Learning Against Uncertainties

Mazen Alamir

CNRS - University of Grenoble-Alpes

Disclaimer

- The following material **is not supposed to be fully understood!**
- Only its **diversity and wide scope** is to be taken home!
- The major part of the concepts and tools will be **shortly outdated**!
- . . . Unless they already are!
- **Conjecture**: the corresponding state of mind **will prevail!**

Disclaimer

- The following material **is not supposed to be fully understood!**
- Only its **diversity and wide scope** is to be taken home!
- The major part of the concepts and tools will be **shortly outdated**!
- . . . Unless they already are!
- **Conjecture**: the corresponding state of mind **will prevail!**

Disclaimer

- The following material **is not supposed to be fully understood!**
- Only its **diversity and wide scope** is to be taken home!
- The major part of the concepts and tools will be **shortly outdated**!
- . . . Unless they already are!
- **Conjecture**: the corresponding state of mind **will prevail!**

Disclaimer

- The following material **is not supposed to be fully understood!**
- Only its **diversity and wide scope** is to be taken home!
- The major part of the concepts and tools will be **shortly outdated**!
- . . . Unless they already are!
- **Conjecture**: the corresponding state of mind **will prevail!**

Disclaimer

- The following material **is not supposed to be fully understood!**
- Only its **diversity and wide scope** is to be taken home!
- The major part of the concepts and tools will be **shortly outdated**!
- . . . Unless they already are!
- **Conjecture**: the corresponding state of mind **will prevail!**

Disclaimer

- The following material **is not supposed to be fully understood!**
- Only its **diversity and wide scope** is to be taken home!
- The major part of the concepts and tools will be **shortly outdated**!
- . . . Unless they already are!
- **Conjecture**: the corresponding state of mind **will prevail!**

Outline

1. Data-driven partial observability assessment under uncertainties
2. Learning-Based Approximate Stochastic NMPC Design
3. Tractable stochastic NMPC by supervised clustering
4. Data-Driven NMPC by cost function identification
5. Learning-Based Monitoring updating period in Real-time NMPC

1 Data-driven partial observability assessment under uncertainties

2 Learning-Based Approximate Stochastic NMPC Design

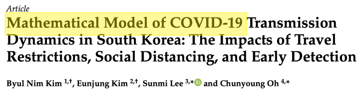
3 Tractable stochastic NMPC by supervised clustering

4 Data-Driven NMPC by cost function identification

5 Learning-Based Monitoring updating period in Real-time NMPC

What is it about?

(Parametric estimation example)



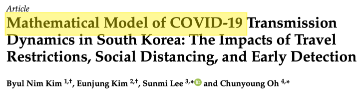
Article

Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of Travel Restrictions, Social Distancing, and Early Detection

Byul Nim Kim [1,†], Eunjung Kim [2,†], Sunmi Lee [3,*] and Chunyoung Oh [4,*]

What is it about?

(Parametric estimation example)

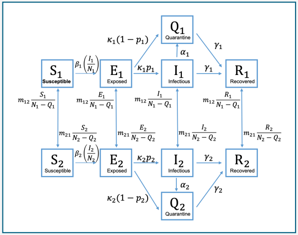


Article

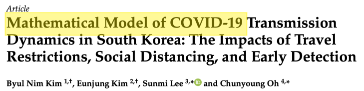
Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of Travel Restrictions, Social Distancing, and Early Detection

Byul Nim Kim [1,†], Eunjung Kim [2,†], Sunmi Lee [3,*] and Chunyoung Oh [4,*]

Model's diagram



What is it about? (Parametric estimation example)

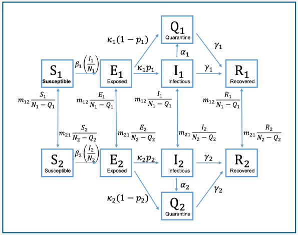


Article

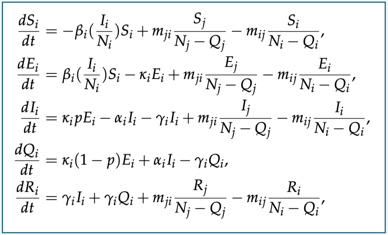
Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of Travel Restrictions, Social Distancing, and Early Detection

Byul Nim Kim [1,†], Eunjung Kim [2,†], Sunmi Lee [3,*] and Chunyoung Oh [4,*]

Model's diagram



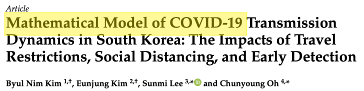
Model's equations



$$\frac{dS_i}{dt} = -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},$$
$$\frac{dE_i}{dt} = \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},$$
$$\frac{dI_i}{dt} = \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},$$
$$\frac{dQ_i}{dt} = \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,$$
$$\frac{dR_i}{dt} = \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},$$

What is it about?

(Parametric estimation example)

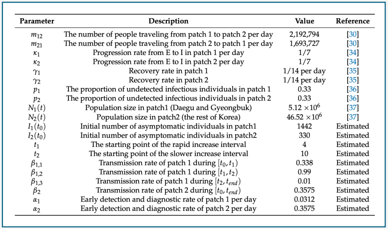


Article

Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of Travel Restrictions, Social Distancing, and Early Detection

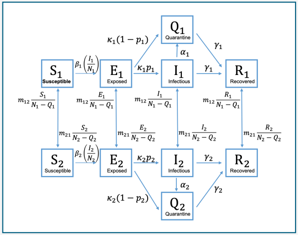
Byul Nim Kim [1,†], Eunjung Kim [2,†], Sunmi Lee [3,*] and Chunyoung Oh [4,*]

Model's parameters



Parameter	Description	Value	Reference
m_{12}	The number of people traveling from patch 1 to patch 2 per day	2,192,794	[30]
m_{21}	The number of people traveling from patch 2 to patch 1 per day	1,693,727	[30]
κ_1	Progression rate from E to I in patch 1 per day	1/7	[34]
κ_2	Progression rate from E to I in patch 2 per day	1/7	[34]
γ_1	Recovery rate in patch 1	1/14 per day	[35]
γ_2	Recovery rate in patch 2	1/14 per day	[35]
p_1	The proportion of undetected infectious individuals in patch 1	0.33	[36]
p_2	The proportion of undetected infectious individuals in patch 2	0.33	[36]
$N_1(t)$	Population size in patch1 (Daegu and Gyeongbuk)	5.12×10^6	[37]
$N_2(t)$	Population size in patch2 (the rest of Korea)	46.52×10^6	[37]
$I_1(t_0)$	Initial number of asymptomatic individuals in patch1	1442	Estimated
$I_2(t_0)$	Initial number of asymptomatic individuals in patch2	330	Estimated
t_1	The starting point of the rapid increase interval	4	Estimated
t_2	The starting point of the slower increase interval	10	Estimated
$\beta_{1,1}$	Transmission rate of patch 1 during $[t_0, t_1)$	0.338	Estimated
$\beta_{1,2}$	Transmission rate of patch 1 during $[t_1, t_2)$	0.99	Estimated
$\beta_{1,3}$	Transmission rate of patch 1 during $[t_2, t_{end})$	0.01	Estimated
β_2	Transmission rate of patch 2 during $[t_0, t_{end})$	0.3575	Estimated
α_1	Early detection and diagnostic rate of patch 1 per day	0.0312	Estimated
α_2	Early detection and diagnostic rate of patch 2 per day	0.3575	Estimated

Model's diagram

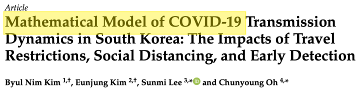


Model's equations

$$\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}$$

What is it about?

(Parametric estimation example)



Article

Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of Travel Restrictions, Social Distancing, and Early Detection

Byul Nim Kim 1,†, Eunjung Kim 2,†, Sunmi Lee 3,* and Chunyoung Oh 4,*

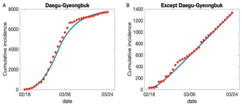
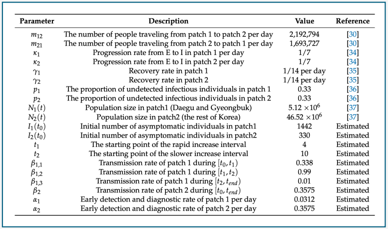


Figure 5. (**A**) Model calibration to Daegu-Gyeongbuk incidence data (blue solid line: model prediction, red dots: data. (**B**) Model calibration to South Korea incidence data except Daegu-Gyeongbuk area.

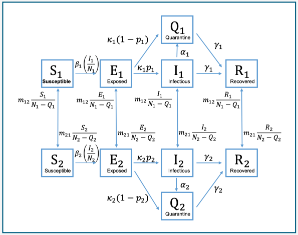
Model calibration

Model's parameters

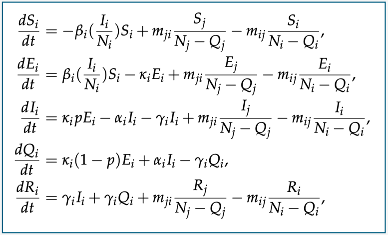


Parameter	Description	Value	Reference
m_{12}	The number of people traveling from patch 1 to patch 2 per day	2,192,794	[30]
m_{21}	The number of people traveling from patch 2 to patch 1 per day	1,693,727	[30]
κ_1	Progression rate from E to I in patch 1 per day	1/7	[34]
κ_2	Progression rate from E to I in patch 2 per day	1/7	[34]
γ_1	Recovery rate in patch 1	1/14 per day	[35]
γ_2	Recovery rate in patch 2	1/14 per day	[35]
p_1	The proportion of undetected infectious individuals in patch 1	0.33	[36]
p_2	The proportion of undetected infectious individuals in patch 2	0.33	[36]
$N_1(t)$	Population size in patch1 (Daegu and Gyeongbuk)	5.12×10^6	[37]
$N_2(t)$	Population size in patch2 (the rest of Korea)	46.52×10^6	[37]
$I_1(t_0)$	Initial number of asymptomatic individuals in patch1	1442	Estimated
$I_2(t_0)$	Initial number of asymptomatic individuals in patch2	330	Estimated
t_1	The starting point of the rapid increase interval	4	Estimated
t_2	The starting point of the slower increase interval	10	Estimated
$\beta_{1,1}$	Transmission rate of patch 1 during $[t_0, t_1)$	0.338	Estimated
$\beta_{1,2}$	Transmission rate of patch 1 during $[t_1, t_2)$	0.99	Estimated
$\beta_{1,3}$	Transmission rate of patch 1 during $[t_2, t_{end})$	0.01	Estimated
β_2	Transmission rate of patch 2 during $[t_0, t_{end})$	0.3575	Estimated
α_1	Early detection and diagnostic rate of patch 1 per day	0.0312	Estimated
α_2	Early detection and diagnostic rate of patch 2 per day	0.3575	Estimated

Model's diagram



Model's equations



$$\frac{dS_i}{dt} = -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},$$

$$\frac{dE_i}{dt} = \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},$$

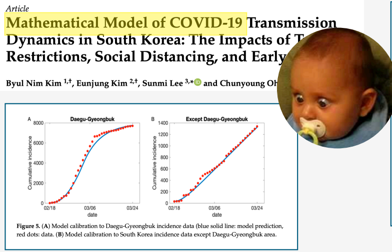
$$\frac{dI_i}{dt} = \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},$$

$$\frac{dQ_i}{dt} = \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,$$

$$\frac{dR_i}{dt} = \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},$$

What is it about?

(Parametric estimation example)



Article

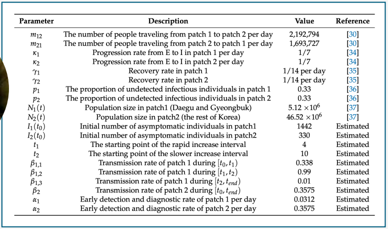
Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of T Restrictions, Social Distancing, and Early

Byul Nim Kim [1,†], Eunjung Kim [2,†], Sunmi Lee [3,*] and Chunyoung Oh

Figure 5. (**A**) Model calibration to Daegu-Gyeongbuk incidence data (blue solid line: model prediction, red dots: data. (**B**) Model calibration to South Korea incidence data except Daegu-Gyeongbuk area.

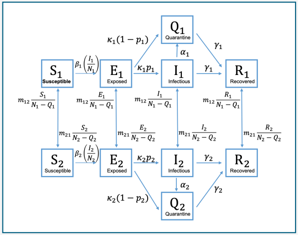
Model calibration

Model's parameters



Parameter	Description	Value	Reference
m_{12}	The number of people traveling from patch 1 to patch 2 per day	2,192,794	[30]
m_{21}	The number of people traveling from patch 2 to patch 1 per day	1,693,727	[30]
κ_1	Progression rate from E to I in patch 1 per day	1/7	[34]
κ_2	Progression rate from E to I in patch 2 per day	1/7	[34]
γ_1	Recovery rate in patch 1	1/14 per day	[35]
γ_2	Recovery rate in patch 2	1/14 per day	[35]
p_1	The proportion of undetected infectious individuals in patch 1	0.33	[36]
p_2	The proportion of undetected infectious individuals in patch 2	0.33	[36]
$N_1(t)$	Population size in patch1 (Daegu and Gyeongbuk)	5.12×10^6	[37]
$N_2(t)$	Population size in patch2 (the rest of Korea)	46.52×10^6	[37]
$I_1(t_0)$	Initial number of asymptomatic individuals in patch1	1442	Estimated
$I_2(t_0)$	Initial number of asymptomatic individuals in patch2	330	Estimated
t_1	The starting point of the rapid increase interval	4	Estimated
t_2	The starting point of the slower increase interval	10	Estimated
$\beta_{1,1}$	Transmission rate of patch 1 during $[t_0, t_1)$	0.338	Estimated
$\beta_{1,2}$	Transmission rate of patch 1 during $[t_1, t_2)$	0.99	Estimated
$\beta_{1,3}$	Transmission rate of patch 1 during $[t_2, t_{end})$	0.01	Estimated
β_2	Transmission rate of patch 2 during $[t_0, t_{end})$	0.3575	Estimated
α_1	Early detection and diagnostic rate of patch 1 per day	0.0312	Estimated
α_2	Early detection and diagnostic rate of patch 2 per day	0.3575	Estimated

Model's diagram

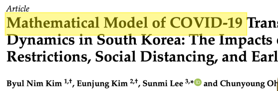


Model's equations

$$
\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}
$$

What is it about?

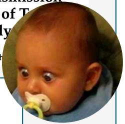
(Parametric estimation example)



Article

Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of T Restrictions, Social Distancing, and Early

Byul Nim Kim [1,†], Eunjung Kim [2,†], Sunmi Lee [3,*] and Chunyoung Oh



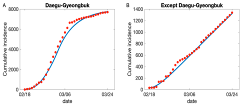
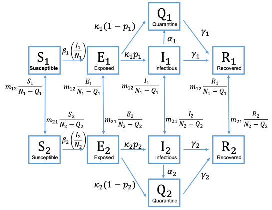


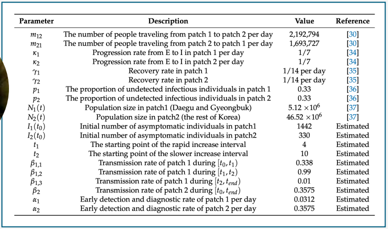
Figure 5. (**A**) Model calibration to Daegu-Gyeongbuk incidence data (blue solid line: model prediction, red dots: data. (**B**) Model calibration to South Korea incidence data except Daegu-Gyeongbuk area.

Model calibration

Model's diagram



Model's parameters



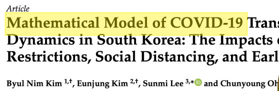
Parameter	Description	Value	Reference
m_{12}	The number of people traveling from patch 1 to patch 2 per day	2,192,794	[30]
m_{21}	The number of people traveling from patch 2 to patch 1 per day	1,693,727	[30]
κ_1	Progression rate from E to I in patch 1 per day	1/7	[34]
κ_2	Progression rate from E to I in patch 2 per day	1/7	[34]
γ_1	Recovery rate in patch 1	1/14 per day	[35]
γ_2	Recovery rate in patch 2	1/14 per day	[35]
p_1	The proportion of undetected infectious individuals in patch 1	0.33	[36]
p_2	The proportion of undetected infectious individuals in patch 2	0.33	[36]
$N_1(t)$	Population size in patch1 (Daegu and Gyeongbuk)	5.12×10^6	[37]
$N_2(t)$	Population size in patch2 (the rest of Korea)	46.52×10^6	[37]
$I_1(t_0)$	Initial number of asymptomatic individuals in patch1	1442	Estimated
$I_2(t_0)$	Initial number of asymptomatic individuals in patch2	330	Estimated
t_1	The starting point of the rapid increase interval	4	Estimated
t_2	The starting point of the slower increase interval	10	Estimated
$\beta_{1,1}$	Transmission rate of patch 1 during $[t_0, t_1)$	0.338	Estimated
$\beta_{1,2}$	Transmission rate of patch 1 during $[t_1, t_2)$	0.99	Estimated
$\beta_{1,3}$	Transmission rate of patch 1 during $[t_2, t_{end})$	0.01	Estimated
β_2	Transmission rate of patch 2 during $[t_0, t_{end})$	0.3575	Estimated
α_1	Early detection and diagnostic rate of patch 1 per day	0.0312	Estimated
α_2	Early detection and diagnostic rate of patch 2 per day	0.3575	Estimated

Data should be sufficiently rich to reveal the true values of the parameters. The more parameters you have, the richer data you need!

(NOTA) Richer $\neq$ high number

What is it about?

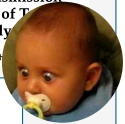
(Parametric estimation example)



Article

Mathematical Model of COVID-19 Transmission Dynamics in South Korea: The Impacts of T
Restrictions, Social Distancing, and Early

Byul Nim Kim [1,†], Eunjung Kim [2,†], Sunmi Lee [3,*] and Chunyoung Oh



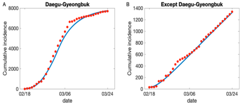
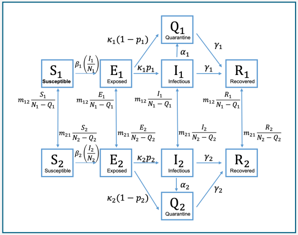


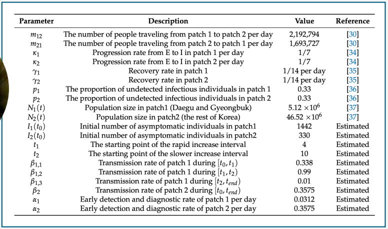
Figure 5. **(A)** Model calibration to Daegu-Gyeongbuk incidence data (blue solid line: model prediction, red dots: data. **(B)** Model calibration to South Korea incidence data except Daegu-Gyeongbuk area.

Model calibration

Model's diagram



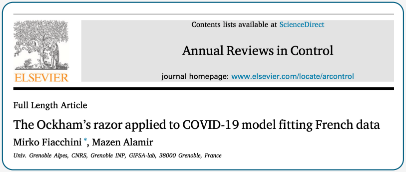
Model's parameters



Parameter	Description	Value	Reference
m_{12}	The number of people traveling from patch 1 to patch 2 per day	2,192,794	[30]
m_{21}	The number of people traveling from patch 2 to patch 1 per day	1,693,727	[30]
κ_1	Progression rate from E to I in patch 1 per day	1/7	[34]
κ_2	Progression rate from E to I in patch 2 per day	1/7	[34]
γ_1	Recovery rate in patch 1	1/14 per day	[35]
γ_2	Recovery rate in patch 2	1/14 per day	[35]
p_1	The proportion of undetected infectious individuals in patch 1	0.33	[36]
p_2	The proportion of undetected infectious individuals in patch 2	0.33	[36]
$N_1(t)$	Population size in patch1 (Daegu and Gyeongbuk)	5.12×10^6	[37]
$N_2(t)$	Population size in patch2 (the rest of Korea)	46.52×10^6	[37]
$I_1(t_0)$	Initial number of asymptomatic individuals in patch1	1442	Estimated
$I_2(t_0)$	Initial number of asymptomatic individuals in patch2	330	Estimated
t_1	The starting point of the rapid increase interval	4	Estimated
t_2	The starting point of the slower increase interval	10	Estimated
$\beta_{1,1}$	Transmission rate of patch 1 during $[t_0, t_1)$	0.338	Estimated
$\beta_{1,2}$	Transmission rate of patch 1 during $[t_1, t_2)$	0.99	Estimated
$\beta_{1,3}$	Transmission rate of patch 1 during $[t_2, t_{end})$	0.01	Estimated
β_2	Transmission rate of patch 2 during $[t_0, t_{end})$	0.3575	Estimated
α_1	Early detection and diagnostic rate of patch 1 per day	0.0312	Estimated
α_2	Early detection and diagnostic rate of patch 2 per day	0.3575	Estimated

Data should be sufficiently rich to reveal the true values of the parameters. The more parameters you have, the richer data you need!

(NOTA) Richer $\neq$ high number



Contents lists available at ScienceDirect

Annual Reviews in Control

journal homepage: www.elsevier.com/locate/arcontrol

Full Length Article

The Ockham's razor applied to COVID-19 model fitting French data

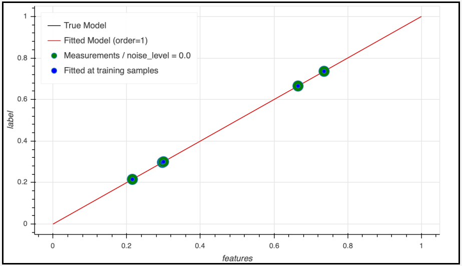
Mirko Fiacchini *, Mazen Alamir

Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France

What is it about? (A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0 / Order=1

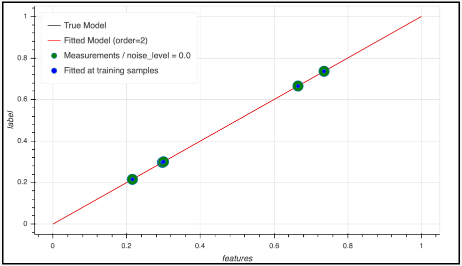


What is it about?

(A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0 / Order=2

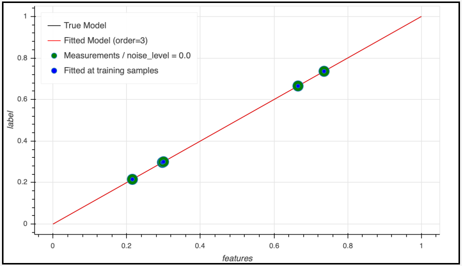


What is it about?

(A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0 / Order=3

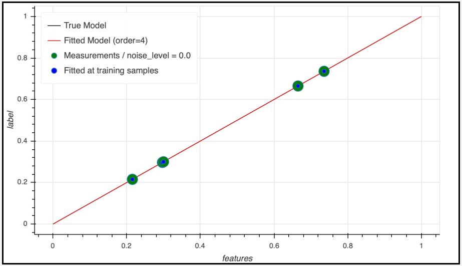


What is it about?

(A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0 / Order=4

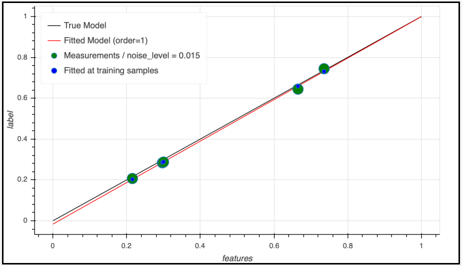


What is it about?

(A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0.015 / Order=1

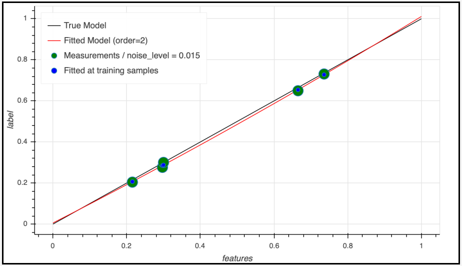


What is it about?

(A simple example)

- True model $y(t) = t$
- $\#$ of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0.015 / Order=2

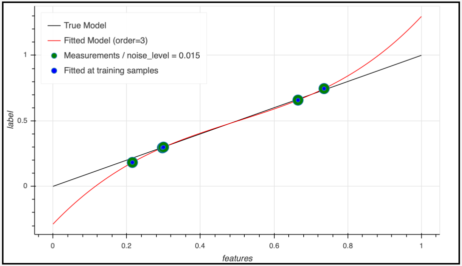


What is it about?

(A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0.015 / Order=3

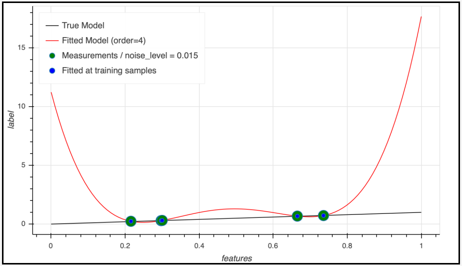


What is it about?

(A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

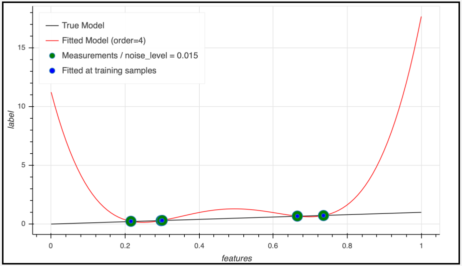
Noise=0.015 / Order=4



What is it about? (A simple example)

- True model $y(t) = t$
- # of measurement $n_{obs} = 5$
- Fit polynomials of $\neq$ degrees
- Training measurements
- Fitted values at training samples
- various noise levels

Noise=0.015 / Order=4



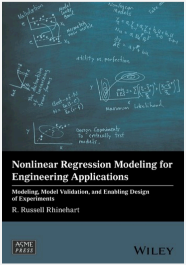
This phenomenon is called **Over-Fitting**

An over fitted model is dangerous because it can lead to very high prediction errors on unseen realizations.

(This can be avoided by assessing the model by cross-validation techniques)

What is it about?

(Away from identification textbooks)



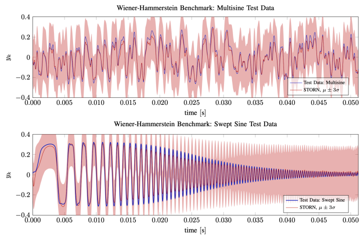
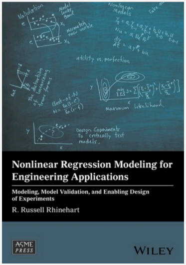


Fig. 12: Wiener Hammerstein benchmark: Time evaluation for multisine and swept sine test data set of best results from Tab. III, i.e. STORN with $h_{\text{dim}} = 40$, $z_{\text{dim}} = 3$, $n_{\text{layers}} = 3$.

- In identification text-books, the previous situation does not prevail.
- The ratio between the samples and the number of parameters is generally quite high.
- Mechatronic systems, electrical systems, . . . etc.
- Identification is done once for all
- Initial state importance is marginal

What is it about? (Away from identification textbooks)



Nonlinear Regression Modeling for Engineering Applications

Modeling, Model Validation, and Enabling Design of Experiments

R. Russell Rhinehart

ASME PRESS

WILEY

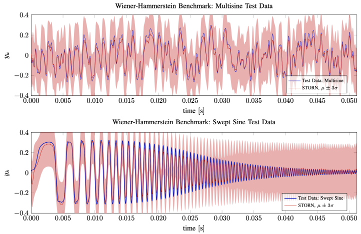
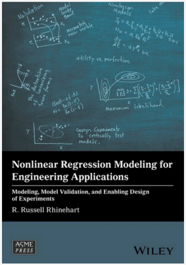


Fig. 12: Wiener Hammerstein benchmark: Time evaluation for multisine and swept sine test data set of best results from Tab. III, i.e. STORN with $h_{\mathrm{dim}} = 40$, $z_{\mathrm{dim}} = 3$, $n_{\mathrm{layers}} = 3$.

- In identification text-books, the previous situation does not prevail.
- The ratio between the samples and the number of parameters is generally quite high.
- Mechatronic systems, electrical systems, . . . etc.
- Identification is done once for all
- Initial state importance is marginal

What is it about? (Away from identification textbooks)



Nonlinear Regression Modeling for Engineering Applications

Modeling, Model Validation, and Enabling Design of Experiments

R. Russell Rhinehart

ASME PRESS

WILEY

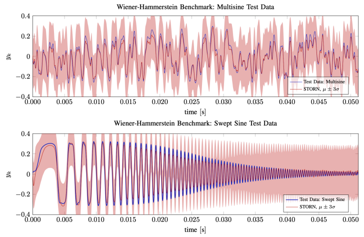
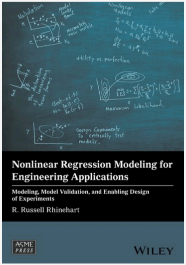


Fig. 12: Wiener Hammerstein benchmark: Time evaluation for multisine and swept sine test data set of best results from Tab. III, i.e. STORN with $h_{\text{dim}} = 40$, $z_{\text{dim}} = 3$, $n_{\text{layers}} = 3$.

- In identification text-books, the previous situation does not prevail.
- The ratio between the samples and the number of parameters is generally quite high.
- Mechatronic systems, electrical systems, . . . etc.
- Identification is done once for all
- Initial state importance is marginal

What is it about? (Away from identification textbooks)



Nonlinear Regression Modeling for Engineering Applications

Modeling, Model Validation, and Enabling Design of Experiments

R. Russell Rhinehart

ASME PRESS

WILEY

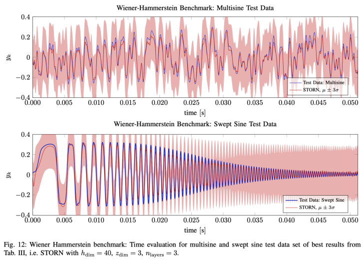
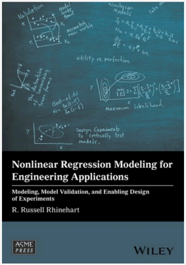


Fig. 12: Wiener Hammerstein benchmark: Time evaluation for multisine and swept sine test data set of best results from Tab. III, i.e. STORN with $h_{\mathrm{dim}} = 40$, $z_{\mathrm{dim}} = 3$, $n_{\mathrm{layers}} = 3$.

- In identification text-books, the previous situation does not prevail.
- The ratio between the samples and the number of parameters is generally quite high.
- Mechatronic systems, electrical systems, . . . etc.
- Identification is done once for all
- Initial state importance is marginal

What is it about? (Away from identification textbooks)



Nonlinear Regression Modeling for Engineering Applications

Modeling, Model Validation, and Enabling Design of Experiments

R. Russell Rhinehart

ASME PRESS

WILEY

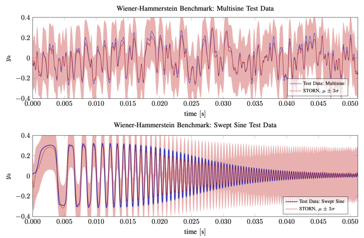
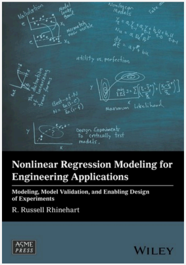


Fig. 12: Wiener Hammerstein benchmark: Time evaluation for multisine and swept sine test data set of best results from Tab. III, i.e. STORN with $h_{\text{dim}} = 40$, $z_{\text{dim}} = 3$, $n_{\text{layers}} = 3$.

- In identification text-books, the previous situation does not prevail.
- The ratio between the samples and the number of parameters is generally quite high.
- Mechatronic systems, electrical systems, . . . etc.
- Identification is done once for all
- Initial state importance is marginal

What is it about? (Away from identification textbooks)


Nonlinear Regression Modeling for Engineering Applications

Modeling, Model Validation, and Enabling Design of Experiments

R. Russell Rhinehart

ASME PRESS

WILEY

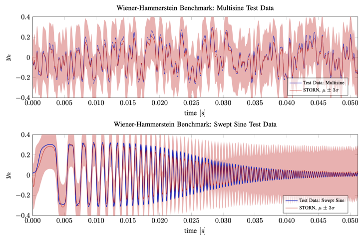
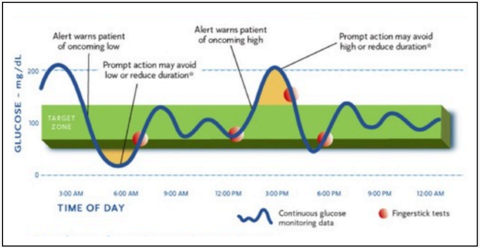


Fig. 12: Wiener Hammerstein benchmark: Time evaluation for multisine and swept sine test data set of best results from Tab. III, i.e. STORN with $h_{\text{dim}} = 40$, $z_{\text{dim}} = 3$, $n_{\text{layers}} = 3$.

- In identification text-books, the previous situation does not prevail.
- The ratio between the samples and the number of parameters is generally quite high.
- Mechatronic systems, electrical systems, . . . etc.
- Identification is done once for all
- Initial state importance is marginal

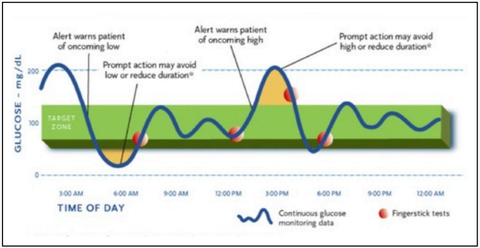
What is it about? (The diabetes example)

https://quizlet.com



- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)



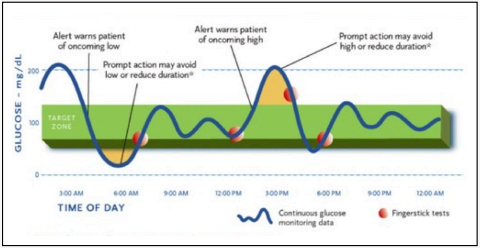
https://quizlet.com

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about?

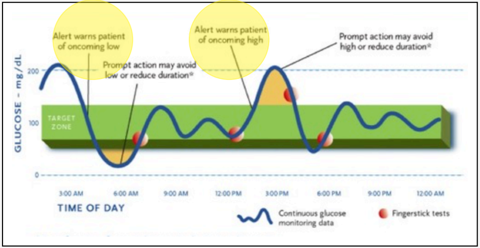
(The diabetes example)

https://quizlet.com



- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

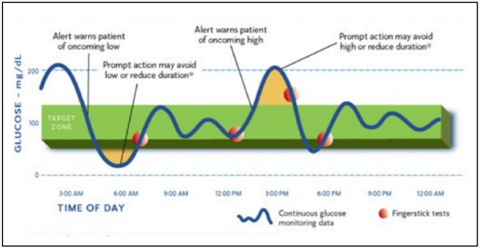
What is it about? (The diabetes example)



https://quizlet.com

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

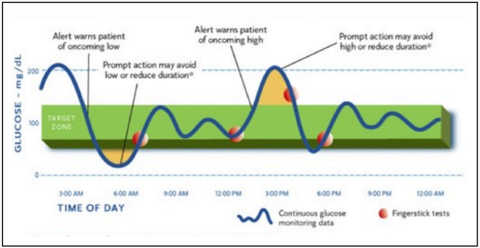
What is it about? (The diabetes example)



https://quizlet.com

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

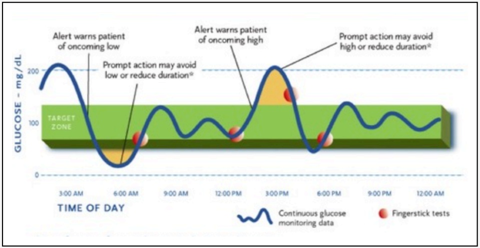
What is it about? (The diabetes example)



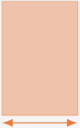
https://quizlet.com

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)



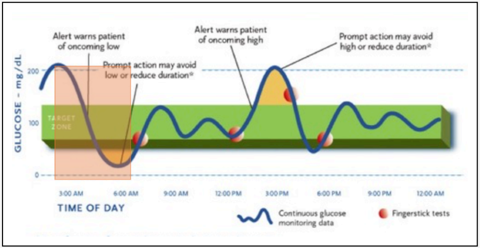
https://quizlet.com



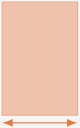
observation/identification window

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)



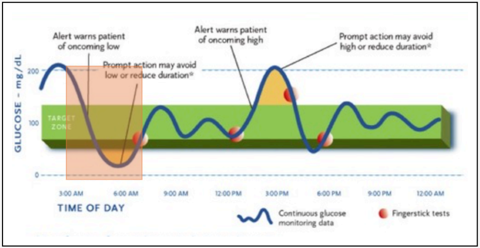
https://quizlet.com



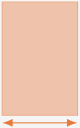
observation/identification window

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)



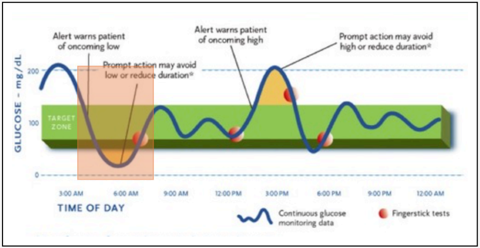
https://quizlet.com



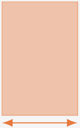
observation/identification window

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)



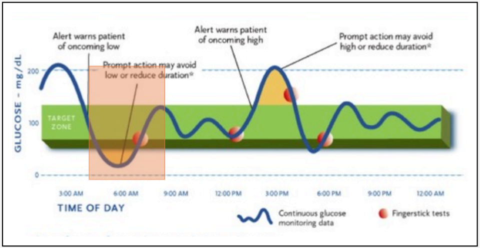
https://quizlet.com



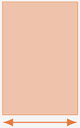
observation/identification window

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)



https://quizlet.com



observation/identification window

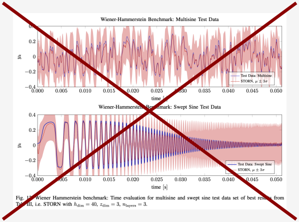
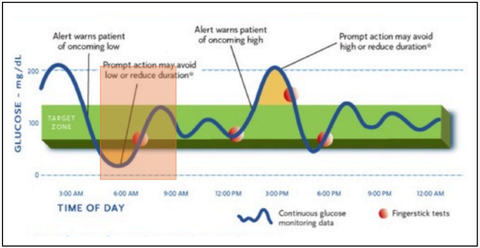


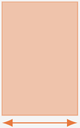
Fig. 12: Wiener Hammerstein benchmark: Time evaluation for multisine and swept sine test data set of best results from Tab. III, i.e. STORN with $h_{\text{dim}} = 40$, $z_{\text{dim}} = 3$, $n_{\text{layers}} = 3$.

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays ($\Rightarrow$ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)



https://quizlet.com



observation/identification window

$$\dot{G}(t) = -(S_g + X(t)) \cdot G(t) + S_g \cdot G_b + \frac{R_a(t)}{V_g} + \mathbf{w(t)}$$

$$\dot{X}(t) = -p_2 \cdot X(t) + p_2 \cdot S_I[I(t) - I_b]$$

$$\dot{Q}_1(t) = -k_\tau \cdot Q_1(t) + m(t)$$

$$\dot{Q}_2(t) = -k_{abs} \cdot Q_2(t) + k_\tau \cdot Q_1(t)$$

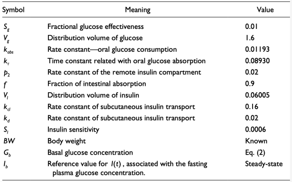
$$\dot{I}_{sc1}(t) = -k_d \cdot I_{sc1}(t) + J_{ctrl}(t)$$

$$\dot{I}_{sc2}(t) = -k_d \cdot I_{sc2}(t) + k_d \cdot I_{sc1}(t)$$

$$\dot{I}_p(t) = -k_{cl} \cdot I_p(t) + k_d \cdot I_{sc2}(t)$$

$$R_a(t) = \frac{k_{abs} \cdot f}{BW} \cdot Q_2(t)$$

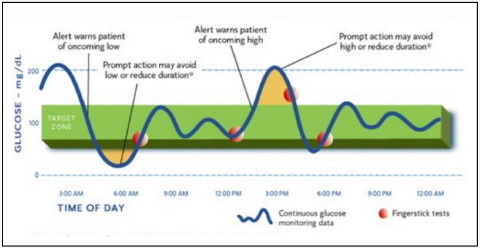
$$I(t) = \frac{I_p}{V_I \cdot BW}$$



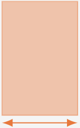
Symbol	Meaning	Value
S_g	Fractional glucose effectiveness	0.01
V_g	Distribution volume of glucose	1.6
k_{abs}	Rate constant—oral glucose consumption	0.01193
k_τ	Time constant related with oral glucose absorption	0.08930
p_2	Rate constant of the remote insulin compartment	0.02
f	Fraction of intestinal absorption	0.9
V_I	Distribution volume of insulin	0.06005
k_{cl}	Rate constant of subcutaneous insulin transport	0.16
k_d	Rate constant of subcutaneous insulin transport	0.02
S_I	Insulin sensitivity	0.0006
BW	Body weight	Known
G_b	Basal glucose concentration	Eq. (2)
I_b	Reference value for $I(t)$, associated with the fasting plasma glucose concentration.	Steady-state

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays ($\Rightarrow$ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

What is it about? (The diabetes example)

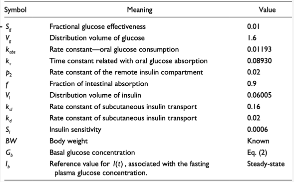


https://quizlet.com



observation/identification window

$$\dot{G}(t) = -(S_g + X(t)) \cdot G(t) + S_g \cdot G_b + \frac{R_a(t)}{V_g} + \mathbf{w(t)}$$
$$\dot{X}(t) = -p_2 \cdot X(t) + p_2 \cdot S_I[I(t) - I_b]$$
$$\dot{Q}_1(t) = -k_\tau \cdot Q_1(t) + m(t)$$
$$\dot{Q}_2(t) = -k_{abs} \cdot Q_2(t) + k_\tau \cdot Q_1(t)$$
$$\dot{I}_{sc1}(t) = -k_d \cdot I_{sc1}(t) + J_{ctrl}(t)$$
$$\dot{I}_{sc2}(t) = -k_d \cdot I_{sc2}(t) + k_d \cdot I_{sc1}(t)$$
$$\dot{I}_p(t) = -k_{cl} \cdot I_p(t) + k_d \cdot I_{sc2}(t)$$
$$R_a(t) = \frac{k_{abs} \cdot f}{BW} \cdot Q_2(t)$$
$$I(t) = \frac{I_p}{V_I \cdot BW}$$



Symbol	Meaning	Value
S_g	Fractional glucose effectiveness	0.01
V_g	Distribution volume of glucose	1.6
k_{abs}	Rate constant—oral glucose consumption	0.01193
k_τ	Time constant related with oral glucose absorption	0.08930
p_2	Rate constant of the remote insulin compartment	0.02
f	Fraction of intestinal absorption	0.9
V_I	Distribution volume of insulin	0.06005
k_{cl}	Rate constant of subcutaneous insulin transport	0.16
k_d	Rate constant of subcutaneous insulin transport	0.02
S_I	Insulin sensitivity	0.0006
BW	Body weight	Known
G_b	Basal glucose concentration	Eq. (2)
I_b	Reference value for $I(t)$, associated with the fasting plasma glucose concentration.	Steady-state

- Diabetes → bad control of Glucose
- Need for control that keeps Glucose in a safe region
- Control is based on Insuline injection
- Dynamics shows large delays (⇒ Needs anticipation)
- The parameters of the dynamics varie:
  - ⋆ Between patients
  - ⋆ Over time for the same patient

Few measurement
Many parameters

What is it about? (Why on earth should we estimate state/parameters?)

- Purely data driven solutions can capture only relationships between **measured quantities**.

What is it about? (Why on earth should we estimate state/parameters?)

- Purely data driven solutions can capture only relationships between **measured quantities**.
- In many problems, either
    - The control objective or
    - The constraints

  involve the internal **non measured state** in their expressions.

What is it about? (Why on earth should we estimate state/parameters?)

