Introduction to Distributed Optimization

Pranav Reddy

25 September 2024

Overview

- Introduction
  - Applications
- Preliminaries
- Distributed Averaging
  - Convergence
  - Computation of Weights
- First-Order Methods
  - Convergence of Consensus Error
  - Convergence of Function Values
- Conclusion

Goals

1. Introduce the setting of distributed optimization

Goals

1. Introduce the setting of distributed optimization
2. Understand the importance of consensus

Goals

1. Introduce the setting of distributed optimization
2. Understand the importance of consensus
3. Introduce and prove convergence of different first-order methods

Goals

1. Introduce the setting of distributed optimization
2. Understand the importance of consensus
3. Introduce and prove convergence of different first-order methods
4. Understand limitations of distributed optimization

Goals

1. Introduce the setting of distributed optimization
2. Understand the importance of consensus
3. Introduce and prove convergence of different first-order methods
4. Understand limitations of distributed optimization
5. Discuss future directions

What is Distributed Optimization?

Consider the problem

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

where $f_1, \ldots, f_N$ are L-smooth, G-Lipschitz convex functions.

What is Distributed Optimization?

Consider the problem

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

where $f_1, \ldots, f_N$ are L-smooth, G-Lipschitz convex functions. We can rewrite this as

$$\begin{aligned} \min_{x_1, \ldots, x_N \in \mathbb{R}^d} \quad & \frac{1}{N} \sum_{i=1}^{N} f_i(x_i) \\ \text{s.t.} \quad & x_1 = x_2 = \cdots = x_N \end{aligned}$$

What is Distributed Optimization?

Consider the problem

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

where $f_1, \ldots, f_N$ are L-smooth, G-Lipschitz convex functions. We can rewrite this as

$$\begin{aligned} \min_{x_1, \ldots, x_N \in \mathbb{R}^d} \quad & \frac{1}{N} \sum_{i=1}^{N} f_i(x_i) \\ \text{s.t.} \quad & x_1 = x_2 = \cdots = x_N \end{aligned}$$

The error in satisfying the constraint $x_1 = \cdots = x_N$ is called the **consensus error**.

Applications: Machine Learning

Suppose agents $1, 2, \ldots, N$ have collected sets $D_1, \ldots, D_N$ of labeled data for supervised learning. If we have cost function $L_i(\theta; d)$, which measures the loss on data sample $d \in D_i$, each agent wishes to minimize

$$f_i(\theta) = \sum_{d \in D_i} L_i(\theta; d).$$

If, say due to computational constraints or privacy concerns, each agent is unwilling to share their data to a central server, the model must learn via coordination. We can formulate this in the distributed optimization framework as

$$\begin{aligned} \min_{\theta_1, \ldots, \theta_N \in \mathbb{R}^d} \quad & \frac{1}{N} \sum_{i=1}^{N} f_i(\theta_i) \\ \text{s.t.} \quad & \theta_1 = \cdots = \theta_N \end{aligned}$$

Applications: State Estimation

Consider a moving object whose state has dynamics

$$x(t+1) = A(t)x(t) + w(t),$$

where $w(t)\ \mathcal{N}(0, W)$ is a sequence of i.i.d. noise for some covariance matrix $W \succ 0$. Suppose we have N drones with a given communication network, where drone i observes the object according to

$$y_i(t) = C_i(t)x(t) + v_i(t),$$

where $v_i(t)\ \mathcal{N}(0, W)$ is a sequence of i.i.d. noise for some covariance matrix $V_i \succ 0$.

Applications: State Estimation 2

Suppose the network observes the object from $t = 0$ to $t = T$, and drone i observes the object at times $\mathcal{T}_i \subseteq \{0, \dots, T\}$, and collects the observed data $\{y_i(t) : t \in \mathcal{T}_i\}$. Assuming that $A(t)$, W, and the distribution of the initial state $x(0) \sim \mathcal{N}(\bar{x}(0), P(0))$, drone i knows $C_i(t)$ and V_i, the drones can estimate the trajectory $(x(0), \dots, x(T))$ of the target by solving

$$\min_{\hat{x}_1, \dots, \hat{x}_T \in \mathbb{R}^d} \quad \|\hat{x}(0) - \bar{x}(0)\|^2_{P(0)^{-1}} + \sum_{t=1}^{T} \|\bar{x}(t) - A(t)\bar{x}(t-1)\|^2_{W^{-1}} + \sum_{i=1}^{N} \sum_{t \in \mathcal{T}_i} \|y_i(t) - C_i(t)\hat{x}(t)\|^2_{V_i^{-1}}.$$

