: Amplified-Differential Compression DGD

Amplified-Differential Compression DGD (ADC-DGD)

The ADC-DGD algorithm:

1. Initialization: Let $k = 1$. Let $x_{i,0} = \tilde{x}_{i,0} = 0, \forall i$. Choose initial step-size α_1 and an amplification exponent γ. Let $x_{i,1} = y_{i,1} = -\alpha_1 \nabla f_i(x_{i,0})$.

Amplified-Differential Compression DGD (ADC-DGD)

The ADC-DGD algorithm:

1. Initialization: Let $k = 1$. Let $x_{i,0} = \tilde{x}_{i,0} = 0, \forall i$. Choose initial step-size α_1 and an amplification exponent γ. Let $x_{i,1} = y_{i,1} = -\alpha_1 \nabla f_i(x_{i,0})$.
2. In k-th iteration: Each node sends compressed amplified-differential $d_{i,k} = C(k^\gamma y_{i,k})$ to its neighbors, where k^γ is the amplification factor, local differential $y_{i,k} = x_{i,k} - \tilde{x}_{i,k-1}$ and $C(\cdot)$ is some unbiased stochastic compression operator.

Amplified-Differential Compression DGD (ADC-DGD)

The ADC-DGD algorithm:

1. Initialization: Let $k = 1$. Let $x_{i,0} = \tilde{x}_{i,0} = 0, \forall i$. Choose initial step-size α_1 and an amplification exponent γ. Let $x_{i,1} = y_{i,1} = -\alpha_1 \nabla f_i(x_{i,0})$.
2. In k-th iteration: Each node sends compressed amplified-differential $d_{i,k} = C(k^\gamma y_{i,k})$ to its neighbors, where k^γ is the amplification factor, local differential $y_{i,k} = x_{i,k} - \tilde{x}_{i,k-1}$ and $C(\cdot)$ is some unbiased stochastic compression operator.
3. Upon reception of all local copies from its neighbors, each node estimates neighbors' values: $\tilde{x}_{j,k} = \tilde{x}_{j,k-1} + d_{j,k}/k^\gamma$. Then, each node updates its local value:

$$x_{i,k+1} = \underbrace{\sum_{j \in \mathcal{N}_i} [\mathbf{W}]_{ij} \tilde{x}_{j,k}}_{\text{Compressed consensus}} - \underbrace{\alpha_k \nabla f_i(x_{i,k})}_{\text{gradient step}}.$$

Each node updates local differential: $y_{i,k+1} = x_{i,k+1} - \tilde{x}_{i,k}$.

Intuition Behind ADC-DGD

- A closer look at ADC-DGD update reveals that:

$$\begin{aligned}\tilde{x}_{j,k} =& \tilde{x}_{j,k-1} + C(k^\gamma y_{j,k})/k^\gamma \\ =& \tilde{x}_{j,k-1} + (k^\gamma y_{j,k} + \epsilon_{k^\gamma y_{j,k}})/k^\gamma \\ =& \tilde{x}_{j,k-1} + y_{j,k} + (\epsilon_{k^\gamma y_{j,k}})/k^\gamma \\ =& x_{j,k} + \underbrace{(\epsilon_{k^\gamma y_{j,k}})/k^\gamma}_{\text{Vanishing noise term}} \quad .\end{aligned}$$

- Thanks to properties of unbiased stochastic operator, the noise term in the last step of has zero mean and a vanishing variance $\frac{\sigma^2}{k^{2\gamma}}$ as $k \to \infty$.

Intuition Behind ADC-DGD

- A closer look at ADC-DGD update reveals that:

$$\begin{aligned}\tilde{x}_{j,k} =& \tilde{x}_{j,k-1} + C(k^\gamma y_{j,k})/k^\gamma \\ =& \tilde{x}_{j,k-1} + (k^\gamma y_{j,k} + \epsilon_{k^\gamma y_{j,k}})/k^\gamma \\ =& \tilde{x}_{j,k-1} + y_{j,k} + (\epsilon_{k^\gamma y_{j,k}})/k^\gamma \\ =& x_{j,k} + \underbrace{(\epsilon_{k^\gamma y_{j,k}})/k^\gamma}_{\text{Vanishing noise term}} \quad .\end{aligned}$$

- Thanks to properties of unbiased stochastic operator, the noise term in the last step of has zero mean and a vanishing variance $\frac{\sigma^2}{k^{2\gamma}}$ as $k \to \infty$.

