A Consensus Approach to Distributed Convex Optimization in Multi-Agent Systems

Filippo Zanella

Ph.D. Defense

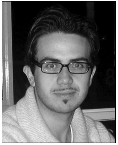
Department of Information Engineering - University of Padova

February 28^{th}, 2013

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Research team

Angelo Cenedese
Assistant Professor



Luca Schenato
Associate Professor

Damiano Varagnolo
Post-Doc

Ruggero Carli
Assistant Professor

Publications

F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, L. Schenato (2012)
Asynchronous Newton-Raphson Consensus for Distributed Convex Optimization
3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys'12)

F. Zanella, A. Cenedese (2012)
Multi-agent tracking in wireless sensor networks: model and algorithm
1st WSEAS International Conference on Information Technology and Computer Networks (ITCN'12)

F. Zanella, A. Cenedese (2012)
Multi-agent tracking in wireless sensor networks: implementation
1st WSEAS International Conference on Information Technology and Computer Networks (ITCN'12)

F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, L. Schenato (2012)
Multidimensional Newton-Raphson consensus for distributed convex optimization
American Control Conference (ACC'12)

F. Zanella, F. Pasqualetti, R. Carli, F. Bullo (2012)
Simultaneous Boundary Partitioning and Cameras Synchronization for Optimal Video Surveillance
3rd IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys'12)

Publications

F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, L. Schenato (2012)
The convergence rate of Newton-Raphson consensus optimization for quadratic cost functions
IEEE Conference on Decision and Control (CDC'12)

F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, L. Schenato (2011)
Newton-Raphson consensus for distributed convex optimization
IEEE Conference on Decision and Control (CDC'11)

F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, L. Schenato
Newton-Raphson Consensus for Distributed Convex Optimization
IEEE Transactions on Automatic Control (submitted)

F. Zanella, D. Varagnolo, A. Cenedese, G. Pillonetto, L. Schenato
Asynchronous Newton-Raphson Consensus for Distributed Convex Optimization
Automatica (to submit)

F. Zanella, A. Cenedese
Channel Model Identification in Wireless Sensor Networks Using a Fully Distributed Consensus Algorithm
Ad-Hoc Networks (submitted)

Publications

- F. Zanella, A. Cenedese
  Multi-agent tracking in wireless sensor networks
  WSEAS International Journal of Systems Engineering, Applications and Development

- F. Zanella, J. R. Peters, M. Spindler, F. Pasqualetti, R. Carli, and F. Bullo
  Distributed cameras synchronization for smart-intruder detection
  IEEE Transaction on Robotics (submitted)

- F. Zanella, A. Cenedese, F. Maran
  Teseo: a multi-agent tracking application in wireless sensor networks
  Ad-Hoc Networks (to submit)

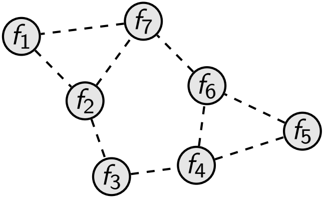
Outline

Table of Contents

Distributed optimization

Multi-agents scenario

collaboration to pursue a common goal: find the optimal common working point $\boldsymbol{x}^*$



Problem formulation

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \left[f(\boldsymbol{x}) := \sum_{i=1}^{N} f_i(\boldsymbol{x}) \right]$$ under ***convexity assumptions***

in an undirected and connected communication graph

Distribution optimization - Example 1

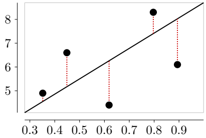
Regression in sensor networks

$$\min_{\boldsymbol{x}} \quad \sum_{i=1}^{N} \phi(y_i - \boldsymbol{u}_i^T \boldsymbol{x})$$

$y_i = \boldsymbol{u}_i^T \boldsymbol{x} + v_i$ linear *measurements* (output)

$\boldsymbol{u}_i$ is the i-th *feature* vector (independent variable)

v_i independent Gaussian noises

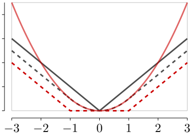


$$\phi(r) = |r|^2 \qquad \text{(least squares)}$$

$$\phi(r) = |r| \qquad \text{(least abs. deviations)}$$

$$\phi(r) = \begin{cases} 0 & \text{if } |r| < 1 \\ |r| - 1 & \text{otherwise} \end{cases} \qquad \text{(Vapnik)}$$

$$\phi(r) = \begin{cases} |r|^2 & \text{if } |r| < 1 \\ 2(|r| - 1) & \text{otherwise} \end{cases} \qquad \text{(Huber)}$$



Distribution optimization - Example 2

Classification in sensor networks

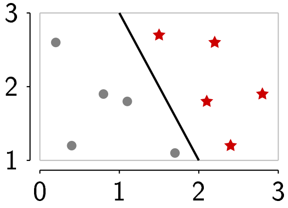
$$\min_{\boldsymbol{x}} \sum_{i=1}^{N} l_i \left(y_i \boldsymbol{u}_i^T \boldsymbol{x} \right) + \lambda \left\| \boldsymbol{x} \right\|^2$$

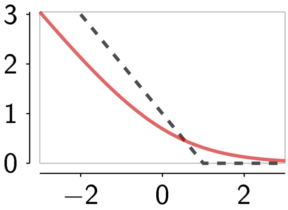
$y_i \in \{-1, 1\}$ is the binary outcome

$\boldsymbol{u}_i$ is the i-th *feature* vector (independent variable)

$l_i : \mathrm{R} \rightarrow \mathrm{R}$ is the convex loss (*Hinge*, exponential)

$\lambda \left\| \boldsymbol{x} \right\|^2$ is a *Tikhonov regularization*





State of the art

3 main categories:

- primal decompositions methods
  (e.g., distributed subgradients [Ozdaglar, Nedić, Lobel, . . .])
- dual decompositions methods
  (e.g., alternating direction method of multipliers [Bertsekas, Boyd, Johansson, . . .])
- tailored methods
  (e.g., Fast-Lipschitz [Fischione], control based approach [Wang-Elia], pairwise equalizing [Lu])

Primal decomposition methods (distributed)

Distributed subgradient methods (DSM) [?]

alternates consensus steps on $x_i(k)$ with subgradient updates

Algorithm

$$x_i(k+1) = \mathcal{P}_{\mathcal{X}}\left[\sum_{j=1}^{N} p_{ij}(k)x_j(k) + \rho_i(k)g_i\left(x_i(k)\right)\right]$$

$\sum_{j=1}^{N} p_{ij}(k)x_j(k) :=$ aver. consensus step on *local* estimates $x_j(k)$

$g_i\left(x_i(k)\right) :=$ *local* (bounded) subgradient of cost $f_i(\cdot)$ at $x_i(k)$

$\rho_i(k) :=$ *local* stepsize

$\mathcal{P}_{\mathcal{X}}$ projection onto the domain $\mathcal{X}$

If $\rho(k) = \rho$ then $\liminf_{k \to +\infty} f\left(x_i(k)\right) = f^* +$ small constant

If $\rho(k) = \rho/k$ it converges to the optimum x^*

Dual decomposition methods (distributed)

Alternating Direction Method of Multipliers (ADMM) [?]

