Distributed optimization and Nash equilibrium seeking for uncertain multi-agent systems

Kostas Margellos

University of Oxford

August 16, 2019

Credit

- Alessandro Falsone

- Simone Garatti

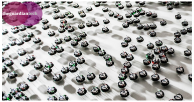
- Maria Prandini

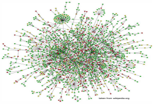
- Filiberto Fele

Motivation

- Networks (Power, Social, etc.)







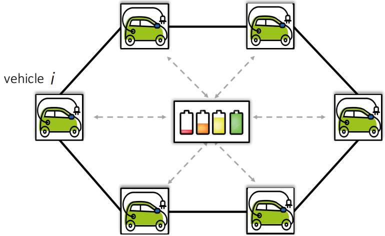
  - Large scale infrastructures
  - Multi-agent – Multiple interacting entities/users
  - Heterogeneous – Different physical or technological constraints per agent; different objectives per agent

- Challenge: Optimizing the performance of a network ...
  - Computation: Problem size too big!
  - Communication: Not all communication links at place; link failures
  - Information privacy: Agents may not want to share information with everyone (e.g. facebook)

Why go distributed?

1. Scalable methodology
   - Communication: Only between neighbors
   - Computation: Only local; in parallel for all agents
2. Information privacy
   - Agents do not reveal information about their preferences (encoded by objective and constraint functions) to each other
3. Resilience to communication failures
4. Numerous applications
   - Wireless networks
   - Optimal power flow
   - Electric vehicle charging control
   - Energy management in building networks

Example: Electric vehicle charging control problem



Cost coupled problems

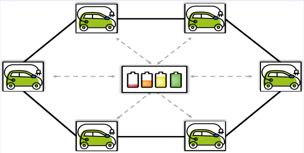
$$\text{minimize} \quad \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i X_i$$

Distributed proximal minimization

Step 1: Local problem of agent i

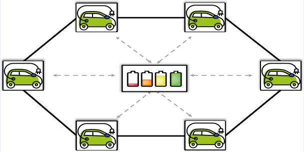


$$\left.\begin{array}{l} \text{minimize } f_i(x_i) + g_i(x_i, z_i) \\ \text{subject to} \\ x_i \in X_i \end{array}\right\} \Rightarrow x_i^*(z_i)$$

- x_i: "copy" of x maintained by agent i
- X_i: local constraint set of agent i
- z_i: information vector – constructed based on the info of agent's i neighbors
- Objective function
  $f_i(x_i)$: local cost/utility of agent i
  $g_i(x_i, z_i)$: Proxy term, penalizing disagreement with other agents

Distributed proximal minimization

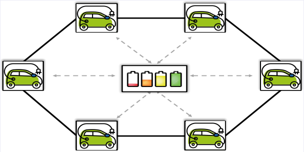
Step 1: Local problem of agent i



$$\left.\begin{array}{l}\text{minimize } f_i(x_i) + g_i(x_i, z_i)\\ \text{subject to}\\ x_i \in X_i\end{array}\right\} \Rightarrow x_i^*(z_i)$$

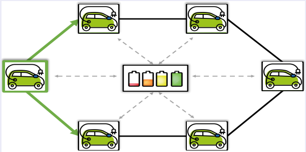
Distributed proximal minimization

Step 1: Local problem of agent i

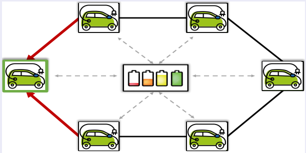


$$\left.\begin{array}{l} \text{minimize } f_i(x_i) + g_i(x_i, z_i) \\ \text{subject to} \\ x_i \in X_i \end{array}\right\} \Rightarrow x_i^*(z_i)$$

Step 2a: Broadcast $x_i^*(z_i)$ to neighbors

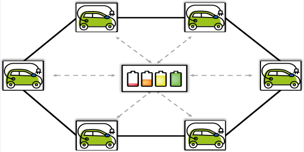


Step 2b: Receive neighbors' solutions



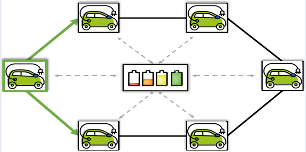
Distributed proximal minimization

Step 1: Local problem of agent i

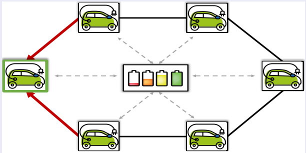


$$\left.\begin{array}{l}\text{minimize } f_i(x_i) + g_i(x_i, z_i)\\ \text{subject to}\\ x_i \in X_i\end{array}\right\} \Rightarrow x_i^*(z_i)$$

