Distributed and Decentralized Optimization

Ernest K. Ryu and Wotao Yin

Large-Scale Convex Optimization via Monotone Operators

Distributed optimization

Distributed optimization uses a set of networked computers, called agents, to solve optimization problems.

Motivation: When an algorithm running on one computer does not meet the required performance, one can

1. upgrade the computer (CPU, memory) and run the same algorithm;
2. use more computers, decompose the problem, and run a distributed optimization algorithm.

Distributed optimization

Distributed optimization uses a set of networked computers, called agents, to solve optimization problems.

Motivation: When an algorithm running on one computer does not meet the required performance, one can

1. upgrade the computer (CPU, memory) and run the same algorithm;
2. use more computers, decompose the problem, and run a distributed optimization algorithm.

Approach 1 is less cost effective and may never reach the required performance.

Approach 2 is the more favorable (often the only) choice of solving extremely large optimization problems.

Decentralized optimization

We distinguish between *distributed methods* and *decentralized methods*.

Decentralized optimization

We distinguish between *distributed methods* and *decentralized methods*.

Distributed methods perform computation over a network (a broader class).

Decentralized methods do so without central coordination (a subclass).

Decentralized optimization

We distinguish between *distributed methods* and *decentralized methods*.

Distributed methods perform computation over a network (a broader class).

Decentralized methods do so without central coordination (a subclass).

Roughly speaking, when communication latency and bandwidth cost much more than computation, decentralized methods are preferred.

Examples: drone fleet control, wireless sensor network, applications of real-time decisions made based on agents' local data

Problem and setup

In this lecture, we solve

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{n}(f_i(x)+h_i(x)), \tag{1}$$

where $f_1,\ldots,f_n$ are CCP (and proximable) and $h_1,\ldots,h_n$ are CCP and differentiable.

Problem and setup

In this lecture, we solve

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{n} (f_i(x) + h_i(x)), \tag{1}$$

where $f_1, \ldots, f_n$ are CCP (and proximable) and $h_1, \ldots, h_n$ are CCP and differentiable.

Setup: agents $i = 1, \ldots, n$ each perform local computation with f_i and h_i and communicate over a network to find the (shared) solution $x^\star$.

Problem and setup

In this lecture, we solve

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{n} (f_i(x) + h_i(x)), \tag{1}$$

where $f_1, \ldots, f_n$ are CCP (and proximable) and $h_1, \ldots, h_n$ are CCP and differentiable.

Setup: agents $i = 1, \ldots, n$ each perform local computation with f_i and h_i and communicate over a network to find the (shared) solution $x^\star$.

The textbook chapter considers the more general formulation

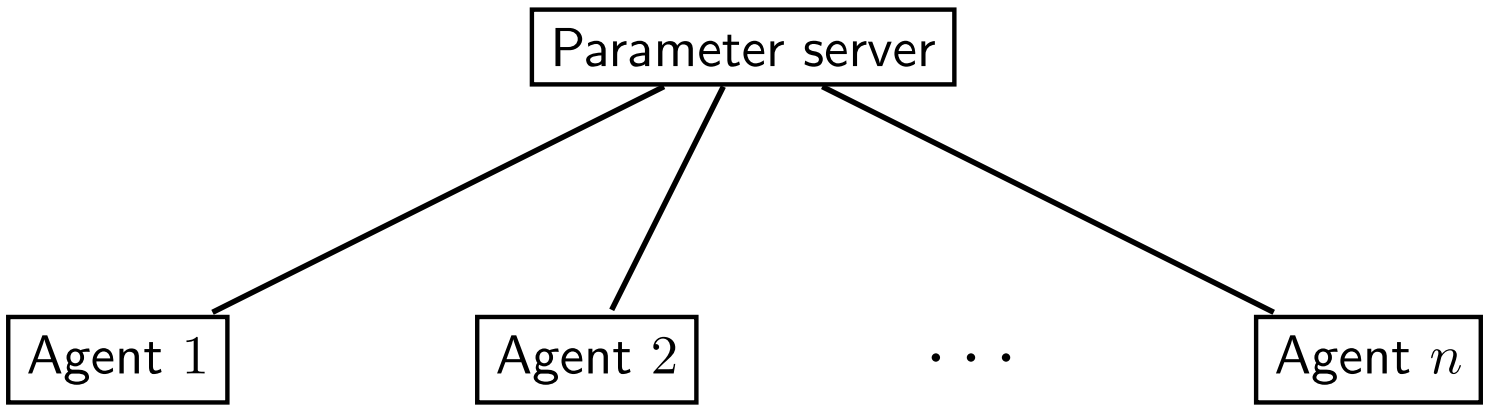
$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \quad r(x) + \sum_{i=1}^{n} (f_i(x) + h_i(x)), \tag{2}$$

