Unified and Refined Analysis of Decentralized Optimization and Learning Algorithms

Sulaiman A. Alghunaim[•]

[•] Kuwait University, Kuwait

August 2023

Outline

Introduction and Review

Unified Decentralized Algorithm (UDA)

Stochastic Unified Decentralized Algorithm (SUDA)

Local Exact-Diffusion

Distributed (consensus) optimization

Network of N nodes (agents, workers, clients) collaborate to solve:

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad \left\{ f(x) \triangleq \frac{1}{N} \sum_{i=1}^{N} f_i(x) \right\} \tag{1}$$

- $f_i : \mathbb{R}^m \to \mathbb{R}$ is a private local function known by node i, which may be stochastic $f_i(x) = \mathbb{E}[F_i(x; \xi_i)]$, where $\{\xi_i\}$ denotes data available at node i
- The goal of each node i is to find a minimizer (solution) of problem (1), denoted by $x^\star$, through local interactions with other nodes without sharing private information (*e.g.*, data ξ_i)

Applications examples

Large scale machine learning: (classification or regression problem)

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad \sum_{j=1}^{m} l_j(x; \xi_j)$$

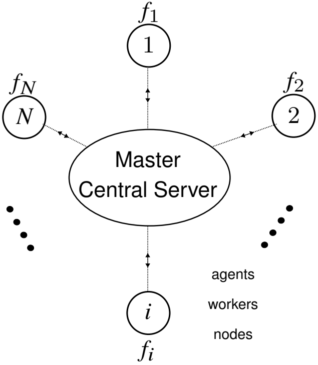
- For large m, storing the data on a single machine may not be feasible
- The problem can be solved by distributing the data across multiple workers, $f_i(x) = \sum_{j \in \mathcal{J}_i} l_j(x; \xi_j)$ where $\mathcal{J}_i$ is the set of training data indices at worker i

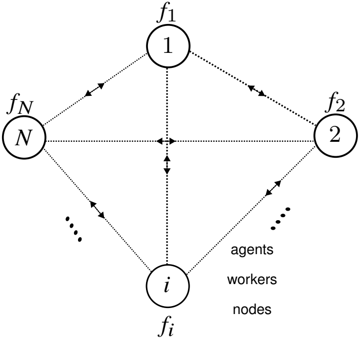
Distributed ML estimation:

$$\underset{x \in \mathbb{R}^m}{\text{minimize}} \quad -\sum_{i=1}^{N} \log p\left(\zeta_i \mid x\right)$$

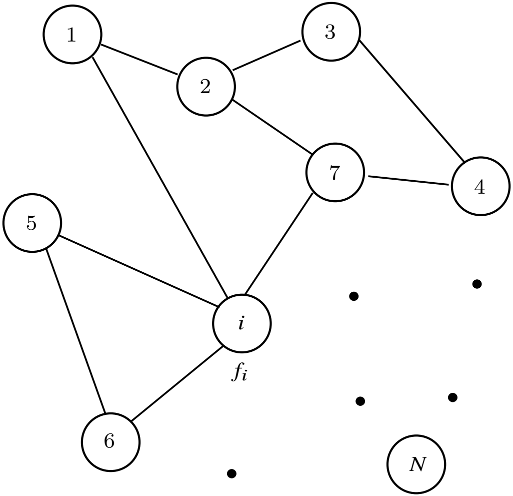
- N sensors, sensor i has a measurement ζ_i (*e.g.*, detecting events such as avalanches)
- ζ_i is modeled as a random variable with density $p\left(\zeta_i \mid x\right)$

Network model: Centralized

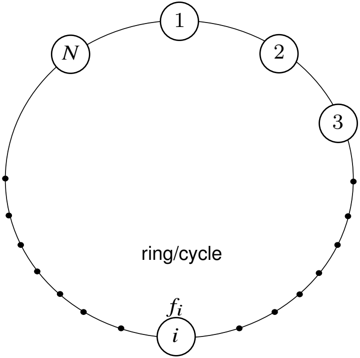


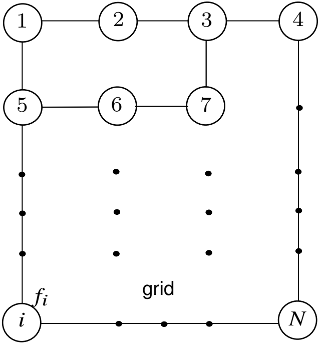


Network model: Decentralized



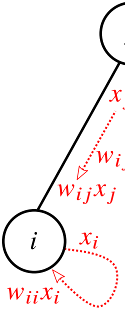
Examples

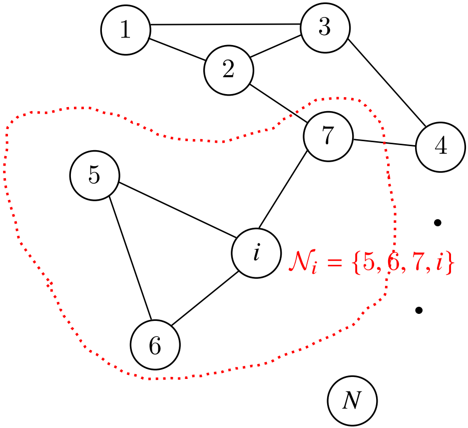




Network modeling

- w_{ij}: **weight** used by node i to scale node j information
- $\mathcal{N}_i$: neighborhood of node i





Weight matrix

$$W \triangleq \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ldots & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{NN} \end{bmatrix}$$

- We assume that the network is undirected and connected
- We can construct a primitive doubly stochastic matrix W
- W has a single eigenvalue at one, denoted by $\lambda_1 = 1$, and all other eigenvalues $\{\lambda_i\}_{i=2}^{N}$ are strictly less than one in magnitude
- The **mixing rate** of the network is:

$$\lambda \triangleq \|W - \frac{1}{N}\mathbf{1}\mathbf{1}^T\| = \max_{i \in \{2,\ldots,N\}} |\lambda_i| < 1$$

Consensus algorithm

Basic consensus problem

- Let a_i be a point known only to node i
- Find the average $\bar{a} = \frac{1}{N}\sum_{i=1}^{N} a_i$ using only decentralized communication

Consensus algorithm: Initialize $x_i^0 = a_i$ and update for each i:

$$x_i^{k+1} = \sum_{j \in \mathcal{N}_i} w_{ij} x_j^k, \quad k = 0, 1, \ldots$$

When $W = [w_{ij}]$ is doubly stochastic and primitive, then the above converges to the average: $x_i^k \to \bar{a}$ for all $i = 1, \ldots, N$

Consensus algorithm: Network notation

Network notation:

- $W \triangleq [w_{ij}], \quad \mathbf{W} \triangleq W \otimes I_m$
- $\mathbf{x}^k \triangleq \mathrm{col}\{x_1^k, x_2^k, \ldots, x_N^k\}$
- The above allows $\mathbf{W}\mathbf{x}^k = \{\sum_{\mathcal{N}_i} w_{ij} x_j^k\}_{i=1}^N$

Consensus algorithm: Initialize $\mathbf{x}^0 = \mathrm{col}\{a_i\}_{i=1}^N$ and update

$$\mathbf{x}^{k+1} = \mathbf{W}\mathbf{x}^k = \mathbf{W}^{k+1}\mathbf{x}^0, \quad k = 0, 1, 2, \ldots$$

When W is doubly stochastic and primitive, then $\mathbf{W}^k \rightarrow \frac{1}{N}\mathbf{1}\mathbf{1}^T \otimes I_m$, hence

$$\mathbf{x}^{k+1} \rightarrow \frac{1}{N}\mathbf{1}\mathbf{1}^T\mathbf{x}^0 = \mathrm{col}\left\{\frac{1}{N}\sum_{j=1}^N a_j, \ldots, \frac{1}{N}\sum_{j=1}^N a_j\right\}$$

Decentralized gradient descent

Problem: $\underset{x\in\mathbb{R}^m}{\text{minimize}} \ \frac{1}{N}\sum_{i=1}^{N} f_i(x)$ over a network

Decentralized/distributed gradient descent (DGD)

- *Consensus form*

$$x_i^{k+1} = \sum_{j\in\mathcal{N}_i} w_{ij} x_j^k - \alpha \nabla f_i(x_i^k) \tag{2}$$

- *Diffusion form (adapt-then-combine)*

$$x_i^{k+1} = \sum_{j\in\mathcal{N}_i} w_{ij} \left(x_j^k - \alpha \nabla f_j(x_j^k)\right) \tag{3}$$

DGD in Network form

Network notation:

- $W \triangleq [w_{ij}], \quad \mathbf{W} \triangleq W \otimes I_m$
- $\mathbf{x}^k \triangleq \mathrm{col}\{x_1^k, x_2^k, \ldots, x_N^k\}$
- $\mathbf{f}(\mathbf{x}) \triangleq \sum_{i=1}^N f_i(x_i)$

