Lecture 5
Asynchronous Iterative Methods and Distributed Optimization over Graphs

Jie Lu (jielu@kth.se)

Richard Combes
Alexandre Proutiere

Automatic Control, KTH

September 19, 2013

Contraction Mapping

- $F : X \to X$ (X closed) is a contraction mapping if $\|F(x) - F(y)\| \leq \alpha\|x - y\|$, $\forall x, y \in X$ for some norm $\|\cdot\|$ and $\alpha \in [0, 1)$.
- A contraction mapping F has a unique fixed point $x^\star \in X$ (i.e., $F(x^*) = x^*$).
- For any initial $x(0) \in X$, the sequence $\{x(t)\}$ generated by the iterative method $x(t+1) = F(x(t))$ converges to $x^\star$ geometrically:

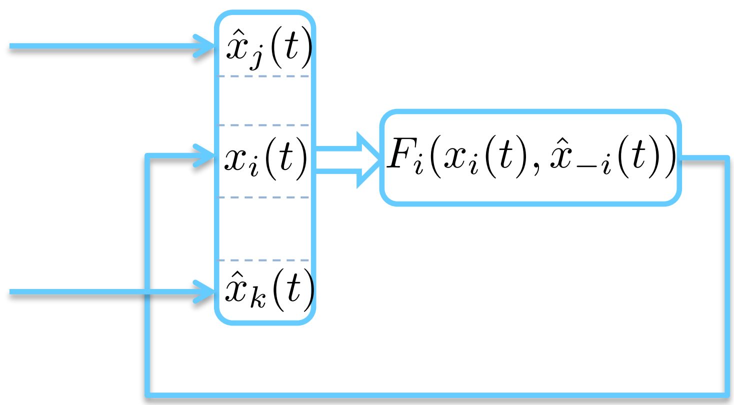
$$\|x(t) - x^\star\| \leq \alpha^t \|x(0) - x^\star\|, \quad \forall t \geq 0.$$

- if the following hold:
  (i) f is twice continuously differentiable
  (ii) $\frac{d\nabla_i f(x)}{dx_i} \leq L$ for some $L > 0$, $\forall x$, $\forall i$
  (iii) $\sum_{j \neq i} |\frac{d\nabla_i f(x)}{dx_j}| + \beta \leq |\frac{d\nabla_i f(x)}{dx_i}|$, $\forall x$, $\forall i$ for some $\beta > 0$
  ($\nabla^2 f$ satisfies a *diagonal dominance condition*),

  then the gradient mapping $F(x) = x - \alpha\nabla f(x)$ with $0 < \alpha < \frac{1}{L}$ is a maximum-norm contracion mapping.

Parallel Computations

- Executing iterative algorithms $x(t+1) = F(x(t))$ in parallel:
  - trivial when $F(\cdot)$ has structure, e.g. $F(x) = \sum_p F^{(p)}(x^{(p)})$
  - or when there is a central coordinator that maintains global state $F(x) = \sum_p F^{(p)}(x)$
- More challenging when state (decision variables) updates are distributed
- Component-wise parallelization: Each processor responsible for one decision variable, executes $x_i(t+1) = F_i(x_i(t), \hat{x}_{-i}(t))$



- Selected issues:
  - How to gather states from other processors?
  - What if this information is delayed, noisy, distorted?
  - How to account for asynchronous execution?

Asynchronous Model

- Let T be the set of event times, when some of the processors executes an update.
- Let $T^{(i)} \subseteq T$ be the event times when processor i updates its state
- $$x_i(t+1) = \begin{cases} F_i(x_1^{(i)}(\tau_1^{(i)}(t)), \ldots, x_i(t), \ldots, x_n^{(i)}(\tau_n^{(i)}(t))) & \text{if } t \in T^{(i)} \\ x_i(t) & \text{otherwise} \end{cases}$$
- $F_i : X \to X_i,\ X = X_1 \times X_2 \times \cdots \times X_n,\ F = (F_1, \ldots, F_n) : X \to X$
- $x_j^{(i)}(\tau_j^{(i)}(t))$ is the most recent version of x_j available to processor i at time t, and was computed at time $\tau_j^{(i)}(t) \in T^{(j)}$, $0 \leq \tau_j^{(i)}(t) \leq t$
- Information from other processors possibly delayed
- Accounts for asynchronicity and information delay.

