Newton-Raphson Consensus: a distributed convex optimization scheme for networks with asynchronous and lossy communications

Nicoletta Bof Ruggero Carli Giuseppe Notarstefano
Luca Schenato **Damiano Varagnolo**

Linköping - Automatic control - ISY

November 10, 2016

Joint work with. . .

Nicoletta
Bof
Univ. of Padova

Ruggero
Carli
Univ. of Padova

Giuseppe
Notarstefano
Univ. of Lecce

Luca
Schenato
Univ. of Padova

Overview

part I: distributed optimization and its needs

part II: Newton-Raphson Consensus

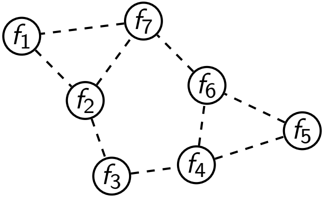
part III: from Newton-Raphson Consensus to Distributed Interior Point Methods

part IV: conclusions

Disclaimer

part I: distributed optimization and its needs

An introduction to distributed optimization



Assumption: neighbors cooperate to find the optimum of an additively separable cost:

$$f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x) \qquad x^* = \mathrm{argmin}_x f(x)$$

Example of a practical optimization problem

Thermal conditioning in datacenters

Example of a practical optimization problem

Thermal conditioning in datacenters

Distributed optimization playfields

Distributed optimization playfields

1. *example: datacenters*
   - topology = fixed and known
   - communications = reliable and synchronous

Distributed optimization playfields

1. *example: datacenters*
   - topology = fixed and known
   - communications = reliable and synchronous

2. *example: network of exploring robots*
   - topology = variable and unknown
   - communications = unreliable and asynchronous

Distributed optimization playfields

1. *example: datacenters*
    - topology = fixed and known
    - communications = reliable and synchronous

2. *example: network of exploring robots*
    - topology = variable and unknown
    - communications = unreliable and asynchronous

3. *NO* variable and unknown topology + reliable and synchronous communications or vice-versa

Personal intuition and opinion

different playfields

$$\Updownarrow$$

different distributed optimization algorithms

State of the art

3 main categories:

- primal decompositions methods
  (e.g. distributed subgradients)
- dual decompositions methods
  (e.g. alternating direction method of multipliers)
- heuristic methods
  (e.g. swarm optimization, genetic algorithms)

Example

Alternating Direction Method of Multipliers (ADMM)

ADMM [Bertsekas Tsitsiklis, 1997]

Primal:

$$\begin{aligned} \min \quad & f_1(x_1) + f_2(x_2) \\ \text{s.t.} \quad & A_1 x_1 + A_2 x_2 - b = 0 \end{aligned}$$

Augmented Lagrangian:

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

Algorithm

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

Drawbacks of ADMM

$$\min_{x} \sum_{i=1}^{N} f_i(x) \quad \Longrightarrow \quad \begin{array}{ll} \min\limits_{\{x_i\},\{z_{ij}\}} & \sum\limits_{i=1}^{N} f_i(x_i) \\ \text{s.t.} & x_i = z_{ij} \ \forall i,j \end{array}$$

$$\sum_{i=1}^{N} f_i(x_i) + \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}^T (x_i - z_{ij}) + \frac{\rho}{2} \sum_{(i,j)\in\mathcal{E}} \|x_i - z_{ij}\|^2$$

Drawbacks of ADMM

$$\min_x \sum_{i=1}^{N} f_i(x) \quad \Longrightarrow \quad \begin{array}{ll} \min\limits_{\{x_i\},\{z_{ij}\}} & \sum\limits_{i=1}^{N} f_i(x_i) \\ \text{s.t.} & x_i = z_{ij} \; \forall i, j \end{array}$$

$$\sum_{i=1}^{N} f_i(x_i) + \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}^T (x_i - z_{ij}) + \frac{\rho}{2} \sum_{(i,j)\in\mathcal{E}} \|x_i - z_{ij}\|^2$$

- hard to manage time-varying network topologies
- hard to manage packet losses

Drawbacks of ADMM

$$\min_{x} \sum_{i=1}^{N} f_i(x) \quad \Longrightarrow \quad \begin{array}{ll} \min\limits_{\{x_i\},\{z_{ij}\}} & \sum\limits_{i=1}^{N} f_i(x_i) \\ \text{s.t.} & x_i = z_{ij} \ \forall i, j \end{array}$$

$$\sum_{i=1}^{N} f_i(x_i) + \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}^T (x_i - z_{ij}) + \frac{\rho}{2} \sum_{(i,j)\in\mathcal{E}} \|x_i - z_{ij}\|^2$$

- hard to manage time-varying network topologies
- hard to manage packet losses

$\Rightarrow$ ADMM $\in$ specific playfield

An other example

Distributed Subgradient Methods (DSMs)

Distributed Subgradient Methods

[Nedic Ozdaglar, 2009]

$$
\begin{aligned}
x_i(k)^+ &= x_i(k) - \alpha_i(k) g_i(x_i(k)) \\
x_i(k+1) &= \sum_{j=1}^{N} a_{ij}(k) x_j^+(k)
\end{aligned}
$$

with

- $g_i(x_i(k)) :=$ local subgradient of local cost $f_i(\cdot)$ at $x_i(k)$
- $\alpha_i(k) :=$ local stepsize

Convergence properties [Nedic Ozdaglar, 2007]

