A small step towards a systematic design of effective and implementable NMPC setting

The Python-package MPC_tuner

Mazen Alamir

CNRS / University of Grenoble Alpes / Gipsa-lab

Outline

Recalls on Nonlinear Model Predictive Control (NMPC)

Real-Time considerations

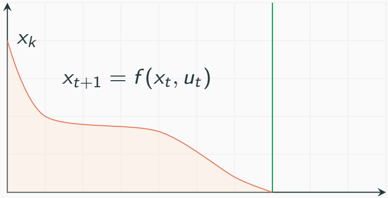
Design of NMPC setting

The `MPC_tuner` package

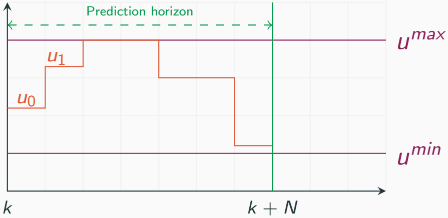
Conclusion and future extensions

Recalls on Nonlinear Model Predictive Control (NMPC)

Predicted state profile $x(\cdot)$



Candidate control profile $\mathbf{u}_k$



$\forall$ candidate control sequence

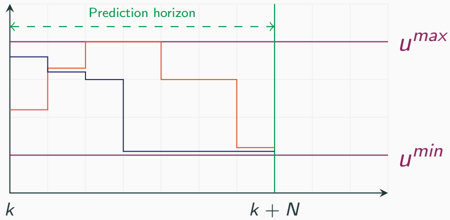
$$\boldsymbol{u}_k := (u_0, \dots, u_{N-1})$$

a cost can be predicted

$$J(\boldsymbol{u}_k \mid x_k)$$

Predicted state profile $x(\cdot)$

Candidate control profile $\mathbf{u}_k$

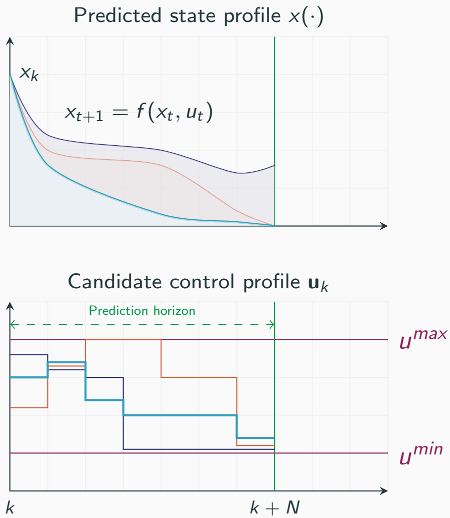


$\forall$ candidate control sequence

$$\boldsymbol{u}_k := (u_0, \dots, u_{N-1})$$

a cost can be predicted

$$J(\boldsymbol{u}_k \mid x_k)$$



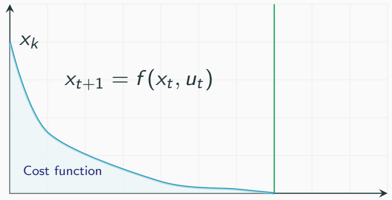
$\forall$ candidate control sequence

$$\boldsymbol{u}_k := (u_0, \dots, u_{N-1})$$

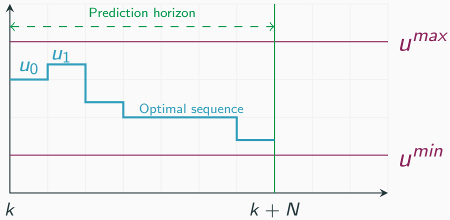
a cost can be predicted

$$J(\boldsymbol{u}_k \mid x_k)$$

Predicted state profile $x(\cdot)$



Candidate control profile $\mathbf{u}_k$



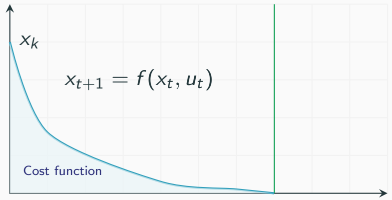
$\forall$ candidate control sequence

$$\boldsymbol{u}_k := (u_0, \dots, u_{N-1})$$

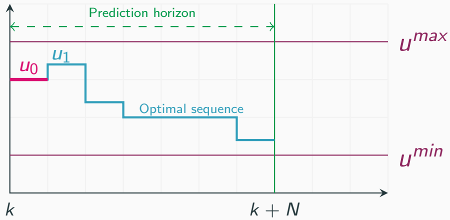
a cost can be predicted

$$J(\boldsymbol{u}_k \mid x_k)$$

Predicted state profile $x(\cdot)$



Candidate control profile $\mathbf{u}_k$



$\forall$ candidate control sequence

$$\boldsymbol{u}_k := (u_0, \dots, u_{N-1})$$

a cost can be predicted

$$J(\boldsymbol{u}_k \mid x_k)$$

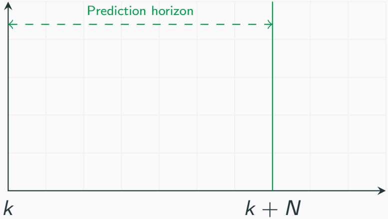
MPC feedback

$$u_k = u_0^\star(x_k) \quad \text{where}$$

$$J(\boldsymbol{u}^\star \mid x_k) = \min_{\boldsymbol{u} \in \mathbb{U}(x)} J(\boldsymbol{u} \mid x_k)$$

Predicted state profile $x(\cdot)$

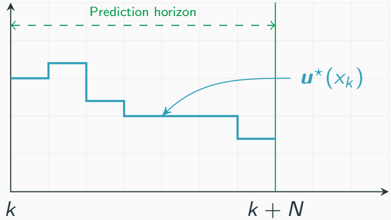
Control profile $u(\cdot)$

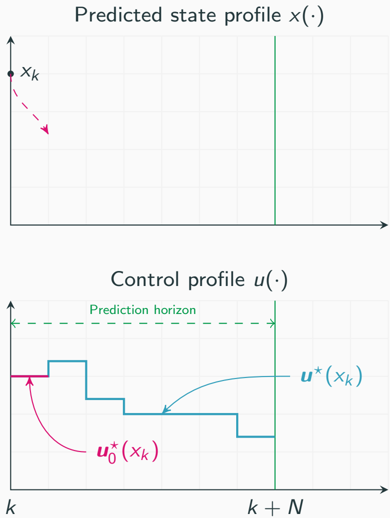


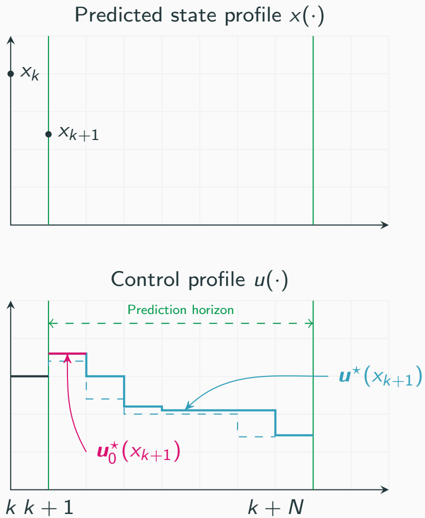
NMPC ← The Receding-horizon principle in action