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + f_2(x_2) \\ \text{subject to} & A_1 x_1 + A_2 x_2 - b = 0 \end{array}$$

Augmented Lagrangian:

$$\begin{aligned} L_\delta(x_1, x_2, \lambda) \;:=\; & f_1(x_1) + f_2(x_2) \\ & + \lambda^T (A_1 x_1 + A_2 x_2 - b) \\ & + \tfrac{\delta}{2} \|A_1 x_1 + A_2 x_2 - b\|_2^2 \end{aligned}$$

Algorithm

1. $\mathbf{x_1(k+1)} = \arg\min_{x_1} L_\delta(\mathbf{x_1}, x_2(k), \lambda(k))$
2. $\mathbf{x_2(k+1)} = \arg\min_{x_2} L_\delta(x_1(k+1), \mathbf{x_2}, \lambda(k))$
3. $\lambda(k+1) = \lambda(k) + \delta (A_1 x_1 + A_2 x_2 - b)$

Tailored methods (distributed)

Distributed Control Method (DCM) [?]

forces the states to the global optimum by controlling the subgradient of the global cost

- subgradient as an input/output map
- small gain theorems to guarantee the convergence
- $0 < \mu < \dfrac{2}{2\max_{i=\{1,\dots,N\}} |\mathcal{N}_i| + 1}$, $\nu > 0$ ensure system stability

Algorithm

1. $z_i(k+1) = z_i(k) + \mu \sum_{j \in \mathcal{N}_i} (x_i(k) - x_j(k))$, $\mathcal{N}_i$ neighbors of i
2. $$x_i(k+1) = x_i(k) + \mu \sum_{j \in \mathcal{N}_i} (x_j(k) - x_i(k)) + \mu \textstyle\sum_{j \in \mathcal{N}_i} (z_j(k) - z_i(k)) - \mu \, \nu \, g_i(x_i(k))$$

Tailored methods (distributed)

Pairwise Equalizing Method (PEM) [?]

a gossip-style, distributed asynchronous iterative algorithm that uses non-gradient-based update rules with no stepsize

- one-time sharing of the f_i's between gossiping agents
- symmetric-gossip communication between agents i and j
- computation of $(f_i' + f_j')^\dagger$, the inverse of $f_i' + f_j'$

Algorithm

1. $\boldsymbol{x}(0) = \arg\min_{\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x})$
2. $$\begin{aligned}\boldsymbol{x}(k+1) &= \boldsymbol{x}(k) - S(k)\Big(\boldsymbol{x}(k-1) \\ &\quad + \Psi[\boldsymbol{f}', k]^\dagger \left(2(I - P_{1/2}(k))\boldsymbol{f}'(\boldsymbol{x}(k))\right)\Big)\end{aligned}$$

Drawbacks of the considered algorithms

Primal based strategies

- may be slow
- may not converge to the optimum

Dual based strategies

- may be computationally expensive
- require topological knowledge
- hard to handle time-varying graphs and time delays

Tailored strategies

- may be slow
- may require complex computations

Motivations

There is no "perfect" algorithm for all situations or scenarios

The algorithm that we want:

1. assured to converge to global optimum
2. easy to be implemented
3. with small computational requirements
4. does not require synchronization or topology knowledge
5. inheriting good properties of standard consensus
   convergence proofs, robustness, . . .

Our position in literature

How the proposed algorithm relates to other techniques?

- primal decomposition method
- unconstrained convex optimization
- uses second-order approximations

Our contribute

- asynchronous algorithm
- better convergence speed for primal methods

Table of Contents

Illustrative example: quadratic local cost functions

Derivation of the algorithm - step 1 on 3

Simplified scalar scenario

$$f_i(x) = \frac{1}{2} a_i \left(x - b_i\right)^2 + c_i \qquad\qquad a_i > 0$$

Corresponding solution

$$x^* = \frac{\displaystyle\sum_{i=1}^{N} a_i b_i}{\displaystyle\sum_{i=1}^{N} a_i} = \frac{\displaystyle\frac{1}{N}\sum_{i=1}^{N} a_i b_i}{\displaystyle\frac{1}{N}\sum_{i=1}^{N} a_i}$$

i.e. ***parallel of 2 average consensus!***

Illustrative example: quadratic local cost functions

Derivation of the algorithm - step 1 on 3

Average consensus algorithm (P matrix)

Letting $y_i(0) := a_i b_i$ and $z_i(0) := a_i$

$$
\begin{aligned}
\boldsymbol{y}(k+1) &= P\,\boldsymbol{y}(k) \\
\boldsymbol{z}(k+1) &= P\,\boldsymbol{z}(k) \\
\boldsymbol{x}(k+1) &= \frac{\boldsymbol{y}(k+1)}{\boldsymbol{z}(k+1)}
\end{aligned}
$$

Then

$$
\lim_{k \to \infty} \boldsymbol{x}(k) = x^* \mathbb{1} = \frac{\sum_{i=1}^{N} y_i(0)}{\sum_{i=1}^{N} z_i(0)} \mathbb{1}
$$

And for generic convex local cost functions?

Derivation of the algorithm - step 2 on 3

Notice that in the quadratic case

- $a_i b_i = f_i''(x)x - f_i'(x) =: g_i(x)$
- $a_i = f_i''(x) =: h_i(x)$

$$x^* = \frac{1/N \sum_{i=1}^{N} a_i b_i}{1/N \sum_{i=1}^{N} a_i}$$

that leads to

$$\widehat{x}^* = \frac{1/N \sum_{i=1}^{N} f_i''(x)x - 1/N \sum_{i=1}^{N} f_i'(x)}{1/N \sum_{i=1}^{N} f_i''(x)} = x - \frac{f'(x)}{f''(x)}$$

intuition: that is a standard Newton-Raphson update step!

$\widehat{x}^*$ generally provides the right descent direction

Can we apply the same consensus strategy again?

And for generic convex local cost functions?

Derivation of the algorithm - step 2 on 3

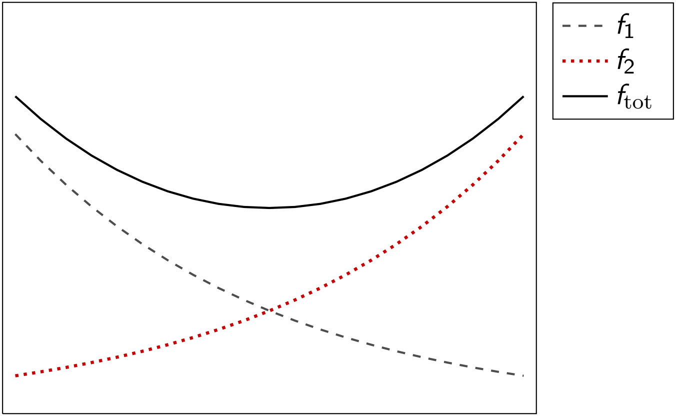
Set $y_i(0) = f_i''(x_i(0))x_i(0) - f_i'(x_i(0))$ and $z_i(0) = f_i''(x_i(0))$
Let each agent choose an $x_i(0)$ and apply again the consensus strategy to compute (up to convergence)

$$\widehat{x}^* = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(f_i''(x_i(0))x_i(0) - f_i'(x_i(0))\right)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x_i(0))} = \frac{\frac{1}{N}\sum_{i=1}^{N} g_i(x_i(0))}{\frac{1}{N}\sum_{i=1}^{N} h_i(x_i(0))}.$$

$$\lim_{k\to\infty} \boldsymbol{x}(k) = \widehat{x}^* \mathbb{1} = \frac{\sum_{i=1}^{N} y_i(0)}{\sum_{i=1}^{N} z_i(0)} \mathbb{1}$$

