Learning and generalization in uncertain networks

Kostas Margellos

University of Oxford

Credit

- Alessandro Falsone

- Luca Deori

- Simone Garatti

- Maria Prandini

Logistics

- Who: Kostas Margellos, Dept Engineering Science, UOXF
  Background: Control, optimization and a bit of energy systems
  contact: `kostas.margellos@eng.ox.ac.uk`
- When: 3 hours @10:00-13:00; 2 10min breaks
- Material: Slides will be available online, together with papers for further reading

Please email me in case of remaining questions!

Course information

- Big picture
  - Decision making in the presence of uncertainty
  - Related to: Randomized/stochastic and robust optimization
  - Builds on: Convex optimization ... and a bit of Statistical Learning Theory
- What do you expect out of it?
  - Your turn!

Course information

- What it is actually about

  1. Introduce data based optimization
  2. Make decisions under uncertainty and accompany them with performance certificates
  3. Offer a new toolkit: easy implementation – difficulty comes in the math
  4. Naturally applicable to many energy management problems: optimal power flow; demand response; building control; electric vehicles

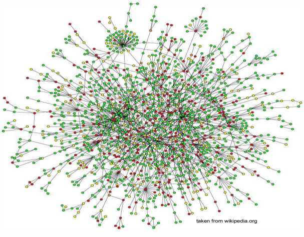
Motivation

- Social networks

- Power networks

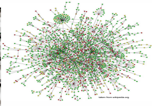
- Robotic networks

- Biological networks



What do these networks have in common?

- Involve agents that what to make decisions

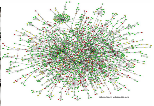


  - Centralized decision making
  - Distributed decision making (yesterday's lecture + a bit today)

What do these networks have in common?

- Involve agents that what to make decisions





  - Centralized decision making
  - Distributed decision making (yesterday's lecture + a bit today)

- They are affected by uncertainty
  - Endogenous uncertainty, e.g., unmodelled dynamics as in biological networks
  - Exogenous uncertainty, e.g., renewable energy sources as in energy systems

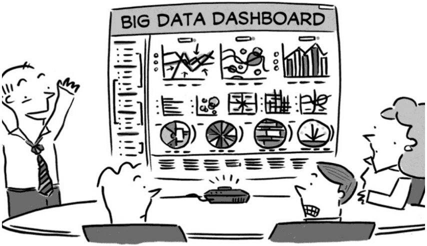
How to deal with uncertainty?

- There are many ways
  - Deterministic: Just stick with the forecasts
    Simple but agnostic!
  - Robust: Consider the worst-case
    Offers immunization but conservative!

How to deal with uncertainty?

- There are many ways
  - Deterministic: Just stick with the forecasts
    Simple but agnostic!
  - Robust: Consider the worst-case
    Offers immunization but conservative!

- Let the DATA speak



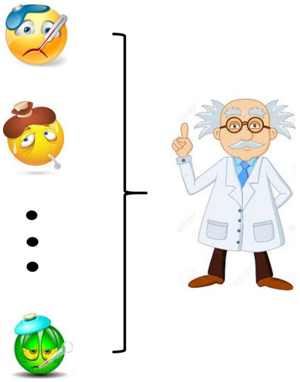
'After careful consideration of all 437 charts, graphs, and metrics,
I've decided to throw up my hands, hit the liquor store,
and get snockered. Who's with me?!"

How to deal with uncertainty – Learning outcomes

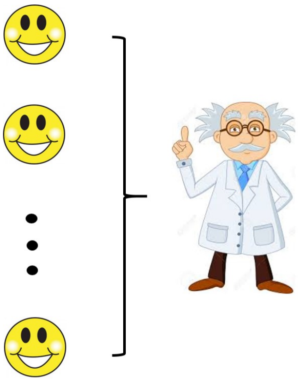
- In this lecture we will

  1. Use information from data to make decisions ⇒ Learning
  2. Accompany these decisions with certificates ⇒ Generalization
  3. Discuss about several applications
     - ★ Generation-load dispatch
     - ★ Energy efficient building control
     - ★ Charging control of electric vehicles