Indeed, proofs for main theorems are based on this intuition

Key Features of ADC-DGD

- Compared to DGD, requires additional memory to store (imprecise) neighbors' values: $\{\tilde{x}_{j,k-1} : (i,j) \in \mathcal{L}\}$
  - Allows to transmit only the difference between successive iterations $y_{i,k} = x_{i,k} - \tilde{x}_{i,k-1}$ rather than $x_{i,k}$ directly
  - Memory requirement is modest since many networks are scale-free (i.e., node degree distribution follows a power law)

Key Features of ADC-DGD

- Compared to DGD, requires additional memory to store (imprecise) neighbors' values: $\{\tilde{x}_{j,k-1} : (i,j) \in \mathcal{L}\}$
  - Allows to transmit only the difference between successive iterations $y_{i,k} = x_{i,k} - \tilde{x}_{i,k-1}$ rather than $x_{i,k}$ directly
  - Memory requirement is modest since many networks are scale-free (i.e., node degree distribution follows a power law)
- Each node sends out a compressed amplified-differential $C(k^\gamma y_{i,k})$, which is more communication-efficient compared to DGD
  - This information will be de-amplified at receiving nodes as $d_{i,k}/k^\gamma$, which results in a noisy version of $y_{i,k}$
  - Based on memory of the previous iteration, each node obtains their neighbors' values estimation $\tilde{x}_{j,k}$, $j \in \mathcal{N}_i$

Key Features of ADC-DGD

- Compared to DGD, requires additional memory to store (imprecise) neighbors' values: $\{\tilde{x}_{j,k-1} : (i,j) \in \mathcal{L}\}$
  - Allows to transmit only the difference between successive iterations $y_{i,k} = x_{i,k} - \tilde{x}_{i,k-1}$ rather than $x_{i,k}$ directly
  - Memory requirement is modest since many networks are scale-free (i.e., node degree distribution follows a power law)
- Each node sends out a compressed amplified-differential $C(k^\gamma y_{i,k})$, which is more communication-efficient compared to DGD
  - This information will be de-amplified at receiving nodes as $d_{i,k}/k^\gamma$, which results in a noisy version of $y_{i,k}$
  - Based on memory of the previous iteration, each node obtains their neighbors' values estimation $\tilde{x}_{j,k}$, $j \in \mathcal{N}_i$
- With $\tilde{x}_{j,k}$, $j \in \mathcal{N}_i$, the update follows the same structure as in DGD
  - Also contains a consensus step and a local gradient step.
  - Complexity of ADC-DGD is almost the same to the original DGD

Open Fundamental Questions

- Will DGD with straightforward compressed info still converge?
- If answer to (1) is "No," could we fix and make it work w/ compressed info?
- If answer to (2) Is "Yes," how fast does the new algorithm converge?

Our Solution: Amplified-Differential Compression DGD

Main Theoretical Results

Assumption 1

The local objective functions $f_i(\cdot)$ satisfy:

- *(Lower boundedness) $\exists$ an optimal x^* with $\|x^*\| < \infty$ such that $\forall x \neq x^*$, $\sum_{i=1}^N f_i(x) \geq \sum_{i=1}^N f_i(x^*)$*
- *(Lipschitz continuous gradient) there exists a constant $L > 0$ such that $\forall x, y$, $\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|$, $\forall i$.*

Main Theoretical Results

Assumption 1

The local objective functions $f_i(\cdot)$ satisfy:

- *(Lower boundedness) $\exists$ an optimal x^* with $\|x^*\| < \infty$ such that $\forall x \neq x^*$, $\sum_{i=1}^N f_i(x) \geq \sum_{i=1}^N f_i(x^*)$*
- *(Lipschitz continuous gradient) there exists a constant $L > 0$ such that $\forall x, y$, $\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|$, $\forall i$.*

Assumption 2 (Growth Rate at Infinity)

$\exists$ a constant $M > 0$ such that:

$$\lim_{\|\mathbf{x}\| \to \infty} \frac{\|\mathbf{x}\|}{f(\mathbf{x})} = \lim_{\sum_{i=1}^n \|x_i\| \to \infty} \frac{\sum_{i=1}^N \|x_i\|}{\sum_{i=1}^N f_i(x_i)} \leq M.$$

Main Theoretical Results

Lemma 2

Any strictly convex function $f(\cdot)$ satisfying Assump. 1 also satisfies Assump. 2.

