Modern Convex Optimization Methods

Mingyi Hong

University Of Minnesota

M. Hong would like to thank Dr. Prashant Khanduri for helping prepare the slides.

Outline

- **Alternating Direction Method of Multipliers (ADMM)**
  - Dual Ascent
  - Dual Decomposition
  - Augmented Lagrangian Method
  - ADMM: Convergence
- **Decentralized ADMM**
  - Assumptions
  - Convergence: Proof Sketch
- **EXTRA**
  - Assumptions
  - Convergence
- **Connection:** EXTRA and Primal-Dual Methods

ADMM Basics: Dual Ascent[1]

- **Problem**

$$\begin{aligned} \text{minimize}_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & A\boldsymbol{x} = b \end{aligned}$$

with $\boldsymbol{x} \in \mathbb{R}^n$, with $A \in \mathbb{R}^{p \times n}$ and $f : \mathbb{R}^n \to \mathbb{R}$ is convex

- **Dual Ascent**

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \underset{\boldsymbol{x}}{\operatorname{argmin}} \left\{ L(\boldsymbol{x}, \boldsymbol{y}^k) = f(\boldsymbol{x}) + \left(\boldsymbol{y}^k\right)^T (A\boldsymbol{x} - b) \right\} \\ \boldsymbol{y}^{k+1} &= \boldsymbol{y}^k + \alpha^k (A\boldsymbol{x}^{k+1} - b) \end{aligned}$$

with $\alpha^k > 0$ being the step-size

[1] Boyd, et al., Distributed optimization and statistical learning via the alternating direction method of multipliers. Now Publishers Inc, 2011.

Dual Decompoisition

- **Problem**

$$\begin{array}{ll} \text{minimize}_{\boldsymbol{x}} & \sum_{i=1}^{m} f_i(\boldsymbol{x}_i) \\ \text{subject to} & A\boldsymbol{x} = b \end{array}$$

  where $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m) \in \mathbb{R}^n$ with $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$ with $i \in [m]$ and

$$A = [A_1, \ldots, A_m] \text{ with } A_i \in \mathbb{R}^{p \times n_i} \Rightarrow A\boldsymbol{x} = \sum_{i=1}^{m} A_i \boldsymbol{x}_i$$

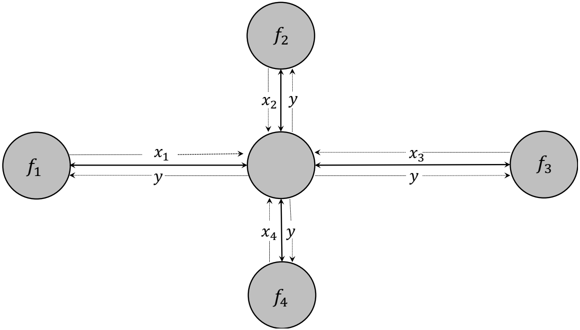
- **Dual Decompostion** (the Lagrangian function becomes separable over i)

$$\boldsymbol{x}_i^{k+1} = \operatorname*{argmin}_{\boldsymbol{x}_i} \left\{ L_i(\boldsymbol{x}_i, \boldsymbol{y}^k) = f_i(\boldsymbol{x}_i) + (\boldsymbol{y}^k)^T A_i \boldsymbol{x}_i - (1/m)(\boldsymbol{y}^k)^T b \right\}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \alpha^k (A\boldsymbol{x}^{k+1} - b)$$

Proporties: Dual Decomposition

- **Implementation**



Dual decomposition with a central server and $m = 4$.

- **Pro:** Decomposes across $\boldsymbol{x}$ for each $i \in [m]$
- **Con:** Requires strong convexity to ensure convergence
- **Solution:** Use Augmented Lagrangian method

ADMM Basics: Augmented Lagrangian Method

- **Problem:** For $\rho > 0$,

$$\begin{array}{ll} \text{minimize}_{\boldsymbol{x}} & f(\boldsymbol{x}) + \frac{\rho}{2}\|A\boldsymbol{x} - b\|^2 \\ \text{subject to} & A\boldsymbol{x} = b \end{array}$$

- **Augmented Lagrangian Method**

$$\boldsymbol{x}^{k+1} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \, \underbrace{\{L_\rho(\boldsymbol{x}, \boldsymbol{y}^k) = f(\boldsymbol{x}) + (\boldsymbol{y}^k)^T (A\boldsymbol{x} - b) + \frac{\rho}{2}\|A\boldsymbol{x} - b\|^2\}}_{\text{Augmented Lagrangian}}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \alpha^k (A\boldsymbol{x}^{k+1} - b)$$

- **Pro:** Better convergence
- **Con:** Does not decompose for $f(\boldsymbol{x}) = \sum_{i=1}^m f_i(\boldsymbol{x}_i)$
- **Solution:** ADMM blends decomposability of dual ascent with superior convergence properties of method of multipliers

Alternating Direction Method of Multipliers (ADMM)

- **Problem** Let $f : \mathbb{R}^{n_x} \to \mathbb{R}$ and $g : \mathbb{R}^{n_z} \to \mathbb{R}$ be convex

$$\begin{array}{rl} \text{minimize}_{\boldsymbol{x},\boldsymbol{z}} & f(\boldsymbol{x}) + g(\boldsymbol{z}) \\ \text{subject to} & A\boldsymbol{x} + B\boldsymbol{z} = c \end{array}$$

  with $\boldsymbol{x} \in \mathbb{R}^{n_x}$ and $\boldsymbol{z} \in \mathbb{R}^{n_z}$ and $A \in \mathbb{R}^{p \times n_x}$, $B \in \mathbb{R}^{p \times n_z}$

- **Augmented Lagrangian**

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{y}^T(A\boldsymbol{x} + B\boldsymbol{z} - c) + \frac{\rho}{2}\|A\boldsymbol{x} + B\boldsymbol{z} - c\|^2$$

- **ADMM**

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \operatorname*{argmin}_{\boldsymbol{x}} L_\rho(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{y}^k) \\ \boldsymbol{z}^{k+1} &= \operatorname*{argmin}_{\boldsymbol{z}} L_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{y}^k) \\ \boldsymbol{y}^{k+1} &= \boldsymbol{y}^k + \rho(A\boldsymbol{x}^{k+1} + B\boldsymbol{z}^{k+1} - c) \end{aligned}$$