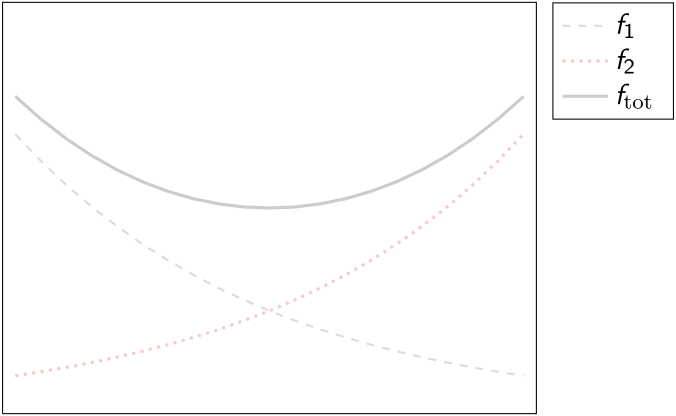
The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



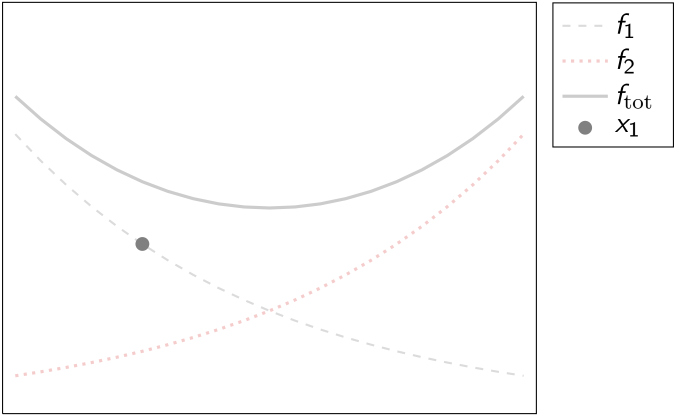
The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



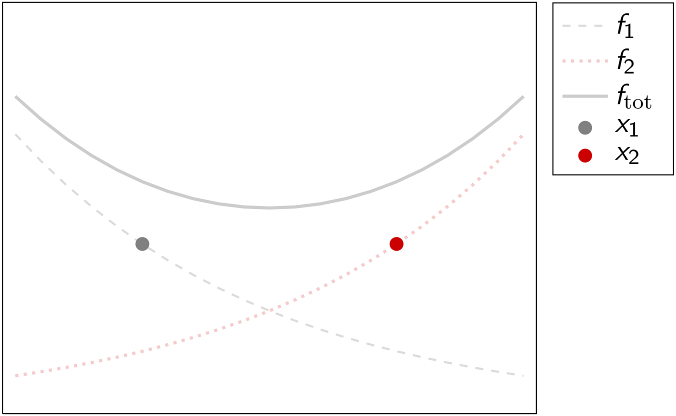
The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



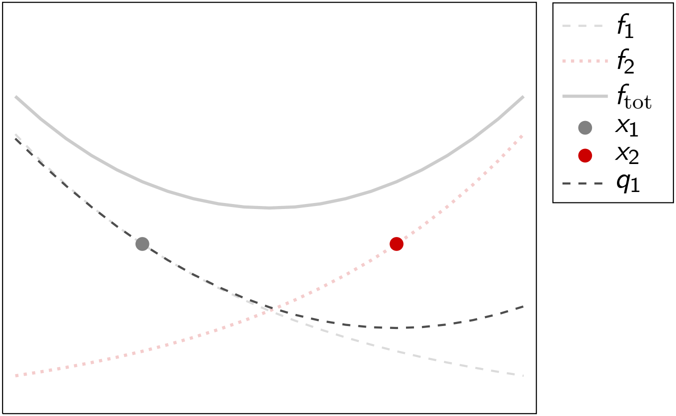
The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



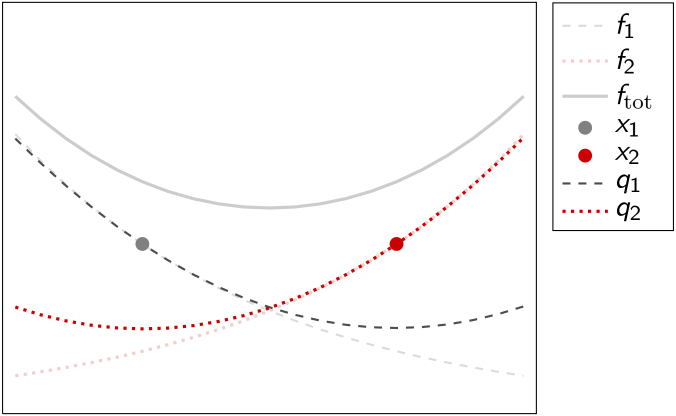
The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



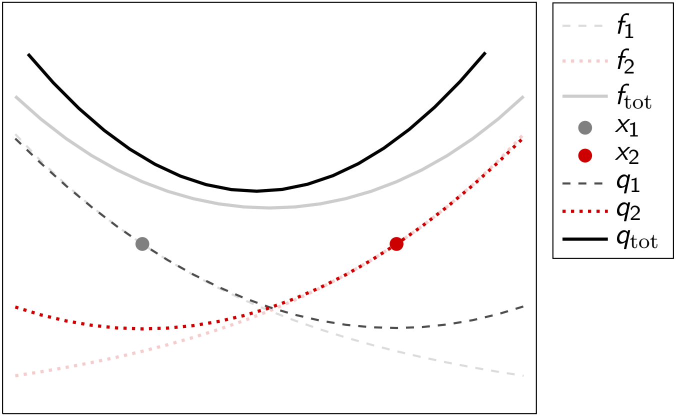
The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



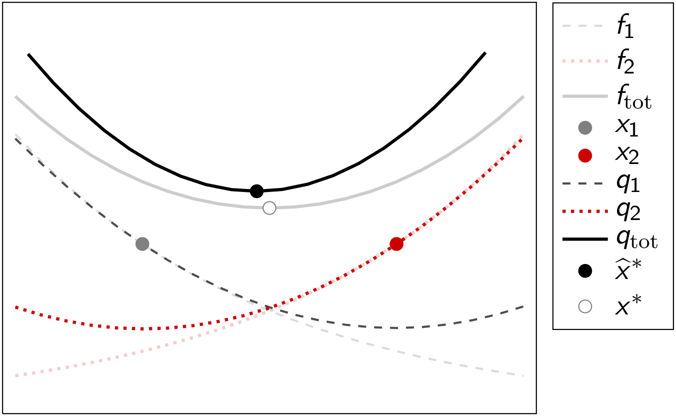
The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



The initial idea

Derivation of the algorithm - step 2 on 3 - graphical interpretation



$$\widehat{x}^* = \frac{\frac{1}{N}\sum_{i=1}^{N} a_i b_i}{\frac{1}{N}\sum_{i=1}^{N} a_i} \approx \frac{\frac{1}{N}\sum_{i=1}^{N}\left(f_i''(x)x_i - f_i'(x)\right)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x)}$$

intuition: $\widehat{x}^*$ is an accurate guess of x^*!

The initial idea

Derivation of the algorithm - step 2 on 3

We have seen that:

1) if all the $x_i(0)$ are equal, i.e., $x_i(0) = x$, $\forall i$, $\widehat{x}^*$ behaves like a Newton-Raphson

$$\widehat{x}^* = x - \frac{f'(x)}{f''(x)}$$

2) depending on the initial conditions $x_i(0)$, $\widehat{x}^*$ is a sensible estimation of x^*

$$\widehat{x}^* = \frac{\frac{1}{N}\sum_{i=1}^{N} g_i\big(x_i(0)\big)}{\frac{1}{N}\sum_{i=1}^{N} h_i\big(x_i(0)\big)}.$$

To get the global optimum we can alternate steps that compute the averages of the g_i's and h_i's and steps that update the local x_i's

The initial idea

Derivation of the algorithm - step 3 on 3 - analysis of the problems

How do we modify the consensus strategy?

