Security of distributed Model Predictive Control under False Data injection

or How I Learned to Stop and Worry about Everything

Rafael Accácio NOGUEIRA

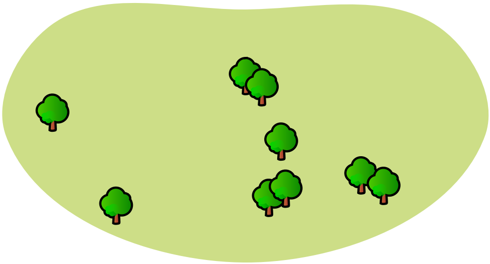
December 12, 2022

https://bit.ly/3g3S6X4

45 minutes !!!!
Good afternoon, thank you all for being here.I'm Rafael Accácio and I'm going to present my work on the security of distributed model predictive control under false data injection.

Context

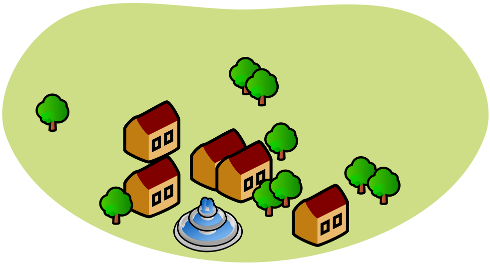
Requirements evolve with time



These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

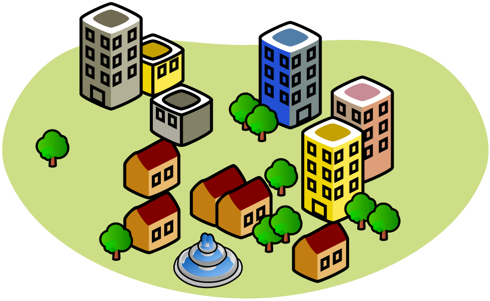
Requirements evolve with time



These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time



These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time

These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

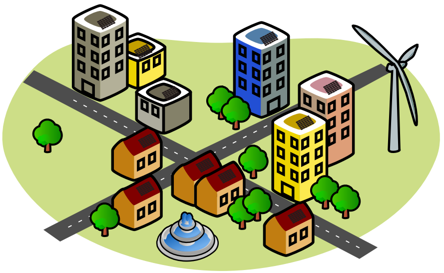
Context

Requirements evolve with time

The systems are usually Geographically distributed These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time



- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management

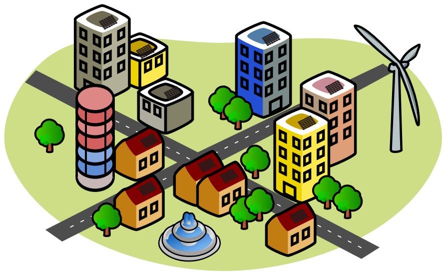
  (include your problem here)

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time



- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management
  (include your problem here)

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Requirements evolve with time

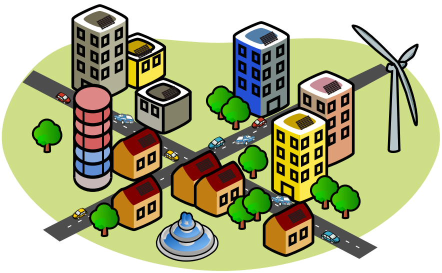
- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management

(include your problem here)

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
A solution is to use MPCThese cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time



- Electricity Distribution System
- Heat distribution
- Water distribution
- Traffic management

  (include your problem here)

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Context

Requirements evolve with time

- Multiple systems interacting
- Coupled by constraints
- Optimization objectives
  - Minimize energy consumption
  - Maximize user satisfaction
  - Follow a trajectory
  - ...
- Solution → MPC

CentraleSupélec

The systems are usually Geographically distributed Coupled by constraints as maximum input power or energy and when they are implemented we try to optimize objectives such as cost, energy, user satisfaction and others.
These cyberphysical systems are the majority of the systems in our everyday lives.We can give example the traffic management,water distribution,electricity distribution, heat and cold and many more.But how to control those kinds of systems. Each has its own Dynamics and constraints, such comfort (Quality of service) or technical. Solution, mpc since we use models and it is easy to integrate the constraints.

Model-based Predictive Control

Find best control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

CentraleSupélec

For those who are not familiar with mpc. Mpc is the model based predictive controller.

Model-based Predictive Control

Find best control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

CentraleSupélec

The objective is to find the best control sequence using predictions based on a model.

Model-based Predictive Control

Find best control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

CentraleSupélec

When we say best,

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

CentraleSupélec

we mean optimal.

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

CentraleSupélec

So we need to solve an optimization problem.

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

CentraleSupélec

And we have the control sequence of u as the decision variable.

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

CentraleSupélec

which is calculated for a horizon N

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

CentraleSupélec

So, we need an objective function. For example follow a trajectory while minimizing the energy.

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

CentraleSupélec

A model of the system

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

with its states

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

CentraleSupélec

and inputs

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

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

CentraleSupélec

But we can also integrate some constraints, such QoS or technical restrictions

Model-based Predictive Control

Find optimal control sequence using predictions based on a model.

- Objective function to optimize
- System's Model (states and inputs)
- Other constraints to respect (QoS, technical restrictions, ...)

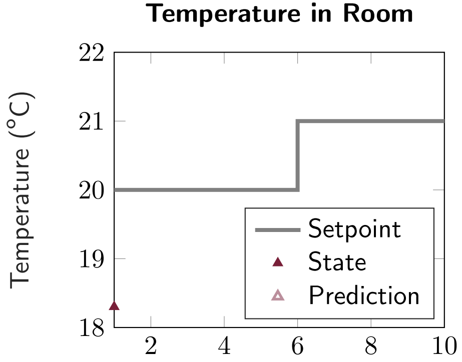
$$
\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & J(\boldsymbol{x}[0|k], \boldsymbol{u}[0:N-1|k]) \\
\text{subject to} \quad & \left.\begin{aligned}
\boldsymbol{x}[\xi|k] &= f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\
g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k])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

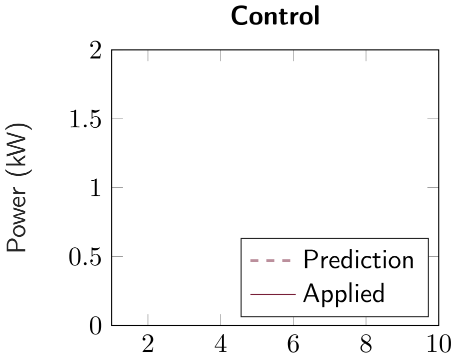
CentraleSupélec

But we can also integrate some constraints, such QoS or technical restrictions

In a nutshell

Find optimal control sequence

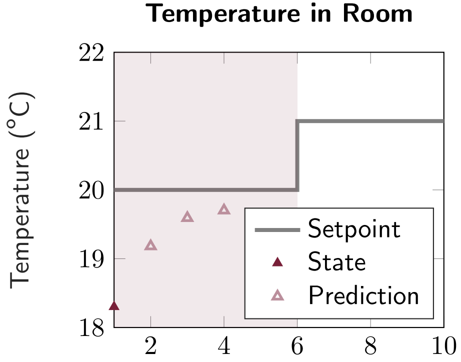


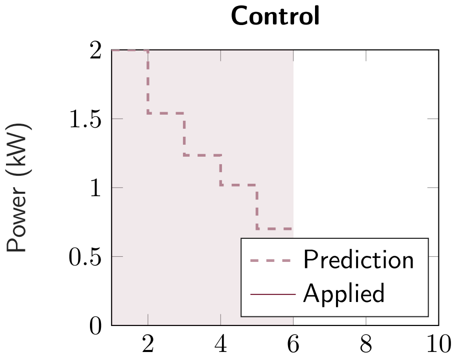


CentraleSupélec

So, for example, if we may have a setpoint to follow

Find optimal control sequence





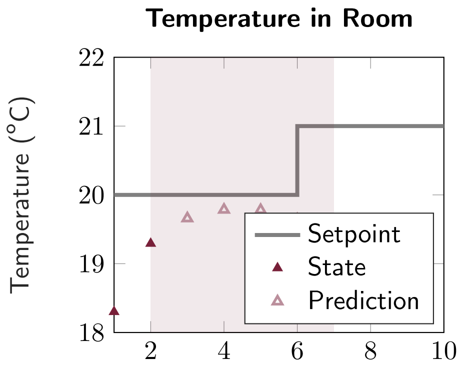
CentraleSupélec

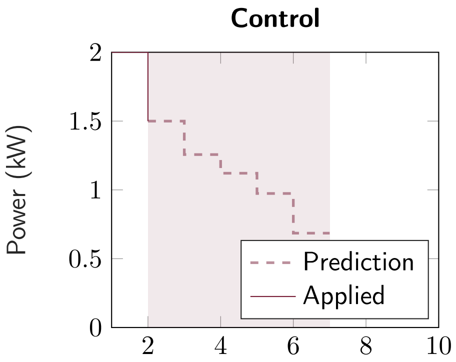
We find an optimal control sequence

Model Predictive Control

In a nutshell

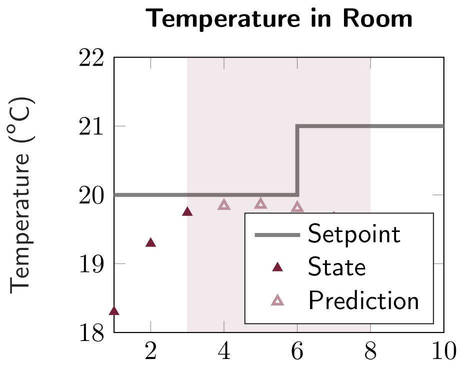
Find optimal control sequence, apply first element

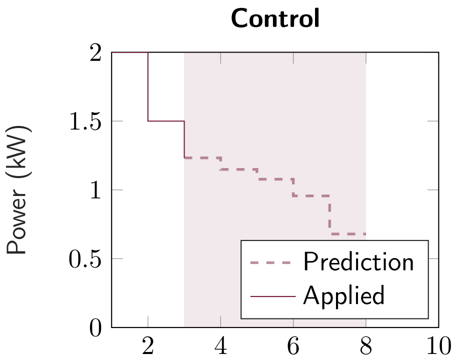




We apply only the first element

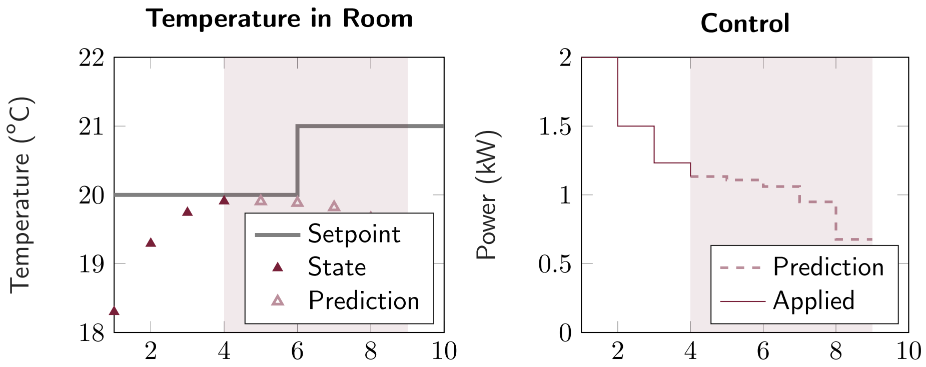
Find optimal control sequence, apply first element, rinse repeat





and then we repeat

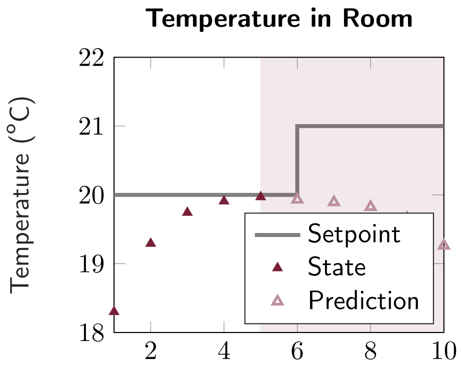
Find optimal control sequence, apply first element, rinse repeat → Receding Horizon

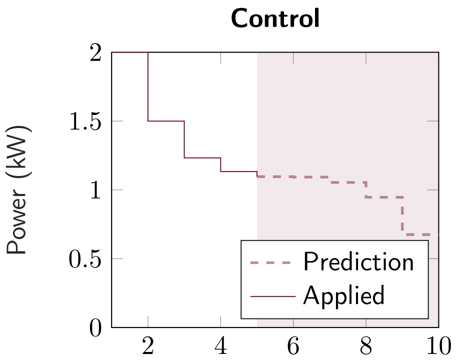


CentraleSupélec

following what we call the receding horizon strategy. However this problem is not always straight-forward to solve, for some cases it can be easier.

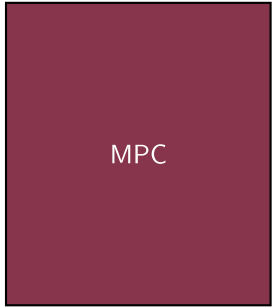
Find optimal control sequence, apply first element, rinse repeat → Receding Horizon





following what we call the receding horizon strategy. However this problem is not always straightforward to solve, for some cases it can be easier.

- Problem: Complexity depends on N, m, p and sizes of $\boldsymbol{x}$ and $\boldsymbol{u}$
- Solution: Divide and Conquer[1]



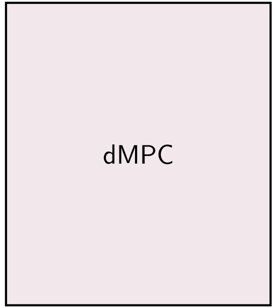
1

CentraleSupélec

However, the solution will depend on the horizon, the number of constraints, and sizes of input and states, increasing the complexity of the calculation

Distributed Model Predictive Control

- Problem: Complexity depends on N, m, p and sizes of $\boldsymbol{x}$ and $\boldsymbol{u}$
- Solution: Divide and Conquer[1]

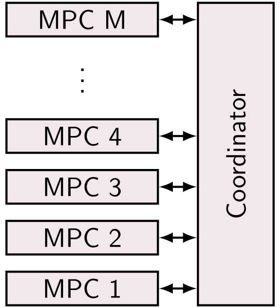


[1] *Distributed Model Predictive Control made easy*

A strategy to alleviate is to distribute the calculation whenever possible. And there are many ways to divide it as the book shows.

Distributed Model Predictive Control

- Problem: Complexity depends on N, m, p and sizes of $\boldsymbol{x}$ and $\boldsymbol{u}$
- Solution: Divide and Conquer[1]



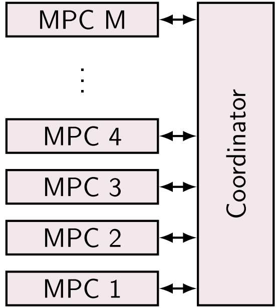
[1] *Distributed Model Predictive Control made easy*

CentraleSupélec

Here we opt for a hierarchical strategy where we use multiple MPCs and an agent to coordinate and manage the coupling aspects of the problem.

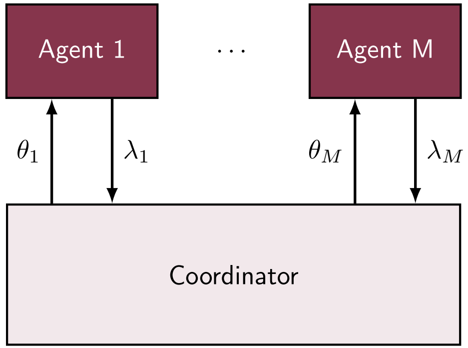
Distributed Model Predictive Control

- Problem: Complexity depends on N, m, p and sizes of $\boldsymbol{x}$ and $\boldsymbol{u}$
- Solution: Divide and Conquer[1]



[1] *Distributed Model Predictive Control made easy*

Here we opt for a hierarchical strategy where we use multiple MPCs and an agent to coordinate and manage the coupling aspects of the problem.