Total Asynchronism

- Updates arbitrarily infrequent, information delays arbitrarily long

- Formally, the execution is ***totally asynchronous*** if
  - The update sets $T^{(i)}$ are infinite, and
  - For every sequence $\{t_k\} \in T^{(i)}$ with $\lim_{k \to \infty} t_k = \infty$, it also holds that $\lim_{k \to \infty} \tau_j^{(i)}(t_k) = \infty$

- No processor ceases to update and communicate its information.

Asynchronous Convergence Theorem

- **Theorem:** If there is a sequence of nonempty sets $\{X(t)\}$ with

$$\cdots \supset X(t-1) \supset X(t) \supset \cdots$$

satisfying

(Synchronous convergence condition)

$$F(x) \in X(t+1) \ \forall t, \ \forall x \in X(t)$$

and for every sequence $\{y(t)\}$ with $y(t) \in X(t) \ \forall t$, every limit point of $\{y(t)\}$ is a fixed point of F

(Box condition)

for every *t* there exists sets $X_i(t) \subset X_i$ such that

$$X(t) = X_1(t) \times X_2(t) \times \cdots \times X_n(t)$$

Then, if $x(0) \in X(0)$, then every limit point of $\{x(t)\}$ is a fixed point of *F*

Max-Norm Contractions Under Total Asynchronism

- Max-norm contraction: There exists $\alpha \in [0, 1)$ such that

$$\|F(x) - F(y)\|_\infty \leq \alpha \|x - y\|_\infty \quad \forall x, y \in X$$

  - Have unique fixed points, linear convergence rates.

- Also converge under total asynchronism, since

$$X(t) = \{x \in \mathbb{R}^n \mid \|x - x^\star\|_\infty \leq \alpha^t \|x(0) - x^\star\|_\infty\}$$

  satisfy the conditions of the asynchronous convergence theorem.

- The gradient method converges totally asynchronously when it is a max-norm contraction.

Partially Asynchronism

- An algorithm is called ***partially asynchronous*** if

  (i) For each i and t, $\{t, t+1, \ldots, t+D-1\} \cap T^{(i)} \neq \emptyset$

  (ii) $t - D < \tau_j^{(i)}(t) \leq t \ \forall t, \ \forall i, j$

- During every window of length D, each processor updates at least once
- The information used by any node is outdated with at most D time units

- If *f* is convex and has Lipschitz gradient (L > 0), then the gradient method

$$x(t+1) = x(t) - \alpha \nabla f(x(t))$$

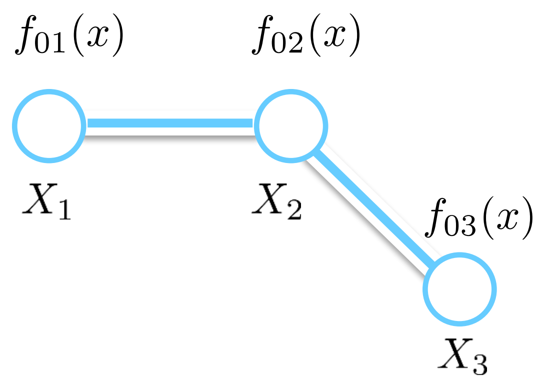
converges under partial asynchronism, provided that

$$\alpha \leq \frac{1}{L\Big(1 + (n+1)D\Big)}$$

Distributed Optimization over Graphs

- Convex optimization problem under (logical) communication constraints

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & \textstyle\sum_{i \in V} f_{0i}(x) \\ \text{subject to} \quad & x \in \cap_{i \in V} X_i, \\ & (V, E) = G. \end{aligned}$$



- Nodes can only exchange information with immediate neighbors in G.