where r is CCP and proximable.

Outline

Centralized consensus

Consider a parameter-server network model with a centralized agent coordinating with n individual agents.



We study distributed methods based on the consensus technique.

Distributed gradient method

Consider

$$\underset{x \in \mathbb{R}^p}{\text{minimize}} \quad \frac{1}{n} \sum_{i=1}^{n} h_i(x),$$

where $h_1, \ldots, h_n$ are differentiable. With consensus set $C = \{(x_1, \ldots, x_n) \,|\, x_1 = \cdots = x_n\}$, obtain the equivalent problem

$$\begin{array}{ll} \underset{x_1, \ldots, x_n \in \mathbb{R}^p}{\text{minimize}} & \frac{1}{n} \sum_{i=1}^{n} h_i(x_i) \\ \text{subject to} & (x_1, \ldots, x_n) \in C. \end{array}$$

Distributed gradient method

Consider

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n} h_i(x),$$

where $h_1, \ldots, h_n$ are differentiable. With consensus set $C = \{(x_1, \ldots, x_n) \,|\, x_1 = \cdots = x_n\}$, obtain the equivalent problem

$$\begin{array}{ll} \underset{x_1,\ldots,x_n\in\mathbb{R}^p}{\text{minimize}} & \frac{1}{n}\sum_{i=1}^{n} h_i(x_i) \\ \text{subject to} & (x_1, \ldots, x_n) \in C. \end{array}$$

FBS is:

$$x_i^{k+1/2} = x^k - \alpha\nabla h_i(x^k)$$

$$x^{k+1} = \frac{1}{n}\sum_{i=1}^{n} x_i^{k+1/2}$$

Distributed gradient method

Equivalent to:

$$\bar{g}^k = \frac{1}{n}\sum_{i=1}^{n} \nabla h_i(x^k)$$
$$x^{k+1} = x^k - \alpha \bar{g}^k$$

This is the distributed gradient method. Assume a solution exists, $h_1, \ldots, h_n$ are L_h-smooth, and $\alpha \in (0, 2/L_h)$. Then $x^k \to x^\star$.
(When $h_1, \ldots, h_n$ not differentiable, can use subgradient method of §7.)

Distributed gradient method

Equivalent to:

$$\bar{g}^k = \frac{1}{n}\sum_{i=1}^{n} \nabla h_i(x^k)$$
$$x^{k+1} = x^k - \alpha \bar{g}^k$$

This is the distributed gradient method. Assume a solution exists, $h_1, \ldots, h_n$ are L_h-smooth, and $\alpha \in (0, 2/L_h)$. Then $x^k \to x^\star$.
(When $h_1, \ldots, h_n$ not differentiable, can use subgradient method of §7.)

This method is (centralized) distributed:

(i) Each agent independently computes $\nabla h_i(x^k)$

(ii) Agents coordinate to compute $\bar{g}^k$ (reduction operation) and the central agent computes and broadcasts x^{k+1} to all individual agents.