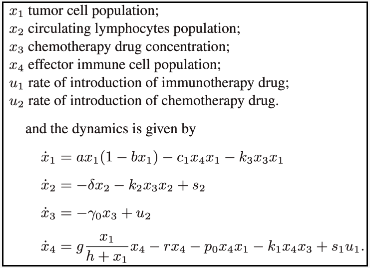
- Purely data driven solutions can capture only relationships between **measured quantities**.
- In many problems, either
    - The control objective or
    - The constraints

  involve the internal **non measured state** in their expressions.
- In these cases, monitoring and/or controlling the system needs the state/parameters to be dynamically reconstructed!

What is it about? (Why on earth should we estimate state/parameters?)

- Purely data driven solutions can capture only relationships between **measured quantities**.
- In many problems, either
    - The control objective or
    - The constraints

  involve the internal **non measured state** in their expressions.
- In these cases, monitoring and/or controlling the system needs the state/parameters to be dynamically reconstructed!



x_1 tumor cell population;
x_2 circulating lymphocytes population;
x_3 chemotherapy drug concentration;
x_4 effector immune cell population;
u_1 rate of introduction of immunotherapy drug;
u_2 rate of introduction of chemotherapy drug.

and the dynamics is given by

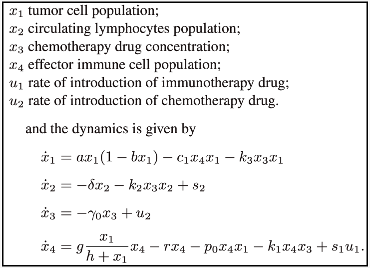
$$\dot{x}_1 = ax_1(1-bx_1) - c_1x_4x_1 - k_3x_3x_1$$
$$\dot{x}_2 = -\delta x_2 - k_2x_3x_2 + s_2$$
$$\dot{x}_3 = -\gamma_0 x_3 + u_2$$
$$\dot{x}_4 = g\frac{x_1}{h+x_1}x_4 - rx_4 - p_0x_4x_1 - k_1x_4x_3 + s_1u_1.$$

A constraint such as $x_4 \geq$ Threshold cannot be addressed via data-driven solutions that do not involve measurement of x_4 **unless** a model and associated state reconstruction algorithm are used.

What is it about? (Why on earth should we estimate state/parameters?)

- Purely data driven solutions can capture only relationships between **measured quantities**.
- In many problems, either
  - The control objective or
  - The constraints

  involve the internal **non measured state** in their expressions.
- In these cases, monitoring and/or controlling the system needs the state/parameters to be dynamically reconstructed!
- Why to rely on ML to approximately rediscover **Newton-law**?



x_1 tumor cell population;
x_2 circulating lymphocytes population;
x_3 chemotherapy drug concentration;
x_4 effector immune cell population;
u_1 rate of introduction of immunotherapy drug;
u_2 rate of introduction of chemotherapy drug.

and the dynamics is given by

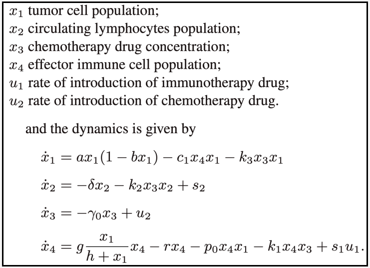
$$\begin{aligned}\dot{x}_1 &= ax_1(1-bx_1) - c_1x_4x_1 - k_3x_3x_1\\ \dot{x}_2 &= -\delta x_2 - k_2x_3x_2 + s_2\\ \dot{x}_3 &= -\gamma_0 x_3 + u_2\\ \dot{x}_4 &= g\frac{x_1}{h+x_1}x_4 - rx_4 - p_0x_4x_1 - k_1x_4x_3 + s_1u_1.\end{aligned}$$

A constraint such as $x_4 \geq$ Threshold cannot be addressed via data-driven solutions that do not involve measurement of x_4 **unless** a model and associated state reconstruction algorithm are used.

What is it about? (Why on earth should we estimate state/parameters?)

- Purely data driven solutions can capture only relationships between **measured quantities**.
- In many problems, either
    - The control objective or
    - The constraints

  involve the internal **non measured state** in their expressions.
- In these cases, monitoring and/or controlling the system needs the state/parameters to be dynamically reconstructed!
- Why to rely on ML to approximately rediscover **Newton-law**?



x_1 tumor cell population;
x_2 circulating lymphocytes population;
x_3 chemotherapy drug concentration;
x_4 effector immune cell population;
u_1 rate of introduction of immunotherapy drug;
u_2 rate of introduction of chemotherapy drug.

and the dynamics is given by

$$\begin{aligned}
\dot{x}_1 &= ax_1(1-bx_1) - c_1x_4x_1 - k_3x_3x_1 \\
\dot{x}_2 &= -\delta x_2 - k_2x_3x_2 + s_2 \\
\dot{x}_3 &= -\gamma_0 x_3 + u_2 \\
\dot{x}_4 &= g\frac{x_1}{h+x_1}x_4 - rx_4 - p_0x_4x_1 - k_1x_4x_3 + s_1u_1.
\end{aligned}$$

A constraint such as $x_4 \geq$ Threshold cannot be addressed via data-driven solutions that do not involve measurement of x_4 **unless** a model and associated state reconstruction algorithm are used.

Question: Am I in the right room?

What is it about? (Why on earth should we estimate state/parameters?)

- Purely data driven solutions can capture only relationships between **measured quantities**.
- In many problems, either
  - The control objective or
  - The constraints

  involve the internal **non measured state** in their expressions.
- In these cases, monitoring and/or controlling the system needs the state/parameters to be dynamically reconstructed!
- Why to rely on ML to approximately rediscover **Newton-law**?

x_1 tumor cell population;
x_2 circulating lymphocytes population;
x_3 chemotherapy drug concentration;
x_4 effector immune cell population;
u_1 rate of introduction of immunotherapy drug;
u_2 rate of introduction of chemotherapy drug.

and the dynamics is given by

$$\begin{aligned}
\dot{x}_1 &= ax_1(1-bx_1) - c_1x_4x_1 - k_3x_3x_1\\
\dot{x}_2 &= -\delta x_2 - k_2x_3x_2 + s_2\\
\dot{x}_3 &= -\gamma_0 x_3 + u_2\\
\dot{x}_4 &= g\frac{x_1}{h+x_1}x_4 - rx_4 - p_0x_4x_1 - k_1x_4x_3 + s_1u_1.
\end{aligned}$$

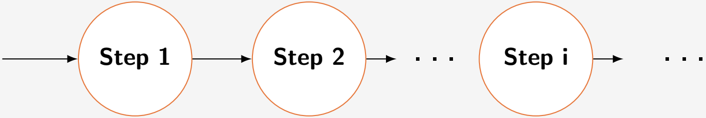
A constraint such as $x_4 \geq$ Threshold cannot be addressed via data-driven solutions that do not involve measurement of x_4 **unless** a model and associated state reconstruction algorithm are used.

Question: Am I in the right room?

Answer: Definitively!

What is precisely a data-driven solution?

A data-driven solution is a multi-step process such that:


Step 1

Step 2

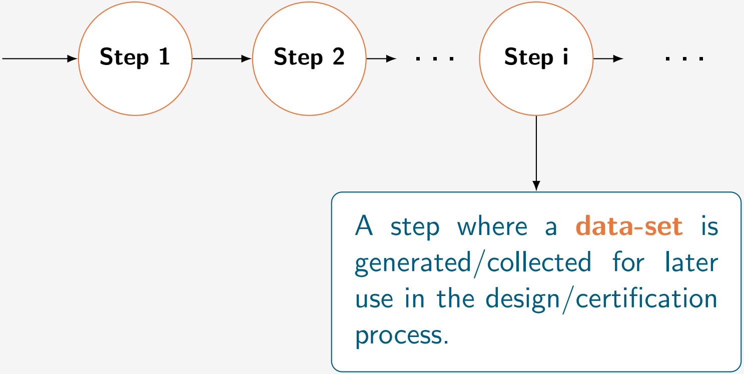
. . .

Step i

. . .

What is precisely a data-driven solution?

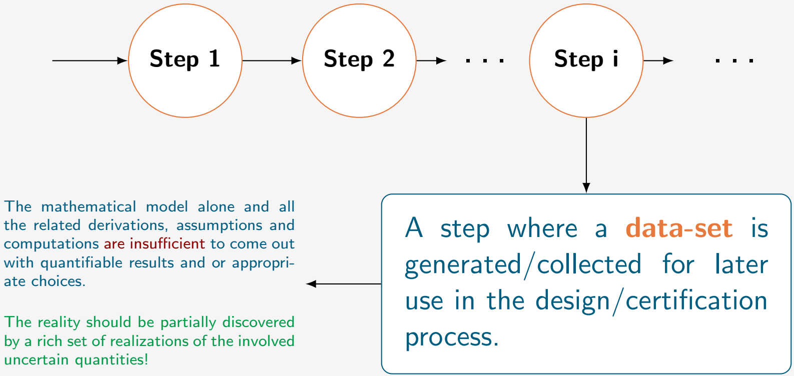
A data-driven solution is a multi-step process such that:



A step where a **data-set** is generated/collected for later use in the design/certification process.

What is precisely a data-driven solution?

A data-driven solution is a multi-step process such that:

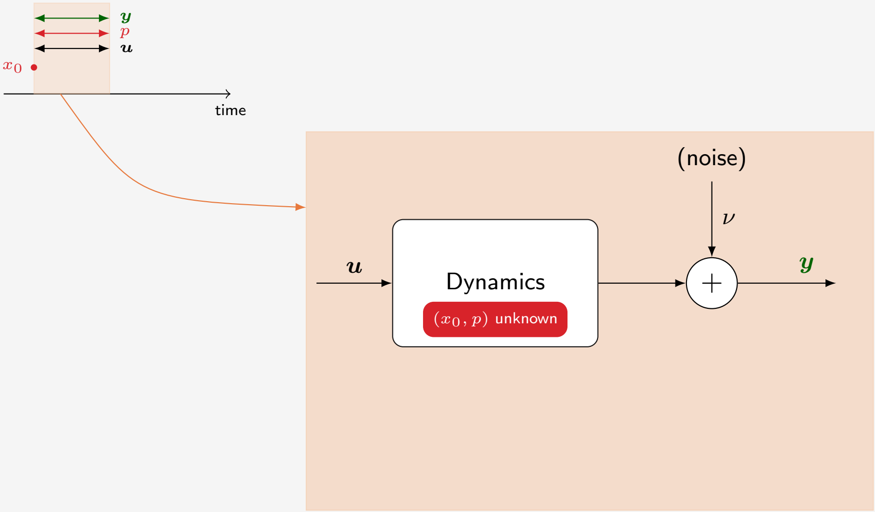


A step where a **data-set** is generated/collected for later use in the design/certification process.

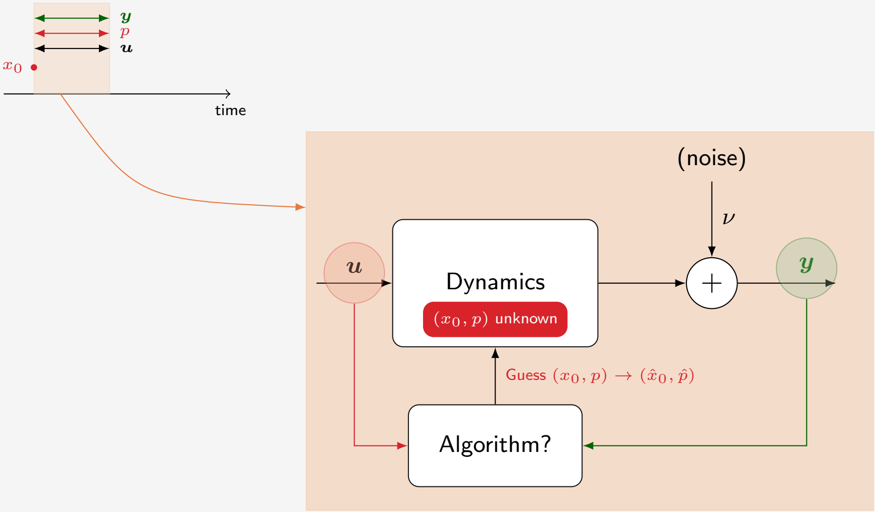
The mathematical model alone and all the related derivations, assumptions and computations are insufficient to come out with quantifiable results and or appropriate choices.

The reality should be partially discovered by a rich set of realizations of the involved uncertain quantities!

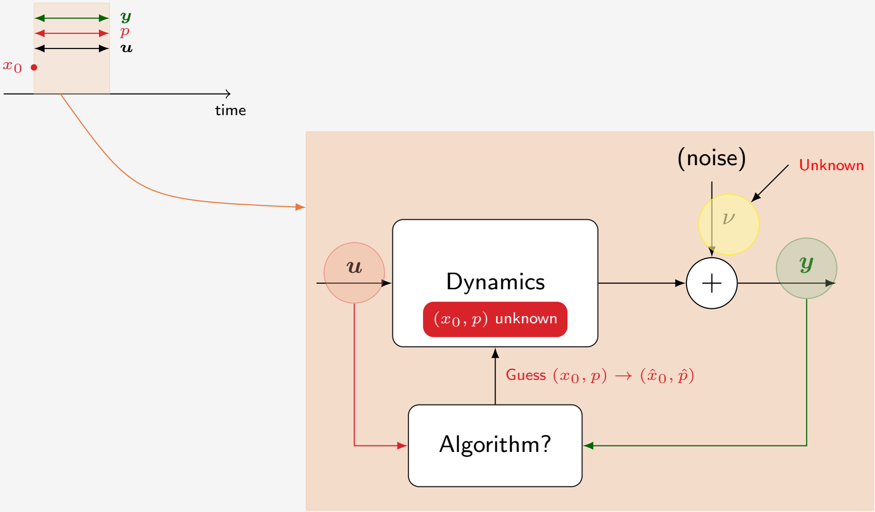
The dynamic extended estimation problem



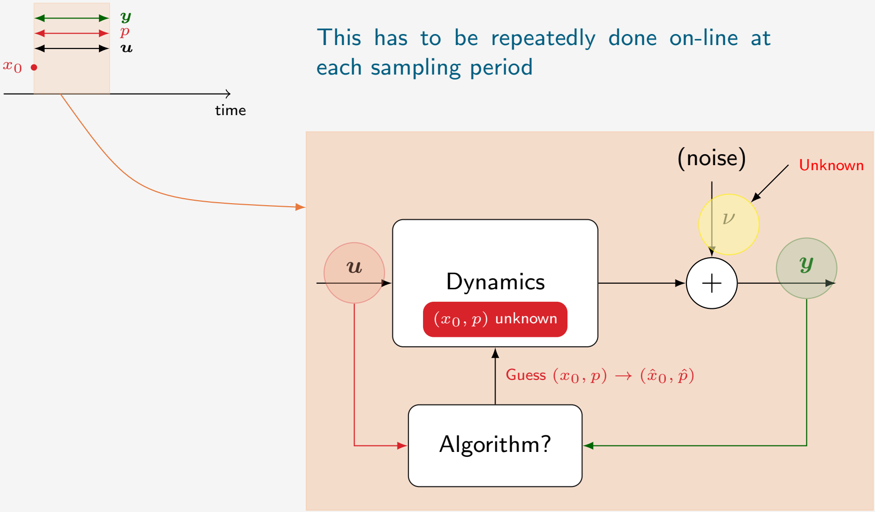
The dynamic extended estimation problem



The dynamic extended estimation problem

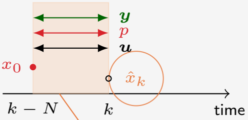


The dynamic extended estimation problem



This has to be repeatedly done on-line at each sampling period

The dynamic extended estimation problem



This has to be repeatedly done on-line at each sampling period

Given the estimation at instant k

$(\hat{x}_0, \hat{p})$

The current state is estimated by

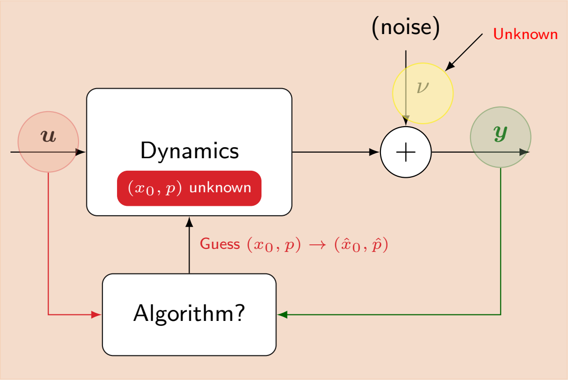
$$\hat{x}_k = X_N(\hat{x}_0, \hat{p}, \boldsymbol{u})$$

More generally, the value of any quantity:

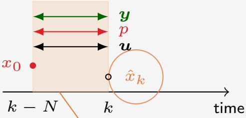
$$z = T(x, p)$$

Can be estimated by:

$$\boxed{\hat{z}_k = T\Big(X_N(\hat{x}_0, \hat{p}, \boldsymbol{u}), \hat{p}\Big)}$$



The dynamic extended estimation problem



Given the estimation at instant k

$(\hat{x}_0, \hat{p})$

The current state is estimated by

$$\hat{x}_k = X_N(\hat{x}_0, \hat{p}, \boldsymbol{u})$$

More generally, the value of any quantity:

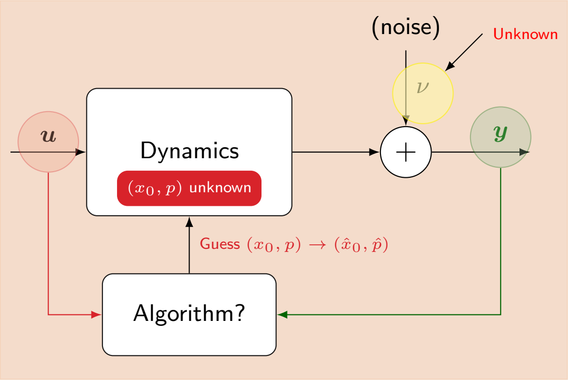
$$z = T(x, p)$$

Can be estimated by:

$$\hat{z}_k = T\Big(X_N(\hat{x}_0, \hat{p}, \boldsymbol{u}), \hat{p}\Big)$$

$z = T(x, p)$ is called hereafter, the estimation target.

Example. A feedback law $K(x, p)$ can be viewed as an estimation target in an observer-based control design.



Problem statement

Given

1. An uncertain model $\dot{x} = f(x, u, p)$
2. Statistics of uncertainties $\mathcal{P}$ on $\mathbb{P}$
3. Statistics of states $\mathcal{X}$ on $\mathbb{X}$
4. Statistics of measurement noise $\mathcal{V}$ on $\mathbb{V}$
5. A given observation target $z = T(x, p)$

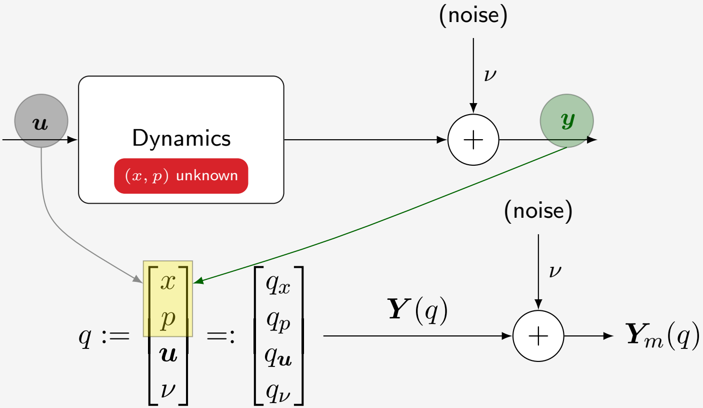
Problem statement

Given

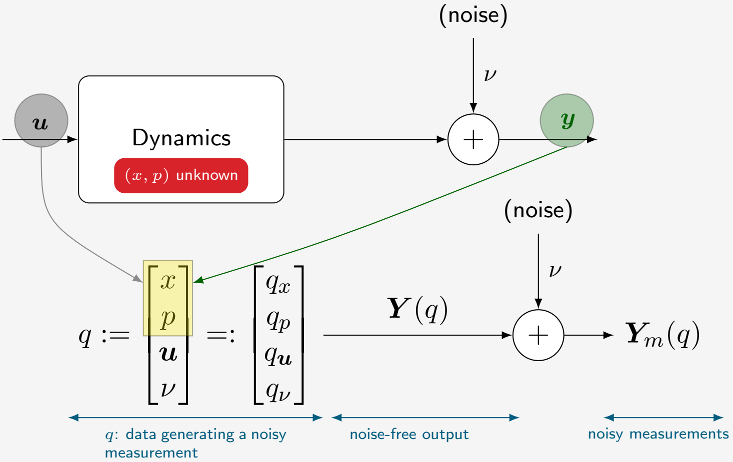
1. An uncertain model $\dot{x} = f(x, u, p)$
2. Statistics of uncertainties $\mathcal{P}$ on $\mathbb{P}$
3. Statistics of states $\mathcal{X}$ on $\mathbb{X}$
4. Statistics of measurement noise $\mathcal{V}$ on $\mathbb{V}$
5. A given observation target $z = T(x, p)$

> **Derive** a certified bound on the excursion of the estimation error on the observation target z together with guidelines towards its minimization.

Notation: The scenarios vector q



Notation: The scenarios vector q



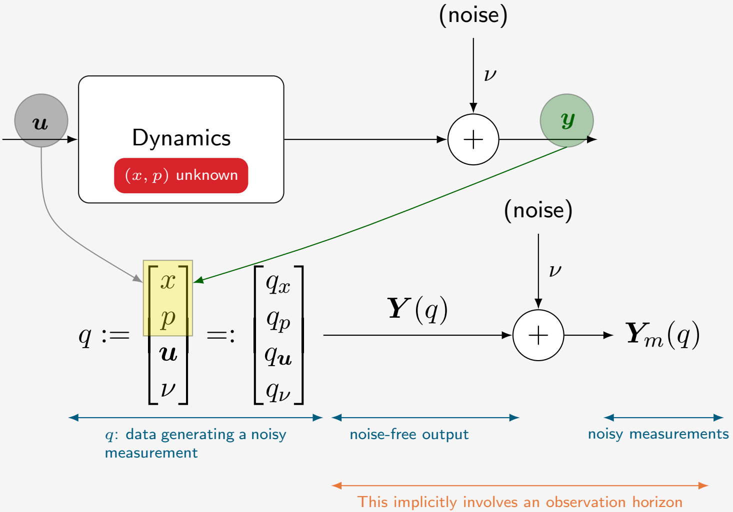
$$q := \begin{bmatrix} x \\ p \\ \boldsymbol{u} \\ \nu \end{bmatrix} =: \begin{bmatrix} q_x \\ q_p \\ q_{\boldsymbol{u}} \\ q_\nu \end{bmatrix}$$

q: data generating a noisy measurement

noise-free output

noisy measurements

Notation: The scenarios vector q



$$q := \begin{bmatrix} x \\ p \\ \boldsymbol{u} \\ \nu \end{bmatrix} =: \begin{bmatrix} q_x \\ q_p \\ q_{\boldsymbol{u}} \\ q_\nu \end{bmatrix} \xrightarrow{\boldsymbol{Y}(q)} \oplus \rightarrow \boldsymbol{Y}_m(q)$$

q: data generating a noisy measurement

noise-free output

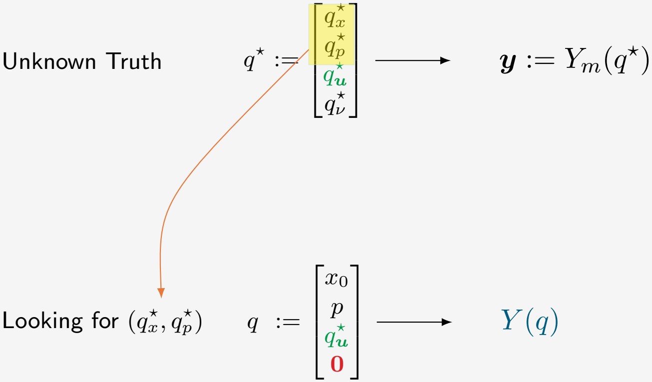
noisy measurements

This implicitly involves an observation horizon

The cost function (Distance between scenarios)

Unknown Truth $$q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \quad \boldsymbol{y} := Y_m(q^\star)$$

The cost function (Distance between scenarios)



Unknown Truth $$q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$$

Looking for $(q_x^\star, q_p^\star)$ $$q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$$

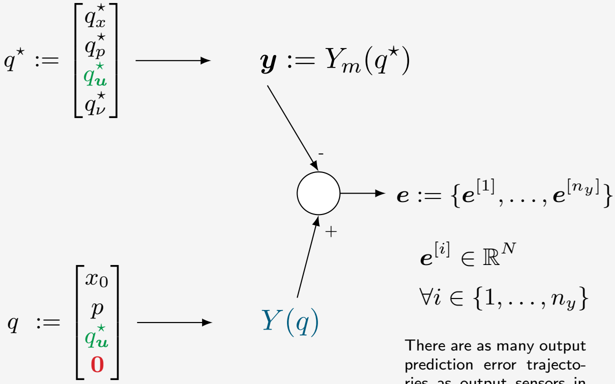
The cost function (Distance between scenarios)

Unknown Truth $$q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$$

Looking for $(q_x^\star, q_p^\star)$ $$q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$$

One might try to find the noise profile.
$\rightarrow$ $(n_x + n_p + N \cdot n_y)$ parameters
vs $(N \cdot n_y)$ information samples.

The cost function (Distance between scenarios)



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$

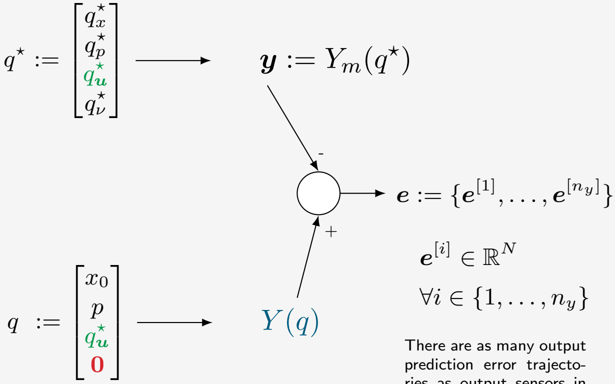
$\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$

$\boldsymbol{e}^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1, \dots, n_y\}$

There are as many output prediction error trajectories as output sensors in the system.

The cost function (Distance between scenarios)



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

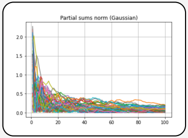
Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$

$\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$

$\boldsymbol{e}^{[i]} \in \mathbb{R}^N$

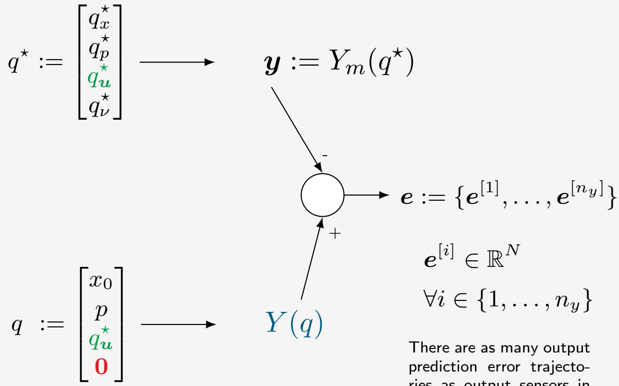
$\forall i \in \{1, \dots, n_y\}$

There are as many output prediction error trajectories as output sensors in the system.



abs(Partial average) of a unitary gaussian noise

The cost function (Distance between scenarios)



Unknown Truth

$$q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$$

$$\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$$

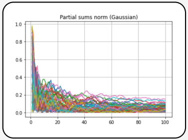
$$\boldsymbol{e}^{[i]} \in \mathbb{R}^N$$

$$\forall i \in \{1, \dots, n_y\}$$

Looking for $(q_x^\star, q_p^\star)$

$$q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$$

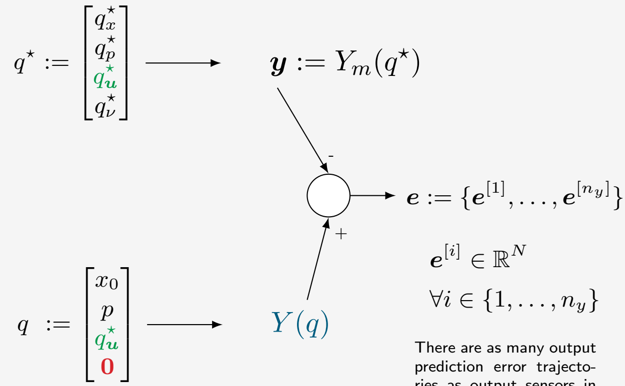
There are as many output prediction error trajectories as output sensors in the system.



abs(Partial average) of a unitary uniform noise

The cost function (Distance between scenarios)

Unknown Truth



$$q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$$

$$\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$$

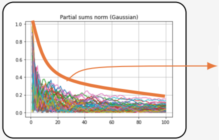
$$\boldsymbol{e}^{[i]} \in \mathbb{R}^N$$

$$\forall i \in \{1, \dots, n_y\}$$

Looking for $(q_x^\star, q_p^\star)$

$$q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$$

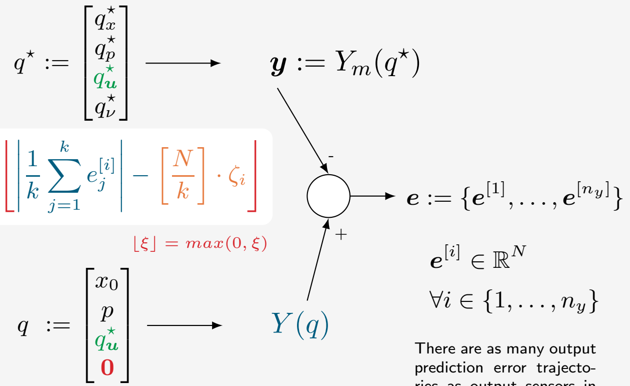
There are as many output prediction error trajectories as output sensors in the system.



$$\left[\frac{N}{k}\right] \cdot \zeta$$

abs(Partial average) of a unitary uniform noise

The cost function (Distance between scenarios)



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

$$J(q\;|\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y}\left\lfloor \left|\frac{1}{k}\sum_{j=1}^{k} e_j^{[i]}\right| - \left[\frac{N}{k}\right]\cdot\zeta_i \right\rfloor$$

$\lfloor\xi\rfloor = max(0,\xi)$

-

+

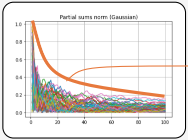
$\longrightarrow \boldsymbol{e} := \{\boldsymbol{e}^{[1]},\dots,\boldsymbol{e}^{[n_y]}\}$

$\boldsymbol{e}^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1,\dots,n_y\}$

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \boldsymbol{0} \end{bmatrix} \longrightarrow Y(q)$

There are as many output prediction error trajectories as output sensors in the system.



$\left[\frac{N}{k}\right]\cdot\zeta$

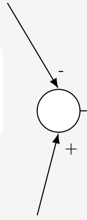
abs(Partial average) of a unitary uniform noise

The cost function (Distance between scenarios)

Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

$$J(q \mid \boldsymbol{y}, \zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left| \frac{1}{k} \sum_{j=1}^{k} e_j^{[i]} \right| - \left[\frac{N}{k}\right] \cdot \zeta_i \right\rfloor$$

$\lfloor \xi \rfloor = max(0, \xi)$



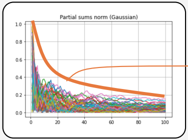
$\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$

$\boldsymbol{e}^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1, \dots, n_y\}$

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \boldsymbol{0} \end{bmatrix} \longrightarrow Y(q)$

There are as many output prediction error trajectories as output sensors in the system.

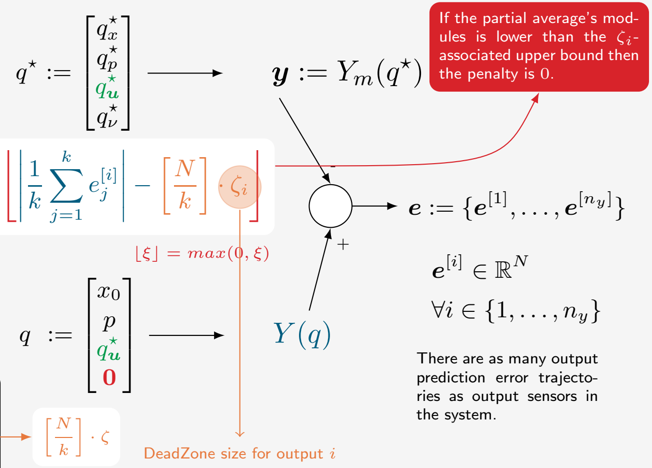


abs(Partial average) of a unitary uniform noise

$\left[\frac{N}{k}\right] \cdot \zeta$

DeadZone size for output i

The cost function (Distance between scenarios)



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

If the partial average's modules is lower than the ζ_i-associated upper bound then the penalty is 0.

$$J(q\ |\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left| \frac{1}{k}\sum_{j=1}^{k} e_j^{[i]} \right| - \left[\frac{N}{k}\right] \cdot \zeta_i \right\rfloor$$

$\lfloor \xi \rfloor = max(0, \xi)$

$-$

$+$

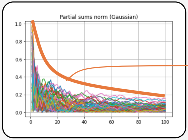
$\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$

$\boldsymbol{e}^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1, \dots, n_y\}$

There are as many output prediction error trajectories as output sensors in the system.

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \boldsymbol{0} \end{bmatrix} \longrightarrow Y(q)$

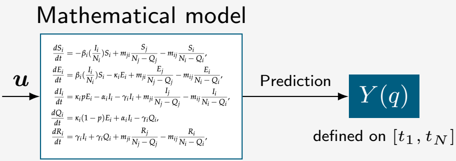


$\left[\frac{N}{k}\right] \cdot \zeta$

DeadZone size for output i

abs(Partial average) of a unitary uniform noise

Estimation as optimization problem (MHE: Moving-Horizon Estimator)



Mathematical model

$\boldsymbol{u}$

$$
\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1 - p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}
$$

Prediction

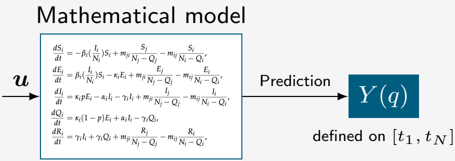
$Y(q)$

defined on $[t_1\,,t_N]$

The prediction of the expected measurement should the initial state be x_0 and the true parameter be p under the control profile $\boldsymbol{u}$.

Estimation as optimization problem (MHE: Moving-Horizon Estimator)

The prediction of the expected measurement should the initial state be x_0 and the true parameter be p under the control profile $\boldsymbol{u}$.



Mathematical model

$\boldsymbol{u}$

$$
\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}
$$

Prediction

$Y(q)$

defined on $[t_1, t_N]$

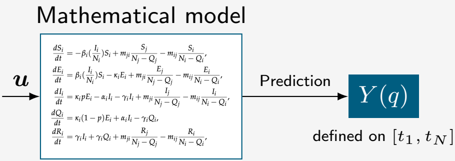
The prediction error associated to q

$$\mathcal{E}(q \mid \boldsymbol{y}) := d\Big(Y(q), \boldsymbol{y}\Big)$$

$$\begin{bmatrix} y(t_1) \\ y(t_2) \\ \vdots \\ y(t_N) \end{bmatrix}$$

Estimation as optimization problem (MHE: Moving-Horizon Estimator)

The prediction of the expected measurement should the initial state be x_0 and the true parameter be p under the control profile $\boldsymbol{u}$.

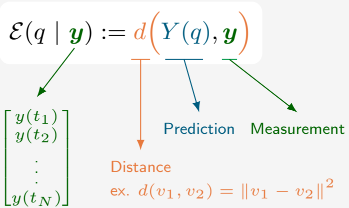


Mathematical model

$$
\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}
$$

defined on $[t_1, t_N]$

The prediction error associated to q



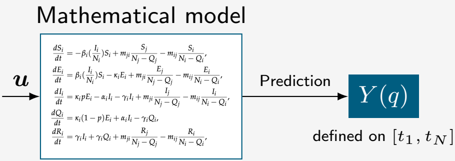
$$\mathcal{E}(q \mid \boldsymbol{y}) := \underbrace{d}_{\text{Distance}}\Big(\underbrace{Y(q)}_{\text{Prediction}}, \underbrace{\boldsymbol{y}}_{\text{Measurement}}\Big)$$

$$\begin{bmatrix} y(t_1) \\ y(t_2) \\ \vdots \\ y(t_N) \end{bmatrix}$$

Distance

ex. $d(v_1, v_2) = \|v_1 - v_2\|^2$

Estimation as optimization problem (MHE: Moving-Horizon Estimator)



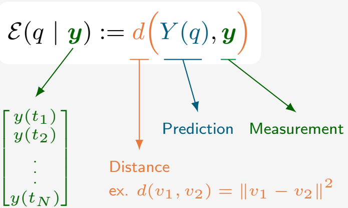
Mathematical model

$$\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i}, \\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i}, \\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i}, \\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i, \\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}$$

defined on $[t_1, t_N]$

The prediction of the expected measurement should the initial state be x_0 and the true parameter be p under the control profile $\boldsymbol{u}$.

The prediction error associated to q



$$\mathcal{E}(q \mid \boldsymbol{y}) := d\Big(Y(q), \boldsymbol{y}\Big)$$

$$\begin{bmatrix} y(t_1) \\ y(t_2) \\ \vdots \\ y(t_N) \end{bmatrix}$$

Prediction

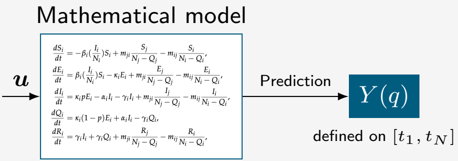
Measurement

Distance

ex. $d(v_1, v_2) = \|v_1 - v_2\|^2$

$$\boxed{(\hat{x}_0, \hat{p}) \leftarrow \arg\min_{(x_0,p)} \Big[\mathcal{E}(q \mid \boldsymbol{y})\Big]}$$

Estimation as optimization problem (MHE: Moving-Horizon Estimator)



Mathematical model

$$
\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}
$$

defined on $[t_1, t_N]$

The prediction of the expected measurement should the initial state be x_0 and the true parameter be p under the control profile $\boldsymbol{u}$.

Michalaksa and Mayne. Moving Horizon Observers. IEEE-TAC 1995.

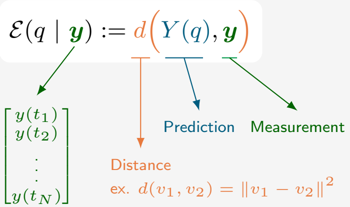
M.A. Nonlinear Moving Horizon Observers. Springer 2007.

Kuhl et al. A real-time algorithm for moving horizon state and parameter estimation. Computers & Chemical Eng. 2011.

Rawlings and Li. Optimization-Based state estimation: Current status and some new results. J. Process Control, 2012

M. A. Muller. Nonlinear Moving Horizon Estimation in Presence of Bounded Disturbances. Automatica, 2017.

The prediction error associated to q



$$\mathcal{E}(q \mid \boldsymbol{y}) := d\Big(Y(q), \boldsymbol{y}\Big)$$

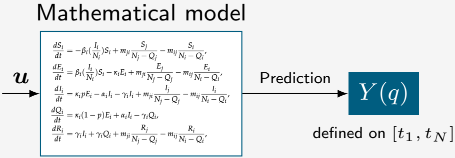
$$\begin{bmatrix} y(t_1) \\ y(t_2) \\ \vdots \\ y(t_N) \end{bmatrix}$$

Distance

ex. $d(v_1, v_2) = \|v_1 - v_2\|^2$

$$\boxed{(\hat{x}_0, \hat{p}) \leftarrow \arg\min_{(x_0,p)} \Big[\mathcal{E}(q \mid \boldsymbol{y})\Big]}$$

Estimation as optimization problem (MHE: Moving-Horizon Estimator)



Mathematical model

$$
\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}
$$

defined on $[t_1, t_N]$

The prediction of the expected measurement should the initial state be x_0 and the true parameter be p under the control profile $\boldsymbol{u}$.

$$J(q\,|\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y}\left[\left|\frac{1}{k}\sum_{j=1}^{k} e_j^{[i]}\right| - \left[\frac{N}{k}\right]\cdot\zeta_i\right]$$

Michalaksa and Mayne. Moving Horizon Observers. IEEE-TAC 1995.

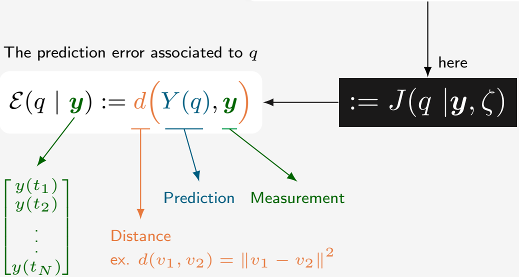
M.A. Nonlinear Moving Horizon Observers. Springer 2007.

Kuhl et al. A real-time algorithm for moving horizon state and parameter estimation. Computers & Chemical Eng. 2011.

Rawlings and Li. Optimization-Based state estimation: Current status and some new results. J. Process Control, 2012

M. A. Muller. Nonlinear Moving Horizon Estimation in Presence of Bounded Disturbances. Automatica, 2017.

The prediction error associated to q



$$\mathcal{E}(q \mid \boldsymbol{y}) := d\Big(Y(q), \boldsymbol{y}\Big)$$

$$:= J(q\,|\boldsymbol{y},\zeta)$$

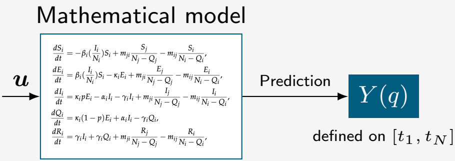
$$\begin{bmatrix} y(t_1)\\ y(t_2)\\ \vdots\\ y(t_N)\end{bmatrix}$$

Distance

ex. $d(v_1, v_2) = \|v_1 - v_2\|^2$

$$\boxed{(\hat{x}_0, \hat{p}) \leftarrow \arg\min_{(x_0,p)}\Big[\mathcal{E}(q \mid \boldsymbol{y})\Big]}$$

Estimation as optimization problem (MHE: Moving-Horizon Estimator)



Mathematical model

$\boldsymbol{u}$

$$\begin{aligned}
\frac{dS_i}{dt} &= -\beta_i(\frac{I_i}{N_i})S_i + m_{ji}\frac{S_j}{N_j - Q_j} - m_{ij}\frac{S_i}{N_i - Q_i},\\
\frac{dE_i}{dt} &= \beta_i(\frac{I_i}{N_i})S_i - \kappa_i E_i + m_{ji}\frac{E_j}{N_j - Q_j} - m_{ij}\frac{E_i}{N_i - Q_i},\\
\frac{dI_i}{dt} &= \kappa_i p E_i - \alpha_i I_i - \gamma_i I_i + m_{ji}\frac{I_j}{N_j - Q_j} - m_{ij}\frac{I_i}{N_i - Q_i},\\
\frac{dQ_i}{dt} &= \kappa_i(1-p)E_i + \alpha_i I_i - \gamma_i Q_i,\\
\frac{dR_i}{dt} &= \gamma_i I_i + \gamma_i Q_i + m_{ji}\frac{R_j}{N_j - Q_j} - m_{ij}\frac{R_i}{N_i - Q_i},
\end{aligned}$$

Prediction

$Y(q)$

defined on $[t_1, t_N]$

The prediction of the expected measurement should the initial state be x_0 and the true parameter be p under the control profile $\boldsymbol{u}$.

$$J(q \mid \boldsymbol{y}, \zeta)) := \sum_{i=1}^{n_y} \left[\left| \frac{1}{k} \sum_{j=1}^{k} e_j^{[i]} \right| - \left[\frac{N}{k} \right] \cdot \zeta_i \right]$$

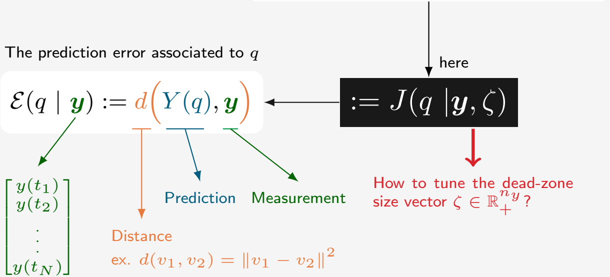
Michalaksa and Mayne. Moving Horizon Observers. IEEE-TAC 1995.