Step 2a: Broadcast $x_i^*(z_i)$ to neighbors



Step 2b: Receive neighbors' solutions

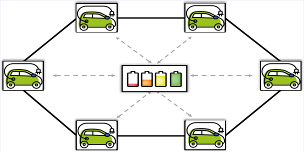


Step 3: Update z_i on the basis of information received

Go to Step 1

Distributed proximal minimization

Local problem of agent i

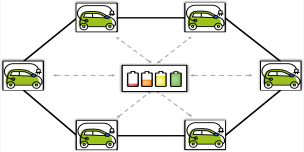


$$\left.\begin{array}{l}\text{minimize } f_i(x_i) + g_i(x_i, z_i)\\ \text{subject to}\\ x_i \in X_i\end{array}\right\} \Rightarrow x_i^*(z_i)$$

- Specify
  - Information vector z_i
  - Proxy term term $g_i(x_i, z_i)$
- Note that these terms change across algorithm iterations

Distributed proximal minimization

Local problem of agent i at iteration $k+1$



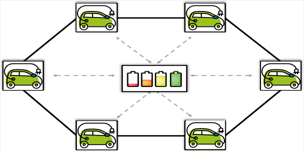
$$z_i(k) = \sum_j a_j^i(k) x_j(k)$$

$$x_i(k+1) = \arg\min_{x_i \in X_i} \; f_i(x_i) + \frac{1}{2c(k)} \|x_i - z_i(k)\|^2$$

- Information vector
  - $z_i(k) = \sum_j a_j^i(k) x_j(k)$
  - $a_j^i(k)$: how agent i weights info of agent j
- Proxy term
  - $\frac{1}{2c(k)} \|x_i - z_i(k)\|^2$: deviation from (weighted) average
  - $c(k)$: trade-off between optimality and agents' disagreement

Distributed proximal minimization

Local problem of agent i at iteration $k+1$



$$z_i(k) = \sum_j a_j^i(k) x_j(k)$$

$$x_i(k+1) = \arg\min_{x_i \in X_i} \; f_i(x_i) + \frac{1}{2c(k)} \|x_i - z_i(k)\|^2$$

- Does this algorithm converge?
- If yes, does it provide the same solution with the centralized problem (had we been able to solve it)?

Recall ...

ADMM algorithm

1. Primal update for x **information from central authority**

$$x(k+1) = \frac{1}{m}\sum_i x_i(k) - \frac{1}{mc}\sum_i \lambda_i(k)$$

2. Primal update for x_i **in parallel for all agents**

$$x_i(k+1) = \arg\min_{x_i \in X_i} f_i(x_i) - \lambda_i(k)^\top x_i + \frac{c}{2}\|x(k+1) - x_i\|^2$$

3. Dual update

$$\lambda_i(k+1) = \lambda_i(k) + c(x(k+1) - x_i(k+1))$$

Algorithm analysis: Assumptions

1. Convexity and compactness
   - $f_i(\cdot)$: convex for all i
   - X_i: compact, convex, non-empty interior for all i

   $\Rightarrow$ There exists a Slater point, i.e. $\exists$ Ball$(\bar{x}, \rho) \subset \bigcap_i X_i$

   $\Rightarrow$ $f_i(\cdot)$: Lipschitz continuous on X_i

Algorithm analysis: Assumptions

1. Convexity and compactness
   - $f_i(\cdot)$: convex for all i
   - X_i: compact, convex, non-empty interior for all i

     $\Rightarrow$ There exists a Slater point, i.e. $\exists$ Ball$(\bar{x}, \rho) \subset \bigcap_i X_i$

     $\Rightarrow$ $f_i(\cdot)$: Lipschitz continuous on X_i
2. Information mix
   - Weights $a^i_j(k)$: non-zero lower bound if link between $i - j$ present

     $\Rightarrow$ Info mixing at a non-diminishing rate
   - Weights $a^i_j(k)$: form a doubly stochastic matrix

     $\Rightarrow$ Agents influence each other equally in the long run

$$\sum_j a^i_j(k) = 1, \ \forall i$$

$$\sum_i a^i_j(k) = 1, \ \forall j$$

Algorithm analysis: Assumptions

3. Choice of the proxy term
   - $\{c(k)\}_k$: non-increasing
   - Should not decrease too fast

$$\sum_k c(k) = \infty \qquad \text{[to approach set of optimizers]}$$

$$\sum_k c(k)^2 < \infty \qquad \text{[to achieve convergence]}$$

   - E.g., harmonic series

$$c(k) = \frac{\alpha}{k+1}$$

Algorithm analysis: Assumptions

3. Network connectivity – All information flows (eventually)

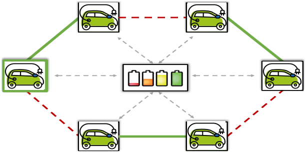
Connectivity

Let (V, E_k) be a directed graph, where V: nodes/agents, and $E_k = \{(j,i) : \; a_j^i(k) > 0\}$: edges Let

$$E_\infty = \{(j,i) : \; (j,i) \in E_k \text{ for infinitely many } k\}.$$

(V, E_∞) is strongly connected and (kind of) periodic, i.e., for any two nodes there exists a path of directed edges that connects.