Then, let $f_i(\hat{x}_1, \dots, \hat{x}_T) = \|\hat{x}(0) - \bar{x}(0)\|^2_{P(0)^{-1}} + \sum_{t=1}^{T} \|\bar{x}(t) - A(t)\bar{x}(t-1)\|^2_{W^{-1}} + N \sum_{t \in \mathcal{T}_i} \|y_i(t) - C_i(t)\hat{x}(t)\|^2_{V_i^{-1}}$.

Graph Theory Basics 1

Definition

A **graph** is a pair $G = (V, E)$ of sets, called vertices and edges, where $E \subseteq V \times V$.

- In the distributed context, vertex i represents an agent, with its local cost function $f_i \colon \mathbb{R}^d \to \mathbb{R}$ and local variable $x_i \in \mathbb{R}^d$.

Graph Theory Basics 1

Definition

A **graph** is a pair $G = (V, E)$ of sets, called vertices and edges, where $E \subseteq V \times V$.

- In the distributed context, vertex i represents an agent, with its local cost function $f_i \colon \mathbb{R}^d \to \mathbb{R}$ and local variable $x_i \in \mathbb{R}^d$.
- Edge (i, j) implies that there is a communication line between agents i and j, so agents i and j are able to communicate.

Graph Theory Basics 1

Definition

A **graph** is a pair $G = (V, E)$ of sets, called vertices and edges, where $E \subseteq V \times V$.

- In the distributed context, vertex i represents an agent, with its local cost function $f_i \colon \mathbb{R}^d \to \mathbb{R}$ and local variable $x_i \in \mathbb{R}^d$.
- Edge (i, j) implies that there is a communication line between agents i and j, so agents i and j are able to communicate.
- We assume that for any pair of agents i and j, there exists a sequence of edges $(i, i_1), (i_1, i_2), \ldots, (i_k, j)$, starting with i and ending with j.

Graph Theory Basics 1

Definition

A **graph** is a pair $G = (V, E)$ of sets, called vertices and edges, where $E \subseteq V \times V$.

- In the distributed context, vertex i represents an agent, with its local cost function $f_i \colon \mathbb{R}^d \to \mathbb{R}$ and local variable $x_i \in \mathbb{R}^d$.
- Edge (i, j) implies that there is a communication line between agents i and j, so agents i and j are able to communicate.
- We assume that for any pair of agents i and j, there exists a sequence of edges $(i, i_1), (i_1, i_2), \ldots, (i_k, j)$, starting with i and ending with j.
  - This property is known as **connectedness**.

Graph Theory Basics 1

Definition

A **graph** is a pair $G = (V, E)$ of sets, called vertices and edges, where $E \subseteq V \times V$.

- In the distributed context, vertex i represents an agent, with its local cost function $f_i \colon \mathbb{R}^d \to \mathbb{R}$ and local variable $x_i \in \mathbb{R}^d$.
- Edge (i, j) implies that there is a communication line between agents i and j, so agents i and j are able to communicate.
- We assume that for any pair of agents i and j, there exists a sequence of edges $(i, i_1), (i_1, i_2), \ldots, (i_k, j)$, starting with i and ending with j.
  - This property is known as **connectedness**.
- We also assume that if $(i, j) \in E$ then $(j, i) \in E$. This means all communication is bidirectional.

Graph Theory Basics 1

Definition

A **graph** is a pair $G = (V, E)$ of sets, called vertices and edges, where $E \subseteq V \times V$.

- In the distributed context, vertex i represents an agent, with its local cost function $f_i \colon \mathbb{R}^d \to \mathbb{R}$ and local variable $x_i \in \mathbb{R}^d$.
- Edge (i, j) implies that there is a communication line between agents i and j, so agents i and j are able to communicate.
- We assume that for any pair of agents i and j, there exists a sequence of edges $(i, i_1), (i_1, i_2), \ldots, (i_k, j)$, starting with i and ending with j.
  - This property is known as **connectedness**.
- We also assume that if $(i, j) \in E$ then $(j, i) \in E$. This means all communication is bidirectional.
  - A graph with this property is called **undirected**.

