Security of distributed Model Predictive Control under False Data Injection

Rafael Accácio NOGUEIRA
rafael.accacio.nogueira@gmail.com

Seminar
École Centrale de Lyon / Laboratoire Ampère
25/05/2023 @ Écully

https://bit.ly/43h2jms

About me

Rafael Accácio Nogueira

Postdoctoral researcher at LAAS/CNRS

Garanteed relative localisation and anticollision scenario for autonomous vehicles

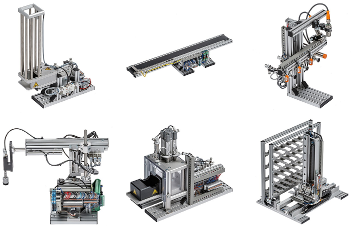
Project AutOCampus (GIS neOCampus)

Advised by Soheib Fergani

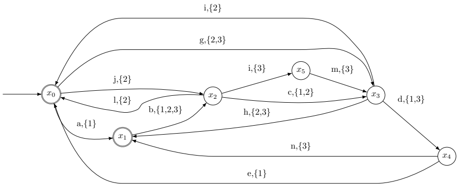
Bachelor Thesis at Escola Politécnica/UFRJ

Identification of DES for fault-diagnosis

Advised by Marcos Vicente de Brito Moreira



I/O
+
DAOCT

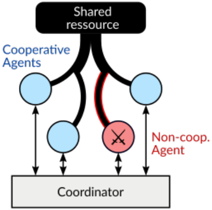


About me

Doctoral Thesis at CentraleSupélec/IETR

Security of dMPC under False Data Injection

Advised by Hervé Guéguen and Romain Bourdais



Context

Smart(er) Cities

Multiple systems interacting

Context

Smart(er) Cities

Multiple systems interacting

- Distribution:
  - Electricity
  - Heat
  - Water
- Traffic

...

Context

Smart(er) Cities

Multiple systems interacting under

- Technical/Comfort Constraints
- We also want
  - Minimize consumption
  - Maximizer satisfaction
  - Follow a trajectory
- Solution → MPC

Brief recap

Find optimal control sequence using predictions based on a model.

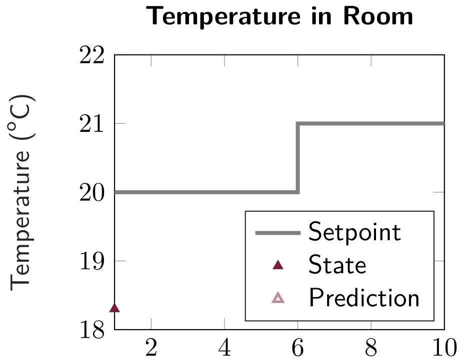
- We need an optimization problem
  - Decision variable is the control sequence calculated over horizon N
  - Objective function to optimize
  - System's Model
  - Other constraints to respect (QoS, technical restrictions, ...)

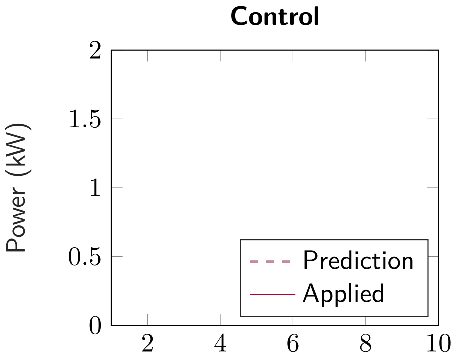
$$
\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
\text{subject to} \quad & \left.\begin{aligned}
\boldsymbol{x}[\xi|k] = f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\
g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \leqslant 0 \\
h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) = 0
\end{aligned}\right\} \begin{aligned}
&\forall \xi \in \{1, \dots, N\} \\
&\forall i \in \{1, \dots, m\} \\
&\forall j \in \{1, \dots, p\}
\end{aligned}
\end{aligned}
$$

Model Predictive Control

In a nutshell

Find optimal control sequence, apply first element, rinse repeat → Receding Horizon

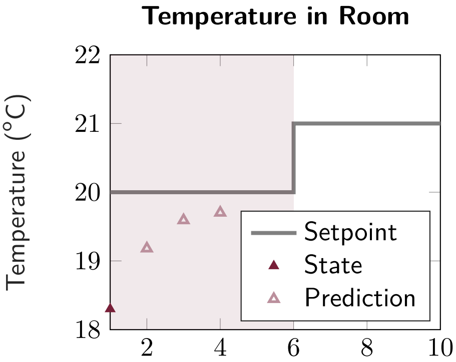


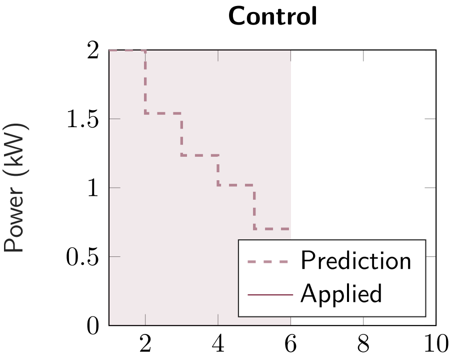


Model Predictive Control

In a nutshell

Find optimal control sequence, apply first element, rinse repeat → Receding Horizon

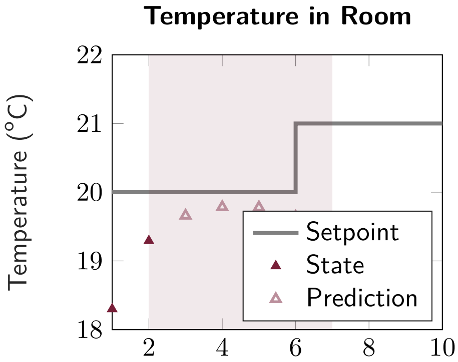


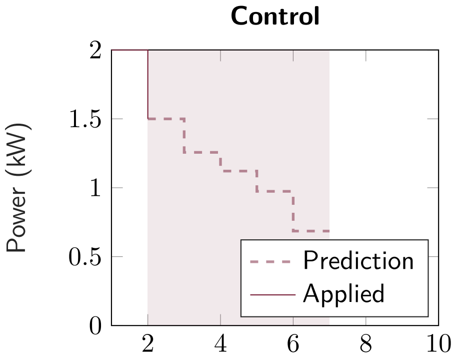


Model Predictive Control

In a nutshell

Find optimal control sequence, apply first element, rinse repeat → Receding Horizon