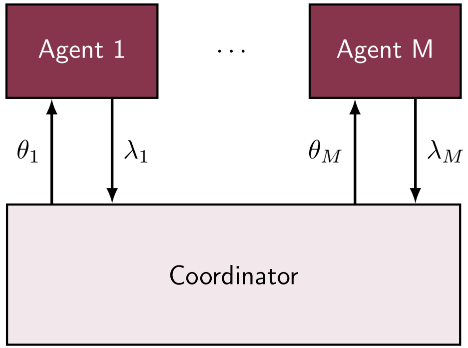


- Agents solve local problems
- Variables are updated

$\Big\}$ Until Convergence

Distributed Model Predictive Control

Optimization Frameworks

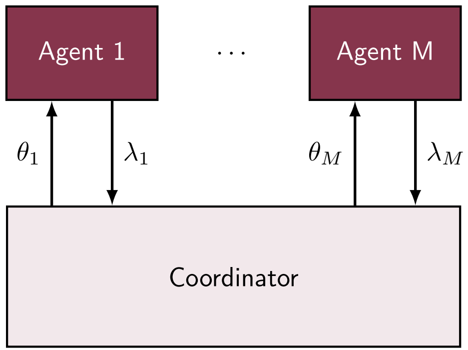


- Agents solve local problems
- Variables are updated

Until Convergence

Distributed Model Predictive Control

Optimization Frameworks



- Agents solve local problems
- Variables are updated

} Until Convergence

CentraleSupélec

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

- What are the consequences of an attack?
- Can we mitigate the effects?

CentraleSupélec

Distributed Model Predictive Control

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

- What are the consequences of an attack?
- Can we mitigate the effects?

CentraleSupélec

Distributed Model Predictive Control

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

- What are the consequences of an attack?
- Can we mitigate the effects?

Distributed Model Predictive Control

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

- What are the consequences of an attack?
- Can we mitigate the effects?

CentraleSupélec

State of art

	Decomposition	Present vulnerabilities?	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Yes	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Yes	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	Yes	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Yes	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Yes	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

State of art

	Decomposition	Present vulnerabilities?	Resilient/Robust	Detection	Mitigation
[Vel+17a] [Mae+21]	Dual	Yes	Robust (Scenario)	NA	NA
[Vel+17b] [Vel+18]	Dual	Yes	Robust (f-robust)	NA	NA
[CMI18]	Jacobi-Gauß	Yes	–	–	–
[Ana+18] [Ana+19] [Ana+20]	Dual	Yes	Resilient	Analyt./Learn.	Disconnect (Robustness)
Our	Primal	Yes	Resilient	Active Analyt./Learn.	Data reconstruction

CentraleSupélec

Outline

1. Vulnerabilities in distributed MPC based on Primal Decomposition
2. Resilient Primal Decomposition-based dMPC for deprived systems
3. Resilient Primal Decomposition-based dMPC using Artificial Scarcity

CentraleSupélec

To respond this this presentation is divided into 3 parts.First we present the decomposition and its vulnerabilites,

Outline

1. Vulnerabilities in distributed MPC based on Primal Decomposition
2. Resilient Primal Decomposition-based dMPC for deprived systems
3. Resilient Primal Decomposition-based dMPC using Artificial Scarcity

CentraleSupélec

To respond this this presentation is divided into 3 parts.First we present the decomposition and its vulnerabilites, We propose a resilient method for two kind of systems with increasing complexities.

Outline

1. Vulnerabilities in distributed MPC based on Primal Decomposition
2. Resilient Primal Decomposition-based dMPC for deprived systems
3. Resilient Primal Decomposition-based dMPC using Artificial Scarcity

CentraleSupélec

To respond this this presentation is divided into 3 parts.First we present the decomposition and its vulnerabilites, We propose a resilient method for two kind of systems with increasing complexities.

Outline

1. Vulnerabilities in distributed MPC based on Primal Decomposition
   - What is the Primal Decomposition?
   - How can an agent attack?
   - Consequences

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

Decompose original problem using primal problem

CentraleSupélec

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

Decompose original problem using primal problem

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

CentraleSupélec

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem. Those coupling constraints are replaced by local constraints with an allocation theta i.

Distributed Model Predictive Control

Decompose original problem using primal problem

$$\begin{aligned} \underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}}\sum_{\xi\in\mathcal{N}}\left[\|\boldsymbol{v}_i[\xi|k]\|^2_{Q_i} + \|\boldsymbol{u}_i[\xi-1|k]\|^2_{R_i}\right] \\ \text{subject to} \quad & \left.\begin{aligned} \boldsymbol{x}[\xi|k] &= f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &\leqslant 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &= 0 \end{aligned}\right\} \begin{aligned} &\forall \xi \in \{1,\dots,N\} \\ &\forall i \in \{1,\dots,m\} \\ &\forall j \in \{1,\dots,p\} \end{aligned} \end{aligned}$$

CentraleSupélec

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem. Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraints

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

E.g. $v_i = w_i - x_i$

Decompose original problem using primal problem

$$
\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & \sum_{i \in \mathcal{M}} \sum_{\xi \in \mathcal{N}} \left[\|\boldsymbol{v}_i[\xi|k]\|^2_{Q_i} + \|\boldsymbol{u}_i[\xi-1|k]\|^2_{R_i} \right] \\
\text{subject to} \quad & \left.\begin{aligned} \boldsymbol{x}[\xi|k] &= f(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &\leqslant 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &= 0 \end{aligned}\right\} \begin{aligned} &\forall \xi \in \{1, \dots, N\} \\ &\forall i \in \{1, \dots, m\} \\ &\forall j \in \{1, \dots, p\} \end{aligned}
\end{aligned}
$$

CentraleSupélec

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem. Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraintsWe can represent the sub-problems as agents and the master problem as a coordinator. The coordinator sends the allocations for each agent, which responds with a dual variable.

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

Decompose original problem using primal problem

$$\begin{aligned} \underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}}\sum_{\xi\in\mathcal{N}} \left[\|\boldsymbol{v}_i[\xi|k]\|^2_{Q_i} + \|\boldsymbol{u}_i[\xi-1|k]\|^2_{R_i}\right] \\ \text{subject to} \quad & \left.\begin{aligned} \boldsymbol{x}[\xi|k] &= A\boldsymbol{x}[\xi-1|k] + B\boldsymbol{u}[\xi-1|k] \\ g_i(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &\leqslant 0 \\ h_j(\boldsymbol{x}[\xi-1|k], \boldsymbol{u}[\xi-1|k]) &= 0 \end{aligned}\right\} \begin{aligned} &\forall \xi \in \{1,\dots,N\} \\ &\forall i \in \{1,\dots,m\} \\ &\forall j \in \{1,\dots,p\} \end{aligned} \end{aligned}$$

CentraleSupélec

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem.
Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraintsWe can represent the sub-problems as agents and the master problem as a coordinator. The coordinator sends the allocations for each agent, which responds with a dual variable.

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

Decompose original problem using primal problem

$$\begin{aligned}
\underset{\boldsymbol{u}[0:N-1|k]}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} \sum_{\xi\in\mathcal{N}} \left[\|\boldsymbol{v}_i[\xi|k]\|^2_{Q_i} + \|\boldsymbol{u}_i[\xi-1|k]\|^2_{R_i} \right] \\
\text{subject to} \quad & \left. \begin{aligned} \boldsymbol{x}[\xi|k] = A_i \boldsymbol{x}[\xi-1|k] + B_i \boldsymbol{u}[\xi-1|k] \\ \sum_{i\in\mathcal{M}} \Gamma_i \boldsymbol{u}_i[\xi|k] \leqslant \boldsymbol{u}_{\max} \end{aligned} \right\} \forall \xi \in \mathcal{N}
\end{aligned}$$

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem. Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraintsWe can represent the sub-problems as agents and the master problem as a coordinator. The coordinator sends the allocations for each agent, which responds with a dual variable.

Distributed Model Predictive Control

Decompose original problem using primal problem

$$\begin{aligned} \underset{\boldsymbol{U}_1[k],\ldots,\boldsymbol{U}_M[k]}{\text{minimize}} \quad & \sum_{i\in\mathcal{M}} \left[\frac{1}{2}\left\|\boldsymbol{U}_i[k]\right\|^2_{H_i} + \boldsymbol{f}_i[k]^T\boldsymbol{U}_i[k]\right] \\ \text{subject to} \quad & \sum_{i\in\mathcal{M}} \left[\bar{\Gamma}_i\boldsymbol{U}_i[k]\right] \preceq \boldsymbol{U}_{\mathsf{max}} \end{aligned}$$

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem. Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraintsWe can represent the sub-problems as agents and the master problem as a coordinator. The coordinator sends the allocations for each agent, which responds with a dual variable.

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

Decompose original problem using primal problem

$$\begin{aligned} &\underset{\boldsymbol{U}_1[k]}{\text{minimize}} && \tfrac{1}{2}\left\|\boldsymbol{U}_1[k]\right\|_{H_1}^2 + \boldsymbol{f}_1[k]^T\boldsymbol{U}_1[k] \\ &\text{subject to} && \bar{\Gamma}_1\boldsymbol{U}_1[k] \leq \boldsymbol{\theta}_1[k] : \boldsymbol{\lambda}_1[k] \\ & && \vdots \\ &\underset{\boldsymbol{U}_M[k]}{\text{minimize}} && \tfrac{1}{2}\left\|\boldsymbol{U}_M[k]\right\|_{H_M}^2 + \boldsymbol{f}_M[k]^T\boldsymbol{U}_M[k] \\ &\text{subject to} && \bar{\Gamma}_M\boldsymbol{U}_M[k] \leq \boldsymbol{\theta}_M[k] : \boldsymbol{\lambda}_M[k] \end{aligned}$$

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

CentraleSupélec

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem.
Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraintsWe can represent the sub-problems as agents and the master problem as a coordinator. The coordinator sends the allocations for each agent, which responds with a dual variable.

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

Decompose original problem using primal problem $\mathcal{S} = \{\boldsymbol{\theta}[k] \mid I_c^M \boldsymbol{\theta}[k] \leq \boldsymbol{U}_{\max}\}$

$$\begin{array}{ll} \underset{\boldsymbol{U}_1[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_1[k]\right\|_{H_1}^2 + \boldsymbol{f}_1[k]^T \boldsymbol{U}_1[k] \\ \text{subject to} & \bar{\Gamma}_1 \boldsymbol{U}_1[k] \leq \boldsymbol{\theta}_1[k] : \boldsymbol{\lambda}_1[k] \\ & \vdots \\ \underset{\boldsymbol{U}_M[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_M[k]\right\|_{H_M}^2 + \boldsymbol{f}_M[k]^T \boldsymbol{U}_M[k] \\ \text{subject to} & \bar{\Gamma}_M \boldsymbol{U}_M[k] \leq \boldsymbol{\theta}_M[k] : \boldsymbol{\lambda}_M[k] \end{array}$$

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

An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem.
Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraintsWe can represent the sub-problems as agents and the master problem as a coordinator. The coordinator sends the allocations for each agent, which responds with a dual variable.

Distributed Model Predictive Control

Primal Decomposition | Quantity Decomposition | Resource Allocation

Decompose original problem using primal problem $\mathcal{S} = \{\boldsymbol{\theta}[k] \mid I_c^M \boldsymbol{\theta}[k] \leq \boldsymbol{U}_{\max}\}$

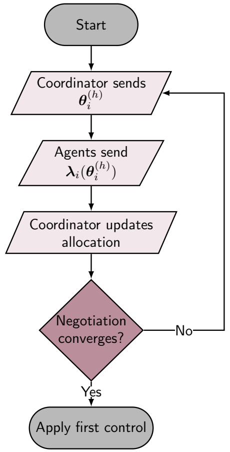
$$\begin{array}{ll} \underset{\boldsymbol{U}_1[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_1[k]\right\|_{H_1}^2 + \boldsymbol{f}_1[k]^T \boldsymbol{U}_1[k] \\ \text{subject to} & \bar{\Gamma}_1 \boldsymbol{U}_1[k] \leq \boldsymbol{\theta}_1[k] : \boldsymbol{\lambda}_1[k] \\ & \vdots \\ \underset{\boldsymbol{U}_M[k]}{\text{minimize}} & \frac{1}{2}\left\|\boldsymbol{U}_M[k]\right\|_{H_M}^2 + \boldsymbol{f}_M[k]^T \boldsymbol{U}_M[k] \\ \text{subject to} & \bar{\Gamma}_M \boldsymbol{U}_M[k] \leq \boldsymbol{\theta}_M[k] : \boldsymbol{\lambda}_M[k] \end{array}$$

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

CentraleSupélec

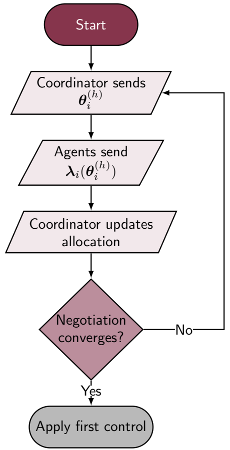
An example of decomposition method is the Quantity decomposition where a semi-decomposable problem with a global coupling constraints can be decomposed into multiple sub-problems, which can be solved in parallel, and a master problem which corresponds to the initial problem. Those coupling constraints are replaced by local constraints with an allocation theta i. The allocation for each sub-problem is updated by a projected subgradient method solving the master problem, thus the original problem. The subgradient used in this method is the dual variable associated to the coupling constraintsWe can represent the sub-problems as agents and the master problem as a coordinator. The coordinator sends the allocations for each agent, which responds with a dual variable.

Quantity Decomposition | Resource Allocation



In a flowchart for a quantity decomposition based DMPC,

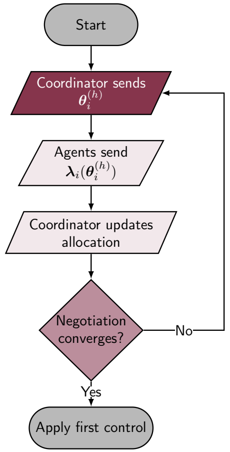
Quantity Decomposition | Resource Allocation



CentraleSupélec

In a flowchart for a quantity decomposition based DMPC,

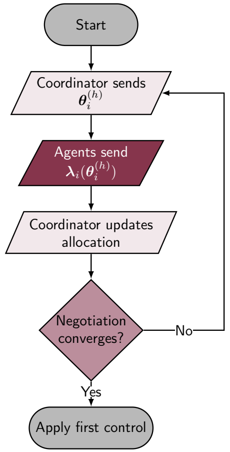
Quantity Decomposition | Resource Allocation



CentraleSupélec

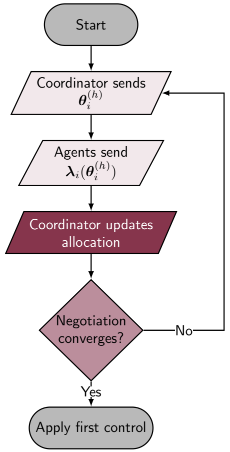
In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta,

Quantity Decomposition | Resource Allocation



In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda,

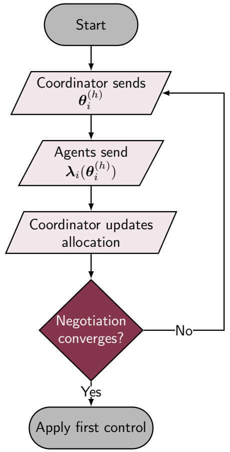
Quantity Decomposition | Resource Allocation



In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda, the coordinator updates the allocation.

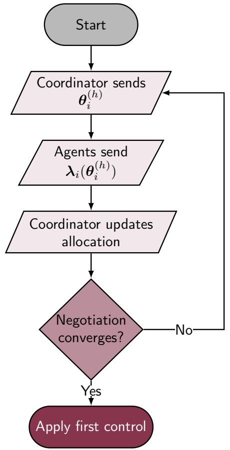
CentraleSupélec

Quantity Decomposition | Resource Allocation



In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda, the coordinator updates the allocation. If negotiation converges,

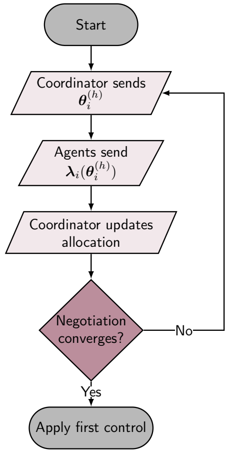
Quantity Decomposition | Resource Allocation



CentraleSupélec

In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda, the coordinator updates the allocation. If negotiation converges, then the negotiation ends and each agent applies the first element of the control found

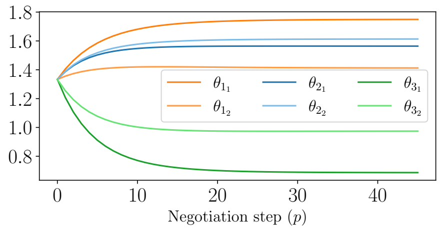
Quantity Decomposition | Resource Allocation

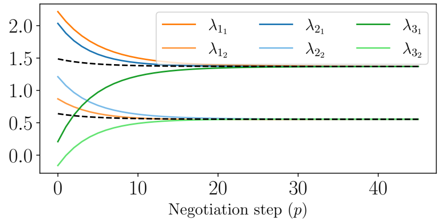


CentraleSupélec

In a flowchart for a quantity decomposition based DMPC,the coordinator sends the allocation theta, the agents send the dual variable lambda, the coordinator updates the allocation. If negotiation converges, then the negotiation ends and each agent applies the first element of the control found

Quantity Decomposition | Resource Allocation





CentraleSupélec

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks (False Data Injection)

CentraleSupélec

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks (False Data Injection)

CentraleSupélec

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks
(False Data Injection)

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks (False Data Injection)

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks (False Data Injection)

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks
(False Data Injection)

CentraleSupélec

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks (False Data Injection)

How can a non-cooperative agent attack?

