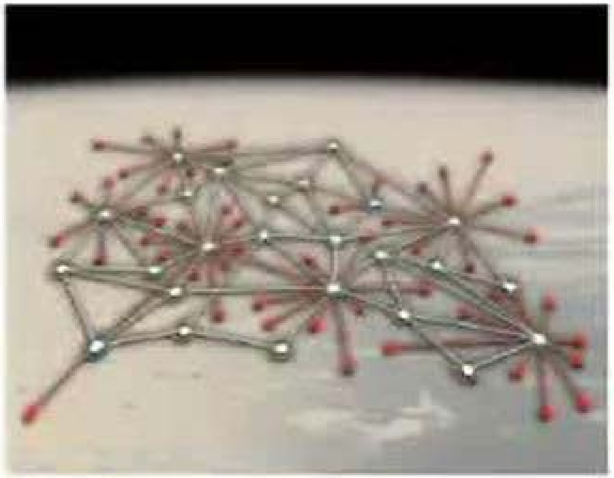
Featured Minisymposium:

Distributed Methods for Optimization

Large-Scale Systems



Key Challenges

Distributed and Large Scale Data

- Lack of central "authority"
  - Centralized architecture **not possible**
    - Size of the network / Proprietary issues
  - Centralized architecture **not desirable**
    - Security issues / Robustness to failures
- Large Data
  - Processing
  - Uncertainties
  - Data mining & learning
  - Statistical inference
- Network connectivity dynamics
  - Mobility of the network
  - Temporal data dynamics
- Data Characteristics
  - Space/ Time variability
  - Sparsity
- Challenges are to control, coordinate, optimize and analyze operations/performance of such distributed and large scale systems

Minisymposium on Distributed Methods for Optimization

- Focused on recently developed techniques for optimization in large scale systems

- Talks
  - Distributed Optimization in Directed Graphs: Push-Sum Based Algorithms
  - Distributed Optimization in Undirected Graphs: Gradient and EXTRA Algorithms
  - On the $O(1/k)$ Convergence of Asynchronous Distributed Alternating Direction Method of Multipliers
  - Blessing of Scalability: A Tractable Dual Decomposition l_0 Approach for Large Graph Estimation

Distributed Optimization in Directed Graphs:

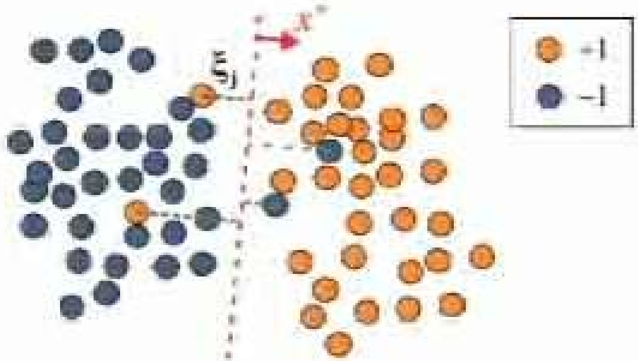
Push-Sum Based Algorithms

Angelia Nedić and Alexander Olshevsky

Industrial and Enterprise Systems Engineering Department
and Coordinated Science Laboratory
University of Illinois at Urbana-Champaign

Example: Support Vector Machine (SVM)
Centralized Case

Given a data set $\{z_j, y_j\}_{j=1}^{p}$, where $z_j \in \mathbb{R}^d$ and $y_j \in \{+1, -1\}$



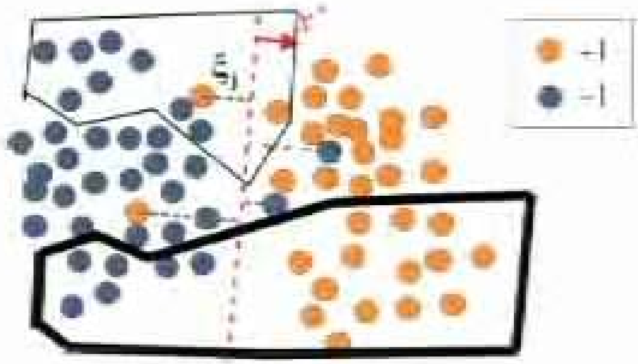
- Find a maximum margin separating hyperplane x^*