DGD: Network notation

- *Consensus form*

$$\mathbf{x}^{k+1} = \mathbf{W}\mathbf{x}^k - \alpha\nabla\mathbf{f}(\mathbf{x}^k) \tag{4}$$

- *Diffusion form (adapt-then-combine)*

$$\mathbf{x}^{k+1} = \mathbf{W}\left(\mathbf{x}^k - \alpha\nabla\mathbf{f}(\mathbf{x}^k)\right) \tag{5}$$

DSGD convergence

- **Stochastic formulation:** minimize $\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[F_i(x;\xi_i)]$
- **DSGD (diffusion):** $x_i^{k+1} = \sum_{j\in\mathcal{N}_i} w_{ij}\left(x_j^k - \alpha\nabla F_j(x_j^k;\xi_j^k)\right)$

Assumptions:

1. W is primitive and doubly stochastic
2. Each function f_i is L-smooth: $\|\nabla f_i(x) - \nabla f_i(y)\| \le L\|x-y\|$
3. The aggregate function $f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x)$ is bounded below: $f(x) \ge f^\star > -\infty\ \forall\, x \in \mathbb{R}^m$ where $f^\star$ denote the optimal value
4. For all $\{i\}_{i=1}^{N}$ and $k = 0, 1, \ldots,$

$$\mathbb{E}_{\xi_i}\left[\nabla F_i(x_i;\xi_i) - \nabla f_i(x_i)\right] = 0$$
$$\mathbb{E}_{\xi_i}\left[\|\nabla F_i(x_i;\xi_i) - \nabla f_i(x_i)\|^2\right] \le \sigma^2,$$

   for some $\sigma^2 \ge 0$, and the random data $\{\xi_i^t\}$ are independent of each other for all $\{i\}_{i=1}^{N}$ and $\{t\}_{t\le k}$

Convergence: Strongly convex case

Convergence: If each f_i is μ-strongly-convex, then for sufficiently small fixed stepsize $\alpha \leq \mathcal{O}(\frac{1-\lambda}{L})$, it holds that:

$$\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}\|x_i^k - x^\star\|^2 \leq (1-\mu\alpha)^k C_0 + \mathcal{O}\left(\frac{\alpha\sigma^2}{N} + \frac{\alpha^2\lambda^2\sigma^2}{1-\lambda} + \frac{\alpha^2\lambda^2 b^2}{(1-\lambda)^2}\right)$$

- C_0 is constant depending on initialization
- $b^2 \triangleq (1/N)\sum_{i=1}^{N} \|\nabla f_i(x^\star)\|^2$

Bias: For noiseless case $\sigma = 0$

$$\frac{1}{N}\sum_{i=1}^{N} \|x_i^k - x^\star\|^2 \to \mathcal{O}\left(\frac{\alpha^2\lambda^2 b^2}{(1-\lambda)^2}\right)$$

It does not have exact convergence for fixed stepsize

Convergence rate: There exists a stepsize $\alpha \leq \mathcal{O}(1/K)$ such that

$$\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\|x_i^K - x^\star\|^2 \leq \tilde{\mathcal{O}}\Bigg(\frac{\sigma^2}{NK} + \frac{\lambda^2\sigma^2}{(1-\lambda)^2K^2} + \frac{\lambda^2 b^2}{(1-\lambda)^2K^2} + \frac{C_0}{(1-\lambda)}\exp\left[-K(1-\lambda)\right]\Bigg)$$

- For large K, the term $\mathcal{O}(\frac{\sigma^2}{NK})$ dominates
- DSGD asymptotically achieve the minibatch SGD rate with batch size N
- the number of iterations needed to achieve this rate (linear speedup) is called the **transient time**:

$$K \geq \mathcal{O}\left(\frac{N}{(1-\lambda)^2}\right)$$

Convergence: Non-convex case

Convergence: Let each function f_i be L-smooth, then for sufficiently small fixed stepsize $\alpha \leq \mathcal{O}(\frac{1-\lambda}{L})$, the average $\bar{x}^k = (1/N)\sum_{i=1}^N x_i^k$ satisfies

$$\frac{1}{K}\sum_{k=0}^{K-1} \mathbb{E}\,\|\nabla f(\bar{x}^k)\|^2 \quad \leq \frac{f(0) - f^\star}{\alpha K} + \frac{\alpha\sigma^2}{N} + \frac{\alpha^2\lambda^2 N\sigma^2}{(1-\lambda)} + \frac{\alpha^2\lambda^2 N\varsigma^2}{(1-\lambda)^2}$$

- **Bias:** For noiseless case $\sigma = 0$

$$\frac{1}{N}\sum_{i=1}^{N} \|x_i^k - x^\star\|^2 \to \mathcal{O}\left(\frac{\alpha^2\lambda^2 N\varsigma^2}{(1-\lambda)^2}\right)$$

  It does not have exact convergence for fixed stepsize

- ς^2 measures the functions heterogeneity across the network

$$\frac{1}{N}\sum_{i=1}^{N} \|\nabla f_i(x) - \nabla f(x)\|_2^2 \leq \varsigma^2, \quad \forall\, x \in \mathbb{R}^m$$

Convergence rate: There exists a stepsize $\alpha \le \mathcal{O}(1/\sqrt{K})$ such that

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\,\|\nabla f(\bar{x}^k)\|^2 \le O\Bigg(\frac{\sigma}{\sqrt{NK}} + \frac{\lambda^{2/3}\sigma^{2/3}}{(1-\lambda)^{1/3}K^{2/3}} + \frac{\lambda^{2/3}\varsigma^{2/3}}{(1-\lambda)^{2/3}K^{2/3}} + \frac{1}{(1-\lambda)K}\Bigg)$$

- For large K, the term $\mathcal{O}(\frac{\sigma}{\sqrt{NK}})$ dominates
- DSGD asymptotically achieve the minibatch SGD rate with batch size N
- **transient time**:

$$K \ge \mathcal{O}\left(\frac{N^3}{(1-\lambda)^4}\right)$$

DSGD Interpretation

Penalized formulation:

$$\min_{\mathbf{x}} \quad \mathbf{f}(\mathbf{x}) + \frac{1}{2\alpha}\|\mathbf{x}\|^2_{\mathbf{I}-\mathbf{W}}$$

- The term $\frac{1}{2\alpha}\|\mathbf{x}\|^2_{\mathbf{I}-\mathbf{W}}$ forces $\{x_i\}$ to be close to each other
- Applying gradient descent with stepsize α:

$$\mathbf{x}^{k+1} = \mathbf{W}\mathbf{x}^k - \alpha\nabla\mathbf{f}(\mathbf{x}^k)$$

  The above is exactly DGD in consensus form (2)

- This means that DGD solves an inexact penalized problem

EXTRA

Node-wise update:

$$x_i^{k+2} = \sum_{j \in \mathcal{N}_i} w_{ij}(2x_j^{k+1} - x_j^k) - \alpha(\nabla f(x_i^{k+1}) - \nabla f(x_i^k))$$

with $x_i^1 = \sum_{j \in \mathcal{N}_i} w_{ij} x_j^0 - \alpha \nabla f_i(x_i^0)$

Network form:

$$\mathbf{x}^{k+2} = \mathbf{W}\left(2\mathbf{x}^{k+1} - \mathbf{x}^k\right) - \alpha\left(\nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k)\right)$$

- for noiseless and constant stepsize it converges exactly with rate $\mathcal{O}(1/K)$
- Can be interpreted as a primal-descent dual-ascent applied to the augmented Lagrangian of problem:

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & \mathbf{f}(\mathbf{x}) + \frac{\rho}{2}\|\mathbf{x}\|_{\mathbf{I}-\mathbf{W}}^2 \\ \text{subject to} & (\mathbf{I} - \mathbf{W})^{1/2}\mathbf{x} = \mathbf{0} \end{array}$$

Exact-Diffusion (a.k.a. D^2 and NIDS)

Node-wise update:

$$x_i^{k+2} = \sum_{j \in \mathcal{N}_i} w_{ij} \Big(2x_j^{k+1} - x_j^k - \alpha \big(\nabla f_j(x_j^{k+1}) - \nabla f_j(x_j^k) \big) \Big),$$

with $x_i^1 = \sum_{j \in \mathcal{N}_i} w_{ij} \big(x_j^0 - \alpha \nabla f_j(x_j^0) \big)$

Network form:

$$\mathbf{x}^{k+2} = \mathbf{W} \Big(2\mathbf{x}^{k+1} - \mathbf{x}^k - \alpha \big(\nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k) \big) \Big)$$

- Compared to EXTRA, it has the adapt-then-combine form
- Shown to be more stable than EXTRA

