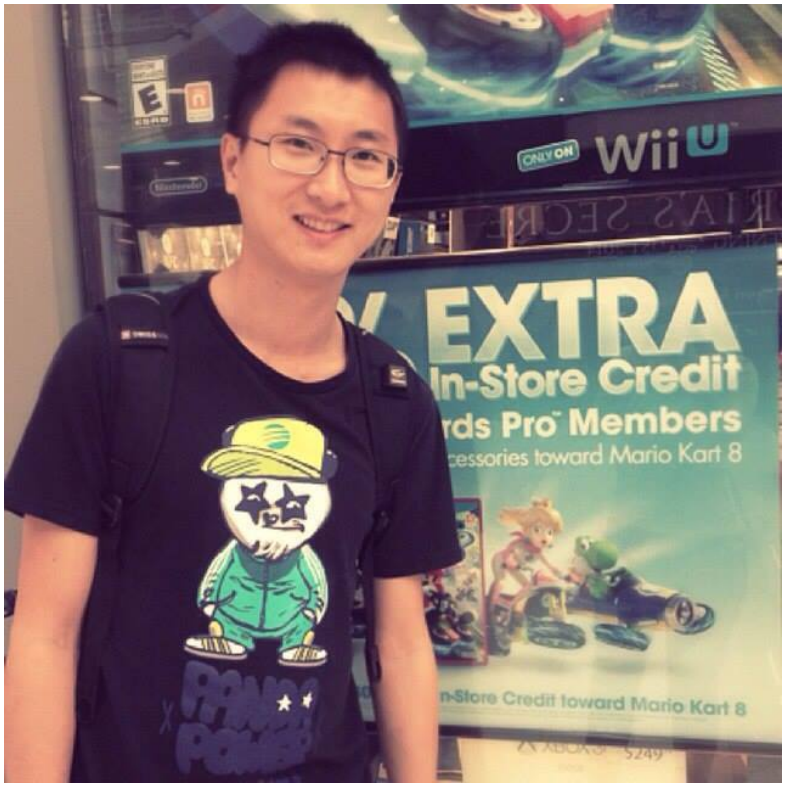
Distributed Algorithms for Optimization in Networks

Angelia.Nedich@asu.edu

School of Electrical, Computer, and Energy Engineering

December 17, 2020

In Fond Memory of Wei (Wilbur) Shi

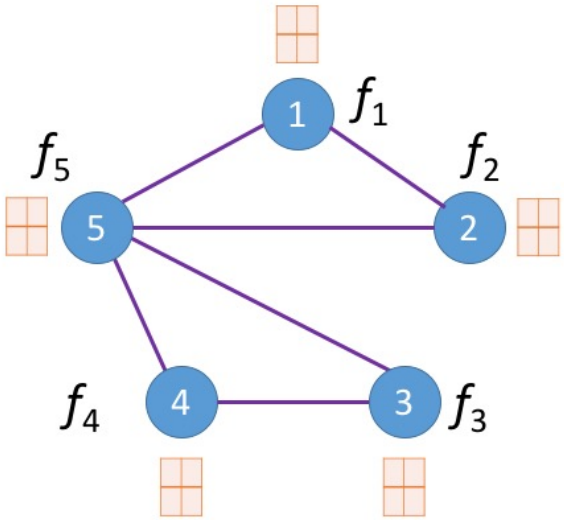


Problem Formulation

We consider a (machine learning) problem

$$\min_{x\in\mathbb{R}^n} \sum_{i=1}^{m} f_i(x)$$

in a system consisting of m agents that are embedded in a communication network.



- Each function $f_i(\cdot)$ is differentiable convex, and $privately\ known$ only to agent i.
- Agents communicate some limited information with their immediate neighbors only
- Agents do not share their functions $f_i(\cdot)$'s.