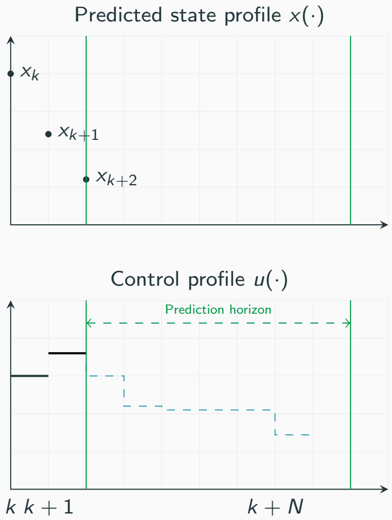
Predicted state profile $x(\cdot)$

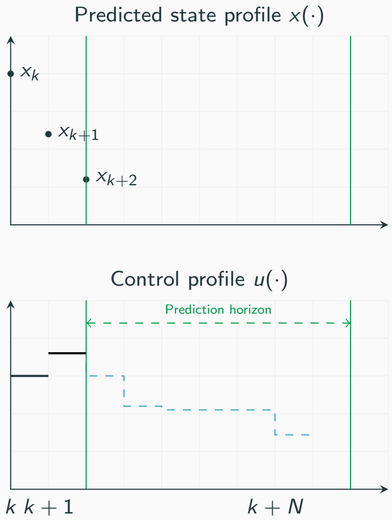
Control profile $u(\cdot)$




Predicted state profile $x(\cdot)$
x_k
Control profile $u(\cdot)$
Prediction horizon
$\boldsymbol{u}^\star(x_k)$
$\boldsymbol{u}_0^\star(x_k)$
k
$k+N$

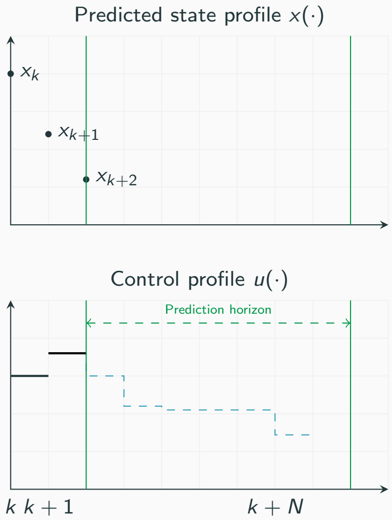

Predicted state profile $x(\cdot)$
x_k
x_{k+1}
Control profile $u(\cdot)$
Prediction horizon
$\boldsymbol{u}^\star(x_{k+1})$
$\boldsymbol{u}_0^\star(x_{k+1})$
k $k+1$
$k+N$


Predicted state profile $x(\cdot)$
x_k
x_{k+1}
x_{k+2}
Control profile $u(\cdot)$
Prediction horizon
k $k+1$
$k+N$



- ✓ At each **updating instant**
- ✓ The prediction horizon is shifted
- ✓ **A new optimization problem is defined**
- ✓ and **solved**
- ✓ The first action is applied
- ✓ until the next **updating instant**
- ✓ The process is repeated
- ✓ ...

NMPC ← The Receding-horizon principle in action

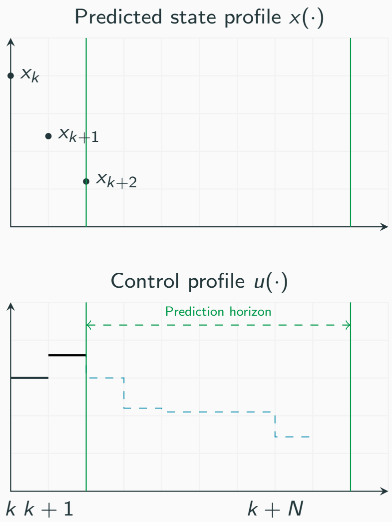


- ✓ At each **updating instant**
- ✓ The prediction horizon is shifted
- ✓ **A new optimization problem is defined**
- ✓ and **solved**
- ✓ The first action is applied
- ✓ until the next **updating instant**
- ✓ The process is repeated
- ✓ ...

--- REAL-TIME AGNOSTIC DEFINITION ---

No reference to computation time!

NMPC ← The Receding-horizon principle in action



- ✓ At each **updating instant**
- ✓ The prediction horizon is shifted
- ✓ **A new optimization problem is defined**
- ✓ and **solved**
- ✓ The first action is applied
- ✓ until the next **updating instant**
- ✓ The process is repeated
- ✓ ...

--- REAL-TIME AGNOSTIC DEFINITION ---

No reference to computation time!

We'll come back to this later...
let's first examine the optimization problem!

NMPC $\leftarrow$ The formulations

MPC feedback

$$u_k = u_0^\star(x_k) \qquad \text{where}$$

$$J(\boldsymbol{u}^\star \mid x_k) = \min_{\boldsymbol{u}\in\mathbb{U}(x)} J(\boldsymbol{u} \mid x_k)$$

More precisely . . .

Automatica
Volume 36, Issue 6, June 2000, Pages 789-814

Survey Paper

Constrained model predictive control: Stability and optimality ☆

D.Q. Mayne [a], J.B. Rawlings [b], C.V. Rao [b], P.O.M. Scokaert [c]

$$\min_{\boldsymbol{u}}\left[\Psi(x_{k+N})+\sum_{i=1}^{N}\ell(x_{k+i},u_{k+i-1})\right]$$

$$\text{under}\left|\begin{array}{l} x_{k+N}\in\mathbb{X}_f \\ x_{k+i}\in\mathbb{X} \\ u_{k+i-1}\in\mathbb{U}\end{array}\right.$$

† where Ψ is a control-Lyapunov-function inside $\mathbb{X}_f$.

NMPC ← The formulations

ELSEVIER

Automatica
Volume 36, Issue 6, June 2000, Pages 789-814

Survey Paper

Constrained model predictive control: Stability and optimality ☆

D.Q. Mayne [a], J.B. Rawlings [b], C.V. Rao [b], P.O.M. Scokaert [c]

$$\min_{\boldsymbol{u}}\left[\Psi(x_{k+N})+\sum_{i=1}^{N}\ell(x_{k+i},u_{k+i-1})\right]$$

$$\text{under}\left|\begin{array}{l} x_{k+N}\in\mathbb{X}_f \\ x_{k+i}\in\mathbb{X} \\ u_{k+i-1}\in\mathbb{U}\end{array}\right.$$

† where Ψ is a control-Lyapunov-function inside $\mathbb{X}_f$.