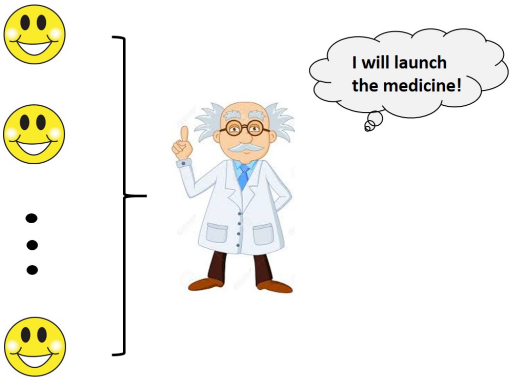
Motivation - The doctor's problem



Motivation - The doctor's problem

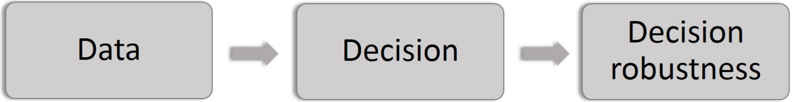


Motivation - The doctor's problem

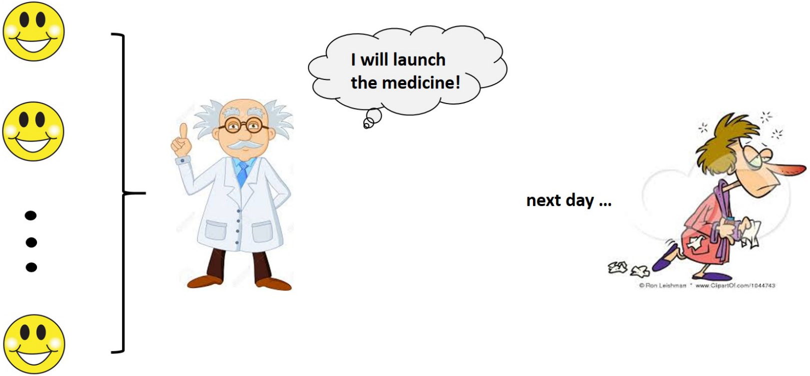


Motivation - The doctor's problem

Motivation - The doctor's problem

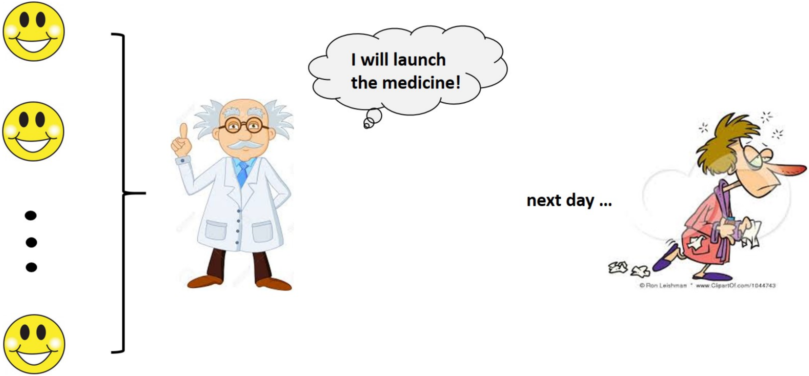


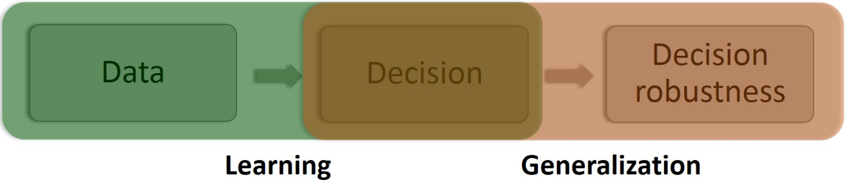
Motivation - The doctor's problem



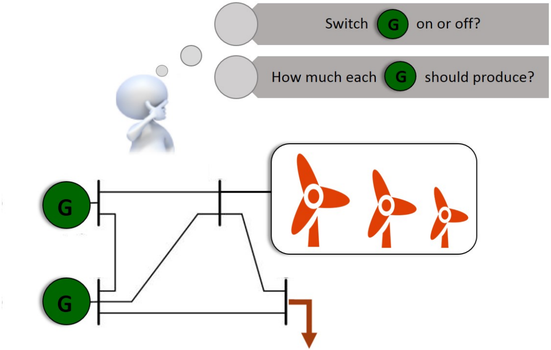
Learning

Motivation - The doctor's problem



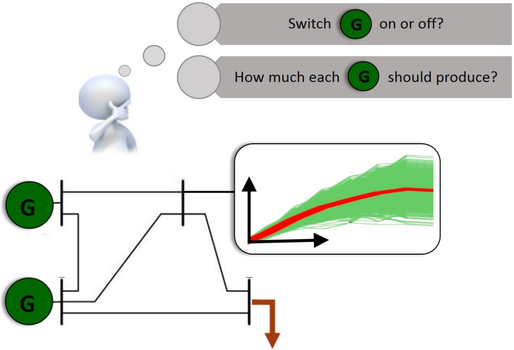


Motivation - Power system planning problems



- Tasks of the Transmission System Operator
  - Make decisions in the presence of uncertainty
  - Decisions are often based on data; Robustness against "unseen" uncertainties (generalization)?

Motivation - Power system planning problems



- Tasks of the Transmission System Operator
  - Make decisions in the presence of uncertainty
  - Decisions are often based on data; Robustness against "unseen" uncertainties (generalization)?

Outline

1. An abstract learning setting
   - Learning from data
   - Generalizing learned decisions
2. Data based optimization
   - Optimal decisions with robustness certificates
   - *Example:* Power system dispatch problems under uncertainty
3. Distributed learning in networks
   - Distributed algorithms – the deterministic case
   - Distributed algorithms – the uncertain case
   - *Example:* Energy management in buildings
4. Summary & ... a glimpse on electric vehicle charging games!

An aside note on randomness

1. Consider the most popular random experiment: coin tossing
   - Toss a fair coin 100 times
   - Calculate the frequency of getting a head, i.e. the head probability
2. Repeat it the experiment 50 times
   - You will get 50 different head probabilities: $0.55, 0.47, 0.53, \ldots$
   - Head probability is itself random!

Question:

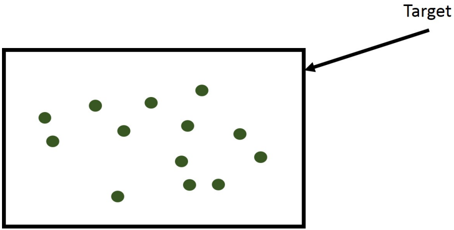
How **many times** shall you toss the coin initially so that the head probability is very close to 0.5 for most of the 50 trials?

or, in other words ...

How **many data** do we need to make probably, approximately correct decisions?

Learning

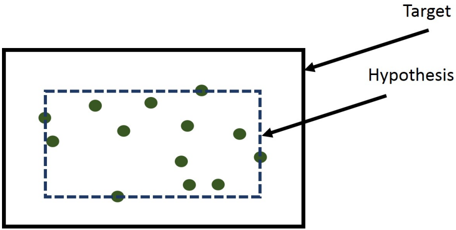
- Decision problem[1]
  - Consider an axis-aligned rectangle – Target set T
  - We are given m samples that belong in T – $\{\delta_i\}_{i=1}^{m}$
- Estimate T as the smallest axis-aligned rectangle that contains the samples – Hypothesis H_m



[1] [Vidyasagar, 1997]

Learning

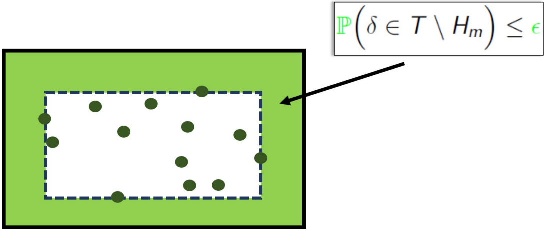
- Decision problem[2]
  - Consider an axis-aligned rectangle – Target set T
  - We are given m samples that belong in T – $\{\delta_i\}_{i=1}^{m}$
- Estimate T as the smallest axis-aligned rectangle that contains the samples – Hypothesis H_m



[2] [Vidyasagar, 1997]

Generalization

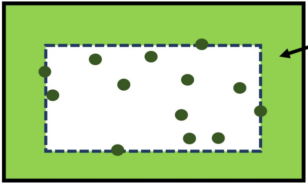
- How likely is it that H_m contains another sample δ?
  - Depends on the "distance": $\mathbb{P}\Big(\delta \in T \setminus H_m\Big)$
  - ☺ if $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$
- Is it the case for any m samples?
  - What is the probability that $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$?



Generalization

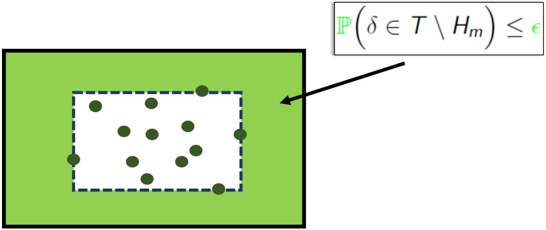
- How likely is it that H_m contains another sample δ?
  - Depends on the "distance": $\mathbb{P}\Big(\delta \in T \setminus H_m\Big)$
  - ☺ if $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$
- Is it the case for any m samples?
  - What is the probability that $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$?

$$\boxed{\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon}$$



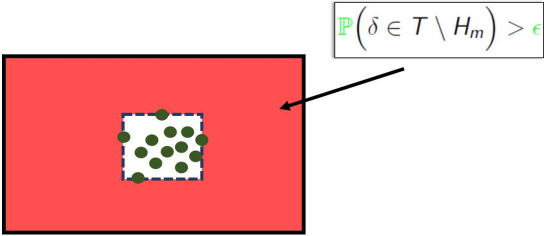
Generalization

- How likely is it that H_m contains another sample δ?
  - Depends on the "distance": $\mathbb{P}\Big(\delta \in T \setminus H_m\Big)$
  - ☺ if $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$
- Is it the case for any m samples?
  - What is the probability that $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$?



Generalization

- How likely is it that H_m contains another sample δ?
  - Depends on the "distance": $\mathbb{P}\Big(\delta \in T \setminus H_m\Big)$
  - ☺ if $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$
- Is it the case for any m samples?
  - What is the probability that $\mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon$?