Literature

- [Vel+17a; CMI18] present some kinds of attacks
  - Objective function
    - Selfish Attack
    - Fake weights
    - Fake reference
  - Fake constraints
  - Liar agent (use different control)

Deception Attacks
(False Data Injection)

How can a non-cooperative agent attack?

Our approach

- $\boldsymbol{\lambda}_i$ is the only interface with coordination
- $\boldsymbol{\lambda}_i$ depends on the parameters of the system
- Malicious agent sends a different $\boldsymbol{\lambda}_i$

CentraleSupélec

How can a non-cooperative agent attack?

Our approach

- $\boldsymbol{\lambda}_i$ is the only interface with coordination
- $\boldsymbol{\lambda}_i$ depends on the parameters of the system
- Malicious agent sends a different $\boldsymbol{\lambda}_i$

How can an agent attack the system? Its only interface with the coordination is lambda

How can a non-cooperative agent attack?

Our approach

- $\boldsymbol{\lambda}_i$ is the only interface with coordination
- $\boldsymbol{\lambda}_i$ depends on the parameters of the system
- Malicious agent sends a different $\boldsymbol{\lambda}_i$

How can an agent attack the system? Its only interface with the coordination is lambdaso we suppose it attacks by sending a different $lambda_i$, modified by a function $gamma_i$ of $lambda_i$

How can a non-cooperative agent attack?

Our approach

- $\boldsymbol{\lambda}_i$ is the only interface with coordination
- $\boldsymbol{\lambda}_i$ depends on the parameters of the system
- Malicious agent sends a different $\boldsymbol{\lambda}_i$

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

CentraleSupélec

How can an agent attack the system? Its only interface with the coordination is lambdaso we suppose it attacks by sending a different $lambda_i$, modified by a function $gamma_i$ of $lambda_i$

Example

Let's suppose $\gamma_i(\boldsymbol{\lambda}_i) = T_i\boldsymbol{\lambda}_i$

We give an example of 4 agents negotiating

Example

Let's suppose $\gamma_i(\boldsymbol{\lambda}_i) = T_i\boldsymbol{\lambda}_i$

4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative

Example

Let's suppose $\gamma_i(\boldsymbol{\lambda}_i) = T_i\boldsymbol{\lambda}_i$

4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when

Example

Let's suppose $\gamma_i(\boldsymbol{\lambda}_i) = T_i\boldsymbol{\lambda}_i$

4 distinct agents

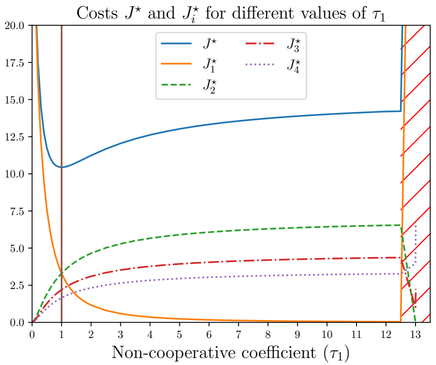
- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high

Example

Let's suppose $\gamma_i(\boldsymbol{\lambda}_i) = T_i\boldsymbol{\lambda}_i$



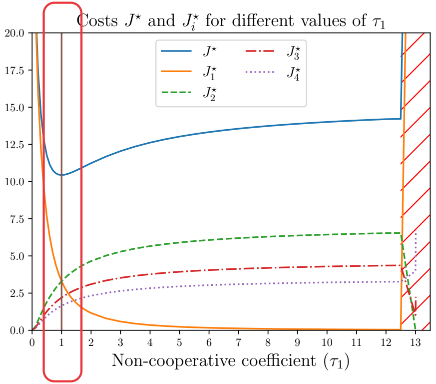
4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high
it can turn the negotiation unstable

Example

Let's suppose $\gamma_i(\boldsymbol{\lambda}_i) = T_i \boldsymbol{\lambda}_i$



4 distinct agents

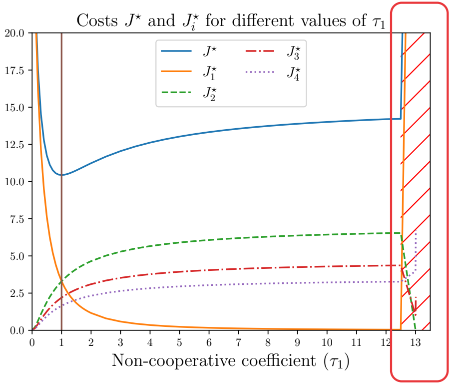
- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

CentraleSupélec

We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high
it can turn the negotiation unstable

Example

Let's suppose $\gamma_i(\boldsymbol{\lambda}_i) = T_i\boldsymbol{\lambda}_i$



4 distinct agents

- Agent 1 is non-cooperative
- It uses $\tilde{\boldsymbol{\lambda}}_1 = \gamma_1(\boldsymbol{\lambda}_1) = \tau_1 I \boldsymbol{\lambda}_1$

CentraleSupélec

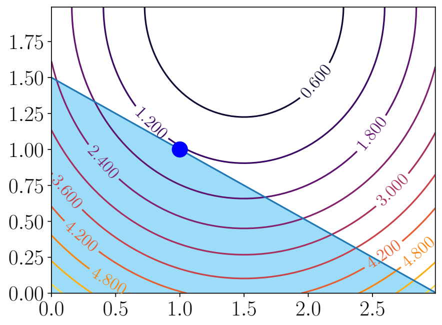
We give an example of 4 agents negotiating Agent 1 is non-cooperative
It uses a linear cheating function gamma tau times identity times the original lambda i
In the figure we can see the cost functions for each agent, we see that agent 1 cost decreases if we increase tau, but the overall cost is increased, The minimum value of the global cost is when tau is equal to one. If tau is close to zero it increases its cost and if tau is too high
it can turn the negotiation unstable

Mitigating

There are vulnerabilities

Mitigating

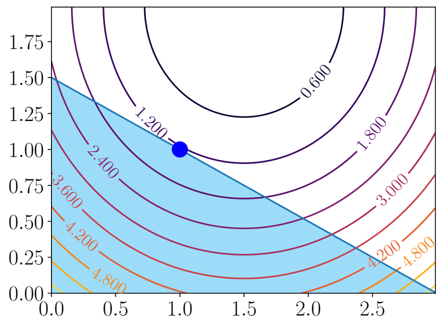
There are vulnerabilities



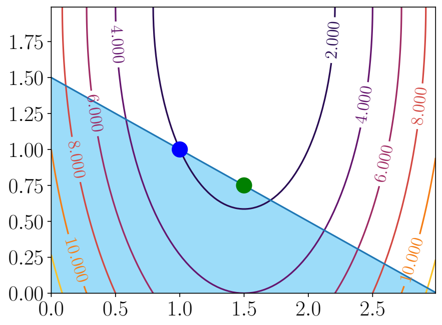
Original minimum.

Mitigating

There are vulnerabilities



Original minimum.



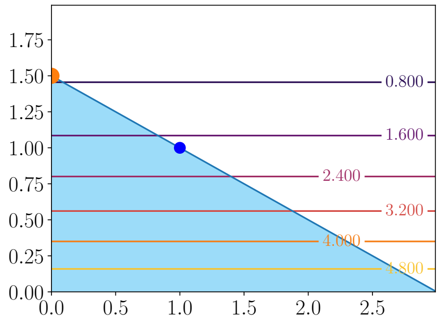
Minimum after attack.

Mitigating

There are vulnerabilities
"Great". But what can we do?

Mitigating

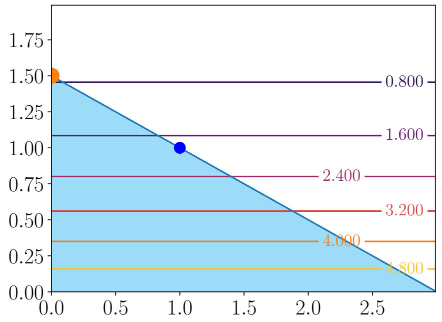
There are vulnerabilities
"Great". But what can we do?



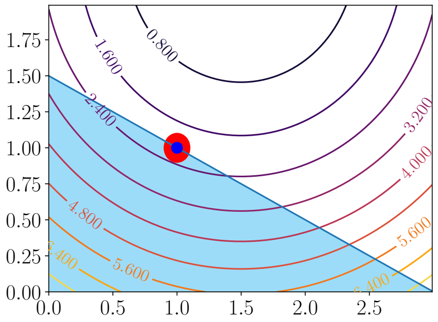
Ignore attacker.

Mitigating

There are vulnerabilities
"Great". But what can we do?



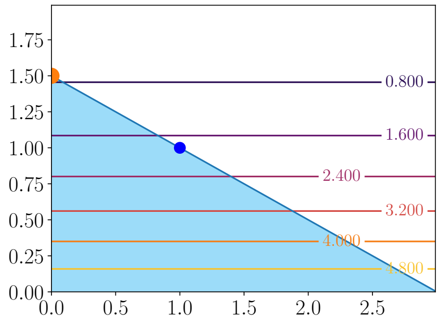
Ignore attacker.



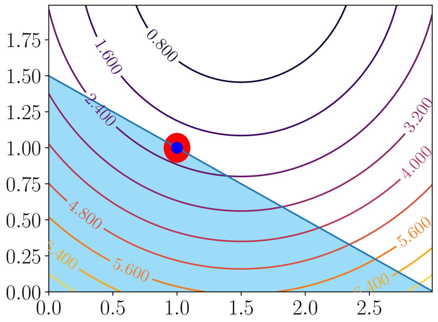
Recover original behavior.

Mitigating

There are vulnerabilities
"Great". But what can we do?



Ignore attacker.



Recover original behavior.

Strategy

- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i)$
- Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?
  - Not necessarily, but let's reason

Strategy

- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i)$
- Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?
  - Not necessarily, but let's reason

CentraleSupélec

Strategy

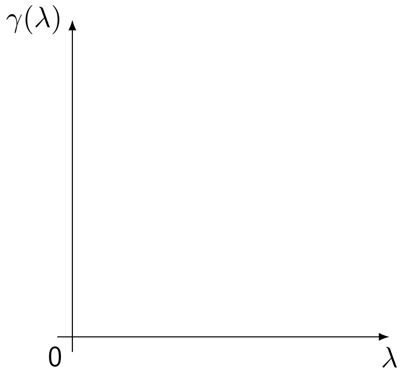
- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i)$
- Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?
  - Not necessarily, but let's reason

Strategy

- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i)$
- Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?
  - Not necessarily, but let's reason

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

Assumptions

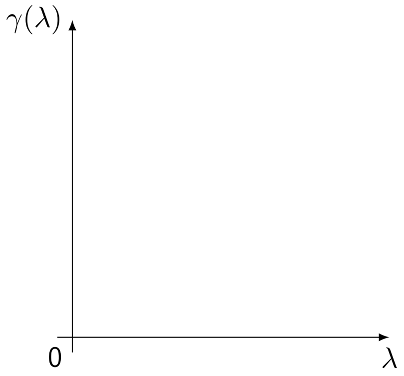
- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart

  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

Assumptions

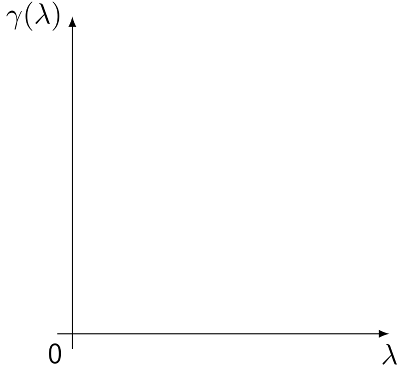
- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart

  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

Assumptions

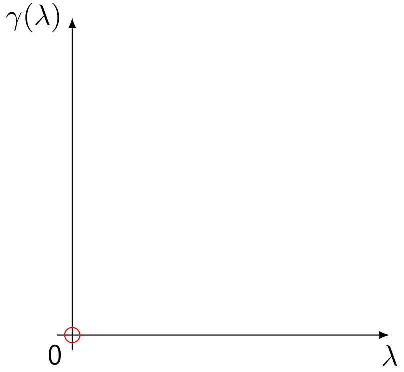
- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart

  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

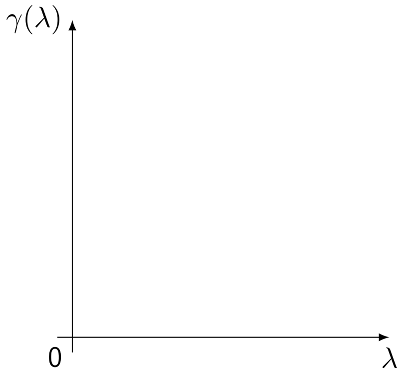
Assumptions

- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart

  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

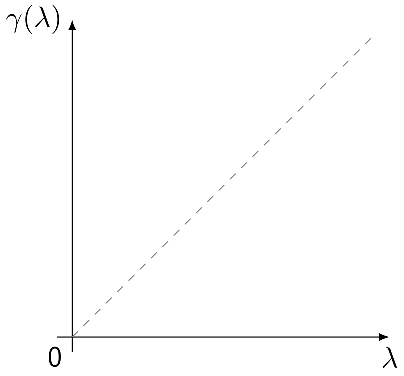
Assumptions

- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart

  $\lambda_b > \lambda_a \Rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

Assumptions

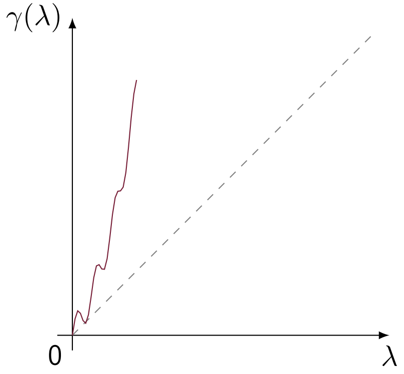
- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart

$\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

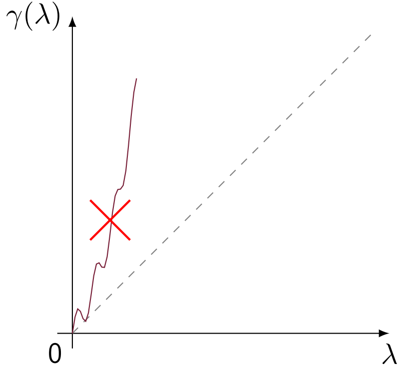
Assumptions

- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart

  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

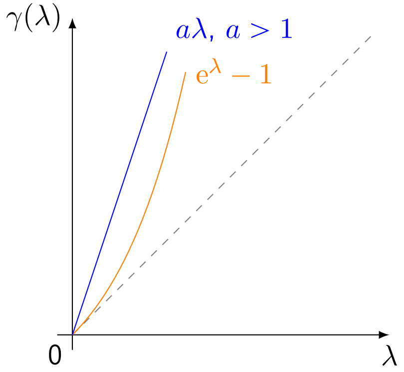
Assumptions

- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart
  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



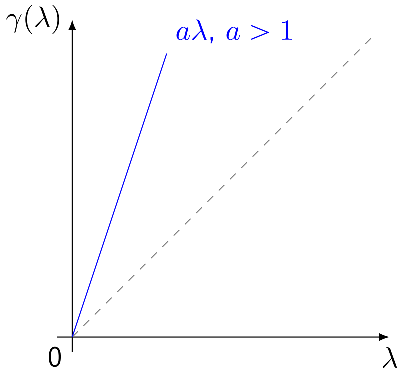
- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

Assumptions

- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart
  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Unidimensional Case



- $\lambda \geqslant 0$ means dissatisfaction
- $\lambda = 0$ means complete satisfaction
  - $\gamma(\lambda) = 0 \Leftrightarrow \lambda = 0$

Assumptions

- Attacker is greedy $\gamma(\lambda) > \lambda$
- But not smart
  $\lambda_b > \lambda_a \rightarrow \gamma(\lambda_b) > \gamma(\lambda_a)$

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Linear Multidimensional Case

- $\gamma(\lambda) = T\lambda$
- $T\lambda = 0 \Leftrightarrow \lambda = 0 \rightarrow \exists T^{-1}$
- Each agent could choose different T_i each time

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Linear Multidimensional Case

- $\gamma(\lambda) = T\lambda$
- $T\lambda = 0 \Leftrightarrow \lambda = 0 \rightarrow \exists T^{-1}$
- Each agent could choose different T_i each time

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Linear Multidimensional Case

- $\gamma(\lambda) = T\lambda$
- $T\lambda = 0 \Leftrightarrow \lambda = 0 \rightarrow \exists T^{-1}$
- Each agent could choose different T_i each time

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Linear Multidimensional Case

- $\gamma(\lambda) = T\lambda$
- $T\lambda = 0 \Leftrightarrow \lambda = 0 \rightarrow \exists T^{-1}$
- Each agent could choose different T_i each time

CentraleSupélec

Is $\gamma_i(\boldsymbol{\lambda}_i)$ invertible?