Gradient-Tracking

The adapt-then-combine gradient-tracking method (ATC-GT) is

$$
\begin{aligned}
x_i^{k+1} &= \sum_{j\in\mathcal{N}_i} w_{ij}(x_j^k - \alpha g_j^k) \\
g_i^{k+1} &= \sum_{j\in\mathcal{N}_i} w_{ij}\big(g_j^k + \nabla f_j(x_j^{k+1}) - \nabla f_j(x_j^k)\big)
\end{aligned}
$$

- g_i^k update employs a dynamic consensus mechanism to tracks the average of the local gradients
- Requires communicating two vectors per communication round
- Works under a variety of settings: directed, time-varying networks

ATC-GT:

$$\begin{aligned}\mathbf{x}^{k+1} &= \mathbf{W}(\mathbf{x}^k - \alpha \mathbf{g}^k)\\ \mathbf{g}^{k+1} &= \mathbf{W}\big(\mathbf{g}^k + \nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k)\big)\end{aligned}$$

Subtracting $\mathbf{W}\mathbf{x}^k$ from both sides of the first equation, we have for $k \geq 0$

$$\begin{aligned}\mathbf{x}^{k+2} - \mathbf{W}\mathbf{x}^{k+1} &= \mathbf{W}\mathbf{x}^{k+1} - \mathbf{W}^2\mathbf{x}^k - \alpha(\mathbf{W}\mathbf{g}^{k+1} - \mathbf{W}^2\mathbf{g}^k)\\ &= \mathbf{W}\mathbf{x}^{k+1} - \mathbf{W}^2\mathbf{x}^k - \alpha\mathbf{W}(\mathbf{g}^{k+1} - \mathbf{W}\mathbf{g}^k)\end{aligned}$$

Using the second equation and rearranging, we get

$$\mathbf{x}^{k+2} = 2\mathbf{W}\mathbf{x}^{k+1} - \mathbf{W}^2\mathbf{x}^k - \alpha\mathbf{W}^2\big(\nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k)\big)$$

Prior transient times

METHOD	WORK	TRANSIENT TIME
DSGD	[prior]	$\mathcal{O}\left(\frac{N^3}{(1-\lambda)^4}\right)$
ED/D^2	[prior]	$\mathcal{O}\left(\frac{N^3}{(1-\lambda)^6}\right)$
ATC-GT	[prior]	$\mathcal{O}\left(\frac{N^3}{(1-\lambda)^6}\right)$

Contributions

- Proposing a unified decentralized algorithm (UDA) that incorporates a variety of existing methods such as EXTRA, Exact-Diffusion, GT
- Extending the framework to handle a common non-smooth term and establishing its linear convergence
- New analysis technique for proving convergence of the methods (EXTRA, Exact-Diffusion, and GT) for the stochastic online nonconvex settings with improved rates than previous works
- Proposing and establishing the convergence of local exact-diffusion, a localized form of exact-diffusion in which nodes perform multiple local updates between each communication

Outline

Introduction and Review

Unified Decentralized Algorithm (UDA)

Stochastic Unified Decentralized Algorithm (SUDA)

Local Exact-Diffusion

Exact methods

Exact formulation:

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & \mathbf{f}(\mathbf{x}) + \frac{1}{2\alpha}\|\mathbf{x}\|_{\mathbf{I}-\mathbf{C}}^2 \\ \text{subject to} & \mathbf{B}\mathbf{x} = \mathbf{0} \end{array} \tag{6}$$

- $\mathbf{B}$ and $\mathbf{C}$ satisfy $\mathbf{A}\mathbf{x} = \mathbf{0}$ and $(\mathbf{I} - \mathbf{C})\mathbf{x} = \mathbf{0}$ if and only if $x_1 = \cdots = x_N$
- Problem (6) can be solved using primal-dual and operator splitting based methods
- The obtained method depends on our choice of $\mathbf{B}$ and $\mathbf{C}$
- The augmented Lagrangian is:

$$L(\mathbf{x}, \mathbf{y}) = \mathbf{f}(\mathbf{x}) + \frac{1}{2\alpha}\|\mathbf{x}\|_{\mathbf{I}-\mathbf{C}}^2 + \mathbf{y}^T\mathbf{B}\mathbf{x}$$

  where $\mathbf{y}$ is the dual variable

Unified Decentralized Algorithm (UDA)

UDA:

$$\mathbf{z}^{k+1} = \mathbf{C}\mathbf{x}^k - \alpha \nabla \mathbf{f}(\mathbf{x}^k) - \mathbf{B}\mathbf{y}^k \tag{7a}$$

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \mathbf{B}\mathbf{z}^{k+1} \tag{7b}$$

$$\mathbf{x}^{k+1} = \mathbf{A}\mathbf{z}^{k+1} \tag{7c}$$

with $\mathbf{y}^0 = \mathbf{0}$

- $\mathbf{A}$ and $\mathbf{C}$ are doubly stochastic matrices
- $\mathbf{B}\mathbf{x} = \mathbf{0} \iff x_1 = x_2 = \cdots = x_N$
- We can recover various state-of-the-art methods by specific choices of $\mathbf{A}, \mathbf{B}, \mathbf{C}$

Special cases

We can rewrite UDA in terms of $\mathbf{x}^k$ only by noting that

$$\begin{aligned} \mathbf{z}^{k+2} - \mathbf{z}^{k+1} &= \mathbf{C}(\mathbf{x}^{k+1} - \mathbf{x}^k) - \alpha\big(\nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k)\big) - \mathbf{B}(\mathbf{y}^{k+1} - \mathbf{y}^k) \\ &= \mathbf{C}(\mathbf{x}^{k+1} - \mathbf{x}^k) - \alpha\big(\nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k)\big) - \mathbf{B}^2\mathbf{z}^{k+1} \end{aligned}$$

Rearranging, we have

$$\mathbf{z}^{k+2} = (\mathbf{I} - \mathbf{B}^2)\mathbf{z}^{k+1} + \mathbf{C}(\mathbf{x}^{k+1} - \mathbf{x}^k) - \alpha\big(\nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k)\big)$$

We now multiply both sides by $\mathbf{A}$ and using $\mathbf{A}\mathbf{B}^2 = \mathbf{B}^2\mathbf{A}$ gives

$$\mathbf{x}^{k+2} = (\mathbf{I} - \mathbf{B}^2 + \mathbf{A}\mathbf{C})\mathbf{x}^{k+1} - \mathbf{A}\mathbf{C}\mathbf{x}^k - \alpha\mathbf{A}\big(\nabla \mathbf{f}(\mathbf{x}^{k+1}) - \nabla \mathbf{f}(\mathbf{x}^k)\big),$$

for $k = 0, 1, \ldots$ with initialization $\mathbf{x}^1 = \mathbf{A}(\mathbf{C}\mathbf{x}^0 - \alpha\nabla \mathbf{f}(\mathbf{x}^0))$

For EXTRA and Exact-Diffusion (ED), we assume $\mathbf{W}$ to be symmetric and positive-semidefinite

EXTRA: When $\mathbf{A} = \mathbf{I}$, $\mathbf{B} = (\mathbf{I} - \mathbf{W})^{1/2}$, $\mathbf{C} = \mathbf{W}$, we get EXTRA:

$$\mathbf{x}^{k+2} = \mathbf{W}\big(2\mathbf{x}^{k+1} - \mathbf{x}^k\big) - \alpha\big(\nabla\mathbf{f}(\mathbf{x}^{k+1}) - \nabla\mathbf{f}(\mathbf{x}^k)\big)$$

Exact-Diffusion: When $\mathbf{A} = \mathbf{W}$, $\mathbf{B} = (\mathbf{I} - \mathbf{W})^{1/2}$, $\mathbf{C} = \mathbf{I}$, we get ED/D^2:

$$\mathbf{x}^{k+2} = \mathbf{W}\Big(2\mathbf{x}^{k+1} - \mathbf{x}^k - \alpha\big(\nabla\mathbf{f}(\mathbf{x}^{k+1}) - \nabla\mathbf{f}(\mathbf{x}^k)\big)\Big)$$

ATC-GT: When $\mathbf{A} = \mathbf{W}^2$, $\mathbf{B} = (\mathbf{I} - \mathbf{W})$, and $\mathbf{C} = \mathbf{I}$, we get:

$$\mathbf{x}^{k+2} = 2\mathbf{W}\mathbf{x}^{k+1} - \mathbf{W}^2\mathbf{x}^k - \alpha\mathbf{W}^2\big(\nabla\mathbf{f}(\mathbf{x}^{k+1}) - \nabla\mathbf{f}(\mathbf{x}^k)\big)$$