The problem is to be solved $distributedly$ i.e., **without a central entity**

The lack of central authority is compensated by agent collaboration and communication

- DeGroot consensus model [DeGroot 1974] - also referred to as agreement model
- A variant of this problem, using consensus model, has been studied in the 80's:

  Borkar & Varaya 1982, Tsitsiklis 1984, Tsitsiklis, Bertsekas & Athens 1986,

  Bertsekas & Tsitsiklis book

  "Parallel and Distributed Computations: Numerical Methods" 1989

Examples

Machine Learning (*Kunal Srivastava, Soomin Lee*):

The function $f_i(x)$ represents the loss associated with a collection of data points available to agent i. Specifically, let the collection $\{(z_p^i, y_p^i), p = 1, \ldots, m_i\}$ represent the data points of agent i, where $z_p^i \in \mathbb{R}^n$ is a feature vector and $y_p^i \in \{-1, +1\}$ is its corresponding label. The agent i objective function is

$$f_i(x) = c\rho(x) + \sum_{p=1}^{m_i} \ell(x; z_p^i, y_p^i),$$

where $c > 0$ is a regularization parameter, $x \in \mathbb{R}^n$ is a decision vector, $\rho(x)$ is a regularizing (strongly convex) function, and $\ell(x; z, y)$ is a loss function associated with the point $(z, y) \in \mathbb{R}^{n+1}$. For linear classifiers, a common choice is the logistic regression loss function, given by

$$\ell(x; z, y) = \log\left(1 + e^{-y\langle x, z\rangle}\right)$$

Sensor Systems:

- Distributed Source Localization (*Sundhar R. Srinivasan*)
- Distributed Uplink Power Scheduling for Mobile Devices (*Sundhar R. Srinivasan*)
- Distributed Hypothesis Testing (*Cesar A. Uribe*)

Distributed Method

To solve the problem

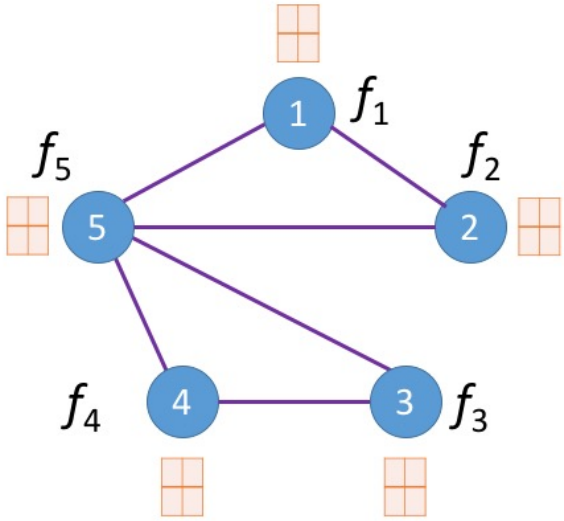
$$\min_{x\in\mathbb{R}^n} \sum_{i=1}^{m} f_i(x)$$

in a network of agents that communicate locally with immediate neighbors, we need to model the communication network.

We start by considering the static network case, where the network is represented by an undirected connected graph[a] $\mathbb{G} = ([m], \mathcal{E})$, where

- $[m] = \{1, 2, \ldots, m\}$ is the node set
- $\mathcal{E} \subset [m] \times [m]$ is the edge set

We assume that the graph $\mathbb{G}$ is connected



[a] This is assumed for simplicity. The graph can be directed or even time-varying.

Problem Reformulation

To take into account the graph, we can reformulate the problem as follows:

$$\text{minimize} \sum_{i=1}^{m} f_i(x_i) \qquad \underbrace{x_i = x_j \quad \text{for all } i \text{ and } j \in N_i}_{\text{agreement constraints}},$$

where $N_i = \{j \in [m] \mid \{i, j\}| \in \mathcal{E}\}$. When the graph is connected the constraint set is equivalent to $x_i = x_j$ for all agents i, j.

When $f_i = 0$, the resulting feasibility problem is known as *agreement problem.*

Agreement problem/ Consensus problem: design a distributed algorithm such that the iterate sequences $\{x_i(t)\}$, $i \in [m]$, generated locally by the agents satisfy

$$\lim_{t\to\infty} x_i(t) = \tilde{x} \qquad \text{for some } \tilde{x} \in \mathbb{R}^n \text{ and all } i,$$

where $x_i(0)$ is arbitrary. An algorithm using **weighted-averaging** solves the agreement problem: at time t, every agent i sends $x_i(t)$ to its neighbors $j \in N_i$, and receives $x_j(t)$ from them; then, every agent updates

$$x_i(t+1) = \sum_{j \in N_i \cup \{i\}} a_{ij} x_j(t),$$

where the weights a_{ij} satisfy

$$a_{ij} > 0 \qquad \sum_{j \in N_i \cup \{i\}} a_{ij} = 1.$$

Note that each agent i chooses the weights a_{ij}, $j \in N_i$, at its own discretion.