Main Theoretical Results

Lemma 2

Any strictly convex function $f(\cdot)$ satisfying Assump. 1 also satisfies Assump. 2.

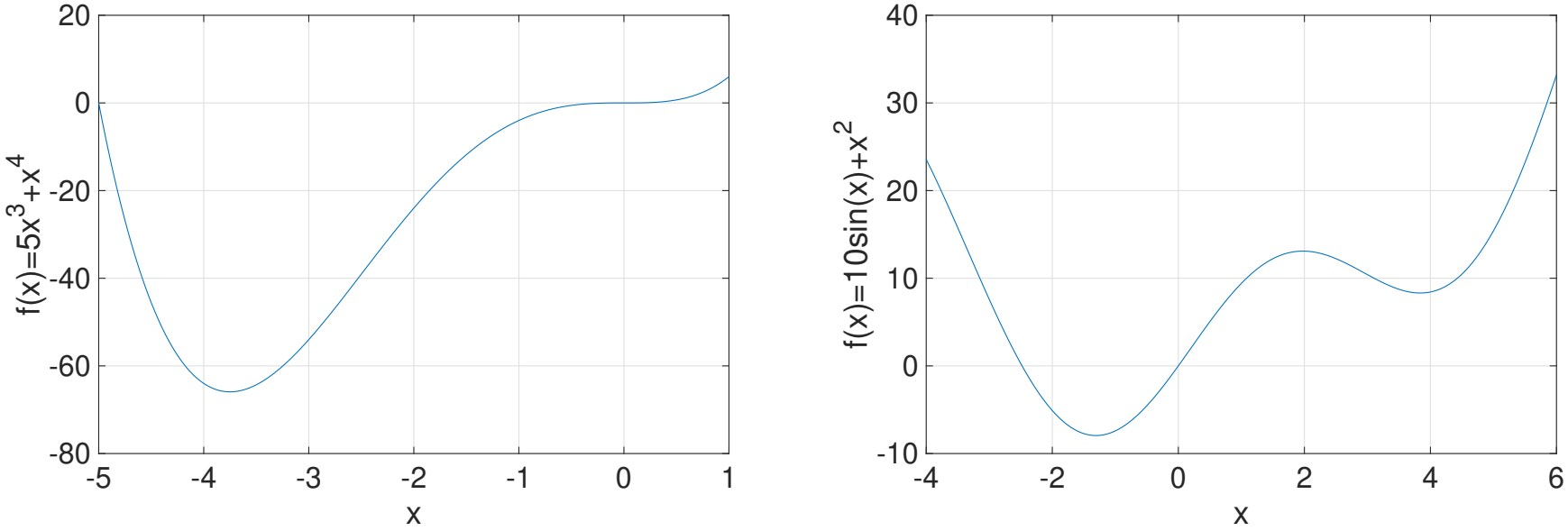


Figure 2: Examples of non-convex functions satisfying Assumption 2.

Main Theoretical Results

Theorem 3 (Convergence to Mean Vector)

Let the mean vector at the k-th iteration be $\bar{\mathbf{x}}^k = \mathbf{1} \otimes \bar{x}^k \in \mathbb{R}^{NP}$, *where* $\bar{x}^k = \frac{1}{N}\sum_{i=1}^{N} x_{i,k}$. *Under Assumptions 1, if* $\mathbb{E}[\|\nabla \mathbf{f}(\mathbf{x}^i)\|] \leq D$ *and the amplifying exponent is* γ, *then:*

- *For constant step-size* $\alpha_i = \alpha, \forall i$, $\mathbb{E}[\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|] \leq \frac{\alpha D}{1-\beta} + O(\frac{\sqrt{NP}\sigma}{k^\gamma})$;
- *For diminishing step-size* $\alpha_i = O(\frac{1}{k^\eta})$ *with some constant* $\eta \in (0, 1]$, $\mathbb{E}[\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|] = O(\frac{1}{k^{\min(\eta,\gamma)}})$.