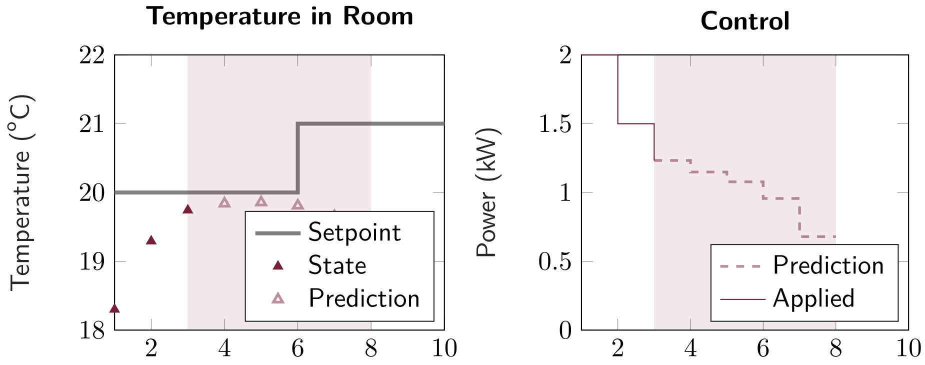




Model Predictive Control

In a nutshell

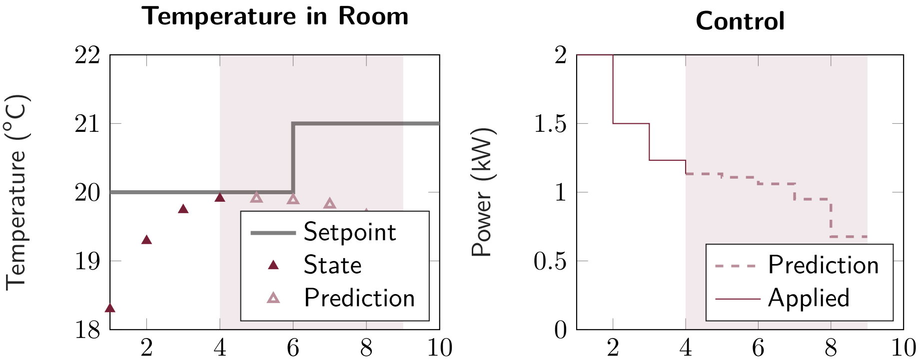
Find optimal control sequence, apply first element, rinse repeat → Receding Horizon



Model Predictive Control

In a nutshell

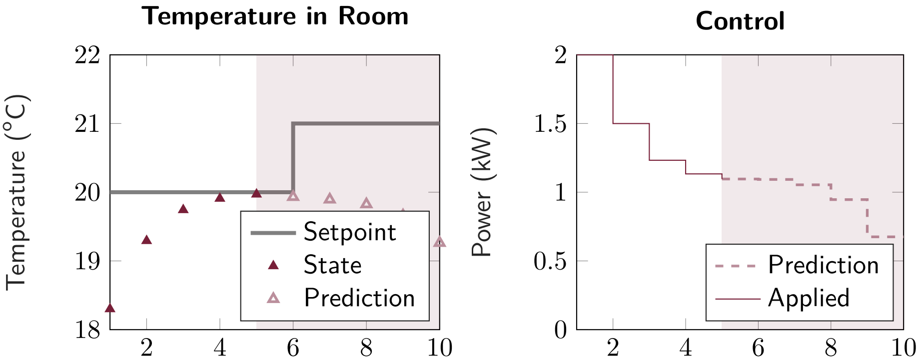
Find optimal control sequence, apply first element, rinse repeat → Receding Horizon



Model Predictive Control

In a nutshell

Find optimal control sequence, apply first element, rinse repeat → Receding Horizon



Model Predictive Control

Nothing is perfect

- Issues
  - Topology
  - Complexity of calculation
  - Flexibility (Add/remove parts)
  - Privacy (RGPD)
- Solution: distributed MPC

Objective

Study security in dMPC context

Security in dMPC context is relatively new[1] (First article from 2017[2])

- CentraleSupélec Rennes - Expertise in MPC for Smart Buildings
- Brittany Region (Sustainable Energy + cybersecurity)

- How fragile are dMPC structures?
- How can agents act non-cooperatively?
- How to identify such agents and mitigate the effects?

[1] <30 documents in scopus

[2] Velarde, Jose Maria Maestre, H. Ishii, et al., "Vulnerabilities in Lagrange-Based DMPC in the Context of Cyber-Security"

Outline

Distributed Model Predictive Control

- We break the MPC optimization problem
- Make agents communicate

In other words

- Agents solve local problems
- Exchange some variables
- Variables are updated

} Until Convergence

Remark

If agents exchange same variable → consensus problem

Distributed Model Predictive Control

Optimization Frameworks

Usually based on optimization decomposition methods[3]:

- Local problems with auxiliary variables
- Update auxiliary variables

Basically 2 choices[4]:

- Modify based on dual problem[5] (Solve with dual and send primal)
- Modify based on primal problem (Solve with primal and send dual)

Security/privacy properties

Many methods:

- Cutting plane, sub-gradient methods, . . .

---

[3] Boyd et al., "Notes on Decomposition Methods"

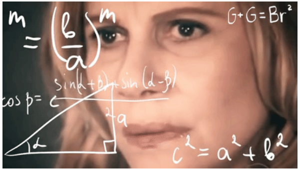
[4] Other approaches, but similar concepts

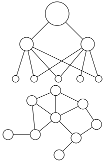
[5] Lagrangian, ADMM, prices, etc +1000 articles in scopus

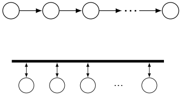
Distributed Model Predictive Control

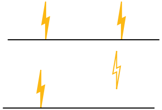
It is about communication

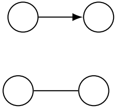
- We break the MPC optimization problem
- Make agents communicate. But how?
  - Many flavors to choose from[6]
    - Hierarchical/Anarchical
    - Parallel/Sequential
    - Synchronous/Asynchronous
    - Bidirectional/Unidirectional
    - ...