Distributed ADMM

Consider

$$\underset{x\in\mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{n} f_i(x).$$

With the consensus technique, obtain the equivalent problem:

$$\begin{aligned} \underset{\substack{x_1,\ldots,x_n\in\mathbb{R}^p \\ y\in\mathbb{R}^p}}{\text{minimize}} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{subject to} \quad & x_i = y \qquad \text{for } i = 1,\ldots,n. \end{aligned}$$

Rewrite to fit ADMM's form:

$$\begin{aligned} \underset{\substack{x_1,\ldots,x_n\in\mathbb{R}^p \\ y\in\mathbb{R}^p}}{\text{minimize}} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{subject to} \quad & \begin{bmatrix} I & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & 0 & \cdots & I \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} -I \\ \vdots \\ -I \end{bmatrix} y = 0. \end{aligned}$$

Apply ADMM:

$$x_i^{k+1} = \operatorname*{argmin}_{x_i \in \mathbb{R}^p} \left\{ f_i(x_i) + \langle u_i^k, x_i - y^k \rangle + \frac{\alpha}{2} \|x_i - y^k\|^2 \right\}$$
$$y^{k+1} = \frac{1}{n} \sum_{i=1}^n \left(x_i^{k+1} + \frac{1}{\alpha} u_i^k \right)$$
$$u_i^{k+1} = u_i^k + \alpha(x_i^{k+1} - y^{k+1}).$$

Simplify the iteration by noting that $u_1^k, \ldots, u_n^k$ has mean 0 after the initial iteration and eliminating y^k:

$$x_i^{k+1} = \mathrm{Prox}_{(1/\alpha) f_i} \left(\bar{x}^k - (1/\alpha) u_i^k \right)$$
$$u_i^{k+1} = u_i^k + \alpha(x_i^{k+1} - \bar{x}^{k+1})$$

for $i = 1, \ldots, n$, where $\bar{x}^k = (1/n)(x_1^k + \cdots + x_n^k)$. This is distributed (centralized) ADMM. Convergence follows from convergence of ADMM.

Distributed ADMM

$$x_i^{k+1} = \mathrm{Prox}_{(1/\alpha)f_i}\left(\bar{x}^k - (1/\alpha)u_i^k\right)$$
$$u_i^{k+1} = u_i^k + \alpha(x_i^{k+1} - \bar{x}^{k+1})$$

is distributed:

(i) each agent independently performs the u^k- and x_i^{k+1}-updates with local computation

(ii) agents coordinate to compute $\bar{x}^{k+1}$ with a reduction.

Distributed ADMM

$$x_i^{k+1} = \mathrm{Prox}_{(1/\alpha)f_i}\left(\bar{x}^k - (1/\alpha)u_i^k\right)$$
$$u_i^{k+1} = u_i^k + \alpha(x_i^{k+1} - \bar{x}^{k+1})$$

is distributed:

(i) each agent independently performs the u^k- and x_i^{k+1}-updates with local computation

(ii) agents coordinate to compute $\bar{x}^{k+1}$ with a reduction.

Exercise 11.7: Obtain distributed ADMM by applying DRS to the equivalent problem

$$\begin{array}{ll} \underset{x_1,\ldots,x_n\in\mathbb{R}^p}{\text{minimize}} & \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} h_i(x_i) \\ \text{subject to} & (x_1,\ldots,x_n) \in C. \end{array}$$

Primal decomposition

Consider

$$\underset{\substack{x_1,\ldots,x_n\in\mathbb{R}^p\\ z\in\mathbb{R}^q}}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n} f_i(x_i,z).$$

With $\phi_i(z) = \inf_x f_i(x,z)$, problem is equivalent to

$$\underset{z\in\mathbb{R}^q}{\text{minimize}} \quad \sum_{i=1}^{n} \phi_i(z).$$

Primal decomposition

Consider

$$\underset{\substack{x_1,\ldots,x_n\in\mathbb{R}^p\\ z\in\mathbb{R}^q}}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n} f_i(x_i,z).$$

With $\phi_i(z) = \inf_x f_i(x,z)$, problem is equivalent to

$$\underset{z\in\mathbb{R}^q}{\text{minimize}} \quad \sum_{i=1}^{n} \phi_i(z).$$

If $\phi_1,\ldots,\phi_n$ are differentiable, use distributed gradient method

$$g_i^k \in \nabla\phi_i(z^k)$$

$$z^{k+1} = z^k - \frac{\alpha}{n}\sum_{i=1}^{n} g_i^k$$

See Exercise 11.2 for computing subgradients of $\phi_1,\ldots,\phi_n$. When $\phi_1,\ldots,\phi_n$ not differentiable, can use subgradient method of §7.