Main Theoretical Results

Theorem 3 (Convergence to Mean Vector)

Let the mean vector at the k-th iteration be $\bar{\mathbf{x}}^k = \mathbf{1} \otimes \bar{x}^k \in \mathbb{R}^{NP}$, *where* $\bar{x}^k = \frac{1}{N}\sum_{i=1}^{N} x_{i,k}$. *Under Assumptions 1, if* $\mathbb{E}[\|\nabla \mathbf{f}(\mathbf{x}^i)\|] \leq D$ *and the amplifying exponent is* γ, *then:*

- *For constant step-size* $\alpha_i = \alpha, \forall i$, $\mathbb{E}[\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|] \leq \frac{\alpha D}{1-\beta} + O(\frac{\sqrt{NP}\sigma}{k^\gamma})$;
- *For diminishing step-size* $\alpha_i = O(\frac{1}{k^\eta})$ *with some constant* $\eta \in (0, 1]$, $\mathbb{E}[\|\mathbf{x}^k - \bar{\mathbf{x}}^k\|] = O(\frac{1}{k^{\min(\eta,\gamma)}})$.

Remark

- *Local copies converge to mean vector with a diminishing step-size*
- *Local copies stay within a bounded error ball of the mean vector with a constant step-size.*

Main Theoretical Results

Theorem 4 (Constant Step-Size)

With constant step-size $\alpha_k = \alpha < \frac{1+\lambda_N(\mathbf{W})}{L}$, $\forall k$, *Under Assump. 1–2, if the amplified exponent* $\gamma > \frac{1}{2}$, *then it holds that:*

$$\min_k \left\{ \mathbb{E}\left[\left\| \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(\bar{x}^k) \right\|^2 \right] \right\} \leq \frac{C_1 \alpha^2}{N(1-\beta)^2} + \frac{C_2}{\alpha k} + O\left(\frac{2L\sqrt{P}\sigma}{\sqrt{N} k^{\gamma}} \right),$$

where $C_1 \triangleq L^4 (B + \|\mathbf{x}^*\|)^2$ *and* $C_2 \triangleq 2[\frac{1}{N}\sum_{i=1}^{N} f_i(0) - \frac{1}{N}\sum_{i=1}^{N} f_i(x^*)]$.

Main Theoretical Results

Theorem 4 (Constant Step-Size)

With constant step-size $\alpha_k = \alpha < \frac{1+\lambda_N(\mathbf{W})}{L}$, $\forall k$, *Under Assump. 1–2, if the amplified exponent* $\gamma > \frac{1}{2}$, *then it holds that:*

$$\min_k \left\{ \mathbb{E}\left[\left\| \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(\bar{x}^k) \right\|^2 \right] \right\} \leq \frac{C_1 \alpha^2}{N(1-\beta)^2} + \frac{C_2}{\alpha k} + O\left(\frac{2L\sqrt{P}\sigma}{\sqrt{N} k^{\gamma}} \right),$$

where $C_1 \triangleq L^4 (B + \|\mathbf{x}^*\|)^2$ *and* $C_2 \triangleq 2[\frac{1}{N}\sum_{i=1}^{N} f_i(0) - \frac{1}{N}\sum_{i=1}^{N} f_i(x^*)]$.

