Distributed Optimization in Multi-agent Systems

Asu Ozdaglar

Laboratory for Information and Decision Systems
Operations Research Center
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology

SIAM Conference on Control and Its Applications
July, 2013
Motivation

- Many networks are large-scale and comprise of agents with local information and heterogeneous preferences.
- This motivated much interest in developing distributed schemes for control and optimization of multi-agent networked systems.
Distributed Multi-agent Optimization

Many of these problems can be represented within the general formulation:
A set of agents (nodes) \( \{1, \ldots, N\} \) connected through a network.

The goal is to cooperatively solve

$$\min_x \sum_{i=1}^{N} f_i(x)$$

s.t. \( x \in \mathbb{R}^n \),

\( f_i(x) : \mathbb{R}^n \rightarrow \mathbb{R} \) is a convex (possibly nonsmooth) function, known only to agent \( i \).

Since such systems often lack a centralized processing unit, algorithms for this problem should involve each agent performing computations locally and communicating this information according to the underlying network.
Machine Learning Example

- A network of 3 sensors.
- Data is collected at different sensors: temperature $t$, electricity demand $d$.

System goal: learn a degree 3 polynomial electricity demand model:

$$d(t) = x_3 t^3 + x_2 t^2 + x_1 t + x_0.$$  

System objective:

$$\min_x \sum_{i=1}^{3} \| A_i' x - d_i \|_2^2.$$  

where $A_i = [1, t_i, t_i^2, t_i^3]'$ at input data $t_i$. 

![Least square fit with polynomial max degree 3](image)
Machine Learning General Set-up

- A network of agents $i = 1, \ldots, N$.
- Each agent $i$ has access to local feature vectors $A_i$ and output $b_i$.
- System objective: train weight vector $x$ to

$$\min_x \sum_{i=1}^{N-1} L(A_i'x - b_i) + p(x),$$

for some loss function $L$ (on the prediction error) and penalty function $p$ (on the complexity of the model).

- **Example:** Least-Absolute Shrinkage and Selection Operator (LASSO):

$$\min_x \sum_{i=1}^{N-1} \|A_i'x - b_i\|_2^2 + \lambda \|x\|_1.$$ 

- Other examples from ML estimation, low rank matrix completion, image recovery [Schizas, Ribeiro, Giannakis 08], [Recht, Fazel, Parrilo 10], [Steidl, Teuber, 10].
Literature: Parallel and Distributed Optimization

- Lagrangian relaxation and dual optimization methods:
  - Dual gradient ascent, (single) coordinate ascent methods.
- Parallel computation and optimization:
  - [Tsitsiklis 84], [Bertsekas and Tsitsiklis 95].
- Consensus and cooperative control:
  - Averaging algorithms: Deterministic averaging of all neighbor estimates.
    [Jadbabaie, Lin, and Morse 03], [Olfati-Saber and Murray 04],
    [Olshevsky and Tsitsiklis 07], [Tahbaz-Salehi and Jadbabaie 08], [Kar
    and Moura 09], [Frasca, Carli, Fagnani and Zampieri 09], [Bullo,
    Cortes, Martinez 09], [Oreshkin, Coates, and Rabbat 10].
  - Gossip algorithms: Random pairwise averaging.
    [Boyd, Ghosh, Prabhakar, and Shah 05], [Dimakis, Sarwate, and
    Wainwright 08], [Fagnani, Zampieri 09], [Aysal, Yildiz, Sarwate, and
    Scaglione 09].
Literature: Distributed Multi-agent Optimization

- Distributed first order primal subgradient methods [Nedic, Ozdaglar 07].

- Various extensions:
  - Local and global constraints [Nedic, Ozdaglar, Parrilo 08], [Zhu and Martinez 10].
  - Randomly varying communication networks [Lobel, Ozdaglar 09], [Baras and Matei 10], [Lobel, Ozdaglar, and Feijer 10].
  - Network effects [Nedic, Olshevsky, Ozdaglar, Tsitsiklis 09]
  - Random gradient errors [Ram, Nedic, Veeravalli 09].