  Centralized (not distributed) formulation

$$\min_{x \in \mathbb{R}^d, \xi \in \mathbb{R}^p} F(x) \triangleq \frac{\rho}{2}\|x\|^2 + \sum_{j=1}^{p} \max\{\xi_j, 1 - y_j \langle x, z_j \rangle\}$$

Support Vector Machine (SVM) - Decentralized Case

Given n locations, each location i with its data set $\{z_j, y_j\}_{j \in J_i}$, where $z_j \in \mathbb{R}^d$ and $y_j \in \{+1, -1\}$



- Find a maximum margin separating hyperplane x^*, without disclosing the data sets

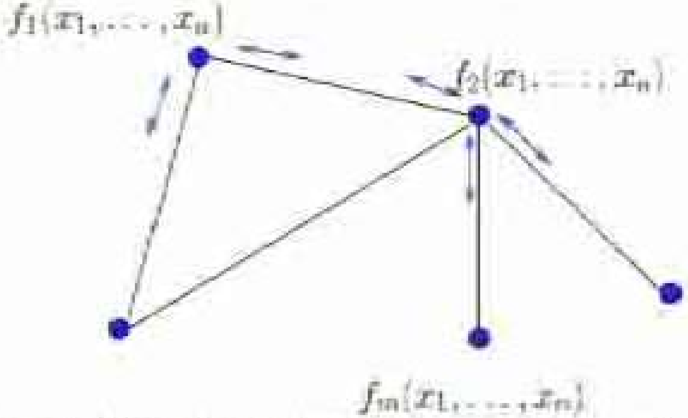
$$\min_{x \in \mathbb{R}^d, \xi \in \mathbb{R}^p} \sum_{i=1}^{n} \left(\frac{\rho}{2n} \|x\|^2 + \sum_{j \in J_i} \max\{\xi_j, 1 - y_j \langle x, z_j \rangle\} \right)$$

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

$$f_i(x) = \frac{\rho}{2n} \|x\|^2 + \sum_{j \in J_i} \min_{\xi_j \in \mathbb{R}} \max\{\xi_j, 1 - y_j \langle x, z_j \rangle\}$$

General Distributed Multi-Agent Model

Distributed Self-organized Agent System



The problem can be formalized:

$$\begin{array}{ll} \text{minimize} & F(x) \triangleq \sum_{i=1}^{n} f_i(x) \\ \text{subject to} & x \in X, \quad X \subseteq \mathbb{R}^d \end{array}$$

- Network of n agents represented by a graph $([n], \mathcal{E}_t)$ where $[n] = \{1, \ldots, n\}$
- The edge set $\mathcal{E}_t$ captures the agent communications at time t
- Each agent i has a **convex** objective function $f_i : \mathbb{R}^d \to \mathbb{R}$ known to that agent only
- All agents know the set X, which is closed and convex
- Each agent sends/receives some information to/from its neighbors

How Can Agents Solve the Problem?

$$\text{minimize } \sum_{i=1}^{n} f_i(x) \text{ subject to } x \in X \subseteq \mathbb{R}^d$$

Decompose the problem: an individual copy of the decision variable per agent

$$\text{minimize } \sum_{i=1}^{n} f_i(x_i) \qquad \text{subject to} \qquad x_i \in X \subseteq \mathbb{R}^d$$

$$x_i = x_j \qquad \text{for all } i, j = 1, \ldots, n \quad \text{agreement constraints}$$

The key is in suitable equivalent re-formulation of the "agreement" constraints

Assume the agents communicate over a static (bi-directional) network

$$x_i = x_j \quad \text{for all } i \text{ and its neighbors } j = 1, \ldots, n$$

$$\Longleftrightarrow \qquad d_{ii} x_i = \sum_{j \in N_i} x_j \quad \text{for every } i \qquad \text{Laplacian form (scalar case) } Lx = 0,$$

where N_i is the set of neighbors of agent i and $d_{ii} = |N_i|$

$$\Longleftrightarrow \qquad x_i = \frac{1}{N_i^s} \sum_{j \in N_i^s} x_j \quad \text{for every } i \qquad \text{equal-neighbor weights (averaging).}$$

$$\Longleftrightarrow \qquad x_i = \sum_{j \in N_i^s} a_{ij} x_j \quad \text{for every } i \qquad \text{weighted-averaging (scalar case) } Ax = x,$$

where $N_i^s = N_i \cup \{i\}$ and $a_{ij} > 0$ with $\sum_{j \in N_i^s} a_{ij} = 1$.