M.A. Nonlinear Moving Horizon Observers. Springer 2007.

Kuhl et al. A real-time algorithm for moving horizon state and parameter estimation. Computers & Chemical Eng. 2011.

Rawlings and Li. Optimization-Based state estimation: Current status and some new results. J. Process Control, 2012

M. A. Muller. Nonlinear Moving Horizon Estimation in Presence of Bounded Disturbances. Automatica, 2017.



here

The prediction error associated to q

$$\mathcal{E}(q \mid \boldsymbol{y}) := d\Big(Y(q), \boldsymbol{y}\Big)$$

$$:= J(q \mid \boldsymbol{y}, \zeta)$$

$$\begin{bmatrix} y(t_1) \\ y(t_2) \\ \vdots \\ y(t_N) \end{bmatrix}$$

Distance

ex. $d(v_1, v_2) = \|v_1 - v_2\|^2$

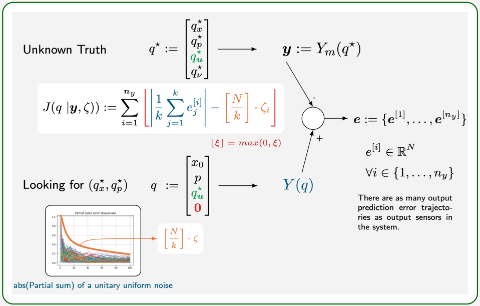
Prediction

Measurement

How to tune the dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$?

$$\boxed{(\hat{x}_0, \hat{p}) \leftarrow \arg\min_{(x_0, p)} \Big[\mathcal{E}(q \mid \boldsymbol{y})\Big]}$$

Dead-Zone consistency condition



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

$$J(q\ |\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left|\frac{1}{k}\sum_{j=1}^{k} e_j^{[i]}\right| - \left[\frac{N}{k}\right]\cdot\zeta_i \right\rfloor$$

$\lfloor\xi\rfloor = max(0,\xi)$

$\boldsymbol{e} := \{\boldsymbol{e}^{[1]},\dots,\boldsymbol{e}^{[n_y]}\}$

$e^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1,\dots,n_y\}$

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$

There are as many output prediction error trajectories as output sensors in the system.

$\left[\frac{N}{k}\right]\cdot\zeta$

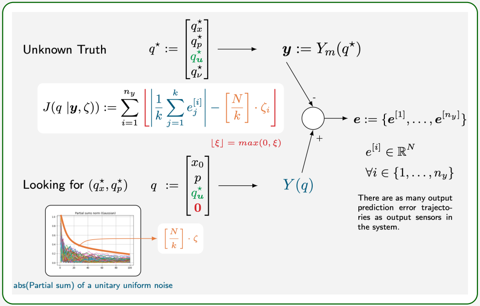
abs(Partial sum) of a unitary uniform noise

Consistency

A dead-zone size vector ζ is consistant (w.r.t the noise) if:

Any candidate scenario q that meets the true value of the triplet $(q_x^\star, q_p^\star, q_{\boldsymbol{u}}^\star)$ should lead to **zero cost** for any possible realization of the noise.

Dead-Zone consistency condition



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

$$J(q\,|\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y}\left\lfloor \left|\frac{1}{k}\sum_{j=1}^{k} e_j^{[i]}\right| - \left[\frac{N}{k}\right]\cdot\zeta_i \right\rfloor$$

$\lfloor\xi\rfloor = max(0,\xi)$

$\boldsymbol{e} := \{\boldsymbol{e}^{[1]},\dots,\boldsymbol{e}^{[n_y]}\}$

$e^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1,\dots,n_y\}$

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \boldsymbol{0} \end{bmatrix} \longrightarrow Y(q)$

There are as many output prediction error trajectories as output sensors in the system.

$\left[\frac{N}{k}\right]\cdot\zeta$

abs(Partial sum) of a unitary uniform noise

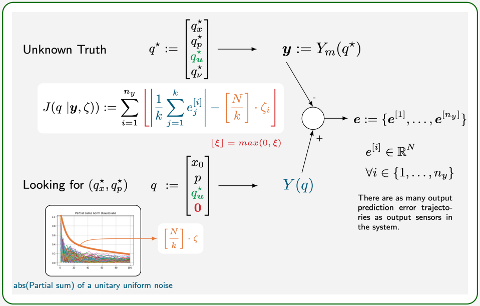
Consistency

A dead-zone size vector ζ is consistant (w.r.t the noise) if:

Any candidate scenario q that meets the true value of the triplet $(q_x^\star, q_p^\star, q_{\boldsymbol{u}}^\star)$ should lead to **zero cost** for any possible realization of the noise.

$$\boxed{J(q^\star \mid Y_m(q^\star), \zeta) = 0}$$

Dead-Zone consistency condition



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

$$J(q \mid \boldsymbol{y}, \zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left| \frac{1}{k} \sum_{j=1}^{k} e_j^{[i]} \right| - \left\lceil \frac{N}{k} \right\rceil \cdot \zeta_i \right\rfloor$$

$\lfloor \xi \rfloor = max(0, \xi)$

$-$ $+$ $\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$

$e^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1, \dots, n_y\}$

There are as many output prediction error trajectories as output sensors in the system.

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \boldsymbol{0} \end{bmatrix} \longrightarrow Y(q)$

$\left\lceil \frac{N}{k} \right\rceil \cdot \zeta$

abs(Partial sum) of a unitary uniform noise

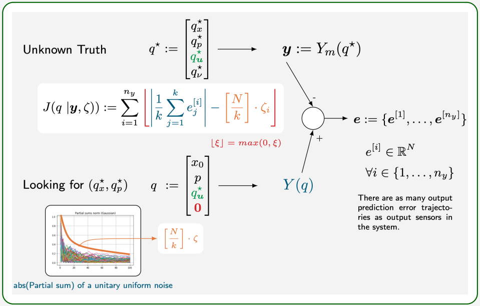
Consistency

A dead-zone size vector ζ is consistant (w.r.t the noise) if:

Any candidate scenario q that meets the true value of the triplet $(q_x^\star, q_p^\star, q_{\boldsymbol{u}}^\star)$ should lead to **zero cost** for any possible realization of the noise.

$$\boxed{J(q^\star \mid Y_m(q^\star), \zeta) = 0}$$

Dead-Zone consistency condition



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

$$J(q\ |\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left|\frac{1}{k}\sum_{j=1}^{k} e_j^{[i]}\right| - \left[\frac{N}{k}\right]\cdot\zeta_i \right\rfloor$$

$\lfloor \xi \rfloor = max(0,\xi)$

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$

$\boldsymbol{e} := \{\boldsymbol{e}^{[1]}, \dots, \boldsymbol{e}^{[n_y]}\}$

$e^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1, \dots, n_y\}$

There are as many output prediction error trajectories as output sensors in the system.

$\left[\frac{N}{k}\right]\cdot\zeta$

abs(Partial sum) of a unitary uniform noise

Consistency

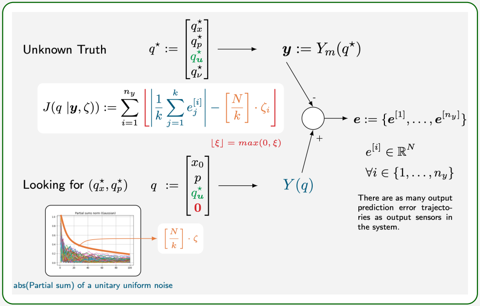
A dead-zone size vector ζ is consistant (w.r.t the noise) if:

Any candidate scenario q that meets the true value of the triplet $(q_x^\star, q_p^\star, q_{\boldsymbol{u}}^\star)$ should lead to **zero cost** for any possible realization of the noise.

$$\boxed{J(q^\star \mid Y_m(q^\star), \zeta) = 0}$$

Note that this can always be made true by taking sufficiently high values of ζ!

Dead-Zone consistency condition



Unknown Truth $q^\star := \begin{bmatrix} q_x^\star \\ q_p^\star \\ q_{\boldsymbol{u}}^\star \\ q_\nu^\star \end{bmatrix} \longrightarrow \boldsymbol{y} := Y_m(q^\star)$

$$J(q\ |\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left|\frac{1}{k}\sum_{j=1}^{k} e_j^{[i]}\right| - \left[\frac{N}{k}\right]\cdot\zeta_i \right\rfloor$$

$\lfloor\xi\rfloor = max(0,\xi)$

Looking for $(q_x^\star, q_p^\star)$ $q := \begin{bmatrix} x_0 \\ p \\ q_{\boldsymbol{u}}^\star \\ \mathbf{0} \end{bmatrix} \longrightarrow Y(q)$

-

+

$\boldsymbol{e} := \{\boldsymbol{e}^{[1]},\dots,\boldsymbol{e}^{[n_y]}\}$

$e^{[i]} \in \mathbb{R}^N$

$\forall i \in \{1,\dots,n_y\}$

There are as many output prediction error trajectories as output sensors in the system.

$\left[\frac{N}{k}\right]\cdot\zeta$

abs(Partial sum) of a unitary uniform noise

Consistency

A dead-zone size vector ζ is consistant (w.r.t the noise) if:

Any candidate scenario q that meets the true value of the triplet $(q_x^\star, q_p^\star, q_{\boldsymbol{u}}^\star)$ should lead to **zero cost** for any possible realization of the noise.

$$\boxed{J(q^\star \mid Y_m(q^\star), \zeta) = 0}$$

Note that this can always be made true by taking sufficiently high values of ζ!

But this reduces the quality of the estimation!

ϵ-observability of an observation target

$$q^{(1)} \qquad q^{(2)}$$

$$\begin{bmatrix} x_0^{(1)} \\ p^{(1)} \\ \boldsymbol{u} \\ \boldsymbol{0} \end{bmatrix} \qquad \begin{bmatrix} x_0^{(2)} \\ p^{(2)} \\ \boldsymbol{u} \\ \boldsymbol{\nu} \end{bmatrix}$$

Notation: $q^{(1)} \bowtie q^{(2)}$ (comparable pair)

Two scenarios that share the same input profile while one is free of noise.

ϵ-observability of an observation target

$$q^{(1)} \qquad q^{(2)}$$

$$\begin{bmatrix} x_0^{(1)} \\ p^{(1)} \\ \boldsymbol{u} \\ \boldsymbol{0} \end{bmatrix} \quad \begin{bmatrix} x_0^{(2)} \\ p^{(2)} \\ \boldsymbol{u} \\ \boldsymbol{\nu} \end{bmatrix} \quad \xrightarrow{\text{If}} \quad J(q^{(1)} \mid Y_m(q^{(2)}), \zeta) = 0$$

Notation: $q^{(1)} \bowtie q^{(2)}$ (comparable pair)

Then: $q^{(1)} \equiv q^{(2)}$ (indistinguishable pair)

Two scenarios that share the same input profile while one is free of noise.

ϵ-observability of an observation target

$$q^{(1)} \qquad\qquad q^{(2)}$$

$$\begin{bmatrix} x_0^{(1)} \\ p^{(1)} \\ \boldsymbol{u} \\ \mathbf{0} \end{bmatrix} \qquad \begin{bmatrix} x_0^{(2)} \\ p^{(2)} \\ \boldsymbol{u} \\ \boldsymbol{\nu} \end{bmatrix} \qquad \xrightarrow{\text{If}} \qquad J(q^{(1)} \mid Y_m(q^{(2)}), \zeta) = 0$$

Notation: $q^{(1)} \bowtie q^{(2)}$ (comparable pair)

Then: $q^{(1)} \equiv q^{(2)}$ (indistinguishable pair)

Two scenarios that share the same input profile while one is free of noise.

ϵ-observability of a target z

Given an observation target $z = T(x,p)$, z is said to be **ϵ-observable on $\mathbb{Q}$** iff:

1. $\exists$ a dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$ that is **consistent on** $\mathbb{Q}$
2. $\forall (q^{(1)}, q^{(2)}) \in \mathbb{Q}^2$:

$$\left(q^{(1)} \bowtie q^{(2)}\right) \text{ AND } \left(q^{(1)} \equiv q^{(2)}\right) \;\Rightarrow\; \|q_z^{(1)} - q_z^{(2)}\| \leq \epsilon$$

where $q_z := T(q_x, q_p)$. In other words, only pairs with ϵ-distant observable targets can be both comparable and indistinguishable.

ϵ-observability of an observation target

$$q^{(1)} \qquad q^{(2)}$$

$$\begin{bmatrix} x_0^{(1)} \\ p^{(1)} \\ \boldsymbol{u} \\ \boldsymbol{0} \end{bmatrix} \qquad \begin{bmatrix} x_0^{(2)} \\ p^{(2)} \\ \boldsymbol{u} \\ \boldsymbol{\nu} \end{bmatrix} \quad \xrightarrow{\text{If}} \quad J(q^{(1)} \mid Y_m(q^{(2)}), \zeta) = 0$$

Consistency on $\mathbb{Q}$

$$\boxed{\forall q \in \mathbb{Q}, \quad J(q \mid Y_m(q), \zeta) = 0}$$

Notation: $q^{(1)} \bowtie q^{(2)}$ (comparable pair)

Then: $q^{(1)} \equiv q^{(2)}$ (indistinguishable pair)

Two scenarios that share the same input profile while one is free of noise.

ϵ-observability of a target z

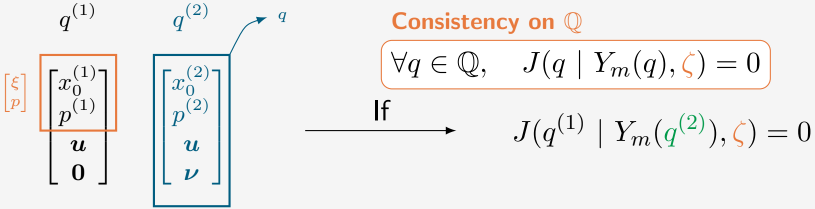
Given an observation target $z = T(x, p)$, z is said to be **ϵ-observable on $\mathbb{Q}$** iff:

1. $\exists$ a dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$ that is **consistent on $\mathbb{Q}$**
2. $\forall (q^{(1)}, q^{(2)}) \in \mathbb{Q}^2$:

$$\left(q^{(1)} \bowtie q^{(2)}\right) \text{ AND } \left(q^{(1)} \equiv q^{(2)}\right) \;\Rightarrow\; \|q_z^{(1)} - q_z^{(2)}\| \leq \epsilon$$

where $q_z := T(q_x, q_p)$. In other words, only pairs with ϵ-distant observable targets can be both comparable and indistinguishable.

ϵ-observability of an observation target



$$q^{(1)} \qquad q^{(2)} \qquad q$$

$$\begin{bmatrix}\xi\\ p\end{bmatrix} \quad \begin{bmatrix} \boxed{\begin{matrix} x_0^{(1)} \\ p^{(1)} \end{matrix}} \\ \boldsymbol{u} \\ \boldsymbol{0} \end{bmatrix} \qquad \boxed{\begin{bmatrix} x_0^{(2)} \\ p^{(2)} \\ \boldsymbol{u} \\ \boldsymbol{\nu} \end{bmatrix}} \qquad \xrightarrow{\text{If}} \qquad J(q^{(1)} \mid Y_m(q^{(2)}), \zeta) = 0$$

Consistency on $\mathbb{Q}$

$$\boxed{\forall q \in \mathbb{Q}, \quad J(q \mid Y_m(q), \zeta) = 0}$$

Notation: $q^{(1)} \bowtie q^{(2)}$ (comparable pair)

Then: $q^{(1)} \equiv q^{(2)}$ (indistinguishable pair)

$\Rightarrow$ The ϵ-observability can be investigated by inspecting realizations of the form:

$$w = \begin{bmatrix} q \\ \xi \\ p \end{bmatrix} \in \mathbb{Q} \times \mathbb{X} \times \mathbb{P}$$

for which a constraint of the form:

$$g(\theta, w) = 0 \qquad \text{where } \theta := (\epsilon, \zeta)$$

should be satisfied

ϵ-observability of a target z

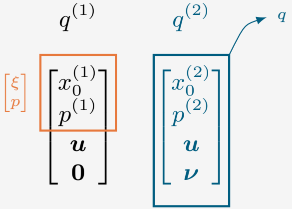
Given an observation target $z = T(x, p)$, z is said to be **ϵ-observable on $\mathbb{Q}$** iff:

1. $\exists$ a dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$ that is **consistent on $\mathbb{Q}$**
2. $\forall (q^{(1)}, q^{(2)}) \in \mathbb{Q}^2$:

$$\left(q^{(1)} \bowtie q^{(2)}\right) \text{ AND } \left(q^{(1)} \equiv q^{(2)}\right) \;\Rightarrow\; \|q_z^{(1)} - q_z^{(2)}\| \leq \epsilon$$

where $q_z := T(q_x, q_p)$. In other words, only pairs with ϵ-distant observable targets can be both comparable and indistinguishable.

ϵ-observability of an observation target



$$q^{(1)} \qquad q^{(2)} \qquad q$$

$$\begin{bmatrix}\xi\\ p\end{bmatrix} \quad \begin{bmatrix}\boxed{\begin{matrix}x_0^{(1)}\\ p^{(1)}\end{matrix}}\\ \boldsymbol{u}\\ \mathbf{0}\end{bmatrix} \quad \boxed{\begin{bmatrix}x_0^{(2)}\\ p^{(2)}\\ \boldsymbol{u}\\ \boldsymbol{\nu}\end{bmatrix}}$$

Notation: $q^{(1)} \bowtie q^{(2)}$ (comparable pair)

$$A(w) \Rightarrow B(w) \Leftrightarrow 0 = f(w) = \begin{cases} 0 & \text{if}(\bar{A}) \cup (A \cap B) \\ 1 & \text{otherwise} \end{cases}$$

⇒ The ϵ-observability can be investigated by inspecting realizations of the form:

$$w = \begin{bmatrix} q \\ \xi \\ p \end{bmatrix} \in \mathbb{Q} \times \mathbb{X} \times \mathbb{P}$$

for which a constraint of the form:

$$g(\theta, w) = 0 \quad \text{where } \theta := (\epsilon, \zeta)$$

should be satisfied

Consistency on $\mathbb{Q}$

$$\boxed{\forall q \in \mathbb{Q}, \quad J(q \mid Y_m(q), \zeta) = 0}$$

$$\xrightarrow{\text{If}} \qquad J(q^{(1)} \mid Y_m(q^{(2)}), \zeta) = 0$$

Then: $q^{(1)} \equiv q^{(2)}$ (indistinguishable pair)

ϵ-observability of a target z

Given an observation target $z = T(x, p)$, z is said to be **ϵ-observable on $\mathbb{Q}$** iff:

1. $\exists$ a dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$ that is **consistent on $\mathbb{Q}$**
2. $\forall (q^{(1)}, q^{(2)}) \in \mathbb{Q}^2$:

$$\left(q^{(1)} \bowtie q^{(2)}\right) \text{ AND } \left(q^{(1)} \equiv q^{(2)}\right) \Rightarrow \|q_z^{(1)} - q_z^{(2)}\| \le \epsilon$$

where $q_z := T(q_x, q_p)$. In other words, only pairs with ϵ-distant observable targets can be both comparable and indistinguishable.

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x,p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x, p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x, p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x,p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x,p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x,p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X}\times\mathbb{P}\times\mathbb{U}\times\mathbb{V}}_{\mathbb{Q}}\times\mathbb{X}\times\mathbb{P}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x, p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

There exists a computable scalar map:

$$g(\theta, w)$$

where $\theta = (\epsilon, \zeta)$ such that:

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x, p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

There exists a computable scalar map:

$$g(\theta, w)$$

where $\theta = (\epsilon, \zeta)$ such that:

if the following condition holds:

$$\boxed{\forall w \in \mathbb{W}, \quad g(\theta, w) = 0}$$

then, solving:

$$(x_0^\star, p^\star) \leftarrow \min_{x_0, p} J\left(\begin{pmatrix} x_0 \\ p \\ \boldsymbol{u} \\ 0 \end{pmatrix} \mid Y_m, \zeta\right)$$

leads to an estimation error on the observation target z that never exceeds ϵ.

and this holds true for any possible realizations inside the sets $\mathbb{X}$, $\mathbb{P}$, $\mathbb{U}$ and $\mathbb{V}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x,p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

There exists a computable scalar map:

$$g(\theta, w)$$

where $\theta = (\epsilon, \zeta)$ such that:

if the following condition holds:

$$\boxed{\forall w \in \mathbb{W}, \quad g(\theta, w) = 0}$$

A robust constraint satisfaction condition

then, solving:

$$(x_0^\star, p^\star) \leftarrow \min_{x_0,p} J\left(\begin{pmatrix} x_0 \\ p \\ \boldsymbol{u} \\ 0 \end{pmatrix} \mid Y_m, \zeta\right)$$

leads to an estimation error on the observation target z that never exceeds ϵ.

and this holds true for any possible realizations inside the sets $\mathbb{X}$, $\mathbb{P}$, $\mathbb{U}$ and $\mathbb{V}$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x,p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X}\times\mathbb{P}\times\mathbb{U}\times\mathbb{V}}_{\mathbb{Q}}\times\mathbb{X}\times\mathbb{P}$

There exists a computable scalar map:

$$g(\theta, w)$$

where $\theta = (\epsilon, \zeta)$ such that:

if the following condition holds:

$$\boxed{\forall w \in \mathbb{W}, \quad g(\theta, w) = 0}$$

A robust constraint satisfaction condition

then, solving:

$$(x_0^\star, p^\star) \leftarrow \min_{x_0,p} J\left(\begin{pmatrix} x_0 \\ p \\ \boldsymbol{u} \\ 0 \end{pmatrix} \mid Y_m, \zeta\right)$$

leads to an estimation error on the observation target z that never exceeds ϵ.

Optimal design

$$\min_{\theta=(\epsilon,\zeta)} \theta_1 \mid g(\theta, w) = 0 \ \forall w \in \mathbb{W}$$

Partial ϵ-observability: a robust constraint satisfaction

Given:

- A dynamic model with uncertain parameter & noisy measurements
- An observation target $z = T(x,p)$
- A dead-zone size vector $\zeta \in \mathbb{R}_+^{n_y}$
- A desired precision $\epsilon > 0$
- A set of interest $\mathbb{W} := \underbrace{\mathbb{X} \times \mathbb{P} \times \mathbb{U} \times \mathbb{V}}_{\mathbb{Q}} \times \mathbb{X} \times \mathbb{P}$

There exists a computable scalar map:

$$g(\theta, w)$$

where $\theta = (\epsilon, \zeta)$ such that:

if the following condition holds:

$$\boxed{\forall w \in \mathbb{W}, \quad g(\theta, w) = 0}$$

A robust constraint satisfaction condition

then, solving:

$$(x_0^\star, p^\star) \leftarrow \min_{x_0,p} J\left(\begin{pmatrix} x_0 \\ p \\ \boldsymbol{u} \\ 0 \end{pmatrix} \mid Y_m, \zeta\right)$$

leads to an estimation error on the observation target z that never exceeds ϵ.

Optimal design

Intractable
over-pessimistic

$$\min_{\theta=(\epsilon,\zeta)} \theta_1 \mid g(\theta, w) = 0 \quad \forall w \in \mathbb{W}$$

Recalls on probabilistic certification

Probabilistic Certification

A general and **tractable** framework to face high uncertainties and still have something to **guarantee** with an EXPLICIT (hopefully high) **probability**.

Probabilistic certification as a relaxed formulation

Optimal Robust constraints satisfaction problem

$$\min_{\theta\in\Theta} J(\theta) \qquad \text{s.t} \qquad (\forall w \in \mathbb{W}) \quad g(\theta, w) \leq 0$$

Probabilistic certification as a relaxed formulation

Optimal Robust constraints satisfaction problem

$$\min_{\theta\in\Theta} J(\theta) \qquad \text{s.t} \qquad (\forall w\in\mathbb{W}) \quad g(\theta,w)\le 0$$

1. Extremely **hard** to solve
2. Uselessly **conservative**
    - worst case analysis
    - even extremely rare bad configurations are accommodated for

Probabilistic certification as a relaxed formulation

Optimal Robust constraints satisfaction problem

$$\min_{\theta\in\Theta} J(\theta) \qquad \text{s.t} \qquad (\forall w \in \mathbb{W}) \quad g(\theta, w) \leq 0$$

1. Extremely **hard** to solve
2. Uselessly **conservative**
   - worst case analysis
   - even extremely rare bad configurations are accommodated for

Relaxed formulations $\Rightarrow$ Probabilistic certification

Probabilistic certification as a relaxed formulation

Define the constraints violation indicator:

$$I(\theta, w) := \begin{cases} 0 & \text{if } g(\theta, w) \leq 0 \\ 1 & \text{otherwise (constraints are violated)} \end{cases}$$

Probabilistic certification as a relaxed formulation

Define the constraints violation indicator:

$$I(\theta, w) := \begin{cases} 0 & \text{if } g(\theta, w) \leq 0 \\ 1 & \text{otherwise (constraints are violated)} \end{cases}$$

Robust constraint $$(\forall w \in \mathbb{W}) \quad g(\theta, w) \leq 0$$

Probabilistic certification as a relaxed formulation

Define the constraints violation indicator:

$$I(\theta,w) := \begin{cases} 0 & \text{if } g(\theta,w) \leq 0 \\ 1 & \text{otherwise (constraints are violated)} \end{cases}$$

First Relaxation (R_1) $$\Pr\Big\{I(\theta,w)=1\Big\} \leq \eta$$

Probabilistic certification as a relaxed formulation

Define the constraints violation indicator:

$$I(\theta, w) := \begin{cases} 0 & \text{if } g(\theta, w) \leq 0 \\ 1 & \text{otherwise (constraints are violated)} \end{cases}$$

First Relaxation (R_1) $\quad \Pr\Big\{I(\theta, w) = 1\Big\} \leq \eta$

Second Relaxation ($R_2(N)$) $\quad \dfrac{1}{N}\sum_{i=1}^{N}\big[I(\theta, w^{(i)})\big] \leq \dfrac{m}{N} \leq \eta$

Probabilistic certification as a relaxed formulation

Define the constraints violation indicator:

$$I(\theta, w) := \begin{cases} 0 & \text{if } g(\theta, w) \leq 0 \\ 1 & \text{otherwise (constraints are violated)} \end{cases}$$

First Relaxation (R_1) $\quad \Pr\Big\{I(\theta, w) = 1\Big\} \leq \eta$

Second Relaxation ($R_2(N)$) $\quad \dfrac{1}{N}\sum_{i=1}^{N}\big[I(\theta, w^{(i)})\big] \leq \dfrac{m}{N} \leq \eta$

For any given m, N must be sufficiently high to get

- ✓ $N \geq \dfrac{m}{\eta}$
- ✓ $\Pr\Big\{R_2(N) \cap (!R_1)\Big\} \leq \delta$

Probabilistic certification as a relaxed formulation

Define the constraints violation indicator:

$$I(\theta, w) := \begin{cases} 0 & \text{if } g(\theta, w) \leq 0 \\ 1 & \text{otherwise (constraints are violated)} \end{cases}$$

First Relaxation (R_1) $\qquad \Pr\Big\{I(\theta, w) = 1\Big\} \leq \eta$

Second Relaxation ($R_2(N)$) $\qquad \dfrac{1}{N}\sum_{i=1}^{N}\big[I(\theta, w^{(i)})\big] \leq \dfrac{m}{N} \leq \eta$

For any given m, N must be sufficiently high to get

- ✓ $N \geq \dfrac{m}{\eta}$
- ✓ $\Pr\Big\{R_2(N) \cap (!R_1)\Big\} \leq \delta$

What value for N?

Probabilistic certification as a relaxed formulation

When the admissible set Θ is discrete with cardinality n_Θ

$$N \geq \frac{1}{\eta}\left(m + \ln(\frac{n_\Theta}{\delta}) + \left(2m\ln(\frac{n_\Theta}{\delta})\right)^{1/2}\right)$$

more formulas are available (Alamo et al. IEEE-TAC 2009)

m	positive integer ($m = 1, 5, 10$ etc.)
n_Θ	card(Θ)
η	Precision
δ	Confidence parameter

Values of N for $\delta = 0.01$, $m = 1$

n_Θ	$\eta = 0.1$	$\eta = 0.05$	$\eta = 0.01$	$\eta = 0.001$
1	132	264	1317	13164
5	154	308	1536	15354
10	163	326	1628	16280
100	193	386	1930	19299
1000	223	445	2225	22249
10000	252	503	2515	25148

NOTA

- ✓ dim(w) does not matter !!!!
- ✓ $(n_\Theta, \delta) \rightarrow$ logarithmically !!

Alamo et al. Randomized strategies for probabilistic solutions of uncertain feasibility and . . . IEEE TAC, 2009.

Algorithm & complexity analysis

Optimal Design

$$\min_{\theta:=(\epsilon,\zeta)} \epsilon \text{ under } \sum_{i=1}^{N} I(\theta, w^{(i)}) \leq m$$

where

- $N = N(\delta, \eta, m)$
- I involves the definition of g
- g involves the cost function J

$$J(q\ |\boldsymbol{y}, \zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left| \frac{1}{k} \sum_{j=1}^{k} e_j^{[i]} \right| - \left[\frac{N}{k} \right] \cdot \zeta_i \right\rfloor$$

Algorithm & complexity analysis

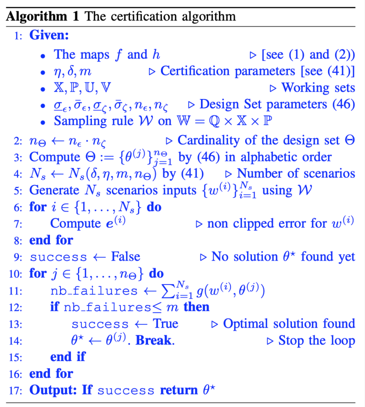
Optimal Design

$$\min_{\theta:=(\epsilon,\zeta)} \epsilon \text{ under } \sum_{i=1}^{N} I(\theta, w^{(i)}) \leq m$$

where

- $N = N(\delta, \eta, m)$
- I involves the definition of g
- g involves the cost function J

$$J(q \;|\boldsymbol{y},\zeta)) := \sum_{i=1}^{n_y}\left\lfloor \left|\frac{1}{k}\sum_{j=1}^{k} e_j^{[i]}\right| - \left[\frac{N}{k}\right]\cdot\zeta_i \right\rfloor$$



Algorithm 1 The certification algorithm

```
Given:
    • The maps $f$ and $h$                                  ▷ [see (1) and (2))
    • $\eta, \delta, m$                     ▷ Certification parameters [see (41)]
    • $\mathbb{X}, \mathbb{P}, \mathbb{U}, \mathbb{V}$                         ▷ Working sets
    • $\underline{\sigma}_\epsilon, \bar{\sigma}_\epsilon, \underline{\sigma}_\zeta, \bar{\sigma}_\zeta, n_\epsilon, n_\zeta$     ▷ Design Set parameters (46)
    • Sampling rule $\mathcal{W}$ on $\mathbb{W} = \mathbb{Q} \times \mathbb{X} \times \mathbb{P}$
$n_\Theta \leftarrow n_\epsilon \cdot n_\zeta$                    ▷ Cardinality of the design set $\Theta$
Compute $\Theta := \{\theta^{(j)}\}_{j=1}^{n_\Theta}$ by (46) in alphabetic order
$N_s \leftarrow N_s(\delta, \eta, m, n_\Theta)$ by (41)      ▷ Number of scenarios
Generate $N_s$ scenarios inputs $\{w^{(i)}\}_{i=1}^{N_s}$ using $\mathcal{W}$
for $i \in \{1, \dots, N_s\}$ do
    Compute $e^{(i)}$                       ▷ non clipped error for $w^{(i)}$
end for
success ← False                           ▷ No solution $\theta^\star$ found yet
for $j \in \{1, \dots, n_\Theta\}$ do
    nb_failures ← $\sum_{i=1}^{N_s} g(w^{(i)}, \theta^{(j)})$
    if nb_failures≤ $m$ then
        success ← True                    ▷ Optimal solution found
        $\theta^\star \leftarrow \theta^{(j)}$. Break.                        ▷ Stop the loop
    end if
end for
Output: If success return $\theta^\star$
```

Algorithm & complexity analysis

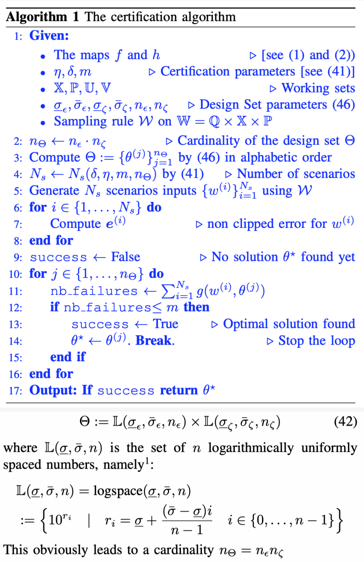
Optimal Design

$$\min_{\theta:=(\epsilon,\zeta)} \epsilon \text{ under } \sum_{i=1}^{N} I(\theta, w^{(i)}) \leq m$$

where

- $N = N(\delta, \eta, m)$
- I involves the definition of g
- g involves the cost function J

$$J(q\ |\boldsymbol{y}, \zeta)) := \sum_{i=1}^{n_y} \left\lfloor \left| \frac{1}{k} \sum_{j=1}^{k} e_j^{[i]} \right| - \left[\frac{N}{k}\right] \cdot \zeta_i \right\rfloor$$



Algorithm 1 The certification algorithm

```
Given:
    • The maps $f$ and $h$                                   ▷ [see (1) and (2)]
    • $\eta, \delta, m$                        ▷ Certification parameters [see (41)]
    • $\mathbb{X}, \mathbb{P}, \mathbb{U}, \mathbb{V}$                          ▷ Working sets
    • $\underline{\sigma}_\epsilon, \bar{\sigma}_\epsilon, \underline{\sigma}_\zeta, \bar{\sigma}_\zeta, n_\epsilon, n_\zeta$      ▷ Design Set parameters (46)
    • Sampling rule $\mathcal{W}$ on $\mathbb{W} = \mathbb{Q} \times \mathbb{X} \times \mathbb{P}$
$n_\Theta \leftarrow n_\epsilon \cdot n_\zeta$                 ▷ Cardinality of the design set $\Theta$
Compute $\Theta := \{\theta^{(j)}\}_{j=1}^{n_\Theta}$ by (46) in alphabetic order
$N_s \leftarrow N_s(\delta, \eta, m, n_\Theta)$ by (41)        ▷ Number of scenarios
Generate $N_s$ scenarios inputs $\{w^{(i)}\}_{i=1}^{N_s}$ using $\mathcal{W}$
for $i \in \{1, \dots, N_s\}$ do
    Compute $e^{(i)}$                         ▷ non clipped error for $w^{(i)}$
end for
success ← False                              ▷ No solution $\theta^\star$ found yet
for $j \in \{1, \dots, n_\Theta\}$ do
    nb_failures ← $\sum_{i=1}^{N_s} g(w^{(i)}, \theta^{(j)})$
    if nb_failures$\leq m$ then
        success ← True                      ▷ Optimal solution found
        $\theta^\star \leftarrow \theta^{(j)}$. Break.                          ▷ Stop the loop
    end if
end for
Output: If success return $\theta^\star$
```

$$\Theta := \mathbb{L}(\underline{\sigma}_\epsilon, \bar{\sigma}_\epsilon, n_\epsilon) \times \mathbb{L}(\underline{\sigma}_\zeta, \bar{\sigma}_\zeta, n_\zeta) \tag{42}$$

where $\mathbb{L}(\underline{\sigma}, \bar{\sigma}, n)$ is the set of n logarithmically uniformly spaced numbers, namely[1]:

$$\mathbb{L}(\underline{\sigma}, \bar{\sigma}, n) = \text{logspace}(\underline{\sigma}, \bar{\sigma}, n)$$
$$:= \left\{ 10^{r_i} \quad | \quad r_i = \underline{\sigma} + \frac{(\bar{\sigma} - \underline{\sigma})i}{n-1} \quad i \in \{0, \dots, n-1\} \right\}$$

This obviously leads to a cardinality $n_\Theta = n_\epsilon n_\zeta$

Algorithm & complexity analysis

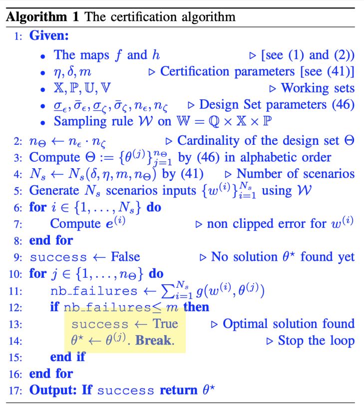
Optimal Design

$$\min_{\theta:=(\epsilon,\zeta)} \epsilon \text{ under } \sum_{i=1}^{N} I(\theta, w^{(i)}) \leq m$$

where

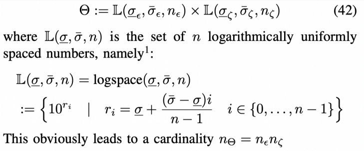
- $N = N(\delta, \eta, m)$
- I involves the definition of g
- g involves the cost function J

$$\boxed{J(q\ |\boldsymbol{y}, \zeta)) := \sum_{i=1}^{n_y} \left[\left| \frac{1}{k} \sum_{j=1}^{k} e_j^{[i]} \right| - \left[\frac{N}{k} \right] \cdot \zeta_i \right]}$$



Algorithm 1 The certification algorithm

```
Given:
    • The maps $f$ and $h$                          ▷ [see (1) and (2))
    • $\eta, \delta, m$                             ▷ Certification parameters [see (41)]
    • $\mathbb{X}, \mathbb{P}, \mathbb{U}, \mathbb{V}$     ▷ Working sets
    • $\underline{\sigma}_\epsilon, \bar{\sigma}_\epsilon, \underline{\sigma}_\zeta, \bar{\sigma}_\zeta, n_\epsilon, n_\zeta$   ▷ Design Set parameters (46)
    • Sampling rule $\mathcal{W}$ on $\mathbb{W} = \mathbb{Q} \times \mathbb{X} \times \mathbb{P}$
$n_\Theta \leftarrow n_\epsilon \cdot n_\zeta$          ▷ Cardinality of the design set $\Theta$
Compute $\Theta := \{\theta^{(j)}\}_{j=1}^{n_\Theta}$ by (46) in alphabetic order
$N_s \leftarrow N_s(\delta, \eta, m, n_\Theta)$ by (41)    ▷ Number of scenarios
Generate $N_s$ scenarios inputs $\{w^{(i)}\}_{i=1}^{N_s}$ using $\mathcal{W}$
for $i \in \{1, \dots, N_s\}$ do
    Compute $\boldsymbol{e}^{(i)}$                  ▷ non clipped error for $w^{(i)}$
end for
success ← False                                    ▷ No solution $\theta^\star$ found yet
for $j \in \{1, \dots, n_\Theta\}$ do
    nb_failures ← $\sum_{i=1}^{N_s} g(w^{(i)}, \theta^{(j)})$
    if nb_failures $\leq m$ then
        success ← True                             ▷ Optimal solution found
        $\theta^\star \leftarrow \theta^{(j)}$. Break.      ▷ Stop the loop
    end if
end for
Output: If success return $\theta^\star$
```



$$\Theta := \mathbb{L}(\underline{\sigma}_\epsilon, \bar{\sigma}_\epsilon, n_\epsilon) \times \mathbb{L}(\underline{\sigma}_\zeta, \bar{\sigma}_\zeta, n_\zeta) \tag{42}$$

where $\mathbb{L}(\underline{\sigma}, \bar{\sigma}, n)$ is the set of n logarithmically uniformly spaced numbers, namely[1]:

$$\mathbb{L}(\underline{\sigma}, \bar{\sigma}, n) = \text{logspace}(\underline{\sigma}, \bar{\sigma}, n)$$
$$:= \left\{ 10^{r_i} \quad | \quad r_i = \underline{\sigma} + \frac{(\bar{\sigma} - \underline{\sigma})i}{n-1} \quad i \in \{0, \dots, n-1\} \right\}$$

This obviously leads to a cardinality $n_\Theta = n_\epsilon n_\zeta$

Application example

Parallel reactor $R \to (P_1, P_2)$

$$\begin{aligned}
\dot{x}_1 &= 1 - w_1 x_1^2 e^{-1/x_3} - w_2 e^{-w_3/x_3} - x_1 \\
\dot{x}_2 &= w_1 x - 1^2 e^{-1/x_3} - x_2 \\
\dot{x}_3 &= u - x_3
\end{aligned}$$

- x_1, x_2 concentrations of R and P_1
- x_3 temperature of the mixture
- u is the jacket temperature
- x_2 and u are measured
- Three possible definitions of z:
    - $z = x$
    - $z = x_1$
    - $z = x_3$

Application example

Parallel reactor $R \rightarrow (P_1, P_2)$

$$
\begin{aligned}
\dot{x}_1 &= 1 - w_1 x_1^2 e^{-1/x_3} - w_2 e^{-w_3/x_3} - x_1 \\
\dot{x}_2 &= w_1 x - 1^2 e^{-1/x_3} - x_2 \\
\dot{x}_3 &= u - x_3
\end{aligned}
$$

- x_1, x_2 concentrations of R and P_1
- x_3 temperature of the mixture
- u is the jacket temperature
- x_2 and u are measured
- Three possible definitions of z:
  - $z = x$
  - $z = x_1$
  - $z = x_3$

Investigated set/statistics options

$\mathbb{X}$ A uniform distribution over a hyperbox $\mathbb{X}$

$\mathbb{P}$ Two possibilities are investigated, namely:

1. A uniform distribution over $\mathbb{P}$
2. A Gaussian distribution around a nominal value.

$\mathbb{U}^N$ Two possibilities are investigated, namely:

1. A uniform distribution over $\mathbb{U}^N$
2. A random choice consisting in sequences (elements of $\mathbb{U}$) of the form ($i \in \{1, \dots, N\}$):

$$
u_i = \text{Sat}_{\mathbb{U}}\left[\sum_{j=1}^{n_f} \beta_j \sin\left(\frac{2j\pi(i\tau)}{N\tau} + \varphi_j\right)\right]
$$

$\mathbb{V}^N$ Gaussian white noise (with different variances) are used to represent measurement noise.