- Any pair of agents communicates infinitely often,
- Intercommunication time is bounded



Algorithm analysis: Assumptions

3. Network connectivity – All information flows (eventually)

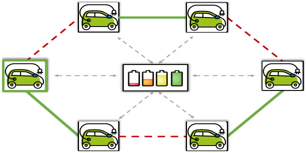
Connectivity

Let (V, E_k) be a directed graph, where V: nodes/agents, and $E_k = \{(j,i) : \ a_j^i(k) > 0\}$: edges Let

$$E_\infty = \{(j,i) : \ (j,i) \in E_k \text{ for infinitely many } k\}.$$

(V, E_∞) is strongly connected and (kind of) periodic, i.e., for any two nodes there exists a path of directed edges that connects.

- Any pair of agents communicates infinitely often,
- Intercommunication time is bounded



Algorithm analysis: Assumptions

3. Network connectivity – All information flows (eventually)

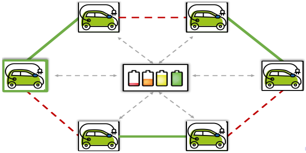
Connectivity

Let (V, E_k) be a directed graph, where V: nodes/agents, and $E_k = \{(j,i): \ a_j^i(k) > 0\}$: edges Let

$$E_\infty = \{(j,i): \ (j,i) \in E_k \text{ for infinitely many } k\}.$$

(V, E_∞) is strongly connected and (kind of) periodic, i.e., for any two nodes there exists a path of directed edges that connects.

- Any pair of agents communicates infinitely often,
- Intercommunication time is bounded



Algorithm analysis: Assumptions

3. Network connectivity – All information flows (eventually)

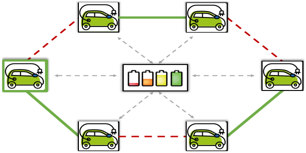
Connectivity

Let (V, E_k) be a directed graph, where V: nodes/agents, and $E_k = \{(j,i) : \ a_j^i(k) > 0\}$: edges Let

$$E_\infty = \{(j,i) : \ (j,i) \in E_k \text{ for infinitely many } k\}.$$

(V, E_∞) is strongly connected and (kind of) periodic, i.e., for any two nodes there exists a path of directed edges that connects.

- Any pair of agents communicates infinitely often,
- Intercommunication time is bounded



Algorithm analysis: Assumptions

3. Network connectivity – All information flows (eventually)

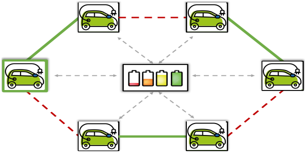
Connectivity

Let (V, E_k) be a directed graph, where V: nodes/agents, and $E_k = \{(j,i): \ a_j^i(k) > 0\}$: edges Let

$$E_\infty = \{(j,i): \ (j,i) \in E_k \text{ for infinitely many } k\}.$$

(V, E_∞) is strongly connected and (kind of) periodic, i.e., for any two nodes there exists a path of directed edges that connects.

- Any pair of agents communicates infinitely often,
- Intercommunication time is bounded



Convergence & optimality result

Theorem [MFGP, TAC 2018]

Under the structural + network assumptions, the proposed proximal algorithm converges to some minimizer x^* of the centralized problem, i.e.,

$$\lim_{k \to \infty} \|x_i(k) - x^*\| = 0, \text{ for all } i$$

- Asymptotic agreement and optimality
- Rate no faster than $c(k)$ – "slow enough" to trade agreement and optimality
- Average gradient tracking could speed things up
  See [Romao et al, CDC 2019]

Connection with gradient algorithms

- Proximal algorithms vs. gradient/subgradient methods

Constrained Consensus and Optimization in Multi-Agent Networks

Angelia Nedić, *Member, IEEE*, Asuman Ozdaglar, *Member, IEEE*, and Pablo A. Parrilo, *Senior Member, IEEE*

Abstract
by multipl
over a netw
general in
multiple ag
where the
objective f
where the
convex sets