Graph Theory Basics 2

- The structure of the graph plays a large role in convergence of distributed algorithms. Highly connected graphs are more favorable, but in practical contexts, maintaining such a communication network can be difficult or expensive.

Graph Theory Basics 2

- The structure of the graph plays a large role in convergence of distributed algorithms. Highly connected graphs are more favorable, but in practical contexts, maintaining such a communication network can be difficult or expensive.

- We will show that as long as the network is connected and undirected, both averaging and first-order methods can achieve consensus, although this will be sensitive to graph structure.

Distributed Averaging 1

Firstly, note that if $\{w_{ij}\}_{i,j=1,\dots,N}$ are nonnegative, then for a convex function f,

$$f\left(\frac{\sum_{j=1}^N w_{ij}x_j}{\sum_{j=1}^N w_{ij}}\right) \leq \frac{\sum_{j=1}^N w_{ij}f(x_j)}{\sum_{j=1}^N w_{ij}}.$$

Thus, an averaging protocol seems to be an appropriate choice for satisfying the consensus constraint $x_1 = \cdots = x_N$.

Distributed Averaging 2

In practice, the communication network is known beforehand and unchangeable. Then, we must ask how to construct the averaging weights w_{ij} in a satisfactory manner and how to use them.

Distributed Averaging 2

In practice, the communication network is known beforehand and unchangeable. Then, we must ask how to construct the averaging weights w_{ij} in a satisfactory manner and how to use them.

Definition

The **distributed averaging** method is defined by

$$x_i(t+1) = \sum_{j=1}^{N} w_{ij} x_j(t), \qquad i = 1, \dots, N.$$

Distributed Averaging 2

In practice, the communication network is known beforehand and unchangeable. Then, we must ask how to construct the averaging weights w_{ij} in a satisfactory manner and how to use them.

Definition

The **distributed averaging** method is defined by

$$x_i(t+1) = \sum_{j=1}^{N} w_{ij} x_j(t), \qquad i = 1, \ldots, N.$$

However, a naive implementation runs some risks.

Distributed Averaging 3

We have some requirements for the method:

1. The weights should be compatible with the topology of the network: if (i, j) is not an edge, then $w_{ij} = 0$.
2. $\sum_{j=1}^{N} w_{ij} = 1$. This ensures that if $x_1(t) = \ldots x_N(t)$, then $x_i(t+1) = x_i(t)$. That is, the mean is a stationary point of the algorithm.
3. $\sum_{i=1}^{N} w_{ij} = 1$. This ensures that the mean is unchanged during the algorithm.

Distributed Averaging 3

We have some requirements for the method:

1. The weights should be compatible with the topology of the network: if (i, j) is not an edge, then $w_{ij} = 0$.
2. $\sum_{j=1}^{N} w_{ij} = 1$. This ensures that if $x_1(t) = \dots x_N(t)$, then $x_i(t+1) = x_i(t)$. That is, the mean is a stationary point of the algorithm.
3. $\sum_{i=1}^{N} w_{ij} = 1$. This ensures that the mean is unchanged during the algorithm.

To see the third point, note that

$$\frac{1}{N}\sum_{i=1}^{N} x_i(t+1) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}x_j(t) = \frac{1}{N}\sum_{j=1}^{N} x_j(t)\sum_{i=1}^{N} w_{ij}$$

$$= \frac{1}{N}\sum_{j=1}^{N} x_j(t).$$

Distributed Averaging 4

We want to analyze the convergence speed of the distributed averaging method. To do this, we need to rearrange the variables into a simpler form. Let

$$X(t) = \begin{bmatrix} x_1(t) & \dots & x_N(t) \end{bmatrix}^\top \in \mathbb{R}^{N \times d},$$

and therefore the iterations can be written as

$$X(t+1) = WX(t).$$

where $W = [w_{ij}]$ is the matrix of weights. We note that $W^\top \mathbf{1} = W\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ is the vector with 1 in every entry.