Remark

Under the conditions of Theorem 4, ADC-DGD has:

- *A convergence rate* $O\left(\frac{1}{k^{\min(1,\gamma)}}\right)$ *until reaching an error ball* $O\left(\left(\frac{\alpha}{1-\beta}\right)^2\right)$
- *The fastest convergence rate of ADC-DGD is* $O\left(\frac{1}{k}\right)$. *(Same as GD)*

Main Theoretical Results

Theorem 5 (Diminishing Step-Sizes)

Under Assump. 1–2, if $\exists$ *a constant* $D > 0$ *such that* $\|\nabla f_i(x)\| \leq D$, $\forall x$, *and* $\gamma > \frac{1}{2}$, $\eta \geq \frac{1}{2}$, *then with diminishing step-size* $\alpha_k = O(\frac{1}{k^\eta})$, *it holds that*

$$\left\| \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(\bar{x}^k) \right\|^2 = o\left(\frac{1}{k^{1-\eta}} \right) \text{ almost surely.}$$

Main Theoretical Results

Theorem 5 (Diminishing Step-Sizes)

Under Assump. 1–2, if $\exists$ *a constant* $D > 0$ *such that* $\|\nabla f_i(x)\| \leq D$, $\forall x$, *and* $\gamma > \frac{1}{2}$, $\eta \geq \frac{1}{2}$, *then with diminishing step-size* $\alpha_k = O(\frac{1}{k^\eta})$, *it holds that*

$$\left\| \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(\bar{x}^k) \right\|^2 = o\left(\frac{1}{k^{1-\eta}} \right) \text{ almost surely.}$$

Remark

- *The exponent* $\eta \geq \frac{1}{2}$. *Thus, the best convergence rate for ADC-DGD is* $o(1/\sqrt{k})$, *faster than all existing first-order approaches.*
- *Convergence is in "Small-O", stronger than conventional "Big-O" results.*

Understanding the Role of the Amplifying Exponent γ

- The amplifying exponent γ is a key component to adjust the communication rate.
  - within $(1/2, 1]$, the larger γ means the faster convergence;
  - a larger γ leads to a larger $k^\gamma \mathbf{y}^k$, which may lead to overflow error for the transmitted value $C(k^\gamma \mathbf{y}^k)$
  - for example, type 'int8' in Matlab could only present data within $(-128, 127)$.

Understanding the Role of the Amplifying Exponent γ

- The amplifying exponent γ is a key component to adjust the communication rate.
  - within $(1/2, 1]$, the larger γ means the faster convergence;
  - a larger γ leads to a larger $k^\gamma \mathbf{y}^k$, which may lead to overflow error for the transmitted value $C(k^\gamma \mathbf{y}^k)$
  - for example, type 'int8' in Matlab could only present data within $(-128, 127)$.

Proposition 1

Under Assumptions 1-2, with $\gamma > \frac{1}{2}$, the transmitted value $k^\gamma \mathbf{y}^k$ satisfies $\mathbb{E}[\|k^\gamma \mathbf{y}^k\|] = o(k^{\gamma - \frac{1}{2}})$.

Understanding the Role of the Amplifying Exponent γ

- The amplifying exponent γ is a key component to adjust the communication rate.
  - within $(1/2, 1]$, the larger γ means the faster convergence;
  - a larger γ leads to a larger $k^\gamma \mathbf{y}^k$, which may lead to overflow error for the transmitted value $C(k^\gamma \mathbf{y}^k)$
  - for example, type 'int8' in Matlab could only present data within $(-128, 127)$.

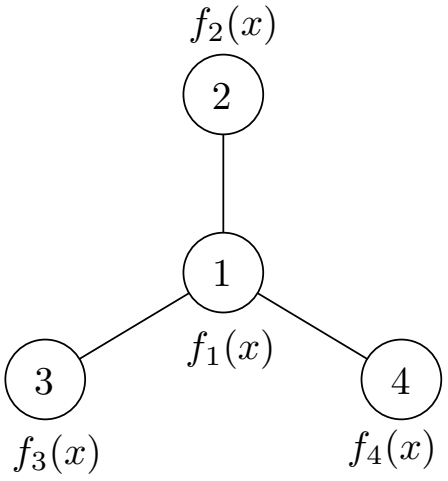
Proposition 1

Under Assumptions 1-2, with $\gamma > \frac{1}{2}$, the transmitted value $k^\gamma \mathbf{y}^k$ satisfies $\mathbb{E}[\|k^\gamma \mathbf{y}^k\|] = o(k^{\gamma - \frac{1}{2}})$.