Application example

Parallel reactor $R \to (P_1, P_2)$

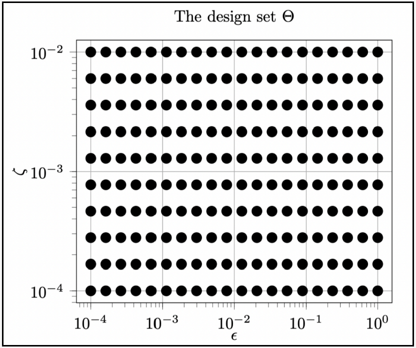
$$\dot{x}_1 = 1 - w_1 x_1^2 e^{-1/x_3} - w_2 e^{-w_3/x_3} - x_1$$

$$\dot{x}_2 = w_1 x - 1^2 e^{-1/x_3} - x_2$$

$$\dot{x}_3 = u - x_3$$

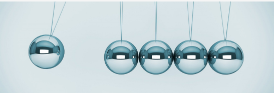
- x_1, x_2 concentrations of R and P_1
- x_3 temperature of the mixture
- u is the jacket temperature
- x_2 and u are measured
- Three possible definitions of z:
  - $z = x$
  - $z = x_1$
  - $z = x_3$

The design set



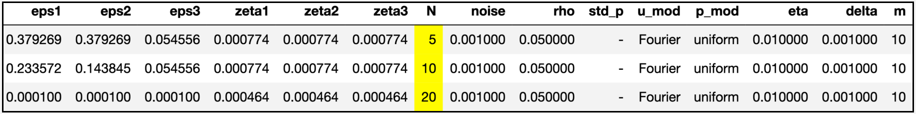
The impact of players on partial observability

- Impact of the **observation horizon**
- Impact of the **measurement noise**
- Impact of the **uncertainty level**
- Impact of the **input type**
- impact of the **certification parameters**



- **Computation time**

Impact of the observation horizon



eps1	eps2	eps3	zeta1	zeta2	zeta3	N	noise	rho	std_p	u_mod	p_mod	eta	delta	m
0.379269	0.379269	0.054556	0.000774	0.000774	0.000774	5	0.001000	0.050000	-	Fourier	uniform	0.010000	0.001000	10
0.233572	0.143845	0.054556	0.000774	0.000774	0.000774	10	0.001000	0.050000	-	Fourier	uniform	0.010000	0.001000	10
0.000100	0.000100	0.000100	0.000464	0.000464	0.000464	20	0.001000	0.050000	-	Fourier	uniform	0.010000	0.001000	10

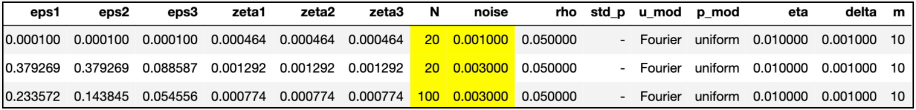
$$z_1 = x,\ z_2 = x_1,\ z_3 = x_3$$

Note that $N = 20$ is needed in order to achieve the certification with the lowest values $\epsilon = 10^{-4}$ considered in the design set Θ over the three observation-target variables z_i, $i = 1, 2, 3$.

Otherwise, **indistinguishability might occur** with quite high error values on the targeted indicators.

Note that only three values of N are studied here, lower values of $N \in [10, 20]$ would have probably be sufficient to achieve the high precision certification results.

Impact of the measurement noise



eps1	eps2	eps3	zeta1	zeta2	zeta3	N	noise	rho	std_p	u_mod	p_mod	eta	delta	m
0.000100	0.000100	0.000100	0.000464	0.000464	0.000464	20	0.001000	0.050000	-	Fourier	uniform	0.010000	0.001000	10
0.379269	0.379269	0.088587	0.001292	0.001292	0.001292	20	0.003000	0.050000	-	Fourier	uniform	0.010000	0.001000	10
0.233572	0.143845	0.054556	0.000774	0.000774	0.000774	100	0.003000	0.050000	-	Fourier	uniform	0.010000	0.001000	10

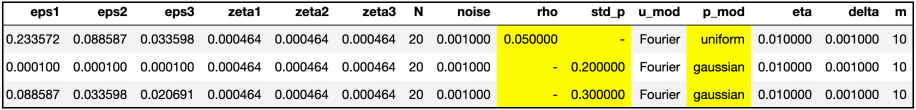
$$z_1 = x,\ z_2 = x_1,\ z_3 = x_3$$

Note that Increasing the noise from 0.001 to 0.003 leads to a **sensitive degradation in the certifiable reconstruction precision**.

By increasing the observation horizon up to $N = 100$ it is possible to recover the levels of precision of the previous setting which was achievable with $N = 10$ and the previous level (0.001) of the noise.

The higher the noise is the longer the observation horizon should be to achieve the same level of certifiable reconstruction precision.

Impact of the uncertainty level

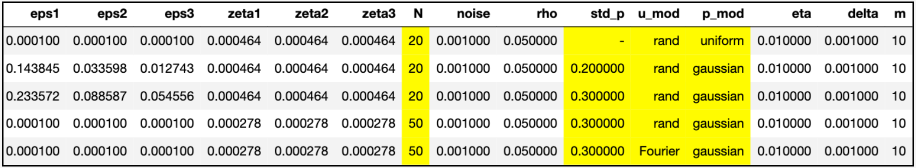


eps1	eps2	eps3	zeta1	zeta2	zeta3	N	noise	rho	std_p	u_mod	p_mod	eta	delta	m
0.233572	0.088587	0.033598	0.000464	0.000464	0.000464	20	0.001000	0.050000	-	Fourier	uniform	0.010000	0.001000	10
0.000100	0.000100	0.000100	0.000464	0.000464	0.000464	20	0.001000	-	0.200000	Fourier	gaussian	0.010000	0.001000	10
0.088587	0.033598	0.020691	0.000464	0.000464	0.000464	20	0.001000	-	0.300000	Fourier	gaussian	0.010000	0.001000	10

$$z_1 = x,\ z_2 = x_1,\ z_3 = x_3$$

Note that The configuration with **uniform distribution** of the parameter vector with $\rho = 0.05$ is **more harmful** to certifiable reconstruction precision than the Gaussian distribution with standard deviation $s_{td} = 0.2$.

Impact of the input type

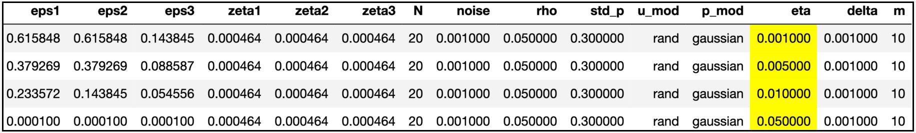


eps1	eps2	eps3	zeta1	zeta2	zeta3	N	noise	rho	std_p	u_mod	p_mod	eta	delta	m
0.000100	0.000100	0.000100	0.000464	0.000464	0.000464	20	0.001000	0.050000	-	rand	uniform	0.010000	0.001000	10
0.143845	0.033598	0.012743	0.000464	0.000464	0.000464	20	0.001000	0.050000	0.200000	rand	gaussian	0.010000	0.001000	10
0.233572	0.088587	0.054556	0.000464	0.000464	0.000464	20	0.001000	0.050000	0.300000	rand	gaussian	0.010000	0.001000	10
0.000100	0.000100	0.000100	0.000278	0.000278	0.000278	50	0.001000	0.050000	0.300000	rand	gaussian	0.010000	0.001000	10
0.000100	0.000100	0.000100	0.000278	0.000278	0.000278	50	0.001000	0.050000	0.300000	Fourier	gaussian	0.010000	0.001000	10

$$z_1 = x,\; z_2 = x_1,\; z_3 = x_3$$

Note that Comparing the first line with the first line of the previous figure suggests that **random input profiles enhance the observability** at least for the setting that is common to these two lines.

Impact of the certification parameters



eps1	eps2	eps3	zeta1	zeta2	zeta3	N	noise	rho	std_p	u_mod	p_mod	eta	delta	m
0.615848	0.615848	0.143845	0.000464	0.000464	0.000464	20	0.001000	0.050000	0.300000	rand	gaussian	0.001000	0.001000	10
0.379269	0.379269	0.088587	0.000464	0.000464	0.000464	20	0.001000	0.050000	0.300000	rand	gaussian	0.005000	0.001000	10
0.233572	0.143845	0.054556	0.000464	0.000464	0.000464	20	0.001000	0.050000	0.300000	rand	gaussian	0.010000	0.001000	10
0.000100	0.000100	0.000100	0.000464	0.000464	0.000464	20	0.001000	0.050000	0.300000	rand	gaussian	0.050000	0.001000	10

$$z_1 = x,\ z_2 = x_1,\ z_3 = x_3$$

Note that A certification precision of $\eta = 10^{-3}$ leads to a certifiable upper bounds on the estimation errors on the observation-target variables which **useless** given the definition of the set $\mathbb{X}$. This is obviously due to the high level of parameter dispersion $s_{td} = 0.3$.

the last line indicates that up to 5% of the samples correspond to the presence of indistinguishable pairs. For the remaining 95% of the cases, an almost zero reconstruction error can be certified **(provided that the optimization problem is correctly solved)**.

CPU

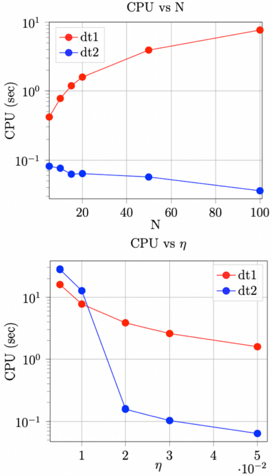


Fig. 8. **Computation times** using $\delta = 0.001$, $m = 10$. (top) cpu (sec) versus the observation horizon N for $\eta = 0.01$. (bottom) cpu (sec) versus η for an observation horizon of $N = 20$.

$\text{CPU} = \text{dt1} + \text{dt2}$

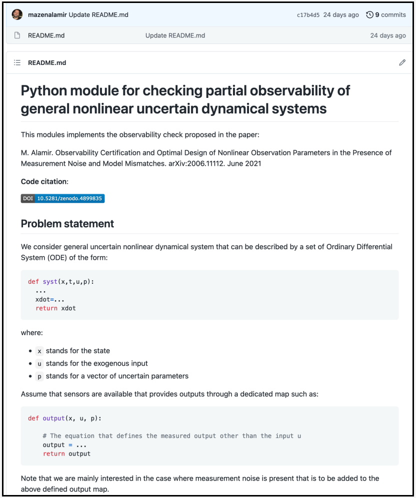
- **dt1** generating the scenarios
- **dt2** finding the optimal design

Recall that the whole scheme is to be executed **off-line** in order to:

1. Evaluate the partial observability
2. Optimize the observation parameter

The GitHub repository

https://github.com/mazenalamir/eps-observability



Python module for checking partial observability of general nonlinear uncertain dynamical systems

This modules implements the observability check proposed in the paper:

M. Alamir. Observability Certification and Optimal Design of Nonlinear Observation Parameters in the Presence of Measurement Noise and Model Mismatches. arXiv:2006.11112. June 2021

Code citation:

DOI 10.5281/zenodo.4899835

Problem statement

We consider general uncertain nonlinear dynamical system that can be described by a set of Ordinary Differential System (ODE) of the form:

```
def syst(x,t,u,p):
  ...
  xdot=...
  return xdot
```

where:

- `x` stands for the state
- `u` stands for the exogenous input
- `p` stands for a vector of uncertain parameters

Assume that sensors are available that provides outputs through a dedicated map such as:

```
def output(x, u, p):

    # The equation that defines the measured output other than the input u
    output = ...
    return output
```

Note that we are mainly interested in the case where measurement noise is present that is to be added to the above defined output map.

MA. Partial Extended Observability Certification and Optimal Design of Moving Horizon Estimators Under Uncertainties.

To appear in IEEE Transactions on Automatic control, July, 2022

https://arxiv.org/abs/2006.11112

Conclusion

- An example of **Data-driven** assessment of an important control-related problem: *dynamic partial extended estimation*.

- The extensive **Data Generation** step is unavoidable in order to derive non pessimistic results.

- **The choice of the sets/statistics remains a non totally solved issue and is certainly a problem-dependent step.**

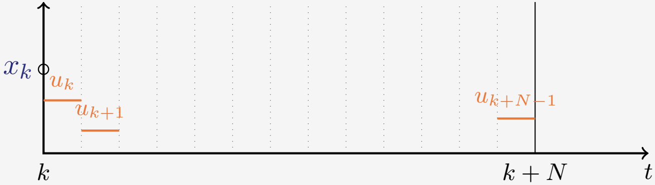
- Data-driven methods $\neq$ ML or AI (at least not always)

1. Data-driven partial observability assessment under uncertainties
2. **Learning-Based Approximate Stochastic NMPC Design**
3. Tractable stochastic NMPC by supervised clustering
4. Data-Driven NMPC by cost function identification
5. Learning-Based Monitoring updating period in Real-time NMPC

RECALLS ON STANDARD MPC

Direct single shooting formulation of MPC:

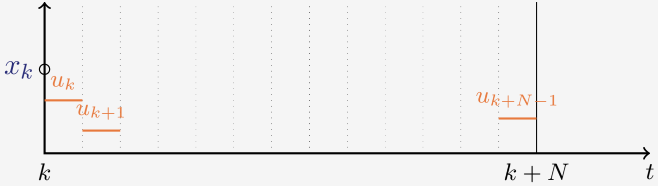
$$\boldsymbol{u}_k^\star \leftarrow \min_{\boldsymbol{u}_k \in \mathbb{U}^N} \Big[J(\boldsymbol{u}_k \mid x_k) \Big] \qquad \boldsymbol{u}_k := \begin{bmatrix} u_k \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \big[\mathbb{R}^{n_u}\big]^N$$



RECALLS ON STANDARD MPC

Direct single shooting formulation of MPC:

$$\boldsymbol{u}_k^\star \leftarrow \min_{\boldsymbol{u}_k \in \mathbb{U}^N} \Big[J(\boldsymbol{u}_k \mid x_k) \Big] \qquad \boldsymbol{u}_k := \begin{bmatrix} u_k \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \big[\mathbb{R}^{n_u}\big]^N$$



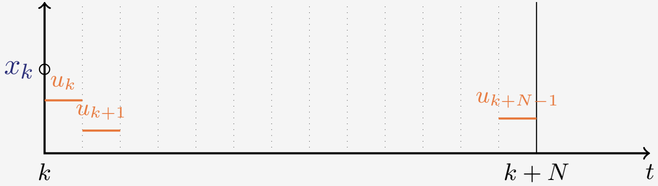
MPC Feedback Control

$$\boxed{K_{\mathrm{MPC}}(x_k) = u_k^\star}$$

RECALLS ON STANDARD MPC

Direct single shooting formulation of MPC:

$$\boldsymbol{u}_k^\star \leftarrow \min_{\boldsymbol{u}_k \in \mathbb{U}^N} \Big[J(\boldsymbol{u}_k \mid x_k)\Big] \qquad \boldsymbol{u}_k := \begin{bmatrix} u_k \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \big[\mathbb{R}^{n_u}\big]^N$$



Standard settings

$$J(\boldsymbol{u}_k \mid x_k) := \sum_{i=1}^{N} \ell(x_{k+i}, u_{k+i-1})$$

$$x_{k+i+1} = f(x_{k+i}, u_{k+i})$$

MPC Feedback Control

$$K_{\text{MPC}}(x_k) = u_k^\star$$

PROBLEM STATEMENT

Dynamics

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w) \\ y_k &= h(x_k, u_k, w) \end{aligned}$$

x the state

u the control

(f, h) known maps

w **constant** unknown parameters

but with known statistics over $\mathbb{W}$

PROBLEM STATEMENT

Dynamics

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w) \\ y_k &= h(x_k, u_k, w) \end{aligned}$$

x the state

u the control

(f, h) known maps

w **constant** unknown parameters

but with known statistics over $\mathbb{W}$

Illustrative example

$$\dot{x}_1 = 1 - w_1 x_1^2 e^{-1/x_3} - w_2 x_1 e^{-w_3/x_3} - x_1$$

$$\dot{x}_2 = w_1 x_1^2 e^{-1/x_3} - x_2$$

$$\dot{x}_3 = u - x_3$$

PROBLEM STATEMENT

Dynamics

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w) \\ y_k &= h(x_k, u_k, w) \end{aligned}$$

x the state
u the control
(f, h) known maps
w **constant** unknown parameters
but with known statistics over $\mathbb{W}$

Illustrative example

$$\begin{aligned} \dot{x}_1 &= 1 - w_1 x_1^2 e^{-1/x_3} - w_2 x_1 e^{-w_3/x_3} - x_1 \\ \dot{x}_2 &= w_1 x_1^2 e^{-1/x_3} - x_2 \\ \dot{x}_3 &= u - x_3 \end{aligned}$$

Ideal Stochastic MPC

$$\mathcal{P}(x): \quad \min_{\boldsymbol{u}} \; \mathbb{E}\Big[J(\boldsymbol{u} \mid x, \cdot)\Big]$$

PROBLEM STATEMENT

Dynamics

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w) \\ y_k &= h(x_k, u_k, w) \end{aligned}$$

x the state

u the control

(f, h) known maps

w **constant** unknown parameters

but with known statistics over $\mathbb{W}$

Illustrative example

$$\dot{x}_1 = 1 - w_1 x_1^2 e^{-1/x_3} - w_2 x_1 e^{-w_3/x_3} - x_1$$

$$\dot{x}_2 = w_1 x_1^2 e^{-1/x_3} - x_2$$

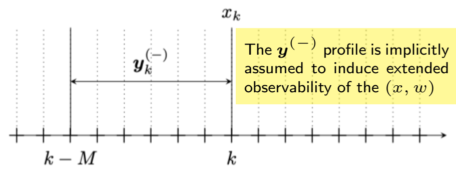
$$\dot{x}_3 = u - x_3$$

Ideal Stochastic MPC

$$\mathcal{P}(x) : \quad \min_{\boldsymbol{u}} \ \mathbb{E}\Big[J(\boldsymbol{u} \mid x, \cdot)\Big]$$

or gathering all the measurements in $\boldsymbol{y}^{(-)}$

$$\mathcal{P}(\boldsymbol{y}^{(-)}) : \quad \min_{\boldsymbol{u}} \ \mathbb{E}\Big[J(\boldsymbol{u} \mid \boldsymbol{y}^{(-)}, \cdot)\Big]$$



The $\boldsymbol{y}^{(-)}$ profile is implicitly assumed to induce extended observability of the (x, w)

More generally

$$\mathcal{P}(\boldsymbol{y}^{(-)}) : \quad \min_{\boldsymbol{u}} \mathbb{E}[J] + \alpha\sigma^{\frac{1}{2}}[J]$$

Possible solutions

Option1 **(Extended Observer)**

Build an extended observer to get

$$(\hat{x}_k, \hat{w}_k)$$

No more need for stochastic setting:

$$\boldsymbol{u}^\star \leftarrow \mathcal{P}(\hat{x}_k, \hat{w}_k)$$

(+) If possible → The best performance
(-) Needs to build the observer!
(-) Computation burden adds to NMPC.

Possible solutions

Option1 **(Extended Observer)**

Build an extended observer to get

$$(\hat{x}_k, \hat{w}_k)$$

No more need for stochastic setting:

$$\boldsymbol{u}^\star \leftarrow \mathcal{P}(\hat{x}_k, \hat{w}_k)$$

(+) If possible → The best performance
(-) Needs to build the observer!
(-) Computation burden adds to NMPC.

Option2 **(Approximate statistics via clustering)**

Standard NMPC with the cost given by

$$\frac{1}{n_c}\sum_{i=1}^{n_c} \mu_j J(\boldsymbol{u}\ | x_k, w_c^{(j)})$$

- $w_c^{(j)}$ centres of computed clusters [obtained via supervised clustering]
- μ_j fraction of populations in cluster j
- n_c moderate thanks to clustering

(-) Might still induce heavy computation!
Multiple shooting optimisation → $N(n_c n_x + n_u)$ d.o.f
$(n_c, N, n_x, n_u) = (10, 100, 3, 1)$ → 3100 instead of 400

Possible solutions

Option1 **(Extended Observer)**

Build an extended observer to get

$$(\hat{x}_k, \hat{w}_k)$$

No more need for stochastic setting:

$$\boldsymbol{u}^\star \leftarrow \mathcal{P}(\hat{x}_k, \hat{w}_k)$$

(+) If possible → The best performance
(-) Needs to build the observer!
(-) Computation burden adds to NMPC.

Option2 **(Approximate statistics via clustering)**

Standard NMPC with the cost given by

$$\frac{1}{n_c}\sum_{i=1}^{n_c} \mu_j J(\boldsymbol{u} \mid x_k, w_c^{(j)})$$

- $w_c^{(j)}$ centres of computed clusters
  [obtained via supervised clustering]
- μ_j fraction of populations in cluster j
- n_c moderate thanks to clustering

(-) Might still induce heavy computation!
Multiple shooting optimisation → $N(n_c n_x + n_u)$ d.o.f
$(n_c, N, n_x, n_u) = (10, 100, 3, 1)$ → 3100 instead of 400

[MA]. On the use of supervised clustering in stochastic NMPC design. IEEE Transactions on automatic control, Volume 65, Issue 12, 2020.

Possible solutions

Option1 **(Extended Observer)**

Build an extended observer to get

$$(\hat{x}_k, \hat{w}_k)$$

No more need for stochastic setting:

$$\boldsymbol{u}^\star \leftarrow \mathcal{P}(\hat{x}_k, \hat{w}_k)$$

(+) If possible → The best performance
(-) Needs to build the observer!
(-) Computation burden adds to NMPC.

Option2 **(Approximate statistics via clustering)**

Standard NMPC with the cost given by

$$\frac{1}{n_c}\sum_{i=1}^{n_c} \mu_j J(\boldsymbol{u} \mid x_k, w_c^{(j)})$$

- $w_c^{(j)}$ ed clusters
  [obtaine g]
- μ_j fra ons in cluster j
- n_c moderate thanks to clustering

(-) Might still induce heavy computation!
Multiple shooting optimisation → $N(n_c n_x + n_u)$ d.o.f
$(n_c, N, n_x, n_u) = (10, 100, 3, 1)$ → 3100 instead of 400

[MA]. On the use of supervised clustering in stochastic NMPC design. IEEE Transactions on automatic control, Volume 65, Issue 12, 2020.

POSSIBLE SOLUTIONS

Option3 **(Approximate Dynamic Programming)**

Approximate the optimal value

$$Q(z) \quad \text{where} \quad z := (x, u)$$

based on the Bellman equation

$$Q(z) = \ell(z) + \gamma \min_{v} [\hat{\mu} + \alpha \hat{\sigma}^{\frac{1}{2}}][Q(f(z), v)]$$

Fixed-point

- ℓ stage cost
- γ discounting rate
- $\hat{\mu}$ approximated statistics over realisations of w

POSSIBLE SOLUTIONS

Option3 **(Approximate Dynamic Programming)**

Approximate the optimal value

$$Q(z) \quad \text{where} \quad z := (x, u)$$

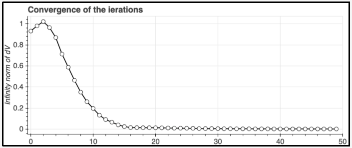
based on the Bellman equation

$$Q(z) = \ell(z) + \gamma \min_v [\hat{\mu} + \alpha\hat{\sigma}^{\frac{1}{2}}][Q(f(z), v)]$$

Fixed-point

- ℓ stage cost
- γ discounting rate
- $\hat{\mu}$ approximated statistics over realisations of w

- Choose a **ML**-based structure for Q
- Fixed-point iteration on Q



POSSIBLE SOLUTIONS

Option3 **(Approximate Dynamic Programming)**

Approximate the optimal value

$$Q(z) \quad \text{where} \quad z := (x,u)$$

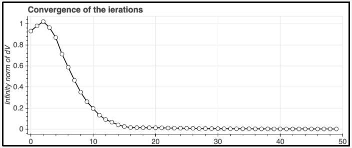
based on the Bellman equation

$$Q(z) = \ell(z) + \gamma \min_{v}[\hat{\mu} + \alpha\hat{\sigma}^{\frac{1}{2}}][Q(f(z),v)]$$

Fixed-point

- ℓ stage cost
- γ discounting rate
- $\hat{\mu}$ approximated statistics over realisations of w

- Choose a **ML**-based structure for Q
- Fixed-point iteration on Q



[MA]. Explicit approximation of stochastic optimal control for combined therapy of cancer. arXiv:1905.04937, 2019.

POSSIBLE SOLUTIONS

Option3 **(Approximate Dynamic Programming)**

Approximate the optimal value

$$Q(z) \quad \text{where} \quad z := (x,u)$$

based on the Bellman equation

$$Q(z) = \ell(z) + \gamma \min_{v} [\hat{\mu} + \alpha\hat{\sigma}^{\frac{1}{2}}][Q(f(z),v)]$$

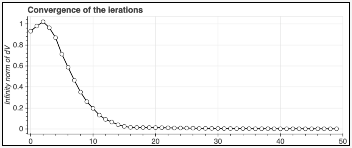
Fixed-point

- ℓ stage cost
- γ discounting rate
- $\hat{\mu}$ approximated statistics over realisations of w

(-) Curse of dimensionality

(-) Fragile choice of the support state domain

- Choose a **ML**-based structure for Q
- Fixed-point iteration on Q



[MA]. Explicit approximation of stochastic optimal control for combined therapy of cancer. arXiv:1905.04937, 2019.

POSSIBLE SOLUTIONS

Option3 **(Approximate Dynamic Programming)**

Approximate the optimal value

$$Q(z) \quad \text{where} \quad z := (x, u)$$

based on the Bellman equation

$$Q(z) = \ell(z) + \gamma \min_{v} [\hat{\mu} + \alpha \hat{\sigma}^{\frac{1}{2}}][Q(f(z), v)]$$

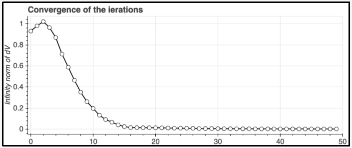
Fixed-point

- ℓ stage cost
- γ discounting rat
- $\hat{\mu}$ approximated statistics over realisations of w

(-) Curse of dimensionality

(-) Fragile choice of the support state domain

- Choose a **ML**-based structure for Q
- Fixed-point iteration on Q



[MA]. Explicit approximation of stochastic optimal control for combined therapy of cancer. arXiv:1905.04937, 2019.

DEFINITION OF THE ASSOCIATED ML PROBLEM (1)

We would like to have a feedback of the form

$$u_k = \mathbf{ML}(\boldsymbol{y}_k^{(-)}) \qquad \text{where} \quad \boldsymbol{y}_k^{(-)} = \begin{bmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_{k-M} \end{bmatrix} \in [\mathbb{R}^{n_y}]^{M+1}$$

- $\boldsymbol{y}_k^{(-)}$ The features
- u_k The label

Where does the label u_k comes from?

DEFINITION OF THE ASSOCIATED ML PROBLEM (2)

Ideally $u_k = \boldsymbol{u}_0^\star(x_k, w)$ where

$$\mathcal{P}(x_k, w) : \qquad \boldsymbol{u}^\star(x_k, w) = \arg \min_{\boldsymbol{u} \in \mathbb{U}^N} \Big[J(\boldsymbol{u} \mid x_k, w) \Big]$$

or in terms of $\boldsymbol{y}_k^{(-)}$ (extended observability required):

$$\mathcal{P}(\boldsymbol{y}_k^{(-)}) : \qquad \boldsymbol{u}^\star(\boldsymbol{y}_k^{(-)}) = \arg \min_{\boldsymbol{u} \in \mathbb{U}^N} \Big[J(\boldsymbol{u} \mid \hat{x}_k, \hat{w}) \Big]$$

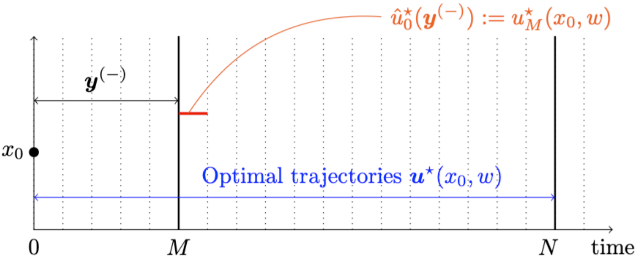
$$\hat{x}_k = \hat{x}(\boldsymbol{y}_k^{(-)})$$

$$\hat{w} = \hat{w}(\boldsymbol{y}_k^{(-)})$$

But we do not want to build observers! (difficult if not impossible)

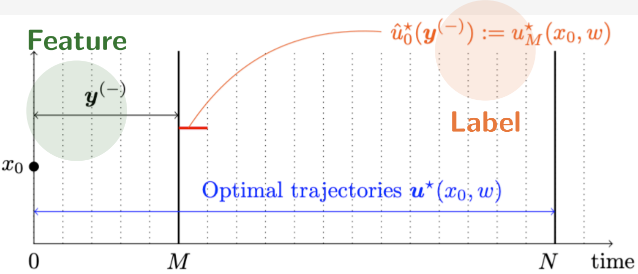
BUILDING THE LEARNING DATA (1)

1. Choose arbitrary pair $q := (x_0, w)$
2. Solve the problem $\mathcal{P}(x_0, w)$ for known (x_0, w)



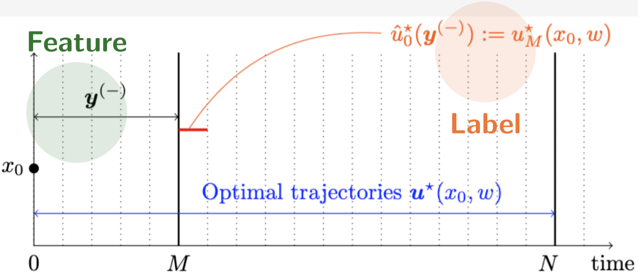
BUILDING THE LEARNING DATA (1)

1. Choose arbitrary pair $q := (x_0, w)$
2. Solve the problem $\mathcal{P}(x_0, w)$ for known (x_0, w)



BUILDING THE LEARNING DATA (1)

1. Choose arbitrary pair $q := (x_0, w)$
2. Solve the problem $\mathcal{P}(x_0, w)$ for known (x_0, w)



NOTA

$\hat{u}_0^\star(\boldsymbol{y}^{(-)})$ is only an approximation of $u_0^\star(\boldsymbol{y}_k^{(-)})$ since the **Bellman** equality

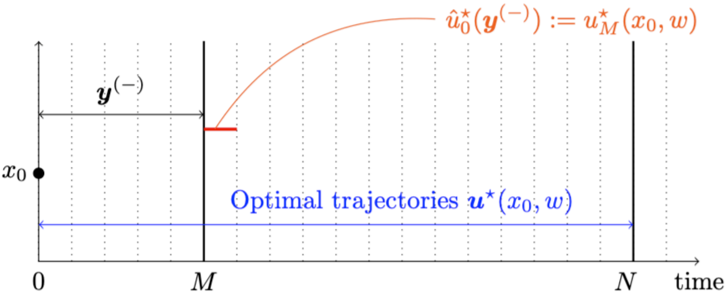
$$u_0^\star(x_M, w) = u_M^\star(x_0, w)$$

holds only for **infinite horizon**.

BUILDING THE LEARNING DATA (2)

Therefore

$$q = (x_0, w) \xrightarrow{\text{Solve } \mathcal{P}(q)} (\boldsymbol{y}^{(-)}(q), u_M^\star(q))$$

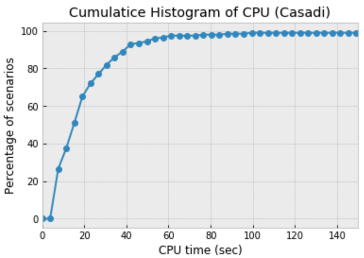


BUILDING THE LEARNING DATA (2)

Therefore

$$q = (x_0, w) \xrightarrow{\text{Solve } \mathcal{P}(q)} (\boldsymbol{y}^{(-)}(q), u_M^\star(q))$$

This is far too expensive to get a single sample in the learning data!



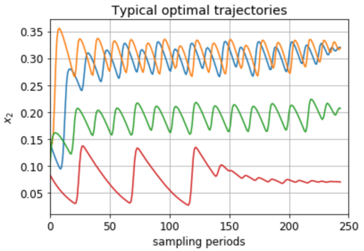
More than 8 days to get a learning data set containing 50000 samples (`Casadi-MS`).

BUILDING THE LEARNING DATA (2)

Therefore

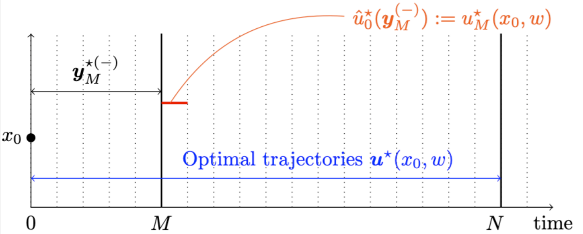
$$q = (x_0, w) \xrightarrow{\text{Solve } \mathcal{P}(q)} (\boldsymbol{y}^{(-)}(q), u_M^\star(q))$$

This is far too expensive to get a single sample in the learning data!



More than 8 days to get a learning data set containing 50000 samples (`Casadi-MS`).

BUILDING THE LEARNING DATA (3)



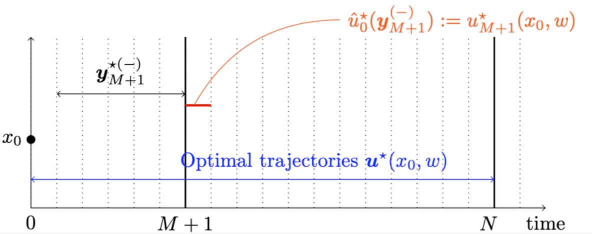
$$\left\{\boldsymbol{y}_{M+i}^{\star(-)}(q), u_{M+i}^\star(q)\right\}_{i=0}^{0}$$

$$\mathcal{D} := \left\{[\boldsymbol{y}_{M+i}^\star(q^{(j)})], u_{M+i}^\star(q^{(j)})\right\}_{(i,j)\in\{0,\dots,m\}\times\{1,\dots,n_r\}}$$

m window's moves + n_r different pairs $q \rightarrow$ card($\mathcal{D}$)$=m \cdot n_r$

$m = 250$, $n_r = 200$ → 50000 samples for the price of 200 NLP solution → 50 minutes!

BUILDING THE LEARNING DATA (3)



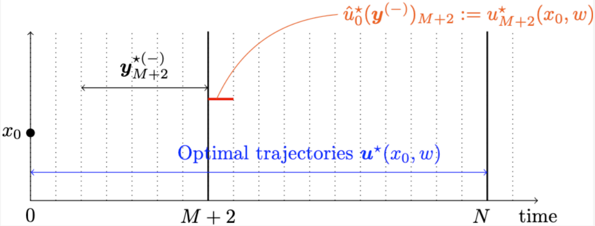
$$\Big\{\boldsymbol{y}_{M+i}^{\star(-)}(q), u_{M+i}^\star(q)\Big\}_{i=0}^{1}$$

$$\mathcal{D} := \Big\{[\boldsymbol{y}_{M+i}^\star(q^{(j)})], u_{M+i}^\star(q^{(j)})\Big\}_{(i,j)\in\{0,\dots,m\}\times\{1,\dots,n_r\}}$$

m window's moves + n_r different pairs q → card($\mathcal{D}$)=$m \cdot n_r$

$m = 250$, $n_r = 200$ → 50000 samples for the price of 200 NLP solution → 50 minutes!

BUILDING THE LEARNING DATA (3)



$$\left\{\boldsymbol{y}^{\star(-)}_{M+i}(q), u^\star_{M+i}(q)\right\}_{i=0}^{2}$$

$$\mathcal{D} := \left\{[\boldsymbol{y}^\star_{M+i}(q^{(j)})], u^\star_{M+i}(q^{(j)})\right\}_{(i,j)\in\{0,\dots,m\}\times\{1,\dots,n_r\}}$$

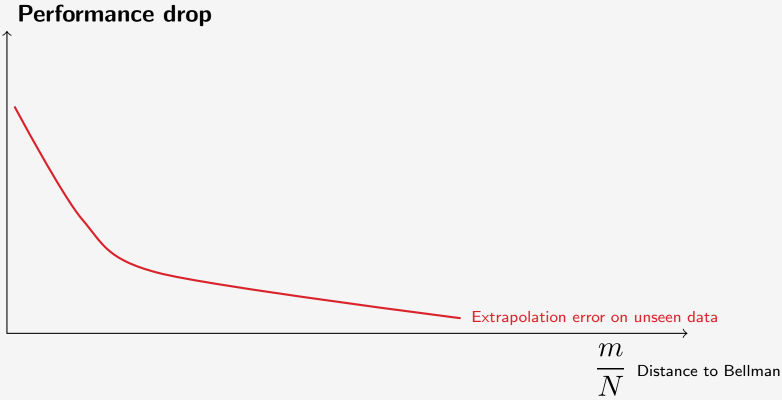
m window's moves + n_r different pairs q → card($\mathcal{D}$)=$m \cdot n_r$

$m = 250$, $n_r = 200$ → 50000 samples for the price of 200 NLP solution → 50 minutes!

BUILDING THE LEARNING DATA (4)

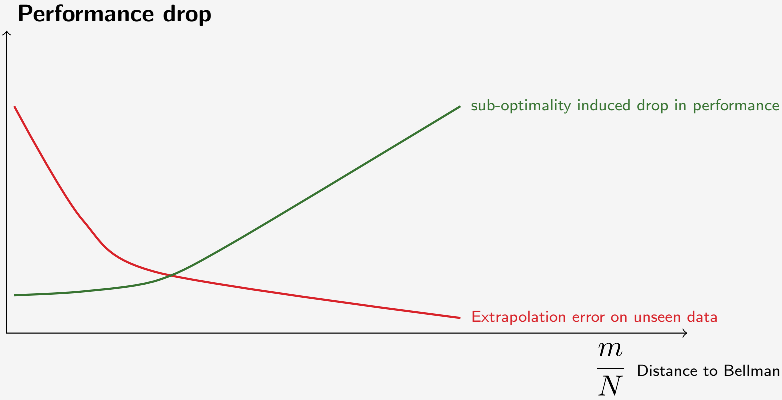
Performance drop for a **Fixed** number of NLP solutions (n_r)

BUILDING THE LEARNING DATA (4)



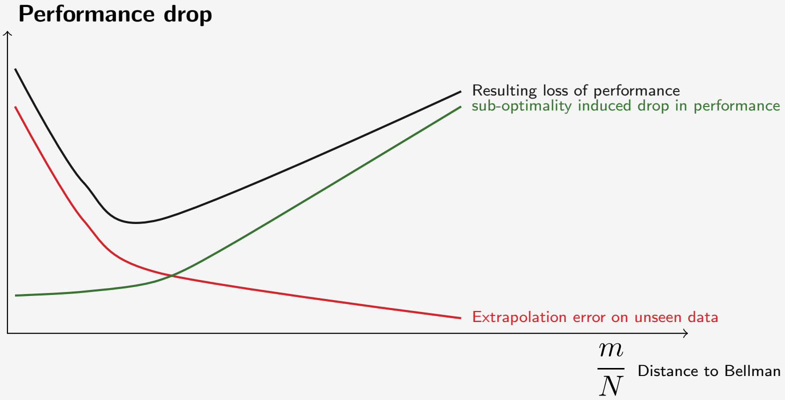
Performance drop for a **Fixed** number of NLP solutions (n_r)

BUILDING THE LEARNING DATA (4)



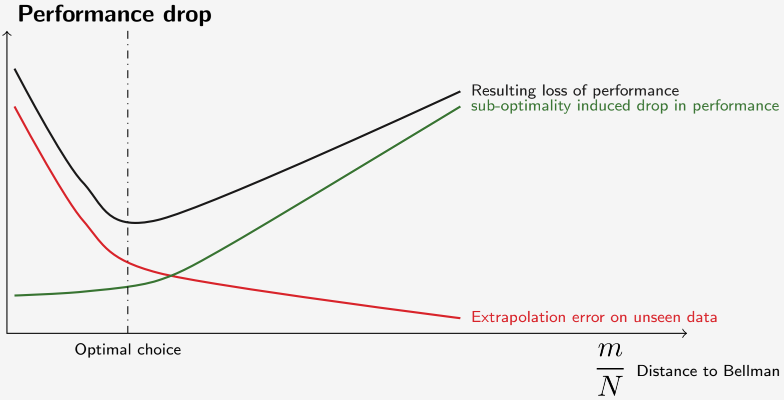
Performance drop for a **Fixed** number of NLP solutions (n_r)

BUILDING THE LEARNING DATA (4)



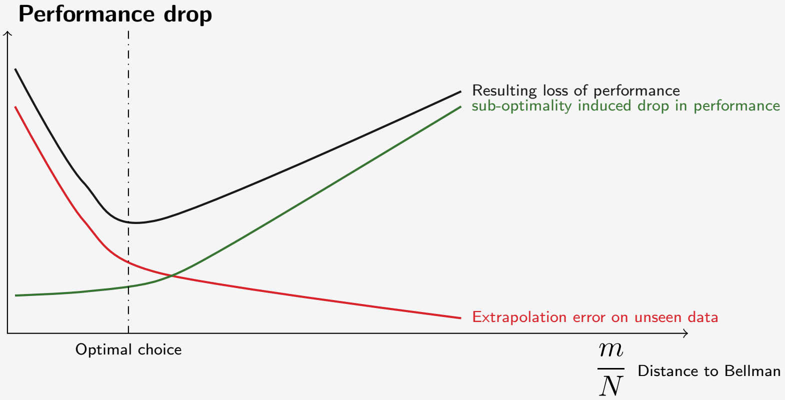
Performance drop for a **Fixed** number of NLP solutions (n_r)

BUILDING THE LEARNING DATA (4)



Performance drop for a **Fixed** number of NLP solutions (n_r)

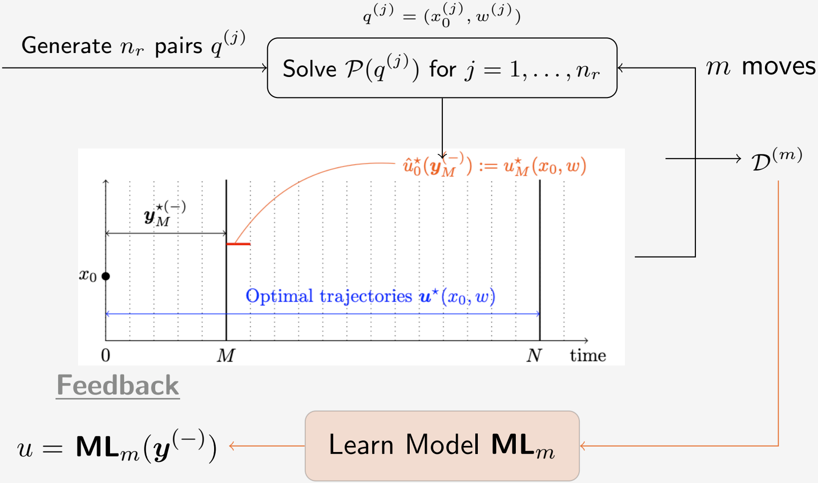
BUILDING THE LEARNING DATA (4)



Performance drop for a **Fixed** number of NLP solutions (n_r)

The optimal choice is problem-dependent.