[6] José M Maestre, Negenborn, et al., Distributed Model Predictive Control made easy

Distributed Model Predictive Control

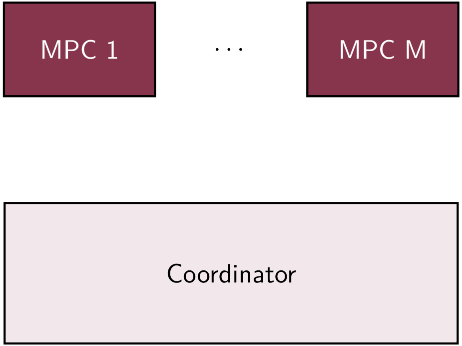
Optimization Decomposition

Distributed Model Predictive Control

Optimization Decomposition

Distributed Model Predictive Control

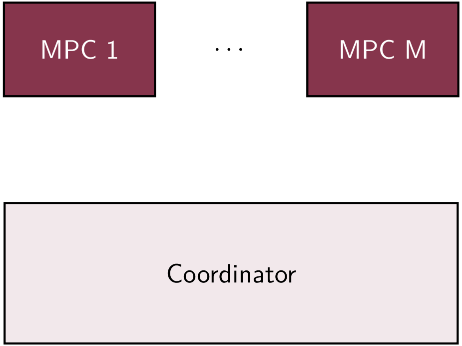
Optimization Decomposition



- Coordinator

Distributed Model Predictive Control

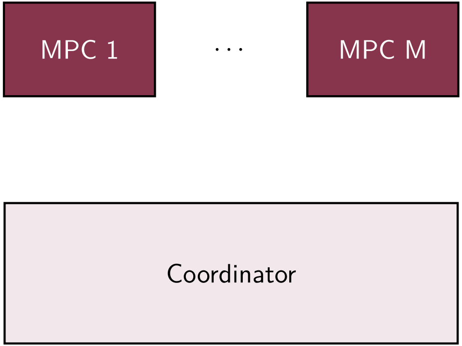
Optimization Decomposition



- Coordinator
  - Enforce global constraints

Distributed Model Predictive Control

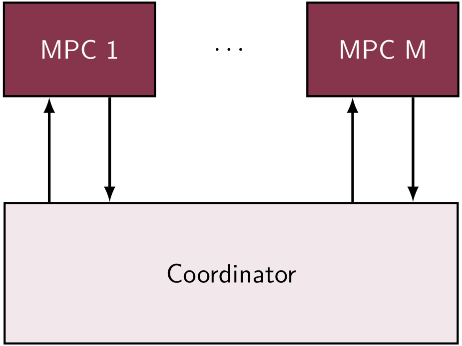
Optimization Decomposition



- Coordinator → Hierarchical
  - Enforce global constraints

Distributed Model Predictive Control

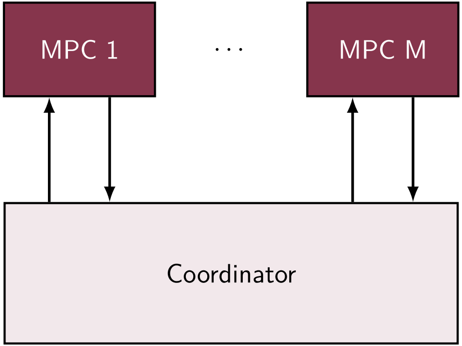
Optimization Decomposition



- Coordinator → Hierarchical
  - Enforce global constraints
- Bidirectional

Distributed Model Predictive Control

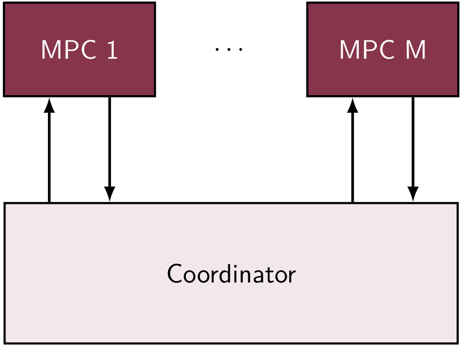
Optimization Decomposition



- Coordinator → Hierarchical
  - Enforce global constraints
- Bidirectional
- No delay → Synchronous

Distributed Model Predictive Control

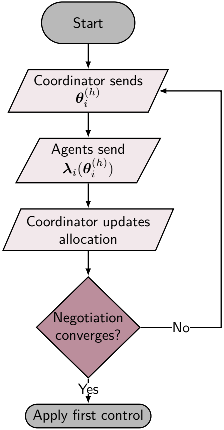
Optimization Decomposition



- Coordinator → Hierarchical
  - Enforce global constraints
- Bidirectional
- No delay → Synchronous
- But what to send?

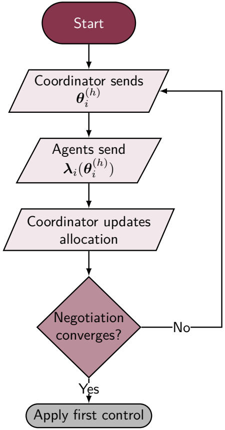
Primal Decomposition

or Quantity Decomposition | or Resource Allocation



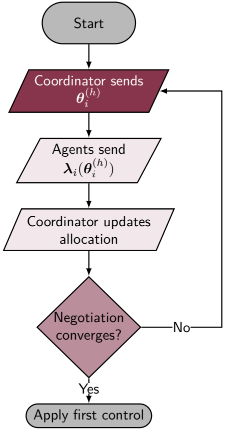
Primal Decomposition

or Quantity Decomposition | or Resource Allocation



Primal Decomposition

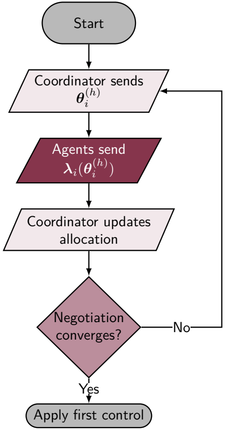
or Quantity Decomposition | or Resource Allocation



