A Distributed Proximal Method for Composite Convex Optimization

Garud Iyengar

Industrial Engineering and Operations Research
Columbia University

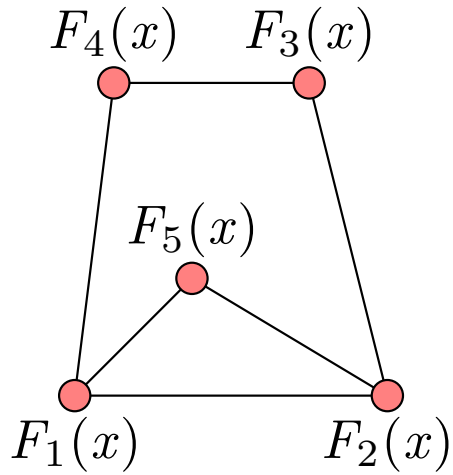
Joint work with Necdet Serhat Aybat and Zi Wang (Penn State)

NSF: DMS-1016571, CMMI-1235023, ONR: N000140310514

Distributed optimization

Compute the optimal solution for

$$F^* := \min_{x \in \mathcal{X}} \sum_{i=1}^{N} F_i(x).$$



Details

- $\mathcal{N} = \{1, \ldots, N\}$ nodes of a connected undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$
  - Node i and j can communicate only if $(i, j) \in \mathcal{E}$
- Composite function $F_i(x) = \rho_i(x) + \gamma_i(x)$ locally known
  - γ_i: convex + Lipschitz continuous gradient
  - ρ_i: convex + possibly non-smooth but efficient prox-map
  - For $t > 0$: $\mathbf{prox}_{t\rho}(x) = \operatorname{argmin}_{y \in \mathbb{R}^n} \left\{ t\rho(y) + \frac{1}{2}\|y - x\|_2^2 \right\}$
- $\mathcal{X}$ conic convex set of the form $\{x : Ex - q \in \mathcal{K}\}$

Canonical example

LASSO with locally known data

$$\min_x \sum_{i=1}^{N} F_i(x) = \underbrace{\tfrac{1}{2}\|A_i x - b_i\|_2^2}_{\gamma_i(x)} + \underbrace{\lambda_i \|x\|_1}_{\rho_i(x)}$$

- γ_i has a Lipschitz continuous gradient: $A_i^\top (A_i x - b_i)$
- ρ_i has an efficient prox map: $\mathbf{prox}_{t\rho_i}(x) = \mathsf{sgn}(x) \cdot (|x| - \lambda_i t)^+$

- Consolidating data at a central node may be memory intensive
- Consolidating data not be allowed because of privacy concerns
- Distributed algorithms are of interest in their own right

Distributed formulation

Distributed formulation

$$\begin{array}{ll} \min_x & \sum_{i=1}^{N} F_i(x_i) \\ \text{s. t.} & x_i = x_j, \quad \forall\ (i,j) \in \mathcal{E} \\ & x_i \in \mathcal{X}, \quad i \in \mathcal{N} \end{array}$$

- Create a local copy of variables for each node
- Impose consistency constraints to force the variables to be the same

Models a variety of important applications,

- distributed linear regression (Mateos et al., 2010),
- distributed control (Necoara and Suykens, 2008),
- machine learning (McDonald et al., 2010),
- estimation using sensor networks (Lesser et al., 2003).

Proposed solution

A distributed first-order augmented Lagrangian (DFAL) algorithm

- uses only local computations
- circumvents privacy, communication and memory issues
- works in asynchronous setting
- can be easily extended to handle global side constraint $Ex - q \in \mathcal{K}$

Need some additional constraints on γ and ρ

- $\gamma \approx$ loss function
- $\rho \approx$ norm or combinations of norms

DFAL: Augmented Lagrangian formulation

General distributed formulation: $\mathbf{x} = (x_1, \ldots, x_N)^\top$

$$\begin{aligned} \text{min} \quad & \underbrace{\textstyle\sum_{i\in\mathcal{N}} \rho_i(x_i)}_{\bar{\rho}(\mathbf{x})} + \underbrace{\textstyle\sum_{i\in\mathcal{N}} \gamma_i(x_i)}_{\bar{\gamma}(\mathbf{x})} \\ \text{s.t.} \quad & A\mathbf{x} = b \qquad (A \in \mathbb{R}^{m\times nN},\ \mathbf{rank}(A) = m\ ) \end{aligned}$$

Augmented Lagrangian formulation: dual $\theta^{(k)}$ and penalty $1/\lambda^{(k)}$

$$\begin{aligned} P^{(k)}(\mathbf{x}) &= \lambda^{(k)} \Big(\bar{\rho}(\mathbf{x}) + \bar{\gamma}(\mathbf{x}) - (\theta^{(k)})^\top (A\mathbf{x} - b) \Big) + \tfrac{1}{2}\|A\mathbf{x} - b\|_2^2 \\ &= \lambda^{(k)} \bar{\rho}(\mathbf{x}) + \underbrace{\lambda^{(k)} \bar{\gamma}(\mathbf{x}) + \tfrac{1}{2}\|A\mathbf{x} - b - \lambda^{(k)}\theta^{(k)}\|_2^2}_{f^{(k)}(\mathbf{x})} \\ &= \lambda^{(k)} \bar{\rho}(\mathbf{x}) + f^{(k)}(\mathbf{x}) \end{aligned}$$

where $\nabla f^{(k)}$ is Lipschitz continuous with constant $\lambda^{(k)} \bar{L} + \sigma_{\max}^2(A)$