Performance of ADMM

LASSO[2]: Given $b \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times p}$ solve:

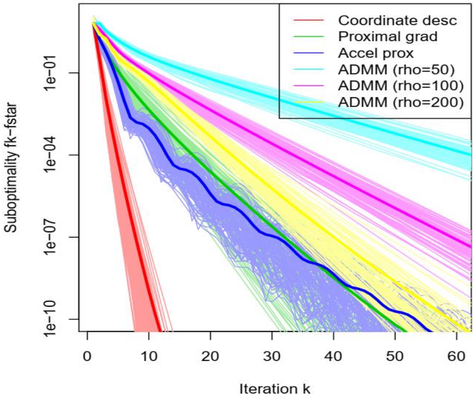
$$\min_{\boldsymbol{x}} \frac{1}{2}\|b - A\boldsymbol{x}\|^2 + \lambda\|\boldsymbol{x}\|_1$$

Rephrased in form suitable for ADMM as:

$$\min_{\boldsymbol{x},\boldsymbol{z}} \underbrace{\frac{1}{2}\|b - A\boldsymbol{x}\|^2}_{f(\boldsymbol{x})} + \underbrace{\lambda\|\boldsymbol{z}\|_1}_{g(\boldsymbol{z})} \quad \text{subject to} \quad \underbrace{\boldsymbol{x} = \boldsymbol{z}}_{\text{linear constraint}}$$

ADMM Comparison[3]

Comparison of various algorithms for lasso regression: 100 random instances with $n = 200$, $p = 50$



[3] Ryan Tibshirani, Convex Optimization, Lecture Slides, Fall 2018

Properties of ADMM

- **Slow** to converge to high accuracy
- Converges to **modest accuracy** within a few iterations
  - Sufficient for many applications
  - Well suited for large scale problems in machine learning and statistical estimation
- Computations can be **distributed** across multiple nodes

Assumptions

Assumption 1

The (extended-real-valued) functions $f : \mathbb{R}^{n_x} \to \mathbb{R} \cup \{+\infty\}$ *and* $g : \mathbb{R}^{n_z} \to \mathbb{R} \cup \{+\infty\}$ *are closed, proper, and convex.*

A function f satisfies Assumption 1 $\Longleftrightarrow$
epi $f = \{(\boldsymbol{x}, t) \in \mathbb{R}^n \times \mathbb{R} : f(\boldsymbol{x}) \leq t\}$ **closed, non-empty, convex set**

For simplicity, we will assume that both f and g differentiable. Otherwise, subgradient notation will be used.

Assumptions

Assumption 2

The unaugmented Lagrangian defined as:

$$L_0(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{y}^T(A\boldsymbol{x} + B\boldsymbol{z} - c)$$

has a saddle point.

There exists (x^*, z^*, y^*) such that (possibly non-unique)

$L_0(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{y}) \leq L_0(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{y}^*) \leq L_0(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}^*)$ **for all** $\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}$

Convergence

Under **Assumption 1** and **Assumption 2**, ADMM iterates satisfy:

- **Residual convergence:** The iterates approach feasibility, i.e., the residuals satisfy

$$r^k = A\boldsymbol{x}^k + B\boldsymbol{z}^k - c \to 0 \ \text{ as } \ k \to \infty$$

- **Objective convergence:** $f(\boldsymbol{x}^k) + g(\boldsymbol{z}^k) \to p^*$ as $k \to \infty$, where p^* is
$$p^* = \inf\{f(\boldsymbol{x}) + g(\boldsymbol{z}) : A\boldsymbol{x} + B\boldsymbol{z} - c = 0\}$$

- **Dual variable convergence:** $\boldsymbol{y}^k \to \boldsymbol{y}^*$ as $k \to \infty$ where $\boldsymbol{y}^*$ is a dual optimal value

Optimality of ADMM

Necessary and sufficient conditions for optimality (KKT conditions)

- **Primal Feasibility**

$$A\boldsymbol{x}^* + B\boldsymbol{z}^* - c = 0 \tag{1.1}$$

- **Stationarity**

$$0 = \nabla f(\boldsymbol{x}^*) + A^T\boldsymbol{y}^* \tag{1.2}$$

$$0 = \nabla g(\boldsymbol{z}^*) + B^T\boldsymbol{y}^* \tag{1.3}$$

- Using $\boldsymbol{z}^{k+1} = \operatorname{argmin}_z L_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{y}^k)$ **check!** that (1.3) is **always satisfied** by $\boldsymbol{z}^{k+1}$ and $\boldsymbol{y}^{k+1}$
- **To achieve optimality:** (1.1) and (1.2) need to be satisfied

Optimality of ADMM

- **Dual Residual:** Stationarity condition (1.2)
  Using $\boldsymbol{x}^{k+1} = \mathrm{argmin}_x\, L_\rho(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{y}^k)$, we get (**Check!**)

$$\underbrace{\rho A^T B(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k)}_{s^{k+1}} = \partial f(\boldsymbol{x}^{\boldsymbol{k+1}}) + A^T \boldsymbol{y}^{k+1}$$

  s^{k+1} is referred to a *dual residual*

- **Primal Residual:** Denote

$$r^{k+1} = A\boldsymbol{x}^{k+1} + B\boldsymbol{z}^{k+1} - c$$

  as the *primal residual*

Goal:

$r^k, s^k \to 0 \;\Rightarrow\; (1.1)$ **and** (1.2) **hold** $\;\Rightarrow\;$ **Optimality of ADMM**

Proof Sketch: ADMM I

Define: Lyupanov Function:

$$V^k = \frac{1}{\rho}\|\boldsymbol{y}^k - \boldsymbol{y}^*\|^2 + \rho\|B(\boldsymbol{z}^k - \boldsymbol{z}^*)\|^2$$

The proof relies on three basic inequalities:

- **Inequality I**:

$$V^{k+1} \leq V^k - \rho\|r^{k+1}\|^2 - \rho\|B(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k)\|^2$$

- **Inequality II**:

$$p^{k+1} - p^* \leq -(\boldsymbol{y}^{k+1})^T r^{k+1} - \rho(B(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k))^T(-r^{k+1} + B(\boldsymbol{z}^{k+1} - \boldsymbol{z}^*))$$