Linear Multidimensional Case

- $\gamma(\lambda) = T\lambda$
- $T\lambda = 0 \Leftrightarrow \lambda = 0 \rightarrow \exists T^{-1}$
- Each agent could choose different T_i each time $\rightarrow T_i[k]$

CentraleSupélec

Strategy

- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i)$ $\rightarrow$ Estimate $T_i[k]^{-1}$
- But how? Analyzing the system

CentraleSupélec

Strategy

- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i) \rightarrow$ Estimate $T_i[k]^{-1}$
- But how? Analyzing the system

Strategy

- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i) \rightarrow$ Estimate $T_i[k]^{-1}$
- But how? Analyzing the system

Strategy

- Recover original behavior
  - Invert effects of function $\gamma_i(\boldsymbol{\lambda}_i) \rightarrow$ Estimate $T_i[k]^{-1}$
- But how? Analyzing the system

Outline

2 Resilient Primal Decomposition-based dMPC for deprived systems
- Analyzing deprived systems
- Building an algorithm
- Applying mechanism

What are deprived systems?

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

- Unconstrained solution $\mathring{\boldsymbol{U}}_i^\star[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^\star[k] > \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)

CentraleSupélec

What are deprived systems?

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

- Unconstrained solution $\mathring{\boldsymbol{U}}_i^\star[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^\star[k] > \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)

CentraleSupélec

What are deprived systems?

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

- Unconstrained solution $\mathring{\boldsymbol{U}}_i^\star[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^\star[k] > \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)

What are deprived systems?

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

- Unconstrained solution $\mathring{\boldsymbol{U}}_i^\star[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^\star[k] > \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)

CentraleSupélec

What are deprived systems?

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

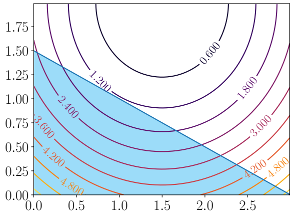
- Unconstrained solution $\mathring{\boldsymbol{U}}_i^\star[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^\star[k] \succ \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)

CentraleSupélec

What are deprived systems?

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

- Unconstrained solution $\mathring{\boldsymbol{U}}_i^{\star}[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i\mathring{\boldsymbol{U}}_i^{\star}[k] > \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)

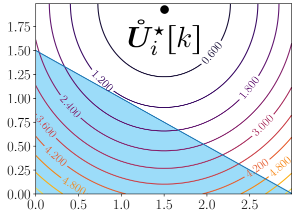


CentraleSupélec

What are deprived systems?

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

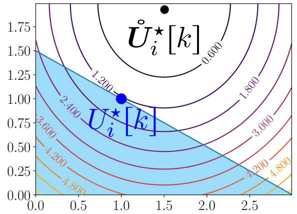
- Unconstrained solution $\mathring{\boldsymbol{U}}_i^\star[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i \mathring{\boldsymbol{U}}_i^\star[k] > \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)



What are deprived systems?

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

- Unconstrained solution $\mathring{\boldsymbol{U}}_i^\star[k] = -H_i^{-1}\boldsymbol{f}_i[k]$
- Deprived if constraints are active $\rightarrow \bar{\Gamma}_i \mathring{\boldsymbol{U}}_i^\star[k] > \boldsymbol{\theta}_i[k], \forall k$
  - Solution projected onto boundaries (equality constraints)



CentraleSupélec

Deprived Systems

But why?

- No Scarcity → All constraints satisfied → $\lambda_i = 0$ → No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

Deprived Systems

But why?

- No Scarcity → All constraints satisfied → $\lambda_i = 0$ → No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

Deprived Systems

But why?

- No Scarcity → All constraints satisfied → $\lambda_i = 0$ → No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

Deprived Systems

But why?

- No Scarcity → All constraints satisfied → $\lambda_i = 0$ → No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

CentraleSupélec

Deprived Systems

But why?

- No Scarcity → All constraints satisfied → $\lambda_i = 0$ → No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

Deprived Systems

But why?

- No Scarcity → All constraints satisfied $\rightarrow \lambda_i = 0 \rightarrow$ No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

CentraleSupélec

Deprived Systems

But why?

- No Scarcity → All constraints satisfied → $\lambda_i = 0$ → No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

CentraleSupélec

Deprived Systems

But why?

- No Scarcity → All constraints satisfied → $\lambda_i = 0$ → No coordination needed
- Scarcity → Competition → Consensus/Compromise (or cheating)

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i = (\bar{\Gamma}_i H_i^{-1}\bar{\Gamma}_i^T)^{-1}$ and $\boldsymbol{s}_i[k] = P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2] Under some conditions

CentraleSupélec

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i = (\bar{\Gamma}_i H_i^{-1} \bar{\Gamma}_i^T)^{-1}$ and $\boldsymbol{s}_i[k] = P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2] Under some conditions see here

CentraleSupélec

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i = (\bar{\Gamma}_i H_i^{-1}\bar{\Gamma}_i^T)^{-1}$ and $\boldsymbol{s}_i[k] = P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2] Under some conditions see here

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i = (\bar{\Gamma}_i H_i^{-1}\bar{\Gamma}_i^T)^{-1}$ and $\boldsymbol{s}_i[k] = P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2] Under some conditions ▸ see here

CentraleSupélec

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i = (\bar{\Gamma}_i H_i^{-1}\bar{\Gamma}_i^T)^{-1}$ and $\boldsymbol{s}_i[k] = P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2] Under some conditions ▸ see here

CentraleSupélec

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i = (\bar{\Gamma}_i H_i^{-1}\bar{\Gamma}_i^T)^{-1}$ and $\boldsymbol{s}_i[k] = P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2] Under some conditions ▸ see here

CentraleSupélec

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i=\left(\bar{\Gamma}_i H_i^{-1}\bar{\Gamma}_i^T\right)^{-1}$ and $\boldsymbol{s}_i[k]=P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2]Under some conditions see here

CentraleSupélec

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis

- We can transform inequality constraints into equality ones[2]
- Solution is analytical and trivial.
  - If we solve for λ_i

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

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

where $P_i = (\bar{\Gamma}_i H_i^{-1}\bar{\Gamma}_i^T)^{-1}$ and $\boldsymbol{s}_i[k] = P_i\bar{\Gamma}_i H_i^{-1}\boldsymbol{f}_i[k]$.

[2] Under some conditions see here

One can notice, that the solution embeds information not only of the objective function and the reference and state (presence of H_i and $\boldsymbol{f}_i[k]$), but also of the constraints (presence of $\bar{\Gamma}_i$ and $\boldsymbol{\theta}_i[k]$). So, changes in these parameters affect the resulting value of $\boldsymbol{\lambda}_i[k]$.

Deprived Systems

Analysis (Continued)

$$\mathcal{S} = \{\boldsymbol{\theta}[k] \mid I_c^M \boldsymbol{\theta}[k] \leqslant \boldsymbol{U}_{\max}\}$$

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

CentraleSupélec

Deprived Systems

Analysis (Continued)

$$\mathcal{S} = \{\boldsymbol{\theta}[k] \mid I_c^M \boldsymbol{\theta}[k] = \boldsymbol{U}_{\max}\}$$

$$\boldsymbol{\theta}_i^{(p+1)} = \boldsymbol{\theta}_i^{(p)} + \rho^{(p)} \left(\boldsymbol{\lambda}_i^{(p)} - \frac{1}{M} \sum_{j=1}^{M} \boldsymbol{\lambda}_j^{(p)} \right), \forall i \in \mathcal{M}$$

Deprived Systems

Analysis (Continued)

$$\mathcal{S} = \{\boldsymbol{\theta}[k] \mid I_c^M \boldsymbol{\theta}[k] = \boldsymbol{U}_{\max}\}$$

$$\boldsymbol{\theta}_i^{(p+1)} = \boldsymbol{\theta}_i^{(p)} + \rho^{(p)} \left(\boldsymbol{\lambda}_i^{(p)} - \frac{1}{M} \sum_{j=1}^{M} \boldsymbol{\lambda}_j^{(p)} \right), \forall i \in \mathcal{M}$$

Deprived Systems

Analysis (Continued)

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

$$\mathcal{S} = \{\boldsymbol{\theta}[k] \mid I_c^M\boldsymbol{\theta}[k] = \boldsymbol{U}_{\max}\}$$

$$\boldsymbol{\theta}_i^{(p+1)} = \boldsymbol{\theta}_i^{(p)} + \rho^{(p)}\left(\boldsymbol{\lambda}_i^{(p)} - \frac{1}{M}\sum_{j=1}^{M}\boldsymbol{\lambda}_j^{(p)}\right), \forall i \in \mathcal{M}$$

Deprived Systems

Analysis (Continued)

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

$$\mathcal{S} = \{\boldsymbol{\theta}[k] \mid I_c^M \boldsymbol{\theta}[k] = \boldsymbol{U}_{\max}\}$$

$$\boldsymbol{\theta}_i^{(p+1)} = \boldsymbol{\theta}_i^{(p)} + \rho^{(p)}\left(\boldsymbol{\lambda}_i^{(p)} - \frac{1}{M}\sum_{j=1}^{M}\boldsymbol{\lambda}_j^{(p)}\right), \forall i \in \mathcal{M}$$

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

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior

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

- Under attack

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

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

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior

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

- Under attack

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

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

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior

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

- Under attack

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

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

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior

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

- Under attack

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

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

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior

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

- Under attack

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

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

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

CentraleSupélec

Deprived Systems

Under attack!

- Normal behavior

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

- Under attack

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

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

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

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

Assumption

Attacker chooses $\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i) = T_i(k)\boldsymbol{\lambda}_i$
$-T_i(k)P_i\boldsymbol{\theta}_i - T_i(k)\boldsymbol{s}_i(k) \rightarrow -\tilde{P}_i\boldsymbol{\theta}_i - \tilde{\boldsymbol{s}}_i(k)$

- We can estimate[1] $\hat{P}_i$ and $\hat{\tilde{\boldsymbol{s}}}_i(k)$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i(\boldsymbol{\theta}_i)) = -\widehat{\tilde{P}_i}(k)\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i(k)$$

- If $\hat{\tilde{P}}_i(k) \neq \bar{P}_i \rightarrow$ Attack

[1] Using Recursive Least Squares

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

Assumption

Attacker chooses $\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i) = T_i(k)\boldsymbol{\lambda}_i$
$-T_i(k)P_i\boldsymbol{\theta}_i - T_i(k)\boldsymbol{s}_i(k) \rightarrow -\tilde{P}_i\boldsymbol{\theta}_i - \tilde{\boldsymbol{s}}_i(k)$

- We can estimate[1] $\hat{P}_i$ and $\hat{\tilde{\boldsymbol{s}}}_i(k)$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i(\boldsymbol{\theta}_i)) = -\widehat{\tilde{P}_i}(k)\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i(k)$$

- If $\hat{\tilde{P}}_i(k) \neq \bar{P}_i \rightarrow$ Attack

[1] Using Recursive Least Squares

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

Assumption

Attacker chooses $\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i) = T_i(k)\boldsymbol{\lambda}_i$
$-T_i(k)P_i\boldsymbol{\theta}_i - T_i(k)\boldsymbol{s}_i(k) \rightarrow -\tilde{P}_i\boldsymbol{\theta}_i - \tilde{\boldsymbol{s}}_i(k)$

- We can estimate[1] $\hat{P}_i$ and $\hat{\tilde{\boldsymbol{s}}}_i(k)$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i(\boldsymbol{\theta}_i)) = -\widehat{\tilde{P}_i}(k)\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i(k)$$

- If $\hat{\tilde{P}}_i(k) \neq \bar{P}_i \rightarrow$ Attack

[1] Using Recursive Least Squares

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

Assumption

Attacker chooses $\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i) = T_i(k)\boldsymbol{\lambda}_i$
$-T_i(k)P_i\boldsymbol{\theta}_i - T_i(k)\boldsymbol{s}_i(k) \rightarrow -\tilde{P}_i\boldsymbol{\theta}_i - \tilde{\boldsymbol{s}}_i(k)$

- We can estimate[1] $\hat{\tilde{P}}_i$ and $\hat{\tilde{\boldsymbol{s}}}_i(k)$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i(\boldsymbol{\theta}_i)) = -\widehat{\tilde{P}_i}(k)\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i(k)$$

- If $\hat{\tilde{P}}_i(k) \neq \bar{P}_i \rightarrow$ Attack

[1] Using Recursive Least Squares

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde
Now we can estimate P tilde and s tilde for a given negotiation in time k

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

Assumption

Attacker chooses $\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i) = T_i(k)\boldsymbol{\lambda}_i$
$-T_i(k)P_i\boldsymbol{\theta}_i - T_i(k)\boldsymbol{s}_i(k) \rightarrow -\tilde{P}_i\boldsymbol{\theta}_i - \tilde{\boldsymbol{s}}_i(k)$

- We can estimate[1] $\hat{P}_i$ and $\hat{\tilde{\boldsymbol{s}}}_i(k)$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i(\boldsymbol{\theta}_i)) = -\widehat{\tilde{P}_i}(k)\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i(k)$$

- If $\hat{\tilde{P}}_i(k) \neq \bar{P}_i \rightarrow$ Attack

[1]Using Recursive Least Squares

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde
Now we can estimate P tilde and s tilde for a given negotiation in time k
and if the estimated P tilde is different from the nominal P, then there is an attack

Detection Mechanism

Assumption

We know nominal $\bar{P}_i$

Assumption

Attacker chooses $\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i) = T_i(k)\boldsymbol{\lambda}_i$
$-T_i(k)P_i\boldsymbol{\theta}_i - T_i(k)\boldsymbol{s}_i(k) \rightarrow -\tilde{P}_i\boldsymbol{\theta}_i - \tilde{\boldsymbol{s}}_i(k)$

- We can estimate[1] $\hat{P}_i$ and $\hat{\tilde{\boldsymbol{s}}}_i(k)$ such as:

$$\tilde{\boldsymbol{\lambda}}_i = \gamma_i(\boldsymbol{\lambda}_i(\boldsymbol{\theta}_i)) = -\widehat{\tilde{P}_i}(k)\boldsymbol{\theta}_i - \hat{\tilde{\boldsymbol{s}}}_i(k)$$

- If $\hat{\tilde{P}}_i(k) \neq \bar{P}_i \rightarrow$ Attack

[1] Using Recursive Least Squares

CentraleSupélec

For the attack detection, let's assume we know the nominal P, called P bar
And we also assume the attacker chooses a linear attack where the lambda sent is equal to a matrix T times the original lambda which yields the formula shown
with modified P and s, called P tilde and s tilde
Now we can estimate P tilde and s tilde for a given negotiation in time k
and if the estimated P tilde is different from the nominal P, then there is an attack

Detection

In detail

- Error $E_i(k) = \|\widehat{\bar{P}}_i(k) - \bar{P}_i\|_F$
- Create threshold ϵ_P
- Indicator $d_i \in \{0, 1\}$ detects the attack in agent i.
- $d_i = 1$ if $E_i(k) > \epsilon_P$, 0 otherwise

CentraleSupélec

So, let's detail the detection mechanism.