DFAL: Inexactly solve the augmented Lagrangian

Two stopping conditions:

$$(a) \quad P^{(k)}(\mathbf{x}^{(k)}) - P^{(k)}(\mathbf{x}_*^{(k)}) \leq \alpha^{(k)}, \quad \mathbf{x}_*^{(k)} \in \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^{nN}} P^{(k)}(\mathbf{x})$$
$$(b) \quad \exists\, g_i^{(k)} \in \partial_{x_i} P^{(k)}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^{(k)}} \text{ s.t. } \max_{i\in\mathcal{N}} \|g_i^{(k)}\|_2 \leq \frac{\xi^{(k)}}{\sqrt{N}} \qquad (1)$$

Algorithm DFAL $\left(\lambda^{(1)}, \alpha^{(1)}, \xi^{(1)}\right)$

Step 0: *Set* $\theta^{(1)} = \mathbf{0}, k = 1, \ c \in (0,1)$

Step k: $(k \geq 1)$

1. *Do prox-gradient steps until* $\mathbf{x}^{(k)}$ *satisfies* (1)(a) *or* (1)(b)
2. $\theta^{(k+1)} = \theta^{(k)} - \frac{A\mathbf{x}^{(k)} - b}{\lambda^{(k)}}$
3. $\lambda^{(k+1)} = c\lambda^{(k)}, \quad \alpha^{(k+1)} = c^2\, \alpha^{(k)}, \quad \xi^{(k+1)} = c^2\, \xi^{(k)}$

DFAL: Iteration complexity

Theorem: There exists explicitly defined $(\lambda^{(1)}, \alpha^{(1)}, \xi^{(1)}, c)$ such that for all $\epsilon > 0$, **DFAL** iterates $\mathbf{x}^{(k)}$ are

- ϵ-feasible, i.e. $\|A\mathbf{x}^{(k)} - b\|_2 \leq \epsilon$, and
- ϵ-optimal, i.e. $|\bar{F}(\mathbf{x}^{(k)}) - \bar{F}^*| \leq \epsilon$

for all $k \geq K(\epsilon) := \log_{\frac{1}{c}}(\frac{\bar{C}}{\epsilon})$ for some $\bar{C} > 0$.

Main ingredients of the proof

- Prove duals $\theta^{(k)}$ are bounded: Assumptions on γ and ρ
- Prove iterates $\mathbf{x}^{(k)}$ are bounded: Assumptions on γ and ρ
- Induction establishes the result

Assumptions on γ and ρ

Smooth function γ

- $\nabla\gamma$ is Lipschitz continuous with constant L_γ
- lower bounded: $\exists\, \underline{\gamma} \in \mathbb{R}$ s.t. $\gamma(x) \geq \underline{\gamma}$ for all $x \in \mathbb{R}^n$.

Examples: quadratic-loss $\|Ax - b\|_2^2$, Huber-loss $\sum_{i=1}^m h_\delta(a_i^\mathsf{T} x - b_i)$ logistic-loss $\sum_{i=1}^m \log\left(1 + e^{-b_i a_i^\mathsf{T} x}\right)$

Non-smooth function ρ

- bounded sub-gradients: $\exists B > 0$ s.t. $\|q\|_2 \leq B \quad \forall q \in \partial\rho(x), \forall x \in \mathbb{R}^n$
- lower bounded by a norm: $\exists \tau > 0$ s.t. $\tau\|x\|_2 \leq \rho(x) \quad \forall x \in \mathbb{R}^n$

Examples: $\|\cdot\|_\alpha$ with $\alpha \in \{1, 2, \infty\}$, group norm, nuclear norm, weighted sum of norms, e.g. sparse group norm

DFAL: Multi-Step APG oracle

Algorithm MS-APG ($\bar{\rho}, f, \mathbf{y}^{(0)}$)

Step 0: Take $\bar{\mathbf{y}}^{(1)} = \mathbf{y}^{(0)}, t^{(1)} = 1$

Step ℓ: $(\ell \geq 1)$

1. $y_i^{(\ell)} = \mathbf{prox}_{\rho_i / L_i} \left(\bar{y}_i^{(\ell)} - \nabla_{y_i} f(\bar{\mathbf{y}}^{(\ell)}) / L_i \right) \quad \forall i \in \mathcal{N}$
2. $t^{(\ell+1)} = (1 + \sqrt{1 + 4 \left(t^{(\ell)} \right)^2})/2$
3. $\bar{\mathbf{y}}^{(\ell+1)} = \mathbf{y}^{(\ell)} + \frac{t^{(\ell)} - 1}{t^{(\ell+1)}} \left(\mathbf{y}^{(\ell)} - \mathbf{y}^{(\ell-1)} \right)$

Remarks:

- L_i can computed locally
- **MS-APG** step length $1/L_i \geq 1/L$ is different for each $i \in \mathcal{N}$
- **MS-APG** $\approx$ APG when $\{L_i\}_{i \in \mathcal{N}}$ are close to each other
- if $\frac{\max_{i \in \mathcal{N}} L_i}{\min_{i \in \mathcal{N}} L_i} \gg 1$, APG can only take very tiny steps

DFAL: Multi-Step APG oracle

Algorithm MS-APG ($\bar{\rho}, f, \mathbf{y}^{(0)}$)