Allocation $\boldsymbol{\theta}_i$

Primal Decomposition

or Quantity Decomposition | or Resource Allocation

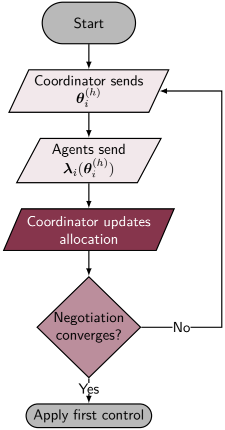


Allocation $\boldsymbol{\theta}_i$

Dissatisfaction $\boldsymbol{\lambda}_i$

Primal Decomposition

or Quantity Decomposition | or Resource Allocation



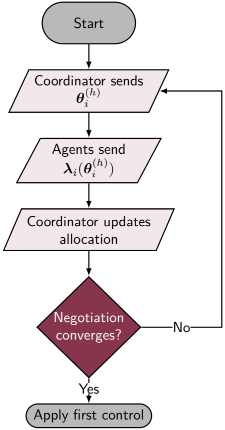
Allocation $\boldsymbol{\theta}_i$

Dissatisfaction $\boldsymbol{\lambda}_i$

Update $\boldsymbol{\theta}_i^+ = f_i(\boldsymbol{\theta}_i, \boldsymbol{\lambda}_i)$

Primal Decomposition

or Quantity Decomposition | or Resource Allocation



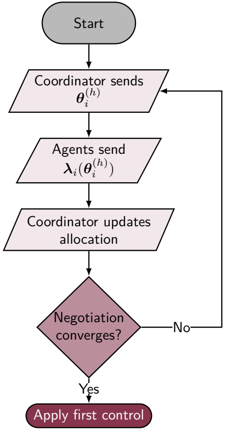
Allocation $\boldsymbol{\theta}_i$

Dissatisfaction $\boldsymbol{\lambda}_i$

Update $\boldsymbol{\theta}_i^+ = f_i(\boldsymbol{\theta}_i, \boldsymbol{\lambda}_i)$

Primal Decomposition

or Quantity Decomposition | or Resource Allocation



Allocation $\boldsymbol{\theta}_i$

Dissatisfaction $\boldsymbol{\lambda}_i$

Update $\boldsymbol{\theta}_i^+ = f_i(\boldsymbol{\theta}_i, \boldsymbol{\lambda}_i)$

Primal Decomposition

In detail

1. Allocate $\boldsymbol{\theta}_i$ for each agent
2. They solve local problems and
3. Send dual variable $\boldsymbol{\lambda}_i$[7]
4. Allocation is updated[8] (respect global constraint)

$$\begin{array}{ll} \underset{\boldsymbol{u}_1,\ldots,\boldsymbol{u}_M}{\text{minimize}} & \sum_{i\in\mathcal{M}} J_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \\ \text{s.t.} & \sum_{i\in\mathcal{M}} \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \preceq \boldsymbol{u}_{\text{total}} \end{array}$$

$\downarrow$ For each $i \in \mathcal{M}$

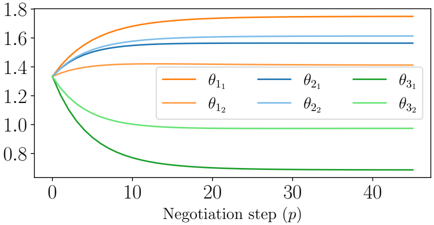
$$\begin{array}{ll} \underset{\boldsymbol{u}_i}{\text{minimize}} & J_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \\ \text{s. t.} & \boldsymbol{h}_i(\boldsymbol{x}_i,\boldsymbol{u}_i) \preceq \boldsymbol{\theta}_i : \boldsymbol{\lambda}_i \end{array}$$

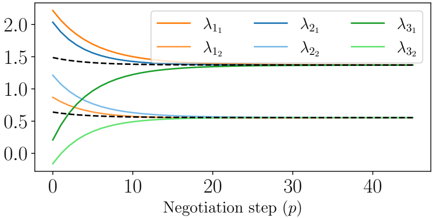
$$\boldsymbol{\theta}[k]^{(p+1)} = \text{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

[8] It obfuscates system's parameters (+ Privacy)
[8] Only equation to change to add/remove agents

Example

Until everybody is evenly[9] dissatisfied





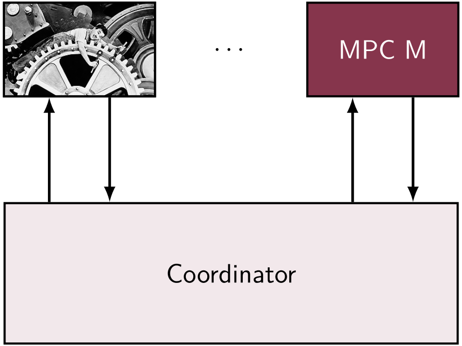
[9] For inequality constraints dynamics are more complex

Distributed Model Predictive Control

Negotiation works if agents comply.
But what if some agents are ill-intentioned and attack the system?

How can a non-cooperative agent attack?

Literature



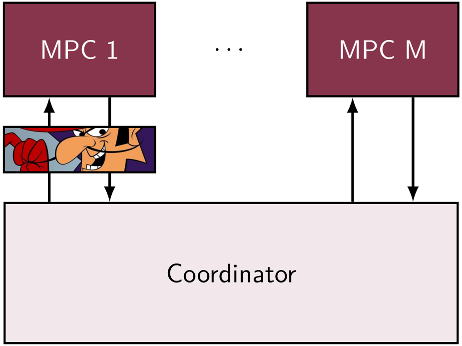
- Common attacks[10]
  - Fake objective function
  - Fake constraints
  - Use different control

  } Deception Attacks

[10] Velarde, Jose Maria Maestre, Hideaki Ishii, et al., "Scenario-based defense mechanism for distributed model predictive control"

How can a non-cooperative agent attack?

Our approach[11]

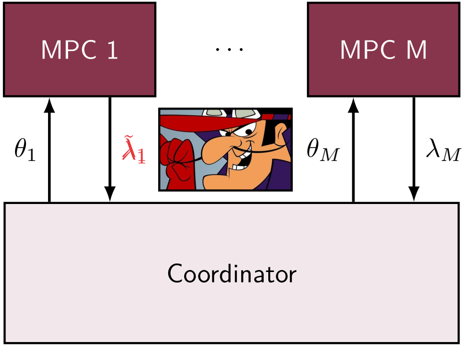


- Primal decomposition
  - Maximum resources fixed
- We are in coordinator's shoes
- What matters is the interface
  - Attacker changes communication
    - False Data Injection

[11] Nogueira, Bourdais, and Guéguen, "Detection and Mitigation of Corrupted Information in Distributed Model Predictive Control Based on Resource Allocation"

How can a non-cooperative agent attack?