THE WHOLE FLOW IN A GLANCE



$q^{(j)} = (x_0^{(j)}, w^{(j)})$

Generate n_r pairs $q^{(j)}$

Solve $\mathcal{P}(q^{(j)})$ for $j = 1, \dots, n_r$

m moves

$\mathcal{D}^{(m)}$

Feedback

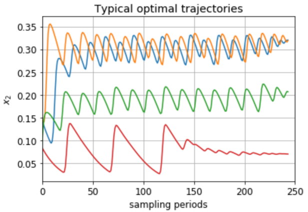
$u = \mathbf{ML}_m(\boldsymbol{y}^{(-)})$

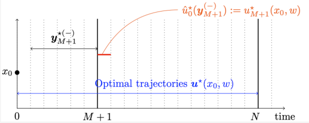
Learn Model $\mathbf{ML}_m$

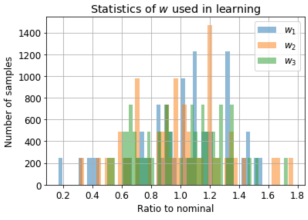
APPLICATION TO THE PARALLEL REACTOR EXAMPLE

Parameters of the method

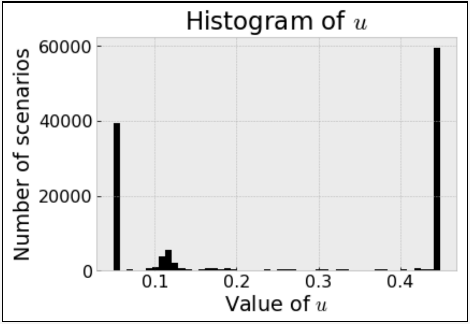
- $N = 250$
- $M = 10$
- $m \in \{10, 50, 100, 150, 200\}$
- $w^0 = (10^4, 400, 0.55)$
- $w_i = (1 + \nu_i)w_i^0$
- $\nu_i \in \mathcal{N}(0, \sigma_i)$ with $\sigma_i = 0.33$
- $\mathbb{X}_0 := [10^{-4}, 0.5] \times [10^{-4}, 0.2] \times [10^{-4}, 0.25]$
- x_0 sampled uniformly in $\mathbb{X}_0$
- $n_r = 500$


Typical optimal trajectories




Statistics of w used in learning

EXAMINATION OF THE CLOUD OF CONTROL

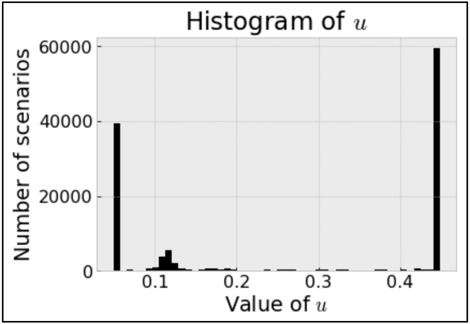


Histogram of the encountered $u^\star_{M+i}(x_0^{(j)}, w^{(j)})$

→ `RandomForestClassifier` - `sklearn`

EXAMINATION OF THE CLOUD OF CONTROL

$$\text{label} = \begin{cases} 1 & \text{if } u \leq 0.075 \\ 2 & \text{if } u \in]0.14] \\ 3 & \text{otherwise} \end{cases}$$



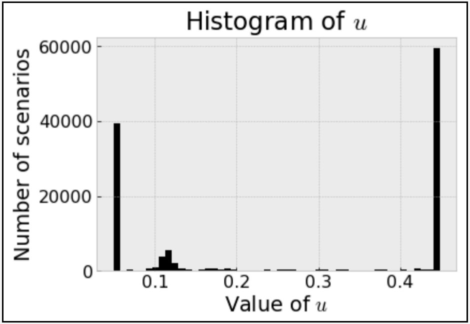
Histogram of the encountered $u^\star_{M+i}(x_0^{(j)}, w^{(j)})$

→ `RandomForestClassifier` - `sklearn`

EXAMINATION OF THE CLOUD OF CONTROL

$$\text{label} = \begin{cases} 1 & \text{if } u \le 0.075 \\ 2 & \text{if } u \in]0.14] \\ 3 & \text{otherwise} \end{cases}$$

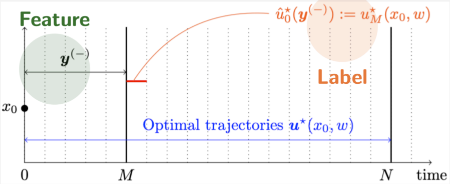
Based on this observation, a classification approach is adopted for the definition of the **ML**-based feedback



Histogram of the encountered $u^\star_{M+i}(x_0^{(j)}, w^{(j)})$

→ `RandomForestClassifier - sklearn`

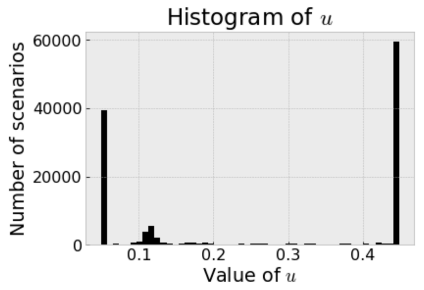
The choice of $\boldsymbol{y}^{(-)}$



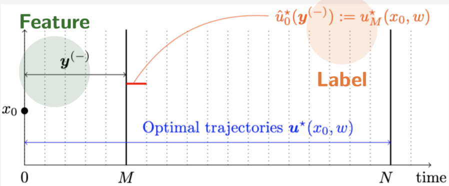
$$\dot{x}_1 = 1 - w_1 x_1^2 e^{-1/x_3} - w_2 x_1 e^{-w_3/x_3} - x_1$$

$$\dot{x}_2 = w_1 x_1^2 e^{-1/x_3} - x_2$$

$$\dot{x}_3 = u - x_3$$



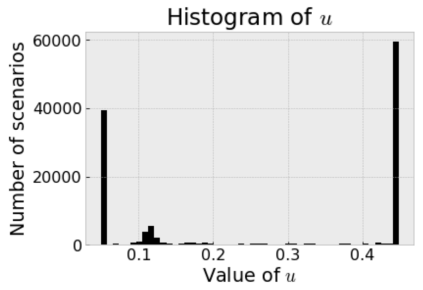
The choice of $\boldsymbol{y}^{(-)}$



$$\dot{x}_1 = 1 - w_1 x_1^2 e^{-1/x_3} - w_2 x_1 e^{-w_3/x_3} - x_1$$

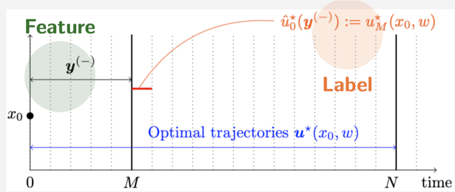
$$\dot{x}_2 = w_1 x_1^2 e^{-1/x_3} - x_2$$

$$\dot{x}_3 = u - x_3$$



If the control is included in $\boldsymbol{y}^{(-)}$, then **ML** algorithms would prefer to define the label to be equal to the most recent control as this would correspond to a very high precision.

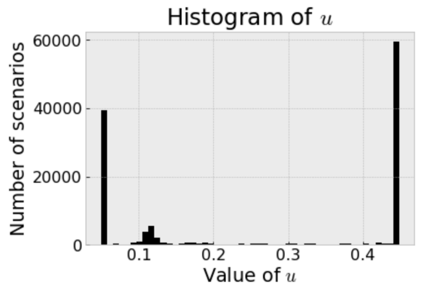
The choice of $\boldsymbol{y}^{(-)}$



$$\dot{x}_1 = 1 - w_1 x_1^2 e^{-1/x_3} - w_2 x_1 e^{-w_3/x_3} - x_1$$

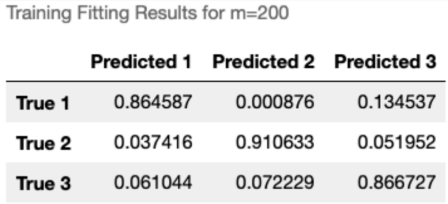
$$\dot{x}_2 = w_1 x_1^2 e^{-1/x_3} - x_2$$

$$\dot{x}_3 = u - x_3$$



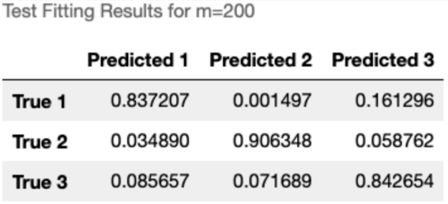
By taking $M = 10$, the conjecture is that, for this specific problem, the use of the measurement profile of $\boldsymbol{y} := x_2$ would be enough to implicitly reveal the value of the state and the parameters and makes the whole framework successful.

LEARNING RESULTS INDICATORS



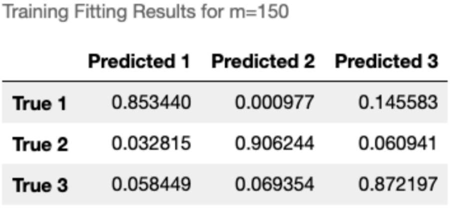
Training Fitting Results for m=200

	Predicted 1	Predicted 2	Predicted 3
True 1	0.864587	0.000876	0.134537
True 2	0.037416	0.910633	0.051952
True 3	0.061044	0.072229	0.866727



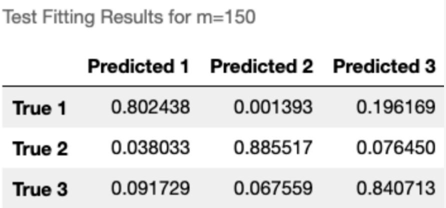
Test Fitting Results for m=200

	Predicted 1	Predicted 2	Predicted 3
True 1	0.837207	0.001497	0.161296
True 2	0.034890	0.906348	0.058762
True 3	0.085657	0.071689	0.842654



Training Fitting Results for m=150

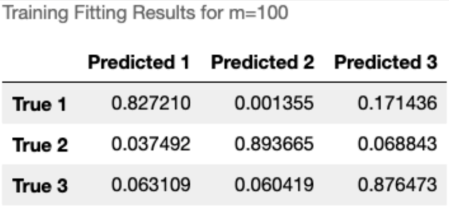
	Predicted 1	Predicted 2	Predicted 3
True 1	0.853440	0.000977	0.145583
True 2	0.032815	0.906244	0.060941
True 3	0.058449	0.069354	0.872197



Test Fitting Results for m=150

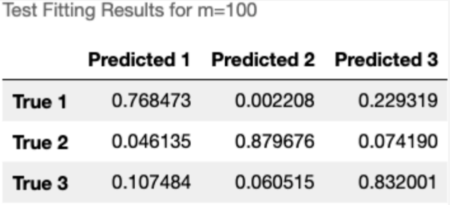
	Predicted 1	Predicted 2	Predicted 3
True 1	0.802438	0.001393	0.196169
True 2	0.038033	0.885517	0.076450
True 3	0.091729	0.067559	0.840713

LEARNING RESULTS INDICATORS



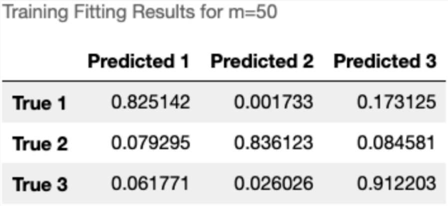
Training Fitting Results for m=100

	Predicted 1	Predicted 2	Predicted 3
True 1	0.827210	0.001355	0.171436
True 2	0.037492	0.893665	0.068843
True 3	0.063109	0.060419	0.876473



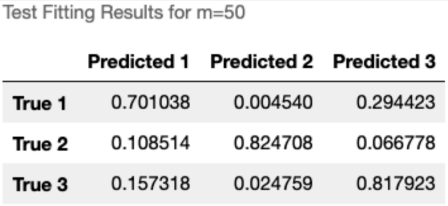
Test Fitting Results for m=100

	Predicted 1	Predicted 2	Predicted 3
True 1	0.768473	0.002208	0.229319
True 2	0.046135	0.879676	0.074190
True 3	0.107484	0.060515	0.832001



Training Fitting Results for m=50

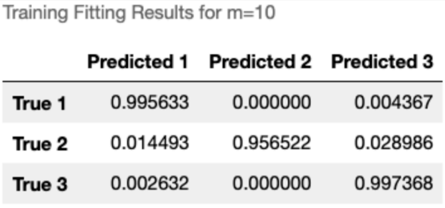
	Predicted 1	Predicted 2	Predicted 3
True 1	0.825142	0.001733	0.173125
True 2	0.079295	0.836123	0.084581
True 3	0.061771	0.026026	0.912203



Test Fitting Results for m=50

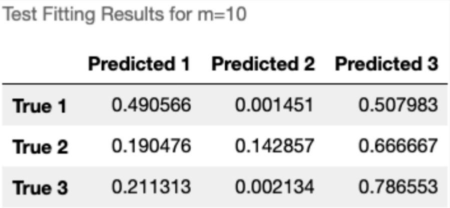
	Predicted 1	Predicted 2	Predicted 3
True 1	0.701038	0.004540	0.294423
True 2	0.108514	0.824708	0.066778
True 3	0.157318	0.024759	0.817923

LEARNING RESULTS INDICATORS



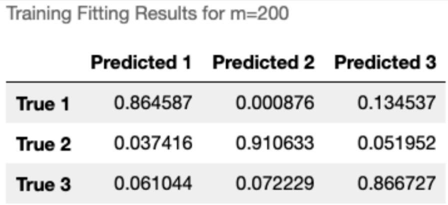
Training Fitting Results for m=10

	Predicted 1	Predicted 2	Predicted 3
True 1	0.995633	0.000000	0.004367
True 2	0.014493	0.956522	0.028986
True 3	0.002632	0.000000	0.997368



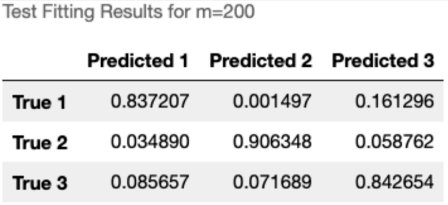
Test Fitting Results for m=10

	Predicted 1	Predicted 2	Predicted 3
True 1	0.490566	0.001451	0.507983
True 2	0.190476	0.142857	0.666667
True 3	0.211313	0.002134	0.786553



Training Fitting Results for m=200

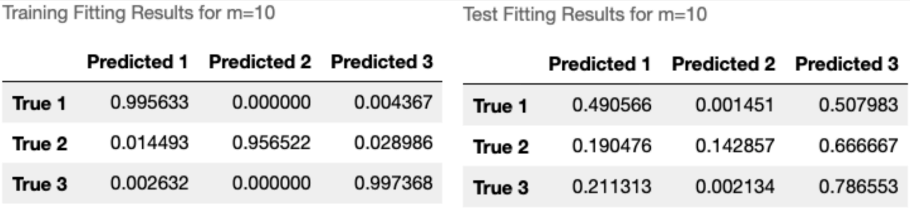
	Predicted 1	Predicted 2	Predicted 3
True 1	0.864587	0.000876	0.134537
True 2	0.037416	0.910633	0.051952
True 3	0.061044	0.072229	0.866727



Test Fitting Results for m=200

	Predicted 1	Predicted 2	Predicted 3
True 1	0.837207	0.001497	0.161296
True 2	0.034890	0.906348	0.058762
True 3	0.085657	0.071689	0.842654

LEARNING RESULTS INDICATORS

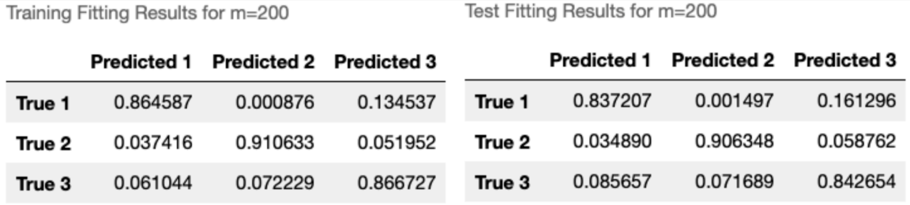


Training Fitting Results for m=10

	Predicted 1	Predicted 2	Predicted 3
True 1	0.995633	0.000000	0.004367
True 2	0.014493	0.956522	0.028986
True 3	0.002632	0.000000	0.997368

Test Fitting Results for m=10

	Predicted 1	Predicted 2	Predicted 3
True 1	0.490566	0.001451	0.507983
True 2	0.190476	0.142857	0.666667
True 3	0.211313	0.002134	0.786553



Training Fitting Results for m=200

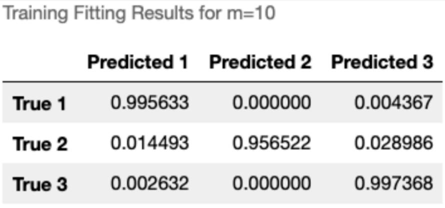
	Predicted 1	Predicted 2	Predicted 3
True 1	0.864587	0.000876	0.134537
True 2	0.037416	0.910633	0.051952
True 3	0.061044	0.072229	0.866727

Test Fitting Results for m=200

	Predicted 1	Predicted 2	Predicted 3
True 1	0.837207	0.001497	0.161296
True 2	0.034890	0.906348	0.058762
True 3	0.085657	0.071689	0.842654

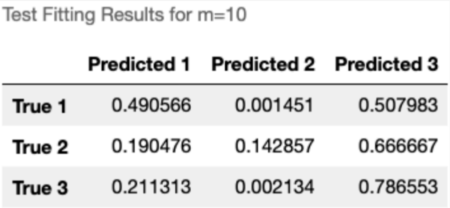
When test results precision is significantly worse than training precision, this indicate **over-fitting** → bad extrapolation capabilities.

LEARNING RESULTS INDICATORS



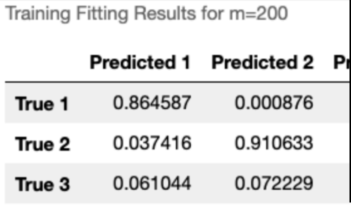
Training Fitting Results for m=10

	Predicted 1	Predicted 2	Predicted 3
True 1	0.995633	0.000000	0.004367
True 2	0.014493	0.956522	0.028986
True 3	0.002632	0.000000	0.997368



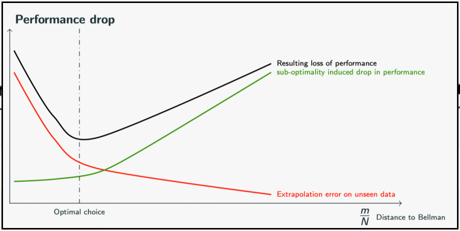
Test Fitting Results for m=10

	Predicted 1	Predicted 2	Predicted 3
True 1	0.490566	0.001451	0.507983
True 2	0.190476	0.142857	0.666667
True 3	0.211313	0.002134	0.786553

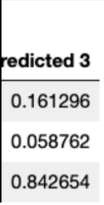


Training Fitting Results for m=200

	Predicted 1	Predicted 2	Pr
True 1	0.864587	0.000876	
True 2	0.037416	0.910633	
True 3	0.061044	0.072229	



Performance drop



redicted 3
0.161296
0.058762
0.842654

When test results precision is significantly worse than training precision, this indicate **over-fitting** → bad extrapolation capabilities.

Validation indicators

Using a new set of 200 pairs of (x_0, w), closed-loop is simulated over N sampling period. shall compare the following performance indices:

Casadi Optimal

Since the closed-loop simulations last also N sampling periods (The same used for the computation of the open-loop optimal control), the `casadi` cost is the optimal one since it perfectly knows the parameter values.

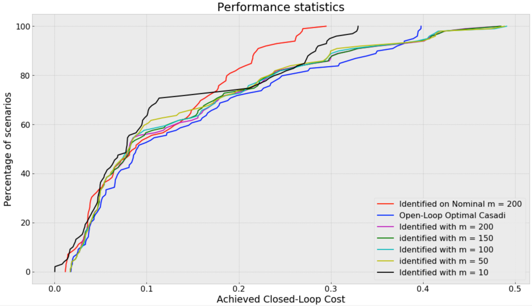
Nominally Optimal

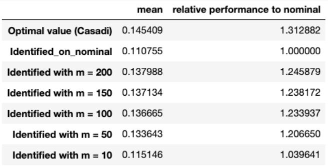
This is the closed loop using learned output feedback from the validation data in which the nominal parameter is used. (Only the initial state is taken from the sampled data).

Proposed for $\neq m$

This is the closed loop using learned output feedback from the validation data $\mathcal{D}_m$ obtained by m moves of the window for each x_0 in the learning data.

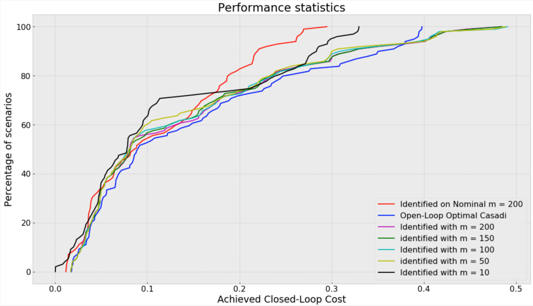
RESULTS

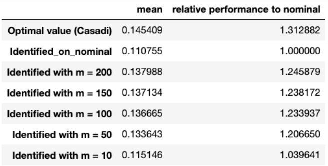




	mean	relative performance to nominal
Optimal value (Casadi)	0.145409	1.312882
Identified_on_nominal	0.110755	1.000000
Identified with m = 200	0.137988	1.245879
Identified with m = 150	0.137134	1.238172
Identified with m = 100	0.136665	1.233937
Identified with m = 50	0.133643	1.206650
Identified with m = 10	0.115146	1.039641

RESULTS

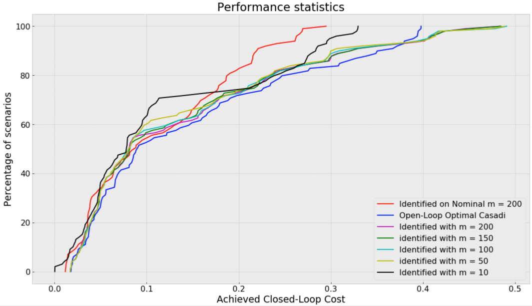


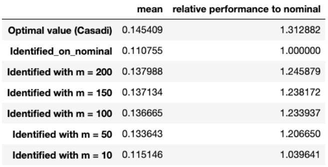


	mean	relative performance to nominal
Optimal value (Casadi)	0.145409	1.312882
Identified_on_nominal	0.110755	1.000000
Identified with m = 200	0.137988	1.245879
Identified with m = 150	0.137134	1.238172
Identified with m = 100	0.136665	1.233937
Identified with m = 50	0.133643	1.206650
Identified with m = 10	0.115146	1.039641

The proposed method enables to recover 78% of the advantage of knowing perfectly the parameters values when $m \in \{200, 150, 100\}$.

Results



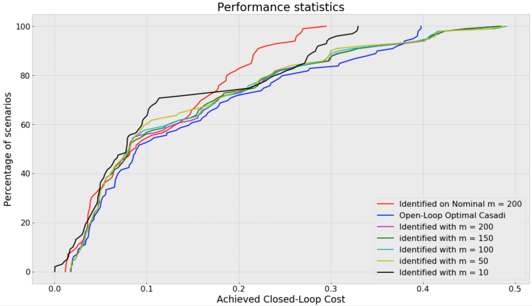


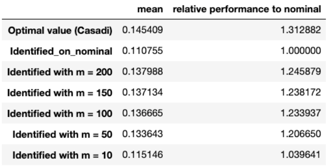
	mean	relative performance to nominal
Optimal value (Casadi)	0.145409	1.312882
Identified_on_nominal	0.110755	1.000000
Identified with m = 200	0.137988	1.245879
Identified with m = 150	0.137134	1.238172
Identified with m = 100	0.136665	1.233937
Identified with m = 50	0.133643	1.206650
Identified with m = 10	0.115146	1.039641

The proposed method enables to recover 78% of the advantage of knowing perfectly the parameters values when $m \in \{200, 150, 100\}$.

For smaller values of m, data is no more sufficiently representative.

RESULTS





	mean	relative performance to nominal
Optimal value (Casadi)	0.145409	1.312882
Identified_on_nominal	0.110755	1.000000
Identified with m = 200	0.137988	1.245879
Identified with m = 150	0.137134	1.238172
Identified with m = 100	0.136665	1.233937
Identified with m = 50	0.133643	1.206650
Identified with m = 10	0.115146	1.039641

The proposed method enables to recover 78% of the advantage of knowing perfectly the parameters values when $m \in \{200, 150, 100\}$.

For smaller values of m, data is no more sufficiently representative.

The identified output feedback outperform the `casadi` output (non achieved optimality) in 10% of the scenarios.

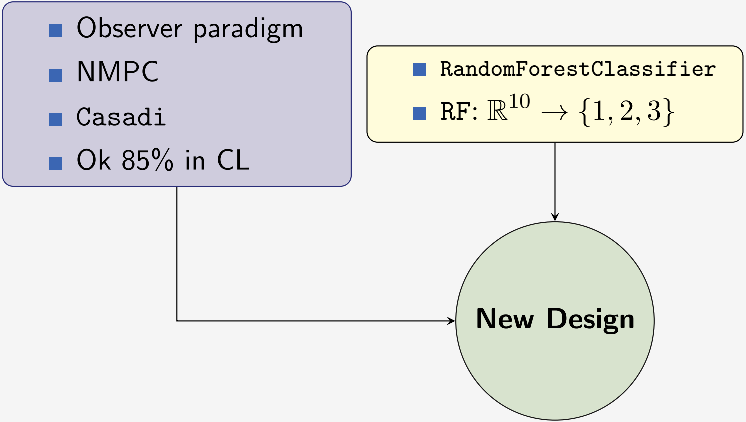
Conclusion

- A heuristic!
- Special care is needed regarding the choice of $\boldsymbol{y}^{(-)}$

- No need for explicit extended observer.
- Close-to-optimal dynamic output feedback.
- Totally scalable (No exponential explosion with state or parameter dimension).

- Unthinkable without the previous development of NLP (more or less) reliable solvers

TAKE HOME MESSAGE



- Observer paradigm
- NMPC
- `Casadi`
- Ok 85% in CL

- `RandomForestClassifier`
- RF: $\mathbb{R}^{10} \rightarrow \{1, 2, 3\}$

New Design

New Design that would be unconceivable without the mix of two cultures!

1. Data-driven partial observability assessment under uncertainties
2. Learning-Based Approximate Stochastic NMPC Design
3. Tractable stochastic NMPC by supervised clustering
4. Data-Driven NMPC by cost function identification
5. Learning-Based Monitoring updating period in Real-time NMPC

STOCHASTIC MPC USING SUPERVISED CLUSTERING

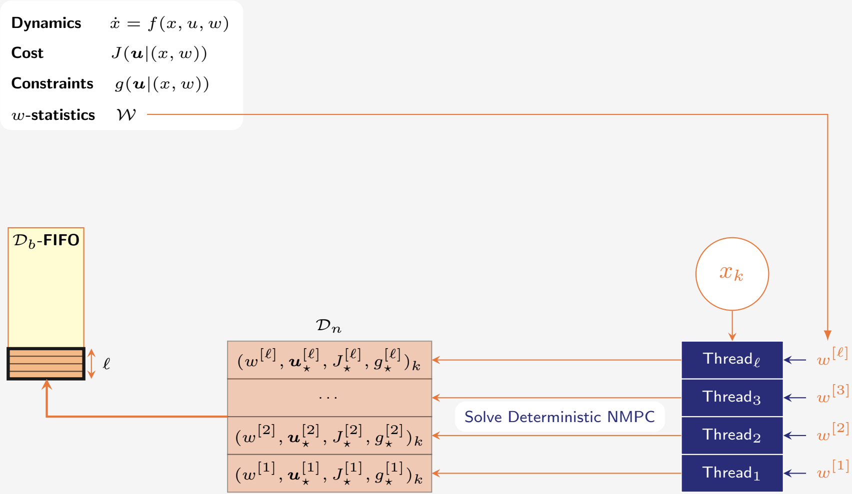
Distributing the data building over the real-life time

Dynamics	$\dot{x} = f(x, u, w)$
Cost	$J(\boldsymbol{u}\|(x, w))$
Constraints	$g(\boldsymbol{u}\|(x, w))$
w-**statistics**	$\mathcal{W}$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time

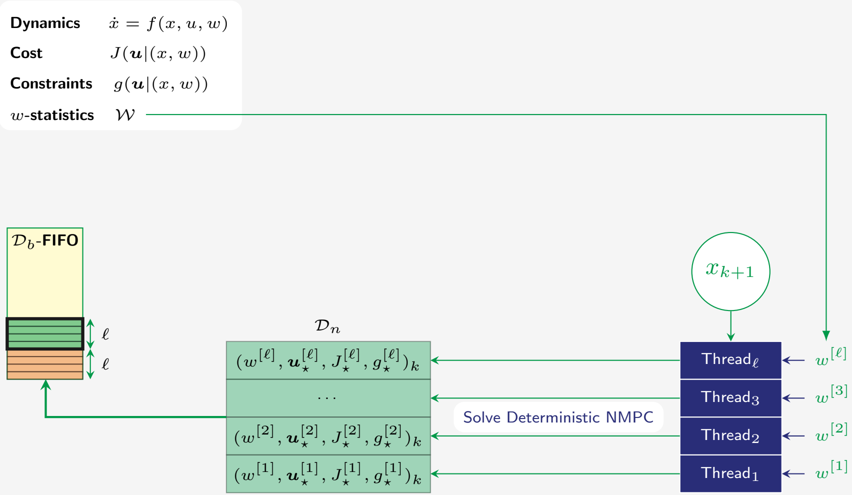


Dynamics	$\dot{x} = f(x, u, w)$
Cost	$J(\boldsymbol{u}\|(x, w))$
Constraints	$g(\boldsymbol{u}\|(x, w))$
w**-statistics**	$\mathcal{W}$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time



Dynamics	$\dot{x} = f(x, u, w)$
Cost	$J(\boldsymbol{u}\|(x, w))$
Constraints	$g(\boldsymbol{u}\|(x, w))$
w**-statistics**	$\mathcal{W}$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

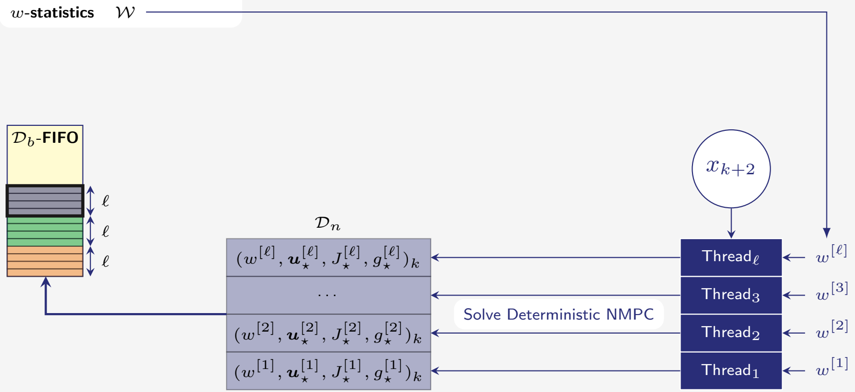
Distributing the data building over the real-life time

Dynamics $\dot{x} = f(x,u,w)$

Cost $J(\boldsymbol{u}|(x,w))$

Constraints $g(\boldsymbol{u}|(x,w))$

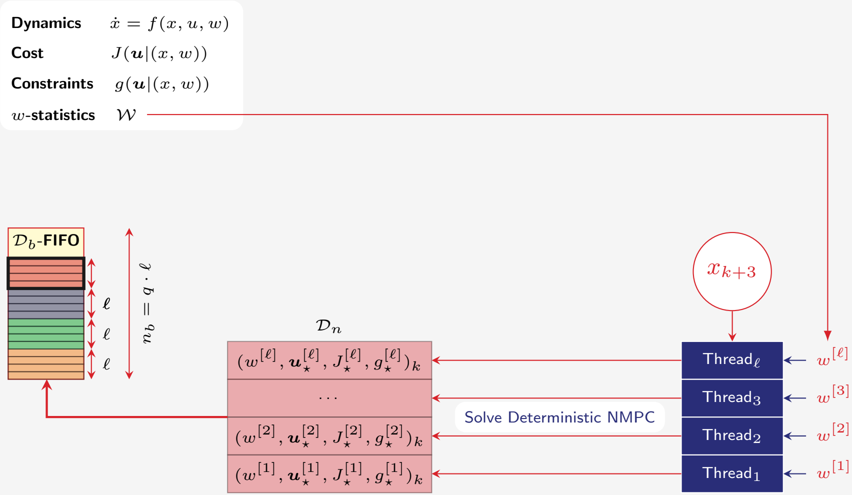
w-**statistics** $\mathcal{W}$



M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time



Dynamics $\dot{x} = f(x, u, w)$

Cost $J(\boldsymbol{u}|(x, w))$

Constraints $g(\boldsymbol{u}|(x, w))$

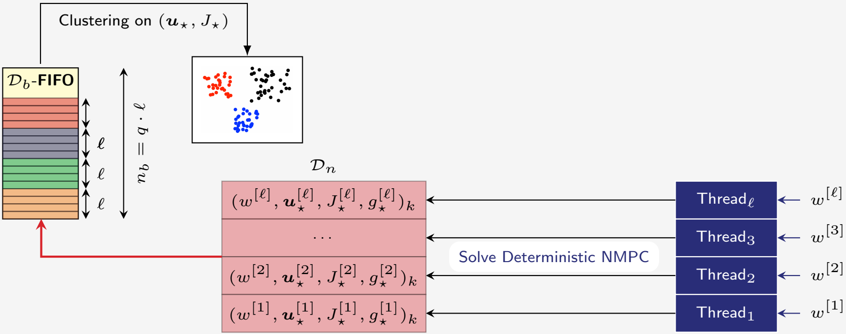
w-statistics $\mathcal{W}$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time

Dynamics	$\dot{x} = f(x, u, w)$
Cost	$J(\boldsymbol{u}\vert(x, w))$
Constraints	$g(\boldsymbol{u}\vert(x, w))$
w-**statistics**	$\mathcal{W}$

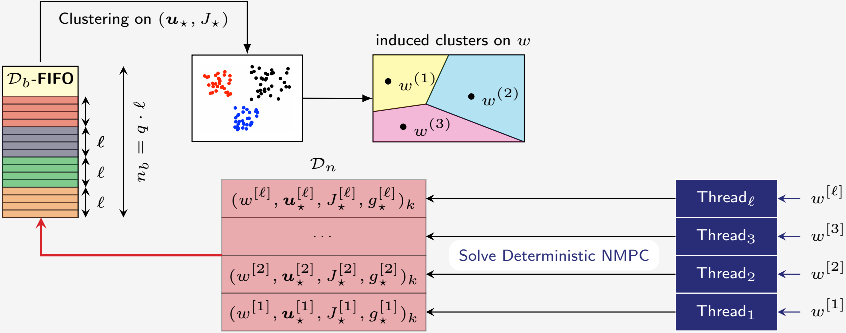


M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time

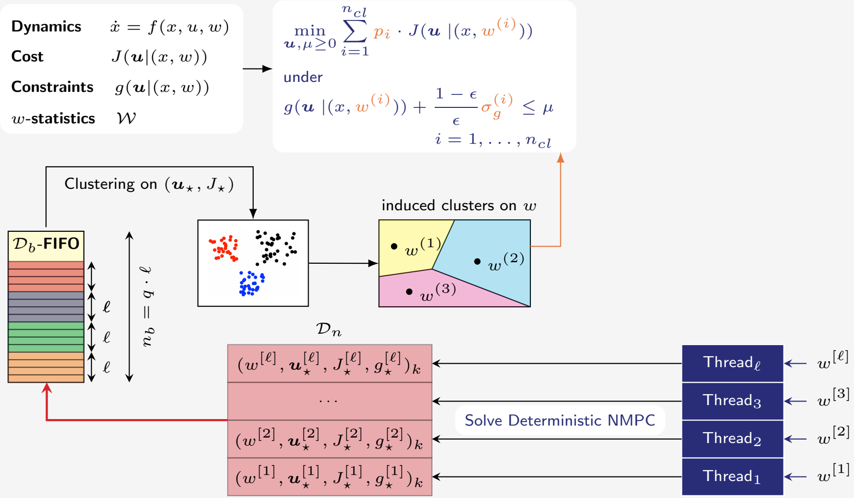
Dynamics	$\dot{x} = f(x, u, w)$
Cost	$J(\boldsymbol{u}\|(x, w))$
Constraints	$g(\boldsymbol{u}\|(x, w))$
w-**statistics**	$\mathcal{W}$



M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time



Dynamics $\dot{x} = f(x,u,w)$

Cost $J(\boldsymbol{u}|(x,w))$

Constraints $g(\boldsymbol{u}|(x,w))$

w-statistics $\mathcal{W}$

$$\min_{\boldsymbol{u},\mu\geq 0} \sum_{i=1}^{n_{cl}} p_i \cdot J(\boldsymbol{u}\;|(x,w^{(i)}))$$

under

$$g(\boldsymbol{u}\;|(x,w^{(i)})) + \frac{1-\epsilon}{\epsilon}\sigma_g^{(i)} \leq \mu$$

$$i = 1,\dots,n_{cl}$$

Clustering on $(\boldsymbol{u}_\star, J_\star)$

induced clusters on w

$w^{(1)}$

$w^{(2)}$

$w^{(3)}$

$\mathcal{D}_b$-**FIFO**

ℓ

ℓ

ℓ

$n_b = q\cdot\ell$

$\mathcal{D}_n$

$(w^{[\ell]},\boldsymbol{u}_\star^{[\ell]},J_\star^{[\ell]},g_\star^{[\ell]})_k$

$\dots$

$(w^{[2]},\boldsymbol{u}_\star^{[2]},J_\star^{[2]},g_\star^{[2]})_k$

$(w^{[1]},\boldsymbol{u}_\star^{[1]},J_\star^{[1]},g_\star^{[1]})_k$

Solve Deterministic NMPC

Thread$_\ell$ $\leftarrow w^{[\ell]}$

Thread$_3$ $\leftarrow w^{[3]}$

Thread$_2$ $\leftarrow w^{[2]}$

Thread$_1$ $\leftarrow w^{[1]}$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time

Dynamics $\dot{x} = f(x,u,w)$

Cost $J(\boldsymbol{u}|(x,w))$

Constraints $g(\boldsymbol{u}|(x,w))$

w**-statistics** $\mathcal{W}$

$$\min_{\boldsymbol{u},\mu\geq 0}\sum_{i=1}^{n_{cl}} p_i\cdot J(\boldsymbol{u}\,|(x,w^{(i)}))$$

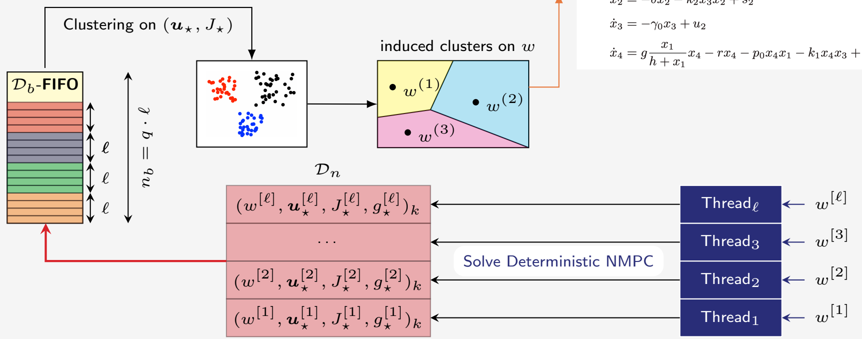
under

$$g(\boldsymbol{u}\,|(x,w^{(i)}))+\frac{1-\epsilon}{\epsilon}\sigma_g^{(i)}\leq\mu$$
$$i=1,\dots,n_{cl}$$

x_1 tumor cell population;
x_2 circulating lymphocytes population;
x_3 chemotherapy drug concentration;
x_4 effector immune cell population;
u_1 rate of introduction of immunotherapy drug;
u_2 rate of introduction of chemotherapy drug.

and the dynamics is given by

$$\begin{aligned}\dot{x}_1 &= ax_1(1-bx_1)-c_1x_4x_1-k_3x_3x_1\\ \dot{x}_2 &= -\delta x_2-k_2x_3x_2+s_2\\ \dot{x}_3 &= -\gamma_0x_3+u_2\\ \dot{x}_4 &= g\frac{x_1}{h+x_1}x_4-rx_4-p_0x_4x_1-k_1x_4x_3+s_1u_1.\end{aligned}$$



Clustering on $(\boldsymbol{u}_\star, J_\star)$

$\mathcal{D}_b$-**FIFO**

ℓ

ℓ

ℓ

$n_b = q\cdot\ell$

induced clusters on w

$w^{(1)}$

$w^{(2)}$

$w^{(3)}$

$\mathcal{D}_n$

$(w^{[\ell]},\boldsymbol{u}_\star^{[\ell]},J_\star^{[\ell]},g_\star^{[\ell]})_k$

$\dots$

$(w^{[2]},\boldsymbol{u}_\star^{[2]},J_\star^{[2]},g_\star^{[2]})_k$

$(w^{[1]},\boldsymbol{u}_\star^{[1]},J_\star^{[1]},g_\star^{[1]})_k$

Solve Deterministic NMPC

Thread$_\ell$ ← $w^{[\ell]}$

Thread$_3$ ← $w^{[3]}$

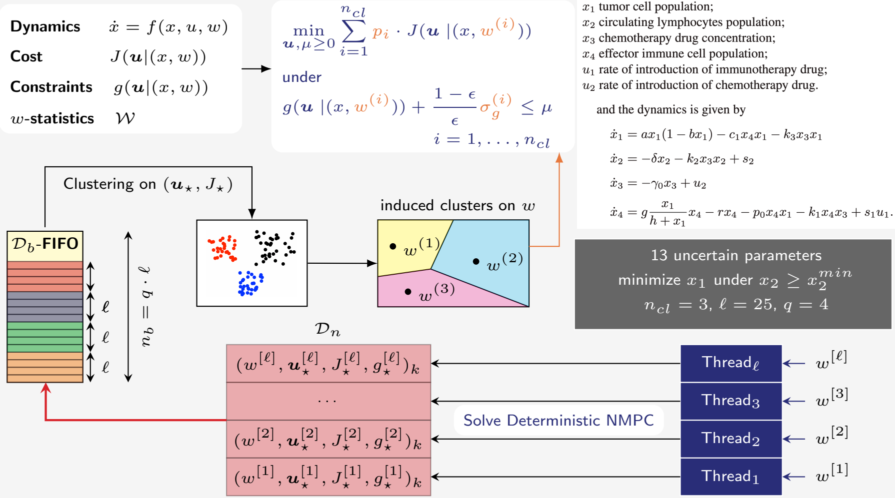
Thread$_2$ ← $w^{[2]}$

Thread$_1$ ← $w^{[1]}$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time



Dynamics $\dot{x} = f(x,u,w)$

Cost $J(\boldsymbol{u}|(x,w))$

Constraints $g(\boldsymbol{u}|(x,w))$

w**-statistics** $\mathcal{W}$

$$\min_{\boldsymbol{u},\mu\geq 0} \sum_{i=1}^{n_{cl}} p_i \cdot J(\boldsymbol{u}\;|(x,w^{(i)}))$$

under

$$g(\boldsymbol{u}\;|(x,w^{(i)})) + \frac{1-\epsilon}{\epsilon}\sigma_g^{(i)} \leq \mu$$
$$i = 1,\dots,n_{cl}$$

x_1 tumor cell population;
x_2 circulating lymphocytes population;
x_3 chemotherapy drug concentration;
x_4 effector immune cell population;
u_1 rate of introduction of immunotherapy drug;
u_2 rate of introduction of chemotherapy drug.

and the dynamics is given by

$$\begin{aligned}
\dot{x}_1 &= ax_1(1-bx_1) - c_1x_4x_1 - k_3x_3x_1\\
\dot{x}_2 &= -\delta x_2 - k_2x_3x_2 + s_2\\
\dot{x}_3 &= -\gamma_0x_3 + u_2\\
\dot{x}_4 &= g\frac{x_1}{h+x_1}x_4 - rx_4 - p_0x_4x_1 - k_1x_4x_3 + s_1u_1.
\end{aligned}$$

13 uncertain parameters
minimize x_1 under $x_2 \geq x_2^{min}$
$n_{cl} = 3$, $\ell = 25$, $q = 4$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time

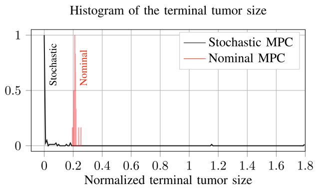
Dynamics	$\dot{x} = f(x, u, w)$
Cost	$J(\boldsymbol{u} \vert (x, w))$
Constraints	$g(\boldsymbol{u} \vert (x, w))$
w**-statistics**	$\mathcal{W}$

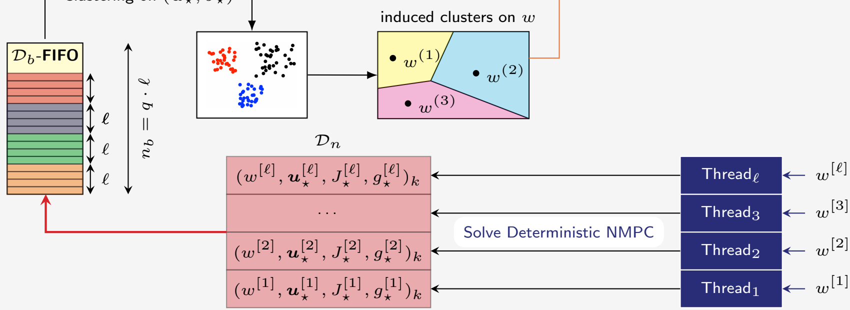
$$\min_{\boldsymbol{u}, \mu \geq 0} \sum_{i=1}^{n_{cl}} p_i \cdot J(\boldsymbol{u} \;\vert (x, w^{(i)}))$$

under

$$g(\boldsymbol{u} \;\vert (x, w^{(i)})) + \frac{1-\epsilon}{\epsilon} \sigma_g^{(i)} \leq \mu$$

$$i = 1, \dots, n_{cl}$$





M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

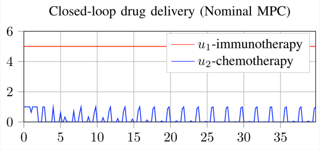
Distributing the data building over the real-life time

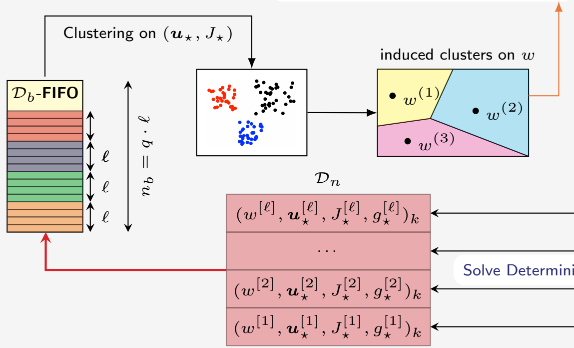
Dynamics	$\dot{x} = f(x, u, w)$
Cost	$J(\boldsymbol{u}\|(x, w))$
Constraints	$g(\boldsymbol{u}\|(x, w))$
w**-statistics**	$\mathcal{W}$

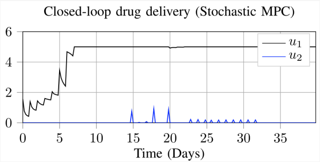
$$\min_{\boldsymbol{u},\mu\geq 0} \sum_{i=1}^{n_{cl}} p_i \cdot J(\boldsymbol{u}\ \vert(x, w^{(i)}))$$

under

$$g(\boldsymbol{u}\ \vert(x, w^{(i)})) + \frac{1-\epsilon}{\epsilon}\sigma_g^{(i)} \leq \mu \qquad i = 1, \dots, n_{cl}$$







Solve Deterministic NMPC

Thread$_\ell$ ← $w^{[\ell]}$

Thread$_3$ ← $w^{[3]}$

Thread$_2$ ← $w^{[2]}$

Thread$_1$ ← $w^{[1]}$

M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

STOCHASTIC MPC USING SUPERVISED CLUSTERING

Distributing the data building over the real-life time

Dynamics $\dot{x} = f(x, u, w)$

Cost $J(\boldsymbol{u}|(x, w))$

Constraints $g(\boldsymbol{u}|(x, w))$

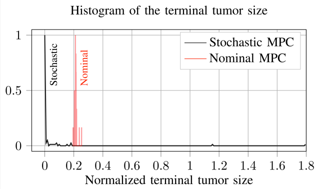
w-statistics $\mathcal{W}$

$$\min_{\boldsymbol{u},\mu\geq 0} \sum_{i=1}^{n_{cl}} p_i \cdot J(\boldsymbol{u}\;|(x, w^{(i)}))$$

under

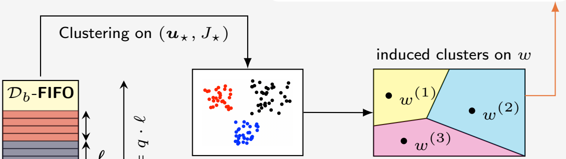
$$g(\boldsymbol{u}\;|(x, w^{(i)})) + \frac{1-\epsilon}{\epsilon}\sigma_g^{(i)} \leq \mu$$

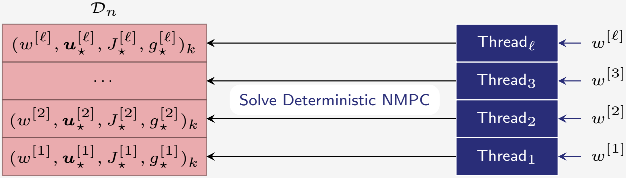
$$i = 1, \dots, n_{cl}$$



STATISTICS OF THE COMPUTATION TIMES (IN MS) OF A SINGLE SOLUTION OF THE OPTIMIZATION PROBLEM. (CASADI [4] (PYTHON VERSION) ON A MACBOOKPRO 2.9 GHZ INTEL CORE I7)

	Mean (ms)	**Standard deviation** (ms)
Nominal	270	76
Stochastic	383	330





M.A On the use of supervised clustering in stochastic NMPC design. IEEE-TAC. Volume 65, Issue 12, 2020.