E.g., for *bounded subgradients* and $\alpha_i(k) = \alpha$ then

$$
\lim_{k \to +\infty} \inf f(x_i(k)) = f^* + \delta \quad (\delta = 0 \text{ if } f_i\text{'s are smooth})
$$

Advantages of DSM

$$
\begin{aligned}
x_i(k)^+ &= x_i(k) - \alpha_i(k) g_i(x_i(k)) \\
x_i(k+1) &= \sum_{j=1}^{N} a_{ij}(k) x_j^+(k)
\end{aligned}
$$

- easy to manage time-varying network topologies
- easy to manage packet losses

Advantages of DSM

$$\begin{aligned} x_i(k)^+ &= x_i(k) - \alpha_i(k) g_i(x_i(k)) \\ x_i(k+1) &= \sum_{j=1}^{N} a_{ij}(k) x_j^+(k) \end{aligned}$$

- easy to manage time-varying network topologies
- easy to manage packet losses

problem: quite slow!

Advantages of DSM

$$
\begin{aligned}
x_i(k)^+ &= x_i(k) - \alpha_i(k) g_i(x_i(k)) \\
x_i(k+1) &= \sum_{j=1}^{N} a_{ij}(k) x_j^+(k)
\end{aligned}
$$

- easy to manage time-varying network topologies
- easy to manage packet losses

problem: quite slow! $\Rightarrow$ DSM $\in$ specific playfield

Wish

find a strategy that works well in every distributed playfield

Wish

find a strategy that works well in every distributed playfield

How?

Wish

find a strategy that works well in every distributed playfield

How?

(personal opinion)

find a strategy that works well in every centralized playfield

$\downarrow$

make it distributed

Wish

find a strategy that works well in every distributed playfield

How?

(personal opinion)

find a strategy that works well in every centralized playfield

$\downarrow$

make it distributed

$\Longrightarrow$ find a distributed Interior Point Method (IPM)

Wish

find a strategy that works well in every distributed playfield

How?

(personal opinion)

find a strategy that works well in every centralized playfield

$\downarrow$

make it distributed

$\Longrightarrow$ find a distributed Interior Point Method (IPM)

$\Longrightarrow$ find a distributed Newton-Raphson (NR)

part II: Newton-Raphson Consensus

Towards distributed NR schemes

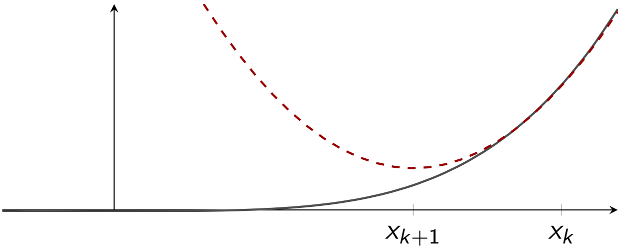
starting point: *simplest case*, i.e.,

- playfield $=$ static reliable networks
- unconstrained optimization problem

Centralized NR

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)} \tag{1}$$

- multidimensional version: $\Delta x = -\left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$
- interpretation: $x_{n+1} =$ minimizer of second order approximation



From centralized NR to distributed ones (1)

Newton update:

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

Then

$$f(x) = \sum_{i=1}^{N} f_i(x) \implies x^+ = x - \frac{\sum_{i=1}^{N} f_i'(x)}{\sum_{i=1}^{N} f_i''(x)}$$

From centralized NR to distributed ones (1)

Newton update:

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

Then

$$f(x) = \sum_{i=1}^{N} f_i(x) \quad \Longrightarrow \quad x^{+} = x - \frac{\sum_{i=1}^{N} f_i'(x)}{\sum_{i=1}^{N} f_i''(x)} = \frac{\sum_{i=1}^{N} \left(f_i''(x)x - f_i'(x) \right)}{\sum_{i=1}^{N} f_i''(x)}$$

From centralized NR to distributed ones (1)

Newton update:

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

Then

$$f(x) = \sum_{i=1}^{N} f_i(x) \quad \Longrightarrow \quad x^+ = x - \frac{\displaystyle\sum_{i=1}^{N} f_i'(x)}{\displaystyle\sum_{i=1}^{N} f_i''(x)} = \frac{\displaystyle\frac{1}{N}\sum_{i=1}^{N}\Big(f_i''(x)x - f_i'(x)\Big)}{\displaystyle\frac{1}{N}\sum_{i=1}^{N} f_i''(x)}$$

i.e., parallel of two average consensi

From centralized NR to distributed ones (1)

What does $$x^{+} = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(f_i''(x)x - f_i'(x)\right)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x)}$$ mean?

From centralized NR to distributed ones (1)

What does
$$x^+ = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(f_i''(x)x - f_i'(x)\right)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x)}$$
mean?

$\Longrightarrow$ approximate *each* $f_i(x)$ with a parabola:

$$\widehat{f}_i(x) = \frac{1}{2}a_i\left(x - b_i\right)^2 \qquad \begin{cases} a_i b_i & = f_i''(x)x - f_i'(x) \\ a_i & = f_i''(x) \end{cases}$$

From centralized NR to distributed ones (1)

What does $$x^+ = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(f_i''(x)x - f_i'(x)\right)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x)}$$ mean?