Distributed Random Projection Algorithm for Convex Optimization

Soomin Lee and Angelia Nedić

***Abstract*—Random proj**
strained optimization whe
advance or the projection
is computationally prohibi
***random projection* algorith**
problems that can be used
time-varying network, wh
function and its own cons
of all agents converge to th
surely. Experiments on
demonstrate good perform

Math. Program., Ser. B (2011) 129:163–195
DOI 10.1007/s10107-011-0472-0

FULL LENGTH PAPER

Incremental proximal methods for large scale convex optimization

Dimitri P. Bertsekas

Connection with gradient algorithms

- Proximal algorithms

$$x_i(k+1) = \arg\min_{x_i \in X_i} \; f_i(x_i) + \frac{1}{2c(k)} \|x_i - z_i(k)\|^2$$

$$\text{but also } x_i(k+1) = P_{X_i}\big[z_i(k) - c(k)\nabla f_i(x_i(k+1))\big]$$

- Gradient algorithms

$$x_i(k+1) = P_{X_i}\big[z_i(k) - c(k)\nabla f_i(z_i(k))\big]$$

- Proximal algorithms allow for
  - No gradient/subgradient calculation – user can feed problem data in any solver
  - Heterogeneous constraint sets
  - No differentiability assumptions
  - ... but if we solve it using subgradient methods $\Rightarrow$ nested iteration

Constraint coupled problems

$$
\begin{aligned}
\text{minimize} \quad & \sum_i f_i(x_i) \\
\text{subject to} \quad & \\
& x_i \in X_i, \ \ \forall i \\
& \sum_i g_i(x_i) \leq 0
\end{aligned}
$$

Algorithm

Primal-dual scheme: Proximal for the dual

1. Information mixing

$$\ell_i(k) = \sum_j a_j^i(k)\lambda_j(k)$$

2. Local optimization (primal-dual) for each agent

$$x_i(k+1) \in \arg\min_{x_i \in X_i} f_i(x_i) + \ell_i(k)^\top g_i(x_i)$$

$$\lambda_i(k+1) = \arg\max_{\lambda_i \geq 0} g_i(x_i(k+1))^\top \lambda_i - \frac{1}{2c(k)} \|\lambda_i - \ell_i(k)\|^2$$

3. Primal recovery

$$\hat{x}_i(k+1) = \hat{x}_i(k) + \frac{c(k)}{\sum_{r=0}^{k} c(r)} (x_i(k+1) - \hat{x}_i(k))$$

Primal recovery & optimality

- The primal iterates $x_i(k)$ do not converge in general to the optimum $\Rightarrow$ an auxiliary sequence $\hat{x}_i(k)$ is needed
- The primal recovery step is equivalent to a running average

$$\hat{x}_i(k+1) = \frac{\sum_{r=0}^{k} c(r) x_i(r+1)}{\sum_{r=0}^{k} c(r)}$$

Dual optimality

There exists dual optimizer λ^*, such that for any agent i,

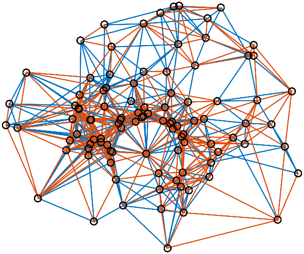
$$\lim_{k \to \infty} \|\lambda_i(k) - \lambda^*\| = 0$$

Primal optimality

If X^* is the set of minimizers, all limit points of $\{\hat{x}_i(k)\}$ are optimal, i.e.

$$\lim_{k \to \infty} \operatorname{dist}\big(\hat{x}_i(k), X^*\big) = 0$$

Example: Electric vehicle charging (simplified version)



Alternate between blue – red links, 100 vehicles, 48 coupling constraints

$$\text{minimize} \quad \sum_i c_i^\top x_i \qquad \texttt{[charging cost]}$$

$$\text{subject to:} \quad x_i \in X_i, \text{ for all } i \qquad \texttt{[limitations on the charging rate]}$$

$$\sum_i \left(A_i x_i - \frac{b}{m} \right) \leq 0 \qquad \texttt{[power grid constraint]}$$

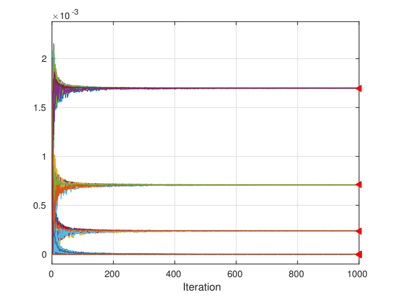
Example: Electric vehicle charging (cont'd)