1. Data-driven partial observability assessment under uncertainties
2. Learning-Based Approximate Stochastic NMPC Design
3. Tractable stochastic NMPC by supervised clustering
4. Data-Driven NMPC by cost function identification
5. Learning-Based Monitoring updating period in Real-time NMPC

Possible use of ML in MPC design

Direct single shooting formulation of MPC:

$$\boldsymbol{u}_k^\star \leftarrow \min_{\boldsymbol{u}_k \in \mathbb{U}^N} \Big[J(\boldsymbol{u}_k \mid x_k) \Big] \qquad \boldsymbol{u}_k := \begin{bmatrix} u_k \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \left[\mathbb{R}^{n_u}\right]^N$$

MPC Feedback Control

$$K_{\text{MPC}}(x_k) = u_k^\star$$

Standard settings

$$J(\boldsymbol{u}_k \mid x_k) := \sum_{i=1}^{N} \ell(x_{k+i}, u_{k+i-1})$$

$$x_{k+i+1} = f(x_{k+i}, u_{k+i})$$

Possible use of ML in MPC design

Option 1 **Learn optimal solution $u_k^\star$**

- Known Dynamics
- NMPC solvers (ACADO, GURBOI, CASADI)
- Solve many problems off-line
- Use ML to fit a model with
    - **features**: x_k
    - **Label**: $u_k^\star$

Possible drawbacks:

(-) Needs the dynamical model

(-) Expensive computation per samples

Direct single shooting formulation of MPC:

$$u_k^* \leftarrow \min_{\mathbf{u}_k \in \mathbb{U}^N} \Big[J(\mathbf{u}_k \mid x_k)\Big] \qquad \mathbf{u}_k := \begin{bmatrix} u_k \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \left[\mathbb{R}^{n_u}\right]^N$$

MPC Feedback Control

$$K_{\text{MPC}}(x_k) = u_k^*$$

Standard settings

$$J(\mathbf{u}_k \mid x_k) := \sum_{i=1}^{N} \ell(x_{k+i}, u_{k+i-1})$$

$$x_{k+i+1} = f(x_{k+i}, u_{k+i})$$

Possible use of ML in MPC design

Option 2 **ML to identify dynamics + on-line solutions**

- **Unknown Dynamics**
- Use ML to fit a dynamic model with
  - **Features**: $\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, u_k$
  - **Label**: y_{k+1}
- Use either on-line optimisation or option 1 discussed earlier.

Advantage:

(-) If successful → Predictive dynamics.

Drawbacks:

(-) Unnecessarily ambitious for MPC

(-) Hard (error propagation)

Direct single shooting formulation of MPC:

$$\boldsymbol{u}_k^* \leftarrow \min_{\boldsymbol{u}_k \in \mathbb{U}^N} \Big[J(\boldsymbol{u}_k \mid x_k) \Big] \qquad \boldsymbol{u}_k := \begin{bmatrix} u_k \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \big[\mathbb{R}^{n_u}\big]^N$$

MPC Feedback Control

$$K_{\text{MPC}}(x_k) = u_k^*$$

Standard settings

$$J(\boldsymbol{u}_k \mid x_k) := \sum_{i=1}^{N} \ell(x_{k+i}, u_{k+i-1})$$

$$x_{k+i+1} = f(x_{k+i}, u_{k+i})$$

Possible use of ML in MPC design

Cool Option Identify cost + on-line solution

- **Unknown Dynamics**
- Gather measurements under
  - open-loop control OR
  - loosely defined feedback
- Use ML to fit a model with
  - **Features**: $\Phi(\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, \boldsymbol{u}_k)$
  - **Label**: $J(\Phi)$
- Use on-line derivative-free solvers (ML Models are rarely differentiable)

(+) fit only what is needed (single model)

(+) Very light computation per samples

(+) output MPC feedback (no observer)

Direct single shooting formulation of MPC:

$$\boldsymbol{u}_k^* \leftarrow \min_{\boldsymbol{u}_k \in \mathbb{U}^N} \Big[J(\boldsymbol{u}_k \mid x_k)\Big] \qquad \boldsymbol{u}_k := \begin{bmatrix} u_k \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \left[\mathbb{R}^{n_u}\right]^N$$

MPC Feedback Control

$$K_{\text{MPC}}(x_k) = u_k^*$$

Standard settings

$$J(\boldsymbol{u}_k \mid x_k) := \sum_{i=1}^{N} \ell(x_{k+i}, u_{k+i-1})$$

$$x_{k+i+1} = f(x_{k+i}, u_{k+i})$$

The illustrative example

Reactor with two parallel reactions

$$R \rightarrow P_1 \quad R \rightarrow P_2$$

$$\begin{aligned}\dot{x}_1 &= 1 - 10^4 x_1^2 e^{-1/x_3} - 400 x_1 e^{-0.55/x_3} - x_1 \\ \dot{x}_2 &= 10^4 x_1^2 e^{-1/x_3} - x_2 \\ \dot{x}_3 &= u - x_3\end{aligned}$$

- x_1 concentration of R
- x_2 concentration of P_1 $(= y)$
- x_3 temperature of the mixture
- $u \in [0.049, 0.449]$ heat flow

[Bailey et al. Cyclic operation of reaction systems: effect of heat and mass transfer resistance. AIChE Journal, 17(4):818-825, 1971].

The illustrative example

Reactor with two parallel reactions

$$R \to P_1 \quad R \to P_2$$

$$\begin{aligned}\dot{x}_1 &= 1 - 10^4 x_1^2 e^{-1/x_3} - 400 x_1 e^{-0.55/x_3} - x_1\\ \dot{x}_2 &= 10^4 x_1^2 e^{-1/x_3} - x_2\\ \dot{x}_3 &= u - x_3\end{aligned}$$

- x_1 concentration of R
- x_2 concentration of P_1 $(= y)$
- x_3 temperature of the mixture
- $u \in [0.049, 0.449]$ heat flow

[Bailey et al. Cyclic operation of reaction systems: effect of heat and mass transfer resistance. AIChE Journal, 17(4):818-825, 1971].

Objective

Maximise the production of $y = x_2$

Cost function

$$J := -\left(\frac{1}{N}\mathbf{1}^T \cdot \boldsymbol{y}_k + \alpha y_{k+N}\right)$$

where

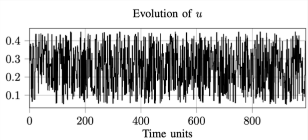
$$\boldsymbol{y}_k := \begin{bmatrix} y_{k+1} \\ \vdots \\ y_{k+N} \end{bmatrix}$$

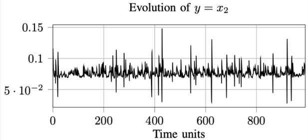
Optimal steady state

$$x_{st} = (0.0832, 0.0846, 0.149)$$

Sketch of the methodology

Step 1. Generate Learning Data

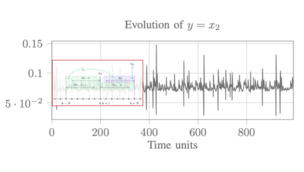


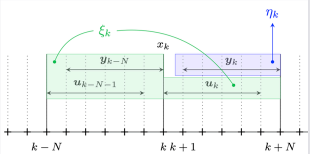


- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

Sketch of the methodology

Step 1. Generate Learning Data

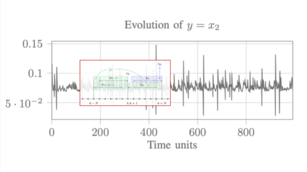


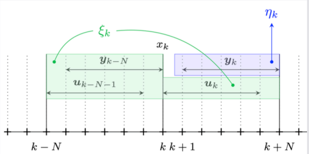


- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

Sketch of the methodology

Step 1. Generate Learning Data

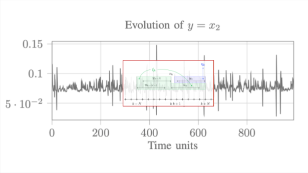


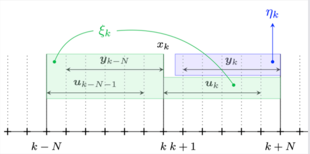


- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

Sketch of the methodology

Step 1. Generate Learning Data

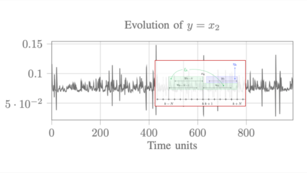


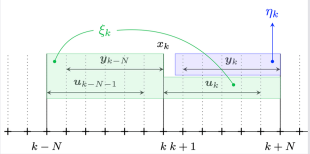


- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

Sketch of the methodology

Step 1. Generate Learning Data

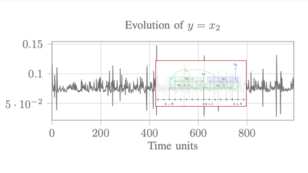


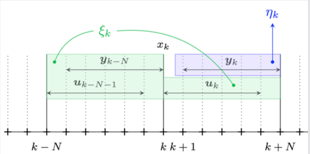


- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

Sketch of the methodology

Step 1. Generate Learning Data





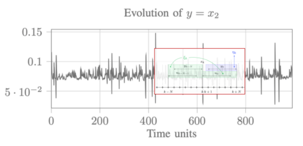
- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

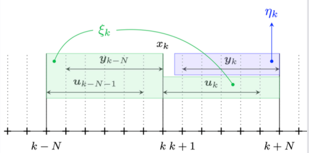
→ number of samples $n_s = 50324 - 2N$ samples:

$$\left\{\xi_s, \eta_s\right\}_{s=1}^{n_s} \qquad \begin{aligned} \xi_s &= (\boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}, \boldsymbol{u}_{k_s}) \in \mathbb{R}^{3N} \\ \eta_s &= J(\boldsymbol{u}_{k_s} \mid \boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}) \in \mathbb{R} \\ &= -\Big(\frac{1}{N}\mathbf{1}^T \cdot \boldsymbol{y}_{k_s} + \alpha y_{k_s+N}\Big) \end{aligned}$$

Sketch of the methodology

Step 1. Generate Learning Data





- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

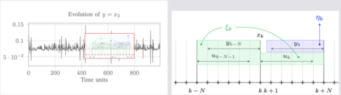
→ number of samples $n_s = 50324 - 2N$ samples:

$$\left\{\xi_s, \eta_s\right\}_{s=1}^{n_s} \qquad \begin{aligned} \xi_s &= (\boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}, \boldsymbol{u}_{k_s}) \in \mathbb{R}^{3N} \\ \eta_s &= J(\boldsymbol{u}_{k_s} \mid \boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}) \in \mathbb{R} \\ &= -\left(\frac{1}{N}\mathbf{1}^T \cdot \boldsymbol{y}_{k_s} + \alpha y_{k_s+N}\right) \end{aligned}$$

$N = 100 \rightarrow \xi_s \in \mathbb{R}^{300}$ too high for an appropriate ML design.

Sketch of the methodology

Step 1. Generate Learning Data

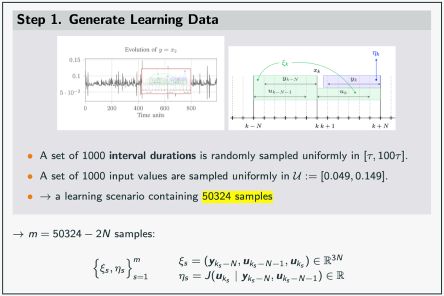


- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

→ $m = 50324 - 2N$ samples:

$$\left\{\xi_s, \eta_s\right\}_{s=1}^{m} \qquad \begin{array}{l} \xi_s = (\boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}, \boldsymbol{u}_{k_s}) \in \mathbb{R}^{3N} \\ \eta_s = J(\boldsymbol{u}_{k_s} \mid \boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}) \in \mathbb{R} \end{array}$$

Sketch of the methodology



Step 1. Generate Learning Data

- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

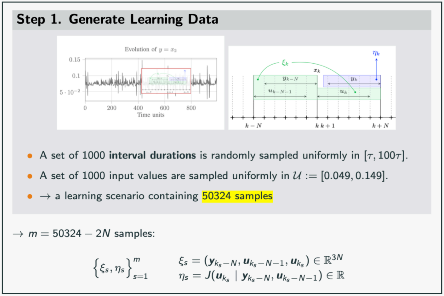
→ $m = 50324 - 2N$ samples:

$$\left\{\xi_s, \eta_s\right\}_{s=1}^{m} \qquad \begin{aligned} \xi_s &= (\boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}, \boldsymbol{u}_{k_s}) \in \mathbb{R}^{3N} \\ \eta_s &= J(\boldsymbol{u}_{k_s} \mid \boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}) \in \mathbb{R} \end{aligned}$$

Step 2. Features selection

$$\begin{aligned} \Phi(\xi) := \Big[&\phi_0(\boldsymbol{y}_{k-N}), \dots, \phi_{n_m-1}(\boldsymbol{y}_{k-N}), \\ &\phi_0(\boldsymbol{u}_{k-N}), \dots, \phi_{n_m-1}(\boldsymbol{u}_{k-N}), \\ &\phi_0(\boldsymbol{u}_k), \dots, \phi_{n_m-1}(\boldsymbol{u}_k)\Big] \in \mathbb{R}^{3n_m} \end{aligned}$$

Sketch of the methodology



Step 1. Generate Learning Data

- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

→ $m = 50324 - 2N$ samples:

$$\left\{\xi_s, \eta_s\right\}_{s=1}^{m} \qquad \begin{aligned}\xi_s &= (\boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}, \boldsymbol{u}_{k_s}) \in \mathbb{R}^{3N} \\ \eta_s &= J(\boldsymbol{u}_{k_s} \mid \boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}) \in \mathbb{R}\end{aligned}$$

Step 2. Features selection

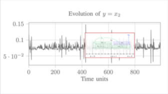
$$\begin{aligned}\Phi(\xi) := \Big[&\phi_0(\boldsymbol{y}_{k-N}), \dots, \phi_{n_m-1}(\boldsymbol{y}_{k-N}),\\ &\phi_0(\boldsymbol{u}_{k-N}), \dots, \phi_{n_m-1}(\boldsymbol{u}_{k-N}),\\ &\phi_0(\boldsymbol{u}_k), \dots, \phi_{n_m-1}(\boldsymbol{u}_k)\Big] \in \mathbb{R}^{3n_m}\end{aligned}$$

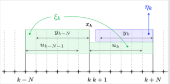
$\boldsymbol{s}$: a signal defined on I

$$\phi_j(\boldsymbol{s}) := \frac{1}{|I|}\sum_{i=1}^{|I|} s^{(j)}(i)$$

Sketch of the methodology

Step 1. Generate Learning Data





- A set of 1000 **interval durations** is randomly sampled uniformly in $[\tau, 100\tau]$.
- A set of 1000 input values are sampled uniformly in $\mathcal{U} := [0.049, 0.149]$.
- → a learning scenario containing 50324 samples

→ $m = 50324 - 2N$ samples:

$$\left\{\xi_s, \eta_s\right\}_{s=1}^{m} \qquad \begin{aligned} \xi_s &= (\boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}, \boldsymbol{u}_{k_s}) \in \mathbb{R}^{3N} \\ \eta_s &= J(\boldsymbol{u}_{k_s} \mid \boldsymbol{y}_{k_s-N}, \boldsymbol{u}_{k_s-N-1}) \in \mathbb{R} \end{aligned}$$

Machine Learning Problem

$$\eta \approx \mathsf{ML}(\Phi(\xi)) =: \hat{J}(\xi)$$

Features $\Phi \in \mathbb{R}^{3n_m} (= \mathbb{R}^9)$
Label η=Cost function

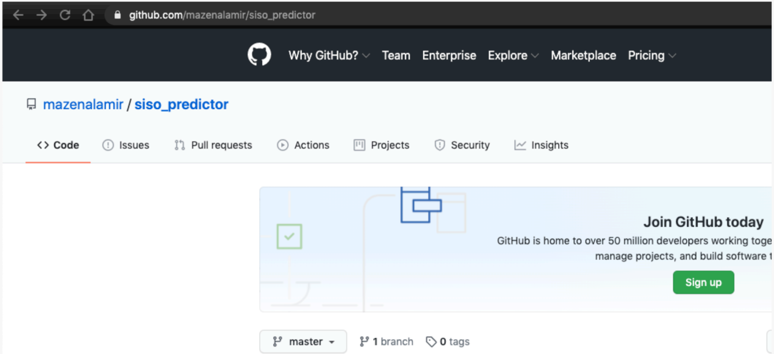
Step 2. Features selection

$$\Phi(\xi) := \Big[\phi_0(\boldsymbol{y}_{k-N}), \dots, \phi_{n_m-1}(\boldsymbol{y}_{k-N}),$$
$$\phi_0(\boldsymbol{u}_{k-N}), \dots, \phi_{n_m-1}(\boldsymbol{u}_{k-N}),$$
$$\phi_0(\boldsymbol{u}_k), \dots, \phi_{n_m-1}(\boldsymbol{u}_k)\Big] \in \mathbb{R}^{3n_m}$$

$\boldsymbol{s}$: a signal defined on I

$$\phi_j(\boldsymbol{s}) := \frac{1}{|I|}\sum_{i=1}^{|I|} s^{(j)}(i)$$

Freely available python implementation on GitHub



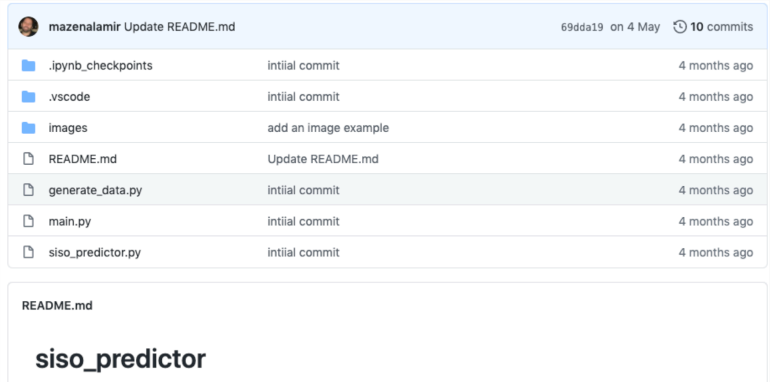
github.com/mazenalamir/siso_predictor

mazenalamir / siso_predictor

Join GitHub today

GitHub is home to over 50 million developers working toge
manage projects, and build software t

Freely available python implementation on GitHub



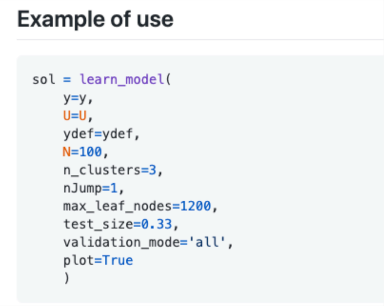
mazenalamir Update README.md 69dda19 on 4 May 10 commits

.ipynb_checkpoints	intiial commit	4 months ago
.vscode	intiial commit	4 months ago
images	add an image example	4 months ago
README.md	Update README.md	4 months ago
generate_data.py	intiial commit	4 months ago
main.py	intiial commit	4 months ago
siso_predictor.py	intiial commit	4 months ago

README.md

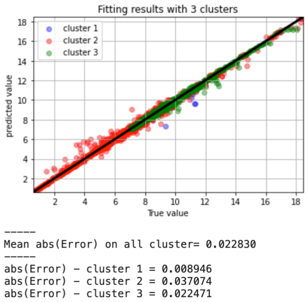
siso_predictor

Freely available python implementation on GitHub



Example of use

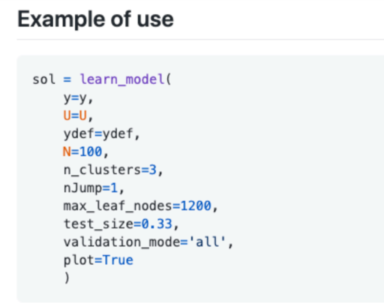
```
sol = learn_model(
    y=y,
    U=U,
    ydef=ydef,
    N=100,
    n_clusters=3,
    nJump=1,
    max_leaf_nodes=1200,
    test_size=0.33,
    validation_mode='all',
    plot=True
    )
```



```
-----
Mean abs(Error) on all cluster= 0.022830
-----
abs(Error) - cluster 1 = 0.008946
abs(Error) - cluster 2 = 0.037074
abs(Error) - cluster 3 = 0.022471
```

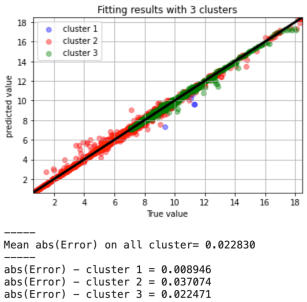
- Random Forest structure (SKLEARN)
- 33% test for train/test data splitting
- Clustering uses KNN (SKLEARN)

Freely available python implementation on GitHub



Example of use

```
sol = learn_model(
    y=y,
    U=U,
    ydef=ydef,
    N=100,
    n_clusters=3,
    nJump=1,
    max_leaf_nodes=1200,
    test_size=0.33,
    validation_mode='all',
    plot=True
    )
```



```
-----
Mean abs(Error) on all cluster= 0.022830
-----
abs(Error) - cluster 1 = 0.008946
abs(Error) - cluster 2 = 0.037074
abs(Error) - cluster 3 = 0.022471
```

- Random Forest structure (SKLEARN)
- 33% test for train/test data splitting
- Clustering uses KNN (SKLEARN)

We have a cost function model

$$J = \Big[\mathrm{ML}\circ\Phi\Big](\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, \boldsymbol{u}_k)$$

Solving the optimization problem

The Model $J = \Big[\mathtt{ML} \circ \Phi\Big](\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, \boldsymbol{u}_k))$

The cost induced by a future control sequence $\boldsymbol{u}_k$ given the past measurements $(\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N})$.

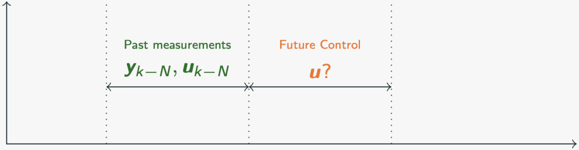
Solving the optimization problem

The Model $$J = \Big[\mathtt{ML} \circ \Phi\Big](\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, \boldsymbol{u}_k))$$

The cost induced by a future control sequence $\boldsymbol{u}_k$ given the past measurements $(\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N})$.

The Data-Driven MPC Formulation at instant k

$$\boldsymbol{u}_k^* \leftarrow \arg\min_{\boldsymbol{u}\in\mathbb{U}^N} \Big[\mathtt{ML} \circ \Phi\Big](\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, \boldsymbol{u})$$



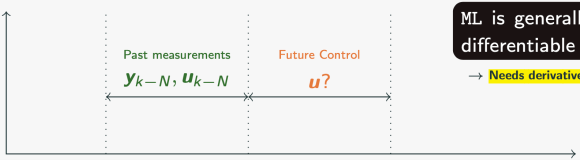
Solving the optimization problem

The Model $J = \Big[\mathtt{ML} \circ \Phi\Big]\big(\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, \boldsymbol{u}_k\big)$

The cost induced by a future control sequence $\boldsymbol{u}_k$ given the past measurements $(\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N})$.

The Data-Driven MPC Formulation at instant k

$$\boldsymbol{u}_k^* \leftarrow \arg \min_{\boldsymbol{u}\in\mathbb{U}^N} \Big[\mathtt{ML} \circ \Phi\Big]\big(\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}, \boldsymbol{u}\big)$$

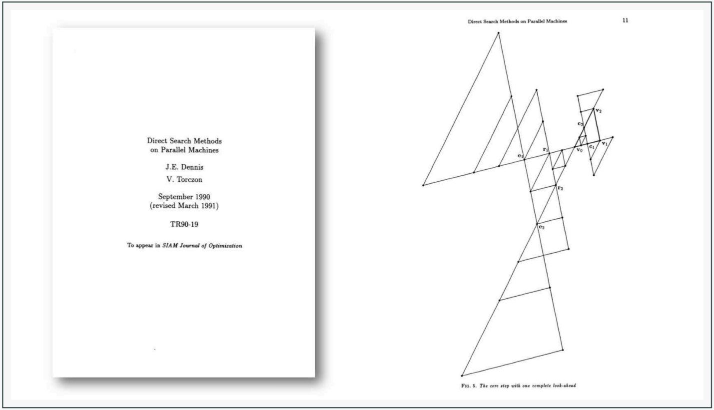


`ML` is generally a non differentiable function

→ **Needs derivative free solver**

Solving the optimization problem

Torczon's Algorithm



Direct Search Methods
on Parallel Machines

J.E. Dennis
V. Torczon

September 1990
(revised March 1991)

TR90-19

To appear in *SIAM Journal of Optimization*

Direct Search Methods on Parallel Machines 11

FIG. 5. *The core step with one complete look-ahead*

V. Torczon. *On the convergence of pattern search algorithms.*
SIAM J. on Optimization, 7(1):1-25, 1997

Solving the optimization problem

Suppose n Decision variables

Take $n+1$ initial nodes

Determine the best node

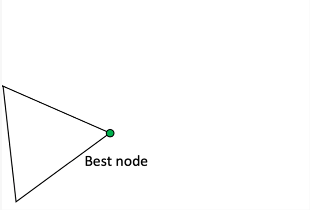


Illustration of Torczon Algorithm for 2D problems

Solving the optimization problem

Suppose n Decision variables

Take $n + 1$ initial nodes

Determine the best node

Try a Reflexion

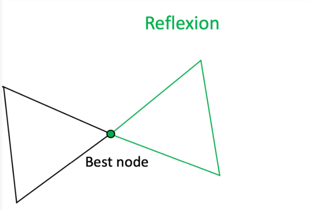


Illustration of Torczon Algorithm for 2D problems

Solving the optimization problem

Suppose *n* Decision variables

Take $n+1$ initial nodes

Determine the best node

Try a Reflexion

If successful Try an extension

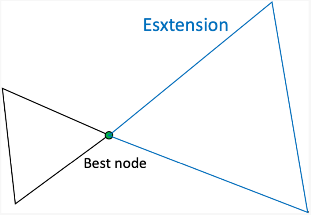


Illustration of Torczon Algorithm for 2D problems

Solving the optimization problem

Suppose n Decision variables

Take $n+1$ initial nodes
Determine the best node

Try a Reflexion

If successful Try an extension

Otherwise adopt a contraction

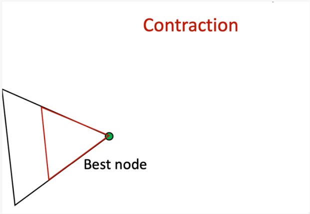


Illustration of Torczon Algorithm for 2D problems

Solving the optimization problem

Suppose n Decision variables

Take $n+1$ initial nodes
Determine the best node

Try a Reflexion

If successful Try an extension

Otherwise adopt a contraction

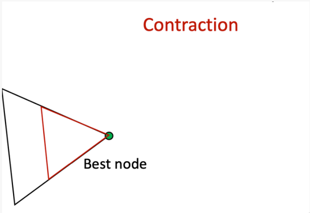


Illustration of Torczon Algorithm for 2D problems

This algorithm is suitable for problems of moderate size!

Solving the optimization problem

Suppose n Decision variables

Take $n+1$ initial nodes

Determine the best node

Try a Reflexion

If successful Try an extension

Otherwise adopt a contraction

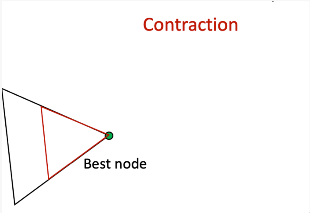
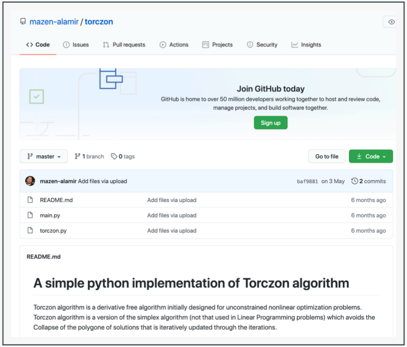


Illustration of Torczon Algorithm for 2D problems

This algorithm is suitable for problems of moderate size!

→ Control parametrisation *[last step!]*

Freely available python solver implementation on GitHub



Join GitHub today

GitHub is home to over 50 million developers working together to host and review code, manage projects, and build software together.

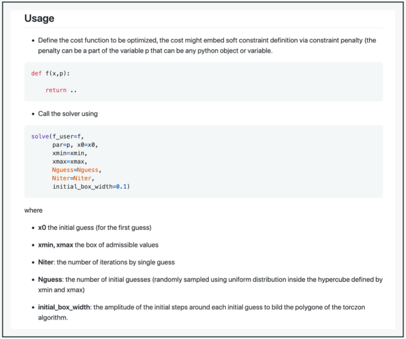
mazen-alamir Add files via upload baf9881 on 3 May 2 commits

README.md	Add files via upload	6 months ago
main.py	Add files via upload	6 months ago
torczon.py	Add files via upload	6 months ago

README.md

A simple python implementation of Torczon algorithm

Torczon algorithm is a derivative free algorithm initially designed for unconstrained nonlinear optimization problems. Torczon algorithm is a version of the simplex algorithm (not that used in Linear Programming problems) which avoids the Collapse of the polygone of solutions that is iteratively updated through the iterations.



Usage

- Define the cost function to be optimized, the cost might embed soft constraint definition via constraint penalty (the penalty can be a part of the variable p that can be any python object or variable.

```
def f(x,p):

    return ..
```

- Call the solver using

```
solve(f_user=f,
      par=p, x0=x0,
      xmin=xmin,
      xmax=xmax,
      Nguess=Nguess,
      Niter=Niter,
      initial_box_width=0.1)
```

where

- **x0** the initial guess (for the first guess)
- **xmin, xmax** the box of admissible values
- **Niter**: the number of iterations by single guess
- **Nguess**: the number of initial guesses (randomly sampled using uniform distribution inside the hypercube defined by xmin and xmax)
- **initial_box_width**: the amplitude of the initial steps around each initial guess to bild the polygone of the torczon algorithm.

Control parametrization

Data-Driven MPC at instant k

$$\boldsymbol{u}_k^* \leftarrow \arg\min_{\boldsymbol{u}\in\mathbb{U}^N} \Big[\mathtt{ML}\circ\Phi\Big](\boldsymbol{y}_{k-N},\boldsymbol{u}_{k-N},\boldsymbol{u})$$

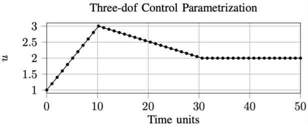
$$\downarrow$$

$$\boldsymbol{p}^* \leftarrow \arg\min_{p\in P} \Big[\mathtt{ML}\circ\Phi\Big](\boldsymbol{y}_{k-N},\boldsymbol{u}_{k-N},\mathcal{U}(p))$$

$$\downarrow$$

MPC Feedback

Apply the first control input in $\mathcal{U}(p^*)$



$\boldsymbol{u} = \mathcal{U}(p)$ where (here)

$$p := \Big[u(0), u(10), u(30)\Big]$$

Results

Two settings are tested $(\tau = 0.02, N = 100)$**, single cluster**

Setting 1

- $p := (u(0), u(10)) \in \mathbb{U}^2$
- $N_{guess} = 1,\ n_{iter}^{max} = 30$ (Torczon)
- $\rightarrow 1 \times 30 \times (2+4) = 180$ functions evals

Setting 2

- $p := (u(0), u(5), u(10), u(20)) \in \mathbb{U}^4$
- $N_{guess} = 3,\ n_{iter}^{max} = 50$ (Torczon)
- $\rightarrow 3 \times 50 \times (4+4) = 1200$ functions evals

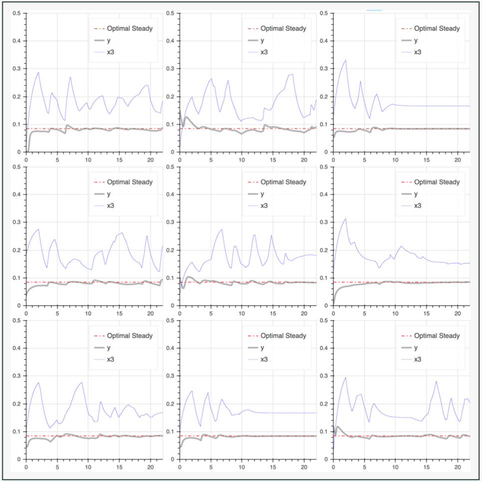
Single evaluation of the random-forest cost model = 1.7 msec

[Python 3 jupyter-notebook on a MacBook-Pro 2017, Mojave 10.14.5 with a 2.9 GHz Core i7 processor]

Results

Setting 1

- $p := (u(0), u(10)) \in \mathbb{U}^2$
- $N_{guess} = 1,\ n_{iter}^{max} = 30$ (Torczon)
- $\rightarrow 1 \times 30 \times (2+4) = 180$ functions evals



NOTA

The first two seconds ($N \times \tau$) under open-loop control waiting for the buffer $\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}$ to be available.

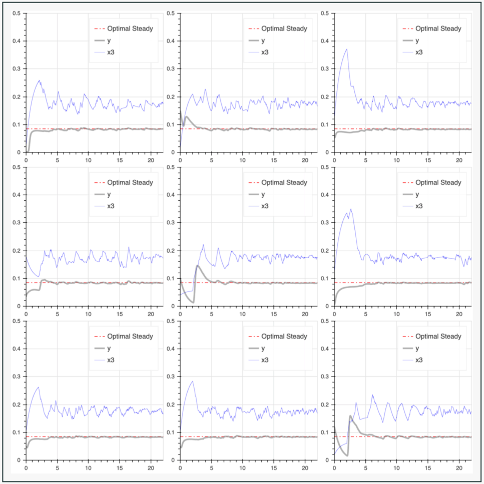
Results

Setting 2

- $p := (u(0), u(5), u(10), u(20)) \in \mathbb{U}^4$
- $N_{guess} = 3,\ n_{iter}^{max} = 50$ (Torczon)
- $\rightarrow 3 \times 50 \times (4+4) = 1200$ functions evals

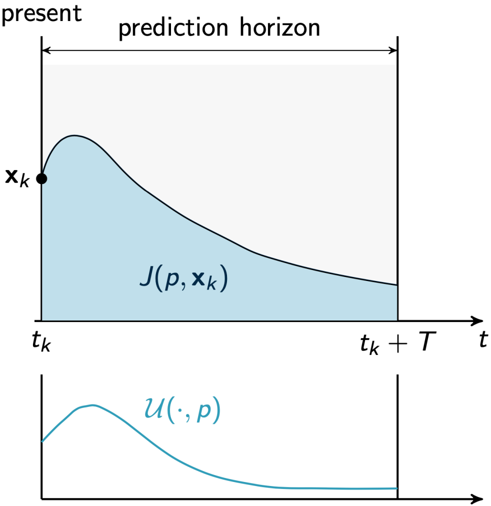
NOTA

The first two time units ($N \times \tau$) under open-loop control waiting for the buffer $\boldsymbol{y}_{k-N}, \boldsymbol{u}_{k-N}$ to be available.