Step 0: Take $\bar{\mathbf{y}}^{(1)} = \mathbf{y}^{(0)}$, $t^{(1)} = 1$

Step ℓ: $(\ell \geq 1)$

1. $y_i^{(\ell)} = \mathbf{prox}_{\rho_i / L_i}\left(\bar{y}_i^{(\ell)} - \nabla_{y_i} f(\bar{\mathbf{y}}^{(\ell)})/L_i\right) \quad \forall i \in \mathcal{N}$
2. $t^{(\ell+1)} = (1 + \sqrt{1 + 4\left(t^{(\ell)}\right)^2})/2$
3. $\bar{\mathbf{y}}^{(\ell+1)} = \mathbf{y}^{(\ell)} + \frac{t^{(\ell)}-1}{t^{(\ell+1)}}\left(\mathbf{y}^{(\ell)} - \mathbf{y}^{(\ell-1)}\right)$

Theorem: **DFAL** can compute an ϵ-feasible, and ϵ-optimal solution within $\mathcal{O}(\frac{1}{\epsilon})$ MS-APG iterations, which requires $\mathcal{O}(\frac{1}{\epsilon})$ communication steps along the edges $\mathcal{E}$.

DFAL: Consensus problem

Write consensus constraints in matrix form

$$x_i - x_j = 0 \ \forall (i,j) \ \in \mathcal{E} \quad \Leftrightarrow \quad C\mathbf{x} = \mathbf{0}, \ C \in \mathbb{R}^{|E|n \times |\mathcal{N}|n}$$

Then

$$\Psi := C^{\mathsf{T}} C = \begin{bmatrix} \Omega_{11} I_n & \Omega_{12} I_n & \cdots & \Omega_{1N} I_n \\ \Omega_{21} I_n & \Omega_{22} I_n & \cdots & \Omega_{2N} I_n \\ \vdots & \vdots & \ddots & \vdots \\ \Omega_{N1} I_n & \Omega_{N2} I_n & \cdots & \Omega_{NN} I_n \end{bmatrix}$$

Here $\Omega \in \mathbb{R}^{N \times N}$ be the **Laplacian** of connected graph $\mathcal{G}$

- $\mathbf{rank}(\Omega) = N - 1$
- **spectral gap** $\psi_{N-1} > 0$, $\psi_N = 0$
- $\sigma^2_{\max}(C) = \psi_1$ and $\sigma^2_{\min}(C) = \psi_{N-1}$

DFAL: Consensus problem

Write consensus constraints in matrix form

$$x_i - x_j = 0 \ \forall (i,j) \ \in \mathcal{E} \quad \Leftrightarrow \quad C\mathbf{x} = \mathbf{0}, \ C \in \mathbb{R}^{|E|n \times |\mathcal{N}|n}$$

Then

$$\Psi := C^{\mathsf{T}} C = \begin{bmatrix} \Omega_{11} I_n & \Omega_{12} I_n & \cdots & \Omega_{1N} I_n \\ \Omega_{21} I_n & \Omega_{22} I_n & \cdots & \Omega_{2N} I_n \\ \vdots & \vdots & \ddots & \vdots \\ \Omega_{N1} I_n & \Omega_{N2} I_n & \cdots & \Omega_{NN} I_n \end{bmatrix}$$

Define $A := \Sigma V^{\mathsf{T}}$ where $C = U \Sigma V^{\mathsf{T}}$ is the reduced SVD of C.

- $A \in \mathbb{R}^{n(N-1) \times nN}$ has **linearly independent rows**.
- $A^{\mathsf{T}} A = C^{\mathsf{T}} C = \Psi$; $\sigma^2_{\max}(A) = \psi_1$, and $\sigma^2_{\min}(A) = \psi_{N-1}$.

DFAL: Consensus problem

Main ingredient in **MS-APG** is the gradient $\nabla_{x_i} f^{(k)}(\mathbf{x})$: Suppose $\theta^{(0)} = 0$

$$\nabla_{x_i} f^{(k)}(\mathbf{x}) = \lambda^{(k)} \nabla \gamma_i(x_i) + d_i \left(x_i + \bar{x}_i^{(k)} \right) - \sum_{j \in \mathcal{O}_i} \left(x_j + \bar{x}_j^{(k)} \right),$$

where $\bar{x}^{(k)} := \lambda^{(k)} \sum_{t=1}^{k-1} \frac{1}{\lambda^{(t)}} x^{(t)}$, $\mathcal{O}_i = \{ j \in \mathcal{N} : \ (i,j) \in \mathcal{E} \}$

Note

- Gradient can be computed without explicitly computing $\theta^{(k)}$.
- Do not need the SVD of C!
- Only need local graph structure

DFAL: Complexity result for consensus problem

Theorem: **DFAL** iterates are ϵ-optimal and ϵ-feasible within $K(\epsilon) = \mathcal{O}(\log(\frac{1}{\epsilon}))$ iterations, requiring at most $\mathcal{O}(\frac{\psi_1^{1.5}}{d_{\min}}\frac{1}{\epsilon})$ communication steps.

Remarks:

1. $d_{\min}$ denotes the degree of smallest degree node in $\mathcal{G}$.
2. Since $\psi_{N-1} \leq d_{\min}$, the main theorem implies the number of communication steps can be bounded above by $\mathcal{O}(\frac{\psi_1^{1.5}}{\psi_{N-1}}\epsilon^{-1})$.