In this way the problem is equivalent to

$$\text{minimize } \sum_{i=1}^{n} f_i(x_i)$$
$$\text{subject to } x_i \in X \subseteq \mathbb{R}^d$$

network impact $A\mathrm{x} = \mathrm{x}$ or $(I - \Lambda L)\mathrm{x} = \mathrm{x}$, constraints coupling the agents

where

$$\mathrm{x} = \begin{bmatrix} x_1' \\ \vdots \\ x_n' \end{bmatrix},$$

and Λ is a diagonal matrix. The linear constraints are distributed and local by noting that each i agent may work with ith row of the corresponding matrix, i.e., each agent knows A_i or L_i, as the values of x_j are supplied from the neighbors

A general approach in optimization exists that proceeds by interleaving two steps

- A step toward minimizing a function (can be with projection when X is simple)
- A step toward "feasibility" - here corresponds to alignment of agreement of vectors $x_j, j = 1, \ldots, n$ expressed in "fixed point equation".

Consensus-Based Distributed Optimization Algorithm

- **Consensus-like step - feasibility step** for constraints $\mathrm{x} = A\mathrm{x}$ (or its Laplacian form)

$$w_i(t+1) = \sum_{j=1}^{n} a_{ij}(t) x_j(t) \qquad \text{with } a_{ij}(t) = 0 \text{ when } j \notin N_i(t)$$

- Followed by **a local gradient-based step**

$$x_i(t+1) = \Pi_X\left[w_i(t+1) - \alpha(t)\nabla f_i(w_i(t+1))\right]$$

where f_i is the local objective of agent i, $\alpha(t) > 0$ is a stepsize, and $\Pi_X[x]$ is the Euclidean projection on the set X

Intuition Behind the Algorithm: It can be viewed as a consensus steered by a "force":

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

- Algorithm works with time varying matrices (graphs) - all have the same fixed point solutions under a graph connectivity assumption

- Such an algorithm can solve the problem (under some technical conditions on $A(t)$)

- Matrices $A(t)$ that lead to the **average**-consensus also yield convergence of the algorithm

- **Main Difficulty**: Understanding the mixing rate in terms of the graph structure and problem data

- **Drawback**: Construction of doubly stochastic matrices requires some additional information exchange
  - It can be accomplished with some additional "weights" exchange in *bi-directional graphs*
  - Hard to do in the networks with communication delays and/or asynchronous updates
  - Computationally prohibitive in directed graphs[¶*]

*¶B. Gharesifard and J. Cortes, "Distributed strategies for generating weight-balanced and doubly stochastic digraphs," European Journal of Control, 18 (6), 539-557, 2012

Distributed Optimization in Directed Networks

Motivated by work of Rabbat, Tsianos and Lawlor addressing practical issues with bi-directional communications

Related Work: all dealing with a **static network**

- A.D. Dominguez-Garcia and C. Hadjicostis "Distributed strategies for average consensus in directed graphs" CDC 2011.
- C. N. Hadjicostis, A.D. Dominguez-Garcia, and N.H. Vaidya "Resilient Average Consensus in the Presence of Heterogeneous Packet Dropping Links" CDC 2012
- K.I. Tsianos "The role of the Network in Distributed Optimization Algorithms: Convergence Rates, Scalability, Communication / Computation Tradeoffs and Communication Delays" PhD thesis, McGill University, ECE Dept., 2013.
- K.I. Tsianos, S. Lawlor, and M.G. Rabbat "Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning" Allerton Conference 2012.
- K.I. Tsianos, S. Lawlor, and M.G. Rabbat "Push-sum distributed dual averaging for convex optimization" IEEE CDC 2012.
- K.I. Tsianos and M.G. Rabbat "Distributed consensus and optimization under communication delays" Allerton Conference 2011.