1. initialization:
   - $y_i(0) := f_i''(x_i(0))x_i(0) - f_i'(x_i(0)) = g_i\big(x_i(0)\big)$
   - $z_i(0) := f_i''(x_i(0)) = h_i\big(x_i(0)\big)$
2. ***average consensus*** (in $\|$, P doubly stochastic):
   - $\boldsymbol{y}(k+1) = P\boldsymbol{y}(k)$
   - $\boldsymbol{z}(k+1) = P\boldsymbol{z}(k)$
3. local updates: $x_i(k+1) = \dfrac{y_i(k+1)}{z_i(k+1)}$

The initial idea

Derivation of the algorithm - step 3 on 3 - analysis of the problems

How do we modify the consensus strategy?

1. initialization:
   - $y_i(0) := f_i''(x_i(0))x_i(0) - f_i'(x_i(0)) = g_i\big(x_i(0)\big)$
   - $z_i(0) := f_i''(x_i(0)) = h_i\big(x_i(0)\big)$
2. ***average consensus*** (in $\|$, P doubly stochastic):
   - $\boldsymbol{y}(k+1) = P\boldsymbol{y}(k)$
   - $\boldsymbol{z}(k+1) = P\boldsymbol{z}(k)$
3. local updates: $x_i(k+1) = \dfrac{y_i(k+1)}{z_i(k+1)}$

We must provide 2 little modifications:

How do we modify the consensus strategy?

1. initialization:
    - $y_i(0) := f_i''(x_i(0))x_i(0) - f_i'(x_i(0)) = g_i\big(x_i(0)\big)$
    - $z_i(0) := f_i''(x_i(0)) = h_i\big(x_i(0)\big)$
2. ***average consensus*** (in $\|$, P doubly stochastic):
    - $\boldsymbol{y}(k+1) = P\boldsymbol{y}(k)$
    - $\boldsymbol{z}(k+1) = P\boldsymbol{z}(k)$
3. local updates: $x_i(k+1) = \dfrac{y_i(k+1)}{z_i(k+1)}$

We must provide 2 little modifications:

- x_i changes! $\Rightarrow$ must track the changing $f_i'(x_i)$ and $f_i''(x_i)$

The initial idea

Derivation of the algorithm - step 3 on 3 - analysis of the problems

How do we modify the consensus strategy?

1. initialization:
   - $y_i(0) := f_i''(x_i(0))x_i(0) - f_i'(x_i(0)) = g_i\big(x_i(0)\big)$
   - $z_i(0) := f_i''(x_i(0)) = h_i\big(x_i(0)\big)$
2. ***average consensus*** (in $\|$, P doubly stochastic):
   - $\boldsymbol{y}(k+1) = P\boldsymbol{y}(k)$
   - $\boldsymbol{z}(k+1) = P\boldsymbol{z}(k)$
3. local updates: $x_i(k+1) = \dfrac{y_i(k+1)}{z_i(k+1)}$

We must provide 2 little modifications:

- x_i changes! $\Rightarrow$ must track the changing $f_i'(x_i)$ and $f_i''(x_i)$
- $x_i(k) = \dfrac{y_i(k)}{z_i(k)}$ too aggressive! $\Rightarrow$ should make it milder

Table of Contents

The Newton-Raphson Consensus (NRC) algorithm

1. quadratic approximations update:
   - $g_i(x_i(k)) = f_i''(x_i(k))x_i(k) - f_i'(x_i(k))$
   - $h_i(x_i(k)) = f_i''(x_i(k))$
2. initialization:

   $\mathbf{x}(k) = \mathbf{y}(k) = \mathbf{z}(k) = \mathbf{g}(\mathbf{x}(-1)) = \mathbf{h}(\mathbf{x}(-1)) = \mathbf{0}$
3. quadratic approximations mixing:
   - $\boldsymbol{y}(k+1) = P\left[\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{k}) - \boldsymbol{g}(\boldsymbol{k}-\mathbf{1})\right]$
   - $\boldsymbol{z}(k+1) = P\left[\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{k}) - \boldsymbol{h}(\boldsymbol{k}-\mathbf{1})\right]$
4. guesses updates:
   - $\boldsymbol{x}(k+1) = (1-\varepsilon)\boldsymbol{x}(k) + \varepsilon \dfrac{\boldsymbol{y}(k+1)}{\boldsymbol{z}(k+1)}$

The Newton-Raphson Consensus (NRC) algorithm

1. quadratic approximations update:
   - $g_i(x_i(k)) = f_i''(x_i(k))x_i(k) - f_i'(x_i(k))$
   - $h_i(x_i(k)) = f_i''(x_i(k))$

2. initialization:

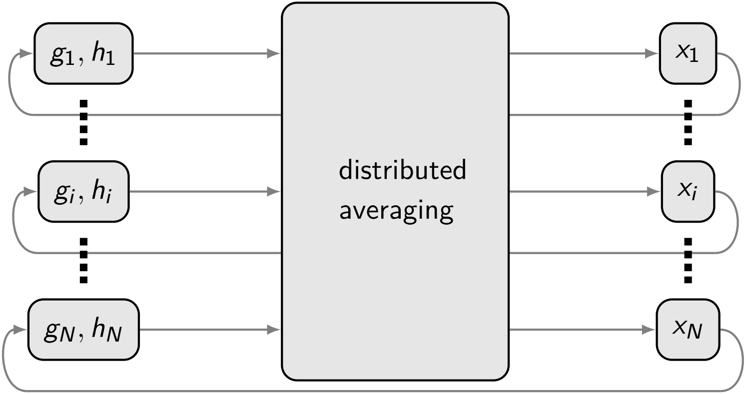
   $\mathbf{x}(k) = \mathbf{y}(k) = \mathbf{z}(k) = \mathbf{g}(\mathbf{x}(-1)) = \mathbf{h}(\mathbf{x}(-1)) = \mathbf{0}$

3. quadratic approximations mixing:
   - $\boldsymbol{y}(k+1) = P\left[\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{k}) - \boldsymbol{g}(\boldsymbol{k}-\mathbf{1})\right]$
   - $\boldsymbol{z}(k+1) = P\left[\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{k}) - \boldsymbol{h}(\boldsymbol{k}-\mathbf{1})\right]$

4. guesses updates:
   - $\boldsymbol{x}(k+1) = (1-\varepsilon)\boldsymbol{x}(k) + \varepsilon\dfrac{\boldsymbol{y}(k+1)}{\boldsymbol{z}(k+1)}$

Important remark: Step 3 can be substituted with any asymptotical average consensus algorithm