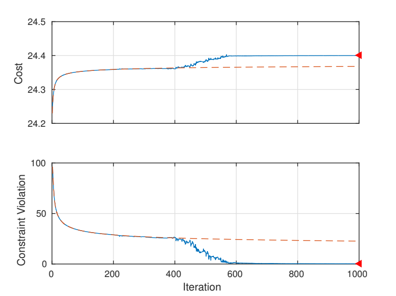
- To speed up convergence consider the sequence

$$\tilde{x}_i(k+1) = \begin{cases} \hat{x}_i(k+1) & k < k_{s,i} \\ \dfrac{\sum_{r=k_{s,i}}^{k} c(r) x_i(r+1)}{\sum_{r=k_{s,i}}^{k} c(r)} & k \geq k_{s,i} \end{cases}$$

- Refresh $c(k)$ when $k_{s,i}$ is an "event" condition

Convergence results: Dual variables – Objective & constraint violation





Summary

1. Distributed algorithms for cost coupled problems
   - Communication: Weighted averaging only between neighbours
   - Computation: Only local; proximal optimization or subgradient projection
   - Structural similarities, but also subtle differences
2. Distributed algorithms for constraint coupled problems
   - Can be thought of a proximal or gradient for the dual
   - Challenge is the primal recovery step

Distributed optimization in the presence of uncertainty

Centralized minimization program

$$\text{minimize } \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i X_i$$

Distributed optimization in the presence of uncertainty

Centralized minimization program

$$\text{minimize} \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i X_i(\delta), \text{ for all } \delta \in \Delta$$

Distributed optimization in the presence of uncertainty

Centralized minimization program

$$\text{minimize} \; \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i \bigcap_{\delta \in \Delta} X_i(\delta)$$

- Stochastic set-up
  - δ: Uncertain parameter $\delta \sim \mathbb{P}$
  - Δ: (Possibly) continuous set
  - Semi-infinite optimization program

Data driven approach

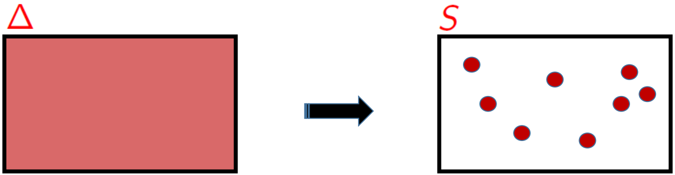
Centralized minimization program

$$\text{minimize} \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i \bigcap_{\delta \in \Delta} X_i(\delta)$$

- Replace Δ with S



Data driven approach

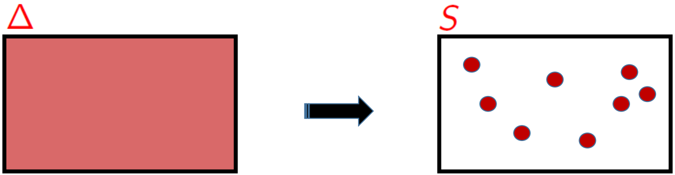
Centralized minimization program

$$\text{minimize} \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i \bigcap_{\delta \in S} X_i(\delta)$$

- Replace Δ with S



Data driven approach

Centralized minimization program

$$\text{minimize} \ \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i \bigcap_{\delta \in S} X_i(\delta)$$

- Two cases:

  1. Agents have the same data set S
  2. Agents have different data sets $\{S_i\}_i$

Data driven approach

Centralized minimization program

$$\text{minimize} \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i \bigcap_{\delta \in S_i} X_i(\delta)$$

- Two cases:
  1. Agents have the same data set S
  2. Agents have different data sets $\{S_i\}_i$

Common data set

Distributed implementation

Data based program

$$\text{minimize} \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i \bigcap_{\delta \in S} X_i(\delta)$$

- Apply proximal algorithm with $\bigcap_{\delta \in S} X_i(\delta)$ in place of X_i
- Let x_S^* denote the converged solution

Common data set

Probabilistic feasibility

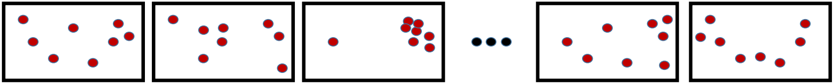
Data based program $\mathcal{P}_S$

$$\begin{aligned} &\text{minimize} \quad \sum_i f_i(x) \\ &\text{subject to} \qquad\qquad \rightarrow x_S^* \\ &x \in \bigcap_i \bigcap_{\delta \in S} X_i(\delta) \end{aligned}$$

Robust program $\mathcal{P}_\Delta$

$$\begin{aligned} &\text{minimize} \quad \sum_i f_i(x) \\ &\text{subject to} \\ &x \in \bigcap_i \bigcap_{\delta \in \Delta} X_i(\delta) \end{aligned}$$

- Is x_S^* feasible for $\mathcal{P}_\Delta$?
- Is this true for any S?