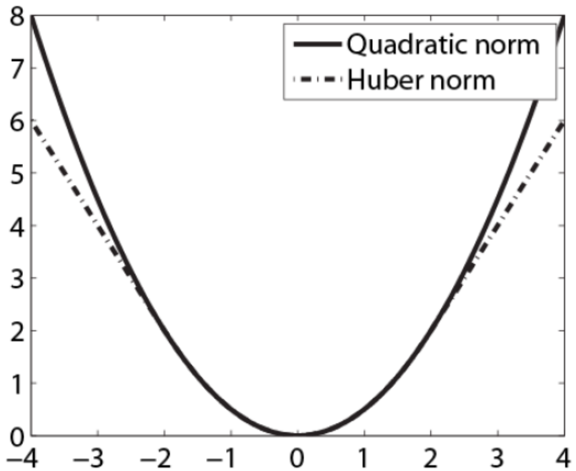
Example: robust estimation

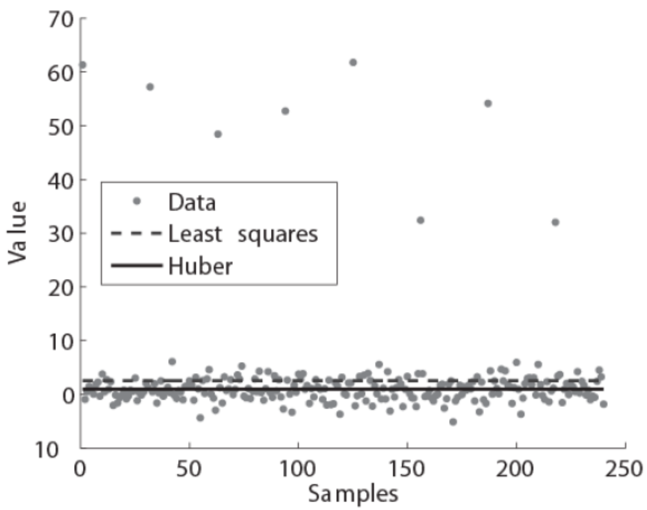
- Nodes measure different noisy versions $y_i(t)$ of the same quantity.

- Would like to agree on common estimate $\hat{x}$ that minimizes

$$\begin{array}{ll} \text{minimize} & \sum_{i \in V} \|y_i(t) - \hat{x}\|_H \\ \text{subject to} & x \in X \\ & (V, E) = G \end{array}$$

where $\|\cdot\|_H$ is the Huber loss



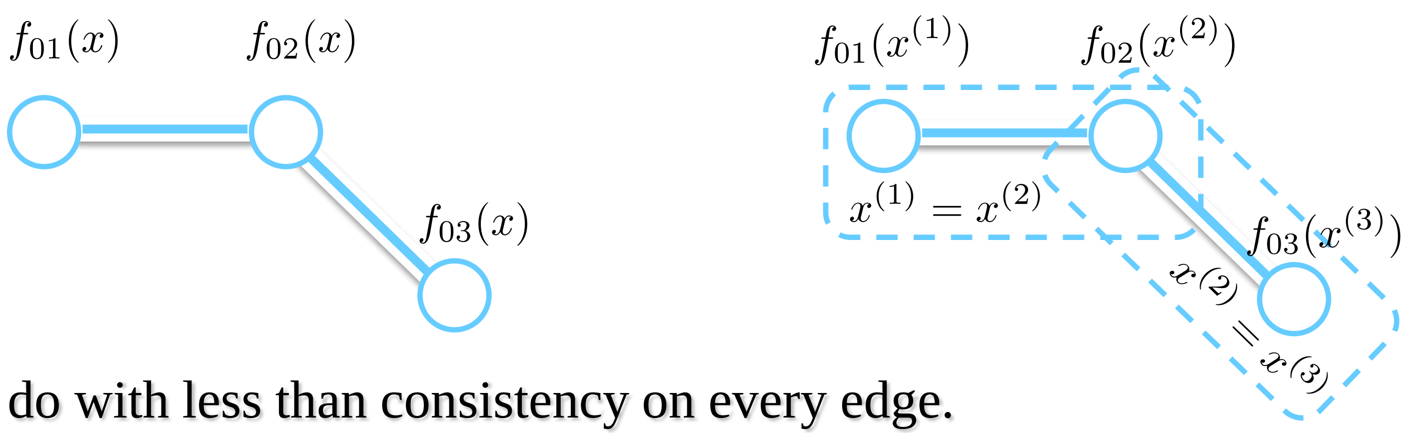


The Dual Approach

- Introduce local decision vector $x^{(i)}$ and re-write problem on the form

$$\begin{array}{lll} \text{minimize} & \sum_i f_{0i}(x^{(i)}) & \\ \text{subject to} & x^{(i)} = x^{(j)} & \forall\,(i,j) \in E \\ & x^{(i)} \in X_i & \end{array}$$

- Relax consistency constraints using Lagrange multipliers, solve dual problem.



- Can do with less than consistency on every edge.