- with $\gamma \in (\frac{1}{2}, 1]$, the growth speed for the transmitted value is slower than $o(\sqrt{k})$, which is not very fast.

Numerical Results

- Effect of Compression:



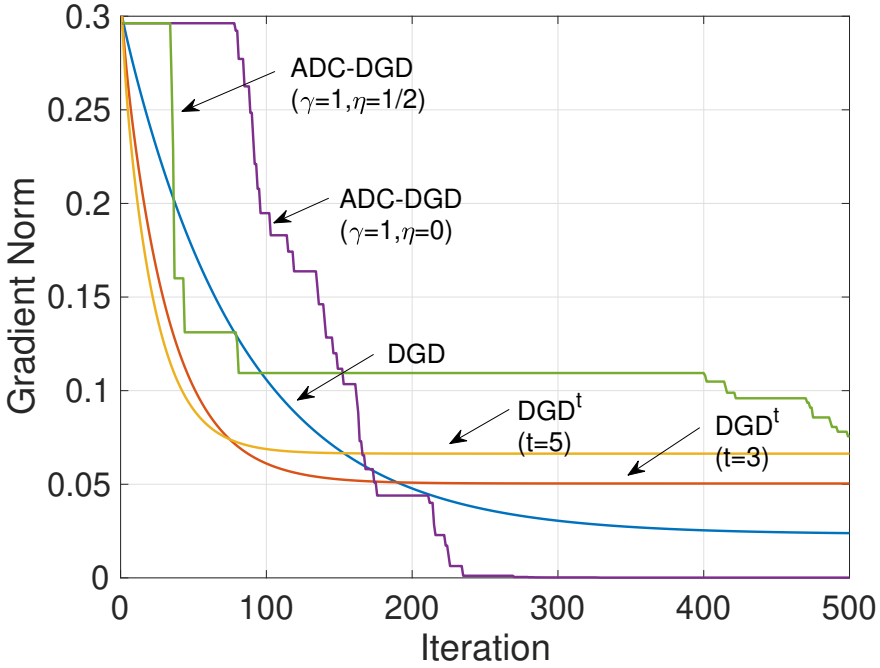
A four-node star network.

$$\mathbf{W} = \begin{bmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 3/4 & 0 & 0 \\ 1/4 & 0 & 3/4 & 0 \\ 1/4 & 0 & 0 & 3/4 \end{bmatrix}$$

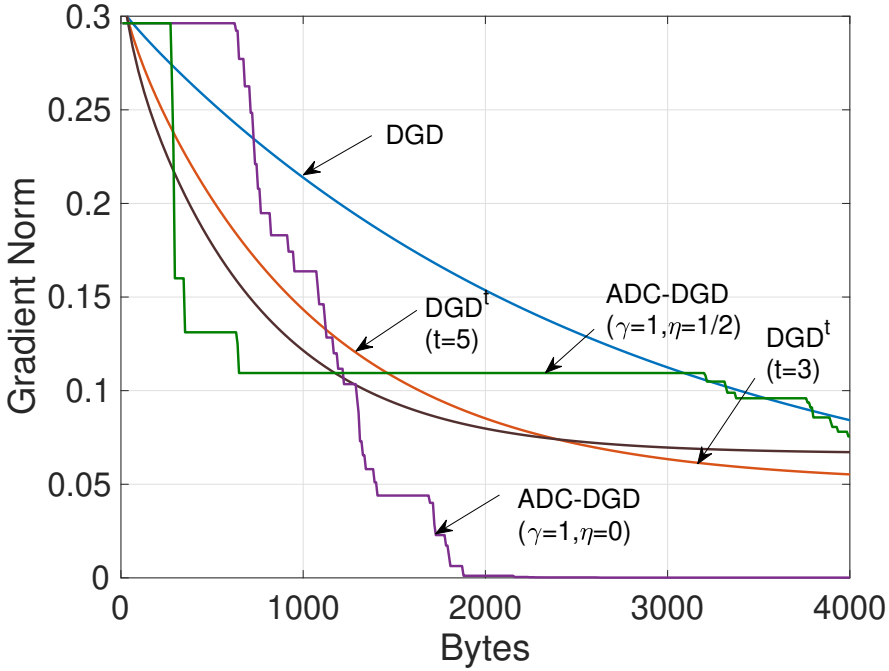
The consensus matrix.