Dual decomposition

Consider the equivalent problem

$$\begin{array}{ll} \underset{\substack{x_1,\ldots,x_n\in\mathbb{R}^p \\ z_1,\ldots,z_n\in\mathbb{R}^q \\ y\in\mathbb{R}^q}}{\text{minimize}} & \sum_{i=1}^n f_i(x_i,z_i) \\ \text{subject to} & z_i = y, \end{array}$$

generated by the Lagrangian

$$\mathbf{L}(x,y,z,v) = \sum_{i=1}^n f_i(x_i,z_i) - \langle v_i, z_i - y\rangle.$$

The dual problem is

$$\begin{array}{ll} \underset{v_1,\ldots,v_n \in \mathbb{R}^q}{\text{maximize}} & -\sum_{i=1}^n \psi_i(v_i) \\ \text{subject to} & v_1 + \cdots + v_n = 0, \end{array}$$

with

$$\psi_i(v_i) = \sup_{\substack{x_i \in \mathbb{R}^p \\ z_i \in \mathbb{R}^q}} \{-f_i(x_i, z_i) + \langle v_i, z_i \rangle\} = f_i^*(0, v_i).$$

When $\psi_1, \ldots \psi_n$ are differentiable, use projected gradient in a distributed manner

$$\begin{aligned} g_i^k &\in \nabla \psi_i(v_i^k) \\ v_i^{k+1} &= v_i^k - \alpha(g_i^k - \bar{g}^k) \end{aligned}$$

where $\bar{g}^k = (1/n)(g_1^k + \cdots + g_n^k)$. See Exercise 11.3 for computing subgradients of $\psi_1, \ldots, \psi_n$. (When $\psi_1, \ldots, \psi_n$ not differentiable, can use projected subgradient method of §7.)

Outline

Note on the word "graph"

"Graph" has two distinct meanings in mathematics.

The first meaning, as in "we plot the graph $\sin(x)$ on a graphing calculator", concerns the relationship between the inputs and outputs of a function. The *graph* of an operator, which we denote as $\operatorname{Gra} \mathbb{A}$, and the scaled relative *graph* uses this first meaning.

Here, we consider the second meaning, the use in discrete mathematics for representing networks.

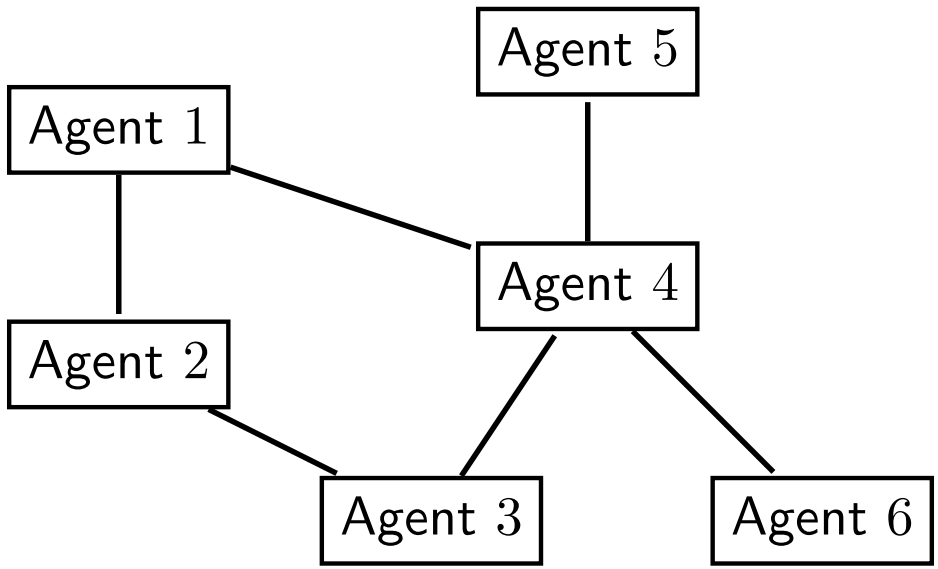
Networks and graphs

A graph $G = (V, E)$ represents a network. V is set of nodes and E is set of edges. Assume