Block schematic representation

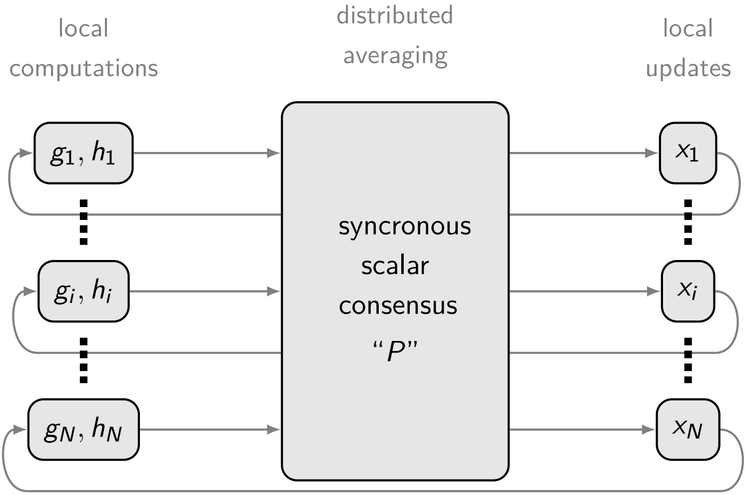


$$g_i(k) = f_i''(x_i(k))x_i(k) - f_i'(x_i(k))$$
$$h_i(k) = f_i''(x_i(k))$$

$$x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \frac{y_i(k+1)}{z_i(k+1)}$$

need just uniformly exponentially converging av. consensus

NRC - Block scheme



$$g_i(k) = f_i''(x_i(k))x_i(k) - f_i'(x_i(k))$$
$$h_i(k) = f_i''(x_i(k))$$

$$x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \frac{y_i(k+1)}{z_i(k+1)}$$

Convergence theorem

Hypotheses

- $f_i \in \mathcal{C}^2(\mathbb{R})$
- f_i' and f_i'' bounded
- f_i strictly convex
- $x^* \neq \pm\infty$
- null initial conditions

Thesis

- there is a positive $\bar{\varepsilon}$ s.t. if $\varepsilon < \bar{\varepsilon}$ then, exponentially,

$$\lim_{k \to +\infty} \mathbf{x}(k) = x^* \mathbb{1}$$

Sketch of the proof

importance of the proof:
gives insights on key properties

1. transform the algorithm in a continuous-time system
2. recognize the existence of a two-time scales dynamical system
3. analyze separately fast and slow dynamics
   (standard singular perturbation model analysis approach [**?**])

Sketch of the proof

Transformation in a continuous-time system

$$\begin{cases} \mathbf{y}(k+1) = P\big(\mathbf{y}(k) + \mathbf{g}(\mathbf{x}(k)) - \mathbf{g}(\mathbf{x}(k-1))\big) \\ \mathbf{z}(k+1) = P\big(\mathbf{z}(k) + \mathbf{h}(\mathbf{x}(k)) - \mathbf{h}(\mathbf{x}(k-1))\big) \\ \mathbf{x}(k+1) = (1-\varepsilon)\mathbf{x}(k) + \varepsilon \dfrac{\mathbf{y}(k+1)}{\mathbf{z}(k+1)} \end{cases}$$

$$\downarrow P = I - K$$

$$\begin{cases} \varepsilon \dot{\mathbf{v}}(t) = -\mathbf{v}(t) + \mathbf{g}\left(\mathbf{x}(t)\right) \\ \varepsilon \dot{\mathbf{w}}(t) = -\mathbf{w}(t) + \mathbf{h}\left(\mathbf{x}(t)\right) \\ \varepsilon \dot{\mathbf{y}}(t) = -K\mathbf{y}(t) + (I-K)\left[\mathbf{g}\left(\mathbf{x}(t)\right) - \mathbf{v}(t)\right] \\ \varepsilon \dot{\mathbf{z}}(t) = -K\mathbf{z}(t) + (I-K)\left[\mathbf{h}\left(\mathbf{x}(t)\right) - \mathbf{w}(t)\right] \\ \dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \dfrac{\mathbf{y}(t)}{\mathbf{z}(t)} \end{cases}$$

Sketch of the proof

Two-time scales dynamical system

$$\begin{cases} \varepsilon\dot{\mathbf{v}}(t) = -\mathbf{v}(t) + \mathbf{g}\left(\mathbf{x}(t)\right) \\ \varepsilon\dot{\mathbf{w}}(t) = -\mathbf{w}(t) + \mathbf{h}\left(\mathbf{x}(t)\right) \\ \varepsilon\dot{\mathbf{y}}(t) = -K\mathbf{y}(t) + (I - K)\left[\mathbf{g}\left(\mathbf{x}(t)\right) - \mathbf{v}(t)\right] \\ \varepsilon\dot{\mathbf{z}}(t) = -K\mathbf{z}(t) + (I - K)\left[\mathbf{h}\left(\mathbf{x}(t)\right) - \mathbf{w}(t)\right] \\ \hline \\ \dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \dfrac{\mathbf{y}(t)}{\mathbf{z}(t)} \end{cases}$$

If ε is sufficiently small ...

first subsystem is much faster than second one

Sketch of the proof

Boundary layer system (fast dynamics)

t and $\mathbf{x}$ are "frozen" parameters (stretched timeline)

$$\begin{cases} \mathbf{v}(t) \to \mathbf{g}\left(\mathbf{x}(t)\right) \\ \mathbf{w}(t) \to \mathbf{h}\left(\mathbf{x}(t)\right) \\ \mathbf{y}(t) \to \left(\frac{1}{N}\mathbb{1}^T\mathbf{g}\left(\mathbf{x}(t)\right)\right)\mathbb{1} \\ \mathbf{z}(t) \to \left(\frac{1}{N}\mathbb{1}^T\mathbf{h}\left(\mathbf{x}(t)\right)\right)\mathbb{1} \end{cases}$$

If ε is sufficiently small . . .

the system is globally exponentially stable

Sketch of the proof

Reduced system (slow dynamics)

$\varepsilon = 0$ "instantaneous" fast transient

$$\dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \frac{\frac{1}{N}\mathbb{1}^T \mathbf{g}\left(\mathbf{x}(t)\right)}{\frac{1}{N}\mathbb{1}^T \mathbf{h}\left(\mathbf{x}(t)\right)}\mathbb{1}, \Longrightarrow \mathbf{x}(t) \to \overline{x}(t)\mathbb{1}$$

If ε is sufficiently small ...

$$\dot{\overline{x}}(t) \approx -\frac{f'\left(\overline{x}(t)\right)}{f''\left(\overline{x}(t)\right)} = -\frac{\sum_{i=1}^N f_i'\left(\overline{x}(t)\right)}{\sum_{i=1}^N f_i''\left(\overline{x}(t)\right)}$$

i.e. a continuous-time Newton-Raphson strategy

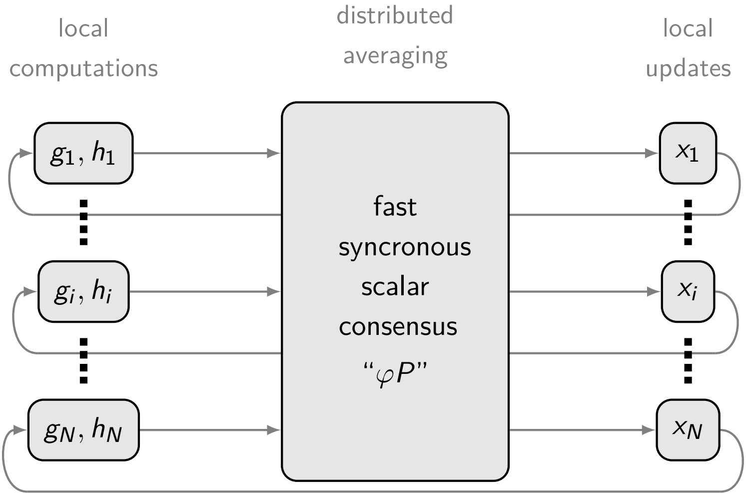
Fast Newton-Raphson Consensus (FNRC)

accelerated version of the NRC, based on the *second order diffusive schedules* [?]