- $\min_x f(x) = f_1(x) + f_2(x) + f_3(x) + f_4(x)$, with $f_1(x) = -4x^2$, $f_2(x) = 2(x-0.2)^2$, $f_3(x) = 2(x+0.3)^2$, and $f_4(x) = 5(x-0.1)^2$.

Numerical Results



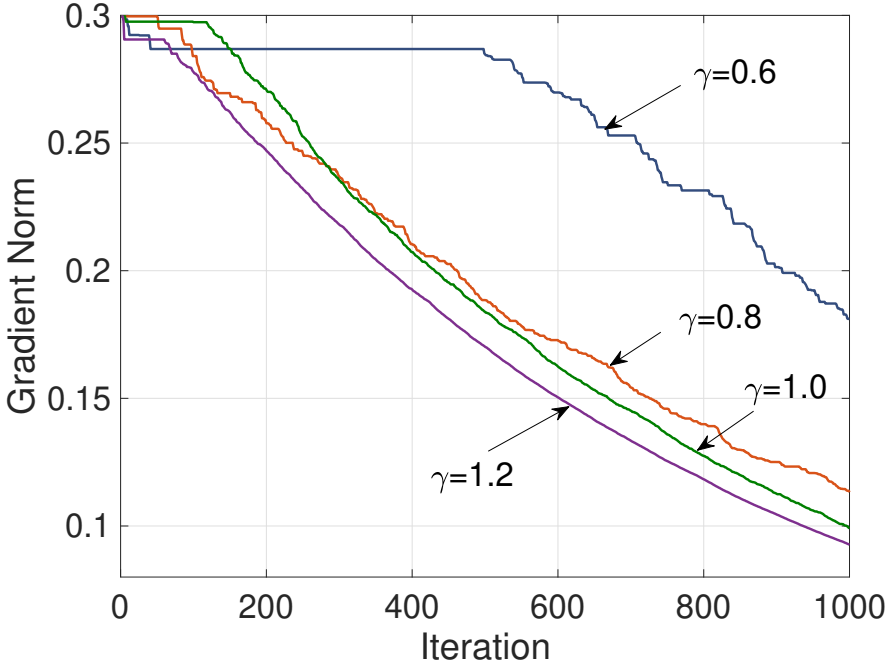
Convergence comparisons between ADC-DGD, DGD, and DGD^t.



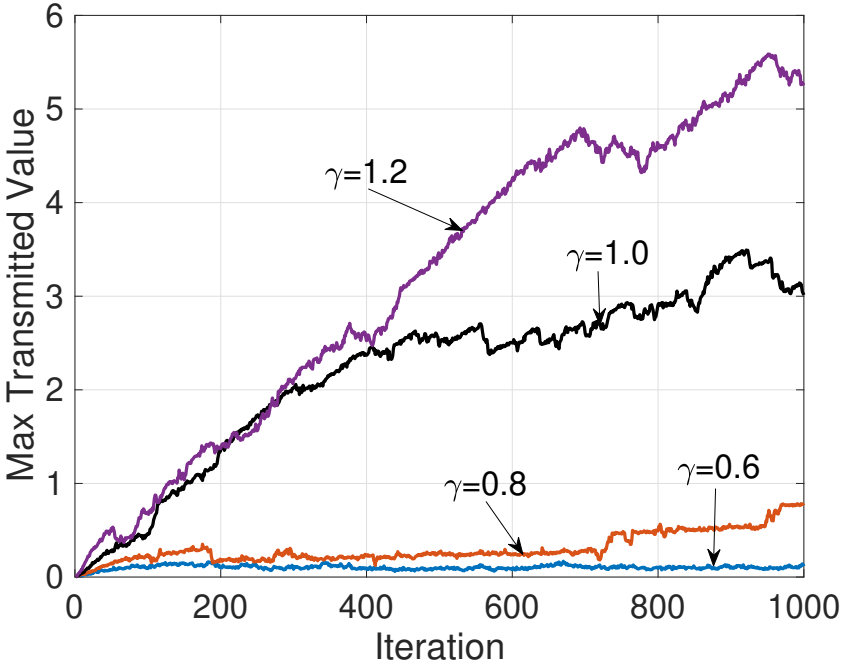
Amount of exchanged information (bytes) vs gradient norm.