Generalization

- In the doctor's problem: Doctor would be satisfied if ...
  - Medicine cures patients with probability at least $1 - \epsilon$
  - If this holds with probability at least $1 - q(m, \epsilon)$ with respect to the m trial patients
  - Let S be a collection of m samples $\delta_1, \ldots, \delta_m$

Problem

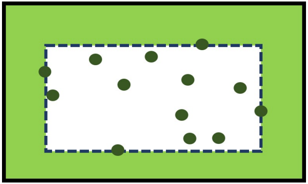
Find conditions for the existence of some $q(m, \epsilon)$ such that

$$\mathbb{P}^m \Big\{ S : \ \mathbb{P}\Big(\delta \in T \setminus H_m \Big) \leq \epsilon \Big\} \geq 1 - q(m, \epsilon)$$

and $\lim_{m \to \infty} q(m, \epsilon) = 0$.

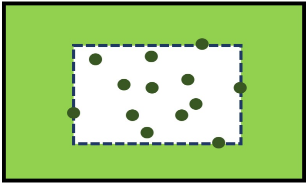
Generalization - sufficient condition

- Observation
  - For any m samples often *only* a subset of them matters



Generalization - sufficient condition

- Observation
  - For any m samples often *only* a subset of them matters



Generalization - sufficient condition

- Observation
  - For any m samples often *only* a subset of them matters

Compression scheme

Assume that for any m samples $\exists d < m$ such that

$$\mathbb{1}_{H_d}(\delta_i) = \mathbb{1}_T(\delta_i), \quad \text{for all } i = 1, \ldots, m$$

- Hypothesis based on d samples agrees with the target on all of them
- Existence of a compression scheme $\Leftrightarrow$ Empirical generalization

Generalization - sufficient condition

Problem

Find conditions for the existence of some $q(m,\epsilon)$ such that

$$\mathbb{P}^m\Big\{S:\ \mathbb{P}\Big(\delta \in T \setminus H_m\Big) \le \epsilon\Big\} \ge 1 - q(m,\epsilon)$$

and $\lim_{m\to\infty} q(m,\epsilon) = 0$.

Generalization - sufficient condition

Theorem

If a **compression scheme with cardinality** d exists, then

$$\mathbb{P}^m \Big\{ S : \; \mathbb{P}\Big(\delta \in T \setminus H_m\Big) \le \epsilon \Big\} \ge 1 - q(m, \epsilon)$$

with $q(m, \epsilon) = \binom{m}{d}(1 - \epsilon)^{m-d}$.

- Inspired by [Floyd & Warmuth, 1995]; Proof and extensions [Margellos, Prandini & Lygeros, 2015]
- It holds $\lim_{m \to \infty} q(m, \epsilon) = 0$

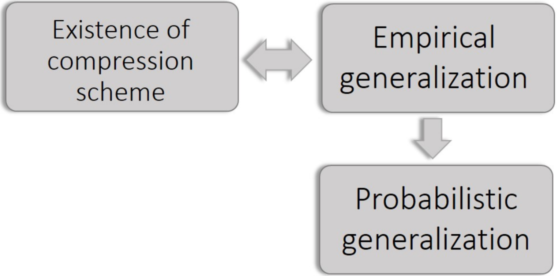
Generalization - sufficient condition

Theorem

If a **compression scheme with cardinality** d exists, then

$$\mathbb{P}^m\left\{S:\ \mathbb{P}\left(\delta \in T \setminus H_m\right) \leq \epsilon\right\} \geq 1 - q(m,\epsilon)$$

with $q(m,\epsilon) = \binom{m}{d}(1-\epsilon)^{m-d}$.



Outline

1. An abstract learning setting
    - Learning from data
    - Generalizing learned decisions
2. Data based optimization
    - Optimal decisions with robustness certificates
    - *Example:* Power system dispatch problems under uncertainty
3. Distributed learning in networks
    - Distributed algorithms – the deterministic case
    - Distributed algorithms – the uncertain case
    - *Example:* Energy management in buildings
4. Summary & ... a glimpse on electric vehicle charging games!

Data based optimization

- Uncertain program

$$\min_{x \in \mathbb{R}^{n_x}} \quad J(x)$$
$$\text{subject to:} \quad g(x,\delta) \leq 0, \ \text{for all } \delta \in S$$

- Finite dimensional optimization problem
  - Finite number of decision variables $x \in \mathbb{R}^{n_x}$
  - Represent uncertainty $\delta \in \mathbb{R}^{n_\delta}$, $\delta \sim \mathbb{P}$ with support Δ, by means of i.i.d. samples
  - Finite number of constraints (one for every sample)
- Denote by x_S^* its minimizer

Data based optimization

- Uncertain program

$$\begin{aligned} \min_{x \in \mathbb{R}^{n_x}} \quad & J(x) \\ \text{subject to:} \quad & \\ & g(x,\delta) \le 0, \text{ for all } \delta \in S \end{aligned}$$

- Connections with learning – Learn the uncertainty space Δ

Target set	$T = \Delta$, (i.e. $\mathbb{1}_T(\delta) = 1, \ \forall \delta \in \Delta$)
Decision	Minimizer $\Rightarrow x_S^*$
Hypothesis	$H_m = \left(\delta \in \Delta : \ g(x_S^*, \delta) \le 0\right)$

- Estimation error = Probability of constraint violation

$$\mathbb{P}\left(\delta \in T \setminus H_m\right) = \mathbb{P}\left(\delta \in \Delta : \ g(x_S^*, \delta) > 0\right)$$

Data based optimization

- Uncertain program

$$\min_{x \in \mathbb{R}^{n_x}} \quad J(x)$$
$$\text{subject to:}$$
$$g(x,\delta) \le 0, \ \text{for all } \delta \in S$$

- Connections with learning – Learn the uncertainty space Δ

Target set	$T = \Delta$, (i.e. $\mathbb{1}_T(\delta) = 1, \ \forall \delta \in \Delta$)
Decision	Minimizer $\Rightarrow x_S^*$
Hypothesis	$H_m = \left(\delta \in \Delta : \ g(x_S^*, \delta) \le 0\right)$

- Estimation error = Probability of constraint violation

$$\mathbb{P}\left(\delta \in T \setminus H_m\right) = \mathbb{P}\left(\delta \in \Delta : \ g(x_S^*, \delta) > 0\right)$$

Data based optimization – Generalization

Theorem

If a **compression scheme with cardinality** d exists, then

$$\mathbb{P}^m\left\{S : \mathbb{P}\Big(\delta \in T \setminus H_m\Big) \leq \epsilon\right\} \geq 1 - q(m,\epsilon)$$

with $q(m,\epsilon) = \binom{m}{d}(1-\epsilon)^{m-d}$.

Data based optimization – Generalization

Theorem

If a **compression scheme with cardinality** d exists, then

$$\mathbb{P}^m\Big\{S:\ \mathbb{P}\Big(\delta\in\Delta:\ g(x_S^*,\delta)>0\Big)\le\epsilon\Big\}\ge 1-q(m,\epsilon)$$

with $q(m,\epsilon)=\binom{m}{d}(1-\epsilon)^{m-d}$.

Data based optimization – Generalization

Theorem

If a **compression scheme with cardinality** d exists, then

$$\mathbb{P}^m\Big\{S:\ \mathbb{P}\Big(\delta\in\Delta:\ g(x_S^*,\delta)>0\Big)\le\epsilon\Big\}\ge 1-q(m,\epsilon)$$

with $q(m,\epsilon)=\binom{m}{d}(1-\epsilon)^{m-d}$.