- ✗ $(\Psi,\mathbb{X}_f)$ almost impossible to compute for real-life systems
- ✗ Never used outside academic toy examples!

NMPC ← The **formulations**

ELSEVIER

Automatica
Volume 36, Issue 6, June 2000, Pages 789-814

Survey Paper

Constrained model predictive control: Stability and optimality ☆

D.Q. Mayne [a], J.B. Rawlings [b], C.V. Rao [b], P.O.M. Scokaert [c]

$$\min_{u} \left[\Psi(x_{k+N}) + \sum_{i=1}^{N} \ell(x_{k+i}, u_{k+i-1}) \right]$$

$$\text{under} \left| \begin{array}{l} x_{k+N} \in \mathbb{X}_f \\ x_{k+i} \in \mathbb{X} \\ u_{k+i-1} \in \mathbb{U} \end{array} \right.$$

† where Ψ is a control-Lyapunov-function inside $\mathbb{X}_f$.

- ✗ $(\Psi, \mathbb{X}_f)$ almost impossible to compute for real-life systems
- ✗ Never used outside academic toy examples!

Don't even try!

NMPC $\leftarrow$ The formulations

Automatica 75 (2017) 288–292

Contents lists available at ScienceDirect

Automatica

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

Brief paper

Contraction-based nonlinear model predictive control formulation without stability-related terminal constraints☆

Mazen Alamir

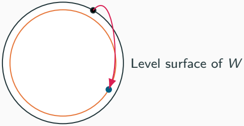
CNRS/Gipsa-lab, University of Grenoble Alpes, France

$$\min_{(\boldsymbol{u},q)} \Big[\alpha \min_{i=1}^{q} W(x_{k+\ell}) + z\sum_{i=1}^{q} \ell(x_{k+i}, u_{k+i-1})\Big]$$

$$\text{under} \left| \begin{array}{l} (\boldsymbol{u}, q) \in \mathbb{U}^q \times \{1,\dots,N\} \\ x_{k+\ell} \in \mathbb{X} \end{array}\right.$$

where z is an internal state of the controller

- ✓ **Contractive formulations**
- ✓ Prediction horizon is a decision variable.



Accept transient increase of the cost function

$\rightarrow$ multi-step Lyapunov function

NMPC ← The formulations

Automatica 87 (2018) 455–459

Contents lists available at ScienceDirect

Automatica

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

Technical communique

Stability proof for nonlinear MPC design using monotonically increasing weighting profiles without terminal constraints☆

Mazen Alamir

CNRS, Univ. Grenoble Alpes, Gipsa-lab, F-3800, Grenoble, France

$$\min_{\boldsymbol{u}} \left[\sum_{i=1}^{N} [i/N]^m \ell(x_{k+i}, u_{k+i-1})\right]$$

$$\text{under} \left| \begin{array}{l} \boldsymbol{u} \in \mathbb{U}^N \\ x_{k+\ell} \in \mathbb{X} \end{array} \right.$$

where $m \in \mathbb{N}$

✓ Enhances stability with short prediction horizons

→ a smoother version o f the terminal state constraints

NMPC $\leftarrow$ The formulations

Automatica 125 (2021) 109420

Contents lists available at ScienceDirect

Automatica

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

ELSEVIER

Brief paper

A new formulation of Economic Model Predictive Control without terminal constraint☆

Mazen Alamir[a,*], Gabriele Pannocchia[b]

[a] Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France
[b] Department of Civil and Industrial Engineering, University of Pisa, 56126 Pisa, Italy

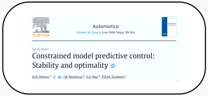
$$\min_{\boldsymbol{u}} \gamma\Big[\ell_{k+N} + \rho_d \Delta_{k+N}\Big] + \sum_{i=1}^{N}\Big[\ell_{k+i} + \rho_d \Delta_{k+i}\Big]$$

$$\text{under} \left| \begin{array}{l} \boldsymbol{u} \in \mathbb{U}^N \\ x_{k+\ell} \in \mathbb{X} \end{array} \right.$$

$$\text{where } \Delta_k := \|x_k - x_{k-1}\|$$

- ✓ Economic MPC
- ✓ Standard proof of stability

Does the formulation matter?


ELSEVIER

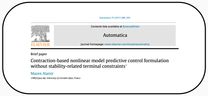
Automatica

Volume 36, Issue 6, June 2000, Pages 789-814

Survey Paper

Constrained model predictive control: Stability and optimality ☆

D.Q. Mayne[a], J.B. Rawlings[b], C.V. Rao[b], P.O.M. Scokaert[c]


Automatica 75 (2017) 288–292

ELSEVIER

Contents lists available at ScienceDirect

Automatica

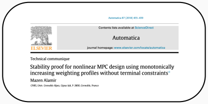
journal homepage: www.elsevier.com/locate/automatica

Brief paper

Contraction-based nonlinear model predictive control formulation without stability-related terminal constraints☆

Mazen Alamir

CNRS/Gipsa-lab, University of Grenoble Alpes, France


Automatica 87 (2018) 455–459

ELSEVIER

Contents lists available at ScienceDirect

Automatica

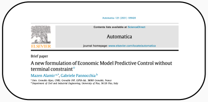
journal homepage: www.elsevier.com/locate/automatica

Technical communique

Stability proof for nonlinear MPC design using monotonically increasing weighting profiles without terminal constraints☆

Mazen Alamir

CNRS, Univ. Grenoble Alpes, Gipsa-lab, F-3800, Grenoble, France


Automatica 125 (2021) 109420

ELSEVIER

Contents lists available at ScienceDirect

Automatica

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

Brief paper

A new formulation of Economic Model Predictive Control without terminal constraint☆

Mazen Alamir[a,*], Gabriele Pannocchia[b]

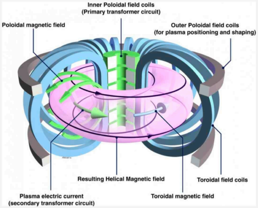
[a] Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France

[b] Department of Civil and Industrial Engineering, University of Pisa, 56126 Pisa, Italy

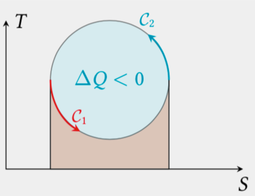
Question: **Does the formulation really matter?**

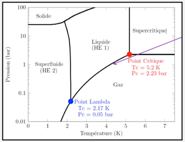
The impact of formulations!

Cryogenic Refrigerators - Problem statement



Source: https://www.euro-fusion.org





Why?

Provide refrigeration capacity to cool down the supra-conducting coils used to accelerate the plasma in Nuclear Fusion Reactors (ITER, JT60)

How?

Force a thermodynamic fluid to make a counter-clock cycle in the (S, T)=(Entropy,Temperature) plan.