Detection

In detail

- Error $E_i(k) = \|\widehat{\bar{P}}_i(k) - \bar{P}_i\|_F$
- Create threshold ϵ_P
- Indicator $d_i \in \{0, 1\}$ detects the attack in agent i.
- $d_i = 1$ if $E_i(k) > \epsilon_P$, 0 otherwise

CentraleSupélec

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)

Detection

In detail

- Error $E_i(k) = \|\hat{\tilde{P}}_i(k) - \bar{P}_i\|_F$
- Create threshold ϵ_P
- Indicator $d_i \in \{0, 1\}$ detects the attack in agent i.
- $d_i = 1$ if $E_i(k) > \epsilon_P$, 0 otherwise

CentraleSupélec

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)
Then, we create a threshold epsilon p

Detection

In detail

- Error $E_i(k) = \|\hat{\bar{P}}_i(k) - \bar{P}_i\|_F$
- Create threshold ϵ_P
- Indicator $d_i \in \{0, 1\}$ detects the attack in agent i.
- $d_i = 1$ if $E_i(k) > \epsilon_P$, 0 otherwise

CentraleSupélec

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)
Then, we create a threshold epsilon p
and finally we create an indicator d i for the attack of agent i

Detection

In detail

- Error $E_i(k) = \|\hat{\bar{P}}_i(k) - \bar{P}_i\|_F$
- Create threshold ϵ_P
- Indicator $d_i \in \{0, 1\}$ detects the attack in agent i.
- $d_i = 1$ if $E_i(k) > \epsilon_P$, 0 otherwise

CentraleSupélec

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)
Then, we create a threshold epsilon p
and finally we create an indicator d i for the attack of agent i
It is equal to 1 if the error is greater than the threshold, indicating an attack or 0 otherwise

Detection

In detail

- Error $E_i(k) = \|\hat{\bar{P}}_i(k) - \bar{P}_i\|_F$
- Create threshold ϵ_P
- Indicator $d_i \in \{0, 1\}$ detects the attack in agent i.
- $d_i = 1$ if $E_i(k) > \epsilon_P$, 0 otherwise

So, let's detail the detection mechanism.
First we calculate the norm of the error between the estimated P and the nominal. (Here we use the Frobenius norm)
Then, we create a threshold epsilon p
and finally we create an indicator d i for the attack of agent i
It is equal to 1 if the error is greater than the threshold, indicating an attack or 0 otherwise

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

- We estimate $\hat{P}_i$ and $\widehat{\tilde{s}}_i(k)$ simultaneously using Recursive Least Squares
- Problem: Estimation during negotiation fails
  - Consecutive λ_i^p and θ_i^p are linearly dependent → low input excitation
- Solution: Send sequence of random values of θ_i until estimates converge

In order to estimate the modified P and s, we use recursive least squares

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

- We estimate $\hat{P}_i$ and $\widehat{\tilde{s}}_i(k)$ simultaneously using Recursive Least Squares
- Problem: Estimation during negotiation fails
  - Consecutive λ_i^p and θ_i^p are linearly dependent → low input excitation
- Solution: Send sequence of random values of θ_i until estimates converge

In order to estimate the modified P and s, we use recursive least squares
The problem of estimating the value during a normal negotiation is that it fails

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

- We estimate $\hat{P}_i$ and $\widehat{\tilde{\boldsymbol{s}}}_i(k)$ simultaneously using Recursive Least Squares
- Problem: Estimation during negotiation fails
  - Consecutive $\boldsymbol{\lambda}_i^p$ and $\boldsymbol{\theta}_i^p$ are linearly dependent → low input excitation
- Solution: Send sequence of random values of θ_i until estimates converge

CentraleSupélec

In order to estimate the modified P and s, we use recursive least squares
The problem of estimating the value during a normal negotiation is that it fails
Consecutive lambdas and thetas are linearly dependent, since they are calculate via the projected subgradient method, which gives us low input excitation

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

- We estimate $\hat{P}_i$ and $\hat{\tilde{s}}_i(k)$ simultaneously using Recursive Least Squares
- Problem: Estimation during negotiation fails
  - Consecutive $\boldsymbol{\lambda}_i^p$ and $\boldsymbol{\theta}_i^p$ are linearly dependent $\rightarrow$ low input excitation
- Solution: Send sequence of random values of $\boldsymbol{\theta}_i$ until estimates converge

In order to estimate the modified P and s, we use recursive least squares
The problem of estimating the value during a normal negotiation is that it fails
Consecutive lambdas and thetas are linearly dependent, since they are calculate via the projected subgradient method, which gives us low input excitation
In order to increase the input excitation we send a sequence of random theta until the convergence is attained

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

- We estimate $\hat{P}_i$ and $\widehat{\tilde{\boldsymbol{s}}}_i(k)$ simultaneously using Recursive Least Squares
- Problem: Estimation during negotiation fails
  - Consecutive $\boldsymbol{\lambda}_i^p$ and $\boldsymbol{\theta}_i^p$ are linearly dependent $\rightarrow$ low input excitation
- Solution: Send sequence of random values of $\boldsymbol{\theta}_i$ until estimates converge

CentraleSupélec

In order to estimate the modified P and s, we use recursive least squares
The problem of estimating the value during a normal negotiation is that it fails
Consecutive lambdas and thetas are linearly dependent, since they are calculate via the projected subgradient method, which gives us low input excitation
In order to increase the input excitation we send a sequence of random theta until the convergence is attained

Mitigation mechanism

- Main idea: Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

Assumption

We suppose $\tilde{\boldsymbol{\lambda}}_i = \mathbf{0}$ only if $\boldsymbol{\lambda}_i = \mathbf{0}$, which implies $T_i(k)$ invertible.

- Estimate the inverse of $T_i(k)$

$$\widehat{T_i(k)^{-1}} = \bar{P}_i \, \widehat{\widetilde{P}_i(k)^{-1}}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\boldsymbol{\lambda}_{i\text{rec}} = \widehat{T_i(k)^{-1}} \tilde{\boldsymbol{\lambda}}_i = -\bar{P}_i \boldsymbol{\theta}_i - \widehat{T_i(k)^{-1}} \hat{\tilde{\mathbf{s}}}_i(k)$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation

Mitigation mechanism

- Main idea: Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

Assumption

We suppose $\tilde{\boldsymbol{\lambda}}_i = \mathbf{0}$ only if $\boldsymbol{\lambda}_i = \mathbf{0}$, which implies $T_i(k)$ invertible.

- Estimate the inverse of $T_i(k)$

$$\widehat{T_i(k)^{-1}} = \bar{P}_i\, \widehat{P_i}(k)^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\boldsymbol{\lambda}_{i\text{rec}} = \widehat{T_i(k)^{-1}}\tilde{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i(k)^{-1}}\hat{\tilde{\mathbf{s}}}_i(k)$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible

Mitigation mechanism

- Main idea: Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

Assumption

We suppose $\tilde{\boldsymbol{\lambda}}_i = \mathbf{0}$ only if $\boldsymbol{\lambda}_i = \mathbf{0}$, which implies $T_i(k)$ invertible.

- Estimate the inverse of $T_i(k)$

$$\widehat{T_i(k)^{-1}} = \bar{P}_i \, \widehat{\tilde{P}_i}(k)^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\boldsymbol{\lambda}_{i\mathrm{rec}} = \widehat{T_i(k)^{-1}} \tilde{\boldsymbol{\lambda}}_i = -\bar{P}_i \boldsymbol{\theta}_i - \widehat{T_i(k)^{-1}} \hat{\tilde{\mathbf{s}}}_i(k)$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible
We can estimate the inverse of T, by using the estimate of P tilde and the nominal value of P like in this equation

Mitigation mechanism

- Main idea: Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

Assumption

We suppose $\tilde{\boldsymbol{\lambda}}_i = \mathbf{0}$ only if $\boldsymbol{\lambda}_i = \mathbf{0}$, which implies $T_i(k)$ invertible.

- Estimate the inverse of $T_i(k)$

$$\widehat{T_i(k)^{-1}} = \bar{P}_i\, \widehat{\tilde{P}_i}(k)^{-1}$$

- Reconstruct $\boldsymbol{\lambda}_i$

$$\boldsymbol{\lambda}_{i\text{rec}} = \widehat{T_i(k)^{-1}}\tilde{\boldsymbol{\lambda}}_i = -\bar{P}_i\boldsymbol{\theta}_i - \widehat{T_i(k)^{-1}}\widehat{\tilde{\boldsymbol{s}}}_i(k)$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible
We can estimate the inverse of T, by using the estimate of P tilde and the nominal value of P like in this equation
With the estimate of the inverse of T and the estimate of s tilde, we can reconstruct the original lambda

Mitigation mechanism

- Main idea: Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

Assumption

We suppose $\tilde{\boldsymbol{\lambda}}_i = \mathbf{0}$ only if $\boldsymbol{\lambda}_i = \mathbf{0}$, which implies $T_i(k)$ invertible.

- Estimate the inverse of $T_i(k)$

$$\widehat{T_i(k)^{-1}} = \bar{P}_i \, \widehat{\tilde{P}_i}(k)^{-1}$$

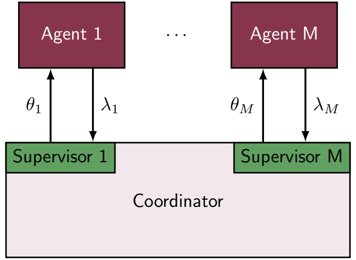
- Reconstruct $\boldsymbol{\lambda}_i$

$$\boldsymbol{\lambda}_{i\mathrm{rec}} = \widehat{T_i(k)^{-1}} \tilde{\boldsymbol{\lambda}}_i = -\bar{P}_i \boldsymbol{\theta}_i - \widehat{T_i(k)^{-1}} \hat{\tilde{\boldsymbol{s}}}_i(k)$$

CentraleSupélec

Now, for the mitigation the main idea is to reconstruct the original lambda from the estimated parameters and use it in the negotiation
We suppose the agent wants to decrease its cost, and the lambda sent is only equal to zero if the original is equal to zero. which implies the matrix T is invertible
We can estimate the inverse of T, by using the estimate of P tilde and the nominal value of P like in this equation
With the estimate of the inverse of T and the estimate of s tilde, we can reconstruct the original lambda

Complete Mechanism

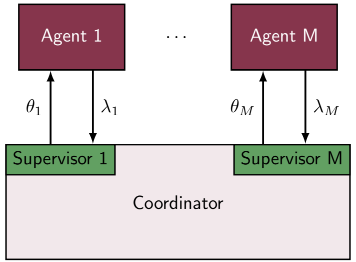


Two phases:

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator

Complete Mechanism

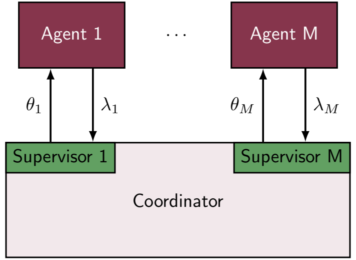


Two phases:

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases,

Complete Mechanism

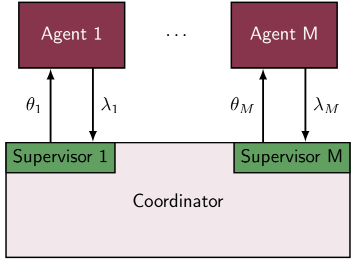


Two phases:

1. Detect which agents are non-cooperative
2. Reconstruct λ_i and use in negotiation

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases, first we detect which agents are non-cooperative

Complete Mechanism

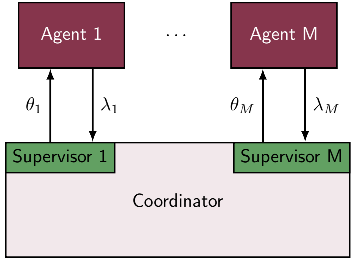


Two phases:

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

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases, first we detect which agents are non-cooperative and then reconstruct the lambda i s and use in the usual negotiation

Complete Mechanism



Two phases:

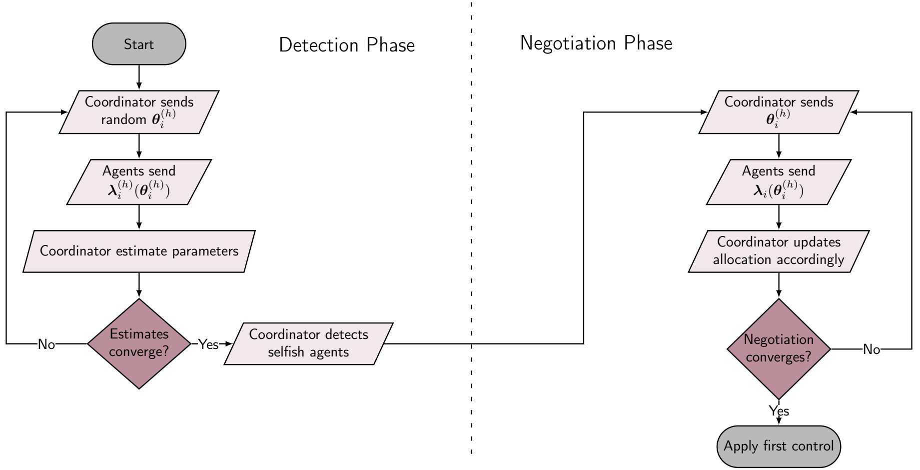
1. Detect which agents are non-cooperative
2. Reconstruct $\boldsymbol{\lambda}_i$ and use in negotiation

CentraleSupélec

The complete mechanism is equivalent to add a supervisor for each agent inside the coordinator
The mechanism is divided into the two phases, first we detect which agents are non-cooperative and then reconstruct the lambda i s and use in the usual negotiation