A Primal Approach

- For simplicity, drop constraint and consider

$$\text{minimize} \quad \sum_{i=1}^{n} f_{0i}(x)$$

- Can we develop a solution approach that works directly with primal variables?

- Yes, if we introduce local decision vectors and reconcile "sufficiently" well

A Two-Step Approach

- Step 1: Nodes take step in gradient direction

$$\hat{x}^{(i)}(t+1) = x^{(i)}(t) - \alpha \nabla f_{0i}(x^{(i)})$$

- Step 2: Reconcile by forming network-wide average

$$x^{(i)}(t+1) = \frac{1}{n}\sum_{j=1}^{n} \hat{x}^{(j)}(t+1)$$

- Recovers standard gradient method

$$x^{(i)}(t+1) = \frac{1}{n}\sum_{j=1}^{n} x^{(j)}(t) - \alpha \sum_{j=1}^{n} \nabla f_{0j}(x^{(j)}) = x^{(i)}(t) - \frac{\alpha}{n}\sum_{j=1}^{n} \nabla f_{0j}(x^{(j)}(t))$$

- Network-averaging possible with peer-to-peer exchanges only

Distributed Averaging and Consensus

- Averaging can be performed distributedly

$$z^{(i)}(t+1) = a_{ii} z^{(i)}(t) + \sum_{j \in N_i} a_{ij} z^{(j)}(t)$$

- For appropriately chosen weights,

$$\lim_{T \to \infty} z^{(i)}(T) = \frac{1}{n} \sum_{i=1}^{n} z^{(i)}(0) = z_{\text{ave}}(0)$$

- Known as distribtued averaging or average consensus.

Consensus Algorithm

- For simplicity, consider scalar $z^{(i)}$. Re-write iterations on matrix form

$$z(t+1) = Az(t)$$

- Convergence to the average

$$\lim_{T\to\infty} z(T) = \lim_{T\to\infty} A^T z(0) = \frac{1}{n}\mathbf{1}\mathbf{1}^T z(0) = \mathbf{1} z_{\text{ave}}(0)$$

occurs if and only if A satisfies

$$\begin{aligned} \mathbf{1}^T A &= \mathbf{1}^T \\ A\mathbf{1} &= \mathbf{1} \\ \rho\left(A - \tfrac{1}{n}\mathbf{1}\mathbf{1}^T\right) &< 1 \end{aligned}$$

- Linear convergence rate governed by $\rho_2(A)$. Mixing time $T_{\text{mix}} \sim \dfrac{1}{\ln \rho_2^{-1}(A)}$

Convergence Rate of the Two-Step approach

- Each optimization step essentially takes T_{mix} iterations to execute

- So convergence time for strongly convex and L-Lipschitz gradient case is

$$\mathcal{O}(T_{\text{mix}} \ln(1/\varepsilon))$$

- Do we really need to converge to average before taking next step?

The Interleaved Version

- Can also consider an interleaved version (single consensus iteration)

$$x^{(i)}(t+1) = \frac{1}{|N_i|} \sum_{j \in N_i} x^{(j)}(t) - \alpha \nabla f_{0i}(x^{(i)})$$

- Can show that

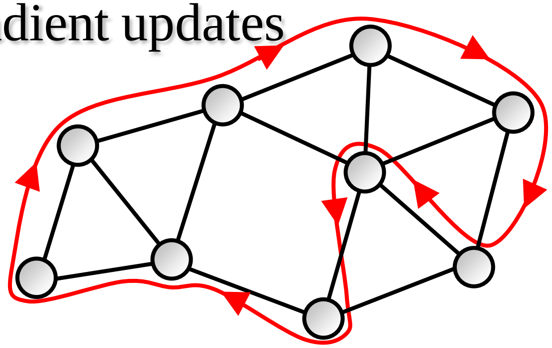
$$|x^{(i)}(t) - \overline{x}(t)| = \mathcal{O}(\alpha T_{\text{mix}} \sum_i |\nabla f_i(\overline{x}(t))|)$$

- Hence, for fixed step-size, error does not vanish at optimality.
- Typically studied for non-smooth or stochastic case
- Convergence rate estimates same flavor as two-phase version
- Versions that perform a multiple consensus steps also exist