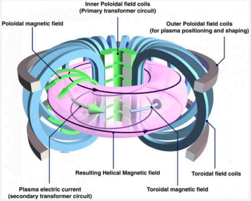
$$\int dQ = \underbrace{\int_{\mathcal{C}_1} TdS}_{>0} + \underbrace{\int_{\mathcal{C}_2} TdS}_{<<0}$$

F. Bonne (CEA)

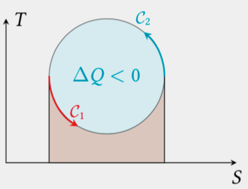
P. Bonnay (CEA)

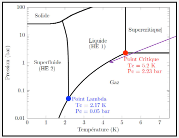
The impact of formulations!

Cryogenic Refrigerators - Problem statement



Source: https://www.euro-fusion.org





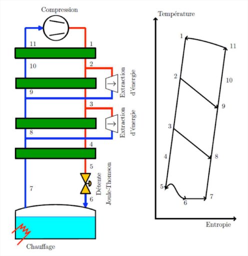
Why?

Provide refrigeration capacity to cool down the supra-conducting coils used to accelerate the plasma in Nuclear Fusion Reactors (ITER, JT60)

How?

Force a thermodynamic fluid to make a counter-clock cycle in the (S,T)=(Entropy,Temperature) plan.

$$\int dQ = \underbrace{\int_{\mathcal{C}_1} TdS}_{>0} + \underbrace{\int_{\mathcal{C}_2} TdS}_{<<0}$$

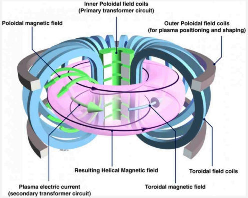


F. Bonne (CEA)

P. Bonnay (CEA)

The impact of formulations!

Cryogenic Refrigerators - Problem statement

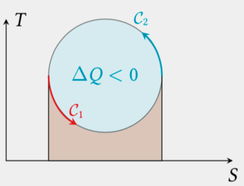


Source: https://www.euro-fusion.org

Why?

Provide refrigeration capacity to cool down the supra-conducting coils used to accelerate the plasma in Nuclear Fusion Reactors (ITER, JT60)

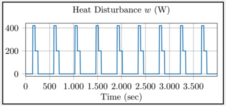
F. Bonne (CEA) P. Bonnay (CEA)



How?

Force a thermodynamic fluid to make a counter-clock cycle in the (S, T)=(Entropy,Temperature) plan.

$$\int dQ = \underbrace{\int_{\mathcal{C}_1} TdS}_{>0} + \underbrace{\int_{\mathcal{C}_2} TdS}_{<<0}$$



Model

$$x^+ = Ax + Bu + Gw$$
$$y = Cx + Du$$

- $(x, u, w) \in \mathbb{R}^{24} \times \mathbb{R}^3 \times \mathbb{R}$
- w unpredictable heat pulse
- $u \in \mathbb{U}$ two valves+heating (bath)

Objective

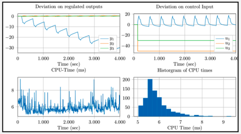
Keep the station sustainable despite of unmeasured and unpredictable w.

The impact of formulations!

Cryogenic refrigerator: The impact of formulation

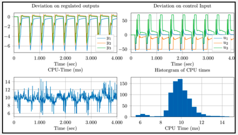
Cost function

$$\sum_{i=1}^{N}\Big[\|y_{k+i}\|^2_{Q_y}+\epsilon\|x_{k+i}\|^2\Big]$$



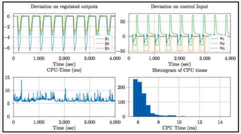
Cost function

$$\sum_{i=1}^{N}\Big[\|y_{k+i}\|^2_{Q_y}+\epsilon\|x_{k+i}\|^2\Big]+\epsilon_f\|x_{i+N}\|^2$$



Cost function

$$\sum_{i=1}^{N}\Big[\frac{i}{N}\Big]^m\Big[\|y_{k+i}\|^2_{Q_y}+\epsilon\|x_{k+i}\|^2\Big]$$



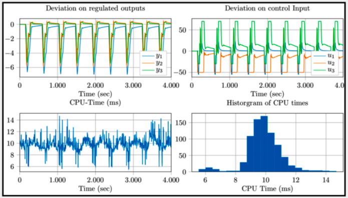
Optimization problems are solved using the GUROBI-Python-3 framework implemented on a Mac-PowerBook 2.9GHz (High Sierra OS version 10.13.3).

The impact of formulations!

Cryogenic refrigerator: The impact of formulation

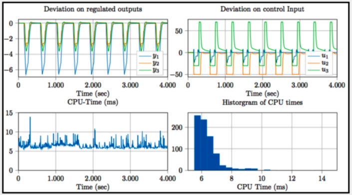
Cost function

$$\sum_{i=1}^{N}\Big[\|y_{k+i}\|^2_{Q_y}+\epsilon\|x_{k+i}\|^2\Big]+\epsilon_f\|x_{i+N}\|^2$$

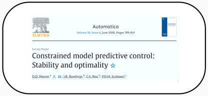


Cost function

$$\sum_{i=1}^{N}\Big[\frac{i}{N}\Big]^m\Big[\|y_{k+i}\|^2_{Q_y}+\epsilon\|x_{k+i}\|^2\Big]$$



Does the formulation matter?

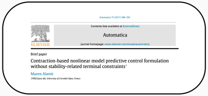


Automatica
Volume 36, Issue 6, June 2000, Pages 789–814

Survey Paper

Constrained model predictive control: Stability and optimality ☆

D.Q. Mayne [a], J.B. Rawlings [b], C.V. Rao [b], P.O.M. Scokaert [c]



Automatica 75 (2017) 288–292

Contents lists available at ScienceDirect

Automatica

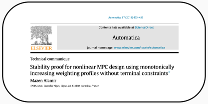
journal homepage: www.elsevier.com/locate/automatica

Brief paper

Contraction-based nonlinear model predictive control formulation without stability-related terminal constraints ☆

Mazen Alamir

CNRS/Gipsa-lab, University of Grenoble Alpes, France



Automatica 87 (2018) 455–459

Contents lists available at ScienceDirect

Automatica

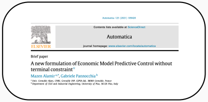
journal homepage: www.elsevier.com/locate/automatica

Technical communique

Stability proof for nonlinear MPC design using monotonically increasing weighting profiles without terminal constraints ☆

Mazen Alamir

CNRS, Univ. Grenoble Alpes, Gipsa-lab, F-3800, Grenoble, France



Automatica 125 (2021) 109420

Contents lists available at ScienceDirect

Automatica

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

Brief paper

A new formulation of Economic Model Predictive Control without terminal constraint ☆

Mazen Alamir [a,*], Gabriele Pannocchia [b]

[a] Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France
[b] Department of Civil and Industrial Engineering, University of Pisa, 56126 Pisa, Italy