Introducing $a_{ij} = 0$ for $j \notin N_i \cup \{i\}$, we define the matrix $A = [a_{ij}]$, so the iterate process can be compactly written as:

$$X(t+1) = AX(t) = \cdots = A^{t+1}X(0),$$

where $X(t) = [x_1'(t); x_2'(t); \ldots; x_m'(t)]$. When A is a row stochastic ($\sum_{j=1}^{m} a_{ij} = 1$ for all i), A can be viewed as a transition matrix of an ergodic Markov Chain, and we have

$$\lim_{t \to \infty} A^t = \mathbf{1}\pi',$$

for a stochastic vector π with all entries positive, i.e., $\pi > \mathbf{0}$. Hence,

$$\lim_{t \to \infty} x_i(t) = \sum_{j=1}^{m} \pi_j x_j(0) \qquad \text{for all } i.$$

Distributed Consensus-Based Gradient Method

At time t, every agent i sends $x_i(t)$ to its neighbors $j \in N_i$, and receives $x_j(t)$ from them; then, every agent updates (AN and A. Ozdaglar 2009)

$$x_i(t+1) = \underbrace{\sum_{j=1}^{m} a_{ij} x_j(t)}_{\text{consensus}} - \alpha_t \nabla f_i(x_i(t)),$$

where $\alpha_t > 0$ is a stepsize.

Every agent moves along the negative gradient of its own objective function, evaluated at its current iterate $x_i(t)$.

Another variant has also been considered (*S.S. Ram et al. 2010*):

$$x_i(t+1) = \underbrace{\sum_{j=1}^{m} a_{ij} x_j(t)}_{\text{consensus}} - \alpha_t \nabla f_i \underbrace{\left(\sum_{j=1}^{m} a_{ij} x_j(t) \right)}_{\text{consensus}}$$

where $\alpha_t > 0$ is a stepsize. The difference from the preceding method is in the point where the gradient of f_i is evaluated.

Basic Convergence Result

Assuming that the problem has a solution, the graph $\mathbb{G}$ is connected, the matrix A is doubly stochastic, the gradients are bounded and stepsize satisfies $\sum_{t=0}^{\infty} \alpha_t = +\infty$ and $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$, we have that for some optimal solution x^*,

$$\lim_{t\to\infty} x_i(t) = x^* \qquad \text{for all } i.$$

- The convergence rate is $O(\frac{\log t}{\sqrt{t}})$. If $\sum_{i=1}^{m} f_i(x)$ is strongly convex, the rate is $O(\frac{\log t}{t})$

NOTES:

- The result holds for time varying graphs, assuming that each $A(t)$ is **doubly stochastic**.
- No knowledge about how the graphs are changing is used. That is considered in a recent work (stronger convergence rate results obtained with a constant stepsize): A. Rogozin, C. A. Uribe, A. Gasnikov, N. Malkovsky and AN *Optimal distributed convex optimization on slowly time-varying graphs*, IEEE Trans. on Control of Network Systems, 7 (2), 829 – 841, June 2020 (arxiv version at https://arxiv.org/abs/1805.06045).
- The method is slow due to the use of **diminishing stepsize**.

Why Doubly Stochastic A?

With a doubly stochastic matrix $A = [a_{ij}]$, the method

$$x_i(t+1) = \sum_{j=1}^{m} a_{ij} x_j(t) - \alpha_t \nabla f_i(x_i(t))$$

solves the problem $\min_x \frac{1}{m} \sum_{i=1}^{m} f_i(x)$.

If A is just row-stochastic, the algorithm would produce the iterates converging to a common point that solves the following problem:

$$\text{minimize} \quad \sum_{i=1}^{m} \pi_i f_i(x),$$

where π is the left-eigenvector of A for the eigenvalue $\lambda = 1$, i.e., $\pi' A = \pi'$.

The algorithm cannot be efficiently implemented in directed time-varying graphs *

An alternative to the weighted averaging is available through a *push-sum algorithm* for consensus (Kempe, Dobra & Gehrke 2003)†

*Gharesifard and Cortés *Distributed strategies for generating weight-balanced and doubly stochastic digraphs*, European Journal of Control, 18 (6), 539-557, 2012

†D. Kempe, A. Dobra, and J. Gehrke *Gossip-based computation of aggregate information*, In Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, pages 482–491, 2003

F. Benezit, V. Blondel, P. Thiran, J. Tsitsiklis, and M. Vetterli *Weighted gossip: distributed averaging using non-doubly stochastic matrices*, In Proceedings of the 2010 IEEE International Symposium on Information Theory, 2010