This is equivalent to ATC-GT

ATC and Non-ATC forms

ATC-UDA: $\mathbf{C} = \mathbf{I}$ and $\mathbf{A}$ is a primitive doubly stochastic matrix:

$$\mathbf{z}^{k+1} = \mathbf{C}\mathbf{x}^k - \alpha\nabla\mathbf{f}(\mathbf{x}^k) - \mathbf{B}\mathbf{y}^k \tag{8a}$$

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \mathbf{B}\mathbf{z}^{k+1} \tag{8b}$$

$$\mathbf{x}^{k+1} = \mathbf{A}\mathbf{z}^{k+1} \tag{8c}$$

(Exact-Diffusion, ATC-GT)

Non-ATC-UDA: $\mathbf{A} = \mathbf{I}$ and $\mathbf{C}$ is a primitive doubly stochastic matrix:

$$\mathbf{x}^{k+1} = \mathbf{C}\mathbf{x}^k - \alpha\nabla\mathbf{f}(\mathbf{x}^k) - \mathbf{B}\mathbf{y}^k \tag{9a}$$

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \mathbf{B}\mathbf{x}^{k+1} \tag{9b}$$

(EXTRA, DIGing, linearized ADMM)

Proximal Unified Decentralized Algorithm

$$\underset{x}{\text{minimize}} \quad \frac{1}{N}\sum_{i=1}^{N} f_i(x) + g(x)$$

$$\begin{aligned}
\mathbf{z}^{k+1} &= \mathbf{C}\mathbf{x}^k - \alpha\nabla\mathbf{f}(\mathbf{x}^k) - \mathbf{B}\mathbf{y}^k \\
\mathbf{y}^{k+1} &= \mathbf{y}^k + \mathbf{B}\mathbf{z}^{k+1} \\
\mathbf{x}^{k+1} &= \mathrm{prox}_{\frac{\alpha}{N}g}(\mathbf{A}\mathbf{z}^{k+1})
\end{aligned}$$

- Typical proximal methods:

$$\begin{aligned}
\mathbf{x}^{k+1} &= \mathrm{prox}_{\alpha g_i}\left(\mathbf{C}\mathbf{x}^k - \alpha\nabla\mathbf{f}(\mathbf{x}^k) - \mathbf{B}\mathbf{y}^k\right) \\
\mathbf{y}^{k+1} &= \mathbf{y}^k + \mathbf{B}\mathbf{x}^{k+1}
\end{aligned}$$

- For different local nonsmooth terms, linear convergence is not possible in the general case

PUDA convergence

Assumptions

- f_i is L-smooth and μ-strongly convex
- g is proper and lower-semicontinuous convex function
- We assume:
  - $\mathbf{A}^2 \leq \mathbf{I} - \mathbf{B}^2$ and $\mathbf{0} \leq \mathbf{I} - \mathbf{C} < 2\mathbf{I}$, for ATC-UDA
  - and $\mathbf{0} \leq \mathbf{B}^2 \leq \mathbf{I} - \mathbf{C} < \mathbf{I}$, for Non-ATC-UDA

  The above holds for all previous methods when $\mathbf{W}$ is symmetric and positive-semidefinite

ATC-UDA: If the step-size satisfies $\alpha < \frac{1+\lambda_{\min}(\mathbf{C})}{L}$, then

$$\|\mathbf{x}^k - \mathbf{1} \otimes x^\star\|^2 \leq \gamma^k C_0$$

where $\gamma = \max\left\{1 - \alpha\mu\left(1 + \lambda_{\min}(\mathbf{C}) - \alpha L\right), 1 - \underline{\sigma}\left(\mathbf{B}^2\right)\right\} < 1$ and $C_0 \geq 0$

Non-ATC-UDA: If the step-size satisfies $\alpha < \frac{2\lambda_{\min}(\mathbf{C})}{L}$, then

$$\|\mathbf{x}^k - \mathbf{1} \otimes x^\star\|_{\mathbf{Q}}^2 \leq \gamma^k C_0$$

where
$\mathbf{Q} = \mathbf{I} - \mathbf{B}^2 > 0, \gamma \triangleq \max\left\{1 - \alpha\mu\left(2 - \frac{\alpha L}{\lambda_{\min}(\mathbf{C})}\right), 1 - \underline{\sigma}\left(\mathbf{B}^2\right)\right\} < 1$ and $C_0 \geq 0$

Outline

Introduction and Review

Unified Decentralized Algorithm (UDA)

Stochastic Unified Decentralized Algorithm (SUDA)

Local Exact-Diffusion

Stochastic Unified Decentralized Algorithm

Stochastic UDA (SUDA)

$$\begin{aligned}
\mathbf{z}^{k+1} &= \mathbf{C}\mathbf{x}^k - \alpha \nabla \mathbf{F}(\mathbf{x}^k, \boldsymbol{\xi}^k) - \mathbf{B}\mathbf{y}^k \\
\mathbf{y}^{k+1} &= \mathbf{y}^k + \mathbf{B}\mathbf{z}^{k+1} \\
\mathbf{x}^{k+1} &= \mathbf{A}\mathbf{z}^{k+1}
\end{aligned}$$

where $\nabla \mathbf{F}(\mathbf{x}, \boldsymbol{\xi}^k) \triangleq \mathrm{col}\{\nabla F_1(x_1, \xi_1^k), \ldots, \nabla F_N(x_N, \xi_N^k)\}$

Assuming $\mathbf{A}\mathbf{B} = \mathbf{B}\mathbf{A}$, then through change of variable $\underline{\mathbf{y}}^k = \mathbf{A}\mathbf{y}^k$, we can describe SUDA as follows:

$$\mathbf{x}^{k+1} = \mathbf{A}\big(\mathbf{C}\mathbf{x}^k - \alpha \nabla \mathbf{F}(\mathbf{x}^k, \boldsymbol{\xi}^k)\big) - \mathbf{B}\underline{\mathbf{y}}^k \tag{10a}$$

$$\underline{\mathbf{y}}^{k+1} = \underline{\mathbf{y}}^k + \mathbf{B}\mathbf{x}^{k+1} \tag{10b}$$

Assumptions

1. W is primitive, doubly stochastic, and symmetric
2. The matrices $\mathbf{A}, \mathbf{B}^2, \mathbf{C}$ are chosen as a polynomial function of $\mathbf{W}$:

$$\mathbf{A} = \sum_{l=0}^{p} a_l \mathbf{W}^l, \quad \mathbf{B}^2 = \sum_{l=0}^{p} b_l \mathbf{W}^l, \quad \mathbf{C} = \sum_{l=0}^{p} c_l \mathbf{W}^l$$

3. Each function f_i is L-smooth: $\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|$
4. The aggregate function $f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x)$ is bounded below, *i.e.*, $f(x) \geq f^\star > -\infty \; \forall \, x \in \mathbb{R}^m$ where $f^\star$ denote the optimal value of f
5. For all $\{i\}_{i=1}^{N}$ and $k = 0, 1, \ldots,$

$$\mathbb{E}_{\xi_i}\left[\nabla F_i(x_i;\xi_i) - \nabla f_i(x_i)\right] = 0$$
$$\mathbb{E}_{\xi_i}\left[\|\nabla F_i(x_i;\xi_i) - \nabla f_i(x_i)\|^2\right] \leq \sigma^2,$$

for some $\sigma^2 \geq 0$, and the random data $\{\xi_i^t\}$ are independent of each other for all $\{i\}_{i=1}^{N}$ and $\{t\}_{t \leq k}$

Fundamental transformations

Transformations I: Let

$$\bar{x}^k \triangleq \frac{1}{N}\left(\mathbf{1}_N^T \otimes I_m\right)\mathbf{x}^k = \frac{1}{N}\sum_{i=1}^{N} x_i^k$$

$$\bar{\mathbf{x}}^k \triangleq \mathbf{1}_N \otimes \bar{x}^k$$

$$\mathbf{s}^k \triangleq \mathbf{B}(\underline{\mathbf{y}}^k - \mathbf{B}\mathbf{x}^k) + \alpha\mathbf{A}\nabla\mathbf{f}(\bar{\mathbf{x}}^k)$$

then (10) can be rewritten as:

$$\mathbf{x}^{k+1} = (\mathbf{AC} - \mathbf{B}^2)\mathbf{x}^k - \mathbf{s}^k - \alpha\mathbf{A}(\nabla\mathbf{f}(\mathbf{x}^k) - \nabla\mathbf{f}(\bar{\mathbf{x}}^k) + \mathbf{w}^k) \tag{11a}$$

$$\mathbf{s}^{k+1} = \mathbf{s}^k + \mathbf{B}^2\mathbf{x}^k + \alpha\mathbf{A}(\nabla\mathbf{f}(\bar{\mathbf{x}}^{k+1}) - \nabla\mathbf{f}(\bar{\mathbf{x}}^k)) \tag{11b}$$

where $\mathbf{w}^k$ is the gradient noise defined as:

$$\mathbf{w}^k \triangleq \nabla\mathbf{F}(\mathbf{x}^k, \boldsymbol{\xi}^k) - \nabla\mathbf{f}(\mathbf{x}^k)$$