Our approach[11]



- $\boldsymbol{\lambda}_i$ is the only interface
- Malicious agent modifies $\boldsymbol{\lambda}_i$

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i)$$

[11] Nogueira, Bourdais, and Guéguen, "Detection and Mitigation of Corrupted Information in Distributed Model Predictive Control Based on Resource Allocation"

Attack model

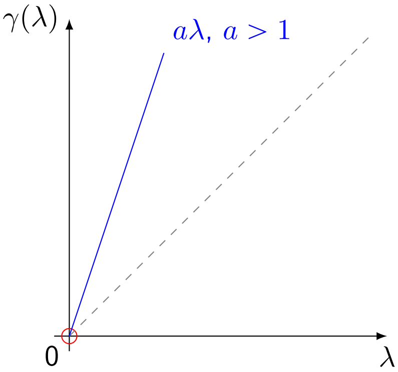
Liar, Liar, Pants of fire

- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction

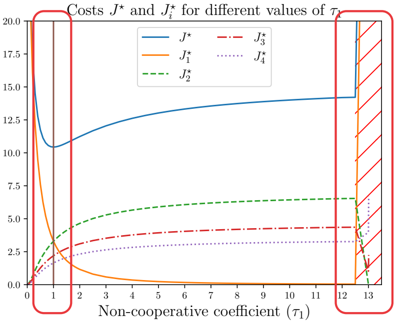
Assumptions

- *Same attack during negotiation*
- *Attacker satisfied only if it really is*
  - $\gamma(\lambda) = 0 \rightarrow \lambda = 0$
- $\tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$

- Attack is invertible $\rightarrow \exists T_i[k]^{-1}$



Example



4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$
- Simulate for different τ_1 get J_i

- We can observe 3 things
  - Global minimum when $\tau_1 = 1$
  - Agent 1 benefits if τ_1 increases (inverse otherwise)
  - All collapses if too greedy

- But can we mitigate these effects?
- Yes! (At least in some cases)

Classification of mitigation techniques

Passive (Robust)

- 1 mode

Active (Resilient)

- 2 modes
  1. Attack free
  2. When attack is detected
     - Detection/Isolation
     - Mitigation

State of art

Security dMPC

	Decomposition	Resilient/Robust	Detection	Mitigation
[12]	Dual	Robust (Scenario)	NA	NA
[13]	Dual	Robust (f-robust)	NA	NA
[14]	Jacobi-Gauß	–	–	–
[15]	Dual	Resilient	Analyt./Learn.	Disconnect (Robustness)

[12] José M. Maestre et al., "Scenario-Based Defense Mechanism Against Vulnerabilities in Lagrange-Based Dmpc".

[13] Velarde, José M. Maestre, et al., "Vulnerabilities in Lagrange-Based Distributed Model Predictive Control".

[14] Chanfreut, J. M. Maestre, and H. Ishii, "Vulnerabilities in Distributed Model Predictive Control based on Jacobi-Gauss Decomposition".

[15] Ananduta et al., "Resilient Distributed Model Predictive Control for Energy Management of Interconnected Microgrids".

Our Approach

Explore Scarcity

- Resilient
- Analytical/Learning
- Data reconstruction

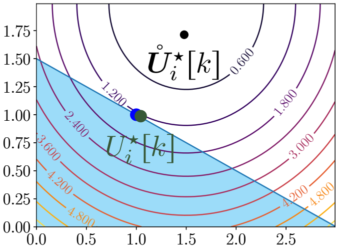
Parameter Estimation

- Explore Scarcity

What are deprived systems?

Systems whose optimal solution has all constraints active

- Unconstrained Solution $\mathring{\boldsymbol{U}}_i^\star[k]$
- $h_i(\mathring{\boldsymbol{U}}_i^\star[k]) > \boldsymbol{\theta}_i[k] \rightarrow$ Scarce resources
  - Solution projected onto boundary
  - Same as with equality constraints[16]



$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] \leq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array} \longrightarrow \begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

[16] If system can have all constraints active simultaneously see here

Analyzing Deprived Systems

Assumptions

- *Quadratic local problems*
- *Linear inequality constraints*
- *Scarcity*

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|^2_{H_i} + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] = \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

- Solution is analytical and affine

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

(local parameters unknown by coordinator) $\left\{ \begin{array}{l} \bullet\ P_i \text{ is time invariant} \\ \bullet\ \boldsymbol{s}_i[k] \text{ is time variant} \end{array} \right.$

Deprived Systems

Under attack!

- Normal behavior
  - Affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

- Under attack $\rightarrow \tilde{\boldsymbol{\lambda}}_i = T_i[k]\boldsymbol{\lambda}_i$
  - Parameters modified

$$\tilde{\boldsymbol{\lambda}}_i[k] = -\tilde{P}_i[k]\boldsymbol{\theta}_i[k] - \tilde{\boldsymbol{s}}_i[k]$$

- But wait! P_i is not supposed to change!
- Change $\rightarrow$ Probably an Attack! Let's take advantage of this!

Detection Mechanism

- We estimate[17] $\hat{P}_i[k]$ and $\widehat{\tilde{\boldsymbol{s}}}_i[k]$ such as:

$$\tilde{\boldsymbol{\lambda}}_i[k] = -\widehat{\tilde{P}}_i[k]\boldsymbol{\theta}_i - \widehat{\tilde{\boldsymbol{s}}}_i[k]$$

Assumption

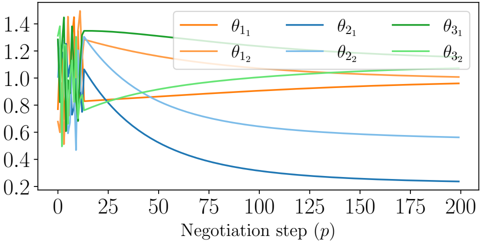
We can estimate $\bar{P}_i$ from a attack free negotiation

- If $\left\|\widehat{\tilde{P}}_i[k] - \bar{P}_i\right\|_F > \epsilon_P \rightarrow$ Attack
- Ok, but how can we estimate $\widehat{\tilde{P}}_i[k]$?

[17] Using Recursive Least Squares for example

Estimating $\widehat{\tilde{P}}_i[k]$

- We estimate $\widehat{\tilde{P}}_i[k]$ and $\widehat{\tilde{\boldsymbol{s}}}_i[k]$ simultaneously using RLS
- Challenge: Online estimation during negotiation fails
  - Update function couples $\boldsymbol{\theta}_i^p$ and $\boldsymbol{\lambda}_i^p$ $\rightarrow$ low input excitation
- Solution: Send a random[18] sequence to increase excitation until convergence.