- We can calculate the number of samples so that

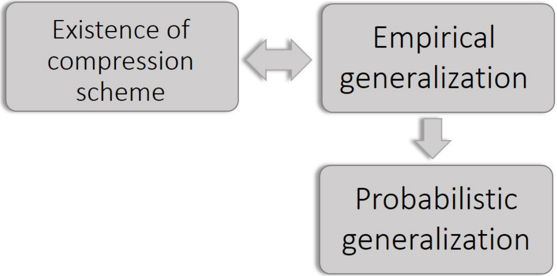
Data based optimization – Generalization

Theorem

If a **compression scheme with cardinality** d exists, then

$$\mathbb{P}^m\Big\{S:\ \mathbb{P}\Big(\delta\in\Delta:\ g(x_S^*,\delta)>0\Big)\le\epsilon\Big\}\ge 1-q(m,\epsilon)$$

with $q(m,\epsilon)=\binom{m}{d}(1-\epsilon)^{m-d}$.

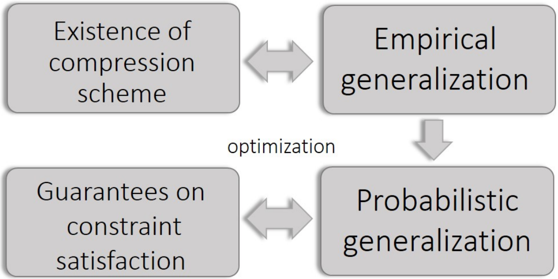


Theorem

If a **compression scheme with cardinality** d exists, then

$$\mathbb{P}^m\Big\{S:\ \mathbb{P}\Big(\delta \in \Delta:\ g(x^*_S,\delta) > 0\Big) \leq \epsilon\Big\} \geq 1 - q(m,\epsilon)$$

with $q(m,\epsilon) = \binom{m}{d}(1-\epsilon)^{m-d}$.



Convex uncertain programs

$$\min_{x \in \mathbb{R}^{n_x}} \quad J(x)$$

subject to:

$$g(x,\delta) \le 0, \text{ for all } \delta \in S$$

- $J(\cdot)$ convex, for any $\delta \in \Delta$, $g(\cdot,\delta)$ convex[3]

Compression scheme

- $d = \#$ support constraints
  - All other constraints are redundant; do not affect the objective
- $\#$ support constraints $\le \; n_x \Rightarrow d \le n_x$
  - Guarantees on the probability of constraint violation: $q(m,\epsilon) = \binom{m}{n_x}(1-\epsilon)^{m-n_x}$

[3] [Calafiore & Campi, 2006], [Margellos, Prandini & Lygeros, 2015]

Convex uncertain programs

$$\min_{x \in \mathbb{R}^{n_x}} \quad J(x)$$
subject to:
$$g(x,\delta) \le 0, \text{ for all } \delta \in S$$

- $J(\cdot)$ convex, for any $\delta \in \Delta$, $g(\cdot,\delta)$ convex[4]

Compression scheme

- d = # support constraints
  - All other constraints are redundant; do not affect the objective
- # support constraints $\le \; n_x \Rightarrow d \le n_x$
  - Guarantees on the probability of constraint violation:
    $q(m,\epsilon) = \binom{m}{n_x}(1-\epsilon)^{m-n_x}$

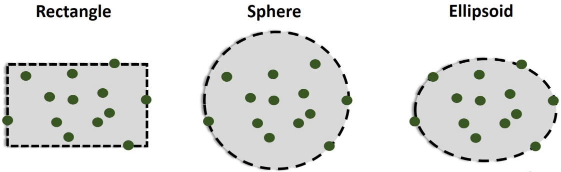
[4] [Calafiore & Campi, 2006], [Margellos, Prandini & Lygeros, 2015]

Probabilistically robust programs

$$\min_{x \in \mathbb{R}^{n_x}} \quad J(x)$$

subject to:

$$g(x,\delta) \leq 0, \text{ for all } \delta \in B\left(S\right)$$



$n_p = 2n_\delta$ $\quad$ $n_p = n_\delta + 1$ $\quad$ $n_p = n_\delta + \frac{n_\delta(n_\delta+1)}{2}$

- Proposed in [Margellos, Goulart & Lygeros, 2014]; solvability using robust optimization techniques, e.g. [Bertsimas & Sim, 2006]

Probabilistically robust programs

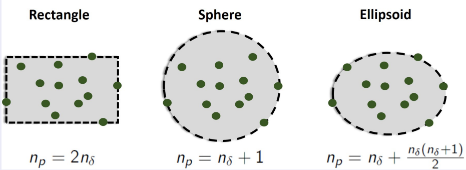
$$\min_{x \in \mathbb{R}^{n_x}} \quad J(x)$$

subject to:

$$g(x,\delta) \leq 0, \text{ for all } \delta \in B\left(S\right)$$

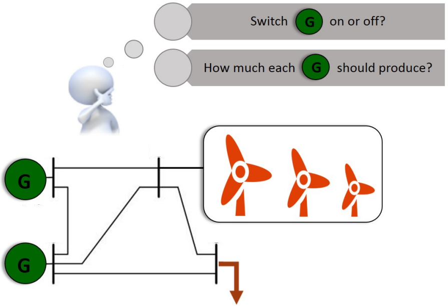
Compression scheme

- $d \leq n_p$, i.e. $\#$ parameters in $B\left(S\right)$



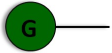
- Guarantees on the probability of constraint violation:
  $q(m,\epsilon) = \binom{m}{n_p}(1-\epsilon)^{m-n_p}$

Reserve scheduling



- Uncertainty
  - Wind power: Sum of a deterministic and a stochastic component $P_w = P_w^f + \delta$
  - Other uncertainties (e.g. PV power, load) can be treated similarly

Reserve scheduling

- Generating power 

    - For each unit G, sum of a deterministic and a stochastic component

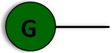
$$P_G - d_G^{\uparrow}\delta - d_G^{\downarrow}\delta$$

    - $d_G^{\uparrow}$, $d_G^{\downarrow}$: Distribution coefficients with $\sum_G d_G^{\uparrow} = 1, \;\; \sum_G d_G^{\downarrow} = 1$

$$\texttt{if}\ \left(\delta \;\Leftrightarrow\; \delta \leq 0\right),\ \texttt{then}\ \uparrow \text{production}$$
$$\texttt{if}\ \left(\delta \;\Leftrightarrow\; \delta > 0\right),\ \texttt{then}\ \downarrow \text{production}$$

- Reserve power
    - $R(\delta) = -d_G^{\uparrow}\delta - d_G^{\downarrow}\delta$
    - $R(0) = 0$, i.e. no reserves in the deterministic set-up
    - Reserve policies, as a function of the uncertainty

Reserve scheduling

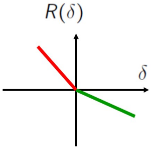
- Generating power 
  - For each unit G, sum of a deterministic and a stochastic component

$$P_G - d_G^{\uparrow}\delta - d_G^{\downarrow}\delta$$

  - $d_G^{\uparrow}$, $d_G^{\downarrow}$: Distribution coefficients with $\sum_G d_G^{\uparrow} = 1$, $\sum_G d_G^{\downarrow} = 1$