Complete algorithm

RPdMPC-DS

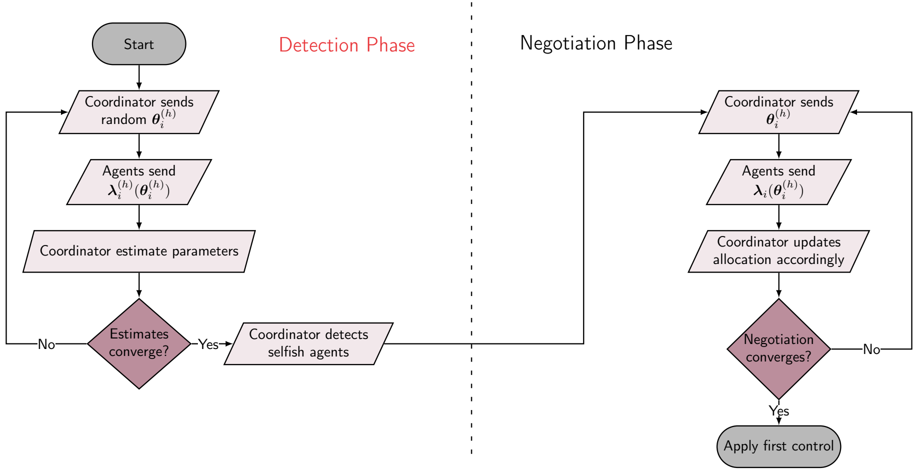


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases

Complete algorithm

RPdMPC-DS

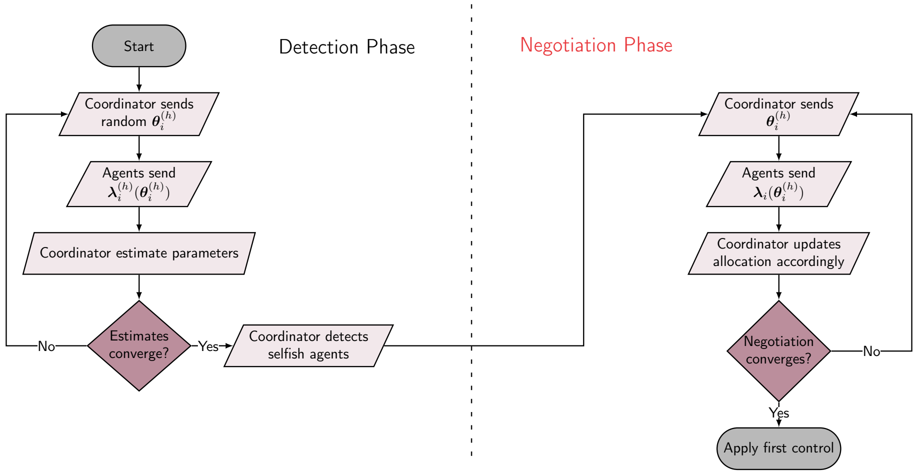


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase

Complete algorithm

RPdMPC-DS

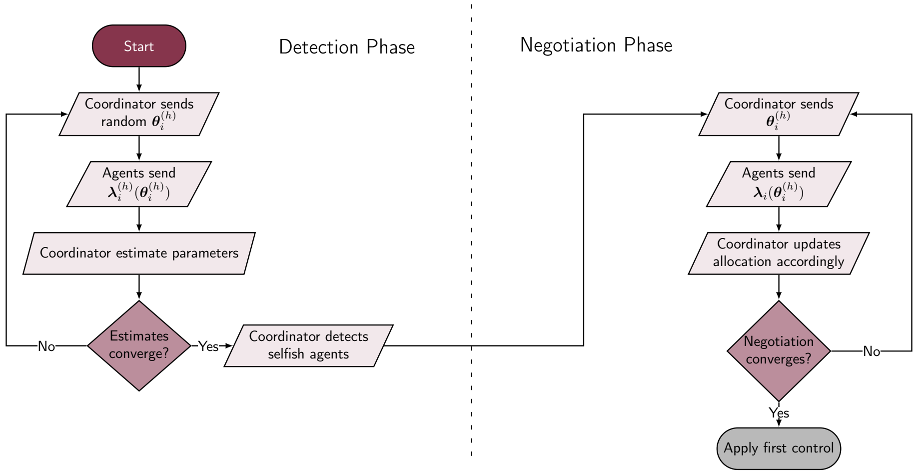


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

RPdMPC-DS

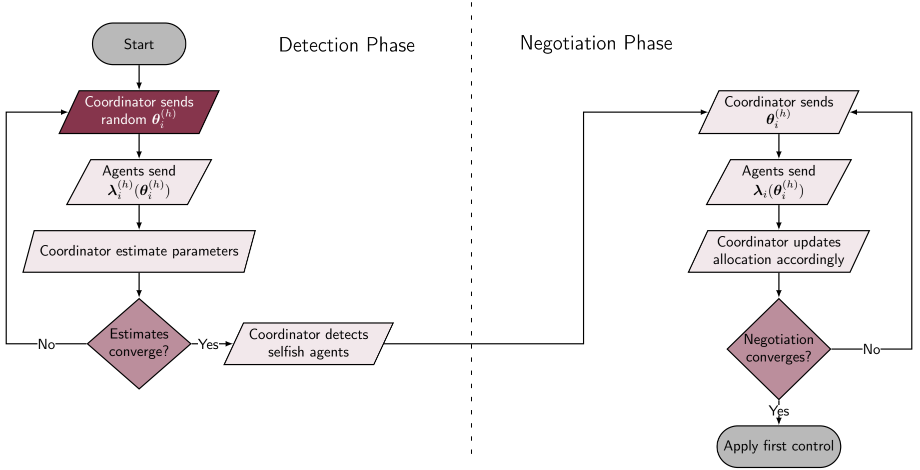


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

RPdMPC-DS

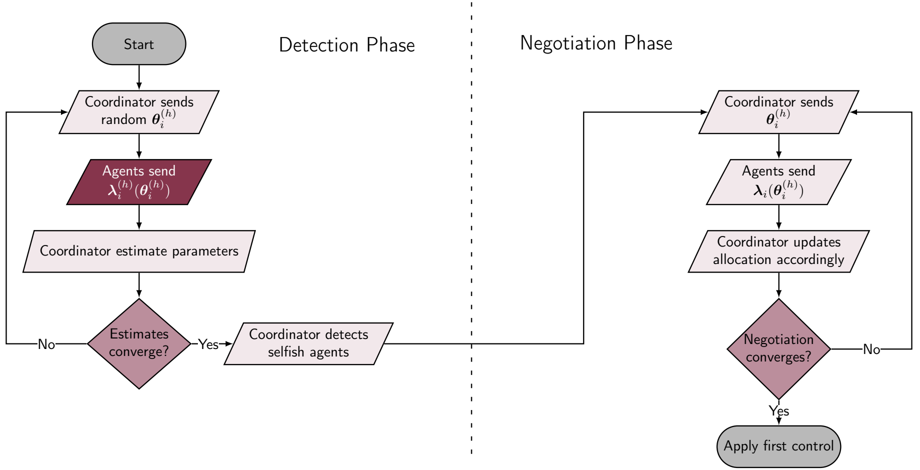


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i

Complete algorithm

RPdMPC-DS

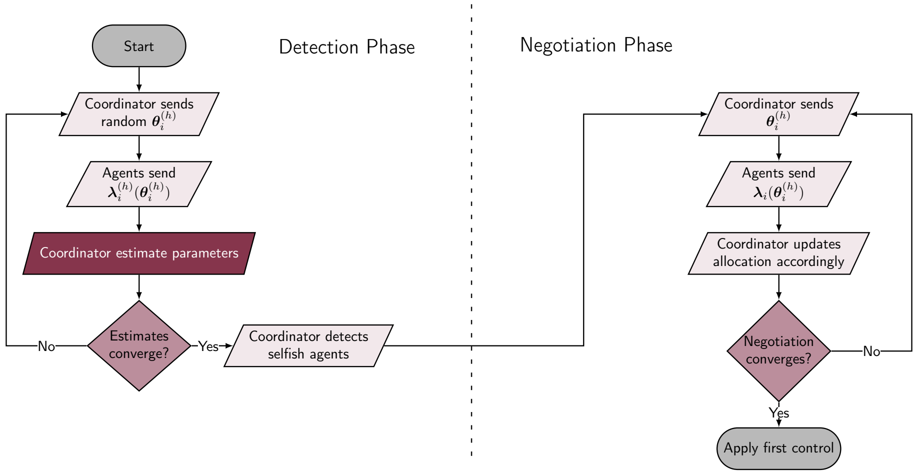


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i

Complete algorithm

RPdMPC-DS

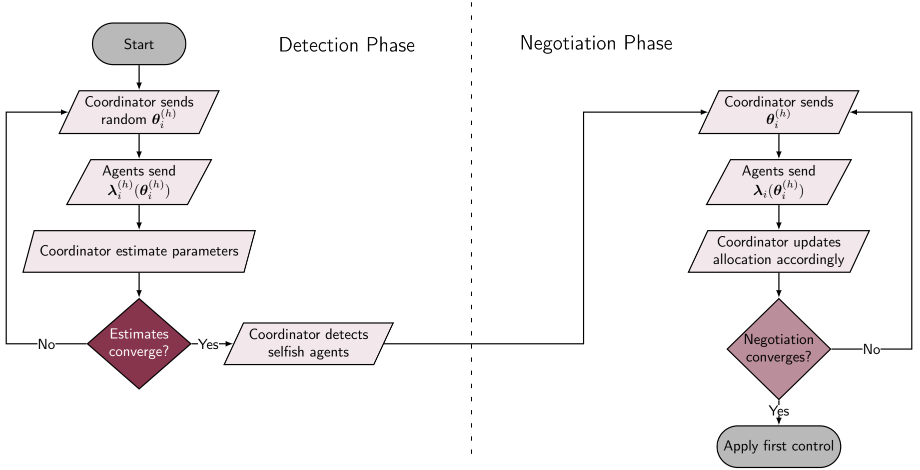


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde

Complete algorithm

RPdMPC-DS

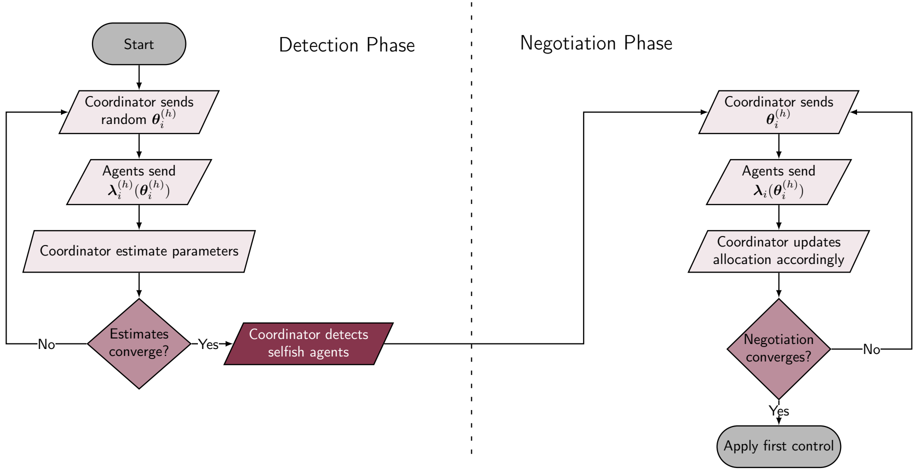


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge

Complete algorithm

RPdMPC-DS

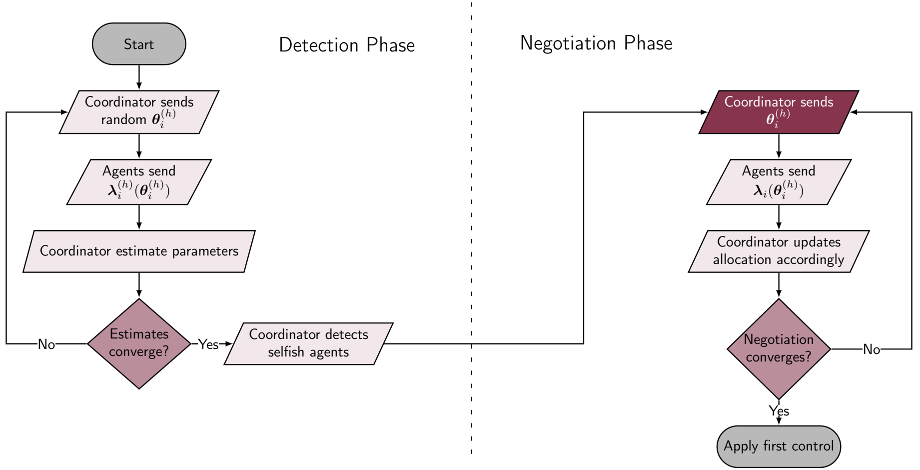


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative

Complete algorithm

RPdMPC-DS

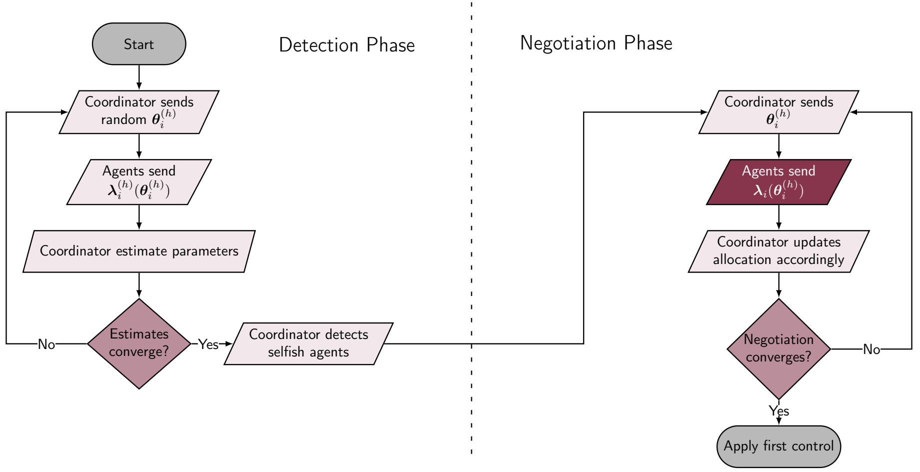


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i

Complete algorithm

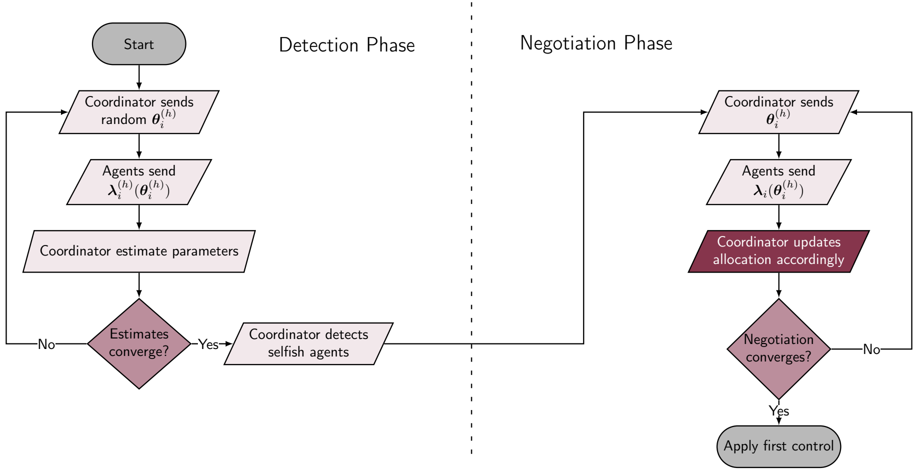
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i

Complete algorithm

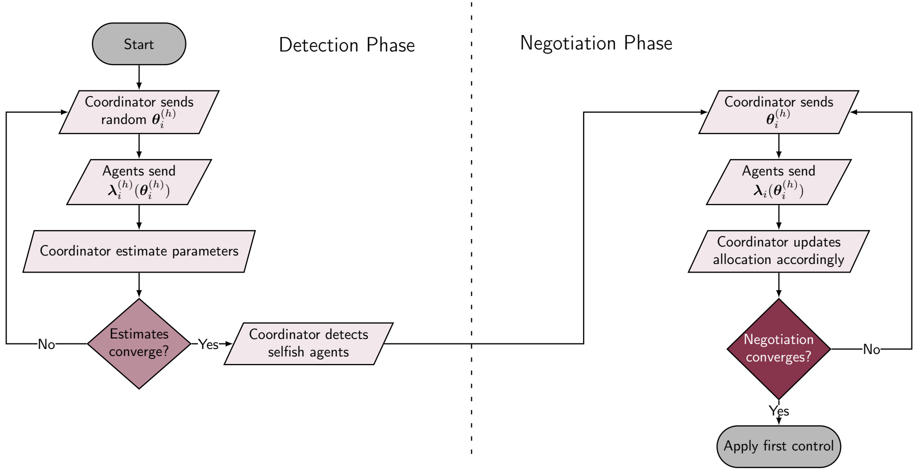
RPdMPC-DS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent

Complete algorithm

RPdMPC-DS

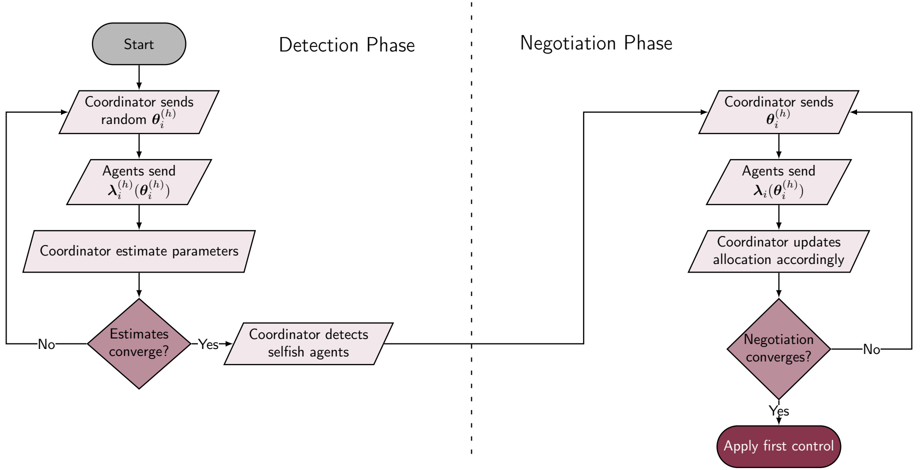


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges

Complete algorithm

RPdMPC-DS

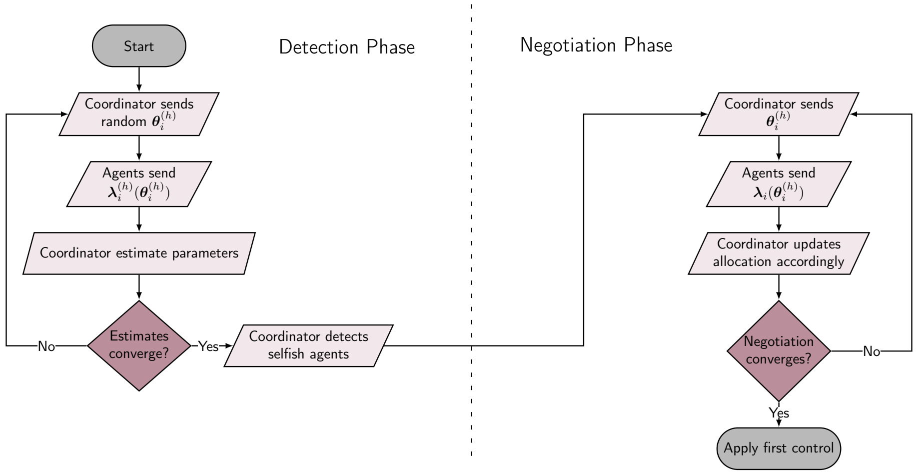


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges
each agent applies the first control

Complete algorithm

RPdMPC-DS



CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges
each agent applies the first control