- Network is finite and with nodes 1 through n, i.e., $V = \{1, \ldots, n\}$.
- Graph is undirected, i.e., an edge $\{i, j\} \in E$ is an unordered pair of distinct nodes i and j.
- Graph has no self-loop, i.e., $\{i, i\} \notin E$ for all $i \in V$.
- Graph is connected, i.e., for any $i, j \in V$ such that $i \neq j$, there is a sequence of edges

$$\{i, v_1\}, \{v_1, v_2\}, \ldots, \{v_{k-1}, v_k\}, \{v_k, j\} \in E.$$

With graphs, we can represent networks without a central coordinating agent. The following graph has $V = \{1, 2, 3, 4, 5, 6\}$ and $E = \{\{1, 2\}, \{1, 4\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{4, 6\}\}$.



A node represents a computational agent that stores data and performs computation, and an edge $\{i, j\}$ represents a direct connection between i and j through which agents i and j can communicate.

If $\{i, j\} \in E$, then we say j is adjacent to i and that j is a neighbor of i (and vice-versa). Write

$$N_i = \{j \in V \,|\, \{i, j\} \in E\}$$

for the set of neighbors i and $|N_i|$ for the number of neighbors of i.

Using the notation of graphs, we can recast problem (1) into

$$\begin{array}{ll} \underset{\{x_i\}_{i \in V} \subset \mathbb{R}^p}{\text{minimize}} & \displaystyle\sum_{i \in V} \left(f_i(x_i) + h_i(x_i)\right) \\ \text{subject to} & x_i = x_j \quad \forall\, \{i, j\} \in E. \end{array} \tag{3}$$

Why decentralized optimization?

In a connected network, all agents can communicate with each other. Any optimization method can be executed over the network through relayed communication over multiple edges.

However, in distributed optimization, communication tends to be the bottleneck. So we consider algorithms that communicate across single edges

- without directly relying on long-range relayed communication,
- without creating a bottleneck by communicating with a single central node.

Not delegating any agent as the central agent also improves reliability against agent failure and helps data privacy.

Decentralized ADMM

Consider $h_1 = \cdots = h_n = 0$. For $e = \{i, j\}$, replace the constraint $x_i = x_j$ with $x_i = y_e$ and $x_j = y_e$ to obtain the equivalent problem

$$\begin{aligned} &\underset{\substack{\{x_i\}_{i\in V}\\ \{y_e\}_{e\in E}}}{\text{minimize}} && \sum_{i\in V} f_i(x_i) \\ &\text{subject to} && \begin{cases} x_i - y_e = 0 \\ x_j - y_e = 0 \end{cases} \qquad \forall\ e = \{i, j\} \in E. \end{aligned}$$

For each $e = \{i, j\} \in E$, introduce the dual variables $u_{e,i}$ for $x_i - y_e = 0$ and $u_{e,j}$ for $x_j - y_e = 0$. The augmented Lagrangian is

$$\begin{aligned} \mathbf{L}_\alpha(x, y, u) = \sum_i f_i(x_i) + \sum_{e=\{i,j\}} \left(\langle u_{e,i}, x_i - y_e \rangle + \langle u_{e,j}, x_j - y_e \rangle \right) \\ + \sum_{e=\{i,j\}} \frac{\alpha}{2} \left(\|x_i - y_e\|^2 + \|x_j - y_e\|^2 \right). \end{aligned}$$

Apply ADMM and obtain

$$x_i^{k+1} = \underset{x_i \in \mathbb{R}^p}{\operatorname{argmin}} \left\{ f_i(x_i) + \sum_{j \in N_i} \left(\langle u_{\{i,j\},i}^k, x_i - y_{\{i,j\}}^k \rangle + \frac{\alpha}{2} \|x_i - y_{\{i,j\}}^k\|^2 \right) \right\} \quad \forall i \in V$$

$$y_e^{k+1} = \underset{y_e \in \mathbb{R}^p}{\operatorname{argmin}} \left\{ \sum_{t=i,j} \left(\langle u_{e,t}^k, x_t^{k+1} - y_e \rangle + \frac{\alpha}{2} \|x_t^{k+1} - y_e\|^2 \right) \right\} \quad \forall e = \{i,j\} \in E$$

$$u_{e,t}^{k+1} = u_{e,t}^k + \alpha(x_t^{k+1} - y_e^{k+1}) \qquad \forall e = \{i,j\} \in E,\ t = i,j.$$

We simplify further.