Distributed Averaging 5

We note that the error can be represented as

$$E(t) = \begin{bmatrix} (x_1(t) - \bar{x})^\top \\ \vdots \\ (x_N(t) - \bar{x})^\top \end{bmatrix} = X(t) - \mathbf{1}\bar{x}^\top = \left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \right) X(t).$$

Distributed Averaging 5

We note that the error can be represented as

$$E(t) = \begin{bmatrix} (x_1(t) - \bar{x})^\top \\ \vdots \\ (x_N(t) - \bar{x})^\top \end{bmatrix} = X(t) - \mathbf{1}\bar{x}^\top = \left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right) X(t).$$

Then,

$$\begin{aligned} E(t+1) = \left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right) X(t+1) &= \left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right) WX(t) \\ &= \left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top W\right) X(t) \\ &= \left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right) X(t) \end{aligned}$$

Distributed Averaging 6

Additionally, note that

$$\left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)\left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right) = W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top - \frac{1}{N}W\mathbf{1}\mathbf{1}^\top + \frac{1}{N^2}\mathbf{1}\mathbf{1}^\top\mathbf{1}\mathbf{1}^\top$$

$$= W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top.$$

Distributed Averaging 6

Additionally, note that

$$\begin{aligned}\left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)\left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right) &= W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top - \frac{1}{N}W\mathbf{1}\mathbf{1}^\top + \frac{1}{N^2}\mathbf{1}\mathbf{1}^\top\mathbf{1}\mathbf{1}^\top \\ &= W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top.\end{aligned}$$

Thus,

$$\begin{aligned}E(t+1) &= \left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)X(t) \\ &= \left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)\left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)X(t) \\ &= \left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)E(t).\end{aligned}$$

Convergence of Distributed Averaging

Thus, if we use the inequality $\|AB\|_F \leq \|A\|_2 \|B\|_F$, we find that

$$\|E(t+1)\|_F^2 \leq \left\| W - \frac{1}{N} \mathbf{1}\mathbf{1}^\top \right\|_2^2 \|E(t)\|_F^2.$$

Convergence of Distributed Averaging

Thus, if we use the inequality $\|AB\|_F \leq \|A\|_2 \|B\|_F$, we find that

$$\|E(t+1)\|_F^2 \leq \left\| W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \right\|_2^2 \|E(t)\|_F^2.$$

Thus, we obtain the following theorem:

Lemma ([1, Theorem 2.1])

Let $\mathcal{G}$ be a connected undirected graph, and we associate it with a nonnegative doubly stochastic matrix W. Suppose $\sigma := \left\| W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \right\|_2 < 1$. Then, the consensus error converges to zero exponentially fast, i.e, we have

$$\frac{1}{N}\sum_{i=1}^{N} \|x_i(t) - \bar{x}\|^2 \leq \sigma^{2t} \cdot \frac{1}{N}\sum_{i=1}^{N} \|x_i(0) - \bar{x}\|^2.$$

Discussion of Distributed Averaging

- The distributed averaging method converges very quickly, and we know that for common choices of σ, $\sigma \leq 1 - \frac{1}{N^2}$, and this bound is tight.
  - This implies that for very large graphs, σ could be very close to 1, resulting in very slow convergence.
- It is possible to speed this up even further, but to my knowledge this only reduces the dependence on σ, not the asymptotic convergence rate.
- We have also not discussed how to construct the weight matrix W, and whether it is always possible to construct a double stochastic matrix W with $\|W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\|_2 < 1$.

Other Useful Notes

Theorem

Suppose W is a doubly stochastic matrix: $W_{ij} \geq 0$, and $W\mathbf{1} = W^\top\mathbf{1} = \mathbf{1}$. Also assume that $W_{ii} > 0$ and $W_{ij} > 0$ if and only if i and j are adjacent in the graph G. Then,

$$\left\| W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \right\|_2 < 1$$

The proof of this uses the Perron-Frobenius theorem, which is an important result from graph theory, but we will not cover it here.

Decentralized Computation of Weights

The previous theorem implies that each agent can choose its own weights without knowing anything about the total structure of the graph. One such choice is known as the **Metropolis weights**,

$$w_{ij} = \begin{cases} \frac{1}{\max\{\deg(i),\deg(j)\}+\varepsilon}, & (i,j) \in E \\ 0, & (i,j) \notin E \\ 1 - \sum_{k \neq i} W_{ik}, & i = j, \end{cases}$$

where $\deg(i)$ is the degree of vertex i, the number of neighbors it has and $\varepsilon > 0$ is any positive real number.