Averages

$$\bar{x}^{k+1} = \bar{x}^k - \alpha \overline{\nabla \mathbf{f}}(\mathbf{x}^k) - \alpha \bar{\mathbf{w}}^k$$

$$\bar{s}^{k+1} = \bar{s}^k + \frac{\alpha}{N} \sum_{i=1}^{N} (\nabla f_i(\bar{x}^{k+1}) - \nabla f_i(\bar{x}^k))$$

if $\bar{s}^0 = \frac{\alpha}{N} \sum_{i=1}^{N} \nabla f_i(\bar{x}^0)$ then $\bar{s}^k = \frac{\alpha}{N} \sum_{i=1}^{N} \nabla f_i(\bar{x}^k)$ for all k

Convergence proof direction: show that

$$x_i^k \to \bar{x}^k$$

$$s_i^k \to \bar{s}^k$$

$$(1/N) \sum_{i=1}^{N} \nabla f_i(x_i) \to 0$$

Weight matrix decomposition

$$W = U\Lambda U^T = \left[\begin{array}{cc} \frac{1}{\sqrt{N}}\mathbf{1} & \hat{U} \end{array} \right] \left[\begin{array}{cc} 1 & 0 \\ 0 & \hat{\Lambda} \end{array} \right] \left[\begin{array}{c} \frac{1}{\sqrt{N}}\mathbf{1}^T \\ \hat{U}^T \end{array} \right],$$

- $\hat{\Lambda} = \mathrm{diag}\,\{\lambda_i\}_{i=2}^N$; U is an orthogonal matrix $\left(UU^T = U^T U = I\right)$
- $\hat{U}$ is matrix that satisfies $\hat{U}\hat{U}^T = I_N - \frac{1}{N}\mathbf{1}\mathbf{1}^T$ and $\mathbf{1}^T\hat{U} = 0$;

It follows that

$$\mathbf{W} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T = \left[\begin{array}{cc} \frac{1}{\sqrt{N}}\mathbf{1} \otimes I_m & \hat{\mathbf{U}} \end{array} \right] \left[\begin{array}{cc} I_m & 0 \\ 0 & \hat{\mathbf{\Lambda}} \end{array} \right] \left[\begin{array}{c} \frac{1}{\sqrt{N}}\mathbf{1}^T \otimes I_m \\ \hat{\mathbf{U}}^T \end{array} \right]$$

where $\hat{\Lambda} \triangleq \hat{\Lambda} \otimes I_m$, $\mathbf{U}$ is an orthogonal matrix, and $\hat{\mathbf{U}} \triangleq \hat{U} \otimes I_m$ satisfies:

$$\hat{\mathbf{U}}^T\hat{\mathbf{U}} = \mathbf{I}, \quad \hat{\mathbf{U}}\hat{\mathbf{U}}^T = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T \otimes I_m, \quad \left(\mathbf{1}^T \otimes I_m\right)\hat{\mathbf{U}} = 0$$

Let $\mathbf{A}, \mathbf{B}^2, \mathbf{C}$ be chosen as polynomial function of $\mathbf{W}$:

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}_a\mathbf{U}^T = \begin{bmatrix} \frac{1}{\sqrt{N}}\mathbf{1} \otimes I_m & \hat{\mathbf{U}} \end{bmatrix} \begin{bmatrix} I_m & 0 \\ 0 & \hat{\mathbf{\Lambda}}_a \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{N}}\mathbf{1}^T \otimes I_m \\ \hat{\mathbf{U}}^T \end{bmatrix}$$

$$\mathbf{C} = \mathbf{U}\mathbf{\Lambda}_c\mathbf{U}^T = \begin{bmatrix} \frac{1}{\sqrt{N}}\mathbf{1} \otimes I_m & \hat{\mathbf{U}} \end{bmatrix} \begin{bmatrix} I_m & 0 \\ 0 & \hat{\mathbf{\Lambda}}_c \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{N}}\mathbf{1}^T \otimes I_m \\ \hat{\mathbf{U}}^T \end{bmatrix}$$

$$\mathbf{B}^2 = \mathbf{U}\mathbf{\Lambda}_b^2\mathbf{U}^T = \begin{bmatrix} \frac{1}{\sqrt{N}}\mathbf{1} \otimes I_m & \hat{\mathbf{U}} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \hat{\mathbf{\Lambda}}_b^2 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{N}}\mathbf{1}^T \otimes I_m \\ \hat{\mathbf{U}}^T \end{bmatrix}$$

where

$$\hat{\mathbf{\Lambda}}_a = \mathrm{diag}\{\lambda_{a,i}\}_{i=2}^N \otimes I_m, \quad \hat{\mathbf{\Lambda}}_c = \mathrm{diag}\{\lambda_{i,c}\}_{i=2}^N \otimes I_m,$$

and $\hat{\mathbf{\Lambda}}_b^2 = \mathrm{diag}\{\lambda_{b,i}^2\}_{i=2}^N \otimes I_m$

Transformation II: Multiplying both sides of (11) by $\mathbf{U}^T$, we get

$$\mathbf{U}^T\mathbf{x}^{k+1} = (\mathbf{\Lambda}_a\mathbf{\Lambda}_c - \mathbf{\Lambda}_b^2)\mathbf{U}^T\mathbf{x}^k - \mathbf{U}^T\mathbf{s}^k - \alpha\mathbf{\Lambda}_a\mathbf{U}^T\big(\nabla\mathbf{f}(\mathbf{x}^k) - \nabla\mathbf{f}(\bar{\mathbf{x}}^k) + \mathbf{w}^k\big)$$
$$\mathbf{U}^T\mathbf{s}^{k+1} = \mathbf{U}^T\mathbf{s}^k + \mathbf{\Lambda}_b^2\mathbf{U}^T\mathbf{x}^k + \alpha\mathbf{\Lambda}_a\mathbf{U}^T\big(\nabla\mathbf{f}(\bar{\mathbf{x}}^{k+1}) - \nabla\mathbf{f}(\bar{\mathbf{x}}^k)\big)$$

- Since $\hat{\mathbf{U}}^T\hat{\mathbf{U}} = \mathbf{I}$ and $\hat{\mathbf{U}}\hat{\mathbf{U}}^T = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T \otimes I_m$, it holds that

$$\|\hat{\mathbf{U}}^T\mathbf{x}\|^2 = \mathbf{x}^T\hat{\mathbf{U}}\hat{\mathbf{U}}^T\hat{\mathbf{U}}\hat{\mathbf{U}}^T\mathbf{x} = \|\hat{\mathbf{U}}\hat{\mathbf{U}}^T\mathbf{x}^k\|^2 \overset{(21)}{=} \|\mathbf{x}^k - \overline{\mathbf{x}}^k\|^2$$

- $\|\hat{\mathbf{U}}^T\mathbf{x}^k\|^2$ measures the deviation of $\mathbf{x}^k$ from the average $\overline{\mathbf{x}}^k$
- $\|\hat{\mathbf{U}}^T\mathbf{s}^k\|^2$ measures the deviation of $\mathbf{s}^k$ with the average $\left(\frac{1}{N}\mathbf{1}\mathbf{1}^T \otimes I_N\right)\mathbf{s}^k = \mathbf{1} \otimes \frac{\alpha}{N}\sum_{i=1}^{N}\nabla f_i\left(\bar{x}^k\right)$

Final transformed recursion

There exists an invertible matrix $\hat{\mathbf{V}}$ such that recursion (11) can be transformed into

$$
\begin{aligned}
\bar{x}^{k+1} &= \bar{x}^k - \alpha\overline{\nabla \mathbf{f}}(\mathbf{x}^k) - \alpha\overline{\mathbf{w}}^k, \\
\hat{\mathbf{e}}^{k+1} &= \mathbf{\Gamma}\hat{\mathbf{e}}^k - \alpha\hat{\mathbf{V}}^{-1}\left[\begin{array}{c} \frac{1}{v}\hat{\mathbf{\Lambda}}_a\hat{\mathbf{U}}^T(\nabla \mathbf{f}(\mathbf{x}^k) - \nabla \mathbf{f}(\overline{\mathbf{x}}^k) + \mathbf{w}^k) \\ \frac{1}{v}\hat{\mathbf{\Lambda}}_b^{-1}\hat{\mathbf{\Lambda}}_a\hat{\mathbf{U}}^T(\nabla \mathbf{f}(\overline{\mathbf{x}}^k) - \nabla \mathbf{f}(\overline{\mathbf{x}}^{k+1})) \end{array}\right],
\end{aligned}
$$

where $\mathbf{\Gamma}$ satisfies $\|\mathbf{\Gamma}\| < 1$, $v > 0$ is an arbitrary constant, and

$$
\hat{\mathbf{e}}^k \triangleq \frac{1}{v}\hat{\mathbf{V}}^{-1}\left[\begin{array}{c} \hat{\mathbf{U}}^T\mathbf{x}^k \\ \hat{\mathbf{\Lambda}}_b^{-1}\hat{\mathbf{U}}^T\mathbf{s}^k \end{array}\right]
$$