- **Inequality III**:

$$p^* - p^{k+1} \leq \boldsymbol{y}^{*T} r^{k+1}$$

Proof Sketch: ADMM II

- Iterating **Inequality I** above, we get

$$\rho \sum_{k=0}^{\infty} \left(\|r^{k+1}\|^2 + \|B(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k)\|^2 \right) \leq V_0$$

  this implies $r^k \to 0$ and $B(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k) \to 0$ as $k \to \infty$
  - $B(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k) \to 0$ further implies that the *dual residual* $s^{k+1} \to 0$ (follows from definition of s^k)
- **Inequality II** and **Inequality III** imply $p^k \to p^*$ as $k \to \infty$, i.e., objective convergence

Inequalities II and III are used to derive Inequality I

Distributed ADMM

- **Problem**

$$\text{minimize}_{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_m,z} \sum_{i=1}^{m} f_i(\boldsymbol{x}_i) \quad \text{subject to} \quad \boldsymbol{x}_i = \boldsymbol{z}, i = 1, \ldots, m$$

- ADMM steps can be **distributed** across m nodes:

$$\begin{aligned}
\boldsymbol{x}_i^{k+1} &= \operatorname*{argmin}_{\boldsymbol{x}_i} f_i(\boldsymbol{x}_i) + \frac{\rho}{2}\|\boldsymbol{x}_i - \boldsymbol{z}^k + \boldsymbol{u}_i^k\|^2 \\
\boldsymbol{z}^{k+1} &= \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{x}_i^{k+1} + \boldsymbol{u}_i^k) \\
\boldsymbol{u}_i^{k+1} &= \boldsymbol{u}_i^k + \boldsymbol{x}_i^{k+1} - \boldsymbol{z}^{k+1}
\end{aligned}$$

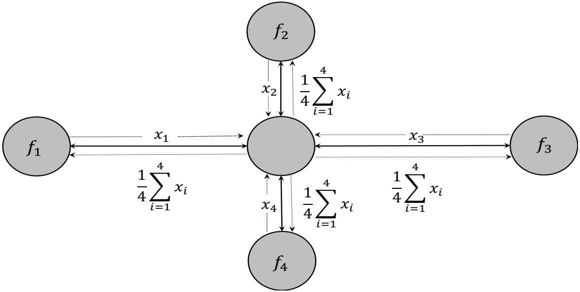
Distributed Implementation

- Simple manipulation yields (**Check!**):

$$x_i^{k+1} = \underset{x_i}{\operatorname{argmin}}\, f_i(x_i) + \frac{\rho}{2}\|x_i - \frac{1}{m}\sum_{i=1}^{m} x_i^k + u_i^k\|^2$$

$$u_i^{k+1} = u_i^k + x_i^{k+1} - \frac{1}{m}\sum_{i=1}^{m} x_i^{k+1}$$

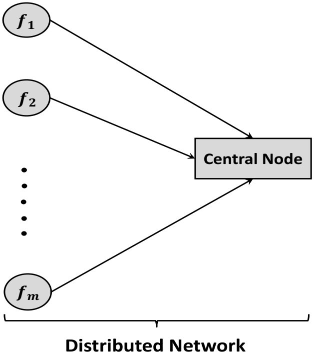
- **Parallel Updates**

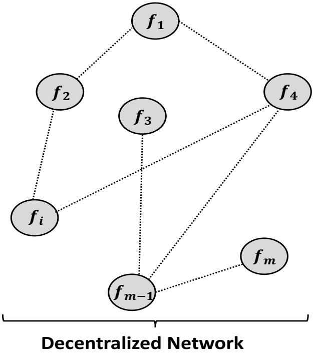


Distributed ADMM with $m = 4$.

Decentralized Setup

- **Distributed ADMM:** Central server for sharing of iterates
  - **Congestion**, **Privacy concerns**, **Single point of failure**
- **Decentralized ADMM:** No central server, nodes exchange information with their immediate neighbors
  - **No risk of network congestion**, **Better privacy**, **Robust to individual node failures**





Decentralized ADMM[4]

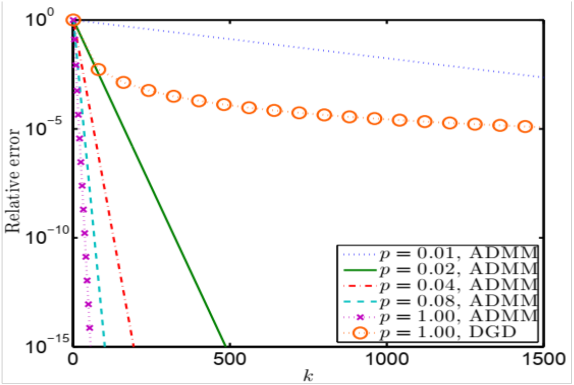
- **Problem**

$$\text{minimize}_{\boldsymbol{x}} \ \sum_{i=1}^{m} f_i(\boldsymbol{x})$$

- **Model**
  - m agents/nodes present in the network
  - Each agent has access to local function $f_i : \mathbb{R}^n \to \mathbb{R}$
- **Network**
  - m agents connected via E edges ($2E$ arcs)
  - **Symmetric directed graph:** $\mathcal{G} = \{\mathcal{V}, \mathcal{A}\}$
  - **Set of vertices:** $\mathcal{V}$ with $|\mathcal{V}| = m$, **Set of Arcs:** $\mathcal{A}$ with $|\mathcal{A}| = 2E$

[4]Shi, et al., On the linear convergence of the ADMM in decentralized consensus optimization, IEEE Transactions on Signal Processing 62.7 (2014): 1750-1761.

ADMM vs DGD



Relative error for least squares consensus ADMM vs DGD with step-size $\frac{1}{k^{1/3}}$

Problem Formulation

- **Reformulation suitable for ADMM**

$$\text{minimize}_{\{\boldsymbol{x}_i\},\{\boldsymbol{z}_{ij}\}} \sum_{i=1}^{m} f_i(\boldsymbol{x}_i)$$

$$\text{subject to } \boldsymbol{x}_i = \boldsymbol{z}_{ij}, \boldsymbol{x}_j = \boldsymbol{z}_{ij}, \forall (i,j) \in \mathcal{A}$$

- **Standard ADMM form**

$$\text{minimize}_{\boldsymbol{x},\boldsymbol{z}} \; f(\boldsymbol{x}) + g(\boldsymbol{z})$$

$$\text{subject to } A\boldsymbol{x} + B\boldsymbol{z} = 0$$

with $\boldsymbol{x} \in \mathbb{R}^{mn}$, $\boldsymbol{z} \in \mathbb{R}^{2En}$ and $g(\boldsymbol{z}) = 0$

- **Matrices:** $A = [A_1; A_2]$ and $B = [-I_{2En}; -I_{2En}]$

Check! Dimensions and structure of A?