Constructing the Optimal Weight Matrix

We want to solve the problem

$$\begin{aligned} \min_{W \in \mathbb{R}^{N \times N}} \quad & \left\| W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \right\|_2 \\ \text{s.t.} \quad & W_{ij} = 0, \quad (i,j) \notin E \text{ and } i \neq j \\ & W\mathbf{1} = W^\top \mathbf{1} = \mathbf{1}. \end{aligned}$$

This is a convex problem, but in general it is nonsmooth. We need to transform this into a form that is easier to solve.

Reformulating the Weight Matrix Problem

To begin, note that

$$\begin{aligned}\left\|\frac{W+W^\top}{2}-\frac{1}{N}\mathbf{1}\mathbf{1}^\top\right\|_2 &= \left\|\frac{W+\frac{1}{N}\mathbf{1}\mathbf{1}^\top}{2}+\left(\frac{W+\frac{1}{N}\mathbf{1}\mathbf{1}^\top}{2}\right)^\top\right\|_2 \\ &\leq \frac{1}{2}\left\|W-\frac{1}{N}\mathbf{1}\mathbf{1}^\top\right\|_2+\frac{1}{2}\left\|\left(W-\frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)^\top\right\|_2 \\ &= \left\|W-\frac{1}{N}\mathbf{1}\mathbf{1}^\top\right\|_2 .\end{aligned}$$

Moreover, due to the undirectedness of the graph G, both W and $W^\top$ are compatible with the graph, so without loss of generality we can assume W to be symmetric.

Semidefinite Representation

Thus, we can reformulate the original problem as

$$\begin{aligned}
\min_{\substack{W\in\mathbb{R}^{N\times N}\\ s\in\mathbb{R}}} \quad & s \\
\text{s.t.} \quad & W_{ij} = 0, \quad (i,j) \notin E \text{ and } i \neq j \\
& W\mathbf{1} = W^\top \mathbf{1} = \mathbf{1} \\
& -sI \preceq W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \preceq sI
\end{aligned}$$

Semidefinite Representation

Thus, we can reformulate the original problem as

$$\begin{aligned} \min_{\substack{W \in \mathbb{R}^{N \times N} \\ s \in \mathbb{R}}} \quad & s \\ \text{s.t.} \quad & W_{ij} = 0, \quad (i,j) \notin E \text{ and } i \neq j \\ & W\mathbf{1} = W^\top \mathbf{1} = \mathbf{1} \\ & -sI \preceq W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \preceq sI \end{aligned}$$

In the case where we have a directed graph, we cannot assume W to be symmetric. Instead, we can use Schur complement to formulate the last constraint as $\begin{bmatrix} sI & W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \\ W^\top - \frac{1}{N}\mathbf{1}\mathbf{1}^\top & sI \end{bmatrix} \succeq 0$ instead.

Naive Distributed Gradient Descent

We can now propose our first distributed algorithm. The proposed algorithm is simple:

$$
\begin{aligned}
y_i(t+1,0) &= x_i(t) - \eta \nabla f_i(x_i(t)) \\
y_i(t+1,k+1) &= \sum_{j=1}^{N} w_{ij} y_i(t+1,k), \qquad k = 1, \ldots, K_{t+1} \\
x_i(t+1) &= y_i(t+1, K_{t+1}).
\end{aligned}
$$

Naive Distributed Gradient Descent

We can now propose our first distributed algorithm. The proposed algorithm is simple:

$$
\begin{aligned}
y_i(t+1,0) &= x_i(t) - \eta \nabla f_i(x_i(t)) \\
y_i(t+1,k+1) &= \sum_{j=1}^{N} w_{ij} y_i(t+1,k), \qquad k = 1, \dots, K_{t+1} \\
x_i(t+1) &= y_i(t+1, K_{t+1}).
\end{aligned}
$$

In essence, we take a gradient step, then run the consensus to bring the iterates close together again. The goal is to ensure that $\|\nabla f_i(x_i(t)) - \nabla f_i(\bar{x}(t))\|$ is very small.