DFAL: Global Constraints

Global constraints $Ex - q \in \mathcal{K}$ where projection $\Pi_{\mathcal{K}}$ is easy

$$\begin{array}{ll} \text{min} & \sum_{i \in \mathcal{N}} \rho(x) + \gamma(x) \\ \text{s.t.} & Ex - q \in \mathcal{K} \end{array} \quad \rightsquigarrow \quad \begin{array}{ll} \text{min} & \bar{\rho}(\mathbf{x}) + \bar{\gamma}(\mathbf{x}) \\ \text{s.t.} & C\mathbf{x} = 0 \\ & Ex_i - q = s_i, \quad s_i \in \mathcal{K} \end{array}$$

Augmented Lagrangian: duals $\theta^{(k)}$, $\{\mu_i^{(k)}\}_{i \in \mathcal{N}}$, and penalty $1/\lambda^{(k)}$

$$P^{(k)}(\mathbf{x}) = \min_{\mathbf{s} \in \mathcal{K}^N} \Big\{ \lambda^{(k)} \Big(\bar{\rho}(\mathbf{x}) + \bar{\gamma}(\mathbf{x}) - (\theta^{(k)})^\top (C\mathbf{x}) \Big) + \tfrac{1}{2} \| C\mathbf{x} \|_2^2 - \sum_{i \in \mathcal{N}} \Big(\lambda^{(k)} (\mu_i^{(k)})^\top (Ex_i - q - s_i) + \tfrac{1}{2} \| Ex_i - q - s_i \|_2^2 \Big) \Big\}$$

DFAL: Global Constraints

Global constraints $Ex - q \in \mathcal{K}$ where projection $\Pi_{\mathcal{K}}$ is easy

$$\begin{array}{ll} \text{min} & \sum_{i\in\mathcal{N}} \rho(x) + \gamma(x) \\ \text{s.t.} & Ex - q \in \mathcal{K} \end{array} \quad \rightsquigarrow \quad \begin{array}{ll} \text{min} & \bar{\rho}(\mathbf{x}) + \bar{\gamma}(\mathbf{x}) \\ \text{s.t.} & C\mathbf{x} = 0 \\ & Ex_i - q = s_i, \quad s_i \in \mathcal{K} \end{array}$$

Augmented Lagrangian: duals $\theta^{(k)}$, $\{\mu_i^{(k)}\}_{i\in\mathcal{N}}$, and penalty $1/\lambda^{(k)}$

$$\begin{aligned} P^{(k)}(\mathbf{x}) \;=\; & \lambda^{(k)}\big(\bar{\rho}(\mathbf{x}) + \bar{\gamma}(\mathbf{x})\big) + \tfrac{1}{2}\| C\mathbf{x} - \lambda^{(k)}\theta^{(k)}\|_2^2 \\ & + \tfrac{1}{2}\sum_{i\in\mathcal{N}} d_{\mathcal{K}}\Big(E\,x_i - q - \lambda^{(k)}\mu_i^{(k)}\Big)^2 \end{aligned}$$

DFAL: Global Constraints

Global constraints $Ex - q \in \mathcal{K}$ where projection $\Pi_{\mathcal{K}}$ is easy

$$\begin{array}{ll} \min & \sum_{i\in\mathcal{N}} \rho(x) + \gamma(x) \\ \text{s.t.} & Ex - q \in \mathcal{K} \end{array} \quad \rightsquigarrow \quad \begin{array}{ll} \min & \bar{\rho}(\mathbf{x}) + \bar{\gamma}(\mathbf{x}) \\ \text{s.t.} & C\mathbf{x} = 0 \\ & Ex_i - q = s_i, \quad s_i \in \mathcal{K} \end{array}$$

Augmented Lagrangian: duals $\theta^{(k)}$, $\{\mu_i^{(k)}\}_{i\in\mathcal{N}}$, and penalty $1/\lambda^{(k)}$

$$\begin{aligned} P^{(k)}(\mathbf{x}) \;=\; & \lambda^{(k)}\big(\bar{\rho}(\mathbf{x}) + \bar{\gamma}(\mathbf{x})\big) + \tfrac{1}{2}\| C\mathbf{x} - \lambda^{(k)}\theta^{(k)}\|_2^2 \\ & + \tfrac{1}{2}\sum_{i\in\mathcal{N}} d_{\mathcal{K}}\Big(E\,x_i - q - \lambda^{(k)}\mu_i^{(k)}\Big)^2 \end{aligned}$$

Let $p(x) = \frac{1}{2} d_{\mathcal{K}}(Ex - q)^2$

- $\nabla p(x) = E^T\left(Ex - q - \Pi_{\mathcal{K}}(Ex - q)\right) = -\Pi_{\mathcal{K}^*}(q - Ex)$
- ∇p is Lipschitz continuous with constant $\sigma_{\max}^2(E)$

DFAL: Global Constraints

Recall the pseudo-code for DFAL

Algorithm DFAL $\left(\lambda^{(1)}, \alpha^{(1)}, \xi^{(1)}\right)$

Step 0: *Set* $\theta^{(1)} = \mathbf{0},\ \mu_i^{(1)} = \mathbf{0},\ i \in \mathcal{N},\ k = 1$

Step k: $(k \geq 1)$

1. *Do prox-gradient steps until* $\mathbf{x}^{(k)}$ *satisfies* (1)(a) *or* (1)(b)
2. $\theta^{(k+1)} = \theta^{(k)} - \frac{C\mathbf{x}^{(k)}}{\lambda^{(k)}}$ (do not compute this explicitly)
3. $\mu_i^{(k+1)} = \frac{1}{\lambda^{(k)}} \Pi_{\mathcal{K}^*}\left(\lambda^{(k)}\mu_i^{(k)} + q - Ex_i^{(k)}\right) \in \mathcal{K}^*$ for all $i \in \mathcal{N}$
4. $\lambda^{(k+1)} = c\lambda^{(k)}, \quad \alpha^{(k+1)} = c^2\ \alpha^{(k)}, \quad \xi^{(k+1)} = c^2\ \xi^{(k)}$