Common data set

Probabilistic feasibility

Data based program $\mathcal{P}_S$

$$\text{minimize} \sum_i f_i(x)$$

subject to $\rightarrow x_S^*$

$$x \in \bigcap_i \bigcap_{\delta \in S} X_i(\delta)$$

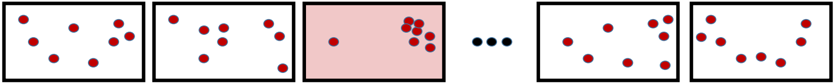
Robust program $\mathcal{P}_\Delta$

$$\text{minimize} \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i \bigcap_{\delta \in \Delta} X_i(\delta)$$

- Is x_S^* feasible for $\mathcal{P}_\Delta$?
- Is this true for any S?



Common data set

Probabilistic feasibility

Data based program $\mathcal{P}_S$

$$\begin{aligned} &\text{minimize} \quad \sum_i f_i(x) \\ &\text{subject to} \qquad\qquad\qquad \to x_S^* \\ &x \in \bigcap_i \bigcap_{\delta \in S} X_i(\delta) \end{aligned}$$

Robust program $\mathcal{P}_\Delta$

$$\begin{aligned} &\text{minimize} \quad \sum_i f_i(x) \\ &\text{subject to} \\ &x \in \bigcap_i \bigcap_{\delta \in \Delta} X_i(\delta) \end{aligned}$$

Feasibility link [Calafiore & Campi, TAC 2006]

Fix $\beta \in (0,1)$ and S. With confidence $\geq 1-\beta$, x_S^* is feasible for $\mathcal{P}_\Delta$ with probability $\geq 1-\epsilon(d,|S|,\beta)$, i.e.

$$\mathbb{P}\Big(\delta \in \Delta : \; x_S^* \notin \bigcap_i X_i(\delta)\Big) \leq \epsilon(d,|S|,\beta) \text{ with prob. } \geq 1-\beta$$

Feasibility link

Fix $\beta \in (0,1)$ and S. With confidence $\geq 1-\beta$, x_S^* is feasible for $\mathcal{P}_\Delta$ with probability $\geq 1-\epsilon(d,|S|,\beta)$, i.e.

$$\mathbb{P}\Big(\delta \in \Delta : \; x_S^* \notin \bigcap_i X_i(\delta)\Big) \leq \epsilon(d,|S|,\beta) \text{ with prob. } \geq 1-\beta$$

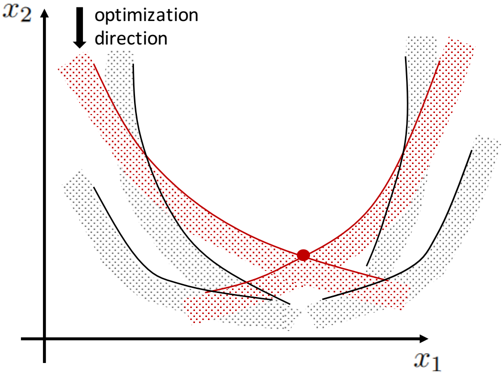
- On which parameters does ϵ depends on?

$$\epsilon = \frac{2}{|S|}\left(d + \ln\frac{1}{\beta}\right)$$

  - Logarithmic in β: $1-\beta$ can be set close to one
  - Linear in $|S|^{-1}$: The more data the better the result
  - Linear in d: # decision variables

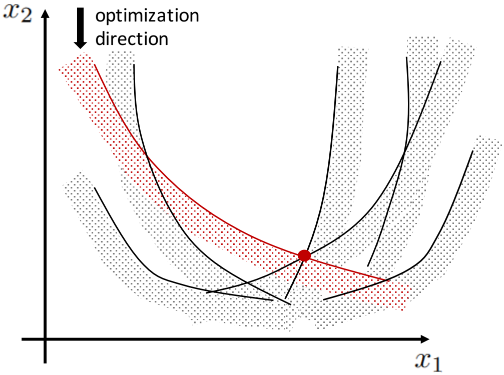
Parameter d: Number of support constraints

- For convex problems $d \leq \#$ decision variables
- Support constraints $\leq$ Active constraints



Parameter d: Number of support constraints

- For convex problems $d \leq \#$ decision variables
- Support constraints $\leq$ Active constraints



Different data sets

Distributed implementation

Data based program

$$\begin{aligned}&\text{minimize} \sum_i f_i(x)\\ &\text{subject to}\\ &x \in \bigcap_i \bigcap_{\delta \in S_i} X_i(\delta)\end{aligned}$$

- Apply proximal algorithm with $\bigcap_{\delta \in S_i} X_i(\delta)$ in place of X_i
- Let x_S^* denote the converged solution, $S = \{S_i\}_i$