- Observe that $\|v\hat{\mathbf{V}}\hat{\mathbf{e}}^k\|^2 = \|\hat{\mathbf{U}}^T\mathbf{x}^k\|^2 + \|\hat{\mathbf{\Lambda}}_b^{-1}\hat{\mathbf{U}}^T\mathbf{s}^k\|^2$
- Thus, the vector $\hat{\mathbf{e}}^k$ can be interpreted as a measure of a weighted deviation of $\mathbf{x}^k$ and $\mathbf{s}^k$ from $\overline{\mathbf{x}}^k$ and $\mathbf{1} \otimes \frac{\alpha}{N}\sum_{i=1}^N \nabla f_i\left(\bar{x}^k\right)$

SUDA convergence

When the step size satisfies α is sufficiently small, then, the iterates $\{\mathbf{x}^k\}$ of SUDA with $\mathbf{x}^0 = \mathbf{1} \otimes x^0$ ($x^0 \in \mathbb{R}^m$) satisfy

$$\frac{1}{K}\sum_{k=0}^{K-1} \mathbb{E}\,\|\overline{\nabla \mathbf{f}}(\mathbf{x}^k)\|^2 \le \frac{8(f(x^0) - f^\star)}{\alpha K} + \frac{4\alpha L\sigma^2}{N} + \frac{12\alpha^2 L^2 v_1^2 v_2^2 \zeta_0^2}{\underline{\lambda_b}^2(1-\gamma)K} + \frac{16\alpha^2 L^2 v_1^2 v_2^2 \lambda_a^2 \sigma^2}{1-\gamma} + \frac{16\alpha^4 L^4 v_1^2 v_2^2 \lambda_a^2 \sigma^2}{\underline{\lambda_b}^2(1-\gamma)^2 N}, \tag{12}$$

where $v_1 \triangleq \|\hat{\mathbf{V}}\|$, $v_2 \triangleq \|\hat{\mathbf{V}}^{-1}\|$ and

$$\gamma \triangleq \|\mathbf{\Gamma}\| < 1, \quad \underline{\lambda_b} \triangleq \frac{1}{\|\mathbf{\Lambda}_b^{-1}\|}, \quad \lambda_a \triangleq \|\mathbf{\Lambda}_a\|,$$

$$\zeta_0^2 \triangleq \tfrac{1}{N}\|(\mathbf{A} - \tfrac{1}{N}\mathbf{1}^T\mathbf{1} \otimes I_m)\big(\nabla \mathbf{f}(\mathbf{x}^0) - \mathbf{1} \otimes \nabla f(x^0)\big)\|^2$$

- For noiseless and constant stepsize, SUDA converges exactly at rate $\mathcal{O}(1/K)$

- for stochastic case, if we set $\alpha = \mathcal{O}(\sqrt{N/K})$, then

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\left\|\overline{\nabla \mathbf{f}}(\mathbf{x}^k)\right\|^2 \leq O\left(\frac{\sigma}{\sqrt{NK}} + \frac{1}{K} + \frac{N\sigma^2}{N+\sigma^2 K} + \frac{N\zeta_0^2}{NK+\sigma^2 K^2}\right)$$

  Rate is $\mathcal{O}(1/\sqrt{KN})$ and SUDA asymptotically achieves linear speedup

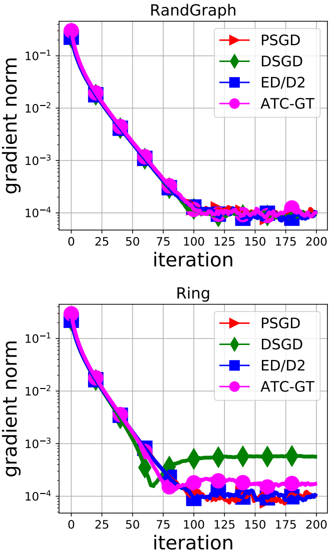
- we can get the rate of ED and GT methods by plugging the parameter from our choice of $\mathbf{A}, \mathbf{B}, \mathbf{C}$

Exact diffusion and gradient tracking rates

METHOD	WORK	CONVERGENCE RATE
ED/D^2	[prior]	$O\left(\frac{1}{\sqrt{NK}} + \frac{N\lambda^2}{(1-\lambda)^3K} + \frac{N\varsigma_0^2}{(1-\lambda)^2K^2}\right)$
	SUDA	$O\left(\frac{1}{\sqrt{NK}} + \frac{N\lambda^2}{(1-\lambda)K} + \frac{N\lambda^2\varsigma_0^2}{(1-\lambda)^2K^2}\right)$
ATC-GT	[prior]	$O\left(\frac{1}{\sqrt{NK}} + \frac{N\lambda^2}{(1-\lambda)^3K} + \frac{\lambda^4\sum_{i=1}^N \|\nabla f_i(0)\|^2}{(1-\lambda)^3K^2}\right)$
	SUDA	$O\left(\frac{1}{\sqrt{NK}} + \frac{N\lambda^4}{(1-\lambda)K} + \frac{N\lambda^4}{(1-\lambda)^4K^2} + \frac{N\lambda^4\varsigma_0^2}{(1-\lambda)^3K^2}\right)$

Transient times

METHOD	WORK	TRANSIENT TIME
DSGD	[prior]	$\mathcal{O}\left(\frac{N^3}{(1-\lambda)^4}\right)$
ED/D^2	[prior]	$\mathcal{O}\left(\frac{N^3}{(1-\lambda)^6}\right)$
	SUDA	$\mathcal{O}\left(\frac{N^3}{(1-\lambda)^2}\right)$
ATC-GT	[prior]	$\mathcal{O}\left(\frac{N^3}{(1-\lambda)^6}\right)$
	SUDA	$\mathcal{O}\left(\max\left\{\frac{N^3}{(1-\lambda)^2},\ \frac{N}{(1-\lambda)^{8/3}}\right\}\right)$



Top-left: $\lambda = 0.32$; Top-right: $\lambda = 0.94$; Bottom-left: $\lambda = 0.99$; Bottom-right: $\lambda = 0.999$

Outline

Introduction and Review

Unified Decentralized Algorithm (UDA)

Stochastic Unified Decentralized Algorithm (SUDA)

Local Exact-Diffusion

Exact-Diffusion (ED)

Recall that ED can be written as

$$\mathbf{z}^{k+1} = \mathbf{x}^k - \alpha \nabla \mathbf{f}(\mathbf{x}^k) - \mathbf{y}^k \tag{13a}$$

$$\mathbf{y}^{k+1} = \mathbf{y}^k + (\mathbf{I} - \mathbf{W})\mathbf{z}^{k+1} \tag{13b}$$

$$\mathbf{x}^{k+1} = \mathbf{W}\mathbf{z}^{k+1} \tag{13c}$$

To get a locally update variant we can run the first step multiple times before communication

Local exact-diffusion (LED)

Given $\mathbf{x}^0$, set $\mathbf{y}^0 = (\mathbf{I} - \mathbf{W})\mathbf{x}^0$ (or $\mathbf{y}^0 = \mathbf{0}$) and update for $r = 0, 1, 2, \ldots$

1. *Local primal updates:* set $\boldsymbol{\Phi}_0^r = \mathbf{x}^r$, for $t = 0, \ldots, \tau - 1$:

$$\boldsymbol{\Phi}_{t+1}^r = \boldsymbol{\Phi}_t^r - \alpha \nabla \mathbf{F}(\boldsymbol{\Phi}_t^r; \boldsymbol{\xi}_t^r) - \beta \mathbf{y}^r \tag{14a}$$

2. *Diffusion round:*

$$\mathbf{x}^{r+1} = \mathbf{W} \boldsymbol{\Phi}_\tau^r \tag{14b}$$

3. *Local dual update:*

$$\mathbf{y}^{r+1} = \mathbf{y}^r + (\mathbf{I} - \mathbf{W}) \boldsymbol{\Phi}_\tau^r \tag{14c}$$

LED

node i input: x_i^0, $\alpha > 0$, $\beta > 0$, and τ
initialize $y_i^0 = x_i^0 - \sum_{j \in \mathcal{N}_i} w_{ij} x_j^0$ (or $y_i^0 = 0$)
repeat for $r = 0, 1, 2, \ldots$