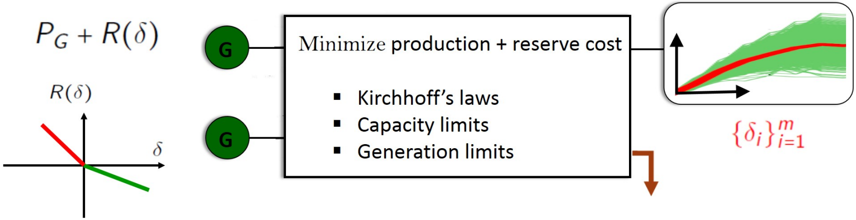
$$\begin{aligned} &\texttt{if } \left(\delta \;\Leftrightarrow\; \delta \leq 0\right), \texttt{ then } \uparrow \text{ production} \\ &\texttt{if } \left(\delta \;\Leftrightarrow\; \delta > 0\right), \texttt{ then } \downarrow \text{ production} \end{aligned}$$

- Reserve power
  - $R(\delta) = -d_G^{\uparrow}\delta - d_G^{\downarrow}\delta$
  - $R(0) = 0$, i.e. no reserves in the deterministic set-up
  - Reserve policies, as a function of the uncertainty



Reserve scheduling

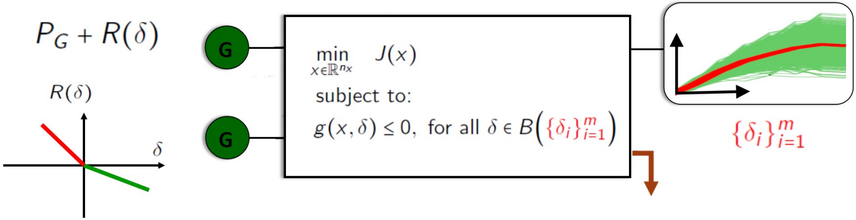
- Optimization problem



- Dynamics (e.g. storage, loads, etc) can be included
- Multi-stage optimization program (in case of day-ahead markets)
- Objective taken as the worst-case cost; similarly for expected value, or some other risk metric

Reserve scheduling

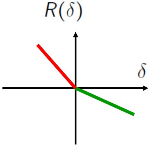
- Optimization problem



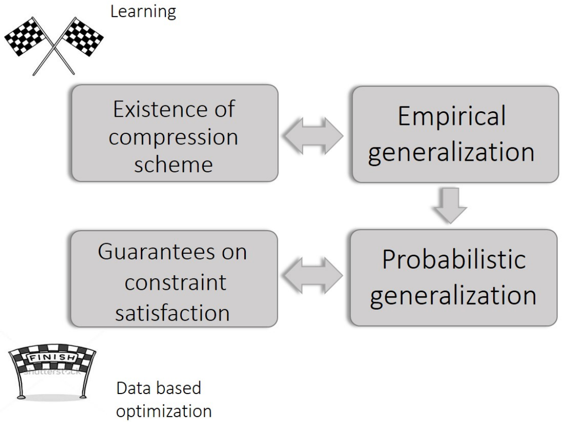
- Dynamics (e.g. storage, loads, etc) can be included
- Multi-stage optimization program (in case of day-ahead markets)
- Objective taken as the worst-case cost; similarly for expected value, or some other risk metric

Reserve scheduling

- Achievements
  - Amount of reserves to purchase
    – (Probabilistic) worst-case values of $R(\delta)$
  - Reserve allocation policy to be applied in real-time
  - A-priori guarantees on the feasibility of the solution, based only on m scenarios; in contrast to a-posteriori validation



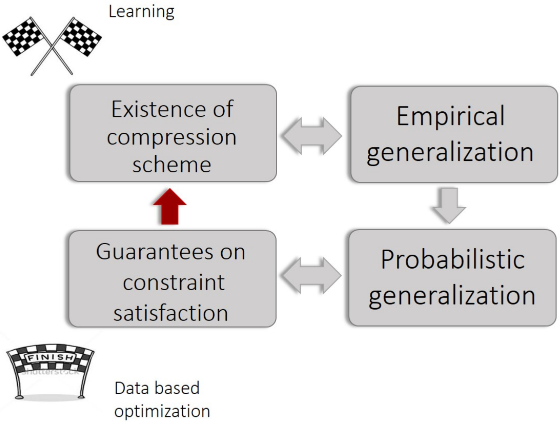
Summary & Extensions



- "Stronger" results (in learning and optimization) also available[5]
- Can we follow the opposite route (necessary conditions)?

[5] [Margellos, Prandini & Lygeros, 2015]

Summary & Extensions



- "Stronger" results (in learning and optimization) also available[6]
- Can we follow the opposite route (necessary conditions)?

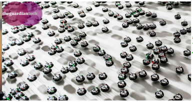
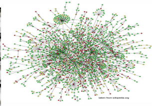
[6] [Margellos, Prandini & Lygeros, 2015]

Outline

1. An abstract learning setting
   - Learning from data
   - Generalizing learned decisions
2. Data based optimization
   - Optimal decisions with robustness certificates
   - *Example:* Power system dispatch problems under uncertainty
3. Distributed learning in networks
   - Distributed algorithms – the deterministic case
   - Distributed algorithms – the uncertain case
   - *Example:* Energy management in buildings
4. Summary & ... a glimpse on electric vehicle charging games!

Motivation

- Networks (Power, Social, etc.)

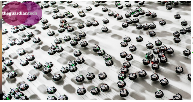
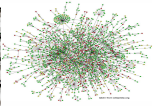


  - Large scale infrastructures
  - Multi-agent – Multiple interacting entities/users
  - Heterogeneous – Different physical or technological constraints per agent; different objectives per agent

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

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

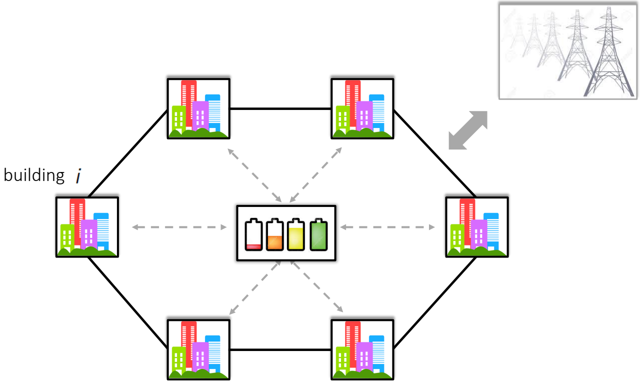
Why go distributed?