ADMM Updates

- **Augmented Lagrangian:** Dual variable $\boldsymbol{y} \in \mathbb{R}^{4En}$

$$L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) = f(\boldsymbol{x}) + \boldsymbol{y}^T(A\boldsymbol{x} + B\boldsymbol{z}) + \frac{\rho}{2}\|A\boldsymbol{x} + B\boldsymbol{z}\|^2$$

- **ADMM updates**

$$\begin{aligned}
\boldsymbol{x} - \text{update}: &\quad \nabla f(\boldsymbol{x}^{k+1}) + A^T\boldsymbol{y}^k + \rho A^T(A\boldsymbol{x}^{k+1} + B\boldsymbol{z}^k) = 0 \\
\boldsymbol{z} - \text{update}: &\quad B^T\boldsymbol{y}^k + \rho B^T(A\boldsymbol{x}^{k+1} + B\boldsymbol{z}^{k+1}) = 0 \\
\boldsymbol{y} - \text{update}: &\quad \boldsymbol{y}^{k+1} - \boldsymbol{y}^k - \rho(A\boldsymbol{x}^{k+1} + B\boldsymbol{z}^{k+1}) = 0
\end{aligned}$$

Goal: Manipulate update equations above to yield decentralized implementation of ADMM!

Manipulating ADMM Updates I

- Multiplying $\boldsymbol{y}$ update by A^T and adding with the $\boldsymbol{x}$ update
- Multiplying $\boldsymbol{y}$ update by B^T and adding with the $\boldsymbol{z}$ update
- We get

$$\begin{aligned} \nabla f(\boldsymbol{x}^{k+1}) + A^T\boldsymbol{y}^{k+1} + \rho A^T B(\boldsymbol{z}^k - \boldsymbol{z}^{k+1}) &= 0 \\ B^T\boldsymbol{y}^{k+1} &= 0 \\ \boldsymbol{y}^{k+1} - \boldsymbol{y}^k - \rho(A\boldsymbol{x}^{k+1} + B\boldsymbol{z}^{k+1}) &= 0 \end{aligned}$$

- Take $\boldsymbol{y} = [\beta; \gamma]$ with $\beta, \gamma \in \mathbb{R}^{2En}$
- **Recall:** $B = [-I_{2En}; -I_{2En}]$, therefore the second equation implies

$$\beta^{k+1} = -\gamma^{k+1}$$

Manipulating ADMM Updates II

Using $\beta^{k+1} = -\gamma^{k+1}$ from previous slide:

- With **initialization** $\beta^0 = -\gamma^0$ and $\boldsymbol{z}^0 = \frac{1}{2} A_+^T \boldsymbol{x}^0$
- **With some manipulation (check!),** the update equations can be equivalently written as:

$$\begin{aligned} \nabla f(\boldsymbol{x}^{k+1}) + A_- \beta^{k+1} - \rho A_+ (\boldsymbol{z}^k - \boldsymbol{z}^{k+1}) &= 0 \\ \beta^{k+1} - \beta^k - \frac{\rho}{2} A_-^T \boldsymbol{x}^{k+1} &= 0 \\ \frac{1}{2} A_+^T \boldsymbol{x}^k - \boldsymbol{z}^k &= 0 \end{aligned} \tag{1.4}$$

with, $A_+ = A_1^T + A_2^T$ and $A_- = A_1^T - A_2^T$ (**Recall:** $A = [A_1; A_2]$)

Equation (1.4) will be used in the analysis of Decentralized ADMM!

Finally: Algorithm Updates

- With **initialization** $\beta^0 = -\gamma^0$ and $\boldsymbol{z}^0 = \frac{1}{2} A_+^T \boldsymbol{x}^0$
- **With some more manipulation (check!)**, the update equations can be further simplified to

$$
\begin{aligned}
\boldsymbol{x} - \text{update}: \nabla f(\boldsymbol{x}^{k+1}) + \alpha^k + 2\rho W \boldsymbol{x}^{k+1} - \rho L_+ \boldsymbol{x}^k &= 0 \\
\alpha - \text{update}: \qquad \alpha^{k+1} - \alpha^k - \rho L_- \boldsymbol{x}^{k+1} &= 0
\end{aligned}
$$

$L_+ = \frac{1}{2} A_+ A_+^T$, $L_- = \frac{1}{2} A_- A_-^T$, $W = \frac{1}{2}(L_+ + L_-)$ and $\alpha = A_- \beta$.

- Matrices A_+, A_-, L_+, L_- and W capture the **network topology**
  - A_+ and A_-: Unoriented and oriented **incidence** matrices
  - L_+ and L_-: Signless and signed **Laplacian** matrices
  - W: **Degree** matrix

Algorithm: Fully Decentralized Implementation

Algorithm updates translate to the following updates for the ith node

Just using definitions of W, L_+ and L_-

Decentralized Consensus Optimization Based on ADMM

- **Input:** Functions f_i; initialize $\boldsymbol{x}_i^0 = 0$, $\alpha_i^0 = 0$, set $\rho > 0$
- **For** $k = 0, 1, \ldots,$ every agent i **do**
- Update $\boldsymbol{x}_i^{k+1}$ by solving
  $$\nabla f_i(\boldsymbol{x}_i^{k+1}) + \alpha_i^k + 2\rho|\mathcal{N}_i|\boldsymbol{x}_i^{k+1} - \rho\Big(|\mathcal{N}_i|\boldsymbol{x}_i^k - \textstyle\sum_{j\in\mathcal{N}_i}\boldsymbol{x}_j^k\Big) = 0$$
- Update $\alpha_i^{k+1} = \alpha_i^k + \rho\Big(|\mathcal{N}_i|\boldsymbol{x}_i^{k+1} - \sum_{j\in\mathcal{N}_i}\boldsymbol{x}_j^{k+1}\Big)$
- **End For**