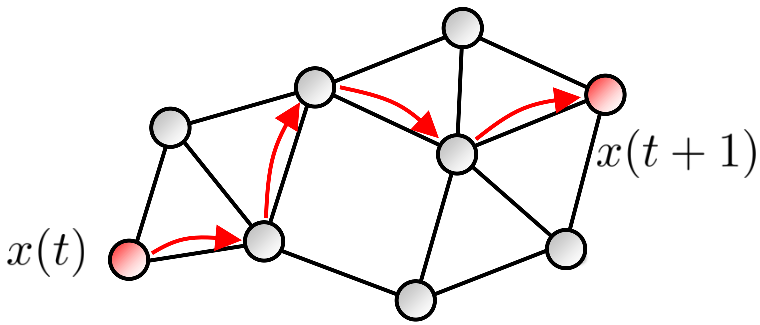
(B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, *Subgradient methods and consensus algorithms for solving convex optimization problems*, CDC 2008)

Alternative Methods

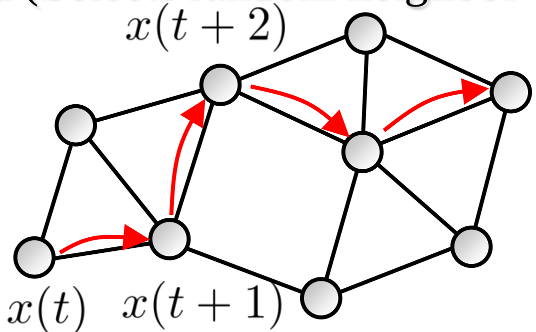
- **Incremental subgradient method**: Pass an estimate on the optimum over the network with subgradient updates
  - Cyclic



  - Uniform



  - Markov-chain-based (Selects random neighbor to update)



Dealing with a Global Constraint

- Resource allocation over a network

$$\begin{array}{ll} \text{minimize} & \sum_i f_{0i}(x_i) \\ \text{subject to} & \mathbf{1}^T x = 1 \\ & G = (V, E) \end{array}$$

- Gradient projection method

$$x(t+1) = P_X \left\{ x(t) - \alpha \nabla f_0(x(t)) \right\} =$$
$$= x(t) - \left(I - \frac{1}{n} \mathbf{1}\mathbf{1}^T \right) \alpha \nabla f(x(t))$$

- Consensus-based projection

$$x(t+1) = x(t) - (I - A^K) \alpha \nabla f(x(t))$$

  - Exact when $K \to \infty$

Dealing with a Global Constraint

- For a single consensus iteration per step,

$$x(t+1) = x(t) - (I - A)\alpha\nabla f(x(t))$$

- We recover the method by Ho et al. (Y. C. Ho, L. Servi, and R. Suri. *A class of center-free resource allocation algorithms. Large Scale Systems*, 1:51--62, 1980.)

$$x(t+1) = x(t) - W\nabla f(x(t))$$

where $W = \alpha(I - A)$ satisfies $\mathbf{1}^T W = 0,\ W\mathbf{1} = 0$

- Hence, resource constraint is satisfied at all times

$$\mathbf{1}^T x(t+1) = \mathbf{1}^T x(t) - \mathbf{1}^T W\nabla f(x(t)) = \mathbf{1}^T x(t)$$

Summary

- Asynchronous iterative methods
  - Models for asynchronous and distributed computation
    - Distribute iteration (e.g. gradient descent) on multiple processors
    - Different update rates, different communication delays
  - Total and partial asynchronism
    - Convergence results for totally asynchronous iterations
    - Gradient method under total and partial asynchronism

- Distributed optimization over graphs
  - Optimization with logical constraints: "who can communicate with whom"
  - Techniques for optimizing additive ("per agent") loss function
    - Dual decomposition
    - Two-step gradient descent/consensus
    - Interleaved gradient descent/consensus
    - An algorithm for maintaining a global constraint.

References

- Asynchronous Iterative methods
  - Dimitri P. Bertsekas and John N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, 1989. (Chapters 3, 6, 7)

- Distributed optimization
  - B. Yang and M. Johansson, *Distributed optimization and games: a tutorial overview*, In A. Bemporad, M. Heemels and M. Johansson, Eds., Networked Control Systems, 2010.
  - A. Nedic and A. Ozdaglar, *Cooperative Distributed Multi-Agent Optimization*, In Y. Eldar and D. Palomar, Eds., Convex Optimization in Signal Processing and Communications, Cambridge University Press, pp. 340-386, 2010.