- Ordinary-Augmented Lagrangian primal-dual subgradient methods
  - [Jakovetic, Xavier, Moura 11], [Zhu, Giannakakis, Cano 09], [Mota, Xavier, Aguiar, Puschel 13]

- Distributed second order methods (for more specialized problems)
  - [Wei, Ozdaglar, Jadambaie 11], [Liu, Sherali 12]
This Talk

- Fundamental ideas and recent advances in designing distributed algorithms for multi-agent optimization problems.
- Outline:
  - Brief overview of distributed primal subgradient methods [Nedic, Ozdaglar 07].
  - More recent progress on faster methods:
    - Synchronous Alternating Direction Method of Multipliers for distributed optimization [Wei and Ozdaglar 12].
    - Asynchronous Alternating Direction Method of Multipliers for distributed optimization [Wei and Ozdaglar 13].
Linear Dynamics and Transition Matrices

- We let $A(k)$ denote the weight matrix $[a_{ij}(k)]_{i,j \in \mathcal{M}}$, and define transition matrices
  \[ \Phi(k, s) = A(k)A(k-1) \cdots A(s+1)A(s) \quad \text{for all } k \geq s \]

- We use these matrices to relate $x_i(k+1)$ to $x_j(s)$ at time $s \leq k$:
  \[
  x_i(k+1) = \sum_{j=1}^{m}[\Phi(k, s)]_{ij}x_j(s) - \sum_{r=s}^{k-1} \sum_{j=1}^{m}[\Phi(k, r+1)]_{ij}\alpha(r)d_j(r) - \alpha(k)d_i(k).
  \]

- We analyze convergence properties of the distributed method by establishing:
  - Convergence of transition matrices $\Phi(k, s)$ (consensus part)
  - Convergence of an approximate subgradient method (effect of optimization)
Assumptions: Weights and Connectivity

Assumption (Weights)

(a) There exists a scalar $\eta \in (0, 1)$ s.t. $a_{ii}(k) \geq \eta$ and if $a_{ij}(k) > 0$, $a_{ij}(k) \geq \eta$.

(b) The weight matrix $A(k)$ is doubly stochastic, $\sum_{j=1}^{m} a_{ij}(k) = 1$ for all $i$ and $\sum_{i=1}^{m} a_{ij}(k) = 1$ for all $j$.

- Double stochasticity ensures agent estimates equally influential in the limit. This guarantees minimizing the sum of the local objective functions.
- Represent information exchange by $(V, E_k)$,
  \[ E_k = \{(j, i) \mid a_{ij}(k) > 0, \ i, j = 1, \ldots, m\} \]

Assumption (Connectivity)

There exists an integer $B \geq 1$ such that the directed graph
\[ (M, E_k \cup \cdots \cup E_{k+B-1}) \]
is strongly connected for all $k \geq 0$. 
Convergence Analysis – Idea

- But $y(k)$ evolution can be written as:

$$y(k+1) = \frac{1}{N} \sum_{j=1}^{N} x_{j}(s) - \frac{\alpha}{N} \sum_{r=s}^{k-1} \sum_{j=1}^{N} d_{j}(r) - \alpha d_{i}(k).$$

- Using the below result, this shows that $y(k)$ and $x_{i}(k)$ get close to each other in the limit: agent “disagreements” disappear and the method behaves as a centralized subgradient method.

Theorem (Nedic, Olshevsky, Ozdaglar, Tsitsiklis 09)

For all $i, j$ and all $k, s$ with $k \geq s$, we have

$$\left| \left[ \Phi(k, s) \right]_{ij} - \frac{1}{N} \right| \leq \left( 1 - \frac{\eta}{4N^2} \right)^{\left[ \frac{k-s+1}{B} \right]^2}.$$
Faster ADMM-based Distributed Algorithms

Motivated by the computational performance and inherent parallel implementation of the classical Augmented Lagrangian/Method of Multipliers methods [Glowinski, Marrocco 75], [Eckstein, Bertsekas 92], [Boyd et al. 10]:

- We develop an Alternating Direction Method of Multipliers (ADMM)-type distributed optimization algorithm.

Several papers have already demonstrated computationally the remarkable potential of ADMM for handling distributed optimization problems for decentralized estimation and compressive sensing applications. [Schizas, Ribeiro, Giannakis 08], [Mota, Xavier, Aguiar, Puschel 11].