Substitute $y_e^{k+1} = \frac{1}{2}\sum_{t=i,j}(x_t^{k+1} + \frac{1}{\alpha}u_{e,t}^k)$:

$$u_{e,i}^{k+1} = u_{e,i}^k + \alpha\left(x_i^{k+1} - \frac{1}{2}\sum_{t=i,j}\left(x_t^{k+1} + \frac{1}{\alpha}u_{e,t}^k\right)\right)$$
$$= \frac{1}{2}(u_{e,i}^k - u_{e,j}^k) + \frac{\alpha}{2}(x_i^{k+1} - x_j^{k+1}), \quad \forall e = \{i,j\} \in E.$$

Using $u_{e,i}^k + u_{e,j}^k = 0$ for all $e = \{i,j\}$ and $k = 1, 2, \ldots$, write $y_e^k = \frac{1}{2}(x_i^k + x_j^k)$, $u_{e,i}^{k+1} = u_{e,i}^k + \frac{\alpha}{2}(x_i^{k+1} - x_j^{k+1})$, and

$$x_i^{k+1} = \operatorname*{argmin}_{x_i \in \mathbb{R}^p}\left\{f_i(x_i) + \frac{\alpha}{2}\sum_{j\in N_i}\left\|x_i - \frac{1}{2}(x_i^k + x_j^k) + \frac{1}{\alpha}u_{\{i,j\},i}^k\right\|^2\right\}$$
$$= \operatorname*{argmin}_{x_i \in \mathbb{R}^p}\left\{f_i(x_i) + \frac{\alpha|N_i|}{2}\left\|x_i - \frac{1}{|N_i|}\sum_{j\in N_i}\left(\frac{1}{2}(x_i^k + x_j^k) - \frac{1}{\alpha}u_{\{i,j\},i}^k\right)\right\|^2\right\}$$

for all $i \in V$.

Defining $v_i^k = \frac{1}{|N_i|} \sum_{j \in N_i} \left(\frac{1}{2}(x_i^k + x_j^k) - \frac{1}{\alpha} u_{\{i,j\},i}^k \right)$ and $a_i^k = \frac{1}{|N_i|} \sum_{j \in N_i} x_j^k$ and obtain: for every $i \in V$

$$
\begin{aligned}
x_i^{k+1} &= \mathrm{Prox}_{(\alpha |N_i|)^{-1} f_i(x_i)}(v_i^k) \\
a_i^{k+1} &= \frac{1}{|N_i|} \sum_{j \in N_i} x_j^{k+1} \\
v_i^{k+1} &= v_i^k + a_i^{k+1} - \frac{1}{2} a_i^k - \frac{1}{2} x_i^k
\end{aligned}
$$

for $i \in V$. This is *decentralized ADMM*. Convergence follows from convergence of ADMM.

Decentralized ADMM

$$x_i^{k+1} = \mathrm{Prox}_{(\alpha|N_i|)^{-1} f_i(x_i)}(v_i^k)$$
$$a_i^{k+1} = \frac{1}{|N_i|} \sum_{j \in N_i} x_j^{k+1}$$
$$v_i^{k+1} = v_i^k + a_i^{k+1} - \frac{1}{2} a_i^k - \frac{1}{2} x_i^k$$

is decentralized:

(i) Each agent independently performs the x^{k+1}- and v^{k+1}-updates with local computation.

(ii) Agents send x_i^{k+1} to its neighbors and each agent computes a_i^{k+1} by averaging the x_j^{k+1}'s received from its neighbors (reduction operation in the neighborhood).

Synchronization

The above decentralized methods are synchronous, which can be an unrealistic requirement.