Two stopping conditions

- (1)(a): $\alpha^{(k)}$-optimality
- (1)(b): $\xi^{(k)}$-stationarity

ADMM by Makhdoumi and Ozdaglar (2014)

Reformulate problem with node and edge variables

$$\begin{aligned} \min_{x,z} \quad & \sum_{i=1}^{N} \underbrace{\rho_i(x_i) + \gamma_i(x_i)}_{F_i(x_i)} \\ \text{s.t.} \quad & \Omega_{ij} x_j = z_{ij}, \quad i \in \mathcal{N},\ j \in \mathcal{N}_i := \mathcal{O}_i \cup \{i\}. \\ & \textstyle\sum_{j \in \mathcal{N}_i} z_{ij} = 0, \quad i \in \mathcal{N} \end{aligned}$$

Augmented Lagrangian with fixed penalty

$$\mathcal{L}(x, z, \theta) = \sum_{i=1}^{N} F_i(x_i) + \sum_{j \in \mathcal{N}_i} p^\top (x_i - z_{ij}) + \frac{\beta}{2} \|x_i - z_{ij}\|_2^2$$

ADMM algorithm

Update x: $x_i^{(k+1)} \leftarrow \operatorname{argmin}_x \mathcal{L}(x, z^{(k)}, p^{(k)})$

Update z: $z^{(k+1)} \leftarrow \operatorname{argmin}_{z: \sum_{j \in \mathcal{N}_i} z_{ij} = 0} \mathcal{L}(x^{(k+1}, z, p^{(k)})$

Update p: $p_{ij}^{(k+1)} = p_{ij}^{(k)} + \beta(x_i^{(k+1)} - z_{ij}^{(k+1)})$

ADMM by Makhdoumi and Ozdaglar (2014)

ADMM algorithm can be simplified as follows.

Algorithm ADMM ($c, \mathbf{x}^{(0)}$)

Initialization: $\mathbf{y}^{(0)} = \mathbf{x}^{(0)}$, $p_i^{(k)} = \tilde{p}_i^{(k)} = \mathbf{0}, \quad i \in \mathcal{N}$

Step k: $(k \geq 0)$ For $i \in \mathcal{N}$ compute

1. $x_i^{(k+1)} = \mathbf{prox}_{\frac{F_i}{\beta(d_i+d_i^2)}} \left(x_i^{(k)} - \frac{\sum_{j\in\mathcal{N}_i} \Omega_{ji}(s_j^{(k)}+p_j^{(k)})}{d_i^2+d_i} \right)$
2. $s_i^{(k+1)} = \frac{1}{d_i+1} \sum_{j\in\mathcal{N}_i} \Omega_{ij} x_j^{(k+1)}$
3. $p_i^{(k+1)} = p_i^{(k)} + s_i^{(k+1)}$

- $\mathcal{O}(\epsilon^{-1})$ communication steps and $\mathcal{O}(\epsilon^{-1})$ $\mathbf{prox}_{F_i}$-computations
- Computing $\mathbf{prox}_{F_i}$ is almost as hard as solving the problem.

DFAL vs ADMM

Motivations for DFAL

- Approximately solve the augmented Lagrangian via prox-gradient steps
- Increase the accuracy as the computation progresses
- Increase the penalty to assist in convergence to feasibility

Convergence results

- Does not need an order over nodes
- Can convert into an asynchronous algorithm
- Side constraints are not a problem

DFAL: Problems with synchronous algorithm

DFAL is a synchronous algorithm

- All nodes have to complete computation before an update
- Rate is dominated by the slowest node
- Serious issue if computational resources at nodes are very variable

Asynchronous **DFAL**

- Computation terminates at node i.
- Node i signals to all its neighbors to terminate, and
- Shares its local variable with them.

Model for analysis

- Node that terminates first is a sample of the random variable $\mathcal{R}$
- Assuming memoryless property
- Can also be viewed as a sampled distributed optimization

ARBCD: Modification of MS-APG oracle

Algorithm ARBCD $\left(\mathbf{z}^{(0)}\right)$

1: $\ell \leftarrow 0, \quad t^{(0)} \leftarrow 1, \quad u_i^{(1)} \leftarrow \mathbf{0}, \quad \forall i \in \mathcal{N}$

2: **while** $\ell \geq 0$ **do**

3: i is a sample of $\mathcal{R}$

4: $z_i^{(\ell+1)} \leftarrow \mathbf{prox}_{t^{(\ell)}\rho_i/L_i}\left(z_i^{(\ell)} - \frac{t^{(\ell)}}{L_i}\nabla_{y_i} f\left(\left(\frac{1}{Nt^{(\ell)}}\right)^2 \mathbf{u}^{(\ell)} + \mathbf{z}^{(\ell)}\right)\right)$

5: $u_i^{(\ell+1)} \leftarrow u_i^{(\ell)} + N^2 t^{(\ell)}(1 - t^{(\ell)})\left(z_i^{(\ell+1)} - z_i^{(\ell)}\right)$

6: $z_{-i}^{(\ell+1)} \leftarrow z_{-i}^{(\ell)}, \quad u_{-i}^{(\ell+1)} \leftarrow u_{-i}^{(\ell)}$

7: $t^{(\ell+1)} \leftarrow \frac{1+\sqrt{1+(2Nt^{(\ell)})^2}}{2N}$

8: **end while**

Remarks:

- A variant of the algorithm in Fercoq and Richtárik (2013)
- In Asynchronous **DFAL** the realization in Step 3 occurs naturally.