Assumptions

Assumption 3 (Strong Convexity)

Local objective functions are strongly convex, i.e, for each agent $i \in [m]$ given any $\boldsymbol{x_1}, \boldsymbol{x_2} \in \mathbb{R}^n$:

$$\langle \nabla f_i(\boldsymbol{x}_1) - \nabla f_i(\boldsymbol{x}_2), \boldsymbol{x}_1 - \boldsymbol{x}_2 \rangle \geq m_f \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|^2 \text{ with } m_{f_i} > 0$$

Assumption 3 implies f is also strongly convex, with parameter $m_f = \min_i m_{f_i}$

Assumption 4 (Lipschitz Continuity)

The gradients of the local objective functions are Lipschitz continuous, i.e. for each agent $i \in [m]$ given any $\boldsymbol{x_1}, \boldsymbol{x_2} \in \mathbb{R}^n$:

$$\|\nabla f_i(\boldsymbol{x}_1) - \nabla f_i(\boldsymbol{x}_2)\| \leq M_{f_i} \|\boldsymbol{x}_1 - \boldsymbol{x}_2\| \text{ with } M_{f_i} > 0$$

Assumption 4 implies f is also Lipschitz continuous with $M_f = \max_i M_{f_i}$

Definitions

Definition 1.1 (Q-Linear Convergence)

A sequence $\boldsymbol{y}^k$, Q-linearly converges to a point y^* if there exist a number $\sigma \in (0, 1)$ such that

$$\lim_{k\to\infty} \frac{\|\boldsymbol{y}^{k+1} - \boldsymbol{y}^*\|}{\|\boldsymbol{y}^k - \boldsymbol{y}^*\|} = \sigma$$

Main Result

Define:

$\boldsymbol{u}^k = [\boldsymbol{z}^k; \beta^k]$, $\boldsymbol{u}^* = [\boldsymbol{z}^*; \beta^*]$ and $G = [\rho I_{2En} \; 0_{2En}; 0_{2En} \; \frac{1}{\rho} I_{2En}]$

Theorem 1.2

Under Assumptions 3 and 4, and **proper initialization**, *for any* $\mu > 0$, $\boldsymbol{u}^k$ *is Q-linearly convergence to* u^* *w.r.t. G-norm:*

$$\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^*\|_G^2 \leq \frac{1}{1+\delta} \|\boldsymbol{u}^k - \boldsymbol{u}^*\|_G^2$$

for $\delta > 0$, *with* δ *dependent on the network and function parameters. Also,* $\boldsymbol{x}^k$ *converges as:*

$$\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2 \leq \frac{1}{m_f} \|\boldsymbol{u}^k - \boldsymbol{u}^*\|_G^2$$

Proper initialization: Dual variable β is in the column space of A_-^T

Proof Sketch: I

Step 1: Use update equations (1.4), KKT conditions, and strong convexity to show:

$$\|\boldsymbol{u}^k - \boldsymbol{u}^{k+1}\|_G^2 + m_f \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2 \leq \|\boldsymbol{u}^k - \boldsymbol{u}^*\|_G^2 - \|\boldsymbol{u}^{k+1} - \boldsymbol{u}^*\|_G^2$$

I. First, subtract update equations (1.4) and the KKT conditions
II. Use strong convexity, the equations derived in (I) above, and the definition of G to reach the conclusion

Proof Sketch II

Step 2: Prove: $\|\boldsymbol{u}^k - \boldsymbol{u}^{k+1}\|_G^2 + m_f\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2 \geq \delta\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^*\|_G^2$

Note: Combining this with the conclusion of Step I leads to the main result of Theorem 1.2

1. Definition of $\boldsymbol{u}$ and G implies equation above is equivalent to:

$$\rho\|\boldsymbol{z}^{k+1} - \boldsymbol{z}^k\|^2 + \frac{1}{\rho}\|\beta^{k+1} - \beta^k\|^2 + m_f\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^*\|^2 \geq \delta\rho\|\boldsymbol{z}^{k+1} - \boldsymbol{z}^*\|^2 + \frac{\delta}{\rho}\|\beta^{k+1} - \beta^*\|^2$$

2. Bound $\|\boldsymbol{z}^{k+1} - \boldsymbol{z}^*\|^2$: Use (1.4) & KKT conditions (**Simple!**)
3. Bound $\|\beta^{k+1} - \beta^*\|^2$: Next!

Proof Sketch III

Step 3: Finally, upper bound: $\|\beta^{k+1} - \beta^*\|^2$

1. Use Lipschitz continuity of local functions, (1.4) and KKT conditions
2. Finally, utilize initialization $\underbrace{\beta \textbf{ lies in the column space of } A_-^T}_{\text{Crucial Step}}$

Conclude, Step II and combine with Step I to retrieve the statement of Theorem 1.2.

Note: R-linear convergence of sequence $\boldsymbol{x}^k$ follows from the statement of **Step I**

Takeaway from Decentralized ADMM

- A general consensus optimization problem can be **reformulated** in ADMM framework
- Global **linear convergence** for strongly convex objectives
- Convergence guarantees capture the effect of
  - **Topology** of the network
  - **Condition number** of the objective function
- **Other variations**
  - Inexact Consensus[5]
  - Asynchronous with convex objectives[6]
  - Linearize the objective[7]

---

[5] Chang et al., Multi-agent distributed optimization via inexact consensus ADMM, IEEE Transactions on Signal Processing 2014

[6] Wei et al., On the O(1/K) convergence of asynchronous distributed alternating direction method of multipliers, IEEE GlobalSIP, 2013.

[7] Ling et al., DLM: Decentralized Linearized Alternating Direction Method of Multipliers, IEEE Transactions on Signal Processing 2015

EXTRA[8]

- **Problem**:

$$\text{minimize}_{\boldsymbol{x}} \; f(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m} f_i(\boldsymbol{x}) \tag{1.5}$$

with $\boldsymbol{x} \in \mathbb{R}^n$ and $f_i : \mathbb{R}^n \to \mathbb{R}$

  - Set of m nodes
  - Each node $i \in [m]$ has access to **convex** function f_i

Goal: Solve the problem in a decentralized fashion!

[8] Shi et al., Extra: An exact first-order algorithm for decentralized consensus optimization, SIAM Journal on Optimization 25.2 (2015): 944-966.