Directed Graphs: Push-sum Consensus and Push-sum Based Gradient Methods

- Dominguez-Garcia and Hadjicostis *Distributed strategies for average consensus in directed graphs* Proc. of the IEEE Conference on Decision and Control, 2011.
- Hadjicostis, Dominguez-Garcia, and Vaidya *Resilient Average Consensus in the Presence of Heterogeneous Packet Dropping Links*, CDC, 2012
- Tsianos and Rabbat *Distributed consensus and optimization under communication delays* Proc. of the 49th Allerton Conference on Communication, Control, and Computing, 2011.
- Tsianos, Lawlor, and Rabbat *Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning* Proceedings of the 50th Allerton Conference on Communication, Control, and Computing, 2012.
- Tsianos, Lawlor, and Rabbat *Push-sum distributed dual averaging for convex optimization* Proceedings of the IEEE Conference on Decision and Control, 2012.

- Tsianos *The role of the Network in Distributed Optimization Algorithms: Convergence Rates, Scalability, Communication / Computation Tradeoffs and Communication Delays* PhD thesis, McGill University, Dept. of Electrical and Computer Engineering, 2013.
- AN and A. Olshevsky *Distributed Optimization over Time-varying Directed Graphs* IEEE Transactions on Automatic Control 60 (3) 601–615, 2015
- AN and A. Olshevsky *Stochastic Gradient-Push for Strongly Convex Functions on Time-varying Directed Graphs* IEEE Transactions on Automatic Control 61 (12) 3936–3947, 2016
- AN *Distributed gradient methods for convex machine learning problems in networks* IEEE Signal Processing Magazine 37 (3) 92–101, 2020

Rate Issue - Quest to Match Centralized Methods' Performance

- Assume that the functions f_i have Lipschitz gradients and $\sum_{i=1}^{m} f_i(x)$ is strongly convex
- The consensus-based gradient algorithm will not necessarily produce convergent iterates with **a constant stepsize** $\alpha_i = \alpha$
- Brought to attention in the work of Shi, Ling, Wu, and Yin (EXTRA algorithm) 2014, 2015
- The consensus-based gradient method with a constant stepsize

$$x_i(t+1) = \sum_{j=1}^{m} a_{ij} x_j(t) - \alpha \nabla f_i(x_i(t))$$

  Assuming the iterates converge to some point $\tilde{x}$: $\tilde{x} = \tilde{x} - \alpha \nabla f_i(\tilde{x}) \implies \nabla f_i(\tilde{x}) = 0$
  For the method to work, the agent functions should have a common minimizer!
- The method has to use diminishing step for convergence.
- It **work wells in the presence of random noise** (noisy gradient computations, noisy communication links - $Kunal\ Srivastava$)

Quest to Match Centralized Methods' Performance

$$\text{minimize} \sum_{i=1}^{m} f_i(x_i) \qquad \underbrace{x_i = x_j \quad \text{for all } i \text{ and } j \in N_i}_{\text{agreement constraints}},$$

Linear equality-constrained convex optimization problem.

- It can be solved distributedly through its dual, as done in our recent work: C. A. Uribe, S. Lee, A. Gasnikov and AN *A Dual Approach for Optimal Algorithms in Distributed Optimization over Networks* Optimization Methods and Software, 2020, arxiv link https://arxiv.org/abs/1809.00710

- The performance of the fastest centralized methods is matched up to a log-factor - $\log t$ per iteration count t.

- Difficult to extend to general time-varying graphs - Rogozin 2020 manages this for slowly-time varying graphs.

- Other alternatives?

Achieving Linear Rate: Methods Tracking $\sum_{i=1}^m \nabla f_i$

- The agents are selfish in the method

$$x_i(t+1) = \underbrace{\sum_{j=1}^m a_{ij} x_j(t)}_{\text{collaborative decision}} - \alpha_t \underbrace{\nabla f_i(x_i(t))}_{\text{selfish direction}}$$

- The agents should be more collaborative in "directions" not just decision. In the models with gradient tracking, the agents are "more aware" of the system objective
- **Basic Idea**: Every agent i uses an estimate $g_i(t)$ for the gradient $\sum_{i=1}^m \nabla f_i(x_i(t))$: at time t, agents exchange both decision estimates $x_j(t)$ and the gradient estimates $g_j(t)$, and update

$$x_i(t+1) = \sum_{j=1}^m a_{ij} x_j(t) - \alpha g_i(t)$$

$$g_i(t+1) = \sum_{j=1}^m a_{ij} g_j(t) + \nabla f_i(x_i(t+1)) - \nabla f_i(x_i(t))$$

where $\alpha > 0$ is a stepsize. The updates are reminiscent of "tracking/filtering" (the innovation term is the gradient difference)