QUESTION: **Does the formulation really matter?**

ANSWER: **Of course!**

A single parameterized formulation

Standard	$\left[\Psi(x_{k+N}) + \sum_{i=1}^{N} \ell_{k+i}\right]$	s.t $c_j(x_{k+i}, u_{k+i-1}) \le 0$
Economic	$\gamma\left[\ell_{k+N} + \rho_d \Delta_{k+N}\right] + \sum_{i=1}^{N}\left[\ell_{k+i} + \rho_d \Delta_{k+i}\right]$	s.t $c_j(x_{k+i}, u_{k+i-1}) \le 0$
Variable weight	$\left[\sum_{i=1}^{N} [i/N]^m \ell(x_{k+i}, u_{k+i-1})\right]$	s.t $c_j(x_{k+i}, u_{k+i-1}) \le 0$

A single parameterized formulation

Standard $\left[\Psi(x_{k+N}) + \sum_{i=1}^{N} \ell_{k+i}\right]$ s.t $c_j(x_{k+i}, u_{k+i-1}) \leq 0$

Economic $\gamma\left[\ell_{k+N} + \rho_d \Delta_{k+N}\right] + \sum_{i=1}^{N}\left[\ell_{k+i} + \rho_d \Delta_{k+i}\right]$ s.t $c_j(x_{k+i}, u_{k+i-1}) \leq 0$

Variable weight $\left[\sum_{i=1}^{N} [i/N]^m \ell(x_{k+i}, u_{k+i-1})\right]$ s.t $c_j(x_{k+i}, u_{k+i-1}) \leq 0$

Unified structure parameterized by ρ_f, ρ_d, m, ρ_{cstr}, N

$$\min_{\boldsymbol{u}} \Bigg[\rho_f \Psi(x_{k+N}, \rho_d \Delta_{k+N}) + \sum_{i=1}^{N} [i/N]^m \left[\ell_{k+i} + \rho_d \Delta_{k+i}\right] + \rho_{\text{cstr}} \max_{j=1}^{n_c} \left\lfloor c_j(x_{k+i}, u_{k+i-1}) \right\rfloor_+ \Bigg]$$

A single parameterized formulation

How to choose the parameters of the formulation?

- ✓ **Constraints** must be respected over the closed-loop trajectory
- ✓ Closed-loop **stability** should be guaranteed
- ✓ The feedback must be **real-time** implementable

Unified structure parameterized by ρ_f, ρ_d, m, ρ_{cstr}, N

$$\min_{\boldsymbol{u}} \Big[\rho_f \Psi(x_{k+N}, \rho_d \Delta_{k+N}) + \sum_{i=1}^{N} [i/N]^m \Big[\ell_{k+i} + \rho_d \Delta_{k+i}\Big] + \rho_{\text{cstr}} \max_{j=1}^{n_c} \big\lfloor c_j(x_{k+i}, u_{k+i-1})\big\rfloor_+ \Big]$$

How to choose the parameters of the formulation?

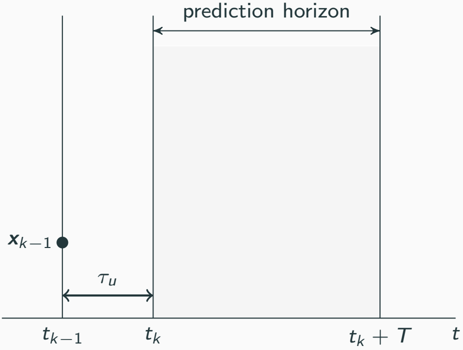
- ✓ **Constraints** must be respected over the closed-loop trajectory
- ✓ Closed-loop **stability** should be guaranteed
- ✓ The feedback must be **real-time** implementable ← SEE NEXT

Unified structure parameterized by ρ_f, ρ_d, m, ρ_{cstr}, N

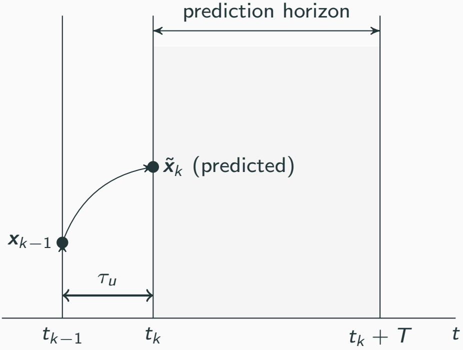
$$\min_{\boldsymbol{u}} \Big[\rho_f \Psi(x_{k+N}, \rho_d \Delta_{k+N}) + \sum_{i=1}^{N} [i/N]^m \Big[\ell_{k+i} + \rho_d \Delta_{k+i}\Big] + \rho_{\text{cstr}} \max_{j=1}^{n_c} \lfloor c_j(x_{k+i}, u_{k+i-1}) \rfloor_+ \Big]$$

Real-Time considerations

MPC: The Control Updating Period τ_u

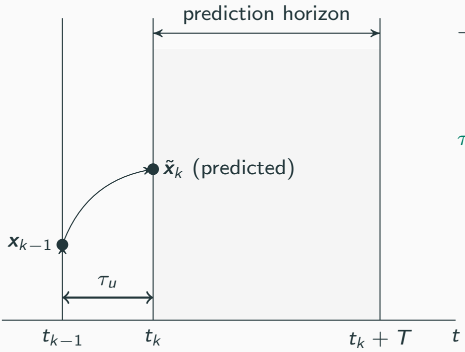


MPC: The Control Updating Period τ_u



1. Predict $\tilde{\boldsymbol{x}}_k$ (CPU-negligible)
2. During $[t_{k-1}, t_k]$ Compute $\mathbf{u}^\star(\tilde{\boldsymbol{x}}_k)$

MPC: The Control Updating Period τ_u



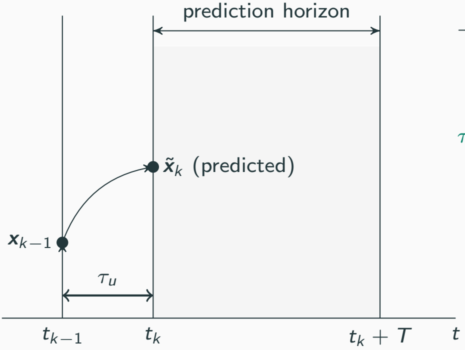
1. Predict $\tilde{\boldsymbol{x}}_k$ (CPU-negligible)
2. During $[t_{k-1}, t_k]$ Compute $\mathbf{u}^\star(\tilde{\boldsymbol{x}}_k)$

$\rightarrow$ Number of iterations during τ_u:

$$\texttt{max\_iter} \le \left\lfloor \frac{\tau_u}{\tau_c} \right\rfloor$$

τ_c: the CPU-cost of a single iteration!