Asynchronous DFAL: Iteration complexity

Lemma: Fix $\alpha > 0$, and $p \in (0,1)$. Let $\{\mathbf{z}_k^{(\ell)}, \mathbf{u}_k^{(\ell)}\}_{\ell \in \mathbb{Z}_+}$ $1 \leq k \leq M$ denote the iterates corresponding to $M = \log(1/p)$ independent calls to **ARBCD**$(\mathbf{y}^{(0)})$. Let $T := 2N\sqrt{\frac{2C}{\alpha}}$ and $\mathbf{y}_k := \left(\frac{1}{Nt^{(T)}}\right)^2 \mathbf{u}_k^{(T+1)} + \mathbf{z}_k^{(T+1)}$. Then

$$\mathbb{P}\left(\min_{k=1,\ldots,M} \Phi(\mathbf{y}_k) - \Phi^* \leq \alpha\right) \geq 1 - p$$

Result follows from Fercoq and Richtárik (2013)

Asynchronous DFAL: Iteration complexity

Lemma: Fix $\alpha > 0$, and $p \in (0,1)$. Let $\{\mathbf{z}_k^{(\ell)}, \mathbf{u}_k^{(\ell)}\}_{\ell \in \mathbb{Z}_+}$ $1 \leq k \leq M$ denote the iterates corresponding to $M = \log(1/p)$ independent calls to **ARBCD**$(\mathbf{y}^{(0)})$. Let $T := 2N\sqrt{\frac{2C}{\alpha}}$ and $\mathbf{y}_k := \left(\frac{1}{Nt^{(T)}}\right)^2 \mathbf{u}_k^{(T+1)} + \mathbf{z}_k^{(T+1)}$. Then

$$\mathbb{P}\left(\min_{k=1,\ldots,M} \Phi(\mathbf{y}_k) - \Phi^* \leq \alpha\right) \geq 1 - p$$

Result follows from Fercoq and Richtárik (2013)

For $K(\epsilon) = \mathcal{O}(\log(1/\epsilon))$, terminate **ARBCD** in k^{th} **DFAL** iteration when

$$\mathbb{P}\left(P^{(k)}\left(\mathbf{x}^{(k)}\right) - P^{(k)}\left(\mathbf{x}_*^{(k)}\right) \leq \alpha^{(k)}\right) \geq (1-p)^{\frac{1}{K(\epsilon)}},$$

Then $x^{(K(\epsilon))}$ can be computed in $\mathcal{O}\left(\frac{1}{\epsilon}\log\left(\frac{1}{p}\right)\right)$ **ARBCD** iterations and

$$\mathbb{P}\Big(\Big|\sum_{i\in\mathcal{N}} F_i\left(x_i^{(K(\epsilon))}\right) - F^*\Big| \leq \epsilon, \quad \max_{(i,j)\in\mathcal{E}}\left\{\|x_i^{(K(\epsilon))} - x_j^{(K(\epsilon))}\|_2\right\} \leq \epsilon\Big) \geq 1 - p$$

Numerical study: Sparse group LASSO

Loss function: Huber loss $h_\delta(y) = \max\left\{u^\top y - \frac{1}{2}\|u\|_2^2 : u_i \in [-\delta, \delta]\right\}$

$$\gamma(x) = \sum_{i=1}^{N} h_\delta(A_i x - b_i)$$

Non-smooth regularizer: $G_i = \{g_{ik} : g_{ik} \subseteq \{1, \ldots, n\}, 1 \le k \le m_i\}$ partition of $\{1, \ldots, n\}$

$$\rho(x) = \beta_1 \sum_{i=1}^{N} \underbrace{\|x\|_1}_{\text{sparse } x} + \beta_2 \sum_{i=1}^{n} \underbrace{\sum_{k=1}^{m_i} \|x_{g_{ik}}\|_2}_{\|x\|_{G_i} \text{ sparse groups}}$$

Optimization problem

$$\min_{x \in \mathbb{R}^n} \left\{ \rho(x) + \sum_{i=1}^{N} h_\delta(A_i x - b_i) \right\}$$

Problem classes

Graph $\mathcal{G}$: either a star or a clique on $N = 5$ or 10 nodes

Partition: $K = 10$ and $n_g \in \{100, 300\}$, i.e. $n = Kn_g \in \{1000, 3000\}$

- Case I: randomly generate 1 partition G. Set $G_i \equiv G$.
- Case II: randomly generate a partition for each node

Data matrices: For each node $m = \frac{n}{4N}$,

- $A_{ij} \sim \mathcal{N}(0, 1)$ for $1 \leq i \leq m$ and $1 \leq j \leq n$
- $b_i = A_i \bar{x}$ where $\bar{x}_j = (-1)^j \ e^{-(j-1)/n_g}$

Machine with 4 (2.75GHz) cores and 64 GB of RAM.