Different data sets

Probabilistic feasibility

A posteriori

Fix $\beta \in (0,1)$ and $\{S_i\}_i$. With confidence $\geq 1-\beta$,

$$\mathbb{P}\Big(\delta \in \Delta : \; x_S^* \notin \bigcap_i X_i(\delta)\Big) \leq \sum_i \epsilon_i(d_i^{S_i})$$

- A posteriori result
  - $d_i^{S_i}$: empirical estimate of "support" samples (wait and judge)
    Changing S_i the result will change
  - Complexity of $\epsilon_i(d_i^{S_i})$ as in the previous case
  - Result thanks to [Campi, Garatti & Ramponi, CDC 2015]

Different data sets

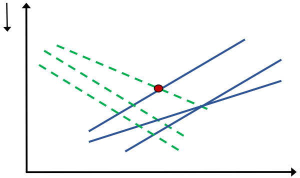
Probabilistic feasibility

A posteriori

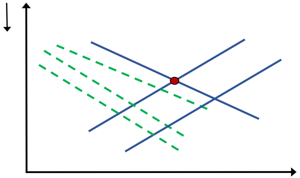
Fix $\beta \in (0, 1)$ and $\{S_i\}_i$. With confidence $\geq 1 - \beta$,

$$\mathbb{P}\Big(\delta \in \Delta : \; x_S^* \notin \bigcap_i X_i(\delta)\Big) \leq \sum_i \epsilon_i(d_i^{S_i})$$

- Two-agent example, $d = 2$



$d_1^{S_1} = 1$ and $d_2^{S_2} = 1$



$d_1^{S_1} = 0$ and $d_2^{S_2} = 2$

A posteriori

Fix $\beta \in (0,1)$ and $\{S_i\}_i$. With confidence $\geq 1-\beta$,

$$\mathbb{P}\left(\delta \in \Delta : \; x_S^* \notin \bigcap_i X_i(\delta)\right) \leq \sum_i \epsilon_i(d_i^{S_i})$$

- A posteriori result
  - Can we turn it into an a priori statement?
  - What is the worst-case value for $\sum_i \epsilon_i(d_i^{S_i})$ that we can "observe"
  - Conservative bound: $d_i^{S_i} < d$ for all i
  - Sharper bound: $\sum_i d_i^{S_i} = d$ (# decision variables)

Different data sets

Probabilistic feasibility

A priori

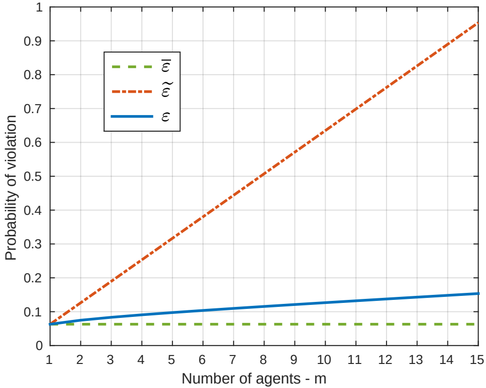
Fix $\beta \in (0,1)$ and $\left\{S_i\right\}_i$. With confidence $\geq 1-\beta$,

$$\mathbb{P}\left(\delta \in \Delta : \; x_S^* \notin \bigcap_i X_i(\delta)\right) \leq \epsilon$$

where

$$\begin{aligned} \epsilon \;\; = \;\; & \text{maximize} \sum_i \epsilon_i(d_i) \\ & \text{subject to} \\ & \sum_i d_i = d \end{aligned}$$

Common vs. different data sets

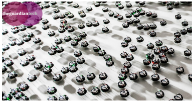
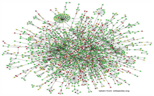


- Approach using different constraint sets
  - Close to the case of common data sets
  - Less conservative than evenly splitting ϵ, β across agents

Summary & Future work

- Making decisions in networks



  - General framework for distributed optimization
  - *A priori* probabilistic feasibility certificates in multi-agent systems
  - Multiple applications: Electric vehicles; Building control ...

- What comes next?
  - Convergence rate analysis See [Romao et al, CDC 2019]
  - Rolling horizon implementations
  - Optimality confidence intervals for distributed scenario approach
  - More applications

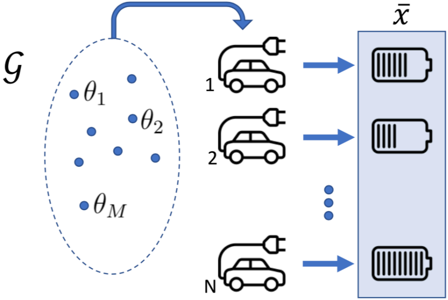
... and a couple of slides on Nash equilibrium seeking

Non-cooperative game

- Agents are selfish, non-cooperative entities
- Interested in minimizing some cost when other agents' strategies are fixed

$$J_i(x_i, x_{-i}) = f_i(x_i, x_{-i}) + \max_{k=1,\dots,|S|} g(x_i, x_{-i}, \delta_k)$$

- Price is subject to volatility $\Rightarrow$ Represent uncertainty by means of scenarios!