- $\varphi = \dfrac{2}{1 + \sqrt{1 - \lambda_2^2}}$ (gradient and the memory weight)
- $\widetilde{\boldsymbol{y}}(0) = \widetilde{\boldsymbol{z}}(0) = \boldsymbol{0}$ (initialization)

$$\begin{cases} \widetilde{\boldsymbol{y}}(k) = \boldsymbol{y}(k-1) + \boldsymbol{g}\left(\boldsymbol{x}(k-1)\right) - \boldsymbol{g}\left(\boldsymbol{x}(k-2)\right) \\ \widetilde{\boldsymbol{z}}(k) = \boldsymbol{z}(k-1) + \boldsymbol{h}\left(\boldsymbol{x}(k-1)\right) - \boldsymbol{h}\left(\boldsymbol{x}(k-2)\right) \\ \boldsymbol{y}(k) = \varphi P\, \widetilde{\boldsymbol{y}}(k) + (1-\varphi)\, \widetilde{\boldsymbol{y}}(k-1) \\ \boldsymbol{z}(k) = \varphi P\, \widetilde{\boldsymbol{z}}(k) + (1-\varphi)\, \widetilde{\boldsymbol{z}}(k-1) \\ \boldsymbol{x}(k) = (1-\varepsilon)\boldsymbol{x}(k-1) + \varepsilon \dfrac{\boldsymbol{y}(k)}{\boldsymbol{z}(k)} \end{cases}$$

FNRC - Block scheme

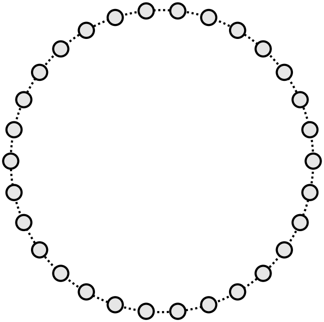


$$g_i(k) = f_i''(x_i(k))x_i(k) - f_i'(x_i(k))$$
$$h_i(k) = f_i''(x_i(k))$$

$$x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \frac{y_i(k+1)}{z_i(k+1)}$$

Experiments description

- circulant graph, $N = 30$
- $P = \begin{bmatrix} 0.5 & 0.25 & & & 0.25 \\ 0.25 & 0.5 & 0.25 & & \\ & \ddots & \ddots & \ddots & \\ & & 0.25 & 0.5 & 0.25 \\ 0.25 & & & 0.25 & 0.5 \end{bmatrix}$
- f_i = sum of exponentials

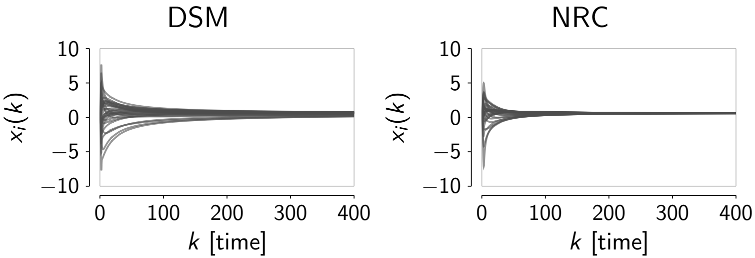


Comparisons with a Distributed Subgradient

Nedić Ozdaglar *Dist. subgr. meth. for multi-agent opt.* (2009)

1. $\mathbf{x}^{(c)}(k) = P\mathbf{x}(k)$ (consensus step)
2. $x_i(k+1) = x_i^{(c)}(k) - \frac{\rho}{k} f_i'\left(x_i^{(c)}(k)\right)$ (local gradient descent)

Numerical comparison

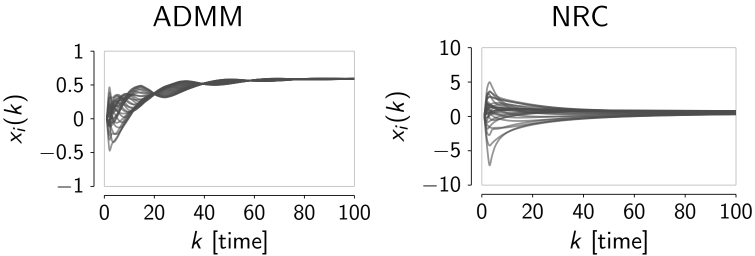


Comparisons with (an) ADMM

Bertsekas Tsitsiklis, *Parall. and Dist. Computation* (1997)

$$
\begin{aligned}
L(x,k) := \quad & \textstyle\sum_i \Big[f_i\,(x_i) + y_i^{(\ell)}(k)\,(x_i - z_{i-1}(k)) + y_i^{(c)}(k)\,(x_i - z_i(k)) \\
& + y_i^{(r)}(k)\,(x_i - z_{i+1}(k)) + \tfrac{\delta}{3}\,|x_i - z_{i-1}(k)|^2 + \\
& \tfrac{\delta}{3}\,|x_i - z_i(k)|^2 + \tfrac{\delta}{3}\,|x_i - z_{i+1}(k)|^2 \Big] \\
x(k+1) \quad & = \arg\min_x L(x,k)
\end{aligned}
$$

Numerical comparison



Overall comparison of the square error

- all the algorithms converge to the global optimum
- DSM ($\rho = 100$) is the slowest to converge
- DCM ($\mu = 0.25$) is significantly faster than DSM
- DCM is slower than NRC ($\varepsilon = 0.9$)
- FNRC ($\varepsilon = 0.9$) and ADMM ($\delta = 0.01$) converge in a comparable amount of time

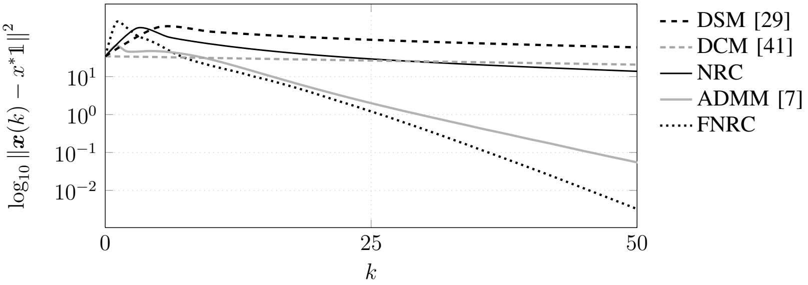
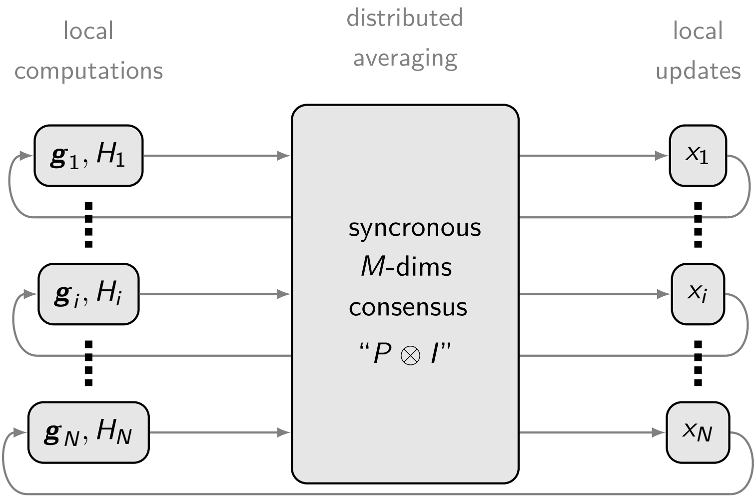