Closely Related Literature and Simultaneous Work – uses tracking

- M. Zhu and S. Martínez, *Discrete-Time Dynamic Average Consensus*, Automatica, 46 (2010),
- J. Xu, S. Zhu, Y. Soh, and L. Xie, *Augmented Distributed Gradient Methods for Multi-Agent Optimization Under Uncoordinated Constant Stepsizes*, in Proceedings of the 54th IEEE Conference on Decision and Control (CDC), 2015, pp. 2055–2060.
- Algorithms NEXT and SONATA
  - P. Di Lorenzo and G. Scutari *Distributed nonconvex optimization over networks*, in IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2015, pp. 229–232.
    P. Di Lorenzo and G. Scutari, *NEXT: In-Network Nonconvex Optimization*, IEEE Transactions on Signal and Information Processing over Networks, 2016.
    P. Di Lorenzo and G. Scutari *Distributed nonconvex optimization over time-varying networks*, in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016, pp. 4124–4128.

- Y.Sun, G. Scutari, D.Palomar *Distributed Nonconvex Multiagent Optimization Over Time-Varying Networks* https://arxiv.org/abs/1607.00249, 2016

  Y. Tian, Y. Sun, B. Du, G. Scutari *ASY-SONATA: Achieving Geometric Convergence for Distributed Asynchronous Optimization* Allerton Conference on Communication, Control, and Computing (Allerton) 2018

  Y. Sun, A. Daneshmand, G. Scutari *Convergence Rate of Distributed Optimization Algorithms Based on Gradient Tracking* https://arxiv.org/abs/1905.02637, 2019

DIGing Method for Undirected Time-Varying Graphs

- **Exchange**:
  Every agent i sends $x_i(k), g_i(k)$ to all its neighbors $j \in N_i(k)$ in the graph $\mathcal{G}(k)$ and receives $x_j(k), g_j(k)$ from its neighbors

- **Update**: Every agent i updates the decision and the direction as follows

$$x_i(k+1) = \sum_{j=1}^m a_{ij}(k)x_j(k) - \alpha g_i(k);$$
$$g_i(k+1) = \sum_{j=1}^m a_{ij}(k)g_j(k) + \nabla f_i(x_i(k+1)) - \nabla f_i(x_i(k)).$$

- The matrix $A(k) = [a_{ij}(k)]$ is doubly stochastic and compatible with the graph $\mathcal{G}(k)$: $a_{ii}(k) > 0$ and $a_{ij}(k) > 0$ for $j \in N_i(k)$, otherwise $a_{ij}(k) = 0$.

- The stepsize α is common to all[1‡]

- The method is initialized with arbitrary $x_i(0) \in \mathbb{R}^n$ and $g_i(0) = \nabla f_i(x_i(0))$ for all i.

‡It can be agent dependent AN, Alex Olshevsky, Wei Shi, Cesar Uribe *Geometrically Convergent Distributed Optimization with Uncoordinated Step-Sizes*, CDC 2016.

Assumptions for Linear Convergence Rate for DIGing

- The functions f_i are convex with Lipschitz gradients (with Lipschitz constant L_i)
- The sum $\frac{1}{m}\sum_{i=1}^{m} f_i$ is strongly convex with a constant $\bar{\mu} > 0$
- The graphs $\mathcal{G}(k)$ are B-connected: for some integer $B \geq 1$, the graph $([m], \cup_{t=k}^{k+B-1}\mathcal{E}(t)$ is connected for all k.
- $A(k)$ is doubly stochastic, compatible with the graph $\mathcal{G}(k)$, and there is a lower bound $\tau > 0$ on its positive entries: for all k,

$$a_{ij}(k) \geq \tau \qquad \text{whenever} \quad a_{ij}(k) > 0.$$

Under these assumptions we have the following result.