Non-cooperative game

- Agents are selfish, non-cooperative entities
- Interested in minimizing some cost when other agents' strategies are fixed

$$J_i(x_i, x_{-i}) = f_i(x_i, x_{-i}) + \max_{k=1,\dots,|S|} g(x_i, x_{-i}, \delta_k)$$

- Price is subject to volatility $\Rightarrow$ Represent uncertainty by means of scenarios!

Nash equilibrium

A set of agents' strategies $(\bar{x}_i, \bar{x}_{-i})$ forms a Nash equilibrium if for all i

$$J_i(\bar{x}_i, \bar{x}_{-i}) \leq J_i(x_i, \bar{x}_{-i}), \text{ for all } x_i \in X_i,$$

where X_i is agent's i local constraint set.

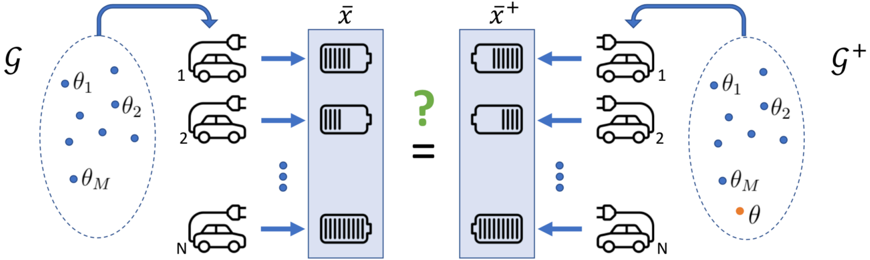
No agent can improve her cost when other agents' strategies are fixed

Non-cooperative game

- Nash equilibrium is a random variable
- How likely is it to remain unchanged when a new uncertainty realization is encountered?

$$\mathbb{P}^M\left\{\delta_1,\ldots,\delta_M : \mathbb{P}\left\{\delta : \bar{x} = \bar{x}^+\right\} \geq 1-\epsilon\right\} \geq 1-\beta$$

- Probably approximately correct Nash equilibrium learning



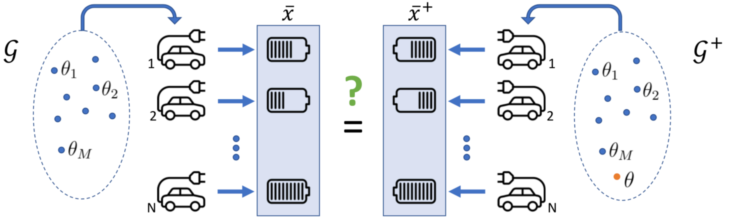
Non-cooperative game

Probably approximately correct Nash equilibrium learning

Fix $\beta \in (0,1)$, and consider the function $\epsilon(\cdot)$ such that

$$\epsilon(M) = 1 \text{ and } \sum_{k=0}^{M-1} \binom{M}{k} (1 - \epsilon(k))^{M-k} = \beta$$

We then have $\mathbb{P}^M \left\{ \delta_1, \ldots, \delta_M : \ \mathbb{P}\left\{ \delta : \ \bar{x} = \bar{x}^+ \right\} \geq 1 - \epsilon(\bar{d}) \right\} \geq 1 - \beta$,
where $\bar{d}$ is the number of support samples, i.e., $\bar{x}_{\bar{d}} = \bar{x}_M$.



References

Margellos, Falsone, Garatti & Prandini (2018)
Distributed constrained optimization and consensus in uncertain networks via proximal minimization
IEEE Transactions on Automatic Control, 63(5), 1372-1387.

Falsone, Margellos, Garatti & Prandini (2017)
Dual decomposition and proximal minimization for multi-agent distributed optimization with coupling constraints
Automatica, 84(10), 149-158.

Romao, Margellos, Notarstefano & Papachristodoulou (2019)
Convergence rate analysis of a subgradient averaging algorithm for distributed optimization with different constraint sets
IEEE Conference on Decision and Control, to appear.

Fele & Margellos (2019)
Probably approximately correct Nash equilibrium learning
IEEE Transactions on Automatic Control (shorter version to appear at CDC), under review.

Thank you for your attention!
Questions?

Contact at:

kostas.margellos@eng.ox.ac.uk