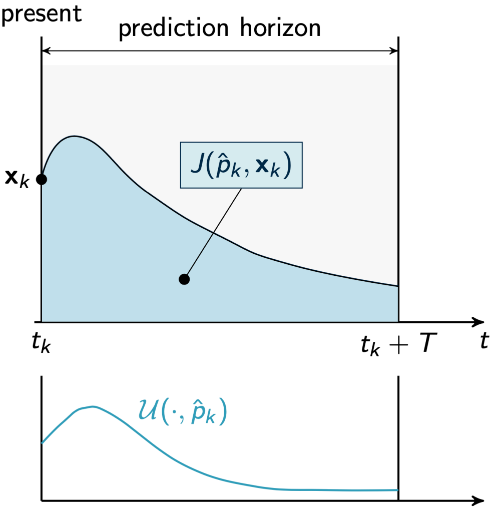


1. Data-driven partial observability assessment under uncertainties
2. Learning-Based Approximate Stochastic NMPC Design
3. Tractable stochastic NMPC by supervised clustering
4. Data-Driven NMPC by cost function identification
5. Learning-Based Monitoring updating period in Real-time NMPC

Ideal Framework: Recalls & Basic Notations



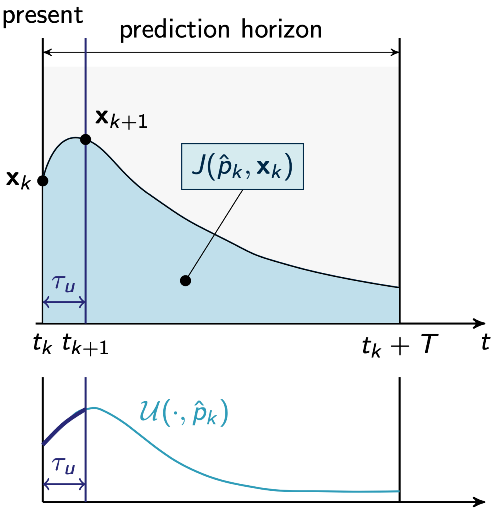
Ideal Framework: Recalls & Basic Notations

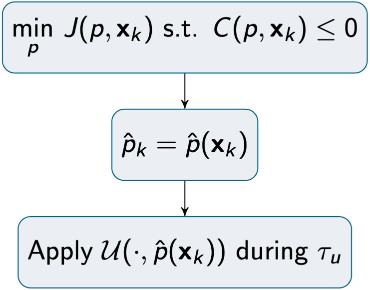


$$\min_{p} J(p, \mathbf{x}_k) \text{ s.t. } C(p, \mathbf{x}_k) \leq 0$$

$$\hat{p}_k = \hat{p}(\mathbf{x}_k)$$

Ideal Framework: Recalls & Basic Notations



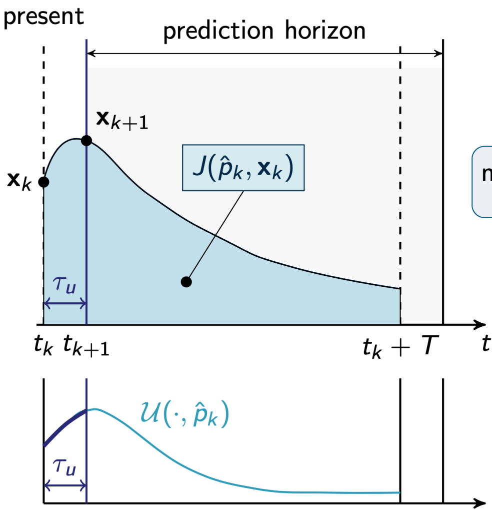


$\min_p J(p, \mathbf{x}_k)$ s.t. $C(p, \mathbf{x}_k) \leq 0$

$\hat{p}_k = \hat{p}(\mathbf{x}_k)$

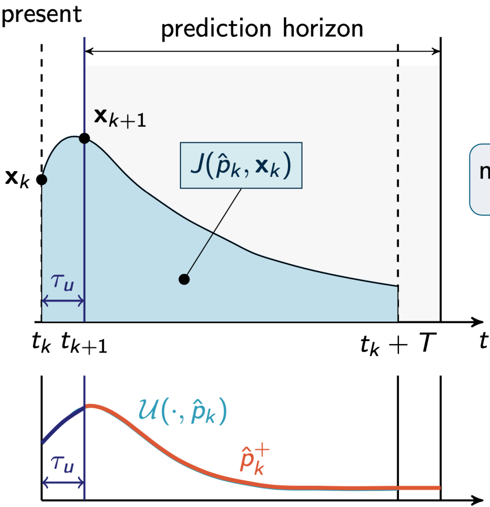
Apply $\mathcal{U}(\cdot, \hat{p}(\mathbf{x}_k))$ during τ_u

Ideal Framework: Recalls & Basic Notations



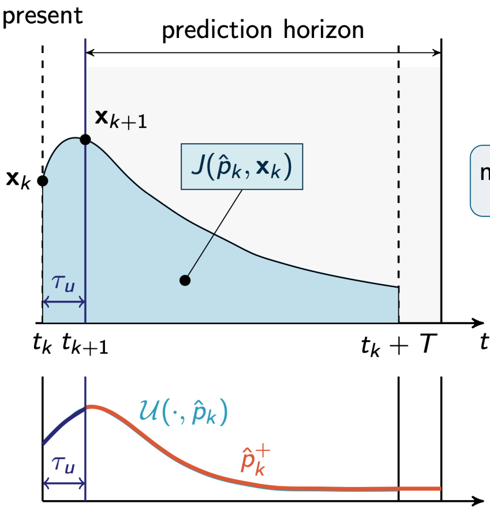
$$\min_{p} J(p, \mathbf{x}_{k+1}) \text{ s.t. } C(p, \mathbf{x}_{k+1}) \leq 0$$

Ideal Framework: Recalls & Basic Notations



$$\min_{p} J(p, \mathbf{x}_{k+1}) \text{ s.t. } C(p, \mathbf{x}_{k+1}) \leq 0$$

Ideal Framework: Recalls & Basic Notations

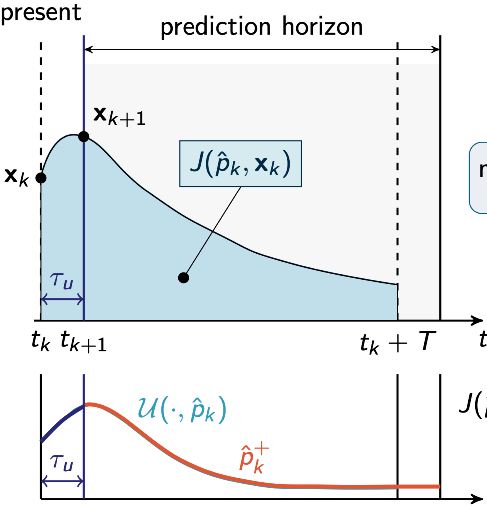


$$\min_{p} \; J(p, \mathbf{x}_{k+1}) \text{ s.t. } C(p, \mathbf{x}_{k+1}) \leq 0$$

Initialize at $\hat{p}_k^+$ (hot start)

$$\hat{p}_{k+1} = \hat{p}(\mathbf{x}_{k+1})$$

Ideal Framework: Recalls & Basic Notations



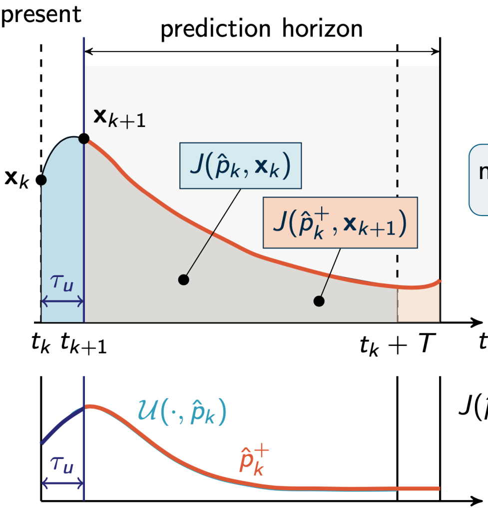
$$\min_p J(p, \mathbf{x}_{k+1}) \text{ s.t. } C(p, \mathbf{x}_{k+1}) \leq 0$$

Initialize at $\hat{p}_k^+$ (hot start)

$$\hat{p}_{k+1} = \hat{p}(\mathbf{x}_{k+1})$$

$$J(\hat{p}_{k+1}, \mathbf{x}_{k+1}) \leq J(\hat{p}_k^+, \mathbf{x}_{k+1})$$

Ideal Framework: Recalls & Basic Notations



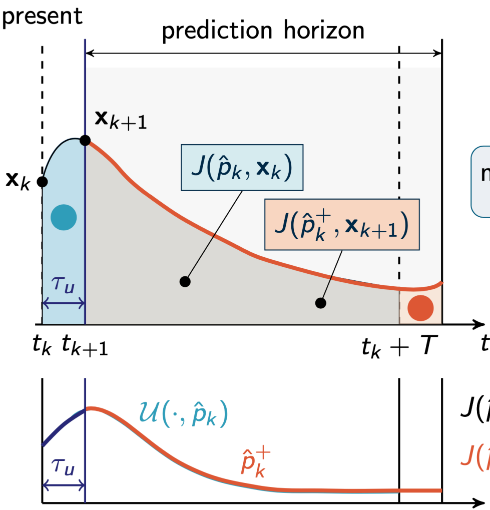
$$\min_p J(p, \mathbf{x}_{k+1}) \text{ s.t. } C(p, \mathbf{x}_{k+1}) \leq 0$$

Initialize at $\hat{p}_k^+$ (hot start)

$$\hat{p}_{k+1} = \hat{p}(\mathbf{x}_{k+1})$$

$$J(\hat{p}_{k+1}, \mathbf{x}_{k+1}) \leq J(\hat{p}_k^+, \mathbf{x}_{k+1})$$

Ideal Framework: Recalls & Basic Notations



$$\min_p J(p, \mathbf{x}_{k+1}) \text{ s.t. } C(p, \mathbf{x}_{k+1}) \leq 0$$

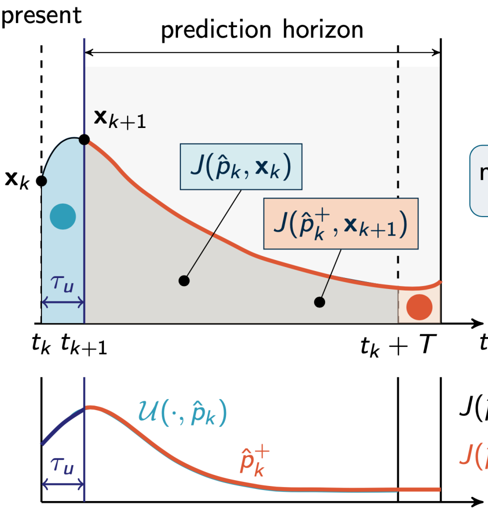
Initialize at $\hat{p}_k^+$ (hot start)

$$\hat{p}_{k+1} = \hat{p}(\mathbf{x}_{k+1})$$

$$J(\hat{p}_{k+1}, \mathbf{x}_{k+1}) \leq J(\hat{p}_k^+, \mathbf{x}_{k+1})$$

$J(\hat{p}_k^+, \mathbf{x}_{k+1}) = J(\hat{p}_k, \mathbf{x}_k) -$ ● $+$ ●

Ideal Framework: Recalls & Basic Notations



$$\min_{p} \; J(p, \mathbf{x}_{k+1}) \text{ s.t. } C(p, \mathbf{x}_{k+1}) \leq 0$$

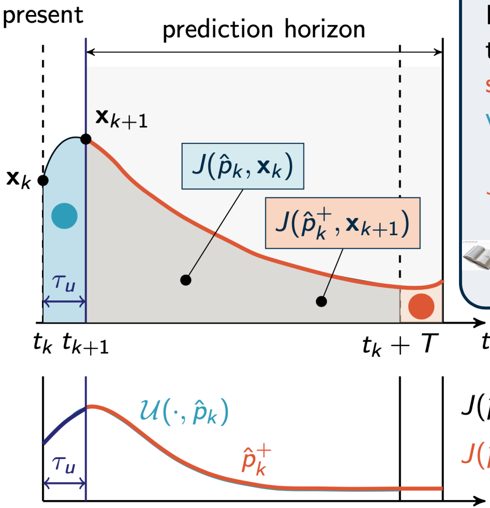
Initialize at $\hat{p}_k^+$ (hot start)

$$\hat{p}_{k+1} = \hat{p}(\mathbf{x}_{k+1})$$

$$J(\hat{p}_{k+1}, \mathbf{x}_{k+1}) \leq J(\hat{p}_k^+, \mathbf{x}_{k+1})$$

$$J(\hat{p}_k^+, \mathbf{x}_{k+1}) = J(\hat{p}_k, \mathbf{x}_k) \underbrace{- \bullet + \bullet}_{\leq 0 \; ?}$$

Ideal Framework: Recalls & Basic Notations



Keep in mind

In the **ideal framework**, when the horizon moves, the hot start $\hat{p}_k^+$ computed from the previous optimal solution $\hat{p}_k$ satisfies:

$$J(\hat{p}_k^+, \mathbf{x}_{k+1}) \leq J(\hat{p}_k, \mathbf{x}_k)$$

Mayne et al. Automatica (2000)

$$J(\hat{p}_{k+1}, \mathbf{x}_{k+1}) \leq J(\hat{p}_k^+, \mathbf{x}_{k+1})$$

$$J(\hat{p}_k^+, \mathbf{x}_{k+1}) = J(\hat{p}_k, \mathbf{x}_k) \underbrace{- \bullet + \bullet}_{\leq 0\ ?}$$

Ideal MPC: The key assumptions

Keep in mind

In the **ideal framework**, when the horizon moves, the hot start $\hat{p}_k^+$ computed from the previous optimal solution $\hat{p}_k$ satisfies:

$$J(\hat{p}_k^+, \mathbf{x}_{k+1}) \leq J(\hat{p}_k, \mathbf{x}_k)$$

Mayne et al. Automatica (2000).

- Formulation involving Final constraints
- (hot start)-compatible parametrization
- $\hat{p}_k$ sufficiently good

Ideal MPC: The key assumptions

Keep in mind

In a **realistic framework**, when the horizon moves, the hot start p_k^+ computed from the previous solution p_k satisfies:

$$J(p_k^+, \mathbf{x}_{k+1}) = J(p_k, \mathbf{x}_k) + D(\tau_u)$$

$D(0) = 0$
$D(\tau_u)$ is **not necessarily** ≤ 0**.**

- Formulation involving Final constraints
- (hot start)-compatible parametrization
- $\hat{p}_k$ sufficiently good

Ideal MPC: The key assumptions

Keep in mind

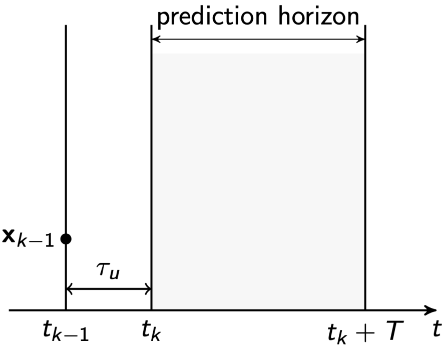
In a **realistic framework**, when the horizon moves, the hot start p_k^+ computed from the previous solution p_k satisfies:

$$J(p_k^+, \mathbf{x}_{k+1}) = J(p_k, \mathbf{x}_k) + D(\tau_u)$$

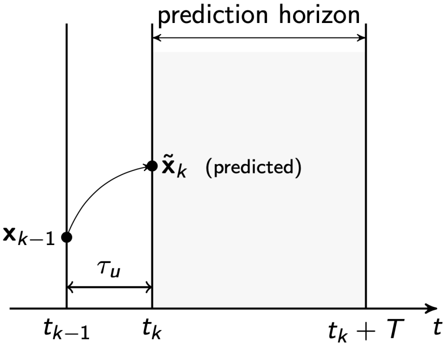
$D(0) = 0$
$D(\tau_u)$ is **not necessarily** ≤ 0.

Even with **perfect undisturbed** model

Preparation & Feedback Steps



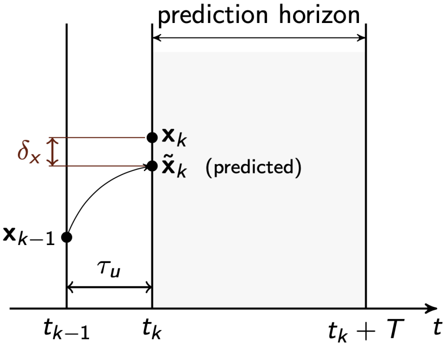
Preparation & Feedback Steps



1. Predict $\tilde{\mathbf{x}}_k$
2. During $[t_{k-1}, t_k]$

   Compute $\hat{p}(\tilde{\mathbf{x}}_k)$ [and $\dfrac{\partial \hat{p}_k}{\partial \mathbf{x}}$]

Preparation & Feedback Steps

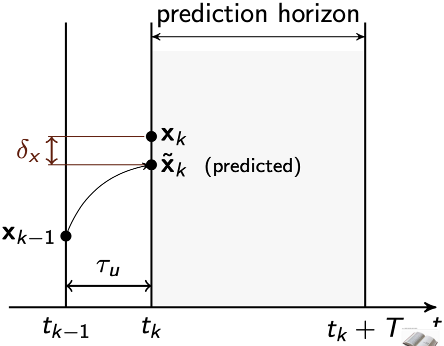


1. Predict $\tilde{\mathbf{x}}_k$
2. During $[t_{k-1}, t_k]$

   Compute $\hat{p}(\tilde{\mathbf{x}}_k)$ [and $\dfrac{\partial \hat{p}_k}{\partial \mathbf{x}}$]
3. Once $\mathbf{x}_k$ is available:

$$\hat{p}_k \leftarrow \hat{p}(\tilde{\mathbf{x}}_k) + \left[\frac{\partial \hat{p}_k}{\partial \mathbf{x}}\right] \cdot \delta_x$$

Preparation & Feedback Steps



1. Predict $\tilde{\mathbf{x}}_k$
2. During $[t_{k-1}, t_k]$

   Compute $\hat{p}(\tilde{\mathbf{x}}_k)$ [and $\dfrac{\partial \hat{p}_k}{\partial \mathbf{x}}$]

   (*preparation step*)
3. Once $\mathbf{x}_k$ is available:

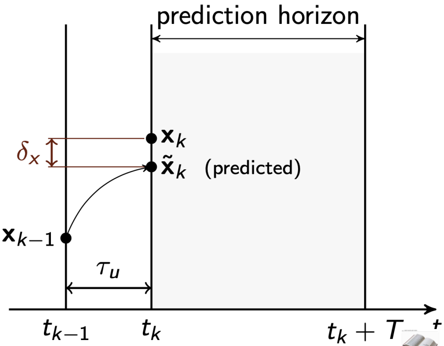
$$\hat{p}_k \leftarrow \hat{p}(\tilde{\mathbf{x}}_k) + \left[\frac{\partial \hat{p}_k}{\partial \mathbf{x}}\right] \cdot \delta_x$$

(*feedback step*)

Diehl et al. SIAM J. Ctrl and Opt. (2005)

Zavala and Biegler. Auomatica (2009)

Preparation & Feedback Steps



1. Predict $\tilde{\mathbf{x}}_k$
2. During $[t_{k-1}, t_k]$

   Compute $\hat{p}(\tilde{\mathbf{x}}_k)$ [and $\frac{\partial \hat{p}_k}{\partial \mathbf{x}}$]

   (*preparation step*)
3. Once $\mathbf{x}_k$ is available:

$$\hat{p}_k \leftarrow \hat{p}(\tilde{\mathbf{x}}_k) + \left[\frac{\partial \hat{p}_k}{\partial \mathbf{x}}\right] \cdot \delta_x$$

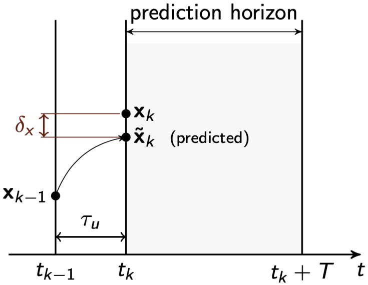
(*feedback step*)

Diehl et al. SIAM J. Ctrl and Opt. (2005)

Zavala and Biegler. Auomatica (2009)

NOTA: Even with feedback, uncertainty potentially increases $D(\tau_u)$

Definition of Fast NMPC Problems

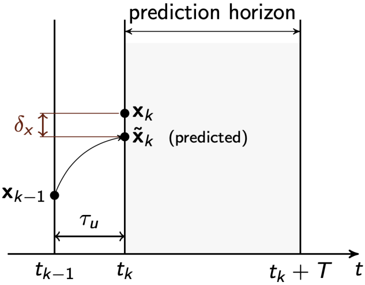


τ_u is the time between two control updating

τ_u is the time during which there is no feedback

$$\Rightarrow \tau_u \le \tau_u^{max}$$

Definition of Fast NMPC Problems



τ_u is the time between two control updating

τ_u is the time during which there is no feedback

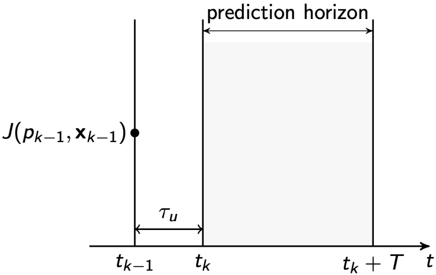
$$\Rightarrow \tau_u \leq \tau_u^{max}$$

Fast NMPC Problems

Fast NMPC problems are those for which

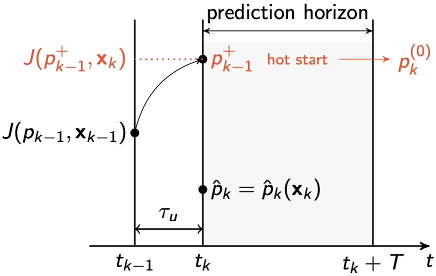
$$\tau_{solve}\Big(NLP(\tilde{\mathbf{x}}_k)\Big) \geq \tau_u^{max}$$

The Iterative Process



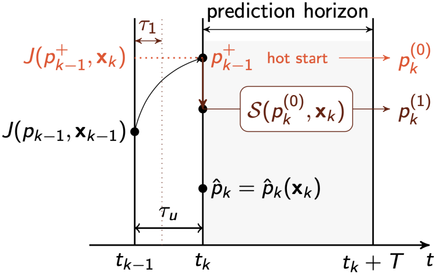
$$\boxed{p^{(i+1)} \leftarrow \mathcal{S}(p^{(i)}, \mathbf{x})}$$

The Iterative Process



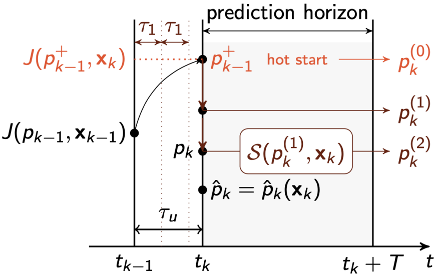
$$\boxed{p^{(i+1)} \leftarrow \mathcal{S}(p^{(i)}, \mathbf{x})}$$

The Iterative Process



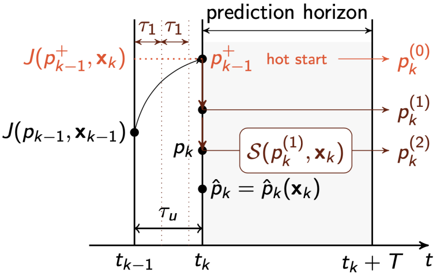
$$\boxed{p^{(i+1)} \leftarrow \mathcal{S}(p^{(i)}, \mathbf{x})}$$

The Iterative Process



$$\boxed{p^{(i+1)} \leftarrow \mathcal{S}(p^{(i)}, \mathbf{x})}$$

The Iterative Process

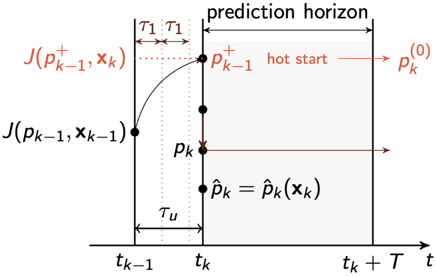


$$\boxed{p^{(i+1)} \leftarrow \mathcal{S}(p^{(i)}, \mathbf{x})}$$

$$\downarrow$$

$$\boxed{q = int\left(\frac{\tau_u}{\tau_1}\right)}$$

The Iterative Process



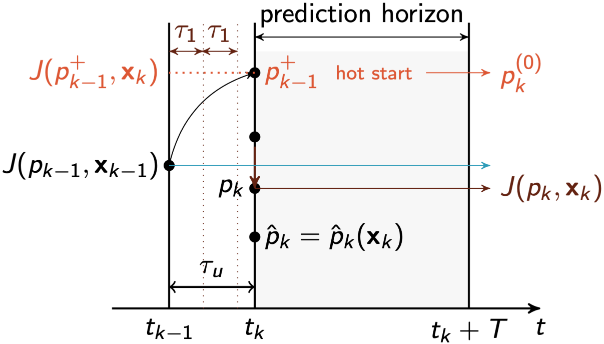
$$p^{(i+1)} \leftarrow \mathcal{S}(p^{(i)}, \mathbf{x})$$

$$q = int\left(\frac{\tau_u}{\tau_1}\right)$$

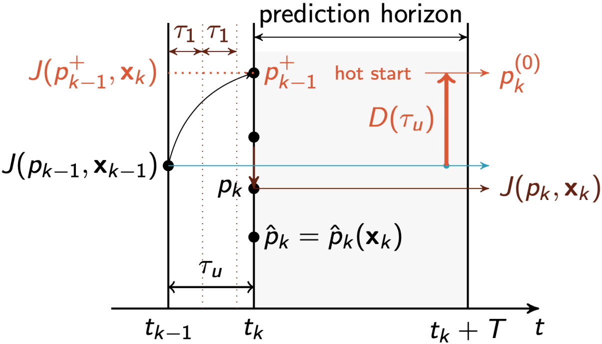
dynamic equation for p

$$p_k \quad := \quad \mathcal{S}^{(q)}(p^+_{k-1}, \mathbf{x}_{k-1})$$

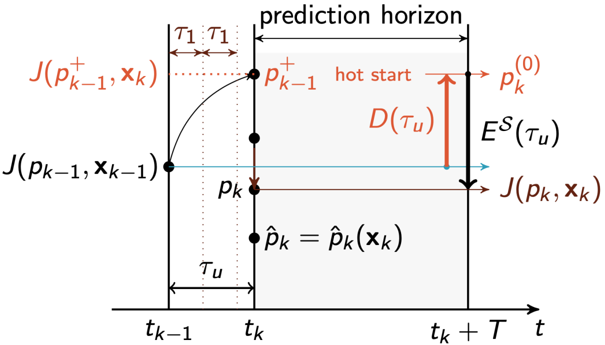
Sufficient Conditions of Success



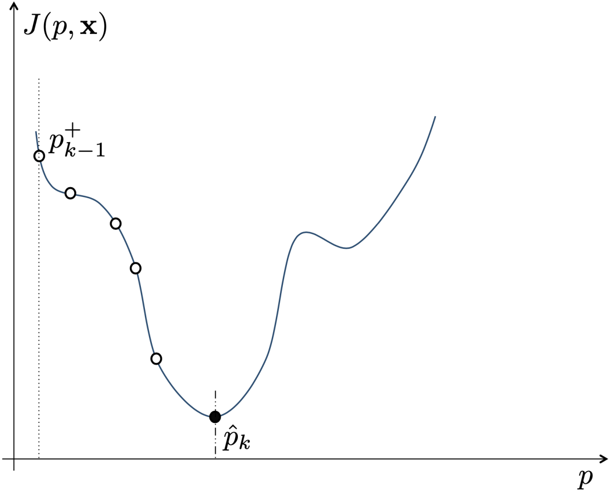
Sufficient Conditions of Success



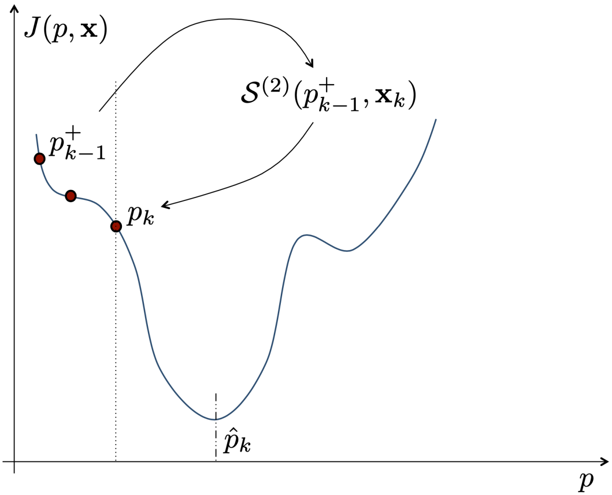
Sufficient Conditions of Success



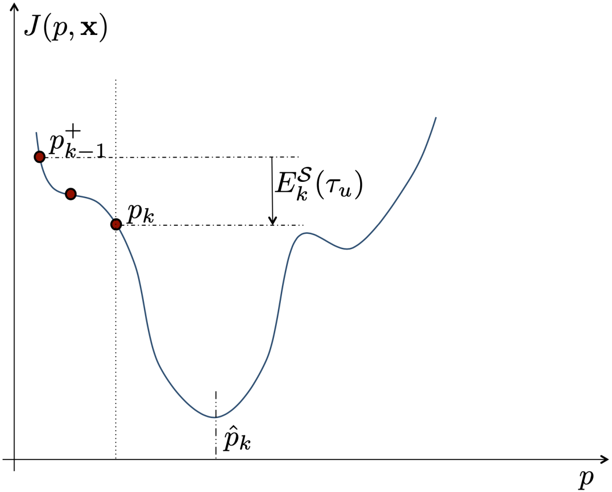
Closed-Loop Evolution of the Cost Function



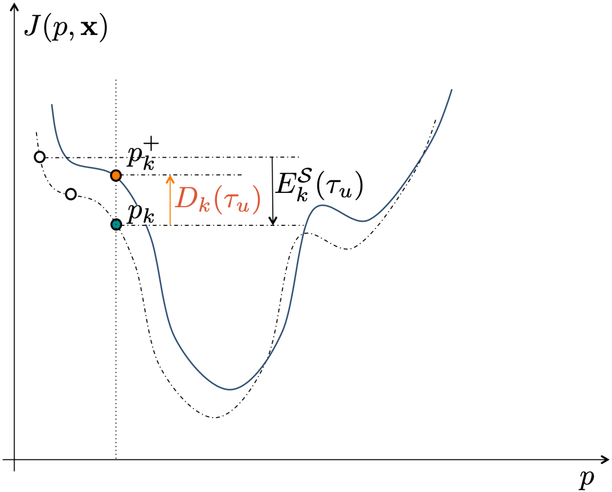
Closed-Loop Evolution of the Cost Function



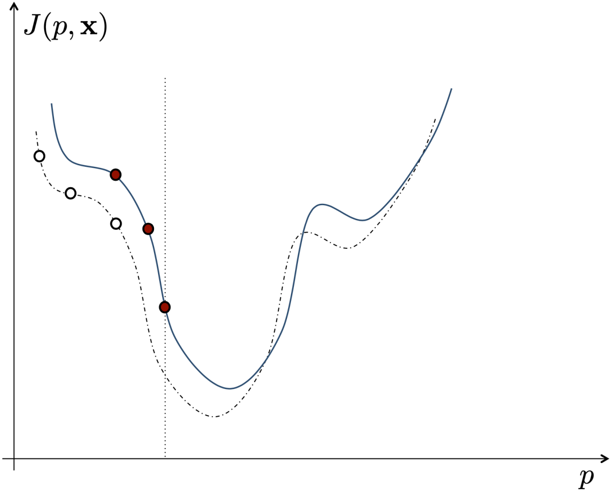
Closed-Loop Evolution of the Cost Function



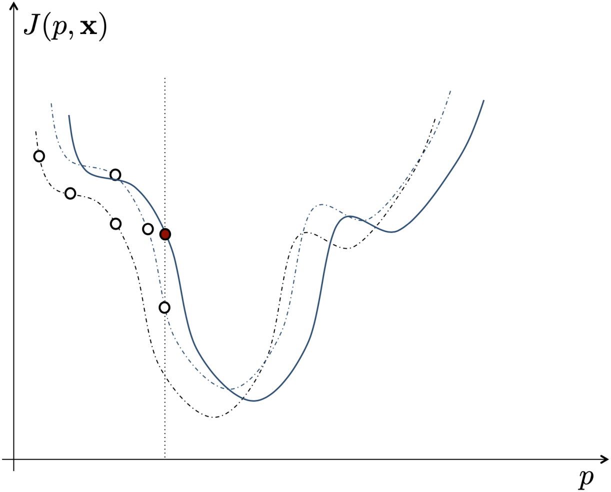
Closed-Loop Evolution of the Cost Function



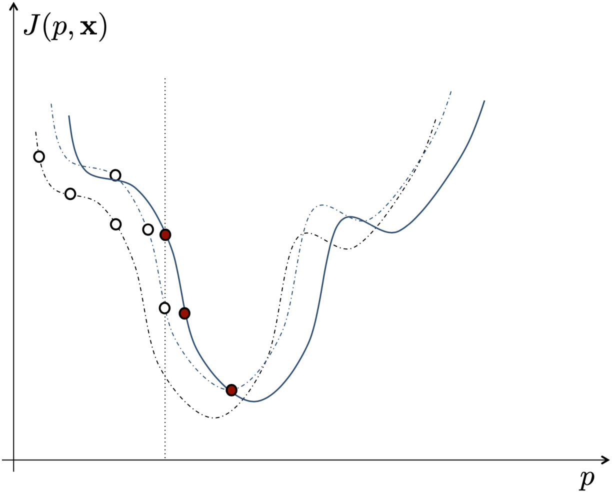
Closed-Loop Evolution of the Cost Function



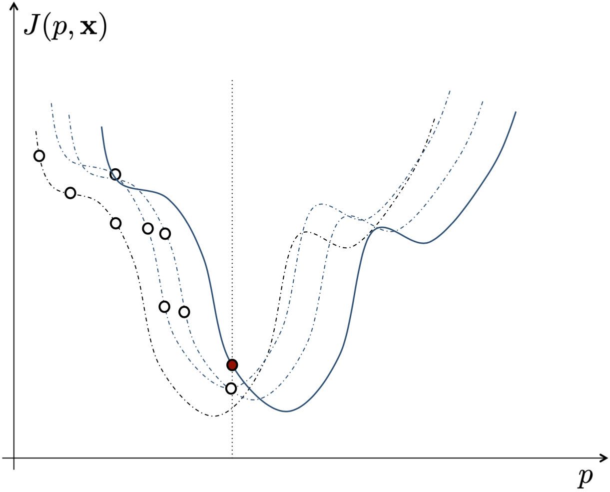
Closed-Loop Evolution of the Cost Function



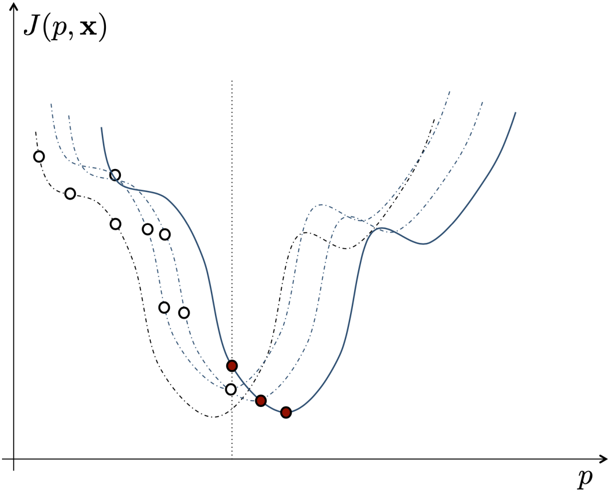
Closed-Loop Evolution of the Cost Function



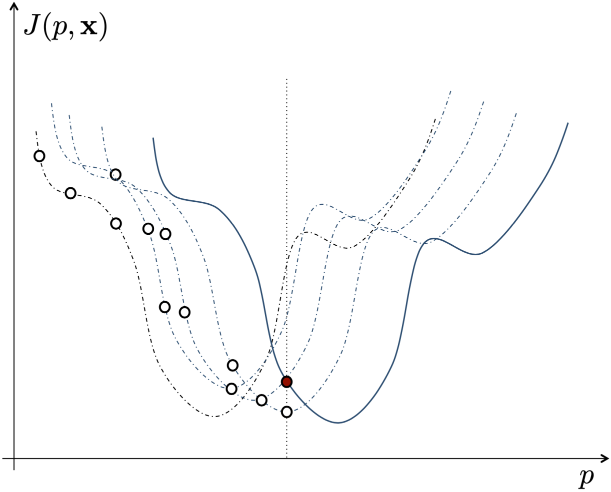
Closed-Loop Evolution of the Cost Function



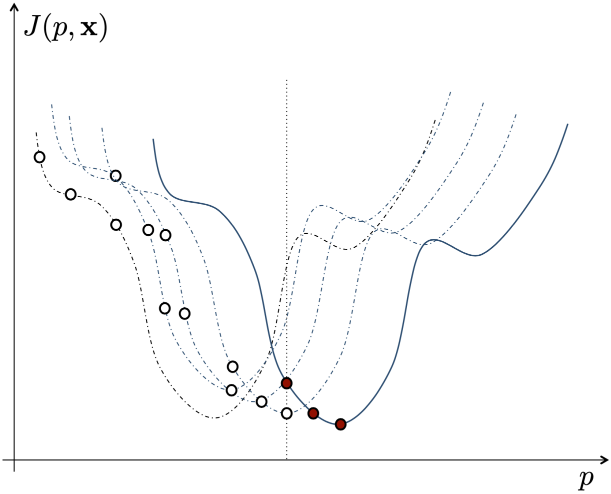
Closed-Loop Evolution of the Cost Function



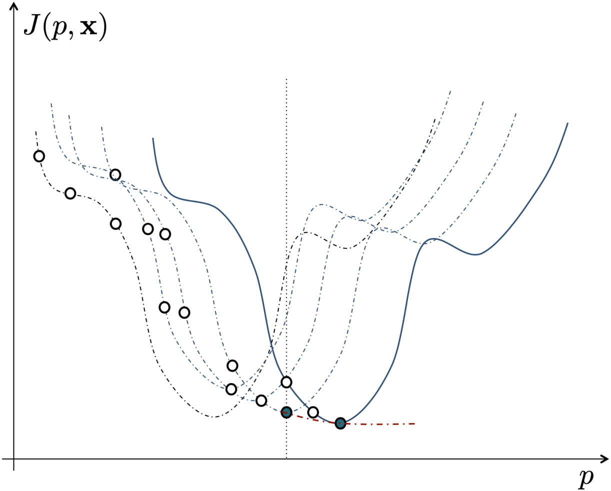
Closed-Loop Evolution of the Cost Function



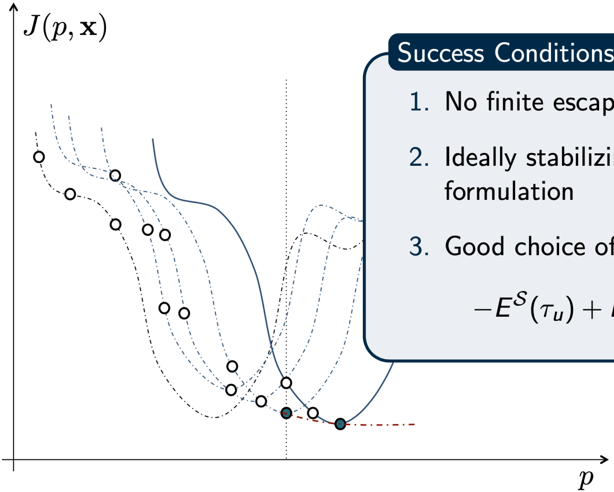
Closed-Loop Evolution of the Cost Function



Closed-Loop Evolution of the Cost Function



Closed-Loop Evolution of the Cost Function



Success Conditions

1. No finite escape time
2. Ideally stabilizing NMPC formulation
3. Good choice of $(\mathcal{S}, \tau_u)$:

$$-E^{\mathcal{S}}(\tau_u) + D(\tau_u) \ < \ 0$$

Reminder

$$\boxed{D(\tau_u) - E^{\mathcal{S}}(\tau_u) \; < 0}$$

Reminder

$$D(\tau_u) - E^{\mathcal{S}}(\tau_u) < 0$$

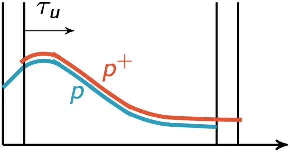
$D(\tau_u)$

$-E^{\mathcal{S}}(\tau_u)$

Reminder

$$D(\tau_u) - E^{\mathcal{S}}(\tau_u) \ < 0$$

$D(\tau_u)$



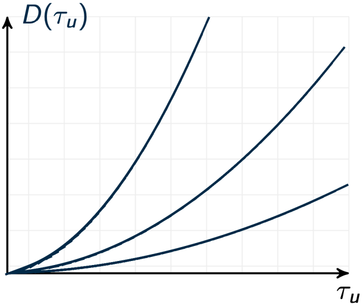
- $D(\tau_u) := J(p^+, \mathbf{x}^+) - J(p, \mathbf{x})$
- $D(0) = 0$, $D(\tau_u)$ can be ≥ 0
- $\tau_u \in [0, \tau_u^{max}]$
- Independent of the solver $\mathcal{S}$

$-E^{\mathcal{S}}(\tau_u)$

Reminder

$$D(\tau_u) - E^{\mathcal{S}}(\tau_u) < 0$$

$D(\tau_u)$



$-E^{\mathcal{S}}(\tau_u)$

Reminder

$$D(\tau_u) - E^{\mathcal{S}}(\tau_u) \; < 0$$

$D(\tau_u)$

$-E^{\mathcal{S}}(\tau_u)$

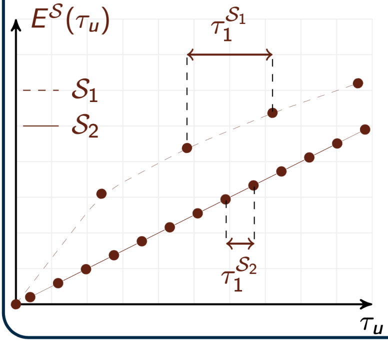
- $E^{\mathcal{S}}(\tau_u) := J(p^{(0)}, \mathbf{x}) - J(p^{(q^{\mathcal{S}})}, \mathbf{x})$
- $q^{\mathcal{S}} = int(\tau_u/\tau_1^{\mathcal{S}})$
- $\tau_1^{\mathcal{S}}$ time for a single iteration
- $\tau_u \in \{1, 2, \dots\} \times \tau_1^{\mathcal{S}}$
- $E^{\mathcal{S}}(0) = 0$

Reminder

$$D(\tau_u) - E^{\mathcal{S}}(\tau_u) \; < 0$$

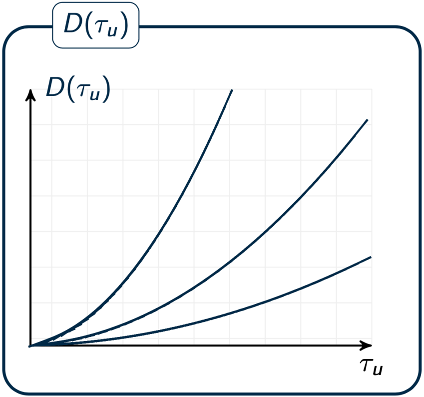
$D(\tau_u)$

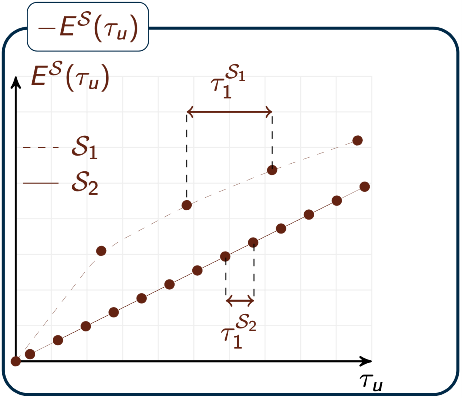
$-E^{\mathcal{S}}(\tau_u)$

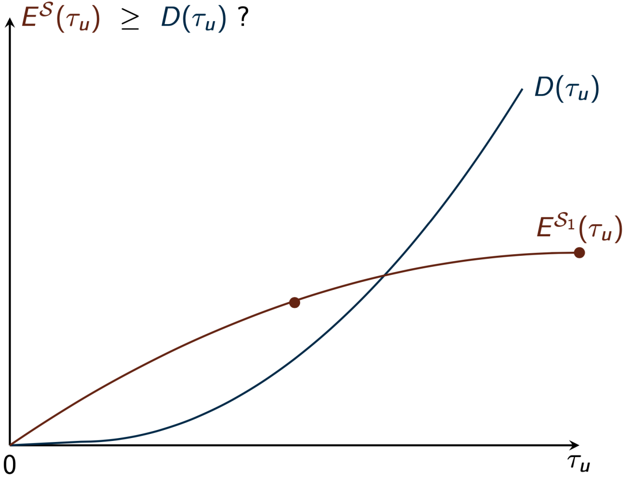


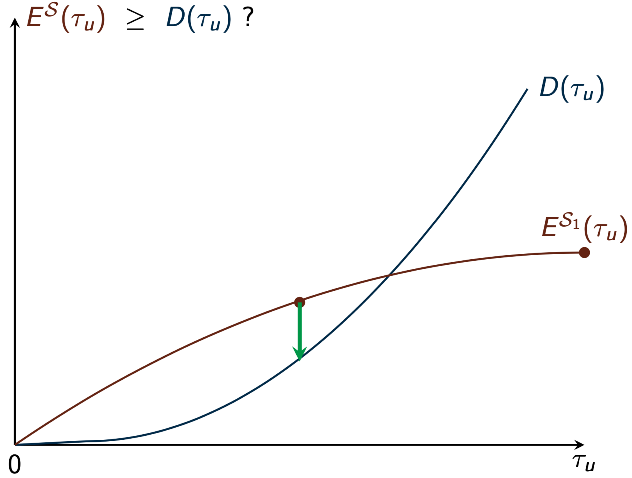
Reminder

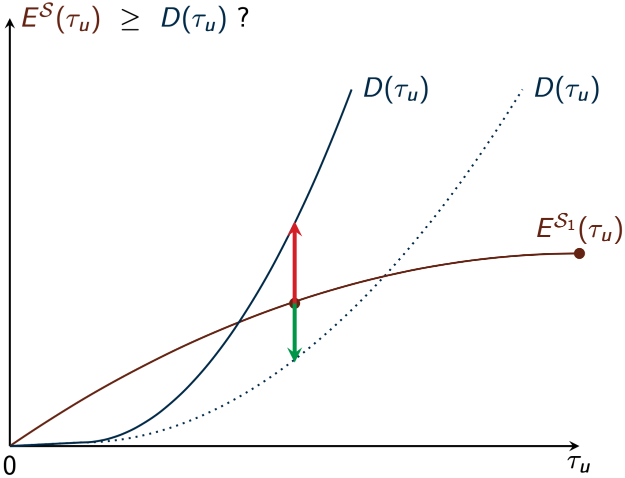
$$D(\tau_u) - E^{\mathcal{S}}(\tau_u) < 0$$

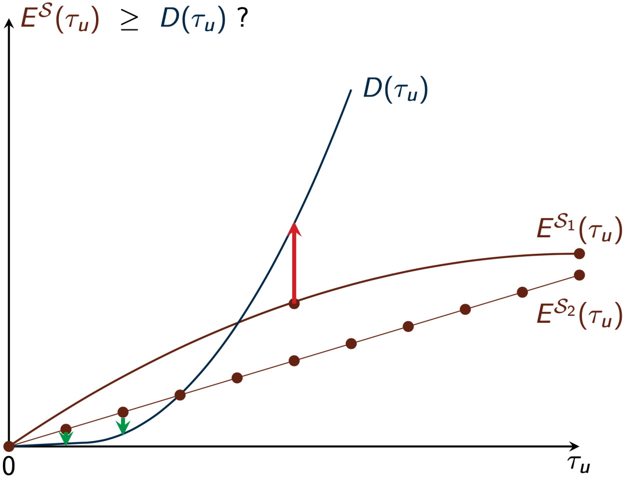




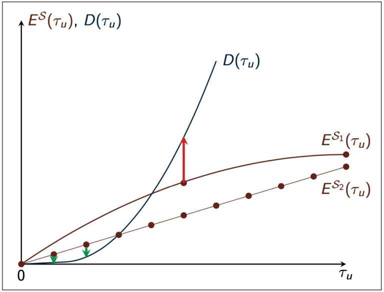

$E^{\mathcal{S}}(\tau_u) \geq D(\tau_u)$?
$D(\tau_u)$
$E^{\mathcal{S}_1}(\tau_u)$
0
τ_u


$E^{\mathcal{S}}(\tau_u) \;\geq\; D(\tau_u)$?
$D(\tau_u)$
$E^{\mathcal{S}_1}(\tau_u)$
0
τ_u


$E^{\mathcal{S}}(\tau_u) \;\geq\; D(\tau_u)$?
$D(\tau_u)$
$D(\tau_u)$
$E^{\mathcal{S}_1}(\tau_u)$
0
τ_u


$E^{\mathcal{S}}(\tau_u) \geq D(\tau_u)$?
$D(\tau_u)$
$E^{\mathcal{S}_1}(\tau_u)$
$E^{\mathcal{S}_2}(\tau_u)$
0
τ_u

Key properties of a solver



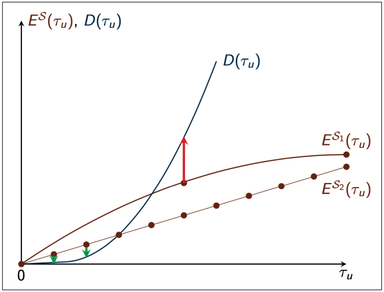
Keep in Mind

It is sometimes better to choose a **less efficient**[a] solver with **shorter preparation step** duration τ_1.