Theorem 1 (DIGing: Geometric rate) *The sequences $\{x_i(k)\}, i \in [m]$, generated by DIGing converge to the (unique) optimal solution x^* at a linear rate $O(\lambda^k)$, where $\lambda \in (0,1)$ depends on the stepsize α, the condition number $\bar{\kappa} = \frac{L}{\bar{\mu}}$ with $L = \max_i L_i$, and on the mixing matrices.*

Specialized Result

Corollary 2 (DIGing: Polynomial networks scalability) *If the graphs are undirected and* $A(k)$ *is a lazy Metropolis matrix*

$$a_{ij}(k) = \begin{cases} 1/\left(1+\max\{d_i(k), d_j(k)\}\right), & \text{if } j \in N_i(k), \\ 1 - \sum_{\ell \in \mathcal{N}(k)} a_{i\ell}(k), & \text{if } j = i, \\ 0, & \text{else}, \end{cases}$$

where $d_i(k) = |N_i(k)|$ *is the degree of a node* i*, and the step-size isr*

$$\alpha = \frac{3(2/71)^2}{128B^2m^{4.5}L\sqrt{\bar{\kappa}}} - \frac{1.5}{\bar{\mu}}\left(\frac{(2/71)^2}{128B^2m^{4.5}\bar{\kappa}^{1.5}}\right)^2,$$

then to reach an ε*-accuracy, the number of iterations needed by DIGing algorithm is*

$$O\left(B^3m^{4.5}\bar{\kappa}^{1.5}\ln\frac{1}{\varepsilon}\right).$$

DIGing method for directed graphs with a linear rate, relying on push-sum consensus is in: AN, A. Olshevsky, W. Shi *Achieving Geometric Convergence for Distributed Optimization over Time-Varying Graphs* SIAM Journal on Optimization 27 (4) 2597–2633, 2017.

However, a polynomial scaling for a directed graph is still an open question.

AB/Push-Pull Method

- It is a variant of DIGing that uses different matrices for mixing the decisions and the directions
- Works on both undirected and directed graphs, but **static** i.e., $\mathbb{G} = ([m], \mathcal{E})$. So we assume that the graph is directed.
- **Exchange**: (from an agent's perspective

  - (**Push**) Every agent i sends $x_i(t)$ to its out-neighbors $N_i^{\text{out}} = \{p \mid (i,p) \in \mathcal{E}\}$, and receives $x_j(k)$ from its in-neighbors $j \in N_i^{\text{in}} = \{p \mid (p,i) \in \mathcal{E}\}$
  - (**Pull**) Every agent i sends $b_{\ell i} g_i(k)$ to all its out-neighbors $\ell \in N_i^{\text{out}}$, and receives $b_{ij} g_j(k)$ from its in-neighbors $j \in N_i^{\text{in}}$
  - The coefficients $b_{\ell i} > 0$ and $b_{ii} > 0$ are selected by the agent i and satisfy

$$\sum_{\ell \in N_i^{\text{out}} \cup \{i\}} b_{\ell i} = 1 \qquad \text{for all } i.$$

- **Update**: Every agent i updates its decision $x_i(k)$ and direction $g_i(k)$ as follows

$$x_i(k+1) = \sum_{j \in N_i^{\text{in}} \cup \{i\}} a_{ij} x_j(k) - \alpha g_i(k)$$

$$g_i(k+1) = \sum_{j \in N_i^{\text{in}} \cup \{i\}} b_{ij} g_j(k) + \nabla f_i(x_i(k+1)) - \nabla f_i(x_i(k)). \qquad (1)$$

The weights a_{ij} are positive and sum to 1:

$$a_{ij} > 0 \qquad \text{for all } j \in N_i^{\text{in}} \cup \{i\} > 0.$$

Agent i selects these weights.

Define

$$a_{ij} = 0 \qquad \text{if } j \notin N_i^{\text{in}} \cup \{i\},$$
$$b_{ij} = 0 \qquad \text{if } j \notin N_i^{\text{out}} \cup \{i\}.$$

The matrix $A = [a_{ij}]$ is row stochastic, while $B = [b_{ij}]$ is a column stochastic.

The method is initialized with arbitrary $x_i(0) \in \mathbb{R}^n$ and $g_i(0) = \nabla f_i(x_i(0))$ for all i.