$\Longrightarrow$ approximate *each* $f_i(x)$ with a parabola:

$$\widehat{f}_i(x) = \frac{1}{2}a_i\left(x - b_i\right)^2 \qquad \begin{cases} a_i b_i & = f_i''(x)x - f_i'(x) \\ a_i & = f_i''(x) \end{cases}$$

Problem: how do we go distributed, i.e., $x_i^+ = x_i + \ldots$*?*

From centralized NR to distributed ones (2)

$$\text{What does} \qquad x_j^+ \frac{\dfrac{1}{N}\displaystyle\sum_{i=1}^{N}\Big(f_i''(x_i)x_i - f_i'(x_i)\Big)}{\dfrac{1}{N}\displaystyle\sum_{i=1}^{N} f_i''(x_i)} \qquad \text{mean?} \tag{2}$$

From centralized NR to distributed ones (2)

What does
$$x_j^{+}\frac{\frac{1}{N}\sum_{i=1}^{N}\Big(f_i''(x_i)x_i - f_i'(x_i)\Big)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x_i)} \qquad (2)$$
mean?

$\Longrightarrow$ approximate *each* $f_i(x_i)$ with a parabola:

$$\widehat{f}_i(x_i) = \frac{1}{2}a_i\left(x_i - b_i\right)^2 \qquad \begin{cases} a_i b_i & = f_i''(x_i)x_i - f_i'(x_i) \\ a_i & = f_i''(x_i) \end{cases}$$

From centralized NR to distributed ones (2)

What does $$x_j^+ \frac{\dfrac{1}{N}\displaystyle\sum_{i=1}^{N}\Big(f_i''(x_i)x_i - f_i'(x_i)\Big)}{\dfrac{1}{N}\displaystyle\sum_{i=1}^{N} f_i''(x_i)}$$ mean? (2)

$\Longrightarrow$ approximate *each* $f_i(x_i)$ with a parabola:

$$\widehat{f}_i(x_i) = \frac{1}{2} a_i \left(x_i - b_i\right)^2 \qquad \begin{cases} a_i b_i & = f_i''(x_i)x_i - f_i'(x_i) \\ a_i & = f_i''(x_i) \end{cases}$$

Problem: this is not the correct Newton step!

From centralized NR to distributed ones (2)

$$\text{What does} \qquad x_j^+ \frac{\dfrac{1}{N}\displaystyle\sum_{i=1}^{N}\Big(f_i''(x_i)x_i - f_i'(x_i)\Big)}{\dfrac{1}{N}\displaystyle\sum_{i=1}^{N} f_i''(x_i)} \qquad \text{mean?} \qquad (2)$$

$\Longrightarrow$ approximate *each* $f_i(x_i)$ with a parabola:

$$\widehat{f}_i(x_i) = \frac{1}{2}a_i\left(x_i - b_i\right)^2 \qquad \begin{cases} a_i b_i & = f_i''(x_i)x_i - f_i'(x_i) \\ a_i & = f_i''(x_i) \end{cases}$$

Problem: this is not the correct Newton step!

Intuition: x_i's close $\Longrightarrow$ (2) = good approximation

Towards the distributed algorithm

Summary of the problems:

- if $x_i \neq x_j$ then $$\frac{\frac{1}{N}\sum_{i=1}^{N}\Big(f_i''(x_i)x_i - f_i'(x_i)\Big)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x_i)}$$ is not the correct Newton step

- to compute the exact averages is time consuming

Towards the distributed algorithm

Summary of the problems:

- if $x_i \neq x_j$ then $$\frac{\frac{1}{N}\sum_{i=1}^{N}\Big(f_i''(x_i)x_i - f_i'(x_i)\Big)}{\frac{1}{N}\sum_{i=1}^{N} f_i''(x_i)}$$ is not the correct Newton step

- to compute the exact averages is time consuming

Solution:

alternate consensus steps on the x_i's and smoothed local guesses updates

The (synchronous) Newton-Raphson Consensus (NRC)

1. initialization:
   - $g_i(-1) = 0 \quad h_i(-1) = 0 \quad y_i(0) = 0 \quad z_i(0) = 0$
2. computation of auxiliary local variables:
   - $g_i(k) := f_i''(x_i(k))x_i(k) - f_i'(x_i(k))$
   - $h_i(k) := f_i''(x_i(k))$
3. average consensus on the Newton direction:
   (P doubly stochastic)
   - $\boldsymbol{y}(k+1) = P\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{k}) - \boldsymbol{g}(\boldsymbol{k}-\mathbf{1})$
   - $\boldsymbol{z}(k+1) = P\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{k}) - \boldsymbol{h}(\boldsymbol{k}-\mathbf{1})$
4. local update:
   - $x_i(k+1) = (\mathbf{1} - \varepsilon)\boldsymbol{x_i}(\boldsymbol{k}) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)}$

The (synchronous) NRC: important features

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

The (synchronous) NRC: important features

Why $x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)}$?

Why $P\boldsymbol{y}(k) + \boldsymbol{g}(k) - \boldsymbol{g}(k-1)$ instead of $P\boldsymbol{y}(k) + \boldsymbol{g}(k)$?