1. `Local primal updates:` set $\phi_{i,0}^r = x_i^r$ and do τ local updates:

$$\phi_{i,t+1}^r = \phi_{i,t}^r - \alpha \nabla F_i(\phi_{i,t}^r; \xi_{i,t}^r) - \beta y_i^r, \quad t = 0, \ldots, \tau - 1 \tag{15a}$$

2. `Diffusion:`

$$x_i^{r+1} = \sum_{j \in \mathcal{N}_i} w_{ij} \phi_{j,\tau}^r \tag{15b}$$

3. `Local dual update:`

$$y_i^{r+1} = y_i^r + \phi_{i,\tau}^r - x_i^{r+1} \tag{15c}$$

Connection with existing methods

SCAFFOLD: Server parameters (x^r, c^r) are sent to the participating nodes $\mathcal{S}_r \subset [N]$. Each participating node $i \in \mathcal{S}_r$ initializes $\phi^r_{i,0} = x^r$:

$$\phi^r_{i,t+1} = \phi^r_{i,t} - \alpha(\nabla F_i(x_{i,t}; \xi_{i,t}) - c^r_i + c^r), \quad t = 0, 1, \ldots, \tau - 1$$

$$c_i^{r+1} = \begin{cases} \text{option I:} & \nabla F_i(x^r; \xi_i), \text{ or} \\ \text{option II:} & c_i^r - c^r + \frac{1}{\tau\alpha}(x^r - \phi^r_{i,\tau}) \end{cases}$$

Server parameters are then updated using the aggregated updates:

$$x^{r+1} = x^r + \frac{\alpha_g}{|\mathcal{S}_r|} \sum_{i \in \mathcal{S}_r} (\phi^r_{i,\tau} - x^r)$$

$$c^{r+1} = c^r + \frac{1}{N} \sum_{i \in \mathcal{S}_r} (c_i^{r+1} - c_i^r)$$

$$\overset{\text{option II}}{=} (1 - \tfrac{|\mathcal{S}_r|}{N})c^r + \frac{1}{\tau\alpha N} \sum_{i \in \mathcal{S}_r} (x^r - \phi^r_{i,\tau})$$

Full participation: Let $\bar{c}_i^r = \tau\alpha(c^r - c_i^r)$, then

$$\begin{aligned}
\phi_{i,t+1}^r &= \phi_{i,t}^r - \alpha(\nabla F_i(x_{i,t};\xi_{i,t}) - c_i^r + c^r) \\
&= \phi_{i,t}^r - \alpha\nabla F_i(x_{i,t};\xi_{i,t}) - \tfrac{1}{\tau}\bar{c}_i^r, \quad t = 0, 1, \ldots, \tau - 1 \\
x^{r+1} &= (1-\alpha_g)x^r + \tfrac{\alpha_g}{N}\sum_{j=1}^N \phi_{j,\tau}^r
\end{aligned}$$

and

$$c^{r+1} = \tfrac{1}{\tau\alpha N}\sum_{j=1}^N (x^r - \phi_{j,\tau}^r)$$

We have

$$c_i^{r+1} - c^{r+1} = -c^{r+1} + c_i^r - c^r + \tfrac{1}{\tau\alpha}(x^r - \phi_{i,\tau}^r)$$

multiplying by $-\tau\alpha$

$$\begin{aligned}\bar{c}_i^{r+1} &= \tau\alpha c^{r+1} + \bar{c}_i^r - (x^r - \phi_{i,\tau}^r) \\ &= \bar{c}_i^r + \tfrac{1}{N}\sum_{j=1}^{N}(x^r - \phi_{j,\tau}^r) - (x^r - \phi_{i,\tau}^r) \\ &= \bar{c}_i^r + \phi_{i,\tau}^r - \tfrac{1}{N}\sum_{j=1}^{N}\phi_{j,\tau}^r\end{aligned}$$

SCAFFOLD with full participation can rewritten as

$$\phi_{i,t+1}^r = \phi_{i,t}^r - \alpha\nabla F_i(x_{i,t};\xi_{i,t}) - \tfrac{1}{\tau}\bar{c}_i^r, \quad t = 0, 1, \ldots, \tau - 1 \tag{16a}$$

$$x^{r+1} = (1-\alpha_g)x^r + \frac{\alpha_g}{N}\sum_{j=1}^{N}\phi_{j,\tau}^r \tag{16b}$$

$$\bar{c}_i^{r+1} = \bar{c}_i^r + \phi_{i,\tau}^r - \frac{1}{N}\sum_{j=1}^{N}\phi_{j,\tau}^r \tag{16c}$$

FedGATE

$$
\begin{aligned}
\phi_{i,t+1}^{r} &= \phi_{i,t}^{r} - \alpha\big(\nabla F_i(\phi_{i,t}^{r};\xi_{i,t}^{r}) - \delta_i^{r}\big) \quad t = 0,\dots,\tau-1 \\
x^{r+1} &= x^{r} - \alpha\gamma\Big(x^{r} - \tfrac{1}{N}\sum_{j=1}^{N}\phi_{j,\tau}^{r}\Big) \\
\delta_i^{r+1} &= \delta_i^{r} - \tfrac{1}{\alpha\tau}\Big(\phi_{i,\tau}^{r} - \tfrac{1}{N}\sum_{j=1}^{N}\phi_{j,\tau}^{r}\Big)
\end{aligned}
$$

Letting $\bar{c}_i^{r} = -\alpha\tau\delta_i^{r}$ and $\alpha_g = \alpha\gamma$, we can rewrite FedGATE as

$$
\phi_{i,t+1}^{r} = \phi_{i,t}^{r} - \alpha\nabla F_i(\phi_{i,t}^{r};\xi_{i,t}^{r}) - \tfrac{1}{\tau}\bar{c}_i^{r} \quad t = 0,\dots,\tau-1 \tag{17a}
$$

$$
x^{r+1} = (1-\alpha_g)x^{r} + \tfrac{\alpha_g}{N}\sum_{j=1}^{N}\phi_{j,\tau}^{r} \tag{17b}
$$

$$
\bar{c}_i^{r} = \bar{c}_i^{r} + \phi_{i,\tau}^{r} - \tfrac{1}{N}\sum_{j=1}^{N}\phi_{j,\tau}^{r} \tag{17c}
$$

This is the same as SCAFFOLD representation (16) with full node participation

Relation with SCAFFOLD, FedGATE, and VRL-SGD

Using the network notation with $\mathbf{x}^r = \mathbf{1} \otimes x^r$, we can rewrite FedGATE and SCAFFOLD as

$$\begin{aligned}
\mathbf{\Phi}_{t+1}^r &= \mathbf{\Phi}_t^r - \alpha \nabla \mathbf{F}(\mathbf{\Phi}_t^r; \boldsymbol{\xi}_t^r) - \tfrac{1}{\tau}\mathbf{y}^r, \quad t = 0, \ldots, \tau - 1 && \text{(18a)} \\
\mathbf{x}^{r+1} &= (1 - \alpha_g)\mathbf{x}^r + \tfrac{\alpha_g}{N}\mathbf{1}\mathbf{1}^T \mathbf{\Phi}_\tau^r && \text{(18b)} \\
\mathbf{y}^{r+1} &= \mathbf{y}^r + (\mathbf{I} - \tfrac{1}{N}\mathbf{1}\mathbf{1}^T)\mathbf{\Phi}_\tau^r. && \text{(18c)}
\end{aligned}$$

- The above reduces to VRL-SGD when $\alpha_g = 1$
- When $\alpha_g = 1$, the update becomes LED (14) with $\mathbf{W} = \frac{1}{N}\mathbf{1}\mathbf{1}^T$ and $\beta = 1/\tau$; In other words, FedGATE and SCAFFOLD with $\alpha_g = 1$ (VRL-SGD) is the same as LED for the fully connected network case
- For $\tau = 1$, these algorithms can be interpreted as the primal-dual method PDFP2O (aka PAPC)

LED rate

There exist a constant stepsizes α that yields the following rates

Nonconvex rate:

$$\frac{1}{R}\sum_{r=0}^{R-1}\left(\mathbb{E}\,\|\nabla f(\bar{x}^r)\|^2 + \frac{1}{\tau}\sum_{t=0}^{\tau-1}\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(\phi_{i,t}^r)\|^2\right)$$

$$\leq \mathcal{O}\left(\frac{L\tilde{f}(\bar{x}^0)\sigma}{N\tau R}\right)^{\frac{1}{2}} + \mathcal{O}\left(\frac{1}{\rho^{1/3}}\left(\frac{\tilde{f}(\bar{x}^0)L\sigma}{\sqrt{\tau}R}\right)^{\frac{2}{3}}\right) + \mathcal{O}\left(\frac{L\frac{\tilde{f}(\bar{x}^0)}{\rho} + \varsigma_0^2}{R}\right)$$