Push-Sum Method (Ratio Consensus): Basic Idea

Having an $n \times n$ column-stochastic matrix A, consider the following process

$$x(t) = Ax(t-1) \qquad \text{for } t \geq 1,$$

starting with some $x(0) \in \mathbb{R}^n$. Under some conditions the matrix A^t converges to a rank-one column-stochastic matrix,

$$\lim_{t\to\infty} x(t) = [\pi 1'] \, x(0) = \left(\sum_{i=1}^{n} x_i(0)\right) \pi, \quad \text{with } \pi_i > 0 \text{ for all } i$$

With a different initial condition, we can run the same process and obtain say $y(t)$,

$$\lim_{t\to\infty} y(t) = [\pi 1'] \, y(0) = \left(\sum_{i=1}^{n} y_i(0)\right) \pi$$

Consider the coordinate-wise ratio process

$$z_i(t) = \frac{x_i(t)}{y_i(t)},$$

for which we have

$$\lim_{t\to\infty} z_i(t) = \frac{\sum_{i=1}^{n} x_i(0)}{\sum_{i=1}^{n} y_i(0)}.$$

Thus, to obtain the average of $\{x_i(0), i \in [n]\}$, we just set $y_i(0) = 1$ for all i

- How about doing this with time-varying matrices $A(t)$?

Push-Sum for Time-Varying Directed Graphs

- Agents communications are given by a time-varying graph sequence $\{G(t)\}$
- $N_i^{\text{in}}(t)$ is the set of "in"-neighbors of node i at time t (in the graph $G(t)$)
- Each node i "knows" its out degree $d_i(t)$ (includes itself) at every time t
- Every node i maintains scalar variables $x_i(t)$ and $y_i(t)$
- These quantities will be updated by the nodes according to the rules,

$$
\begin{aligned}
x_i(t+1) &= \sum_{j \in N_i^{\text{in}}(t)} \frac{x_j(t)}{d_j(t)}, \\
y_i(t+1) &= \sum_{j \in N_i^{\text{in}}(t)} \frac{y_j(t)}{d_j(t)}, \\
z_i(t+1) &= \frac{x_i(t+1)}{y_i(t+1)}
\end{aligned}
\tag{1}
$$

- The method[††] is initiated with an arbitrary $x_i(0)$ and $y_i(0) = 1$ for all i.

[†]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, Oct. 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, Jun 2010.

Perturbed Push-Sum: Scalar Case

$$\begin{aligned} w_i(t+1) &= \sum_{j \in N_i^{\text{in}}(t)} \frac{x_j(t)}{d_j(t)}, \\ y_i(t+1) &= \sum_{j \in N_i^{\text{in}}(t)} \frac{y_j(t)}{d_j(t)}, \\ z_i(t+1) &= \frac{w_i(t+1)}{y_i(t+1)} \\ x_i(t+1) &= w_i(t+1) + \epsilon_i(t+1) \end{aligned} \tag{2}$$

where $\epsilon_i(t+1)$ are perturbations

Error-Bound Result

Consider the sequences $\{z_i(t)\}$, $i = 1, \ldots, n$, generated by the push-sum method.

Lemma 1 (Key) *Assuming that the graph sequence $\{G(t)\}$ is B-uniformly strongly connected, for all $t \geq 1$ we have*

$$\left| z_i(t+1) - \frac{\sum_{i=1}^n x_i(t)}{n} \right| \leq \frac{8}{\delta} \left(\lambda^t \|x(0)\|_1 + \sum_{s=1}^{t} \lambda^{t-s} \|\epsilon(s)\|_1 \right),$$

where $\delta > 0$ and $\lambda \in (0,1)$ satisfy

$$\delta \geq \frac{1}{n^{nB}}, \qquad \lambda \leq \left(1 - \frac{1}{n^{nB}}\right)^{1/B}.$$

Define matrices $A(t)$ by $A_{ij}(t) = 1/d_j(t)$ for $j \in N_i^{\mathrm{in}}(t)$ and 0 otherwise
If each of the matrices $A(t)$ are doubly stochastic, then

$$\delta = 1, \qquad \lambda \leq \left(1 - \frac{1}{4n^3}\right)^{1/B}.$$

Optimization

The subgradient-push method for minimizing $F(z) = \sum_{i=1}^{n} f_i(z)$ **over** $z \in \mathbb{R}^d$
Every node i maintains scalar variables $\mathrm{x}_i(t), \mathrm{w}_i(t)$ in $\mathbb{R}$, as well as an auxiliary scalar variable $y_i(t)$, initialized as $y_i(0) = 1$ for all i. These quantities will be updated by the nodes according to the rules,