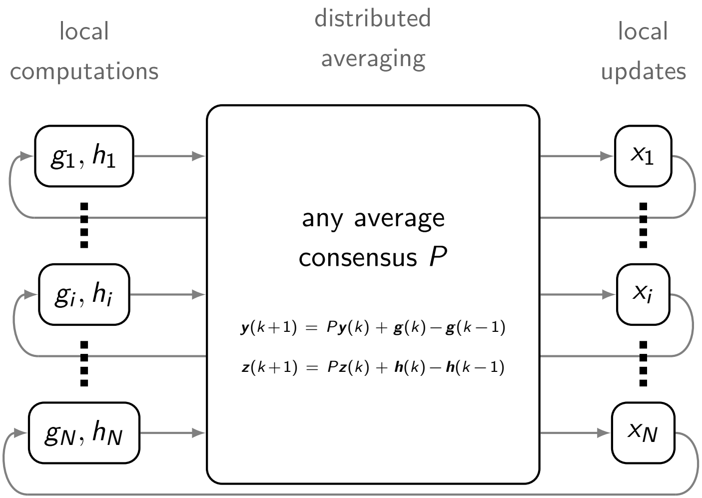
The (synchronous) NRC: important features

Why $x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)}$?

Why $P\boldsymbol{y}(k) + \boldsymbol{g}(k) - \boldsymbol{g}(k-1)$ instead of $P\boldsymbol{y}(k) + \boldsymbol{g}(k)$?

Why $g_i(-1) = 0 \quad h_i(-1) = 0 \quad y_i(0) = 0 \quad z_i(0) = 0$?

Block schematic representation



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

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

Convergence proof (singular perturbation theory)

$$
\left\{
\begin{array}{ll}
\boldsymbol{x}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \boldsymbol{0} & \text{initialization} \\
\hline
\boldsymbol{y}(k+1) = P\big(\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{x}(k)) - \boldsymbol{g}(\boldsymbol{x}(k-1))\big) & \text{fast dynamics} \\
\boldsymbol{z}(k+1) = P\big(\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{x}(k)) - \boldsymbol{h}(\boldsymbol{x}(k-1))\big) & \\
\hline
x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)} & \text{slow dynamics}
\end{array}
\right.
$$

Convergence proof (singular perturbation theory)

$$\begin{cases} \boldsymbol{x}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \boldsymbol{0} & \text{initialization} \\ \hline \boldsymbol{y}(k+1) = P(\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{x}(k)) - \boldsymbol{g}(\boldsymbol{x}(k-1))) & \text{fast dynamics} \\ \boldsymbol{z}(k+1) = P(\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{x}(k)) - \boldsymbol{h}(\boldsymbol{x}(k-1))) & \\ \hline x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)} & \text{slow dynamics} \end{cases}$$

Fast dynamics

- $\varepsilon \approx 0 \implies \boldsymbol{x}(k+1) \approx \boldsymbol{x}(k) = \boldsymbol{x}$ (constant)

Convergence proof (singular perturbation theory)

$$\begin{cases} \boldsymbol{x}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \boldsymbol{0} & \text{initialization} \\ \hline \boldsymbol{y}(k+1) = P\big(\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{x}(k)) - \boldsymbol{g}(\boldsymbol{x}(k-1))\big) & \text{fast dynamics} \\ \boldsymbol{z}(k+1) = P\big(\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{x}(k)) - \boldsymbol{h}(\boldsymbol{x}(k-1))\big) & \\ \hline x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)} & \text{slow dynamics} \end{cases}$$

Fast dynamics

- $\varepsilon \approx 0 \implies \boldsymbol{x}(k+1) \approx \boldsymbol{x}(k) = \boldsymbol{x}$ (constant)
- $\implies y_i(k+1) \to \frac{1}{N}\sum_{i=1}^{N} g_i(x_i) = \frac{1}{N}\sum_{i=1}^{N} f_i''(x_i)x_i - f_i'(x) = \bar{g}(\boldsymbol{x})$
- $\implies z_i(k+1) \to \frac{1}{N}\sum_{i=1}^{N} h_i(x_i) = \frac{1}{N}\sum_{i=1}^{N} f_i''(x_i) = \bar{h}(\boldsymbol{x})$

Convergence proof

$$
\begin{cases}
\boldsymbol{x}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \boldsymbol{0} & \text{initialization} \\
\hline
\boldsymbol{y}(k+1) = P(\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{x}(k)) - \boldsymbol{g}(\boldsymbol{x}(k-1))) & \text{fast dynamics} \\
\boldsymbol{z}(k+1) = P(\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{x}(k)) - \boldsymbol{h}(\boldsymbol{x}(k-1))) & \\
\hline
x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)} & \text{slow dynamics}
\end{cases}
$$

Slow dynamics

- $y_i = \bar{g}(\boldsymbol{x}) \quad z_i = \bar{h}(\boldsymbol{x})$

Convergence proof

$$\begin{cases} \boldsymbol{x}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \boldsymbol{0} & \text{initialization} \\ \hline \boldsymbol{y}(k+1) = P(\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{x}(k)) - \boldsymbol{g}(\boldsymbol{x}(k-1))) & \text{fast dynamics} \\ \boldsymbol{z}(k+1) = P(\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{x}(k)) - \boldsymbol{h}(\boldsymbol{x}(k-1))) & \\ \hline x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)} & \text{slow dynamics} \end{cases}$$

Slow dynamics

- $y_i = \bar{g}(\boldsymbol{x}) \quad z_i = \bar{h}(\boldsymbol{x})$
- $\implies x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{\bar{g}(\boldsymbol{x}(k))}{\bar{h}(\boldsymbol{x}(k))}$