DGD vs ADMM vs EXTRA

- **Recall from last lecture: DGD** to solve (1.5)
  - **Fixed** step-size: **Neighborhood** convergence
  - **Decreasing** step-size: **Slow** convergence
- **ADMM**
  - Can be **computationally expensive**
  - Requires **successive minimizations**
- **EXTRA:** Exact First-Order Algorithm
  - **Exact** convergence with **fixed** step-size
  - **Faster** than **DGD**
  - Convex objectives: $O(1/k)$, Strongly convex objectives: **Linear**

EXTRA: Utilizes the gradient estimate of the previous iteration.

Assumptions

Assumption 5 (Mixing Matrix)

Consider a connected network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with $\mathcal{V} = [m]$ agents and a set of undirected edges $\mathcal{E}$. The mixing matrices $W = [w_{ij}] \in \mathbb{R}^{m \times m}$ and $\tilde{W} = [\tilde{w}_{ij}] \in \mathbb{R}^{m \times m}$ satisfy

1. *(Decentralized) If $i \neq j$ and $(i, j) \notin \mathcal{E}$, then $w_{ij} = 0$.*
2. *(Symmetry) $W = W^T$,*
3. *(Null Space) null$\{I - W\} \supseteq$ span$\{\mathbf{1}\}$*

- Eigenvalues of W lie in $(-1, 1]$
- Slightly simplified assumption, for full details see the original paper

Assumption

Assumption 6 (Convex objective with Lipschitz continuous gradient)

Objective functions f_i are proper closed convex and Lipschitz differentiable:

$$\|\nabla f_i(\boldsymbol{x}_1) - \nabla f_i(\boldsymbol{x}_2)\| \leq M_{f_i} \|\boldsymbol{x}_1 - \boldsymbol{x}_2\| \text{ for all } \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^n$$

where $M_{f_i} > 0$

- Function $f(\mathbf{x}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}_i)$ is M_f-Lipschitz with $M_f = \max_i\{M_{f_i}\}$

Assumption 7 (Solution Existence)

Problem (1.5) *has a non-empty set of optimal solutions:* $\mathcal{X}^* \neq \emptyset$

Algorithm: EXTRA

Algorithm: **EXTRA**

- Choose $\alpha > 0$ and mixing matrices $W \in \mathbb{R}^{m\times m}$ and $\tilde{W} \in \mathbb{R}^{m\times m}$
- **For** $i = 1, 2, \ldots, m$
- Pick any $\boldsymbol{x}_i^0 \in \mathbb{R}^n$
- $\boldsymbol{x}_i^1 = \sum_{j=1}^m w_{ij}\boldsymbol{x}_j^0 - \alpha\nabla f_i(\boldsymbol{x}_i^0)$
- **For** $k = 0, 1, \ldots$ **do**
- $$\begin{aligned}\boldsymbol{x}_i^{k+2} = \boldsymbol{x}_i^{k+1} + \textstyle\sum_{j=1}^m w_{ij}\boldsymbol{x}_j^{k+1} - \sum_{j=1}^m (w_{ij}+1)/2\boldsymbol{x}_j^k \\ -\alpha\big[\nabla f_i(\boldsymbol{x}_i^{k+1}) - \nabla f_i(\boldsymbol{x}_i^k)\big]\end{aligned}$$
- **End For**
- **End For**

Algorithm: EXTRA

Stacking the iterates together, we have the following compact form

$$\boldsymbol{x}^{k+2} = \boldsymbol{x}^{k+1} + W\boldsymbol{x}^{k+1} - \frac{I+W}{2}\boldsymbol{x}^{k} - \alpha(\nabla f(\boldsymbol{x}^{k+1}) - \nabla f(\boldsymbol{x}^{k})) \tag{1.6}$$

Convergence

- Denote $\mathbf{x} \in \mathbb{R}^{m\times n}$ and $\mathbf{x}^* \in \mathbb{R}^{m\times n}$ as a matrices with ith rows $\boldsymbol{x}_i^T$ and $\boldsymbol{x}_i^{*T}$, respectively
- Introduce auxiliary sequence:

$$\mathbf{q}^k = \sum_{t=0}^{k} U\mathbf{x}^t \quad \text{with} \quad U = (\tilde{W} - W)^{1/2}$$

and for each k

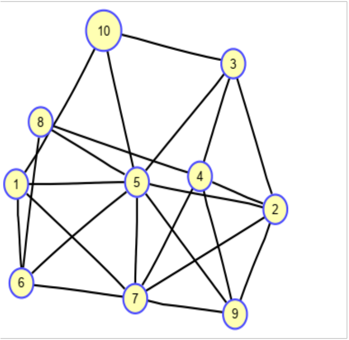
$$\boldsymbol{z}^k = (\mathbf{q}^k; \mathbf{x}^k), \quad \boldsymbol{z}^* = (\mathbf{q}^k; \mathbf{x}^*), \quad G = (I \quad \mathbf{0}; \mathbf{0} \quad \tilde{W})$$

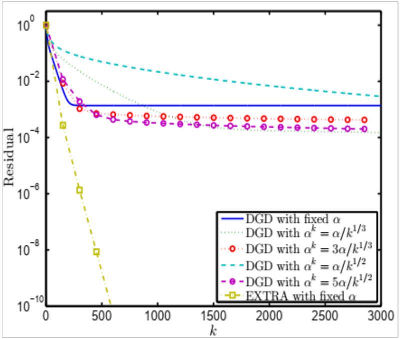
Theorem 1.3

Under Assumptions 5, 6 and 7 if α satisfies $0 < \alpha < \frac{2\lambda_{min}(\tilde{W})}{M_f}$, then

$$\frac{1}{k}\sum_{t=1}^{k} \|\mathbf{z}^t - \mathbf{z}^{t-1}\|_G^2 = O\left(\frac{1}{k}\right).$$

EXTRA vs DGD: Performance





Performance of EXTRA vs DGD for least squares consensus problem

Connection: Extra and Primal-Dual Methods[9] [10]

- **Notations:** Assume $n = 1$, denote $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)^T \in \mathbb{R}^m$
- $W = \mathbb{R}^{n \times n}$ mixing matrix
- **Goal** is to solve (1.5), repeated here for convenience

$$\text{minimize}_{\boldsymbol{x}} \ f(\boldsymbol{x}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}) \tag{1.7}$$

- **EXTRA** update rule

$$\begin{aligned} \boldsymbol{x}^1 &= W\boldsymbol{x}^0 - \alpha \nabla f(\boldsymbol{x}^0) \\ \boldsymbol{x}^{k+2} &= W\boldsymbol{x}^{k+1} - \alpha \nabla f(\boldsymbol{x}^{k+1}) - (I + W)/2\boldsymbol{x}^k + \alpha \nabla f(\boldsymbol{x}^k) \end{aligned} \tag{1.8}$$

[9] Jakovetić, A unification and generalization of exact distributed first-order methods, IEEE Transactions on Signal and Information Processing over Networks, 2018.