In the rest of the talk, we present synchronous and asynchronous ADMM-type algorithms and show that they converge at the faster rate of $O(1/k)$ [Wei, Ozdaglar 12 (CDC)], [Wei, Ozdaglar 13].
Standard ADMM

- Standard ADMM solves a separable problem, where decision variable decomposes into two (linearly coupled) variables:

$$
\min_{x,y} \ f(x) + g(y) \\
\text{s.t.} \ Ax + By = c.
$$

- Commonly referred to as two-splitting in the literature.
- Consider an Augmented Lagrangian function:

$$
L_\beta(x, y, p) = f(x) + g(y) - p'(Ax + By - c) + \frac{\beta}{2} \|Ax + By - c\|^2.
$$

- ADMM: approximate version of classical Augmented Lagrangian method.
  - Primal variables: approximately minimize augmented Lagrangian through a single-pass coordinate descent (in a Gauss-Seidel manner).
  - Dual variable: updated through gradient ascent.
ADMM for Multi-agent Optimization Problem

- Multi-agent optimization can be reformulated in the ADMM framework:
- Consider a set of agents $V = \{1, \ldots, N\}$ connected through an undirected connected graph $G = \{V, E\}$.
- We introduce a local copy $x_i$ for each of the agents and impose $x_i = x_j$ for all $(i, j) \in E$.

$$\min_{x} \sum_{i=1}^{N} f_i(x_i)$$

s.t. $x_i = x_j$, for $(i, j) \in E$,

- This can be viewed as a multi-splitting version of the standard ADMM.
General Asynchronous ADMM: Problem Set-up

- Almost all distributed algorithms in the literature are synchronous.\(^1\)
- Highly decentralized nature of the problem calls for an asynchronous algorithm.
- We consider a more general formulation:

\[
\min_{x_i \in X_i, z \in Z} \sum_{i=1}^{N} f_i(x_i)
\]

\[s.t. \quad Dx + Hz = 0,\]

where \(f_i : \mathbb{R}^n \to \mathbb{R}\) is a convex function, \(X_i\) and \(Z\) are closed convex subsets of \(\mathbb{R}^n\) and \(\mathbb{R}^W\), and \(x = [x'_1, \ldots, x'_N]'\) is a partition of the decision vector.

- Multi-agent optimization is a special case of this formulation.

\(^1\)Exceptions: [Ram, Nedic, Veeravalli 09], [Lutzeler, Bianchi, Ciblat, and Hachem 13] without any rate results.
Asynchronous Implementation

- We impose an asynchronous implementation on our algorithm:
  - At each iteration, a subset of the constraints $\psi^k$ are randomly selected (active constraints), which in turn selects the corresponding component variables $x_i$, denoted by $\phi^k$ (active components or agents).
  - **Multi-agent optimization**: this corresponds to picking edges randomly and activating the agents incident to those edges.
- At each iteration only active components of the decision vector and active dual variables (for active constraints) are updated.
- We define:
  $$f^k(x) = \sum_{i \in \phi^k} f_i(x_i),$$
  $$D_{\phi^k} = \sum_{i \in \phi^k} D_i, \quad H_{\psi^k} = \sum_{l \in \psi^k} H_l,$$
  where $D_i$ picks up the columns corresponding to $x_i$ and $H_l$ picks up the diagonal element corresponding to constraint $l$ (has zeroes elsewhere).
- Sets $\bar{\phi}^k$ and $\bar{\psi}^k$ are complements of $\phi^k$, $\psi^k$ respectively.
Special Case: Multi-agent Asynchronous ADMM - Problem Formulation

\[
\min_x \sum_{i=1}^{N} f_i(x_i)
\]

s.t. \( x_i = x_j \), for \((i, j) \in E\).

- We can reformulate the problem to decouple \( x_i \) and \( x_j \) in each constraint by introducing the \( z \) variable [Bertsekas, Tsitsiklis 89].

- Along every edge \( e = (i, j) \) (constraint), for each node \( i \), introduce an auxiliary variable \( z_{ei} \), which allows us to simultaneously update \( x \) variables and potentially improve performance.

- Each constraint \( x_i - x_j = 0 \) for edge \( e = (i, j) \) becomes

\[
x_i = z_{ei}, \quad -x_j = z_{ej},
\]

\[
z_{ei} + z_{ej} = 0.
\]
Special Case: Multi-agent Asynchronous ADMM - Algorithm

$$\min_{x,z} \sum_{i=1}^{N} f_i(x_i)$$

s.t. \( x_i = z_{ei}, -x_j = z_{ej} \) for \((i,j) \in E\),

\( x \in X, \quad i = 1, \ldots, N, \quad z \in Z. \)

- Set \( Z = \{ z \mid z_{ei} + z_{ej} = 0 \text{ for all } e = (i,j) \} \).
- We associate an independent Poisson local clock with each edge.
- At iteration \( k \), if the clock corresponding to edge \((i,j)\) ticks:
  - \( \psi^k \) picks the constraints \( x_i = z_{ei}, -x_j = z_{ej} \) (subject to \( z_{ei} + z_{ej} = 0 \)).
  - \( \phi^k = \{i,j\} \),
    \[
    f^k(x) = f_i(x_i) + f_j(x_j).
    \]
- Update of \( z \) has closed form solution: can be easily computed in a distributed way.
Convergence

Assumption

(a) *(Infinitely often updates):* For all $k$ and all $l \in \{1, \ldots, W\}$, $\mathbb{P}(l \in \Psi^k) > 0$.