Numerical Results

- Effect of Amplifying Exponent: Choose $\gamma \in \{0.6, 0.8, 1.0, 1.2\}$



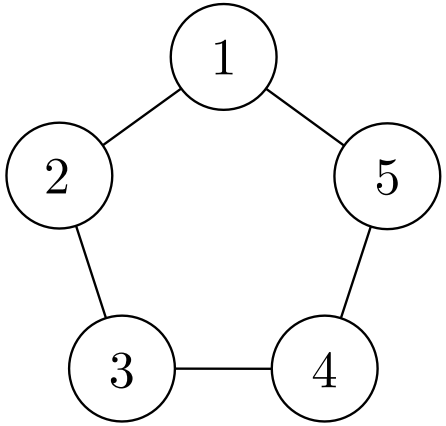
Convergence behaviors under different γ's.



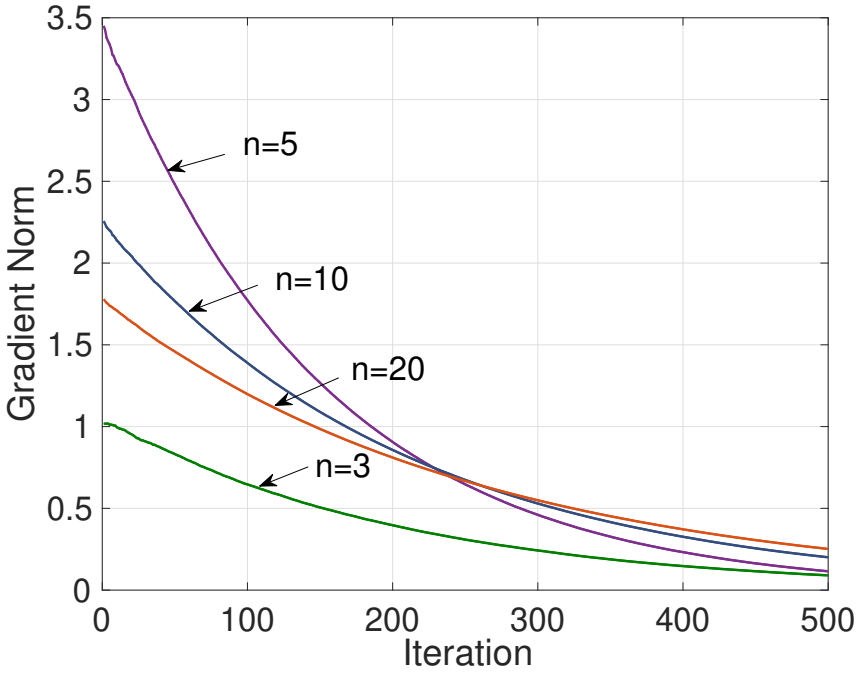
Growth of transmitted values ($k^{\gamma y}$) vs. number of iterations.

Numerical Results

- Effect of Network Size: Choose $N \in \{3, 5, 10, 20\}$



The 5-node circle topology.



The effect of network size.

Conclusion

Developed a new algorithm called ADC-DGD based on compression to reduce communication costs:

1. Convergence results with constant and diminished step-sizes
2. Understanding of compression in network consensus optimization

Conclusion

Developed a new algorithm called ADC-DGD based on compression to reduce communication costs:

1. Convergence results with constant and diminished step-sizes
2. Understanding of compression in network consensus optimization

Future work:

- We claim that the memory requirement is modest and store the additional values, but the network are sensors with limited memory;
- The large values of the amplifying exponent γ may lead to overflow errors, and a solution for setting γ in a distributed manner is necessary;
- This work is mainly based on the quantized compression operator, which might not be the most efficient. Designing more efficient unbiased stochastic compression operator would be helpful.

Thank You!