[18] A random signal causes persistent excitation of any order (Adaptive Control)

Classification of mitigation techniques

- Active (Resilient)
  1. Detection/Isolation ✔
  2. Mitigation ?

Mitigation mechanism

Reconstructing $\boldsymbol{\lambda}_i$

- Now, we have $\widehat{\tilde{P}}_i[k]$
  - Since $\tilde{P}_i[k] = T_i[k]\bar{P}_i$
  - We can recover $T_i[k]^{-1}$

$$\widehat{T_i[k]^{-1}} = P_i \widehat{\tilde{P}}_i[k]^{-1}$$

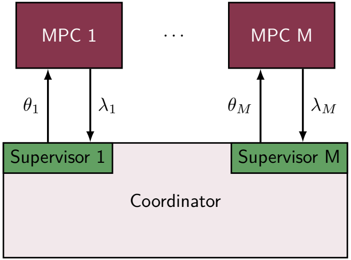
- Reconstruct $\boldsymbol{\lambda}_i$

$$\overset{\text{rec}}{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i[k]^{-1}}\widehat{\tilde{\boldsymbol{s}}}_i[k]$$

- Choose adequate version for coordination

$$\overset{\text{mod}}{\boldsymbol{\lambda}}_i = \begin{cases} \overset{\text{rec}}{\boldsymbol{\lambda}}_i, & \text{if attack detected} \\ \tilde{\boldsymbol{\lambda}}_i, & \text{otherwise} \end{cases}$$

Complete Mechanism



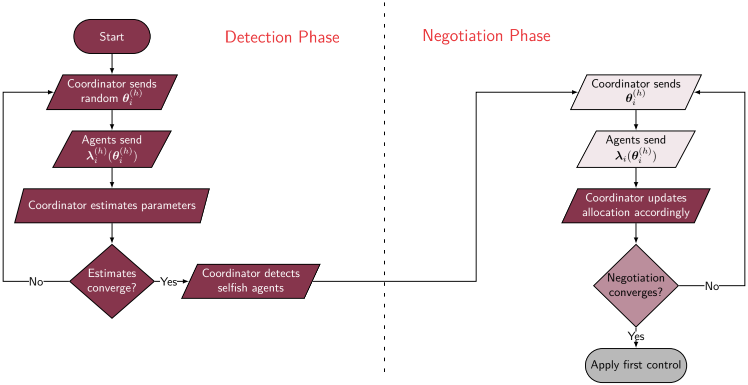
- Supervise exchanges by inquiring the agents
- Estimate how they will behave

Two Phases

1. Detect which agents are non-cooperative
2. Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

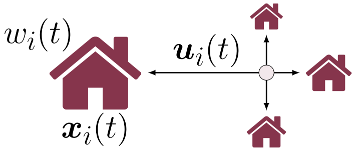
Complete algorithm

RPdMPC-DS[19]



[19] Nogueira, Bourdais, and Guéguen, "Detection and Mitigation of Corrupted Information in Distributed Model Predictive Control Based on Resource Allocation".

Example

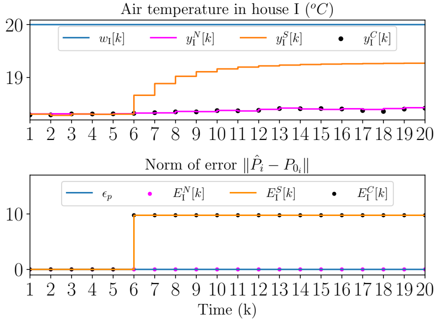


District Heating Network (4 Houses)

- Houses modeled using 3R-2C (monozone)
- Not enough power
- Period of 5h ($T_s = 0.25h \rightarrow k = \{1:20\}$)
- Prediction horizon ($N = 4$)
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-DS)

Results

Temporal



Temperature in house I.
Error $E_I(k)$.
N Nominal, S Selfish, C Corrected

- Agent starts cheating in $k = 6$
- S Agent increases its comfort
- C Restablish behavior close to N

Results

Costs

Objective functions J_i (Normalized error %)

Agent	Selfish	Corrected
I	-36.3	0.5
II	21.67	-0.55
III	17.39	-0.0
IV	17.63	-0.09
Global	3.53	0.02

Relaxing scarcity assumption

- Systems are not completely deprived
  - We can't change our constraints to equality ones anymore
  - Nor use the simpler update equation

$$\begin{array}{ll} \underset{\boldsymbol{U}_i[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|_{H_i}^2 + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k] \\ \text{subject to} & \bar{\Gamma}_i\boldsymbol{U}_i[k] \preceq \boldsymbol{\theta}_i[k] : \boldsymbol{\lambda}_i[k] \end{array}$$

$$\boldsymbol{\theta}[k]^{(p+1)} = \mathrm{Proj}^{\mathcal{S}}(\boldsymbol{\theta}[k]^{(p)} + \rho^{(p)}\boldsymbol{\lambda}[k]^{(p)})$$

Analyzing System

Solution for $\lambda_i[k]$

Instead of having one single affine solution

$$\boldsymbol{\lambda}_i[k] = -P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$$

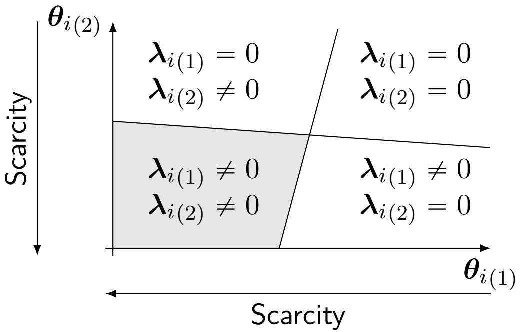
Now, we may have multiple (Piecewise affine function)

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^0_{\boldsymbol{\lambda}_i} \\ \vdots & \vdots \\ -P_i^{(Z)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(Z)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^Z_{\boldsymbol{\lambda}_i} \end{cases}$$

Still the $P_i^{(z)}$ are time independent

Analyzing System

Solution for $\lambda_i[k]$ (Continued)



Separation surfaces depend on state and local parameters.
Unknown by the coordinator.