(b) *(Decoupled constraints):* Matrix $H$ is diagonal, matrix $D$ has full column rank and each constraint involves only one $x_i$.

Theorem

Let $\{x^k, z^k, p^k\}$ be the iterates generated by the general asynchronous ADMM algorithm. The sequence $\{x^k, z^k, p^k\}$ converges to a saddle point $(x^*, z^*, p^*)$ of the Lagrangian, i.e., $(x^k, z^k)$ converges to a primal optimal solution $(x^*, z^*)$ almost surely.

Proof Sketch

- Define auxiliary full information iterates $y^k$, $v^k$ and $\mu^k$.

\[
y^{k+1} \in \arg\min_{y \in X} \sum_{i=1}^{N} f_i(y_i) - (p^k - \beta Hz^k)'D_i y + \frac{\beta}{2} \|D_i y\|^2,
\]

\[
v^{k+1} \in \arg\min_{v \in Z} \sum_{l=1}^{W} -(p^k - \beta Dy^{k+1})' H_l v + \frac{\beta}{2} \|H_l v\|^2,
\]
Convergence Analysis – Idea

• In view of the decoupled structure of the constraints, active components of asynchronous iterates take the same value as full information iterates:

\[ x_{k+1}^{(\phi_k)} = y_{k+1}^{(\phi_k)}, \quad z_{k+1}^{(\psi_k)} = v_{k+1}^{(\psi_k)}, \quad p_{k+1}^{(\psi_k)} = \mu_{k+1}^{(\psi_k)}. \]

• Inactive components remain at their previous value:

\[ x_{k+1}^{(\phi_k)} = x_k^{(\phi_k)}, \quad z_{k+1}^{(\psi_k)} = z_k^{(\psi_k)}, \quad p_{k+1}^{(\psi_k)} = p_k^{(\psi_k)}. \]

• Using the Lyapunov function \( \frac{1}{2\beta} \left\| p_{k+1} - p^* \right\|^2 + \frac{\beta}{2} \left\| H(z_{k+1} - z^*) \right\|^2 \), we can show full information iterates converge to an optimal solution.

• To develop a Lyapunov function for the asynchronous iterates, define probabilities

\[ \lambda_I = P(I \in \Psi^k) \]

and weighted norm induced by matrix \( \bar{\Lambda} \) where \( \bar{\Lambda}_{II} = 1/\lambda_I \).

• Using supermartingale arguments, we show that the probability adjusted norm,

\[ \frac{1}{2\beta} \left\| p_{k+1} - p^* \right\|^2_{\bar{\Lambda}} + \frac{\beta}{2} \left\| H(z_{k+1} - z^*) \right\|^2_{\bar{\Lambda}} \]

serves as a Lyapunov function for the asynchronous iterates.
Image denoising

Given a noisy image measurement \( b \), recover the original image by solving the following problem:

\[
\min_x \frac{1}{2} \| x - b \|_2^2 + \lambda \| x \|_{TV},
\]

where \( \| x \|_{TV} = \sum_{i \sim j} |x_i - x_j| \).

Figure: Original cameraman figure.

Figure: Added white noise with standard deviation 25.
Image denoising

Recover the original image by solving the following problem:

$$\min_x \frac{1}{2} \|x - b_1\|_2^2 + \frac{1}{2} \|x - b_2\|_2^2 + \lambda \|x\|_{TV},$$

with asynchronous ADMM algorithm with 3 agents. Algorithm converged after 87 iterations.

Figure: Original cameraman figure.
Figure: Noisy image data in 2 parts.
Figure: Recovered using total variation denoising formula with $\lambda = 20$. 
Conclusions and Future Work

- We presented a number of different ideas and methodologies for designing distributed algorithms for multi-agent optimization problems.

- For general convex problems, we showed that sub gradient-type distributed methods converge at rate $O(1/\sqrt{k})$ whereas ADMM-type distributed methods converge at the much faster rate $O(1/k)$.

- Simulation results illustrate the superior performance of ADMM (even for network topologies with slow mixing).

Ongoing and Future Work:
- Second-order methods for multi-agent optimization problems.
- Online and dynamic distributed optimization problems.
- ADMM type algorithm for time-varying graph topology.