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

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

Building energy management problem

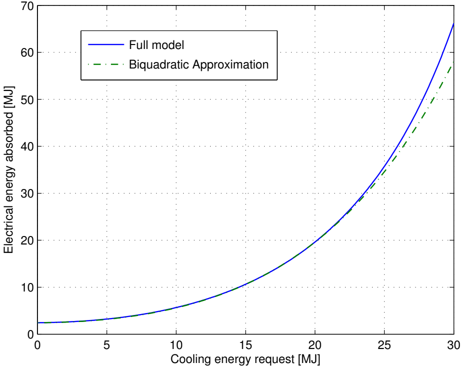


- Each building equipped with a chiller plant
- Shared storage device
- Determine optimal use of storage + temperature set-points to minimize the cost of chiller's energy request

Building energy management problem

Main modules

1. Chiller plant
   - Electrical energy into cooling energy
   - Chiller's electrical vs. cooling energy



2. Building's energy contribution
   - Walls-zones energy exchange – building thermal dynamics
   - Energy due to people occupancy
   - Zone thermal inertia
   - Other internal energy contribution, e.g. internal lighting, radiation through windows

3. Thermal storage

$$S(k+1) = \alpha S(k) - \sum_i s_i(k)$$

   - $S(k)$: Energy stored
   - $s_i(k)$: Energy exchange between building i and storage
     < 0: charging; > 0: discharging
   - α: energy losses coefficient

Building energy management problem

Optimization problem

minimize Cost of chiller's energy request

subject to

1. Chiller's energy request = Building's energy request − storage energy
2. Storage dynamics
3. Storage limits, chiller's limits, comfort limits

Compact form – x: Temperature set-points, storage power

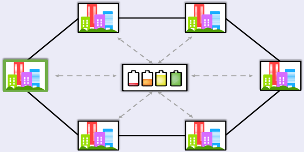
$$\text{minimize} \sum_i f_i(x)$$

subject to

$$x \in \bigcap_i X_i$$

Proposed distributed algorithm

Step 1: Local problem of agent i

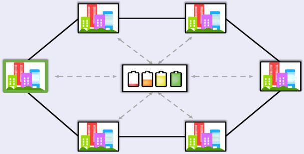


$$\left.\begin{array}{l} \text{minimize } f_i(x_i) + g(x_i, p_i) \\ \text{subject to} \\ x_i \in X_i \end{array}\right\} \Rightarrow x_i^*(p_i)$$

- x_i: "copy" of x maintained by agent i
- X_i: local constraint set of agent i
- p_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
  $g(x_i, p_i)$: Proxy term, penalizing disagreement with other agents

Proposed distributed algorithm

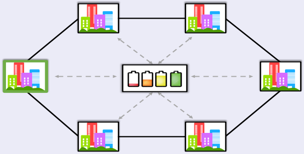
Step 1: Local problem of agent i



$$\left.\begin{aligned} &\text{minimize } f_i(x_i) + g(x_i, p_i) \\ &\text{subject to} \\ &x_i \in X_i \end{aligned}\right\} \Rightarrow x_i^*(p_i)$$

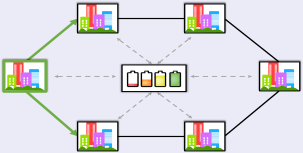
Proposed distributed algorithm

Step 1: Local problem of agent i

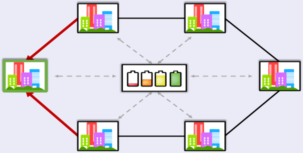


$$\left.\begin{array}{l}\text{minimize } f_i(x_i) + g(x_i, p_i) \\ \text{subject to} \\ x_i \in X_i\end{array}\right\} \Rightarrow x_i^*(p_i)$$

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

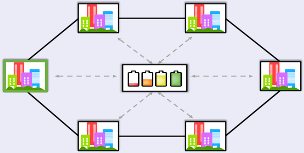


Step 2b: Receive neighbors' solutions



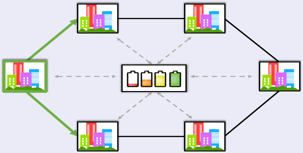
Proposed distributed algorithm

Step 1: Local problem of agent i

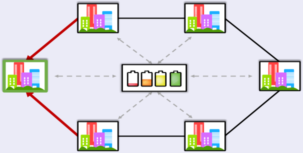


$$\left.\begin{array}{l} \text{minimize } f_i(x_i) + g(x_i, p_i) \\ \text{subject to} \\ x_i \in X_i \end{array}\right\} \Rightarrow x_i^*(p_i)$$

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



Step 2b: Receive neighbors' solutions

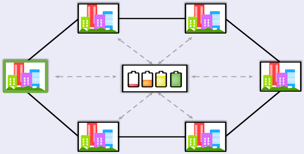


Step 3: Update p_i on the basis of information received

Go to Step 1

Proposed distributed algorithm

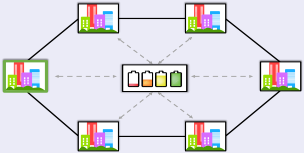
Local problem of agent i



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

Proposed distributed algorithm

Local problem of agent i

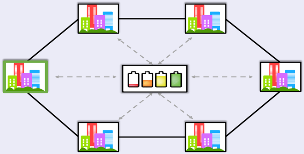


$$\left.\begin{array}{l} \text{minimize } f_i(x_i) + g(x_i, p_i) \\ \text{subject to} \\ x_i \in X_i \end{array}\right\} \Rightarrow x_i^*(p_i)$$

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

Proposed distributed algorithm

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

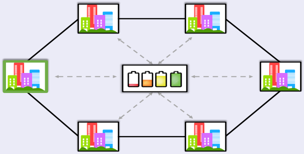


$$z_i(k) = \sum_j a_j^i(k) x_j(k)$$

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

Proposed distributed algorithm

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



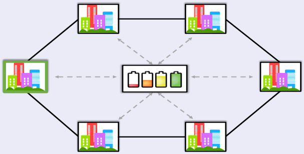
$$z_i(k) = \sum_j a^i_j(k) x_j(k)$$

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

- Information vector
  - $z_i(k) = \sum_j a^i_j(k) x_j(k)$
  - $a^i_j(k)$: 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

Proposed distributed algorithm

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)?

Algorithm analysis

Assumptions

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

1. Convexity and compactness
   - $f_i(\cdot)$: convex for all i
   - X_i: compact, convex, non-empty interior for all i
     $\Rightarrow f_i(\cdot)$: Lipschitz continuous on X_i
2. Choice of the proxy term
   - $\{c(k)\}_k$: non-increasing
   - Should not decrease too fast

$$\sum_k c(k) = \infty$$
$$\sum_k c(k)^2 < \infty$$

   - E.g., harmonic series

Algorithm analysis

Assumptions

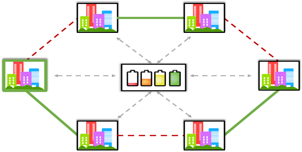
3. Information mix
   - Weights $a^i_j(k)$: non-zero lower bound if link between $i - j$ present
     $\Rightarrow$ Info mixing at a non-diminishing rate
   - Weights $a^i_j(k)$: form a doubly stochastic matrix
     $\Rightarrow$ Agents influence each other equally in the long run

Algorithm analysis

Assumptions

3. 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

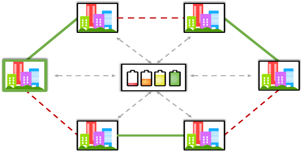
4. Network connectivity – All information flows (eventually)
   - Any pair of agents communicates infinitely often



   - Bounded intercommunication time

3. 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

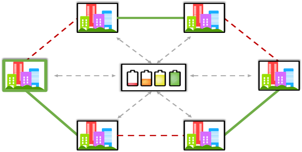
4. Network connectivity – All information flows (eventually)
   - Any pair of agents communicates infinitely often



   - Bounded intercommunication time

3. 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

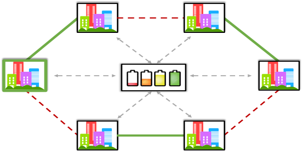
4. Network connectivity – All information flows (eventually)
   - Any pair of agents communicates infinitely often