Distributed ADMM Algorithm

ADMM: Makhdoumi and Ozdaglar (2014) implemented ADMM on

$$\begin{aligned} \min_{x,z} \quad & \sum_{i=1}^{N} \underbrace{\rho_i(x_i) + \gamma_i(x_i)}_{F_i(x_i)} \\ \text{s.t.} \quad & \Omega_{ij} x_j = z_{ij}, \quad i \in \mathcal{N},\ j \in \mathcal{N}_i := \mathcal{O}_i \cup \{i\}. \end{aligned}$$

- $\mathcal{O}(\epsilon^{-1})$ communication steps and $\mathcal{O}(\epsilon^{-1})$ $\mathbf{prox}_{F_i}$-computations
- Computing $\mathbf{prox}_{F_i}$ is almost as hard as solving the problem.

SADMM: To overcome the $\mathbf{prox}_{F_i}$ bottleneck, we propose "splitting" x_i

$$\begin{aligned} \min_{x,z,\tilde{z}} \quad & \sum_{i \in \mathcal{N}} \rho_i(x_i) + \gamma_i(y_i) \\ \text{s.t.} \quad & \Omega_{ij} x_j = z_{ij}, \quad i \in \mathcal{N},\ j \in \mathcal{N}_i \\ & \Omega_{ij} y_j = \tilde{z}_{ij}, \quad i \in \mathcal{N},\ j \in \mathcal{N}_i \\ & x_i = q_i,\ y_i = q_i, \quad i \in \mathcal{N} \end{aligned}$$

Group sparse LASSO: Results for $n_g = 100$

Alg.	Rel. Suboptimality		Consensus Violation (CV)		CPU Time (sec.)		Iterations	
	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2
SDPT3 (C)	0	0	0	0	28	85	24	22
APG (C)	1E-3	N/A	0	N/A	10	N/A	2173	N/A
DFAL (D)	6E-4, 7E-4	4E-4, 6E-4	5E-5, 2E-5	5E-5, 3E-5	**5**, **5**	**5**, **5**	1103, 1022	1105, 1108
AFAL (D)	3E-4, 8E-4	3E-4, 6E-4	3E-6, 5E-6	2E-6, 5E-6	**16**, **11**	**17**, **11**	9232, 9083	9676, 9844
ADMM (D)	6E-5, 5E-5	7E-5, 7E-5	1E-4, 1E-4	1E-4, 1E-4	1125, 808	1090, 771	353, 253	363, 261
SADMM (D)	1E-4, 3E-4	1E-4, 3E-4	1E-4, 1E-4	1E-4, 1E-4	771, 784	772, 804	592, 606	593, 623
			$N = 5,\ n = 1000$					
SDPT3 (C)	0	0	0	0	28	89	24	22
APG (C)	1E-3	N/A	0	N/A	10	N/A	2173	N/A
DFAL (D)	4E-4, 7E-4	4E-4, 3E-4	6E-5, 1E-5	6E-5, 2E-5	**14**, **12**	**14**, **13**	1794, 1439	1812, 1560
AFAL (D)	4E-4, 8E-4	2E-4, 8E-4	7E-6, 2E-6	8E-6, 2E-6	**48**, **65**	**49**, **62**	20711, 41125	21519, 41494
ADMM (D)	9E-3, 2E-2	8E-3, 2E-2	9E-4, 5E-4	8E-4, 4E-4	T, T	T, T	354, 353	373, 372
SADMM (D)	3E-4, 9E-3	4E-4, 1E-2	2E-4, 1E-3	2E-4, 1E-3	T, T	T, T	867, 883	865, 879
			$N = 10,\ n = 1000$					

Termination time T = 3 hours = 1800 seconds

Group sparse LASSO: Results for $n_g = 300$

Alg.	Rel. Suboptimality		Consensus Violation (CV)		CPU Time (sec.)		Iterations	
	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2	Case 1	Case 2
SDPT3 (C)	0	0	0	0	806	1653	26	29
APG (C)	1E-3	N/A	0	N/A	253	N/A	8663	N/A
DFAL (D)	2E-4, 5E-4	2E-4, 4E-4	6E-5, 4E-5	5E-5, 5E-5	**77**, **64**	**80**, **65**	1818, 1511	1897, 1535
AFAL (D)	1E-4, 6E-4	2E-5, 6E-4	5E-7, 2E-5	6E-8, 2E-5	**164**, **99**	**273**, **99**	21747, 8760	37212, 8736
ADMM (D)	5E-2, 1E-3	5E-2, 1E-3	5E-3, 1E-3	5E-3, 1E-3	T, T	T, T	109, 118	109, 118
SADMM (D)	2E-2, 7E-2	2E-2, 8E-2	2E-3, 3E-3	2E-3, 3E-3	T, T	T, T	269, 274	268, 273
			$N = 5,\ n = 3000$					
SDPT3 (C)	0	0	0	0	806	1641	26	29
APG (C)	1E-3	N/A	0	N/A	253	N/A	8663	N/A
DFAL (D)	1E-4, 6E-4	6E-4, 1E-3	7E-5, 4E-5	9E-5, 5E-5	**130**, **80**	**122**, **82**	2942, 1721	2794, 1769
AFAL (D)	2E-4, 7E-4	6E-4, 1E-3	8E-7, 1E-5	3E-7, 1E-5	**350**, **294**	**437**, **288**	48214, 29946	63110, 30371
ADMM (D)	5E-2, 8E-2	5E-2, 8E-2	7E-3, 9E-3	7E-3, 9E-3	T, T	T, T	114, 124	113, 123
SADMM (D)	3E-1, 3E+0	3E-1, 3E+0	4E-3, 2E-2	4E-3, 2E-2	T, T	T, T	255, 269	256, 268
			$N = 10,\ n = 3000$					