MPC: The Control Updating Period τ_u



1. Predict $\tilde{\boldsymbol{x}}_k$ (CPU-negligible)
2. During $[t_{k-1}, t_k]$ Compute $\mathbf{u}^\star(\tilde{\boldsymbol{x}}_k)$

$\rightarrow$ Number of iterations during τ_u:

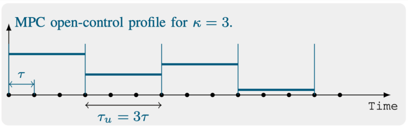
$$\texttt{max\_iter} \leq \left\lfloor \frac{\tau_u}{\tau_c} \right\rfloor$$

τ_c: the CPU-cost of a single iteration!

$\downarrow$

τ_u is a highly impacting implementation setting choice.

Implementation options

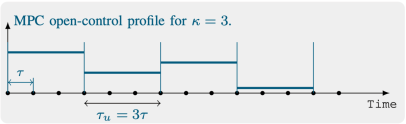


τ_u	The updating period.
τ	Basic sampling period†

† Largest τ making $\mathtt{RK}(\tau)$ appropriate

$$\tau_u = \kappa\tau \qquad \kappa \in \mathbb{N}$$

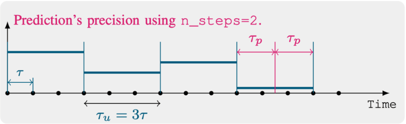
Implementation options



τ_u	The updating period.
τ	Basic sampling period[†]

† Largest τ making $\texttt{RK}(\tau)$ appropriate

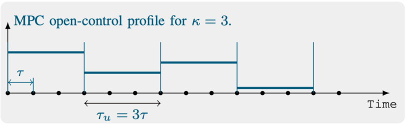
$$\tau_u = \kappa\tau \qquad \kappa \in \mathbb{N}$$



$\rightarrow$ Instead of $\kappa \times \texttt{RK}(\tau)$,

$\rightarrow$ use $\left(\texttt{n\_steps} \times \texttt{RK}(\frac{\tau_u}{\texttt{n\_steps}})\right)$

Implementation options

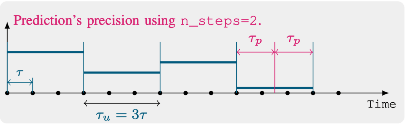


τ_u The updating period.

τ Basic sampling period†

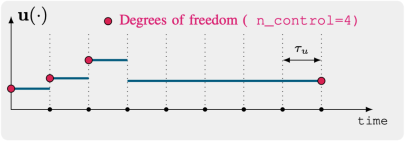
† Largest τ making $\mathtt{RK}(\tau)$ appropriate

$$\tau_u = \kappa\tau \qquad \kappa \in \mathbb{N}$$



→ Instead of $\kappa \times \mathtt{RK}(\tau)$,

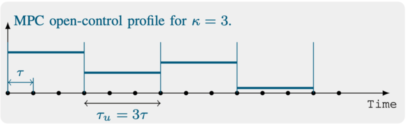
→ use $\left(\mathtt{n\_steps} \times \mathtt{RK}(\frac{\tau_u}{\mathtt{n\_steps}})\right)$



Use reduced control horizon leading to

$\mathtt{n\_control} \times n_u$ d.o.f.

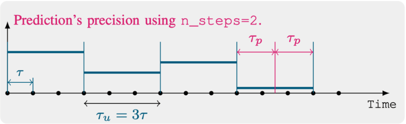
Implementation options



τ_u	The updating period.
τ	Basic sampling period†

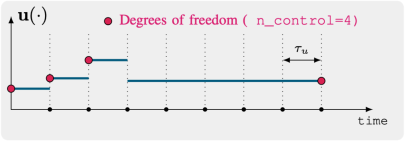
† Largest τ making $\mathtt{RK}(\tau)$ appropriate

$$\tau_u = \kappa\tau \qquad \kappa \in \mathbb{N}$$



$\rightarrow$ Instead of $\kappa \times \mathtt{RK}(\tau)$,

$\rightarrow$ use $\left(\mathtt{n\_steps} \times \mathtt{RK}(\frac{\tau_u}{\mathtt{n\_steps}})\right)$



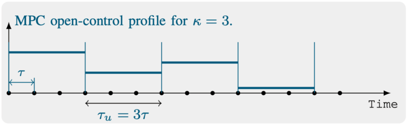
Use reduced control horizon leading to

$$\mathtt{n\_control} \times n_u \qquad \text{d.o.f.}$$

Choose the maximum number of iterations:

$$\mathtt{max\_iter} \qquad \left(\leq \left\lfloor \frac{\kappa\tau}{\tau_c} \right\rfloor\right)$$

Implementation options

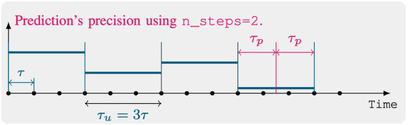


τ_u The updating period.

τ Basic sampling period†

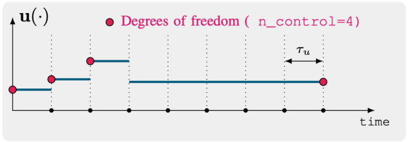
† Largest τ making $\mathtt{RK}(\tau)$ appropriate

$$\tau_u = \kappa\tau \qquad \kappa \in \mathbb{N}$$



$\rightarrow$ Instead of $\kappa \times \mathtt{RK}(\tau)$,

$\rightarrow$ use $\left(\mathtt{n\_steps} \times \mathtt{RK}(\frac{\tau_u}{\mathtt{n\_steps}})\right)$



Use reduced control horizon leading to

$$\mathtt{n\_control} \times n_u \qquad \text{d.o.f.}$$

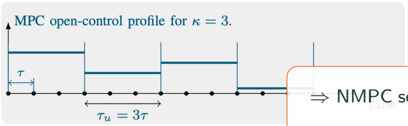
Choose the maximum number of iterations:

$$\mathtt{max\_iter} \qquad \left(\leq \left\lfloor \frac{\kappa\tau}{\tau_c} \right\rfloor\right)$$

Notice that $\tau_c = F(\mathtt{n\_step}, \mathtt{n\_control}, N_{\text{pred}})$

where the prediction horizon in given by:

$$T_p = N_{\text{pred}} \times \tau_u$$

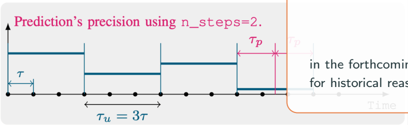


τ_u The updating period.