Convergence proof

$$
\begin{cases}
\boldsymbol{x}(0) = \boldsymbol{y}(0) = \boldsymbol{z}(0) = \boldsymbol{g}(\boldsymbol{x}(-1)) = \boldsymbol{h}(\boldsymbol{x}(-1)) = \boldsymbol{0} & \text{initialization} \\
\hline
\boldsymbol{y}(k+1) = P(\boldsymbol{y}(k) + \boldsymbol{g}(\boldsymbol{x}(k)) - \boldsymbol{g}(\boldsymbol{x}(k-1))) & \text{fast dynamics} \\
\boldsymbol{z}(k+1) = P(\boldsymbol{z}(k) + \boldsymbol{h}(\boldsymbol{x}(k)) - \boldsymbol{h}(\boldsymbol{x}(k-1))) & \\
\hline
x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{y_i(k+1)}{z_i(k+1)} & \text{slow dynamics}
\end{cases}
$$

Slow dynamics

- $y_i = \bar{g}(\boldsymbol{x}) \quad z_i = \bar{h}(\boldsymbol{x})$
- $\implies x_i(k+1) = (1-\varepsilon)x_i(k) + \varepsilon \dfrac{\bar{g}(\boldsymbol{x}(k))}{\bar{h}(\boldsymbol{x}(k))}$
- same forcing term $\implies \lim_{k\to\infty} x_i(k) - x_j(k) = 0$

Convergence proof

Slow dynamics

- same forcing term $\implies$ eventually $x_i = x_j = \bar{x}$

Convergence proof

Slow dynamics

- same forcing term $\implies$ eventually $x_i = x_j = \bar{x}$
- $\implies$

$$\begin{aligned}
\bar{x}^+ &= (1-\varepsilon)\bar{x} + \varepsilon \frac{\bar{g}(\bar{x}\mathbf{1})}{\bar{h}(\bar{x}\mathbf{1})} \\
&= (1-\varepsilon)\bar{x} + \varepsilon \frac{\frac{1}{N}\sum_{i=1}^{N} f_i''(\bar{x})\bar{x} - f_i'(\bar{x})}{\frac{1}{N}\sum_{i=1}^{N} f_i''(\bar{x})} \\
&= (1-\varepsilon)\bar{x} + \varepsilon \left(\bar{x} - \frac{\frac{1}{N}\sum_{i=1}^{N} f_i'(\bar{x})}{\frac{1}{N}\sum_{i=1}^{N} f_i''(\bar{x})} \right) \\
&= \bar{x} - \varepsilon \frac{f'(\bar{x})}{f''(\bar{x})}
\end{aligned}$$

Convergence proof

Slow dynamics

- same forcing term $\implies$ eventually $x_i = x_j = \bar{x}$
- $\implies$

$$\begin{aligned}
\bar{x}^+ &= (1-\varepsilon)\bar{x} + \varepsilon \frac{\bar{g}(\bar{x}\mathbf{1})}{\bar{h}(\bar{x}\mathbf{1})} \\
&= (1-\varepsilon)\bar{x} + \varepsilon \frac{\frac{1}{N}\sum_{i=1}^N f_i''(\bar{x})\bar{x} - f_i'(\bar{x})}{\frac{1}{N}\sum_{i=1}^N f_i''(\bar{x})} \\
&= (1-\varepsilon)\bar{x} + \varepsilon \left(\bar{x} - \frac{\frac{1}{N}\sum_{i=1}^N f_i'(\bar{x})}{\frac{1}{N}\sum_{i=1}^N f_i''(\bar{x})} \right) \\
&= \bar{x} - \varepsilon \frac{f'(\bar{x})}{f''(\bar{x})}
\end{aligned}$$

Centralized Newton-Raphson!!

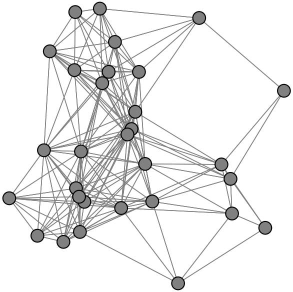
Formal results

- f_i quadratic $\implies$ global exponential convergence with rate $\mathrm{sr}(P)$ for $\varepsilon = 1$ for any connected graph
- complete graph $\implies$ centralized Newton-Raphson
- $f_i \in \mathcal{C}^3$ and convex $\implies$ local exponential stability for $0 < \varepsilon < \varepsilon_c$
- global boundedness of $\dfrac{f' \cdot f'''}{(f'')^2}$ and f'' $\implies$ global exponential stability for $0 < \varepsilon < \varepsilon_c$

Simulations: SVM Classification

Spam-nonspam classification

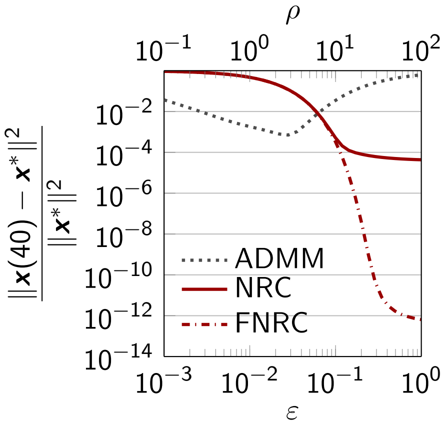
- $\boldsymbol{x} \in \mathrm{R}^4$ (frequency of specific words)
- $y \in \{0, 1\}$ (spam, non spam)
- network:

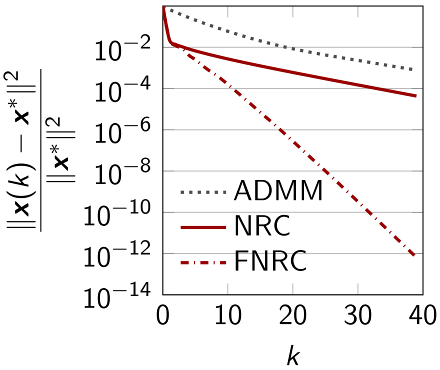


- cost: $f_i(x) := \sum_j \log\left(1 + \exp\left(-y_j\left(\chi_j^T \boldsymbol{x} + x_0\right)\right)\right) + \gamma \|\boldsymbol{x}\|_2^2$

Simulations: SVM Classification

Spam-nonspam classification

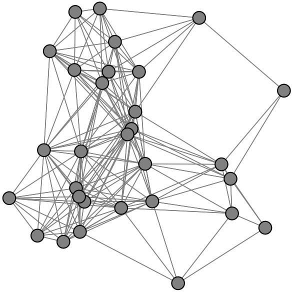




Simulations: regression

Housing regression

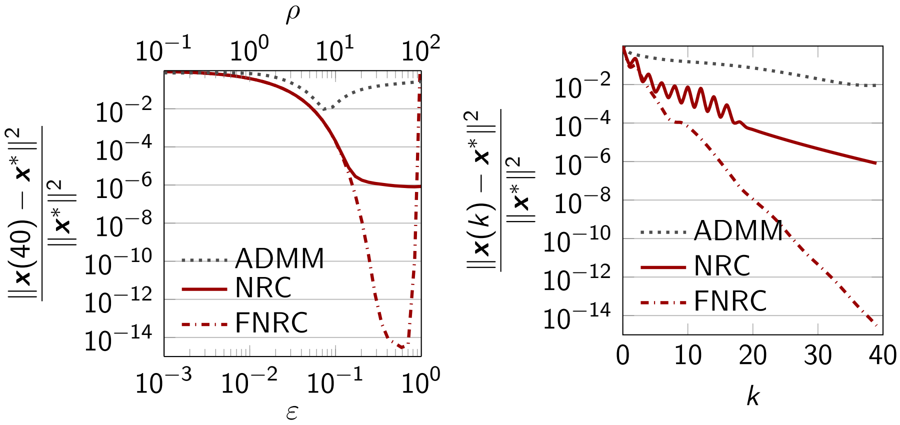
- $\boldsymbol{x} \in \mathrm{R}^4$ (size, distance from downtown, etc.)
- $y \in \mathbb{R}$ (house price)
- network:



- cost: $f_i(x) := \sum_j \frac{\left(y_j - \chi_j^T \boldsymbol{x} - x_0\right)^2}{\left|y_j - \chi_j^T \boldsymbol{x} - x_0\right| + \beta} + \gamma \|\boldsymbol{x}\|_2^2$

Simulations: regression

Housing regression



problem: can we play in the other playfield?

i.e., with asynchronous broadcast communications without channel feedback?

Ratio consensus

asynchronous communications with perfect channel feedback [Bénézit et al. 2010]

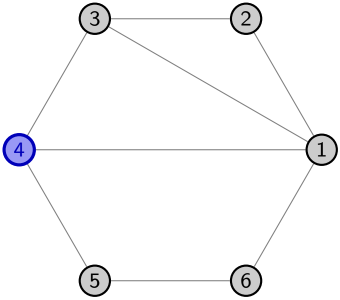
$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

Ratio consensus

asynchronous communications with perfect channel feedback [Bénézit et al. 2010]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

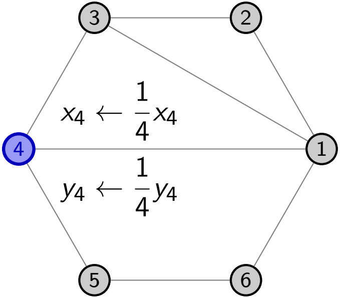


Ratio consensus

asynchronous communications with perfect channel feedback [Bénézit et al. 2010]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

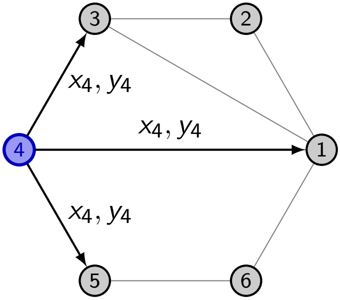


Ratio consensus

asynchronous communications with perfect channel feedback [Bénézit et al. 2010]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

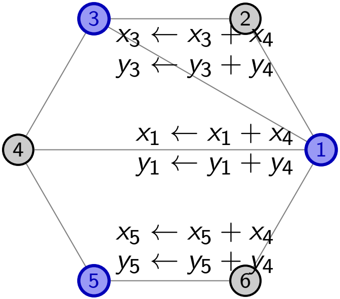


Ratio consensus

asynchronous communications with perfect channel feedback [Bénézit et al. 2010]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

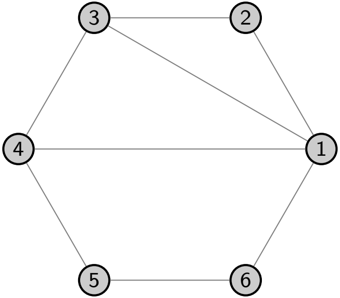
$$P(k) = \begin{bmatrix} 1 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