---

[a] per iteration

Key properties of a solver



Keep in Mind

It is sometimes better to choose a **less efficient**[a] solver with **shorter preparation step** duration τ_1.

---

[a] per iteration

Gradient-based studies

Bemporad and Patrinos (2012), Jones et al. (2012), MA (2013).

Heuristics for second order methods

Bock et al. SIAM (2007)

Comparing Algorithms: An Example

Houska et al. Automatica (2011)

Model

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= u \qquad u \in [-1, +1] \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -g \sin x_3 - u \cos x_3 - b x_4
\end{aligned}$$

Comparing Algorithms: An Example

Houska et al. Automatica (2011)

Model

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u \qquad u \in [-1, +1] \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= -g \sin x_3 - u \cos x_3 - b x_4 \end{aligned}$$

Cost

$$\sum_{i=1}^{N_p} \|\mathbf{x}_{k+i}^T\|_Q^2 + \|\mathbf{u}_{k+i-1}^T\|_R^2$$

Comparing Algorithms: An Example

Houska et al. Automatica (2011)

Model

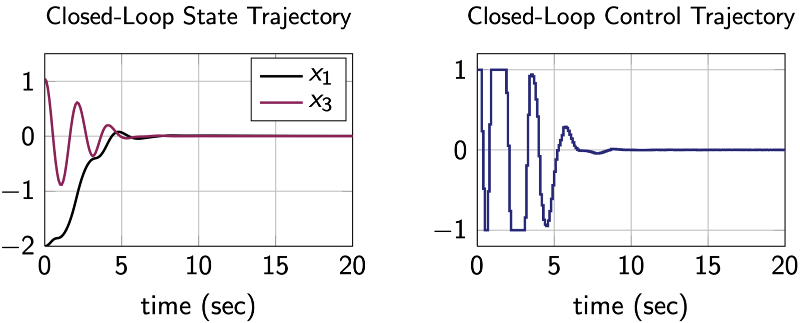
$$
\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= u \qquad u \in [-1, +1] \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -g \sin x_3 - u \cos x_3 - b x_4
\end{aligned}
$$

Cost

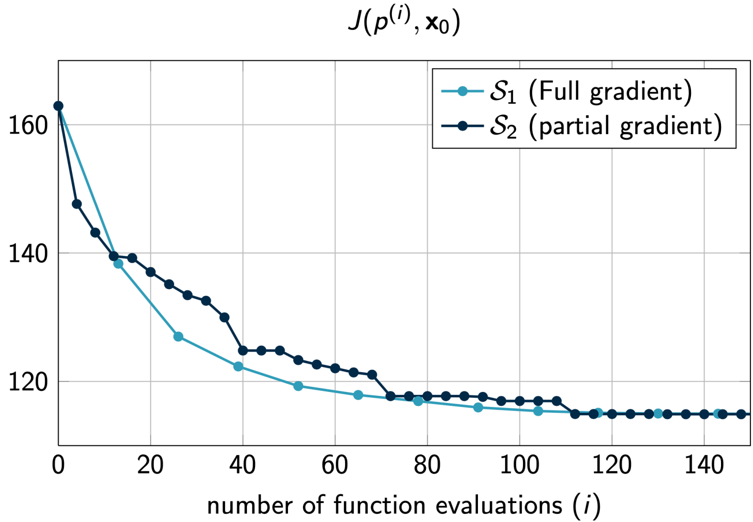
$$
\sum_{i=1}^{N_p} \|\mathbf{x}_{k+i}^T\|_Q^2 + \|\mathbf{u}_{k+i-1}^T\|_R^2
$$

- $N_p = 10$ and $T_p = 3\ sec$
- Two solvers:
  - $\mathcal{S}_1$ Full gradient
  - $\mathcal{S}_2$ Partial gradient
- $\tau_1^{\mathcal{S}_2} = [\frac{4}{13}] \times \tau_1^{\mathcal{S}_1}$
- $\tau_u(\mathcal{S}_2) = [\frac{4}{13}] \times \tau_u(\mathcal{S}_1)$
- $\tau_u(\mathcal{S}_1) \in \{5, 50, 100\}\ ms$

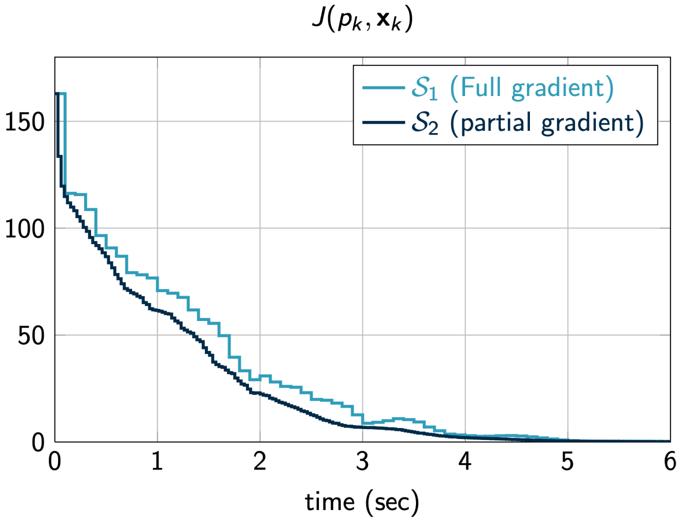
Typical Closed-Loop Behavior



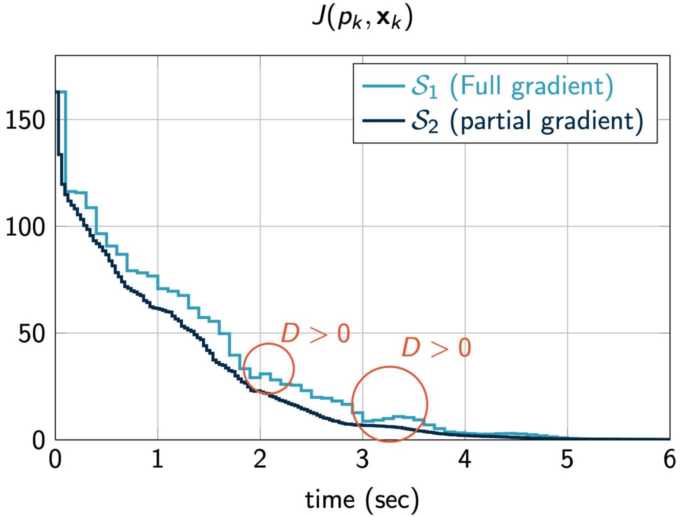
$\mathcal{S}_1$ is more efficient than $\mathcal{S}_2$ . . . **per iteration** !



Closed-Loop Evolutions of the Predicted Cost



Closed-Loop Evolutions of the Predicted Cost



Comparison of Effective Closed-Loop Cost

$$J_{cl} := \frac{1}{N_{sim}}\sum_{k=1}^{N_{sim}}\Big[\|\mathbf{x}_k\|_Q^2 + \|u_{k-1}\|_R^2\Big]$$

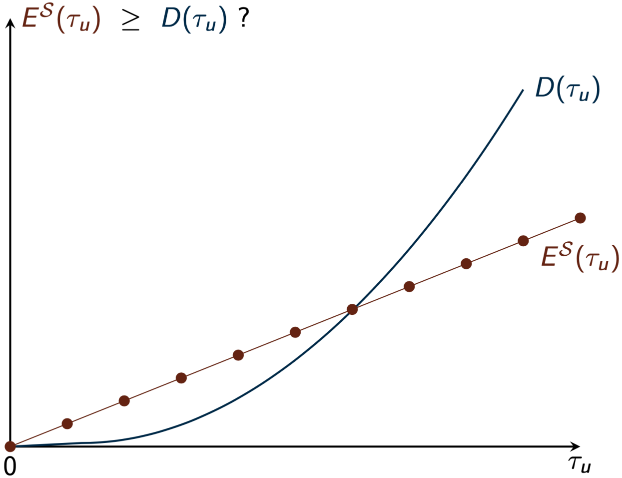
$\tau_u(\mathcal{S}_1)$	100 ms	50 ms	5 ms
Full Gradient	0.728	0.430	0.027
Partial Gradient	0.300	0.110	0.008
Gain %	59%	74%	70%

Updating Scheme For a Given Solver

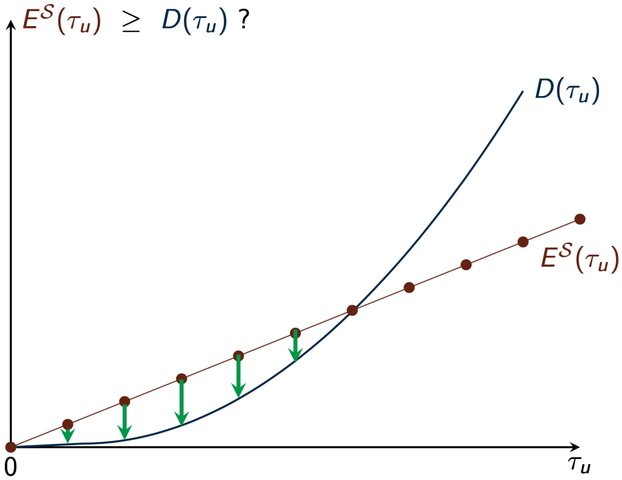
Assume that a solver $\mathcal{S}$ has been chosen . . .

Is there any remaining choice ?

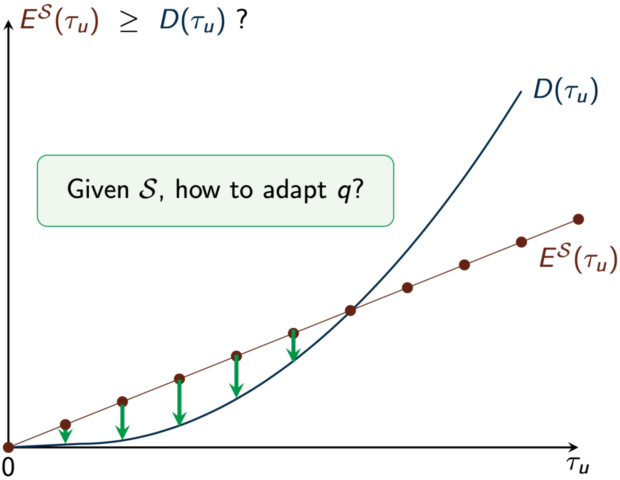
What is the optimal τ_u for a given solver?



What is the optimal τ_u for a given solver?



What is the optimal τ_u for a given solver?



Given $\mathcal{S}$, how to adapt q?

Updating τ_u is a control problem . . . !

$$\boxed{p_{k+1} \quad = \quad \mathcal{S}^{(q(\tau_u))}\big({p_k}^+, \mathbf{x}_k\big)}$$

Updating τ_u is a control problem . . . !

$$\boxed{\begin{aligned} p_{k+1} &= \mathcal{S}^{(q(\tau_u))}\left(p_k{}^+, \mathbf{x}_k\right)\big) \\ \mathbf{x}_{k+1} &= f\left(\mathbf{x}_k, \mathcal{U}(0, p_k)\right) \end{aligned}}$$

Updating τ_u is a control problem . . . !

$$\boxed{\begin{pmatrix} p_{k+1} \\ \mathbf{x}_{k+1} \end{pmatrix} = F\left(\begin{pmatrix} p_k \\ \mathbf{x}_k \end{pmatrix}, \tau_u\right)}$$

Updating τ_u is a control problem . . . !

$$\boxed{\begin{aligned}\begin{pmatrix}p_{k+1}\\ \mathbf{x}_{k+1}\end{pmatrix} &= F\left(\begin{pmatrix}p_k\\ \mathbf{x}_k\end{pmatrix},\tau_u\right)\\ y &= J(p_k,\mathbf{x}_k)\end{aligned}}$$

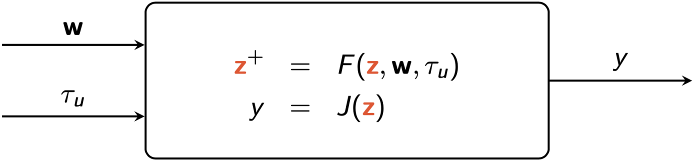
Updating τ_u is a control problem . . . !

$$\boxed{\begin{aligned} \mathbf{z}^+ &= F(\mathbf{z}, \tau_u) \\ y &= J(\mathbf{z}) \end{aligned}}$$

Updating τ_u is a control problem . . . !

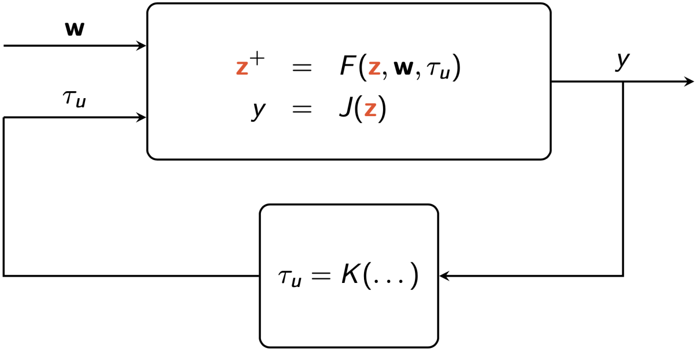
$$\boxed{\begin{aligned} \mathbf{z}^+ &= F(\mathbf{z}, \mathbf{w}, \tau_u) \\ y &= J(\mathbf{z}) \end{aligned}}$$

Updating τ_u is a control problem . . . !



$$\begin{aligned} \mathbf{z}^+ &= F(\mathbf{z}, \mathbf{w}, \tau_u) \\ y &= J(\mathbf{z}) \end{aligned}$$

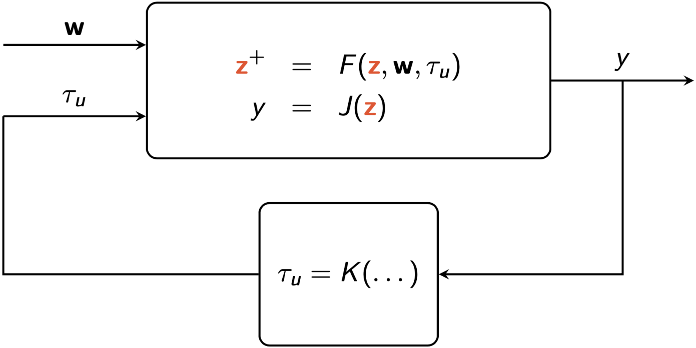
Updating τ_u is a control problem . . . !



$$\begin{aligned} \mathbf{z}^+ &= F(\mathbf{z}, \mathbf{w}, \tau_u) \\ y &= J(\mathbf{z}) \end{aligned}$$

$$\tau_u = K(\dots)$$

Updating τ_u is a control problem . . . !

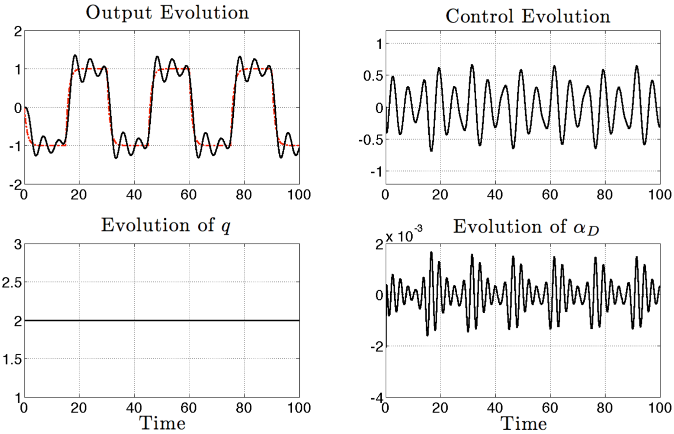


$$\begin{aligned} \mathbf{z}^+ &= F(\mathbf{z}, \mathbf{w}, \tau_u) \\ y &= J(\mathbf{z}) \end{aligned}$$

$$\tau_u = K(\dots)$$

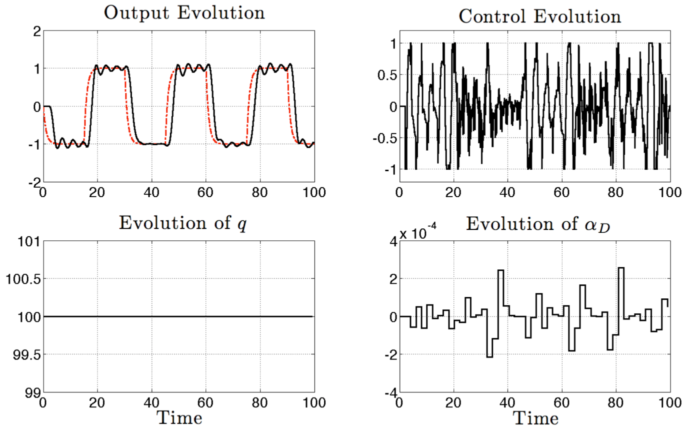
M.A. ECC (2013)

$q = 2$ without adaptation



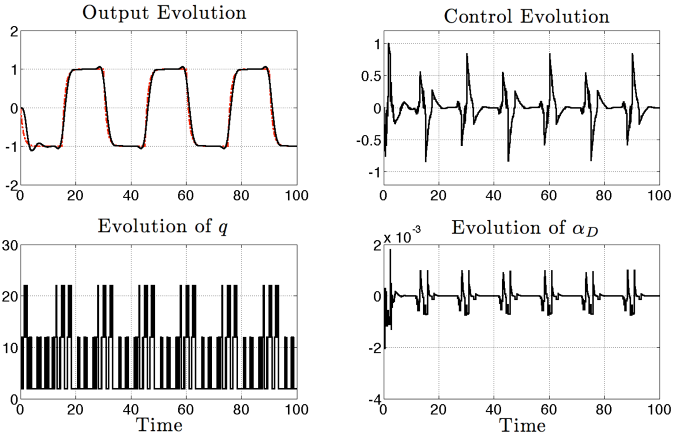
MA, ECC (2013)

$q = 100$ without adaptation



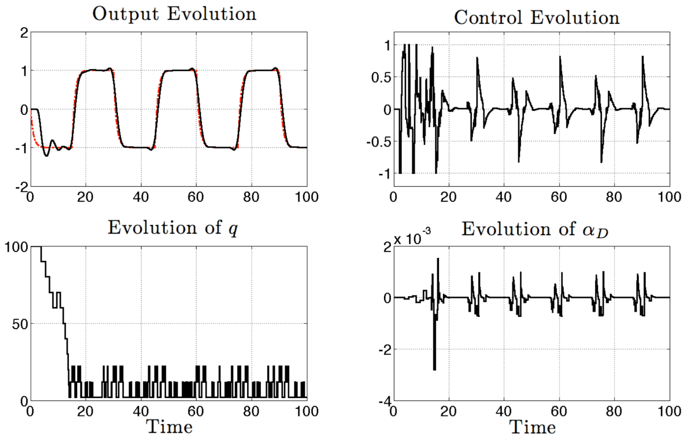
MA, ECC (2013)

$q^{(0)} = 2$ with adaptation



MA, ECC (2013)

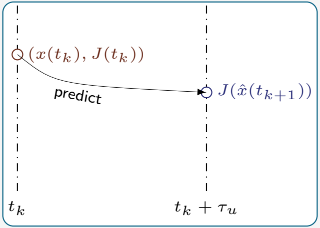
$q^{(0)} = 100$ with adaptation



MA, ECC (2013)

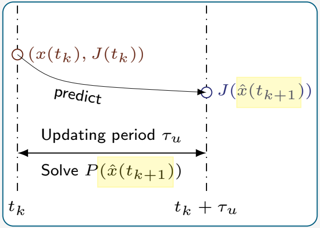
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)



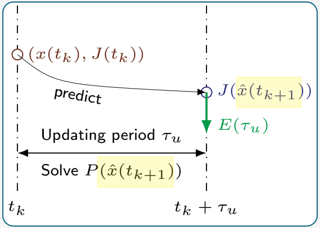
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)



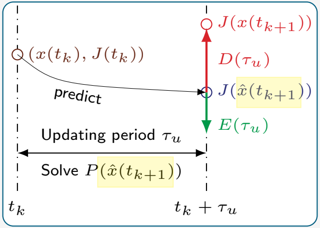
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)



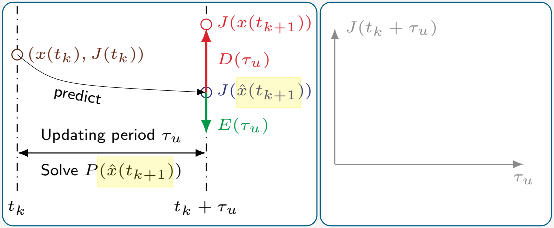
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)



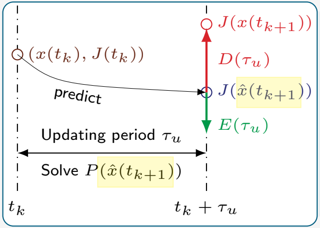
MONITORING CONTROL UPDATING PERIOD

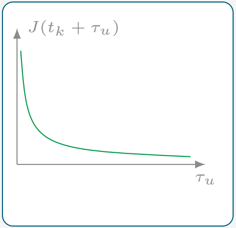
New ML-induced opportunities (preliminary)



MONITORING CONTROL UPDATING PERIOD

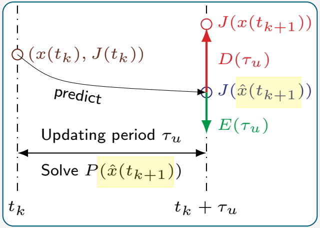
New ML-induced opportunities (preliminary)

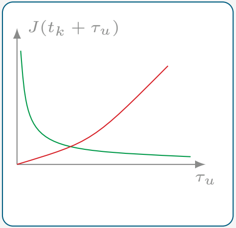




MONITORING CONTROL UPDATING PERIOD

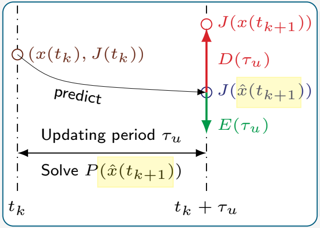
New ML-induced opportunities (preliminary)

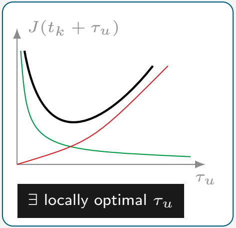




MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)

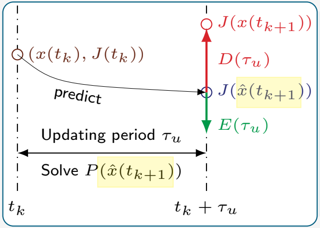


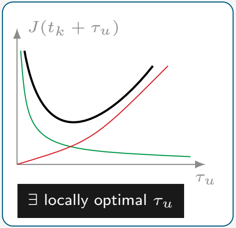


∃ locally optimal τ_u

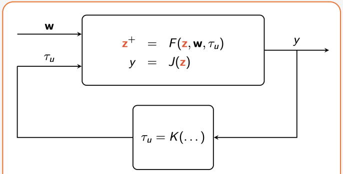
Monitoring Control Updating Period

New ML-induced opportunities (preliminary)





∃ locally optimal τ_u

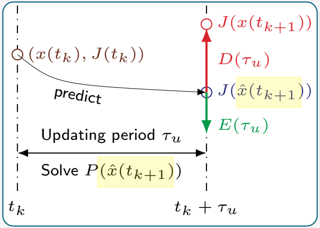


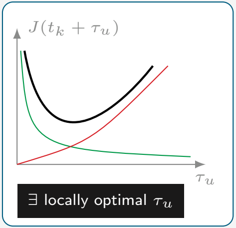
Complexity: (5±, 5×, 5÷ and 1 log)

M.A. Monitoring Control Updating Period In Fast Gradient-Based NMPC. ECC2013, 2013

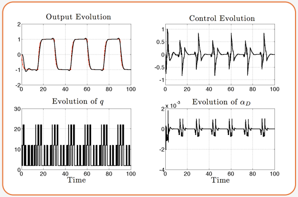
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)





∃ locally optimal τ_u

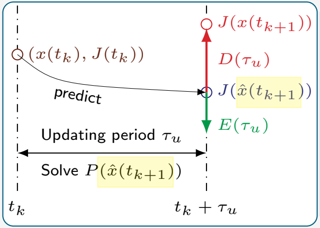


$\tau_u := q \times$ cpu of a single iteration

M.A. Monitoring Control Updating Period In Fast Gradient-Based NMPC. ECC2013, 2013

MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)



$(x(t_k), J(t_k))$

predict

$J(x(t_{k+1}))$

$D(\tau_u)$

$J(\hat{x}(t_{k+1}))$

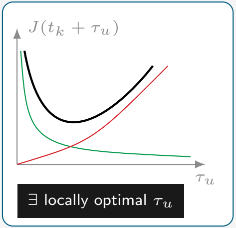
$E(\tau_u)$

Updating period τ_u

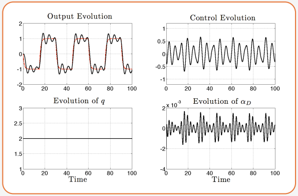
Solve $P(\hat{x}(t_{k+1}))$

t_k

$t_k + \tau_u$



$\exists$ locally optimal τ_u

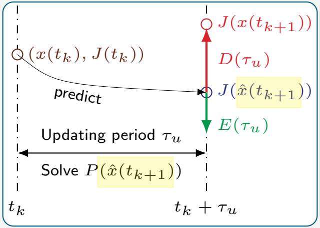


$\tau_u := q \times$ cpu of a single iteration

M.A. Monitoring Control Updating Period In Fast Gradient-Based NMPC. ECC2013, 2013

MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)



$(x(t_k), J(t_k))$

predict

$J(x(t_{k+1}))$

$D(\tau_u)$

$J(\hat{x}(t_{k+1}))$

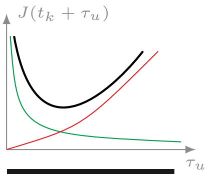
$E(\tau_u)$

Updating period τ_u

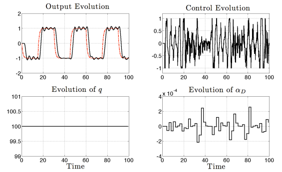
Solve $P(\hat{x}(t_{k+1}))$

t_k

$t_k + \tau_u$



∃ locally optimal τ_u

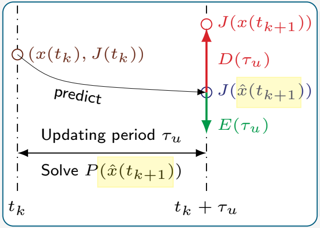


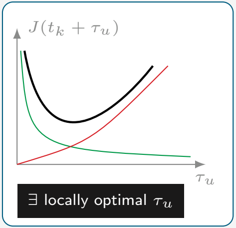
$\tau_u := q \times$ cpu of a single iteration

M.A. Monitoring Control Updating Period In Fast Gradient-Based NMPC. ECC2013, 2013

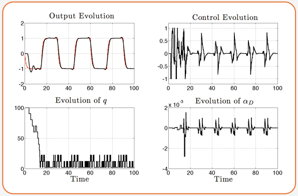
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)





∃ locally optimal τ_u

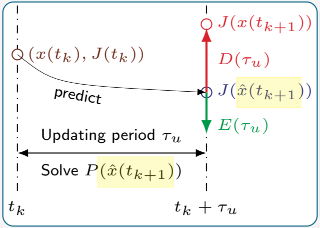


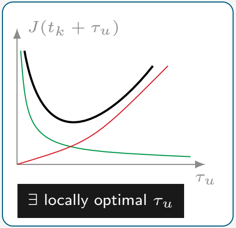
$\tau_u := q \times$ cpu of a single iteration

M.A. Monitoring Control Updating Period In Fast Gradient-Based NMPC. ECC2013, 2013

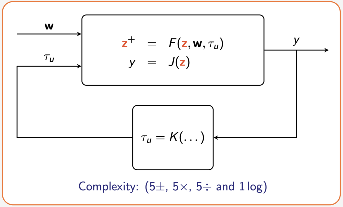
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)

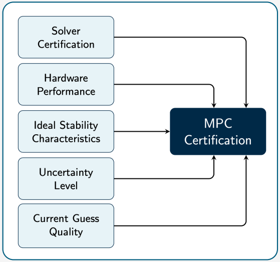




∃ locally optimal τ_u



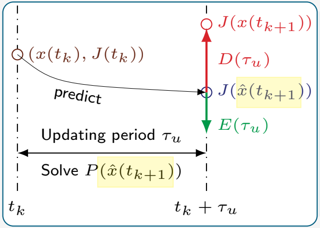
Complexity: (5±, 5×, 5÷ and 1 log)

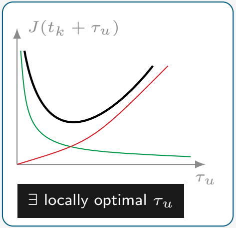


M. Alamir. A State-Dependent Updating Period For Certified Real-Time MPC. IEEE TAC, 2017.

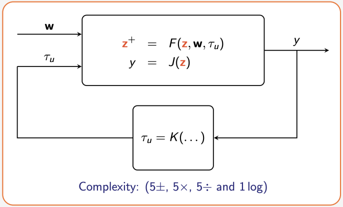
Monitoring Control Updating Period

New ML-induced opportunities (preliminary)

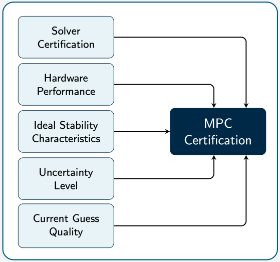


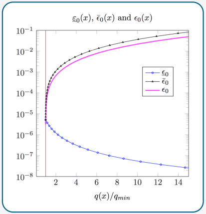


∃ locally optimal τ_u



Complexity: (5±, 5×, 5÷ and 1 log)





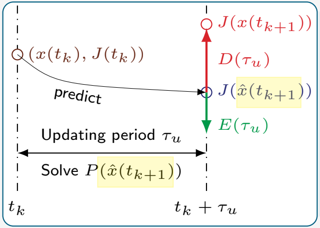
ϵ_0: required precision

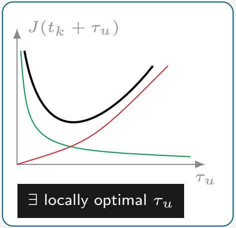
$q(x) \leftarrow$ stage cost

M. Alamir. A State-Dependent Updating Period For Certified Real-Time MPC. IEEE TAC, 2017.

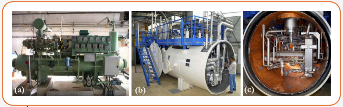
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)

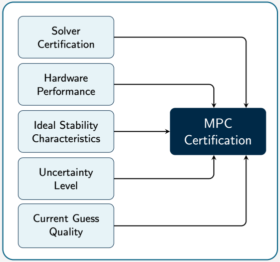


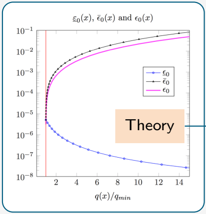


$\exists$ locally optimal τ_u



Simulate CL-NMPC
Fast Gradient
Warm start
Random (τ_u, w)


Solver Certification
Hardware Performance
Ideal Stability Characteristics
Uncertainty Level
Current Guess Quality
MPC Certification



Theory

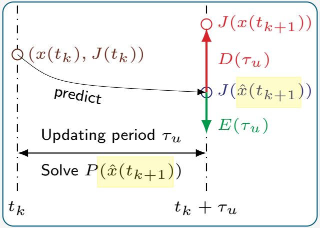
check

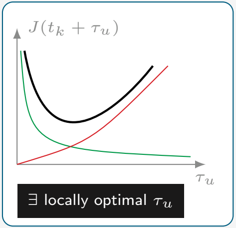
ϵ_0: required precision
$q(x) \leftarrow$ stage cost

M. Alamir. A State-Dependent Updating Period For Certified Real-Time MPC. IEEE TAC, 2017.

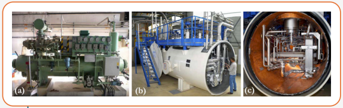
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)





∃ locally optimal τ_u

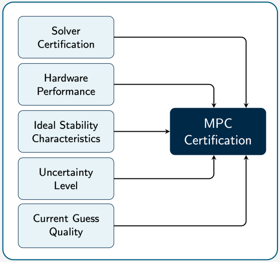


Simulate CL-NMPC
Fast Gradient
Warm start
Random (τ_u, w)

Learning data:

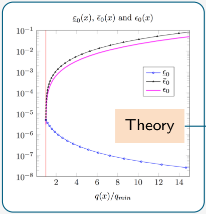
$$X = (J, \tau_u, w)$$

$$y = \frac{J^+}{J+1}$$



Solver Certification
Hardware Performance
Ideal Stability Characteristics
Uncertainty Level
Current Guess Quality

MPC Certification



Theory

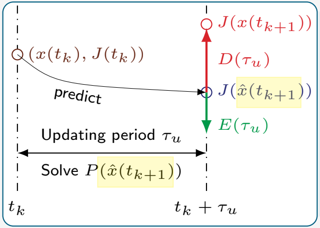
check

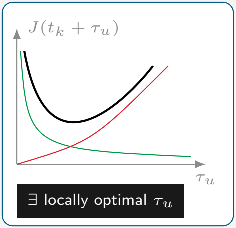
ϵ_0: required precision
$q(x) \leftarrow$ stage cost

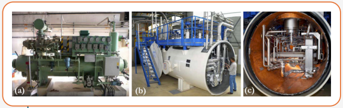
M. Alamir. A State-Dependent Updating Period For Certified Real-Time MPC. IEEE TAC, 2017.

MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)





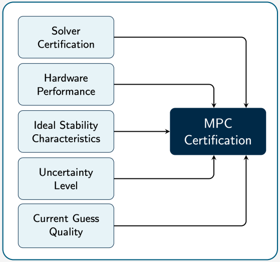


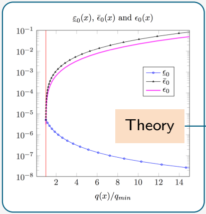
- Simulate CL-NMPC
- Fast Gradient
- Warm start
- Random (τ_u, w)

Learning data:

$$X = (J, \tau_u, w)$$

$$y = \frac{J^+}{J+1}$$

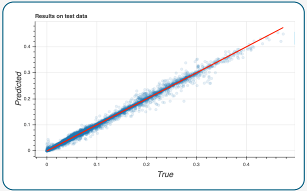




ϵ_0: required precision

$q(x) \leftarrow$ stage cost

check

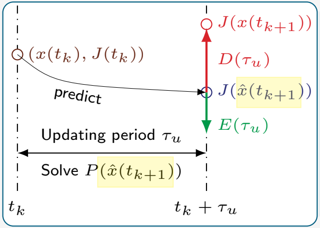


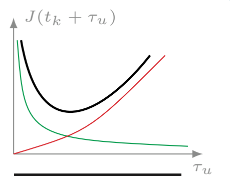
$J^+ = \text{ML}(J(x), \tau_u, w)$

M. Alamir. A State-Dependent Updating Period For Certified Real-Time MPC. IEEE TAC, 2017.

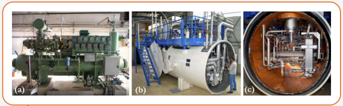
MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)





∃ locally optimal τ_u

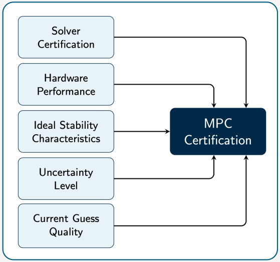


Simulate CL-NMPC
Fast Gradient
Warm start
Random (τ_u, w)

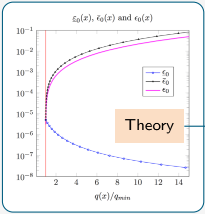
Learning data:

$$X = (J, \tau_u, w)$$

$$y = \frac{J^+}{J+1}$$



Solver Certification
Hardware Performance
Ideal Stability Characteristics
Uncertainty Level
Current Guess Quality
MPC Certification



Theory

check

ϵ_0: required precision
$q(x) \leftarrow$ stage cost

$$J(t_{k+1}) = \mathbf{ML}(J(t_k), \tau_u, w)$$
$$t_{k+1} = t_k + \tau_u$$

$$\min_{\tau_u(\cdot)} \mathbb{E}(J(T_P)) \text{ s.t. } \begin{cases} \tau_u^{(i)} > 0 \\ \sum_i \tau_u^{(i)} = T_p \end{cases}$$

M. Alamir. A State-Dependent Updating Period For Certified Real-Time MPC. IEEE TAC, 2017.

MONITORING CONTROL UPDATING PERIOD

New ML-induced opportunities (preliminary)

$$J(t_{k+1}) = \textbf{ML}(J(t_k), \tau_u, w)$$
$$t_{k+1} = t_k + \tau_u$$

$$\min_{\tau_u(\cdot)} \mathbb{E}(J(T_P)) \text{ s.t. } \begin{cases} \tau_u^{(i)} > 0 \\ \sum_i \tau_u^{(i)} = T_p \end{cases}$$

Approximate Stochastic Dynamic Programming

$$\tau_u(t_k) := \tau^\star(J(t_k) \mid \mathcal{W})$$

Option3 (Approximate Dynamic Programming)

Approximate the optimal value

$$Q(z) \quad \text{where} \quad z := (x, u)$$

based on the Bellman equation

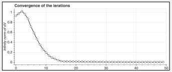
$$Q(z) = \ell(z) + \gamma \min_v [\hat{\mu} + \alpha\hat{\sigma}^{\frac{1}{2}}][Q(f(z), v)]$$

Fixed-point

- ℓ stage cost
- γ discounting rat
- $\hat{\mu}$ approximated statistics over realisations of w

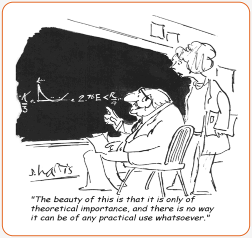
(-) Curse of dimensionality
(-) Fragile choice of the support state domain

- Choose a **ML**-based structure for Q
- Fixed-point iteration on Q



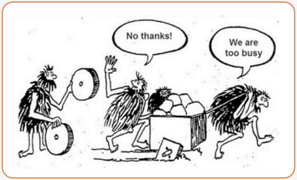
[MA]. Explicit approximation of stochastic optimal control for combined therapy of cancer. arXiv:1905.04937, 2019.

CONCLUSION

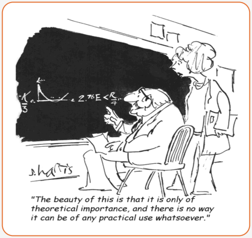


"The beauty of this is that it is only of theoretical importance, and there is no way it can be of any practical use whatsoever."

US



CONCLUSION

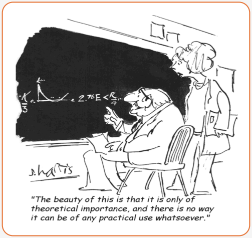


"The beauty of this is that it is only of theoretical importance, and there is no way it can be of any practical use whatsoever."

US

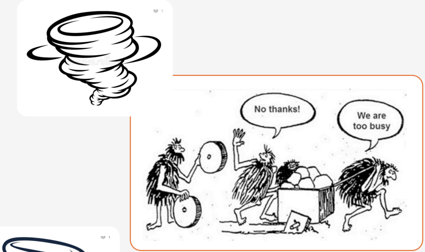
Machine Learning

CONCLUSION



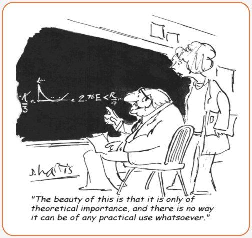
US

Machine Learning



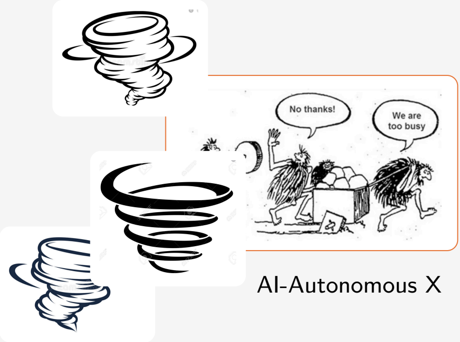
Computational technologies
GPU, FPGA, Containers, . . .

CONCLUSION



US

Machine Learning

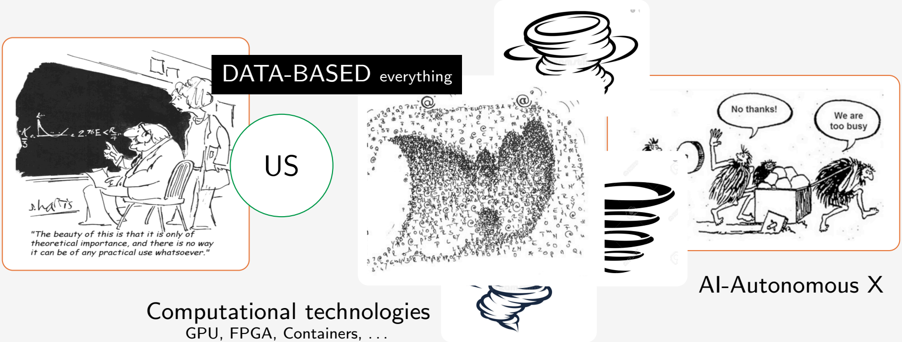


AI-Autonomous X

Computational technologies

GPU, FPGA, Containers, . . .

CONCLUSION



Machine Learning

DATA-BASED everything

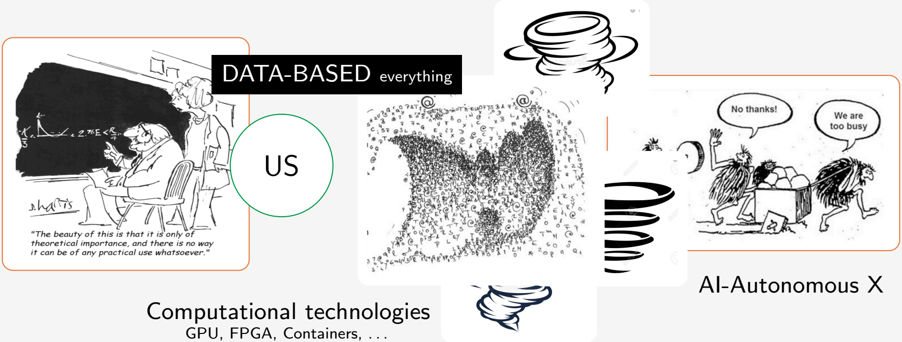
US

AI-Autonomous X

Computational technologies
GPU, FPGA, Containers, . . .

Conclusion

Business as usual is not an option!



Machine Learning

DATA-BASED everything

US

Computational technologies
GPU, FPGA, Containers, . . .

AI-Autonomous X