One can use asynchronous decentralized methods, which combine the asynchrony of §6 with the methods of this section.

Outline

Distributed optimization with centralized consensus

Decentralized optimization with graph consensus

Decentralized optimization with mixing matrices

Decentralized notation

Define stack operator and use boldface to denote stacked variables:

$$\mathbf{x} = \mathrm{stack}(x_1, \ldots, x_n) = \begin{bmatrix} — & x_1^\intercal & — \\ & \vdots & \\ — & x_n^\intercal & — \end{bmatrix} \in \mathbb{R}^{n \times p}.$$

Write $x^\star \in \mathbb{R}^p$ and $\mathbf{x}^\star = \mathrm{stack}(x^\star, \ldots, x^\star) \in \mathbb{R}^{n \times p}$ for the solution.
For $\mathbf{x} = \mathrm{stack}(x_1, \ldots, x_n)$ and $\mathbf{y} = \mathrm{stack}(y_1, \ldots, y_n)$, define

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^n \langle x_i, y_i \rangle.$$

For $A \succeq 0$, define $\|\mathbf{x}\|_A^2 = \langle \mathbf{x}, A\mathbf{x} \rangle$. Specifically, $\|\mathbf{x}\|^2 = \|\mathbf{x}\|_I^2 = \langle \mathbf{x}, \mathbf{x} \rangle$.

Define

$$f(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i), \qquad h(\mathbf{x}) = \sum_{i=1}^{n} h_i(x_i)$$

$$\mathrm{Prox}_{\alpha f}(\mathbf{x}) = \mathrm{stack}(\mathrm{Prox}_{\alpha f_1}(x_1), \ldots, \mathrm{Prox}_{\alpha f_n}(x_n))$$

$$\nabla h(\mathbf{x}) = \mathrm{stack}(\nabla h_1(x_1), \ldots, \nabla h_n(x_n)).$$

We say $\mathbf{x} = \mathrm{stack}(x_1, \ldots, x_n)$ is *in consensus* if $x_1 = \cdots = x_n$.
Any feasible point of (3) is in consensus. The methods of this section produce iterates that are in consensus in the limit.

Mixing matrices

Informally, $W \in \mathbb{R}^{n \times n}$ is a mixing matrix when an application of W represents a round of communication and the aggregation of the communicated information. Write $\lambda_1, \ldots, \lambda_n$ for the eigenvalues of W.

W is a decentralized mixing matrix with respect to $G = (V, E)$ if $W_{ij} = 0$ when $i \neq j$ and $\{i, j\} \notin E$. (W_{ii} may be nonzero. W_{ij} may be nonzero only if i and j are directly linked.)

Wy can be evaluated in a decentralized manner if W is decentralized

$$(Wy)_i = \sum_{i=1}^{n} W_{ij} y_j = \sum_{j \in N_i \cup \{i\}} W_{ij} y_j.$$

Example: Local averaging matrix

With mixing matrix

$$W_{i,j} = \begin{cases} \frac{1}{|Ni|} & \text{if } \{i,j\} \in E \\ 0 & \text{otherwise} \end{cases}$$

for $i, j \in \{1, \ldots, n\}$ and

$$\tilde{f}(\mathbf{x}) = \sum_{i=1}^{n} \frac{1}{|N_i|} f_i(x_i),$$

we can express decentralized ADMM as

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathrm{Prox}_{\alpha \tilde{f}}(\mathbf{v}^k) \\ \mathbf{a}^{k+1} &= W\mathbf{x}^{k+1} \\ \mathbf{v}^{k+1} &= \mathbf{v}^k + \mathbf{a}^{k+1} - \frac{1}{2}\mathbf{a}^k - \frac{1}{2}\mathbf{x}^k. \end{aligned}$$

Conclusion

Distributed optimization takes advantages of problem structures and can solve extremely large optimization problems.

With a central coordinator, the methods rely on aggregating distributed gradients or averaging distributed iterates through a reduce operation.

Decentralized optimization uses neighborhood communication (i.e., decentralized mixing) instead of a global reduce operation.

By applying splitting and variable metric techniques, we obtain decentralized optimization methods.