Table of Contents

Multidimensional scenario - Block scheme



$g_i(k) := H_i(k) x_i(k) - \nabla f_i(x_i(k))$

$H_i(k)$ to be defined

$x_i(k+1) = (1-\varepsilon) x_i(k) + \varepsilon (Z_i(k+1))^{-1} y_i(k+1)$

Different approaches

Newton-Raphson Consensus

$$H_i(k) := \nabla^2 f_i(\boldsymbol{x}_i(k)) \implies \dot{\overline{x}} \approx -\left(\nabla^2 f(\overline{x})\right)^{-1} \nabla f(\overline{x})$$

Jacobi Consensus

$$H_i(k) := \begin{bmatrix} \left.\frac{\partial^2 f_i}{\partial x_1^2}\right|_{\boldsymbol{x}_i(k)} & & 0 \\ & \ddots & \\ 0 & & \left.\frac{\partial^2 f_i}{\partial x_N^2}\right|_{\boldsymbol{x}_i(k)} \end{bmatrix} \implies \dot{x}_{\mathrm{a}} \approx -\left(\mathrm{diag} \nabla^2 f(\overline{x})\right)^{-1} \nabla f(\overline{x})$$

Gradient Descent Consensus

$$H_i(k) := I \implies \dot{\overline{x}} \approx -\nabla f(\overline{x})$$

Tradeoffs

Cost associated to the previous strategies

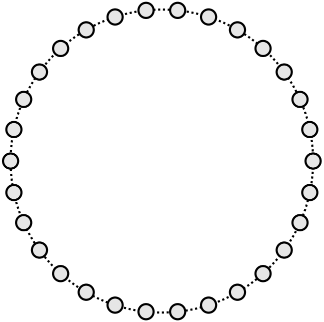
Algorithm	NRC	JC	GDC
Computational Cost	$O\left(M^3\right)$	$O\left(M\right)$	$O\left(M\right)$
Communication Cost	$O\left(M^2\right)$	$O\left(M\right)$	$O\left(M\right)$
Memory Cost	$O\left(M^2\right)$	$O\left(M\right)$	$O\left(M\right)$

approximations of the Hessians that do not maintain symmetry and positive definiteness or are bad conditioned require additional modification steps (e.g. Cholesky)

Numerical examples

- circulant graph, $N = 30$, $M = 2$
- P as in the scalar case
- $f_i(\mathbf{x}) = \text{Exp}\left((\mathbf{x} - \mathbf{b}_i)^T A_i (\mathbf{x} - \mathbf{b}_i)\right)$
  $\mathbf{b}_i \sim [\mathcal{U}[-5,5], \mathcal{U}[-5,5]]^T$,
  $A_i = D_i D_i^T > 0$,

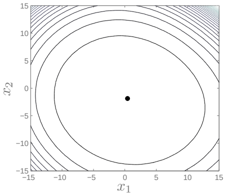
$$D_i = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \in \mathbb{R}^{2\times 2} .$$



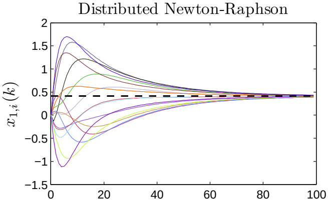
First scenario

The axes are randomly (uniformly) oriented in the 2-D plane

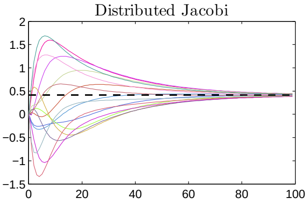
Global fun.



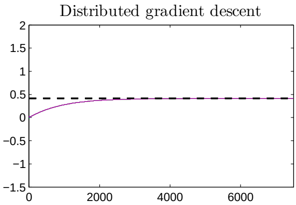
NRC



JC

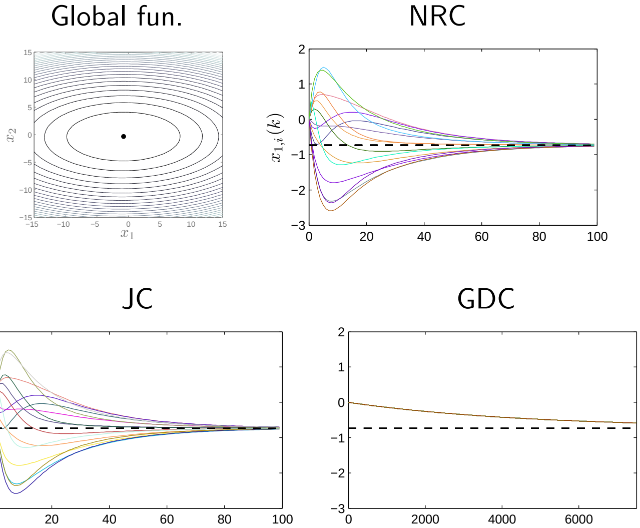


GDC



Second scenario

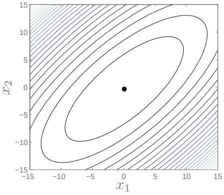
The axes are aligned with the the reference system



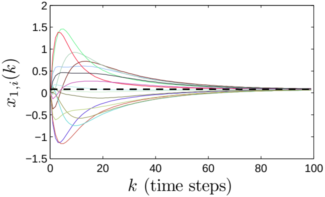
Third scenario

The axes are preferentially aligned with bisector of 1st-3th quadrant

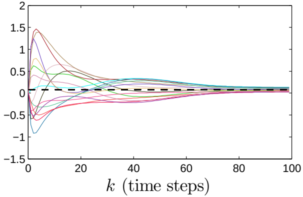
Global fun.



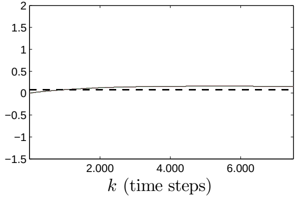
NRC



JC

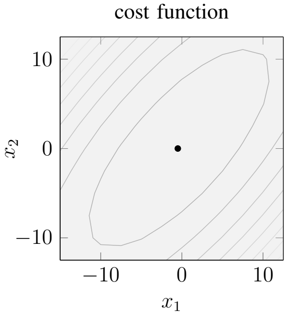


GDC



Square error comparison (1st-3th quadrant alignment)

- evident differences between NRC and JC ($\varepsilon = 0.25$ for both)
- GDC ($\varepsilon = 1$) presents a slower convergence rate



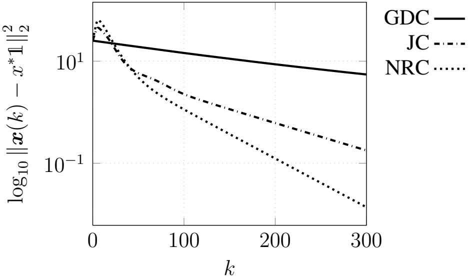
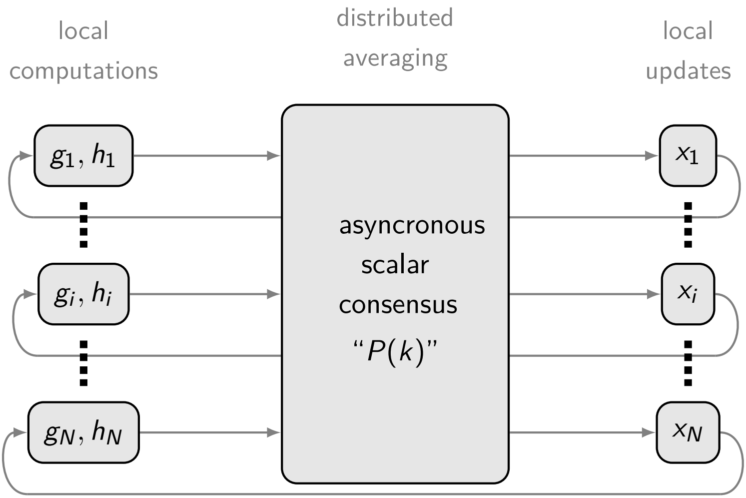