   - Bounded intercommunication time

3. 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

4. Network connectivity – All information flows (eventually)
   - Any pair of agents communicates infinitely often



   - Bounded intercommunication time

Main result

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

Connection with other methods

- 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 netv 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 const 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 other methods

- Proximal algorithms

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

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

- 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

Problem set-up

Centralized minimization program

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

subject to

$$x \in \bigcap_i X_i$$

Problem set-up

Centralized minimization program

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

subject to

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

Problem set-up

Centralized minimization program

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

subject to

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

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

Problem set-up

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

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

Problem set-up

Data based 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

Problem set-up

Data based 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 $\{S_i\}_i$

Problem set-up

Data based 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

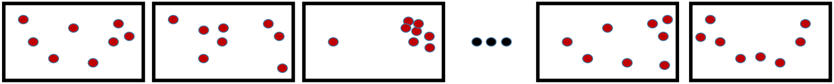
Data based program $\mathcal{P}_S$

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

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 $\mathcal{P}_S$

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

subject to $\qquad \rightarrow x_S^*$

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

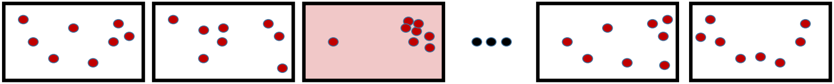
Robust program $\mathcal{P}_\Delta$

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

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 $\mathcal{P}_S$

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

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

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

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

Different data sets

Distributed implementation

Data based program

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

subject to

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

- 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

Single-agent - a posteriori

Fix $\beta_i \in (0,1)$ and S_i. With confidence $\geq 1 - \beta_i$,

$$\mathbb{P}\Big(\delta \in \Delta : \ x_S^* \notin X_i(\delta)\Big) \leq \epsilon_i(d_i^{S_i}, |S_i|, \beta_i)$$

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

Different data sets

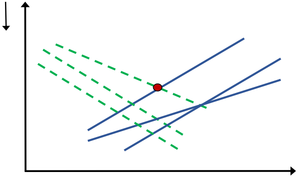
Probabilistic feasibility

Single-agent - a posteriori

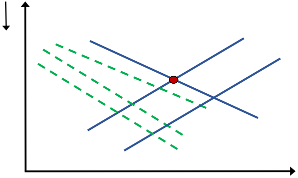
Fix $\beta_i \in (0,1)$ and S_i. With confidence $\geq 1 - \beta_i$,

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

- Two-agent example, $d = 2$



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



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

Different data sets

Probabilistic feasibility

Multi-agent - 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
  - 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)

Multi-agent - a priori

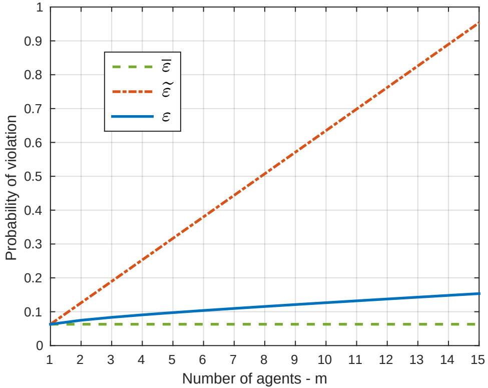
Fix $\beta \in (0,1)$ and $\left\{S_i\right\}_i$. With confidence $\geq 1-\beta$,

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

where

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

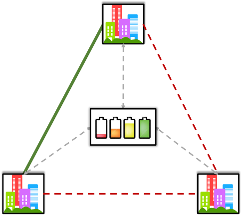
Common vs. different data sets



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

Building network problem revisited – Simulation results

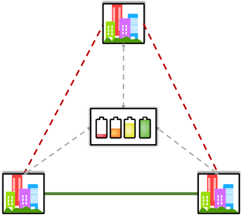
- Set-up
  - 3 buildings - 3 zones each (different chiller per building)
  - Pair-wise communication (gossip)



- Implementation
  - Simulation in MATLAB
  - Optimization solver SEDUMI via the MATLAB interface YALMIP

Building network problem revisited – Simulation results

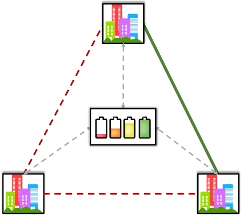
- Set-up
  - 3 buildings - 3 zones each (different chiller per building)
  - Pair-wise communication (gossip)



- Implementation
  - Simulation in MATLAB
  - Optimization solver SEDUMI via the MATLAB interface YALMIP

Building network problem revisited – Simulation results

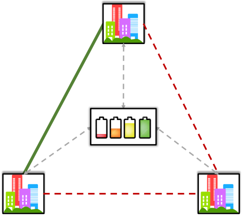
- Set-up
  - 3 buildings - 3 zones each (different chiller per building)
  - Pair-wise communication (gossip)



- Implementation
  - Simulation in MATLAB
  - Optimization solver SEDUMI via the MATLAB interface YALMIP

Building network problem revisited – Simulation results

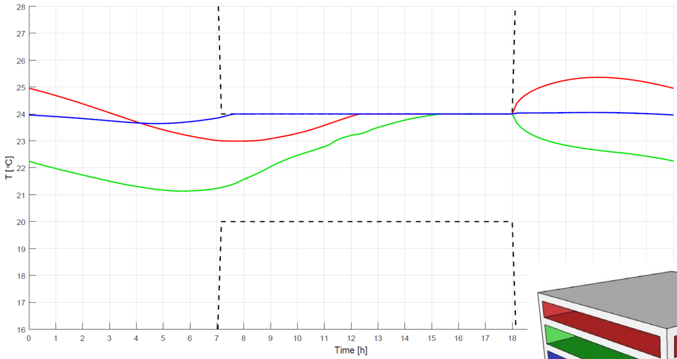
- Set-up
  - 3 buildings - 3 zones each (different chiller per building)
  - Pair-wise communication (gossip)

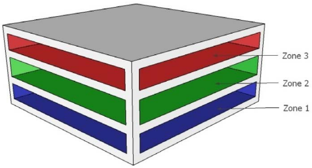


- Implementation
  - Simulation in MATLAB
  - Optimization solver SEDUMI via the MATLAB interface YALMIP

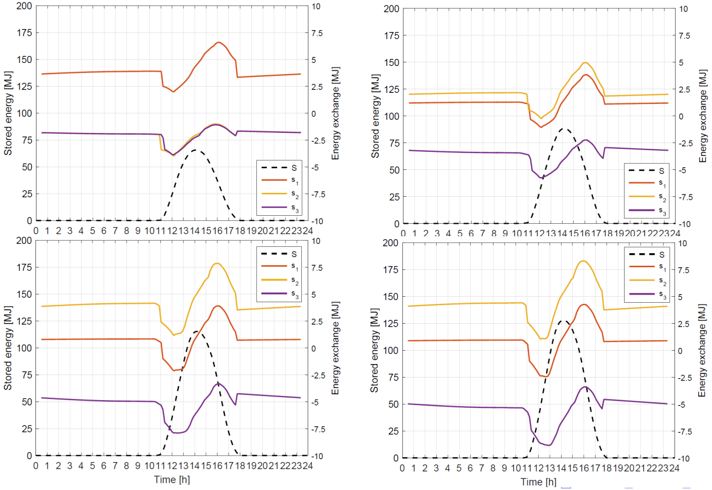
Simulation results

- Temperature evolution in 24h for each building zone





- Energy exchange with the storage
  - The low-chiller building ("yellow") uses the storage charged by the others



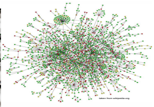
Outline

1. An abstract learning setting
   - Learning from data
   - Generalizing learned decisions
2. Data based optimization
   - Optimal decisions with robustness certificates
   - *Example:* Power system dispatch problems under uncertainty
3. Distributed learning in networks
   - Distributed algorithms – the deterministic case
   - Distributed algorithms – the uncertain case
   - *Example:* Energy management in buildings
4. Summary & ... a glimpse on electric vehicle charging games!