Ratio consensus

asynchronous communications with perfect channel feedback [Bénézit et al. 2010]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$



$$P(k) = \begin{bmatrix} 1 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

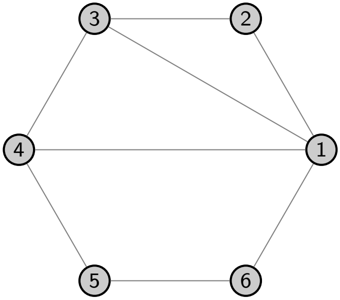
$$\begin{cases} x_i(k) \to \beta_i(k) \sum_j x_i(0) \\ y_i(k) \to \beta_i(k) \sum_j y_i(0) \end{cases}$$

Ratio consensus

asynchronous communications with perfect channel feedback [Bénézit et al. 2010]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



$$\begin{cases} x_i(k) \to \beta_i(k) \sum_j x_i(0) \\ \\ y_i(k) \to \beta_i(k) \sum_j y_i(0) \end{cases} \implies \quad z_i(k) := \frac{x_i(k)}{y_i(k)} \to \frac{\sum_i x_i(0)}{\sum_i y_i(0)} = \frac{1}{N} \sum_i \theta_i$$

Robust ratio consensus

asynch. comm. ***without*** perfect channel feedback [Dominguez-Garcia et al. 2011]

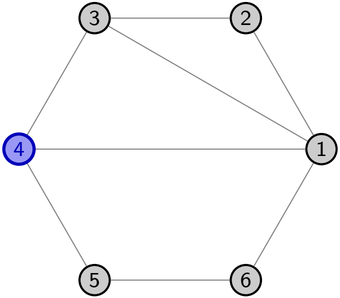
$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

Robust ratio consensus

asynch. comm. ***without*** perfect channel feedback [Dominguez-Garcia et al. 2011]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

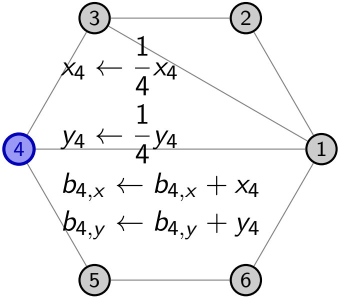


Robust ratio consensus

asynch. comm. ***without*** perfect channel feedback [Dominguez-Garcia et al. 2011]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



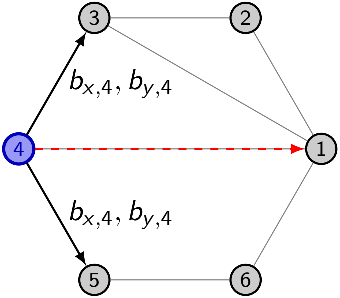
- $b_{i,x}$: total cumulative mass of x_i
- $\beta_{i,x}^{(j)}$: j's local estimate of $b_{i,x}$

Robust ratio consensus

asynch. comm. ***without*** perfect channel feedback [Dominguez-Garcia et al. 2011]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



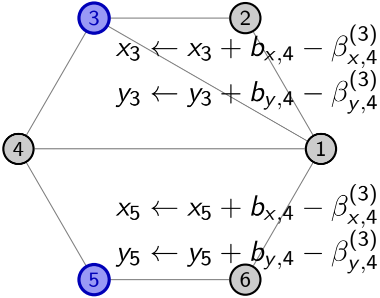
- $b_{i,x}$: total cumulative mass of x_i
- $\beta^{(j)}_{i,x}$: j's local estimate of $b_{i,x}$

Robust ratio consensus

asynch. comm. ***without*** perfect channel feedback [Dominguez-Garcia et al. 2011]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



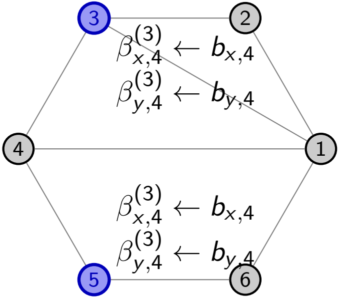
- $b_{i,x}$: total cumulative mass of x_i
- $\beta^{(j)}_{i,x}$: j's local estimate of $b_{i,x}$

Robust ratio consensus

asynch. comm. ***without*** perfect channel feedback [Dominguez-Garcia et al. 2011]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



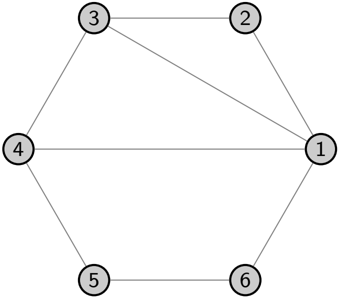
- $b_{i,x}$: total cumulative mass of x_i
- $\beta^{(j)}_{i,x}$: j's local estimate of $b_{i,x}$

Robust ratio consensus

asynch. comm. ***without*** perfect channel feedback [Dominguez-Garcia et al. 2011]

$$\begin{cases} \boldsymbol{x}(k+1) = P(k)\boldsymbol{x}(k) \\ x_i(0) = \theta_i \\ \boldsymbol{y}(k+1) = P(k)\boldsymbol{y}(k) \\ y_i(0) = 1 \end{cases}$$

$$P(k) = \begin{bmatrix} 1 & 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 0 & 0 \\ 0 & 0 & 0 & 1/4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



$$z_i(k) = \frac{x_i(k)}{y_i(k)} \to \frac{1}{N}\sum_j \theta_i$$

- $b_{i,x}$: total cumulative mass of x_i
- $\beta_{i,x}^{(j)}$: j's local estimate of $b_{i,x}$