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under $20^o\mathrm{C}$
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\mathrm{kW}\right)$
- Simulated for a period of $5\mathrm{h}$
- ZOH $T_s = 0.25\mathrm{h}$
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under 20^oC
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\text{kW}\right)$
- Simulated for a period of 5h
- ZOH $T_s = 0.25\text{h}$
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under 20^oC
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\mathrm{kW}\right)$
- Simulated for a period of $5\mathrm{h}$
- ZOH $T_s = 0.25\mathrm{h}$
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under 20^oC
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\text{kW}\right)$
- Simulated for a period of 5h
- ZOH $T_s = 0.25$h
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under $20^o\mathrm{C}$
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\mathrm{kW}\right)$
- Simulated for a period of $5\mathrm{h}$
- ZOH $T_s = 0.25\mathrm{h}$
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under $20^o\mathrm{C}$
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\mathrm{kW}\right)$
- Simulated for a period of $5\mathrm{h}$
- ZOH $T_s = 0.25\mathrm{h}$
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h
with sampling time of 15 minutes

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under $20^o\mathrm{C}$
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\mathrm{kW}\right)$
- Simulated for a period of $5\mathrm{h}$
- ZOH $T_s = 0.25\mathrm{h}$
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

CentraleSupélec

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h
with sampling time of 15 minutes
3 scenarios are simulated, the nominal, one where agent 1 presents non cooperative behavior from k greater than 6, and another with the selfish behavior and the secure mechanism activated

Example

District Heating of 4 Distinct Houses Under Power Scarcity

- 4 distinct rooms modeled using 3R-2C
- Initial temperature under $20^{o}\mathrm{C}$
- Not enough power to achieve setpoint $\left(\sum_{i=1}^{4} \boldsymbol{u}_i(k) \leqslant 4\mathrm{kW}\right)$
- Simulated for a period of $5\mathrm{h}$
- ZOH $T_s = 0.25\mathrm{h}$
- 3 scenarios
  1. Nominal
  2. Agent I non cooperative from k>6 with T=4*I
  3. Similar but with secure algorithm

We give an academic example of the temperature control of 4 distinct rooms under power scarcity
The 4 rooms are distinct using the 3 resistor 2 capacitor model
the initial temperature of all rooms is under 20 degrees celsius, which is the final setpoint
But they are under power scarcity that prevents them from reaching the setpoint
we simulate for a period of 5h
with sampling time of 15 minutes
3 scenarios are simulated, the nominal, one where agent 1 presents non cooperative behavior from k greater than 6, and another with the selfish behavior and the secure mechanism activated

Results

Temporal

- N Nominal
- S Selflish behavior
- C selfish behavior with Correction

In the figure we see the air temperature and the estimation error for room 1

Results

Temporal

N Nominal

S Selflish behavior

C selfish behavior with Correction

CentraleSupélec

In the figure we see the air temperature and the estimation error for room 1
The nominal behavior is in orange, and as said it cannot reach the setpoint, in blue

Results

Temporal

N Nominal

S Selflish behavior

C selfish behavior with Correction

In the figure we see the air temperature and the estimation error for room 1
The nominal behavior is in orange, and as said it cannot reach the setpoint, in blue
When the room presents selfish behavior (in blue), it reduces its cost and get closer to the setpoint, we see that the attack increases the error

Results

Temporal

- N Nominal
- S Selflish behavior
- C selfish behavior with Correction

CentraleSupélec

In the figure we see the air temperature and the estimation error for room 1
The nominal behavior is in orange, and as said it cannot reach the setpoint, in blue
When the room presents selfish behavior (in blue), it reduces its cost and get closer to the setpoint, we see that the attack increases the error
Now, for the case with correction (the red dots), even if it attacks the system, the temperature is close to its nominal value

Results

Temporal

N Nominal

S Selfish behavior

C selfish behavior with Correction

In the figure we see the air temperature and the estimation error for room 1
The nominal behavior is in orange, and as said it cannot reach the setpoint, in blue
When the room presents selfish behavior (in blue), it reduces its cost and get closer to the setpoint, we see that the attack increases the error
Now, for the case with correction (the red dots), even if it attacks the system, the temperature is close to its nominal value

Results

Table 1: Comparison of costs J_i^N and J_G^N

Agent	Nominal	Selfish	Selfish + correction
I	103	64	104
II	73	91	73
III	100	123	101
IV	132	154	131
Global	408	442	409

CentraleSupélec

Now, if we compare the costs for each scenario we see how the cost of agent 1 decreases when it attacks, while the cost of other agents increase.

Results

Table 1: Comparison of costs J_i^N and J_G^N

Agent	Nominal	Selfish	Selfish + correction
I	103	64	104
II	73	91	73
III	100	123	101
IV	132	154	131
Global	408	442	409

CentraleSupélec

Now, if we compare the costs for each scenario we see how the cost of agent 1 decreases when it attacks, while the cost of other agents increase.
When the secure algorithm is activated the costs are very close to the original ones. So

Results

Table 1: Comparison of costs J_i^N and J_G^N

Agent	Nominal	Selfish	Selfish + correction
I	103	64	104
II	73	91	73
III	100	123	101
IV	132	154	131
Global	408	442	409

CentraleSupélec

Now, if we compare the costs for each scenario we see how the cost of agent 1 decreases when it attacks, while the cost of other agents increase.
When the secure algorithm is activated the costs are very close to the original ones. So

Outline

CentraleSupélec

Analysing

$$\boldsymbol{\lambda}_i[k] = \begin{cases} -P_i^{(0)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(0)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}_{\boldsymbol{\lambda}_i}^{n} \\ \vdots & \vdots \\ -P_i^{(2^{n_{\text{ineq}}}-1)}\boldsymbol{\theta}_i[k] - \boldsymbol{s}_i^{(2^{n_{\text{ineq}}}-1)}[k], & \text{if } \boldsymbol{\theta}_i[k] \in \mathcal{R}_{\boldsymbol{\lambda}_i}^{2^{n_{\text{ineq}}}-1} \end{cases}. \quad (1)$$

CentraleSupélec

Artificial Scarcity

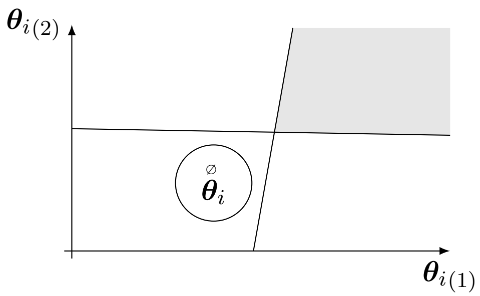


Figure 1: Ball $\mathcal{B}(\overset{\varnothing}{\boldsymbol{\theta}}_i, r)$.

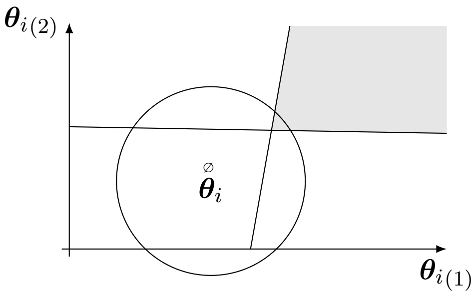


Figure 2: Ball $\mathcal{B}(\overset{\varnothing}{\boldsymbol{\theta}}_i, r)$ traversing zones.

Expectation Maximization

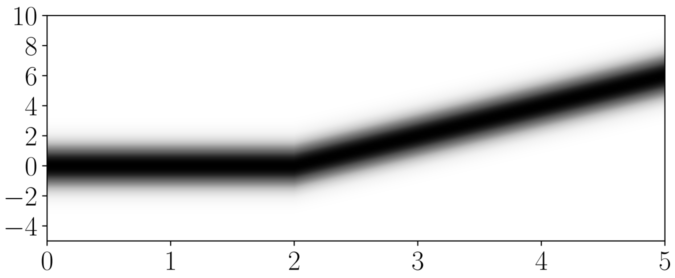


Figure 3: Gaussian Mixture for a 1D PWA function with 2 modes.

Expectation Maximization

Algorithm

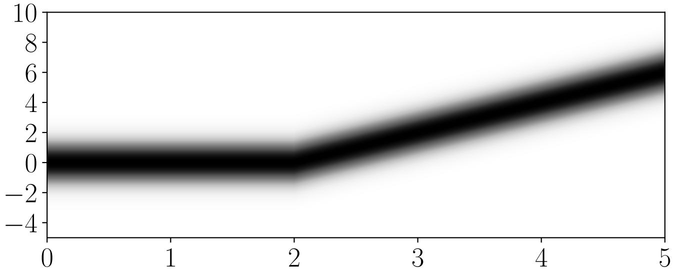
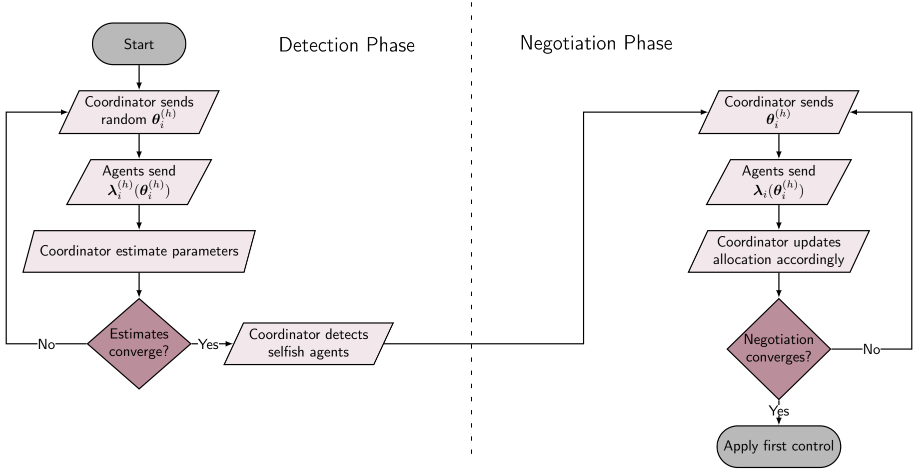


Figure 4: Gaussian Mixture for a 1D PWA function with 2 modes.

Complete algorithm

RPdMPC-AS

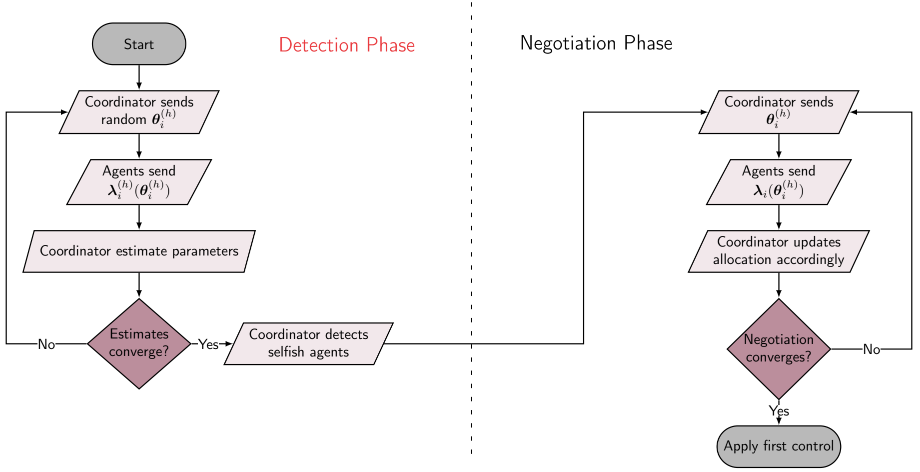


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases

Complete algorithm

RPdMPC-AS

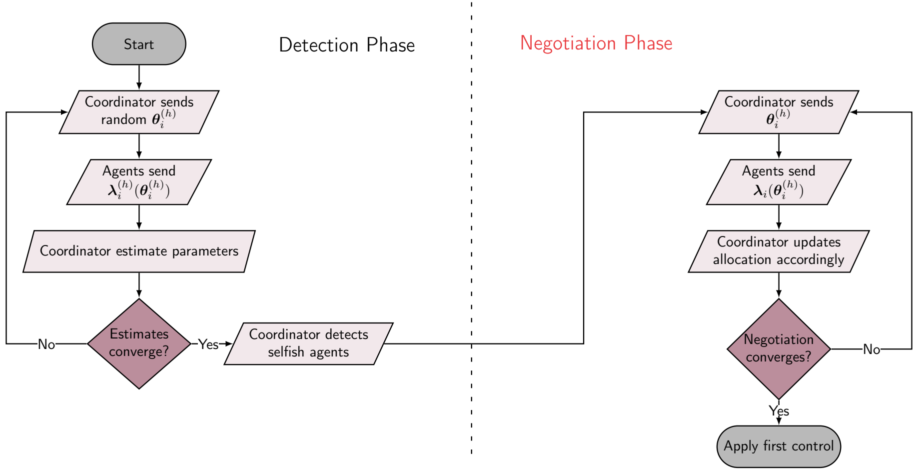


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase

Complete algorithm

RPdMPC-AS

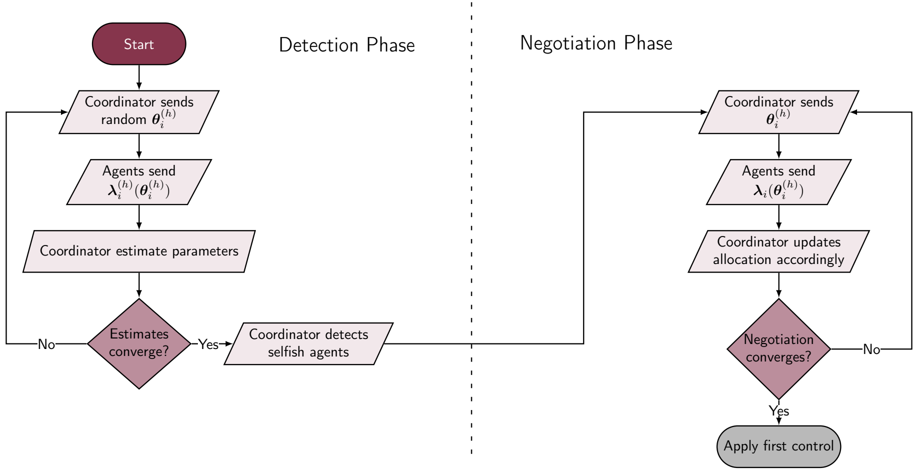


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

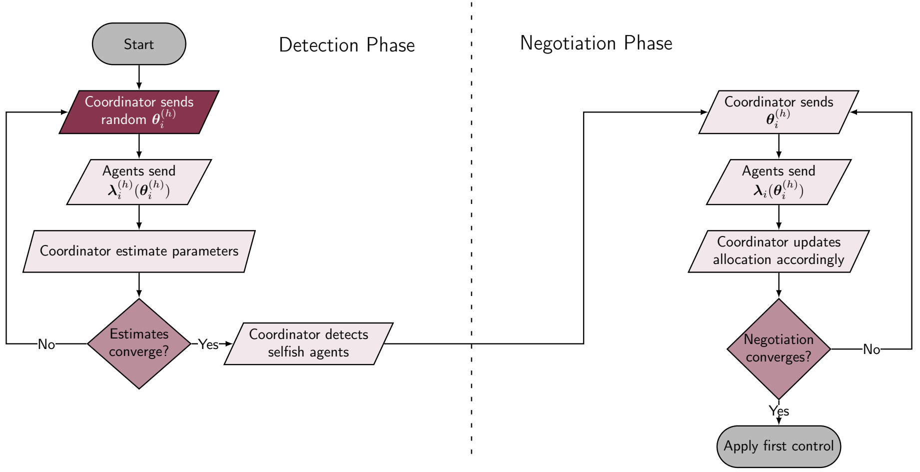
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase

Complete algorithm

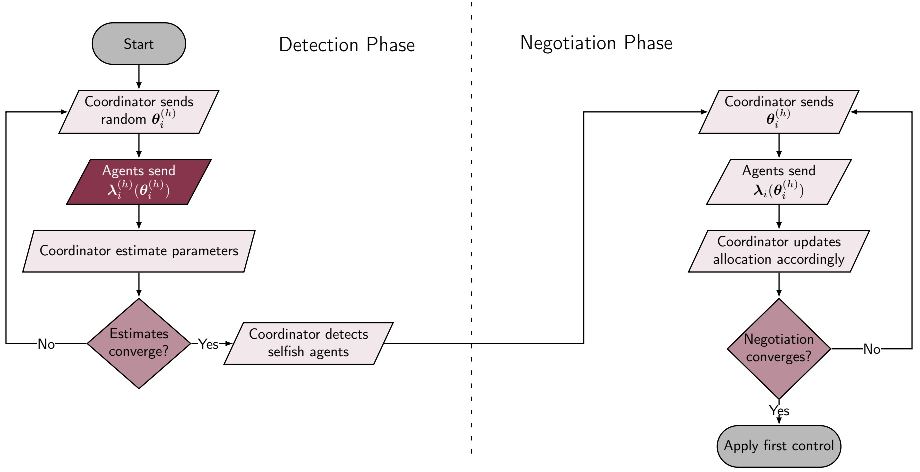
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i