Analyzing System

Solution for $\lambda_i[k]$ (Continued) Still?

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^0_{\boldsymbol{\lambda}_i} \\ \vdots & \vdots \\ -P_i^{(Z)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(Z)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^Z_{\boldsymbol{\lambda}_i} \end{cases}$$

Scarcity

All constraints active	$-P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k]$	$\rightarrow$	$-P_i\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i[k]$
None constraints active	$-P_i^{(Z)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(Z)}[k]$	$\rightarrow$	$\mathbf{0}$

Assumptions

The region $\mathcal{R}^0_{\boldsymbol{\lambda}_i} \neq \emptyset$ and we known a point $\overset{\varnothing}{\boldsymbol{\theta}}_i \in \mathcal{R}^0_{\boldsymbol{\lambda}_i}$

Analyzing System

Under attack!

$$\tilde{\boldsymbol{\lambda}}_i[k] = T_i[k]\boldsymbol{\lambda}_k$$

Parameters are modified. But not the regions' limits

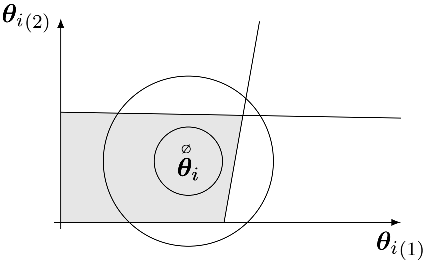
$$\tilde{\boldsymbol{\lambda}}_i[k] = \begin{cases} -\widetilde{P}_i^{(0)}\boldsymbol{\theta}_i[k] - \widetilde{\boldsymbol{s}}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^0 \\ \vdots & \vdots \\ -\widetilde{P}_i^{(Z)}\boldsymbol{\theta}_i[k] - \widetilde{\boldsymbol{s}}_i^{(Z)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}^Z_{\boldsymbol{\lambda}_i} \end{cases}$$

- If we can estimate $\widetilde{P}_i^{(0)}$ we can use same strategy than before
- Problem: We don't know in which region $\boldsymbol{\theta}_i$ is
- Solution: Let's force it using Artificial Scarcity

Artificial Scarcity

What you thought was way too much is not enough

- We use the point $\overset{\varnothing}{\boldsymbol{\theta}}_i$, which activates all constraints[20]



- Generate points close to $\overset{\varnothing}{\boldsymbol{\theta}}_i$
- Estimate $\widehat{\widetilde{P}}_i^{(0)}[k]$
- How do we known the radius?
  - Unfortunately we don't.
- How to estimate $\widehat{\widetilde{P}}_i^{(0)}[k]$ nonetheless?
  - Expectation Maximization

[20] If we have local constraints, we suppose this point respects them.

Expectation Maximization

- Iterative method to estimate parameters of multimodal models[21]
- We give multiple observations $\boldsymbol{\theta}_i^o[k]$ and $\tilde{\boldsymbol{\lambda}}_i^o[k]$
- At each step we calculate
  - (E) the probability of each $(\widehat{\widetilde{P}}_i^{(z)}[k], \widehat{\tilde{\boldsymbol{s}}}_i^{(z)}[k])$ having generated each $\tilde{\boldsymbol{\lambda}}_i^o[k]$
  - (M) new estimates $(\widehat{\widetilde{P}}_i^{(z)}[k], \widehat{\tilde{\boldsymbol{s}}}_i^{(z)}[k])$ based on the probabilities
- At the end we have
  1. Parameters with associated region index
  2. Observations with associated region index
- We consult the index associated to $\overset{\varnothing}{\boldsymbol{\theta}}_i$
- We recover the associated parameter, i.e., $\widehat{\widetilde{P}}_i^{(0)}[k]$

[21] Such as our PWA function after using some tricks

Detection and Mitigation

Same same, but different

Assumption

We estimate nominal $\bar{P}_i^{(0)}$ *from attack free negotiation*

- Detection

$$\left\| \widehat{\tilde{P}}_i^{(0)}[k] - \bar{P}_i^{(0)} \right\|_F \geqslant \epsilon_{P_i^{(0)}}$$

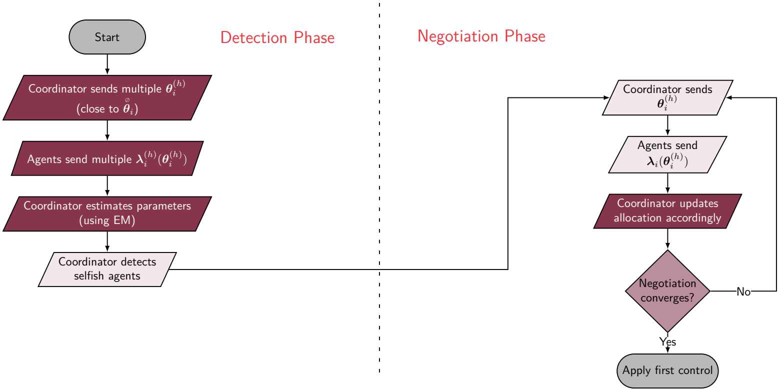
- Mitigation

$$\widehat{T_i[k]^{-1}} = \bar{P}_i^{(0)} \widehat{\tilde{P}}_i^{(0)}[k]^{-1}.$$

$$\overset{\text{rec}}{\boldsymbol{\lambda}}_i = \widehat{T_i[k]^{-1}} \tilde{\boldsymbol{\lambda}}_i.$$

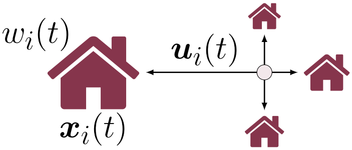
Complete algorithm

RPdMPC-AS[22]



[22] Nogueira, Bourdais, Leglaive, et al., "Expectation-Maximization Based Defense Mechanism for Distributed Model Predictive Control".