[10] Mokhtari et al., DSA: Decentralized double stochastic averaging gradient algorithm, Journal of Machine Learning Research, vol. 17, pp. 1-35, 2016.

Connection: Extra and Primal-Dual Methods

- **Equivalent Reformulation:** Let us denote $\mathcal{L} = I - \mathcal{W}$
- **Goal** is to minimize

$$\text{minimize}_{\boldsymbol{x}} \; f(\boldsymbol{x}) = \sum_{i=1}^{m} f_i(\boldsymbol{x}_i) \;\; \text{subject to} \;\; \underbrace{\frac{1}{\alpha}\mathcal{L}^{1/2}\boldsymbol{x} = 0}_{\text{Ensures consensus}} \qquad (1.9)$$

- $\mathcal{L}^{1/2}\boldsymbol{x} = 0 \Longleftrightarrow \boldsymbol{x}_1 = \boldsymbol{x}_2 =, \ldots, = \boldsymbol{x}_m$ (**Check!**)

This implies that (1.7) is **equivalent** to (1.9)

Connection: Extra and Primal-Dual Methods

- **Augmented Lagrangian:** From (1.9), penalty parameter $\rho = \alpha$,

$$L_\alpha(\boldsymbol{x}, \boldsymbol{y}) = f(\boldsymbol{x}) + \frac{1}{\alpha}\boldsymbol{y}^T \mathcal{L}^{1/2}\boldsymbol{x} + \frac{1}{2\alpha}\boldsymbol{x}^T \mathcal{L}\boldsymbol{x}$$

- Using the notation: $\boldsymbol{u}^k = \frac{1}{\alpha}\mathcal{L}^{1/2}\boldsymbol{y}^k$, we can write the **primal-dual** update as (**Check!**)

$$\begin{aligned} \boldsymbol{x}^{k+1} &= \boldsymbol{x}^k - \alpha\left(\frac{1}{\alpha}\mathcal{L}\boldsymbol{x}^k + \nabla f(\boldsymbol{x}^k) + \boldsymbol{u}^k\right) \\ \boldsymbol{u}^{k+1} &= \boldsymbol{u}^k + \frac{1}{\alpha}\mathcal{L}\boldsymbol{x}^{k+1} \end{aligned} \tag{1.10}$$

Lemma 1.4

The sequence $\{\boldsymbol{x}^k\}$ *generated by* **EXTRA** (1.8), *with initialization* $\boldsymbol{x}_i^0 = \boldsymbol{x}_j^0$, $\forall i, j$, *is same as the sequence generated by* **primal-dual** *iterations* (1.10) *with the same initialization* $\boldsymbol{x}_i^0$ $\forall i \in [m]$ *and* $\boldsymbol{u}^0 = 0$.

Story Thus Far

- We studied **Primal-Dual Methods: ADMM**
  - Distributed/Decentralized implementations of ADMM
  - Assumptions and Convergence Guarantees
- We studied **EXTRA**
  - Limitations of DGD
  - Assumptions and Performance
- Finally, we studied **connection** between EXTRA and Primal-Dual methods
  - Specifically, we noted that EXTRA can be developed with a specific Primal-Dual construction

Next: Gradient Tracking, Push-Sum methods

Gradient Tracking[11]

- Alternate approach to **EXTRA**
- **Goal:** To solve (1.7) in a decentralized fashion
- **Gradient Tracking Based Algorithm**
  - **Idea:** Maintain an iterative estimate of the true gradient
- Static undirected networks
- **Convex** and **Strongly** convex objectives
  - **Sublinear** convergence of $O(1/k)$ for convex objectives
  - **Linear** convergence for strongly-convex objectives

[11]Guannan et al., Harnessing smoothness to accelerate distributed optimization, IEEE Transactions on Control of Network Systems, 2017.

Assumptions

Assumption 8 (Mixing Matrix, W)

Graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, consensus weight matrix $W = [w_{ij}] \in \mathbb{R}^{m \times m}$:

1. *For $i, j \in \mathcal{E}$, we have $w_{ij} > 0$ other wise $w_{ij} = 0$*
2. *W is doubly stochastic*

Assumption 9 (Convexity and Lipschitz smoothness)

Local functions f_i for all $i \in [m]$ are convex and M_f-smooth

Assumption 10 (Strong Convexity)

Local functions f_i for all $i \in [m]$ are m_f-strongly convex

Recall: Assumptions 3, 4 and 6 used earlier

Algorithm: Gradient Tracking

Algorithm: Gradient Tracking for Decentralized Optimization

- **Initialize:** $\boldsymbol{x}_i^0$ and $\boldsymbol{g}_i^0 = \nabla f_i(x_i^0)$
- **For** $k = 0, 1, \ldots$ **do**
- Iterate update

$$\boldsymbol{x}_i^{k+1} = \sum_{j=1}^{m} w_{ij} \boldsymbol{x}_j^k - \alpha \boldsymbol{g}_i^k$$

- Descent direction update

$$\boldsymbol{g}_i^{k+1} = \sum_{j=1}^{m} w_{ij} \boldsymbol{g}_i^k + \nabla f_i(\boldsymbol{x}_i^{k+1}) - \nabla f_i(\boldsymbol{x}_i^k)$$

- **End For**

Convergence of Gradient Tracking Algorithm

Theorem 1.5

Under Assumptions 8 and 9, we have for all $i \in [m]$

$$f(\hat{\boldsymbol{x}}_i^{k+1}) - f^* \leq O\left(\frac{1}{k}\right)$$

where $\hat{\boldsymbol{x}}_i^{k+1}$ *is a running average of iterates for* i*th agent*