Table of Contents

Asynchronous NRC (ANRC) - Block scheme



$$g_i(k) = f_i''(x_i(k))x_i(k) - f_i'(x_i(k))$$
$$h_i(k) = f_i''(x_i(k))$$

$$x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \frac{y_i(k+1)}{z_i(k+1)}$$

Convergence

Theorem

uniform activation$^{(1)}$ $\Rightarrow$ ***global convergence***

(1): on the long run all the nodes are activated
the *same number* of times

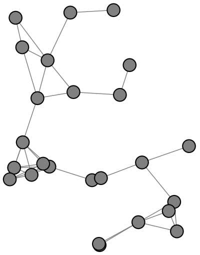
Theorem

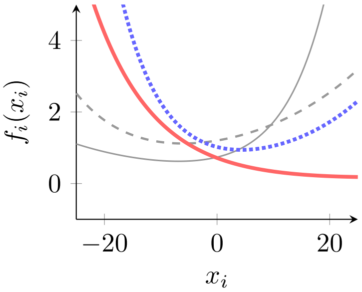
persistent activation$^{(2)}$ $\Rightarrow$ ***local convergence***

(2): every agent is activated *at least once*
in every sufficient large time windows

Experiments description

- $N = 25$
- (complete) random geometric graph
- activation sequence as independent permutations of agents/edges (ensuring uniform activation)
- f_i = sum of exponentials

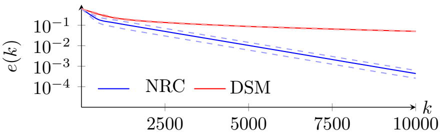




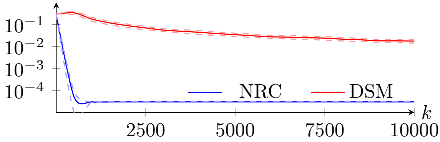
Comparison of the mean error (Montecarlo trial)

$$e(k) := \frac{1}{MN}\sum_{m=1}^{M} ||\boldsymbol{x}_m(k) - x^*\mathbb{1}||, \quad \text{with } M \text{ independent trials}$$

- algorithms always converge to x^*
- topology of the network play a crucial role on the convergence
- ANRC ($\varepsilon = 0.15$) statistically better than the DSM ($\rho = 100$)



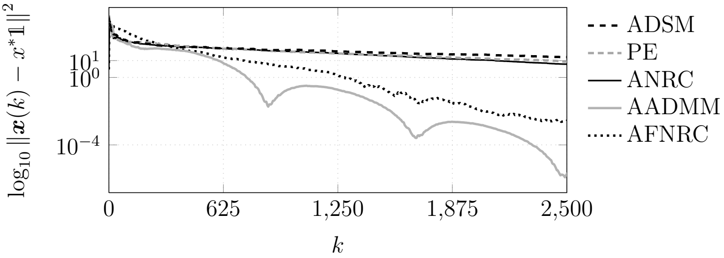
Incomplete graph



Complete graph

Comparison of the square error - random graph

- all the algorithms converge to the global optimum
- ADSM ($\rho = 30$) is the slowest to converge
- PE is comparable to DSM
- ANRC is significantly slower than ADMM ($\varepsilon = 0.9$)
- AFNRC ($\varepsilon = 0.1$, $\varphi = 1.55$) is closed to ADMM
- ADMM ($\delta = 0.05$) is the fastest one



Comparison of the square error - complete graph

- all the algorithms converge to the global optimum
- ADSM ($\rho = 55$) is the slowest to converge
- PE is better than ADSM but slower than AADMM
- ANRC and AFNRC ($\varepsilon = 0.9$, $\varphi = 1$) are identical
- ADMM ($\delta = 0.001$) is comparable to ANRC

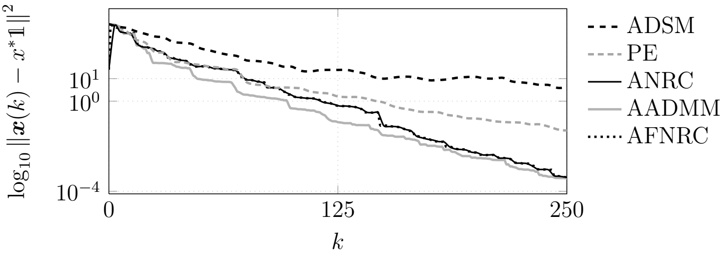


Table of Contents

Conclusions

The algorithm we proposed ...

- is a distributed Newton-Raphson strategy (+)
- requires minimal network topology knowledge (+)
- requires minimal agents synchronization (+)
- is simple to be implemented (+)
- converges to global optimum under convexity and smoothness assumptions (+ / -)
- is numerically faster than subgradients (+) but slower than ADMM (-) (if not speeded up)

Future works

Principal open problems

- analytical characterization of the convergence speed
  (with comparisons to other methods)
- relax the assumptions
  (strict convexity, $\mathcal{C}^2$, ...)
- tune ε on-line

Generalizations

- constrained optimization
  (e.g. barrier functions)
- distributed consensus with non-linearities
  (e.g. delays, link failures, packet losses, ...)

A Consensus Approach to Distributed Convex Optimization in Multi-Agent Systems

Filippo Zanella

Ph.D. Defense

Department of Information Engineering - University of Padova

February 28$^{\mathrm{th}}$, 2013

filippo.zanella@dei.unipd.it

www.dei.unipd.it/∼fzanella/

K. C. Kiwiel (2004)
Convergence of approximate and incremental subgradient methods for convex optimization
SIAM Journal on Optimization

D. P. Bertsekas and J. N. Tsitsiklis (1997)
Parallel and Distributed Computation: Numerical Methods
Athena Scientific

J. Wang and N. Elia (2010)
Control approach to distributed optimization
48th Annual Allerton Conference

Lu, J., C. Y. Tang, P. R. Regier, and T. D. Bow (2011)
Gossip Algorithms for Convex Consensus Optimization Over Networks
IEEE Transactions on Automatic Control

A. Nedić and A. Ozdaglar (2009)
Distributed subgradient methods for multi-agent optimization
IEEE Transactions on Automatic Control

B. Johansson (2008)
On Distributed Optimization in Networked Systems
Ph.D. Thesis, KTH

A. Nedić and A. Ozdaglar (2007)
On the Rate of Convergence of Distributed Subgradient Methods for Multi-agent Optimization
CDC

S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein (2010)
Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers
Foundations and Trends in Machine Learning

S. Muthukrishnan, B. Ghosh, and M. H. Schultz (1998)
First and Second Order Diffusive Methods for Rapid, Coarse, Distributed Load Balancing
Theory of Computing Systems

H. K. Khalil (2002)
Nonlinear Systems
Prentice Hall