Robust Asynchronous NRC (RA-NRC)

Initialization

$$
\begin{cases}
x_i & \leftarrow x^o \\
y_i = g_i^{\text{old}} = g_i & \leftarrow f_i''(x^o)\, x^o - f_i'(x^o) \\
z_i = h_i^{\text{old}} = h_i & \leftarrow f_i''(x^o)
\end{cases}
$$

Robust Asynchronous NRC (RA-NRC)

Transmission

$$
\begin{aligned}
y_i &\leftarrow \frac{1}{|\mathcal{N}_i^{\text{out}}|+1}\left[y_i + g_i - g_i^{\text{old}}\right] \\
z_i &\leftarrow \frac{1}{|\mathcal{N}_i^{\text{out}}|+1}\left[z_i + h_i - h_i^{\text{old}}\right] \\
x_i &\leftarrow (1-\varepsilon)x_i + \varepsilon \frac{y_i}{[z_i]_c}
\end{aligned}
$$

$$
\begin{aligned}
b_{i,y} &\leftarrow b_{i,y} + y_i \\
b_{i,z} &\leftarrow b_{i,z} + z_i \\
g_i^{\text{old}} &\leftarrow g_i \\
h_i^{\text{old}} &\leftarrow h_i \\
g_i &\leftarrow f_i''(x_i)x_i - f_i'(x_i) \\
h_i &\leftarrow f_i''(x_i)
\end{aligned}
$$

Robust Asynchronous NRC (RA-NRC)

Reception

$$\begin{aligned} y_j &\leftarrow y_j + b_{i,y} - \beta_{i,y}^{(j)} + g_j - g_j^{\text{old}} \\ z_j &\leftarrow z_j + b_{i,z} - \beta_{i,z}^{(j)} + h_j - h_j^{\text{old}} \\ x_j &\leftarrow (1-\varepsilon)x_j + \varepsilon \frac{y_j}{[z_j]_c} \end{aligned}$$

$$\begin{aligned} \beta_{i,y}^{(j)} &\leftarrow b_{i,y} \\ \beta_{i,z}^{(j)} &\leftarrow b_{i,z} \\ g_j^{\text{old}} &\leftarrow g_j \\ h_j^{\text{old}} &\leftarrow h_j \\ g_j &\leftarrow f_i''(x_j)x_i - f_i'(x_j) \\ h_j &\leftarrow f_i''(x_j) \end{aligned}$$

Convergence properties of RA-NRC

Assumptions

- $f_i \in \mathcal{C}^2, \quad f_i''(x) > c$
- fixed, strongly connected and directed network
- communications are persistent
  (i.e., at least 1 communication in every $[t, t+\tau]$*)*
- bounded packet losses
  (i.e., number of consecutive failures is limited)

Proposition

$\exists\ B_\delta(x^*)$ and $\varepsilon_c \in \mathbb{R}_+$ s.t. if $x^o \in B_\delta$ and $0 < \varepsilon < \varepsilon_c$ then

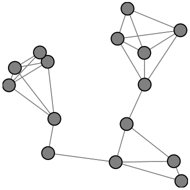
$$|x_i(k) - x^*| \leq c\lambda^k \qquad \forall i$$

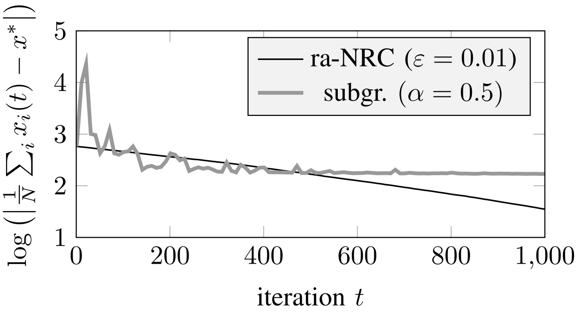
for opportune $c \in \mathbb{R}_+$ and $\lambda < 1$

Numerical experiments: RA-NRC vs. DSM

algorithms tuned with their best parameters and packet loss probability $p = 0.1$

$$f_i(x) = \frac{(y_i - \langle \boldsymbol{\chi}_i, \widetilde{\boldsymbol{x}} \rangle)^2}{|y_i - \langle \boldsymbol{\chi}_i, \widetilde{\boldsymbol{x}} \rangle| + \beta} + \gamma \, \|\boldsymbol{x}\|_2^2$$





part III: the route from Newton-Raphson Consensus to Distributed Interior Point Methods

Missing features

- handling constraints

Missing features

- handling constraints
- distributed stepsize selection

Missing features

- handling constraints
- distributed stepsize selection
- partition-based optimization

Missing features

- handling constraints
- distributed stepsize selection
- partition-based optimization
- distributed termination criteria

Missing features

- handling constraints
- distributed stepsize selection
- partition-based optimization
- distributed termination criteria
- quasi-Newton methods

part IV: conclusions

Take-home messages

- NRC ladders on average consensus for distributedly computing Newton directions
- NRC is a good candidate for developing distributed IPMs; nonetheless it still lacks of some development

if you want to collaborate on this area we are super keen to do so

Newton-Raphson Consensus: a distributed convex optimization scheme for networks with asynchronous and lossy communications

Nicoletta Bof   Ruggero Carli   Giuseppe Notarstefano
Luca Schenato   **Damiano Varagnolo**

Linköping - Automatic control - ISY

November 10, 2016

damiano.varagnolo@ltu.se