$$\begin{aligned}
\mathrm{w}_i(t+1) &= \sum_{j \in N_i^{\mathrm{in}}(t)} \frac{\mathrm{x}_j(t)}{d_j(t)}, \\
y_i(t+1) &= \sum_{j \in N_i^{\mathrm{in}}(t)} \frac{y_j(t)}{d_j(t)}, \\
\mathrm{z}_i(t+1) &= \frac{\mathrm{w}_i(t+1)}{y_i(t+1)}, \\
\mathrm{x}_i(t+1) &= \mathrm{w}_i(t+1) - \alpha(t+1)\mathrm{g}_i(t+1), \qquad (3)
\end{aligned}$$

where $\mathrm{g}_i(t+1)$ is a subgradient of the function f_i at $\mathrm{z}_i(t+1)$. The method is initiated with arbitrary $\mathrm{x}_i(0)$ and $y_i(0) = 1$ for all i.

Convergence Result

Our first result demonstrates the correctness of the subgradient-push method

Proposition 1 *Suppose that:*

(a) *The graph sequence $\{G(t)\}$ is B-uniformly strongly connected.*

(b) *Each function $f_i(\mathbf{z})$ is convex and the set $Z^* = \arg\min_{\mathbf{z}\in\mathbb{R}^d} \sum_{i=1}^{m} f_i(\mathbf{z})$ is nonempty.*

(c) *The subgradients of each $f_i(\mathbf{z})$ are uniformly bounded, i.e., there is $L_i < \infty$ such that*

$$\|\mathbf{g}_i\|_2 \leq L_i \qquad \text{for all subgradients } \mathbf{g}_i \text{ of } f_i(\mathbf{z}) \text{ at all points } \mathbf{z} \in \mathbb{R}^d.$$

Then, the distributed subgradient-push method with the stepsize satisfying the conditions $\sum_{t=1}^{\infty} \alpha(t) = \infty$ and $\sum_{t=1}^{\infty} \alpha^2(t) < \infty$ has the following property

$$\lim_{t\to\infty} \mathbf{z}_i(t) = \mathbf{z}^* \qquad \text{for all } i \text{ and for some } \mathbf{z}^* \in Z^*.$$

Proof Idea

$$\begin{aligned}
\mathrm{w}_i(t+1) &= \sum_{j\in N_i^{\mathrm{in}}(t)} \frac{\mathrm{x}_j(t)}{d_j(t)},\\
y_i(t+1) &= \sum_{j\in N_i^{\mathrm{in}}(t)} \frac{y_j(t)}{d_j(t)},\\
\mathrm{z}_i(t+1) &= \frac{\mathrm{w}_i(t+1)}{y_i(t+1)},\\
\mathrm{x}_i(t+1) &= \mathrm{w}_i(t+1) - \alpha(t+1)\mathrm{g}_i(t+1).
\end{aligned}$$

Due to matrices $A(t)$ being column stochastic, we have

$$\frac{1}{n}\sum_{i=1}^{n} \mathrm{x}_i(t+1) = \frac{1}{n}\sum_{j=1}^{n} \mathrm{x}_j(t) - \frac{\alpha(t+1)}{n}\sum_{i=1}^{n} \mathrm{g}_i(t+1)$$

with $\mathrm{g}_i(t+1) \in \partial f_i(\mathrm{z}_i(t+1))$

Use the Key Lemma to approximate the differences $\mathrm{z}_i(t+1) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{x}_i(t+1)$ and exploit the Lipschitz continuity and convexity of f_i.

Key difficulty: non-linearity of the model; weak-ergodicity of the matrix sequence

Convergence Rate

Our second result gives explicit rate at which the objective function converges to its optimal value. As standard with subgradient methods, we will make two tweaks in order to get a convergence rate result:

(i) we take a stepsize which decays as $\alpha(t) = 1/\sqrt{t}$ (stepsizes which decay at faster rates usually produce inferior convergence rates),

(ii) each node i will maintain a convex combination of the values $\mathrm{z}_i(1), \mathrm{z}_i(2), \ldots$ for which the convergence rate will be obtained.