Complete algorithm

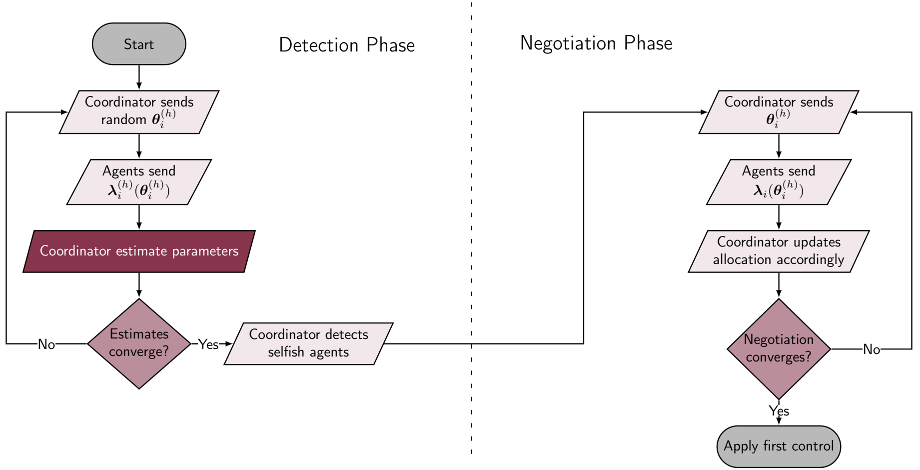
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i

Complete algorithm

RPdMPC-AS

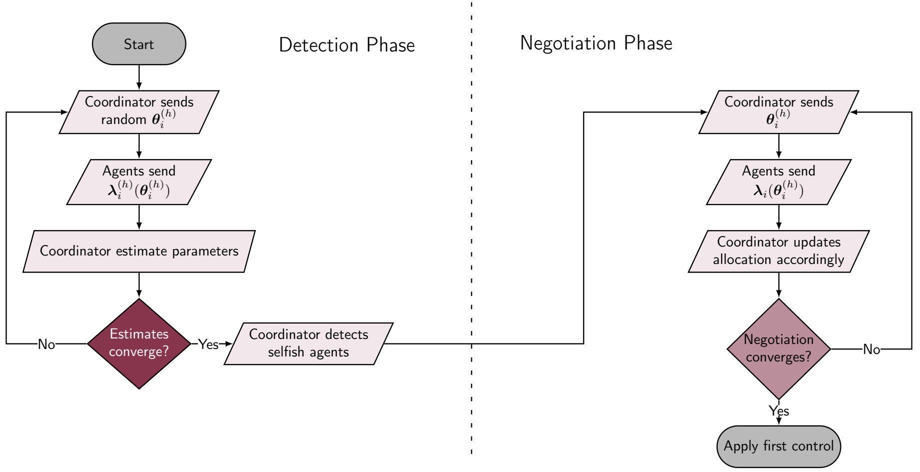


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde

Complete algorithm

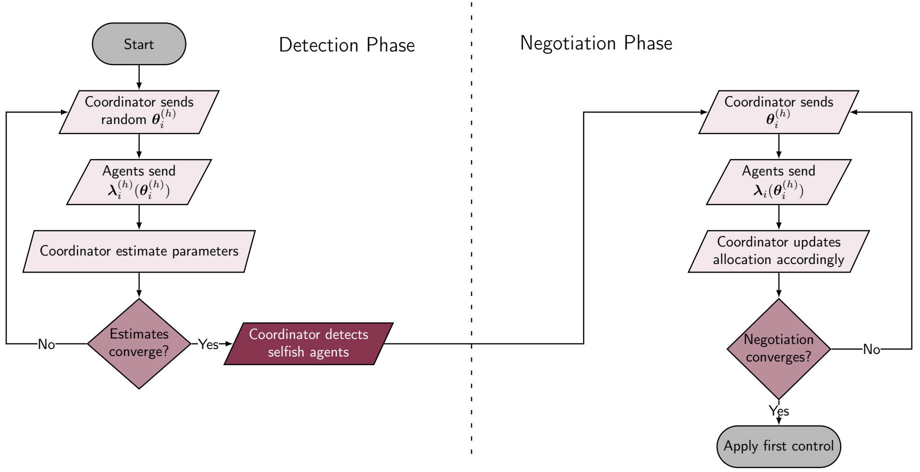
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge

Complete algorithm

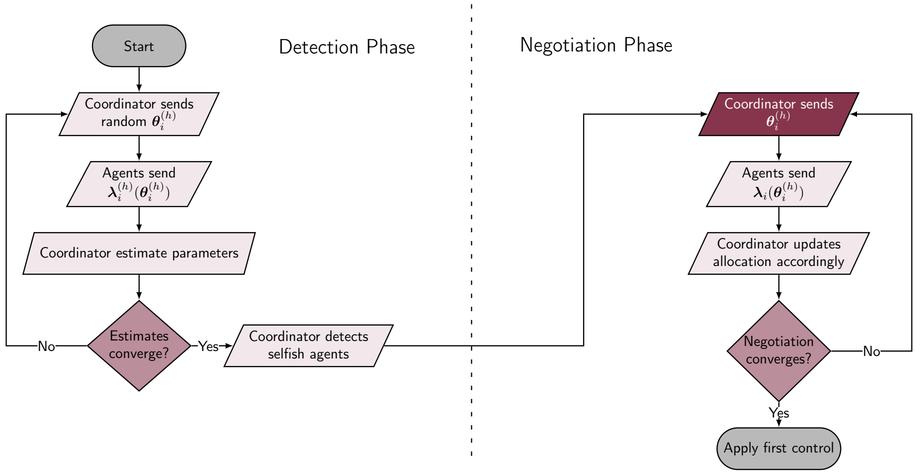
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative

Complete algorithm

RPdMPC-AS

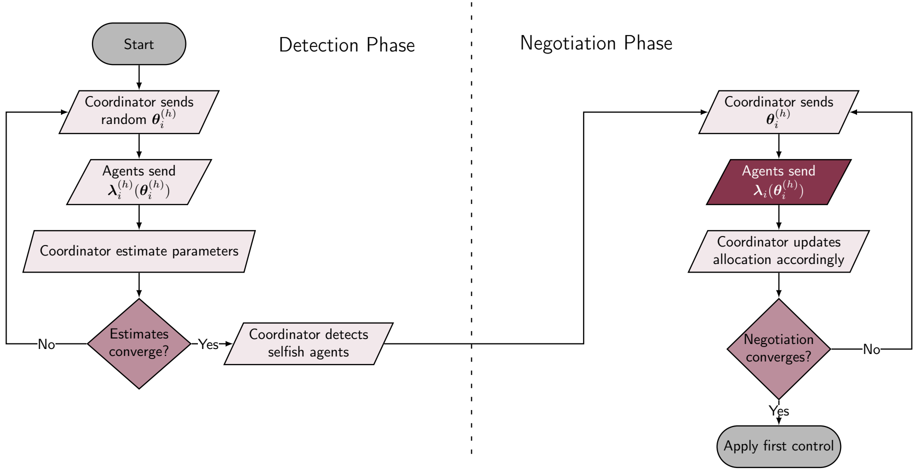


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i

Complete algorithm

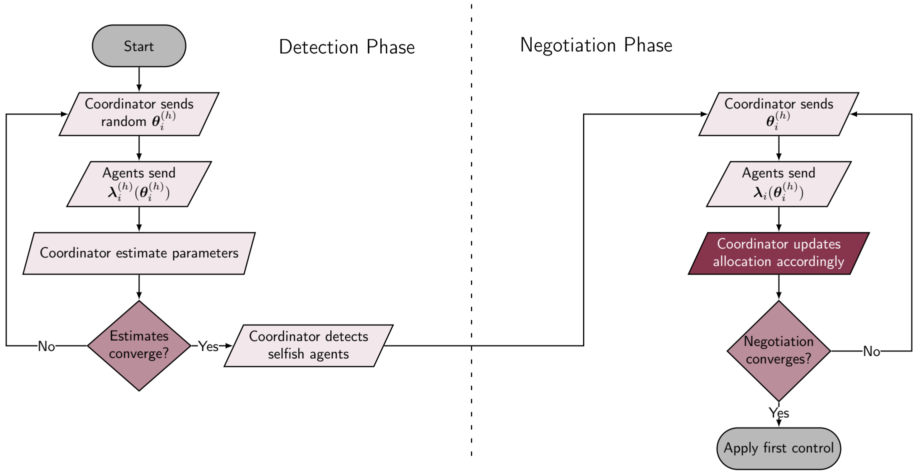
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i

Complete algorithm

RPdMPC-AS

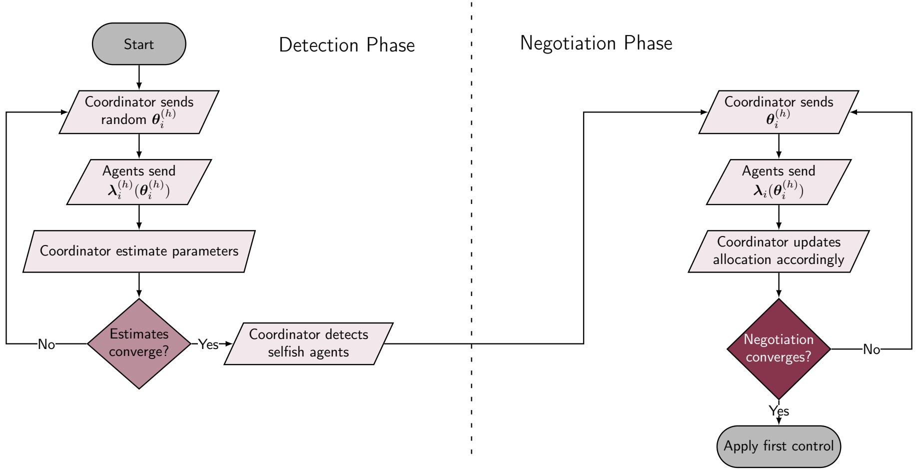


CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent

Complete algorithm

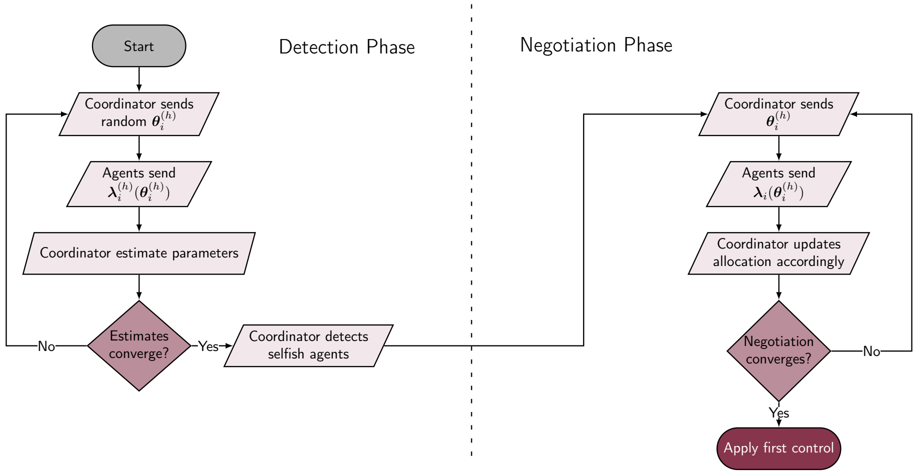
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges

Complete algorithm

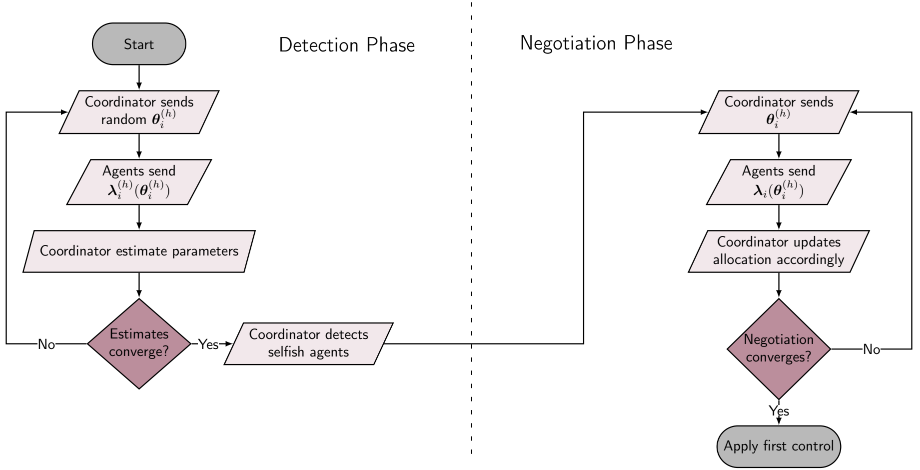
RPdMPC-AS



Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges
each agent applies the first control

Complete algorithm

RPdMPC-AS



CentraleSupélec

Now, for the complete secure DMPC algorithm, as said it is divided into two phases
The Detection phase
and the negotiation phase
The coordinator sends random theta i
The agents send dual variable lambda i
The coordinator estimates the parameters P and s tilde
when the estimates converge
The coordinator detects which agents are non-cooperative
then the negotiation phase begins, the coordinator sends the theta i
the agents send the dual variable lambda i
and the coordinator updates the allocation accordingly using the reconstructed lambda or the one sent by the agent
and once the negotiation converges
each agent applies the first control

Thank you!

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

Contact
rafael.accacio.nogueira@gmail.com

If you want to see the simulations of this paper we have a github repository, and if you want to send me an email about this paper or this presentation you can flash the QR code in the right. Thank you!

For Further Reading I

José M Maestre, Rudy R Negenborn, et al. *Distributed Model Predictive Control made easy*. Vol. 69. Springer, 2014. ISBN: 978-94-007-7005-8.

Wicak Ananduta et al. "Resilient Distributed Model Predictive Control for Energy Management of Interconnected Microgrids". In: *Optimal Control Applications and Methods* 41.1 (2020), pp. 146–169. DOI: 10.1002/oca.2534. URL: https://onlinelibrary.wiley.com/doi/pdf/10.1002/oca.2534.

José M. Maestre et al. "Scenario-Based Defense Mechanism Against Vulnerabilities in Lagrange-Based Dmpc". In: *Control Eng Pract* 114 (2021), p. 104879. ISSN: 0967-0661. DOI: 10.1016/j.conengprac.2021.104879.

CentraleSupélec

As a recommended reading I give a book about distributed MPC and an article with another secure dmpc algorithm based in another decomposition method. That's all

For Further Reading II

Pablo Velarde et al. "Vulnerabilities in Lagrange-Based Distributed Model Predictive Control". In: *Optimal Control Applications and Methods* 39.2 (Sept. 2018), pp. 601–621. DOI: 10.1002/oca.2368.

Wicak Ananduta et al. "Resilient Distributed Energy Management for Systems of Interconnected Microgrids". In: *2018 IEEE Conference on Decision and Control (CDC)*. 2018, pp. 3159–3164. DOI: 10.1109/CDC.2018.8619548.

Wicak Ananduta et al. "A Resilient Approach for Distributed MPC-Based Economic Dispatch in Interconnected Microgrids". In: *2019 18th European Control Conference (ECC)*. 2019, pp. 691–696. DOI: 10.23919/ECC.2019.8796208.

CentraleSupélec

For Further Reading III

P. Chanfreut, J. M. Maestre, and H. Ishii. “Vulnerabilities in Distributed Model Predictive Control based on Jacobi-Gauss Decomposition”. In: *2018 European Control Conference (ECC)*. June 2018, pp. 2587–2592. DOI: 10.23919/ECC.2018.8550239.

Pablo Velarde et al. “Scenario-based defense mechanism for distributed model predictive control”. In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE. Dec. 2017, pp. 6171–6176. DOI: 10.1109/CDC.2017.8264590.

Pablo Velarde et al. “Vulnerabilities in Lagrange-Based DMPC in the Context of Cyber-Security”. In: *2017 IEEE International Conference on Autonomic Computing (ICAC)*. July 2017, pp. 215–220. DOI: 10.1109/ICAC.2017.53.

CentraleSupélec

Conditions

back

One way to ensure this, is to make the original constraint (??) to have at most as many rows as columns, i.e., $\# \boldsymbol{u}_{\max} \leqslant n_u$, although it may be a little restrictive.

θ dynamics

back

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

where

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

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

CentraleSupélec