Example

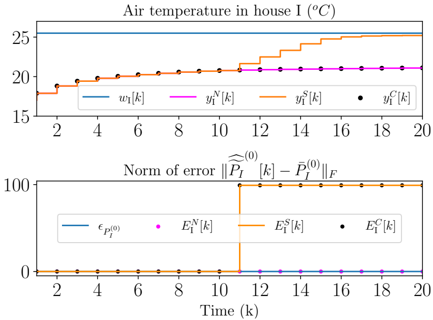


District Heating Network (4 Houses)

- Houses modeled using 3R-2C
- Not enough power~~Not enough power~~ (Change $(\boldsymbol{x}_0, \boldsymbol{w}_0)$)
- Period of 5h $(T_s = 0.25h \rightarrow k = \{1:20\})$
- Prediction horizon $(N = 4)$
- 3 scenarios
  - N Nominal
  - C Agent I cheats (dMPC)
  - S Agent I cheats (RPdMPC-AS)

Results

Temporal



Temperature in house I.
Error $E_I(k)$.
N Nominal, S Selflish C Corrected

Results

Costs

Objective functions J_i (Normalized error %)

Agent	Selfish	Corrected
I	-36.49	$-4.12e-05$
II	35.81	$1.74e-05$
III	29.22	$2.14e-05$
IV	37.54	$1.73e-05$
Global	10.69	$-6e-07$

Too good to be true!

It's a kind of magic!~~It's a kind of magic!~~

- No disturbance in communication
- Unfortunately EM is not magic
  - Slow convergence
  - Dependency on initialization
    - No guarantees of achieving global optimal
- Some "solutions":
  - Force some parameters to converge faster (case dependant)
  - Run multiple times with different initialization and pick best
  - Associate with other methods of the same family

Conclusion

Main takeaways

- Distributed MPC
  - increases privacy and flexibility
  - reduces complexity of calculation
  - in security context, it still is in its baby steps
- Primal decomposition
  - prevents agent to use more resources than agreed upon
  - increases privacy by communicating dual variables instead of primal
- Security for DMPC
  - Attacker can change the communication to receive more ressources.
  - The consequences of an attack are suboptimality and instability
  - We can explore scarcity information to mitigate

Open questions/Future directions

- Reconstruction with partial information (Current work)
- Study of error propagation (Current work)
- Robustness when add noise
- Estimation as Switched Auto-Regressive Exogenous System
- Sensibility to other topologies (more/less vulnerable?)
- Study of security on similar problems (flocking/consensus/averaging/federated learning etc)
- ...

Questions? Comments?

Repository
https://github.com/Accacio/thesis

Contact
rafael.accacio.nogueira@gmail.com

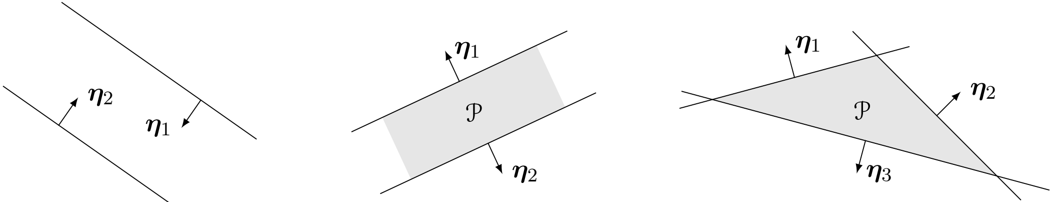
For Further Reading I

- Åström, K.J. and B. Wittenmark. Adaptive Control. Addison-Wesley series in electrical and computer engineering: Control engineering. Addison-Wesley, 1989. ISBN: 9780201097207. DOI: 10.1007/978-3-662-08546-2_24.
- Maestre, José M, Rudy R Negenborn, et al. Distributed Model Predictive Control made easy. Vol. 69. Springer, 2014. ISBN: 978-94-007-7005-8.
- Nogueira, Rafael Accácio. "Security of DMPC under False Data Injection". 2022CSUP0006. PhD thesis. CentraleSupélec, 2022. URL: http://www.theses.fr/2022CSUP0006.

Conditions

back

One way to ensure this, is to make the original constraints to form a cone.



No intersection

$\measuredangle^{\eta_2}_{\eta_1} = 180^o$

A 3-sided polyhedron.

θ dynamics

◂ back

$$\boldsymbol{\theta}^{(p+1)} = \mathcal{A}_\theta \boldsymbol{\theta}^{(p)} + \mathcal{B}_\theta[k]$$

where

$$\mathcal{A}_\theta = \begin{bmatrix} I - \frac{M-1}{M}\rho^{(p)}P_1 & \frac{1}{M}\rho^{(p)}P_2 & \dots & \frac{1}{M}\rho^{(p)}P_M \\ \frac{1}{M}\rho^{(p)}P_1 & I - \frac{M-1}{M}\rho^{(p)}P_2 & \dots & \frac{1}{M}\rho^{(p)}P_M \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{M}\rho^{(p)}P_1 & \frac{1}{M}\rho^{(p)}P_2 & \dots & I - \frac{M-1}{M}\rho^{(p)}P_M \end{bmatrix}$$

$$\mathcal{B}_\theta[k] = \begin{bmatrix} -\frac{M-1}{M}\rho^{(p)}\boldsymbol{s}_1[k] + \frac{1}{M}\rho^{(p)}\boldsymbol{s}_2[k] \dots - \frac{1}{M}\rho^{(p)}\boldsymbol{s}_M[k] \\ \frac{1}{M}\rho^{(p)}\boldsymbol{s}_1[k] - \frac{M-1}{M}\rho^{(p)}\boldsymbol{s}_2[k] \dots - \frac{1}{M}\rho^{(p)}\boldsymbol{s}_M[k] \\ \vdots \\ \frac{1}{M}\rho^{(p)}\boldsymbol{s}_1[k] + \frac{1}{M}\rho^{(p)}\boldsymbol{s}_2[k] \dots - \frac{M-1}{M}\rho^{(p)}\boldsymbol{s}_M[k] \end{bmatrix}$$

Parameters estimated depending on Prediction Horizon N

$\#$ constraints depend on $\#$ global constraints c and prediction horizon N

- Number of Regions $= 2^{Nc}$
- Parameters in each region $=$ Matrix $P_i^{(z)} = (Nc)^2 +$ vector $\boldsymbol{s}_i^{(z)}[k] = Nc$
  - Total $((Nc)^2 + Nc)2^{Nc}$

Some examples

- 1 constraint
  - $N = 3 \rightarrow$ 96 elements
  - $N = 4 \rightarrow$ 320 elements

Remark

We can reduce number of elements estimated from $P_i^{(z)}$ if we assume $P_i^{(z)} \in \mathbb{S}$
New total $\rightarrow ((Nc)^2 + 3Nc)2^{Nc-1}$