Summary & Future work

- Making decisions in networks



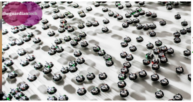



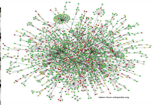
  - General framework for learning from data
  - Generalizing learned decisions – probabilistic feasibility certificates
  - Multiple applications: Dispatch problems; Building control

Summary & Future work

- Making decisions in networks



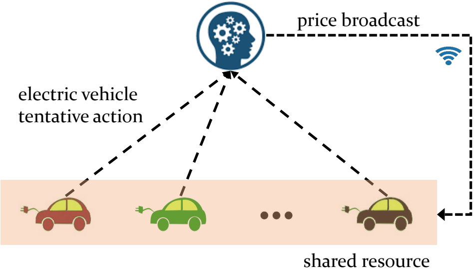




  - General framework for learning from data
  - Generalizing learned decisions – probabilistic feasibility certificates
  - Multiple applications: Dispatch problems; Building control

- What comes next?
  - Convergence rate analysis
  - Rolling horizon implementations
  - Optimality confidence intervals for distributed scenario approach
  - More applications – Charging control of EVs; many others

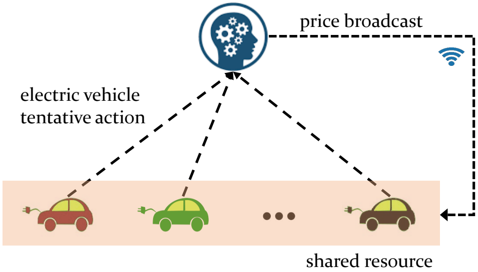
Electric vehicle charging control[7]



- Problem set-up
  1. Aggregator sends a signal incentive to vehicles, e.g. price
  2. Each vehicle responds to the price by solving a local problem to determine consumption
  3. Each vehicle broadcasts updated consumption to aggregator
  4. Process is repeated

[7] Deori, Margellos, Prandini (under review), 2017

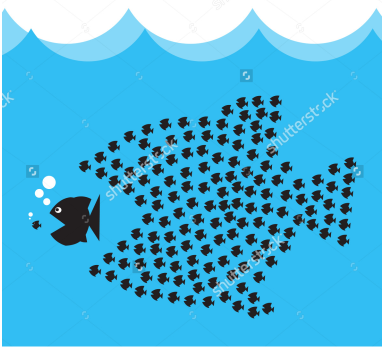
Electric vehicle charging control



- Two cases
  - Vehicles cooperate $\Rightarrow$ social welfare optimum
  - Vehicles act as non-cooperative entities $\Rightarrow$ multi-agent game
  - How do these differ?

From non-cooperative to cooperative ...

- What if vehicles were "anarchists", interested in individual objectives?
- What is the "price of anarchy"[8], i.e.
  relative difference between social welfare optimum and Nash value?
- One can't do much ... but if it is many of them

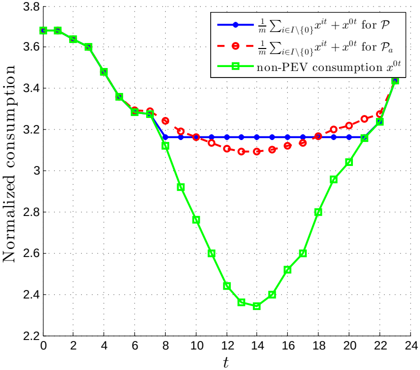


[8]Koutsoupias & Papadimitriou, 1999

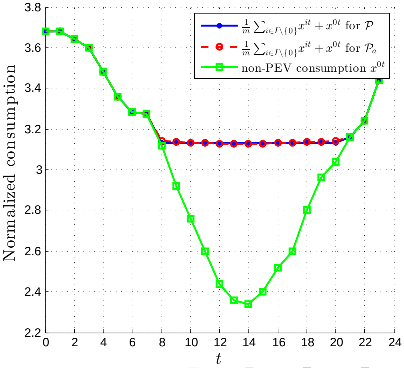
From non-cooperative to cooperative ...

- As agents tend to infinity
  $\Rightarrow$ Social welfare optimum $\rightarrow$ Value of the game (with probability 1)
- There is no "price of anarchy"!

- 5 vehicles



- 100 vehicles



Thank you for your attention!
Questions?

Contact at:
kostas.margellos@eng.ox.ac.uk

References

Vidyasagar (1997)
Learning and Generalization
Springer-Verlag.

Floyd & Warmuth (1995)
Sample compression, learnability, and the Vapnik-Chervonenkis dimension.
Machine Learning, 1-36.

Calafiore & Campi (2006)
The scenario approach to robust control design
IEEE Transactions on Automatic Control, 51(5), 742-753.

Campi & Garatti (2008)
19(3):1211-1230
SIAM Journal on Optimization, 19(3), 1211-1230.

Bertsimas & Sim (2006)
Tractable approximations to robust conic optimization problems
Mathematical Programming, 107, 5-36.

Margellos, Prandini & Lygeros (2015)
On the connection between compression learning and scenario based single-stage and cascading optimization problems
IEEE Transactions on Automatic Control, 60(10), 2716-2721.

Margellos, Goulart & Lygeros (2014)
On the road between robust optimization and the scenario approach for chance constrained optimization problems
IEEE Transactions on Automatic Control, 59(8), 2258-2263.

Margellos & Oren (2016)
Capacity controlled demand side management: A stochastic pricing analysis
IEEE Transactions on Power Systems, 31(1), 706-717.

References

Bertsekas & Tsitsiklis (1989)
Parallel and distributed computation: Numerical methods
Athena Scientific.

Bertsekas (2011)
Incremental proximal methods for large scale convex optimization
Mathematical Programming, 129, 163-195.

Nedich, Ozdaglar & Parrilo (2010)
Constrained consensus and optimization in multi-agent networks
IEEE Transactions on Automatic Control, 55(4), 922-938.

Koutsoupias & Papadimitriou (1999)
Worst-case equilibria
Springer-Verlag, 404-413.

Campi, Garatti & Ramponi (2015)
Non-convex scenario optimization with application to system identification
IEEE Conference on Decision and Control, 4023-4028.

Margellos, Falsone, Garatti & Prandini (2016)
Distributed constrained optimization and consensus in uncertain networks via proximal minimization
IEEE Transactions on Automatic Control, under review, 1-15.

Falsone, Margellos, Garatti & Prandini (2016)
Dual decomposition and proximal minimization for multi-agent distributed optimization with coupling constraints
Automatica, under review, 1-14.

Deori, Margellos, & Prandini (2016)
Nash equilibria in electric vehicle charging control games: Decentralized computation and connection with social optima
Automatica, under review, 1-13.