Convex rate:

$$\frac{1}{R}\sum_{r=0}^{R-1}\mathbb{E}[f(\bar{x}^r) - f(x^\star)]$$

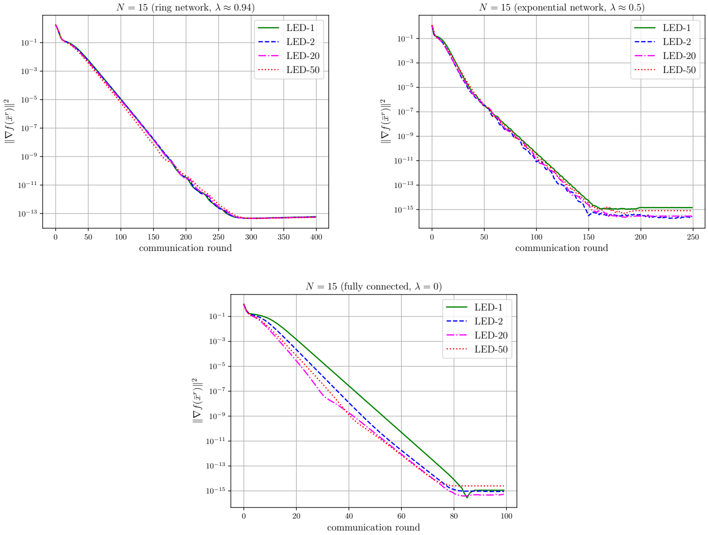
$$\leq \mathcal{O}\left(\frac{L\|\bar{x}^0 - x^\star\|^2\sigma}{N\tau R}\right)^{\frac{1}{2}} + \mathcal{O}\left(\frac{L^{1/3}}{\rho^{1/3}}\left(\frac{\|\bar{x}^0 - x^\star\|^2\sigma}{\sqrt{\tau}R}\right)^{\frac{2}{3}}\right) + \mathcal{O}\left(\frac{L\frac{\|\bar{x}^0 - x^\star\|^2}{\rho} + \varsigma_0^2}{R}\right)$$

Strongly-convex rate:

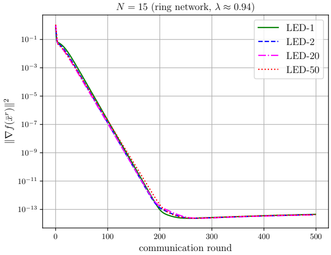
$$\mathbb{E}\,\|\bar{x}^R - x^\star\|^2 \le \tilde{\mathcal{O}}\left(\frac{\sigma^2}{\tau N R}\right) + \tilde{\mathcal{O}}\left(\frac{\sigma^2}{\rho \tau R^2}\right) + \tilde{\mathcal{O}}\left(\exp\left[-\rho R\right]\left(\|\bar{x}^0 - x^\star\|^2 + \varsigma_0\right)\right)$$

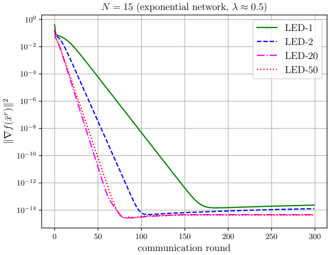
- $\tilde{f}(\bar{x}^0) \triangleq f(\bar{x}^0) - f^\star$
- $\bar{x}^0 \triangleq (1/N)\sum\limits_{i=1}^{N} x_i^0$
- $\rho \triangleq 1 - \lambda$
- $\varsigma_0^2 \triangleq \frac{1}{N}\sum_{i=1}^{N} \|\nabla f_i(\bar{x}^0) - \nabla f(\bar{x}^0)\|^2$
- The notation $\tilde{\mathcal{O}}(\cdot)$ ignores logarithmic factors

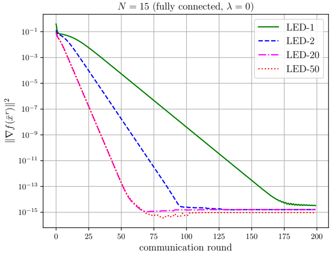
Benefits of local steps? Large data heterogeneity



Benefits of local steps? Similar data across the network







Main References

UDA and SUDA

1. S. A. Alghunaim, Ernest K. Ryu, K. Yuan, and A. H. Sayed, “Decentralized proximal gradient algorithms with linear convergence rates,” IEEE Trans. Automatic Control, vol. 66, no. 6, pp. 2787-2794, June 2021.
2. S. A. Alghunaim and K. Yuan, “A unified and refined convergence analysis for non-convex decentralized learning,” IEEE Trans. on Signal Processing, vol. 70, pp. 3264–3279, June 2022.

Exact-Diffusion

3. K. Yuan, S. A. Alghunaim, B. Ying, and A. H. Sayed. “On the influence of bias-correction on distributed stochastic optimization,” IEEE Trans. on Signal Processing, vol. 68, 4352-4367, July 2020.
4. K. Yuan, S. A. Alghunaim, and Xinmeng Huang, “Removing data heterogeneity influence enhances network topology dependence of decentralized SGD,” arXiv:2105.08023, May, 2021.
5. S. A. Alghunaim, “Local exact-diffusion for decentralized optimization and learning,” arXiv:2302.00620, Feb., 2023.

Gradient-tracking

6. S. A. Alghunaim and K. Yuan, “An enhanced gradient-tracking bound for distributed online stochastic convex optimization,” arXiv:2301.02855, Jan., 2023.

Other references I

Primal-dual decentralized methods

7. S. A. Alghunaim and A. H. Sayed, “Linear convergence of primal-dual gradient methods and their performance in distributed optimization,” Automatica, Volume 117, July 2020.
8. S. A. Alghunaim, K. Yuan, and A. H. Sayed, “A linearly convergent proximal gradient algorithm for decentralized optimization,” in Advances on Neural Information Processing Systems (NeurIPS), Vancouver, Canada, December 2019.

DSGD

9. J. Chen and A. H. Sayed, “Distributed Pareto optimization via diffusion strategies,” IEEE J. Selected Topics in Signal Processing, vol. 7, no. 2, pp. 205-220, April 2013.
10. A. H. Sayed, “Adaptation, learning, and optimization over neworks.,” Foundations and Trends in Machine Learning, vol. 7, no. 4-5, pp. 311–801, 2014.
11. K. Yuan, Q. Ling, and W. Yin, “On the convergence of decentralized gradient descent,” SIAM Journal on Optimization, vol. 26, no. 3, pp. 1835–1854, 2016.
12. X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, “Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent,” in Advances in Neural Information Processing Systems (NIPS), (Long Beach, CA, USA), pp. 5330–5340, 2017.
13. A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich, “A unified theory of decentralized SGD with changing topology and local updates,” in Proceedings of the International Conference on Machine Learning, vol. 119, (Virtual/online), pp. 5381–5393, PMLR, Jul 2020.

Other references II

GT

14. R. Xin, U. A. Khan, and S. Kar, "An improved convergence analysis for decentralized online stochastic non-convex optimization," IEEE Transactions on Signal Processing, vol. 69, pp. 1842–1858, 2021.
15. A. Koloskova, T. Lin, and S. U. Stich, "An improved analysis of gradient tracking for decentralized machine learning," Advances in Neural Information Processing Systems, vol. 34, pp. 11422– 11435, 2021.

SCAFFOLD

16. S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in Proceedings of the International Conference on Machine Learning, vol. 119, (Virtual/online), pp. 5132–5143, PMLR, Jul 2020.

FedGATE

17. F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, "Federated learning with compression: Unified analysis and sharp guarantees," in Proceedings of The International Conference on Artificial Intelligence and Statistics, vol. 130, (Virtual/online), pp. 2350–2358, PMLR, Apr 2021.

VRL-SGD

18. X. Liang, S. Shen, J. Liu, Z. Pan, E. Chen, and Y. Cheng, "Variance reduced local SGD with lower communication complexity," Preprint on arXiv:1912.12844, 2019.

Other references III

PDFP2O/PAPC

19. P. Chen, J. Huang, and X. Zhang, “A primal-dual fixed point algorithm for convex separable minimization with applications to image restoration,” Inverse Problems, vol. 29, p. 025011, Jan. 2013.
20. Y. Drori, S. Sabach, and M. Teboulle, “A simple algorithm for a class of nonsmooth convex–concave saddle-point problems,” Operations Research Letters, vol. 43, no. 2, pp. 209–214, 2015.
21. I. Loris and C. Verhoeven, “On a generalization of the iterative soft-thresholding algorithm for the case of non-separable penalty,” Inverse problems, vol. 27, no. 12, p. 125007, 2011.