τ Basic sampling period†

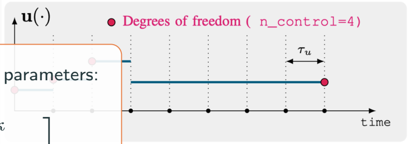
† Largest τ making RK(τ) appropriate

$$\tau_u = \kappa\tau \qquad \kappa \in \mathbb{N}$$



→ Instead of $\kappa \times$ RK(τ),

→ use $\left(\texttt{n\_steps} \times \text{RK}\left(\frac{\tau_u}{\texttt{n\_steps}}\right)\right)$



Use reduced control horizon leading to

$$\texttt{n\_control} \times n_u \qquad \text{d.o.f.}$$

Choose the maximum number of iterations:

$$\texttt{max\_iter} \qquad \left(\leq \left\lfloor \frac{\kappa\tau}{\tau_c} \right\rfloor\right)$$

Notice that $\tau_c = F(\texttt{n\_step}, \texttt{n\_control}, N_{pred})$

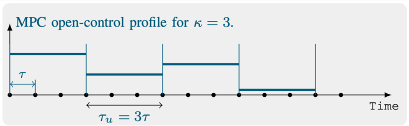
where the prediction horizon in given by:

$$T_p = N_{pred} \times \tau_u$$

⇒ NMPC setting parameters:

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{pred} \\ \rho_f \\ \rho_{cstr} \\ \rho_d \\ m \end{bmatrix}$$

in the forthcoming results: $\rho_d = m = 0$
for historical reasons :-)

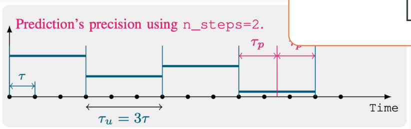


τ_u The updating perio

τ Basic sampling peri

† Largest τ making RK(τ)

$\tau_u = \kappa\tau \qquad \kappa \in \mathbb{N}$

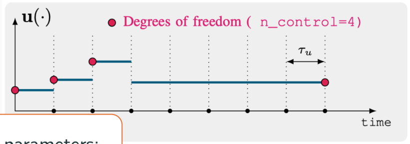


$\rightarrow$ Instead of $\kappa \times \mathtt{RK}(\tau)$,

$\rightarrow$ use $\left(\mathtt{n\_steps} \times \mathtt{RK}\left(\frac{\tau_u}{\mathtt{n\_steps}}\right)\right)$

$\Rightarrow$ NMPC setting parameters:

$$\pi := \begin{bmatrix} \kappa \\ \mathtt{n\_steps} \\ \mathtt{n\_control} \\ \mathtt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$



ced control horizon leading to

control $\times\, n_u$ d.o.f.

ximum number of iterations:

_iter $\left(\leq \left\lfloor \frac{\kappa\tau}{\tau_c} \right\rfloor\right)$

Notice that $\tau_c = F(\mathtt{n\_step}, \mathtt{n\_control}, N_{\text{pred}})$

where the prediction horizon in given by:

$$T_p = N_{\text{pred}} \times \tau_u$$

Design of NMPC setting

Design of NMPC setting: Problem statement

Problem Statement

Given the following items:

- ✓ Problem ingredients: (dynamics, constraints, $\overbrace{\text{parameterized cost}}^{\Psi,\ell}$)
- ✓ An admissible set $\Pi := [\underline{\pi}, \bar{\pi}]$;
- ✓ A computation targeted device (charcaterized by `dev_acc`)
- ✓ A given optimization algorithm,

Design of NMPC setting: Problem statement

Problem Statement

Given the following items:

- ✓ Problem ingredients: (dynamics, constraints, $\overbrace{\text{parameterized cost}}^{\Psi,\ell}$)
- ✓ An admissible set $\Pi := [\underline{\pi}, \bar{\pi}]$;
- ✓ A computation targeted device (charcaterized by `dev_acc`)
- ✓ A given optimization algorithm,

Derive a **tractable heuristic** that:

- ✓ Either yields a rational choice of the vector of parameters π addressing stability, constraints satisfaction and real-time implementability concerns;

Problem Statement

Given the following items:

- ✓ Problem ingredients: (dynamics, constraints, $\overbrace{\text{parameterized cost}}^{\Psi,\ell}$)
- ✓ An admissible set $\Pi := [\underline{\pi}, \bar{\pi}]$;
- ✓ A computation targeted device (charcaterized by `dev_acc`)
- ✓ A given optimization algorithm,

Derive a **tractable heuristic** that:

- ✓ Either yields a rational choice of the vector of parameters π addressing stability, constraints satisfaction and real-time implementability concerns;
- ✓ Or it suggests that these requirements cannot be met given the data of the problem.

Design of NMPC setting: Problem statement

Problem Statement

Given the following items:

- ✓ Problem ingredients: (dynamics, constraints, $\overbrace{\text{parameterized cost}}^{\Psi,\ell}$)
- ✓ An admissible set $\Pi := [\underline{\pi}, \bar{\pi}]$;
- ✓ A computation targeted device (charcaterized by `dev_acc`)
- ✓ A given optimization algorithm,

What does this precisely mean?

Derive a **tractable heuristic** that:

- ✓ Either yields a rational choice of the vector of parameters π addressing stability, constraints satisfaction and real-time implementability concerns;
- ✓ Or it suggests that these requirements cannot be met given the data of the problem.

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all $\texttt{sc} \in \mathcal{A}$:

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all $\texttt{sc} \in \mathcal{A}$:

RT-feasibility

$$C_{RT}(\pi|\texttt{sc}) := \max_{k=1,\dots,K}\left\lfloor \frac{\tau_{\text{solver}}}{\tau_u}(k) - \texttt{dev\_acc}\right\rfloor_+ = 0$$

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all $\texttt{sc} \in \mathcal{A}$:

γ-Contraction

$$C_\gamma(\pi|\texttt{sc}) := \lfloor J_{ol}(K) - \gamma J_{ol}(1) \rfloor_+ = 0$$

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all $\texttt{sc}\in\mathcal{A}$:

Constraints satisfaction

$$C_{\text{cstr}}(\pi|\texttt{sc}) := \max_{k=1,\dots,K} \lfloor \max_{i\leq n_c} c_i(k) \rfloor_+ = 0$$

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all sc$\in \mathcal{A}$:

RT-feasibility

$$C_{RT}(\pi|\texttt{sc}) := \max_{k=1,\dots,K}\left\lfloor \frac{\tau_{\text{solver}}}{\tau_u}(k) - \texttt{dev\_acc}\right\rfloor_+ = 0$$

γ-Contraction

$$C_\gamma(\pi|\texttt{sc}) := \lfloor J_{ol}(K) - \gamma J_{ol}(1)\rfloor_+ = 0$$

Constraints satisfaction

$$C_{\text{cstr}}(\pi|\texttt{sc}) := \max_{k=1,\dots,K}\lfloor \max_{i\le n_c} c_i(k)\rfloor_+ = 0$$

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all sc$\in \mathcal{A}$:

RT-feasibility

$$C_{RT}(\pi|\mathtt{sc}) := \max_{k=1,\dots,K} \left\lfloor \frac{\tau_{\text{solver}}}{\tau_u}(k) - \mathtt{dev\_acc} \right\rfloor_+ = 0$$

γ-Contraction

$$C_\gamma(\pi|\mathtt{sc}) := \lfloor J_{ol}(K) - \gamma J_{ol}(1) \rfloor_+ = 0$$

Constraints satisfaction

$$C_{\text{cstr}}(\pi|\mathtt{sc}) := \max_{k=1,\dots,K} \lfloor \max_{i \le n_c} c_i(k) \rfloor_+ = 0$$

$$\pi := \begin{bmatrix} \kappa \\ \mathtt{n\_steps} \\ \mathtt{n\_control} \\ \mathtt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Question 1:

What is a scenario sc?