We then demonstrate that the subgradient-push converges at a rate of $O(\ln t/\sqrt{t})$. The result makes use of the matrix $A(t)$ that captures the weights used in the construction of $\mathrm{w}_i(t+1)$ and $y_i(t+1)$ in Eq. (3), which are defined by

$$A_{ij}(t) = \begin{cases} 1/d_j(t) & \text{whenever } j \in N_i^{\text{in}}(t), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Convergence Rate I

Proposition 2 *Suppose all the assumptions of Proposition 1 hold and, additionally, $\alpha(t) = 1/\sqrt{t}$ for $t \geq 1$. Moreover, suppose that every node i maintains the variable $\tilde{\mathbf{z}}_i(t) \in \mathbb{R}^d$ initialized at time $t = 1$ to $\tilde{\mathbf{z}}_i(1) = \mathbf{z}_i(1)$ and updated as*

$$\tilde{\mathbf{z}}_i(t+1) = \frac{\alpha(t+1)\mathbf{z}_i(t+1) + S(t)\tilde{\mathbf{z}}_i(t)}{S(t+1)},$$

where $S(t) = \sum_{s=0}^{t-1} \alpha(s+1)$. Then, we have that for all $t \geq 1$, $i = 1, \ldots, n$, and any $\mathbf{z}^* \in Z^*$,

$$F(\tilde{\mathbf{z}}_i(t)) - F(\mathbf{z}^*) \leq \frac{n}{2}\frac{\|\bar{\mathbf{x}}(0) - \mathbf{z}^*\|_1}{\sqrt{t}} + \frac{n}{2}\frac{\left(\sum_{i=1}^n L_i\right)^2}{4}\frac{(1 + \ln t)}{\sqrt{t}} + \frac{16}{\delta(1-\lambda)}\left(\sum_{i=1}^n L_i\right)\frac{\sum_{j=1}^n \|\mathbf{x}_j(0)\|_1}{\sqrt{t}} + \frac{16}{\delta(1-\lambda)}\left(\sum_{i=1}^n L_i^2\right)\frac{(1 + \ln t)}{\sqrt{t}}$$

where $\bar{\mathbf{x}}(0) = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i(0)$, and the scalars λ and δ are functions of the graph sequence $G(1), G(2), \ldots$, with the same properties properties as in Proposition 1.

The rate is $O(\ln t/\sqrt{t})$

Convergence Rate II

Theorem 3 *Suppose the assumptions of Proposition 1 hold and all functions f_i are strongly convex. Let $\alpha(t) = p/t$ for $t \geq 1$ where p is a constant (tuned). Moreover, suppose that every node i maintains the variable $\widehat{\mathbf{z}}_i(t) \in \mathbb{R}^d$ initialized at time $t = 1$ to $\widehat{\mathbf{z}}_i(1) = \mathbf{z}_i(1)$ and updated as*

$$\widehat{\mathbf{z}}_i(t+1) = \frac{t\mathbf{z}_i(t+1) + S(t)\widehat{\mathbf{z}}_i(t)}{S(t+1)},$$

where $S(t) = t(t-1)/2$. Then, we have that for all $t \geq 2$, $i = 1, \ldots, n$,

$$F\left(\widehat{\mathbf{z}}_i(t)\right) - F(\mathbf{z}^*) + \sum_{j=1}^{n} \mu_j \|\widehat{\mathbf{z}}_i(t) - \mathbf{z}^*\|^2 \leq \frac{80L}{t\delta}\frac{\lambda}{1-\lambda}\sum_{j=1}^{n} \|\mathbf{x}_j(0)\|_1 + \frac{p}{t}\sum_{j=1}^{n} L_j^2 + \frac{80pLn\sqrt{d}\,\max_i L_i}{t\delta}\frac{1}{1-\lambda}(1 + \ln(t-1))$$

where $\mathbf{z}^*$ is the solution of the problem, L_i is the maximum norm subgradient in a ball centered at origin, $L = \sum_{j=1}^{n} L_j$, and the scalars λ and δ are functions of the graph sequence $G(1), G(2), \ldots$, with the same properties properties as in Proposition 1.

The rate is $O(\ln t/t)$

Conclusion & Future work

- The rate results are by factor $\ln t$ worse than that of centralized algorithms
- Such scaling is expected as the graphs are "general" time-varying graphs
- Aspects for future studies
  - Scalability with network size
  - Dealing with constraints

Thank you