Termination time T = 3 hours = 1800 seconds

Optimal AC power flow

Power network $\mathcal{G} = (\mathcal{E}, \mathcal{N})$

- buses (nodes) $\mathcal{N} = \{0, \cdots, N\}$: $V_k \in \mathbb{C}$ complex voltage at bus k
- lines (edges) $\mathcal{E}$: $y_{kl} = g_{kl} + \mathbf{i} b_{kl}$ admittance of line (k, l)
- power injected in bus k: s_k

Optimal power flow (OPF) problem (nonconvex)

$$\begin{array}{ll} \min\limits_{s, V, s_0} & \sum\limits_{(k,l)\in\mathcal{E}} g_{kl} \, |V_k - V_l|^2 \\ \text{s.t.} & s_k = \sum\limits_{l\in\mathcal{N}(k)} V_k (V_k^* - V_l^*) y_{kl}^* \quad k \in \mathcal{N} \\ & \underline{s}_k \preceq s_k \preceq \overline{s}_k, \quad k \in \mathcal{N} \setminus \{0\} \\ & \underline{V}_k \leq |V_k| \leq \overline{V}_k, \quad i \in \mathcal{N} \setminus \{0\} \end{array}$$

where $\mathcal{N}(k) := \{l : (k, l) \in \mathcal{E}\}$.

Distributed formulation for semidefinite relaxation

The distributed OPF problem

$$
\begin{aligned}
\min_{s,\,W,\,s_0} \quad & \tfrac{1}{2}\textstyle\sum_{k\in\mathcal{N}}\sum_{l\in\mathcal{N}(k)} g_{kl}\big(w_{kk}^{(k)} - w_{kl}^{(k)} - w_{lk}^{(k)} + w_{kk}^{(k)}\big) \\
\text{s.t.} \quad & AW = 0 \qquad\qquad \text{(consensus constraints)} \\
& \left.\begin{array}{l} s_k = \sum_{l\in\mathcal{N}(k)} y_{ij}^{*}\big(w_{kk}^{(k)} - w_{kl}^{(k)}\big) \\ \underline{s}_k \preceq s_k \preceq \overline{s}_k \\ \underline{V}_k^2 \le w_{kk}^{(k)} \le \overline{V}_k^2 \\ W_{kl}^{(k)} \succeq \mathbf{0}, (k,l) \in \mathcal{E} \end{array}\right\} = \chi_k, \quad k \in \mathcal{N}
\end{aligned}
$$

where $W = \{w^{(k)}\}_{k=1}^{N}$, $w^{(k)} = \{w_{kl}^{(k)}\}_{l\in\mathcal{N}(i)\cup\{i\}}$.

Each node holds a $(2d_i + 1)$-dimensional variable.

Projection onto χ_k is efficient:

- Each projection is small
- $W_{kl}^{(k)} \succeq 0$ is an SOCP since $W_{kl}^{(k)} \in \mathbb{C}^{2\times 2}$.

OPF numerical results

Test Problem	Algorithm	f^*	Consensus Viol.	Wallclock Time (s)	Iterations
B. Lesieutre 3 Bus	Mosek(C)	1.27E-05	0	3	13
	DFAL(D)	1.28E-05	7.11E-06	2	10
IEEE 14 Bus	Mosek(C)	3.10E-03	0	3	14
	DFAL(D)	3.13E-03	8.05E-06	4	20
IEEE 30 Bus	Mosek(C)	7.08E-04	0	4	17
	DFAL(D)	7.02E-04	2.24E-06	5	25
IEEE 57 Bus	Mosek(C)	1.11E-03	0	5	19
	DFAL(D)	1.09E-03	1.59E-06	8	40

Summary

- Propose a distributed first-order Augmented Lagrangian algorithm
- Guarantees a bound on the consensus error
- Can handle global constraints

References I

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2(1):183–202.

Chen, A. I. and Ozdaglar, A. (2012). A fast distributed proximal-gradient method. In *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pages 601–608. IEEE.

Duchi, J. C., Agarwal, A., and Wainwright, M. J. (2012). Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Trans. Automat. Contr.*, 57(3):592–606.

Fercoq, O. and Richtárik, P. (2013). Accelerated, parallel and proximal coordinate descent. *arXiv preprint arXiv:1312.5799*.

Jakovetic, D., Xavier, J., and Moura, J. (2011). Fast distributed gradient methods.

Lesser, V., Ortiz Jr, C. L., and Tambe, M. (2003). *Distributed sensor networks: A multiagent perspective*, volume 9. Springer.

Makhdoumi, A. and Ozdaglar, A. (2014). Broadcast-based distributed alternating direction method of multipliers. In *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*, pages 270–277. IEEE.

Mateos, G., Bazerque, J. A., and Giannakis, G. B. (2010). Distributed sparse linear regression. *Signal Processing, IEEE Transactions on*, 58(10):5262–5276.

References II

McDonald, R., Hall, K., and Mann, G. (2010). Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464. Association for Computational Linguistics.

Necoara, I. and Suykens, J. A. K. (2008). Application of a smoothing technique to decomposition in convex optimization. *Automatic Control, IEEE Transactions on*, 53(11):2674–2679.

Nedic, A. and Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *Automatic Control, IEEE Transactions on*, 54(1):48–61.

Wei, E. and Ozdaglar, A. (2012). Distributed alternating direction method of multipliers.

Wei, E. and Ozdaglar, A. (2013). On the o (1/k) convergence of asynchronous distributed alternating direction method of multipliers. *arXiv preprint arXiv:1307.8254*.