Simultaneous Work

- S. Pu, W. Shi, J. Xu, and AN, *A push-pull gradient method for distributed optimization in networks*, Proceedings of the 54th IEEE Conference on Decision and Control (CDC), 2018. **Journal version** to appear soon in IEEE Transactions on Automatic Control; journal version is on arxiv: https://arxiv.org/abs/1810.06653
- C. Xi, V. S. Mai, R. Xin, E. H. Abed, and U. A. Khan, *Linear convergence in optimization over directed graphs with row-stochastic matrices*, IEEE Transactions on Automatic Control, 2018.
- R. Xin, C. Xi, and U. A. Khan, *Frost–fast row-stochastic optimization with uncoordinated step-sizes*, EURASIP Journal on Advances in Signal Processing, 2019.
- R. Xin and U. A. Khan, *A linear algorithm for optimization over directed graphs with geometric convergence*, arXiv preprint arXiv:1803.02503, 2018; IEEE Control Systems Letters 2 (3) 315–320, 2018.

To recognize simultaneous development of the method by Xin and Khan, we refer to the method AB/Push-Pull in the future work.

Interpretation of AB/Push-Pull Method

- Suppose the graph $([m], \mathcal{E})$ is **strongly connected**, and suppose $x \in \mathbb{R}$ (for simplicity)
- Reformulate the problem of $\min_{x\in\mathbb{R}^n} \sum_{i=1}^m f_i(x)$:

$$\min_{x_i, i\in[m]} \sum_{i=1}^m f_i(x_i), \qquad x_i = x_j, j \in \mathcal{N}_i^{\text{in}}, i \in [m]$$

- Define $\mathbf{x} = \text{col}(x_1, \ldots, x_m)$ and

$$F(\mathbf{x}) = \sum_{i=1}^m f_i(x_i).$$

Then the optimality condition can be stated as: $\mathbf{x}^*$ is optimal if and only if

$$\mathbf{x}^* = a^*\mathbf{1}, \qquad \mathbf{1}'\nabla F(\mathbf{x}^*) = 0,$$

where $a^* \in \mathbb{R}$ and $\nabla F(\mathbf{x}) = \text{col}(f_1'(x_1), \ldots, f_m'(x_m))$.

- In this compact form AB/Push-Pull iterations are

$$\mathbf{x}(k+1) = A\mathbf{x}(k) - \alpha\mathbf{g}(k)$$
$$\mathbf{g}(k+1) = B\mathbf{g}(k) + \nabla F(\mathbf{x}(k+1)) - \nabla F(\mathbf{x}(k))$$

with $\mathbf{g}(k) = (g_1(k), \ldots, g_m(k))'$.

If the sequences were convergent, i.e., $\mathbf{x}(k) \to \bar{x}$ and $\mathbf{g}(k) \to \bar{\mathbf{g}}$, then we would have

$$\bar{\mathbf{x}} = A\bar{\mathbf{x}} - \alpha\bar{\mathbf{g}}, \qquad \bar{\mathbf{g}} = B\bar{\mathbf{g}}.$$

Since B is column-stochastic, compatible with the graph and the graph is strongly connected, there is a unique probability vector $\pi > 0$ such that

$$\pi = B\pi \quad \Longrightarrow \quad \bar{\mathbf{g}} = c\pi.$$

$$(A - I)\bar{\mathbf{x}} = \alpha\bar{\mathbf{g}} \quad \Longrightarrow \quad (A - I)\bar{\mathbf{x}} = \alpha c\pi \quad (I \text{ is the identity matrix})$$

So, π is in the intersection of the range space of $A - I$ and the null space of $B - I$.

- When these two spaces have only the zero vector in common, we have $c = 0$, so that

$$\bar{\mathbf{g}} = 0 \implies A\bar{\mathbf{x}} = \bar{\mathbf{x}}.$$

Since A is row-stochastic, compatible with the graph and the graph is strongly connected, the vector $\mathbf{1} = (1, \ldots, 1)'$ is the unique vector (up to a scaling) satisfying

$$A\mathbf{1} = \mathbf{1} \quad \Longrightarrow \quad \bar{\mathbf{x}} = a\mathbf{1}.$$

- Since B is column stochastic, from the update equation it can be seen that

$$\mathbf{1}'\mathbf{g}(k) = \mathbf{1}'\nabla F(\mathbf{x}(k)) \implies \mathbf{1}'\bar{\mathbf{g}} = \mathbf{1}'\nabla F(\bar{\mathbf{x}}) \implies 0 = \mathbf{1}'\nabla F(\bar{\mathbf{x}}),$$

where in the last step we use $\bar{\mathbf{g}} = 0$.

- $\bar{\mathbf{x}} = a\mathbf{1}$ and $0 = \mathbf{1}'\nabla F(\bar{\mathbf{x}})$ are the conditions for $\bar{\mathbf{x}}$ to be a solution of the problem.