First Observations

Define

$$\varepsilon(t) = \frac{1}{N}\sum_{i=1}^{N}(\nabla f_i(x_i(t)) - \nabla f_i(\bar{x}(t))).$$

We find that we have two key relations:

$$\bar{x}(t+1) = \bar{x}(t) - \eta(\nabla f(\bar{x}(t)) + \varepsilon(t)),$$

$$E(t+1) = \left(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)^{K_{t+1}}\left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top\right)\begin{bmatrix}(x_1(t) - \eta\nabla f_1(x_1(t)))^\top \\ \vdots \\ (x_N(t) - \eta\nabla f_N(x_N(t)))^\top\end{bmatrix}$$

Proof of Convergence 2

Also, if each f_i is L-smooth, then

$$\begin{aligned}\|\varepsilon(t)\|^2 &\leq \frac{1}{N}\sum_{i=1}^{N}\|\nabla f_i(x_i(t)) - \nabla f_i(\bar{x}(t))\|^2\\ &\leq \frac{1}{N}\sum_{i=1}^{N} L^2\|x_i(t) - \bar{x}(t)\|^2\\ &= \frac{L^2}{N}\|E(t)\|_F^2.\end{aligned}$$

Proof of Convergence 3

Now, we need to relate $E(t+1)$ to $E(t)$, and we first note

$$\begin{aligned}
&\left\| \nabla f_i(x_i(t)) - \frac{1}{N}\sum_{j=1}^{N} \nabla f_j(x_j(t)) \right\| \\
&\leq \|\nabla f_i(x_i(t)) - \nabla f_i(\bar{x}(t))\| + \|\nabla f_i(\bar{x}(t)) - \nabla f(\bar{x}(t))\| \\
&\qquad + \left\| \frac{1}{N}\sum_{j=1}^{N} \nabla f_j(\bar{x}(t)) - \frac{1}{N}\sum_{j=1}^{N} \nabla f_j(x_j(t)) \right\| \\
&\leq L\|x_i(t) - \bar{x}(t)\| + 2G + \frac{1}{N}\sum_{j=1}^{N} L\|x_j(t) - \bar{x}(t)\|
\end{aligned}$$

Proof of Convergence 4

Then,

$$
\left\| \left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top \right) \begin{bmatrix} (x_1(t) - \eta \nabla f_1(x_1(t)))^\top \\ \vdots \\ (x_N(t) - \eta \nabla f_N(x_N(t)))^\top \end{bmatrix} \right\|_F^2
$$

$$
= \sum_{i=1}^{N} \left\| x_i(t) - \bar{x}(t) - \eta \left(\nabla f_i(x_i(t)) - \frac{1}{N} \sum_{j=1}^{N} \nabla f_j(x_j(t)) \right) \right\|^2
$$

$$
\leq \sum_{i=1}^{N} \left((1 + \eta L)\|x_i(t) - \bar{x}(t)\| + 2\eta G + \frac{\eta L}{N} \sum_{j=1}^{N} \|x_j(t) - \bar{x}(t)\| \right)^2
$$

Proof of Convergence 5

$$\sum_{i=1}^{N}\left((1+\eta L)\|x_i(t)-\bar{x}(t)\|+2\eta G+\frac{\eta L}{N}\sum_{j=1}^{N}\|x_j(t)-\bar{x}(t)\|\right)^2$$

$$=\left\|\left((1+\eta L)I+\frac{\eta L}{N}\mathbf{1}\mathbf{1}^\top\right)\begin{bmatrix}\|x_1(t)-\bar{x}(t)\|\\ \vdots\\ \|x_N(t)-\bar{x}(t)\|\end{bmatrix}+2\eta G\mathbf{1}\right\|^2$$

$$\leq\left(\left\|(1+\eta L)I+\frac{\eta L}{N}\mathbf{1}\mathbf{1}^\top\right\|_2\left\|\begin{bmatrix}\|x_1(t)-\bar{x}(t)\|\\ \vdots\\ \|x_N(t)-\bar{x}(t)\|\end{bmatrix}\right\|+2\eta\sqrt{N}G\right)^2$$

$$\leq((1+2\eta L)\|E(t)\|_F+2\eta\sqrt{N}G)^2.$$

Proof of Convergence 6

Thus,

$$\|E(t+1)\|_F \leq \sigma^{K_{t+1}}(1+2\eta L)\|E(t)\|_F + 2\eta\sqrt{N}G)^2.$$

This gives the following result:

Theorem

Consider the problem $\min_{x\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} f_i(x)$*, where* $f_1,\dots,f_N$ *are* L*-smooth,* G*-Lipschitz, convex functions.*
For the naive distributed gradient descent algorithm, with $\eta \in (0, \frac{1}{L}]$ *and* $K_t \geq \left\lceil \frac{\ln(4(t+1)^2+12)}{-\ln\sigma} \right\rceil$*, we have*

$$\|E(t)\|_F \leq \frac{2N\eta G}{(t+1)^2}$$

Proof of Convergence 7

To analyze the function values, we will use a result that simplifies lengthy analysis:

Theorem

Let $f : \mathbb{R}^d \to \mathbb{R}$ be convex and L-smooth. Consider the iterations

$$x(t+1) = x(t) - \eta(\nabla f(x(t)) + e(t)),$$

where $\eta \in (0, \frac{1}{L}]$. Then,

$$f\left(\frac{1}{t}\sum_{\tau=1}^{t} x(\tau)\right) - f(x^\star) \leq \frac{1}{2\eta t}\left(\|x(0) - x^\star\| + \sum_{\tau=0}^{t-1} \eta\|e(\tau)\|\right)^2.$$

Proof of Convergence 8

To show that this results in convergence, note that

$$\sum_{\tau=0}^{t-1} \eta \|\varepsilon(t)\| \leq \sum_{\tau=0}^{t-1} \frac{\eta L}{N} \|E(t)\|_F \leq 2\eta^2 LG \sum_{\tau=0}^{t-1} \frac{1}{(\tau+1)^2} \leq 4\eta^2 LG.$$

Proof of Convergence 8

To show that this results in convergence, note that

$$\sum_{\tau=0}^{t-1} \eta \|\varepsilon(t)\| \leq \sum_{\tau=0}^{t-1} \frac{\eta L}{N} \|E(t)\|_F \leq 2\eta^2 LG \sum_{\tau=0}^{t-1} \frac{1}{(\tau+1)^2} \leq 4\eta^2 LG.$$

Thus,

$$f\left(\frac{1}{t}\sum_{\tau=1}^{t} x(\tau)\right) - f(x^\star) \leq \frac{1}{2\eta t}\left(\|x(0) - x^\star\| + 4\eta^2 LG\right)^2.$$

Discussion

- In this algorithm, agent i must do $K_t \cdot \deg(i)$ communications per iteration, in addition to a gradient step.

Discussion

- In this algorithm, agent i must do $K_t \cdot \deg(i)$ communications per iteration, in addition to a gradient step.
- Although we recover the $O(\frac{1}{t})$ convergence rate of centralized gradient descent, the communication cost could be prohibitively high if the graph is unfavorable.

Discussion

- In this algorithm, agent i must do $K_t \cdot \deg(i)$ communications per iteration, in addition to a gradient step.
- Although we recover the $O(\frac{1}{t})$ convergence rate of centralized gradient descent, the communication cost could be prohibitively high if the graph is unfavorable.
- Ideally, we would like to bypass the need for excessive communication between agents and instead do more gradient steps, since that is the main objective

Discussion

- In this algorithm, agent i must do $K_t \cdot \deg(i)$ communications per iteration, in addition to a gradient step.
- Although we recover the $O(\frac{1}{t})$ convergence rate of centralized gradient descent, the communication cost could be prohibitively high if the graph is unfavorable.
- Ideally, we would like to bypass the need for excessive communication between agents and instead do more gradient steps, since that is the main objective
  - Slower convergence of consensus error is acceptable, because we can run the consensus algorithm after the iterations, and this converges extremely quickly.

Conclusion

- We have introduced the setting of distributed optimization, and seen situations where such a framework is useful.
- We also discussed the feature of consensus error, which distinguishes distributed optimization from its centralized counterpart
- We proved the convergence of one naive algorithm for distributed optimization, as well as discussed its limitations.
- There are much more advanced and preferable algorithms, some of which incorporate addition internal dynamics to offset the negative effects of consensus error on the gradient updates.
- A key takeaway is that distributed first-order algorithms are theoretically similar to inexact first-order methods, where controlling the inexactness is needed to ensuring convergence.

References I

[1] Yujie Tang. Fundamentals of distributed optimization, 2024. URL https://tyj518.github.io/files/Notes_on_Distributed_Optimization.pdf.