Choice of step-size α depends on m_f and W

Comparison to EXTRA

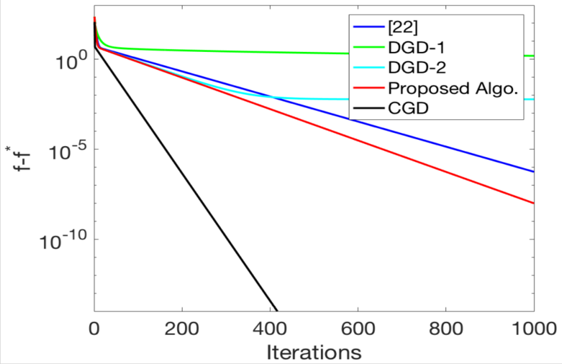
Pros:

- Easy to extended to many centralized methods
- Achieves optimality for both residuals and the objective error which is a more direct measure of optimality
  - **EXTRA** without strong convexity, achieved convergence in terms of the optimality residuals (see Theorem 1.3)

Con:

- Step-size depends on W, whereas for **EXTRA** it is independent of W

Performance: Gradient Tracking vs EXTRA and DGD



Performance of **Gradient tracking** vs **EXTRA** [22] and DGD for linear regression problem. **DGD** 1 is **DGD** with fixed step-size and **DGD** 2 is **DGD** with vanishing step-size; **CGD** is the centralized Gradient Descent

(Sub) Gradient Push Methods[12]

- Based on **Push-Sum** algorithm
- **Goal:** To solve (1.7) in a decentralized fashion
  - **Time-Varying** and **Directed** Networks
  - Does not assume smoothness: Only **bounded subgradients**

Assumption 11 (Uniform Strong Connectivity)

Graph $\mathcal{G}^k = \{\mathcal{V}, \mathcal{E}^k\}$

- *Vertex set:* $\mathcal{V}$ *with* $|\mathcal{V}| = m$
- *Edge set at time* k*:* $\mathcal{E}^k$
- $\mathcal{G}^k$ *is* **uniformly strongly connected**, *i.e.,* $\mathcal{E}_B^k = \bigcup_{i=kB}^{(k+1)B-1} \mathcal{E}^i$ *is strongly connected for every* $k \geq 0$

[12] Nedić et al., Distributed optimization over time-varying directed graphs." IEEE Transactions on Automatic Control, 2014.

Algorithm

- **Notation**
  - **Neighbourhoods:** $\mathcal{N}_{i,\text{in}}^k = \{j : (j,i) \in \mathcal{E}^k\} \cup \{i\}$ is the in, and $\mathcal{N}_{i,\text{out}}^k = \{j : (i,j) \in \mathcal{E}^k\} \cup \{i\}$ is the out neighbourhood
  - **Out-Degree:** $d_i^k = |\mathcal{N}_{i,\text{out}}^k|$
- **Subgradient:** $\boldsymbol{g}_i^k$ is the (sub)gradient of the function f_i at $\boldsymbol{x}_i^{k+1}$

Assumption 12 (Convex Bounded Subgradient)

1. *Each function f_i is convex over $\mathbb{R}^n$ and the set $\mathcal{X}^* = \operatorname{argmin}_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x})$ is non-empty*
2. *Subgradients $\boldsymbol{g}_i$ are bounded, i.e., $\|\boldsymbol{g}_i\| \leq L_i$*

Algorithm: (Sub) Gradient Push

Each node $i \in [m]$ maintains variables $\boldsymbol{y}_i^k, \boldsymbol{z}_i^k \in \mathbb{R}^n$ and scalar u_i^k

Algorithm: (Sub) Gradient Push

- **Initialize:** $\boldsymbol{z}_i^0$ and $u_i^0 = 1$ for all $i \in [m]$
- **For** $k = 0, 1, \ldots,$ and for all $i \in [m]$ **do**

$$\boldsymbol{y}_i^{k+1} = \sum_{j \in \mathcal{N}_{i,\text{in}}^k} \frac{\boldsymbol{z}_j^k}{d_j^k}$$

$$u_i^{k+1} = \sum_{j \in \mathcal{N}_{i,\text{in}}^k} \frac{u_j^k}{d_j^k}$$

$$\boldsymbol{x}_i^{k+1} = \frac{\boldsymbol{y}_i^{k+1}}{y_i^{k+1}}$$

$$\boldsymbol{z}_i^{k+1} = \boldsymbol{y}_i^{k+1} - \alpha^{k+1} \boldsymbol{g}_i^{k+1}$$

- **End For**

Convergence: (Sub) Gradient Push

Theorem 1.6

Under Assumptions 11 and 12 and with non-increasing step-sizes $\alpha^k > 0$ satisfying: $\sum_{k=1}^{\infty} \alpha^k = \infty$ and $\sum_{k=1}^{\infty} (\alpha^k)^2 < \infty$:

$$\lim_{k \to \infty} \boldsymbol{x}_i^k = \boldsymbol{x}^*$$

for some $\boldsymbol{x}^* \in \mathcal{X}^*$

Theorem 1.7

With $\alpha^k = \frac{1}{\sqrt{k}}$ and each node maintaining $\tilde{\boldsymbol{x}}_i^{k+1} = \frac{\alpha^{k+1} \boldsymbol{x}_i^{k+1} + S^k \tilde{\boldsymbol{x}}^k}{S^{k+1}}$ for $k > 0$ with $S^0 = 0$ and $S^k = \sum_{s=1}^{k} \alpha^k$:

$$f(\tilde{\boldsymbol{x}}_i^{k+1}) - f(\boldsymbol{x}^*) \leq O\left(\frac{\ln k}{\sqrt{k}}\right)$$

Conclusion

- **Primal-Dual** algorithms for decentralized optimization
- **EXTRA**, **Gradient tracking** and **Push-(Sub)Gradient method**
- Many algorithms exist based on combination of similar ideas for decentralized optimization over **directed** and **dynamic** networks
  - **EXTRA** and **Push-Sum**[13]
  - **Push-Sum** and **Gradient Tracking** (**DIGing** based methods)[14]

Next: Non-convex decentralized optimization

[13] Zeng et al., Extrapush for convex smooth decentralized optimization over directed networks." arXiv preprint arXiv, 2015.

[14] Nedic et al., Achieving geometric convergence for distributed optimization over time-varying graphs." SIAM Journal on Optimization 2017.