Convergence Result

Assume that:

- The graph $\mathbb{G}$ is directed and strongly connected
- The matrices A and B are compatible with the graph and, respectively, row-stochastic and column-stochastic
- Each f_i has Lipschitz continuous gradients with a constant $L > 0$
- The sum $\sum_{i=1}^{m} f_i$ is strongly convex with a constant $\mu > 0$

Proposition 3 *Under these assumptions the AB/Push-Pull Method produces the iterate sequences* $\{x_i(t)\}$ *that converge geometrically fast to the optimal solution of the problem for a sufficiently small stepsize* α.

The analysis makes use of the Perron vectors (probability vectors) for A and B:

$$\pi_a' A = \pi_a', \qquad B\pi_b = \pi_b,$$

and the weighted average of $x_1(k), \ldots, x_m(k)$.

$$\bar{x}(k) = \pi_a' \mathbf{x}(k).$$

The progress of the algorithm is measured in terms of three quantities:

$$|\bar{x}(k) - x^*|, \qquad \|\mathbf{x}(k) - \bar{x}(k)\mathbf{1}\|_{\pi_a} = \left(\sum_{i=1}^{m} [\pi_a]_i (x_i(k) - \bar{x}(k))^2 \right)^{1/2},$$

$$\|\mathbf{g}(k) - s(k)\pi_b\|_{\pi_b^{-1}} = \left(\sum_{i=1}^{m} \frac{(g_i(k) - s(k)[\pi_b]_i)^2}{[\pi_b]_i} \right)^{1/2}, \qquad \text{with } s(k) = \sum_{i=1}^{m} g_i(k)$$

Main relation:

$$\begin{bmatrix} |\bar{x}(k+1)-x^*|^2 \\ \|\mathbf{x}(k+1)-\bar{x}(k+1)\mathbf{1}\|^2_{\pi_a} \\ \|\mathbf{g}(k+1)-s(k+1)\,\pi_b\|^2_{\pi_b^{-1}} \end{bmatrix} \leq D(\alpha) \begin{bmatrix} |\bar{x}(k)-x^*|^2 \\ \|\mathbf{x}(k)-\bar{x}(k)\mathbf{1}\|^2_{\pi_a} \\ \|\mathbf{g}(k)-s(k)\,\pi_b\|^2_{\pi_b^{-1}} \end{bmatrix}$$

$$D(\alpha) = \begin{bmatrix} 1-O(\alpha) & O(\alpha) & O(\alpha^2) \\ O(1) & 1-\sigma_2(A) & O(\alpha^2) \\ O(1) & O(1) & 1-\sigma_2(B)+O(\alpha) \end{bmatrix}$$

- $\sigma_2(A)$ is the second largest singular value of a matrix A

Then, all three quantities converge to 0 at a linear rate with coefficient $\rho_{D(\alpha)} < 1$, where ρ_D is a spectral radius of a matrix D, provided that the stepsize is small enough.

- Details and some simulations can be found in:
  S. Pu, W. Shi, J. Xu, and AN *Push-pull gradient methods for distributed optimization in networks*, arXiv preprint at https://arxiv.org/abs/1810.06653. IEEE TAC 2020.
- Closely related recent paper:
  R. Xin, A.K. Sahu, U.A. Khan, and S. Kar *Distributed stochastic optimization with gradient tracking over strongly connected networks* CDC 2019
- Recent survey paper: R. Xin, S. Pu, AN, U.A. Khan, *A General Framework for Decentralized Optimization With First-Order Methods* Proceedings of the IEEE 2020

Conclusion

- Distributed algorithms have attracted a lot of attention
- Fast distributed gradient methods are developed that can match the best performance of centralized gradient methods
- New directions
  - Solving nonconvex problems: T. Tatarenko & B. Touri 2017, A. Scutari's group at Purdue, S. Shahrampour TAMU 2020
  - Asynchronous implementations: S. Pu (CUHK-China), A. Scutari's group
  - Impact of network topology: N. Neglia at INRIA, A. Olshevsky at BU
  - Impact of delays: M. Johansson at KTH, M.G. Rabbat at Facebook/McGill
  - Performance in presence of malicious agents: H.-T. Wai (CUHK-Hong Kong), M. Alizadeh (UCSB), S. Sundaram (Purdue), N. Vaidya (GM), A. Scaglione (ASU), W.U. Bajwa (Rutgers), AN
  - Privacy in optimization: Y. Wang at Clemson University