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all sc$\in \mathcal{A}$:

RT-feasibility

$$C_{RT}(\pi|\texttt{sc}) := \max_{k=1,\dots,K} \left\lfloor \frac{\tau_{\text{solver}}}{\tau_u}(k) - \texttt{dev\_acc} \right\rfloor_+ = 0$$

γ-Contraction

$$C_\gamma(\pi|\texttt{sc}) := \lfloor J_{ol}(K) - \gamma J_{ol}(1) \rfloor_+ = 0$$

Constraints satisfaction

$$C_{\text{cstr}}(\pi|\texttt{sc}) := \max_{k=1,\dots,K} \lfloor \max_{i\le n_c} c_i(k) \rfloor_+ = 0$$

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Question 1:

What is a scenario sc?

Question 2:

What is a sufficiently rich set of scenarios $\mathcal{A}$?

What is a scenario?

A scenario `sc` is an instance of the necessary information needed to perform a closed-loop simulation should the NMPC setting π be given!

What is a scenario?

A scenario $\mathtt{sc}$ is an instance of the necessary information needed to perform a closed-loop simulation should the NMPC setting π be given!

$$\mathtt{sc} := \begin{bmatrix} x_0 \\ p \\ q \end{bmatrix}$$

x_0	Initial state vector
p	Model's parameter vector
q	Exogenous/context items' vector
	(set-points, observed quantities, disturbance, etc.)

What is a **scenario?**

A scenario $\mathtt{sc}$ is an instance of the necessary information needed to perform a closed-loop simulation should the NMPC setting π be given!

$$\mathtt{sc} := \begin{bmatrix} x_0 \\ p \\ q \end{bmatrix}$$

x_0	Initial state vector
p	Model's parameter vector
q	Exogenous/context items' vector
	(set-points, observed quantities, disturbance, etc.)

RT-feasibility

$$C_{RT}(\pi|\mathbf{sc}) := \max_{k=1,\dots,K} \left\lfloor \frac{\tau_{\text{solver}}}{\tau_u}(k) - \mathbf{dev\_acc} \right\rfloor_+ = 0$$

γ-Contraction

$$C_\gamma(\pi|\mathbf{sc}) := \lfloor J_{ol}(K) - \gamma J_{ol}(1) \rfloor_+ = 0$$

Constraints satisfaction

$$C_{\text{cstr}}(\pi|\mathbf{sc}) := \max_{k=1,\dots,K} \lfloor \max_{i \leq n_c} c_i(k) \rfloor_+ = 0$$

What is a sufficiently rich set of scenarios $\mathcal{A}$?

This is conditioned by:

- ✓ An appropriate sampling of (x_0, p, q) inside their sets of possibilities
- ✓ A sufficiently high number `n_samples` of instances

What is a sufficiently rich set of scenarios $\mathcal{A}$?

This is conditioned by:

✓ An appropriate sampling of (x_0, p, q) inside their sets of possibilities

✓ A sufficiently high number `n_samples` of instances

# candidates	$\eta = 0.1$	$\eta = 0.05$	$\eta = 0.01$	$\eta = 0.001$
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

Table 1: The required card($\mathcal{A}$) as a function of the precision parameter η for a confidence parameter of $\delta = 10^{-3}$ and a number of admitted failure $= 1$.

The values of `n_samples` needed to be confident at (1-δ)100% that the properties are satisfied with a probability greater than (1-η)100%

[Alamo et al. Randomized strategies for probilistic solution of uncertain feasibility ans optimization problems, IEEE TAC, 2009]

Design of NMPC setting: Problem Statement

Find a design parameter vector π s.t. over a sufficiently rich set of scenarios $\mathcal{A}$, an K-steps closed-loop trajectories under the π-corresponding NMPC satisfy for all sc$\in \mathcal{A}$:

RT-feasibility

$$C_{RT}(\pi|\texttt{sc}) := \max_{k=1,\dots,K}\Big\lfloor \frac{\tau_{\text{solver}}}{\tau_u}(k) - \texttt{dev\_acc}\Big\rfloor_+ = 0$$

γ-Contraction

$$C_\gamma(\pi|\texttt{sc}) := \lfloor J_{ol}(K) - \gamma J_{ol}(1)\rfloor_+ = 0$$

Constraints satisfaction

$$C_{\text{cstr}}(\pi|\texttt{sc}) := \max_{k=1,\dots,K}\lfloor \max_{i\le n_c} c_i(k)\rfloor_+ = 0$$

$$\pi := \begin{bmatrix} \kappa \\ \texttt{n\_steps} \\ \texttt{n\_control} \\ \texttt{max\_iter} \\ N_{\text{pred}} \\ \rho_f \\ \rho_{cstr} \end{bmatrix}$$

Question 1:

What is a scenario sc?

Question 2:

What is a sufficiently rich set of scenarios $\mathcal{A}$?

Design of NMPC setting: Principle & Difficulties

Principle

- ✓ $(\pi, \mathtt{sc}) \in \Pi \times \mathcal{A} \rightarrow$ successful or not!
- ✓ π is eligible if successful on all $\mathtt{sc}$ in $\mathcal{A}$

Design of NMPC setting: Principle & Difficulties

Principle

- ✓ $(\pi, \texttt{sc}) \in \Pi \times \mathcal{A} \rightarrow$ successful or not!
- ✓ π is eligible if successful on all $\texttt{sc}$ in $\mathcal{A}$

Difficulties

- ✓ checking admissibility of $\pi \leftarrow$ `n_samples` $\textsc{cl}$-simulations
- ✓ Π is rather high dimensional non convex set!

$\rightarrow$ **intractable** without a specific hints and heuristics.

Design of NMPC setting: Principle & Difficulties

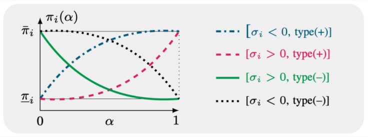
Principle

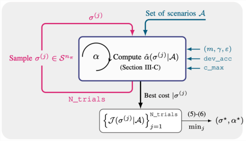
- ✓ $(\pi, \mathtt{sc}) \in \Pi \times \mathcal{A} \rightarrow$ successful or not!
- ✓ π is eligible if successful on all $\mathtt{sc}$ in $\mathcal{A}$

Difficulties

- ✓ checking admissibility of $\pi \leftarrow$ $\mathtt{n\_samples}$ $\textsc{cl}$-simulations
- ✓ Π is rather high dimensional non convex set!

$\rightarrow$ **intractable** without a specific hints and heuristics.

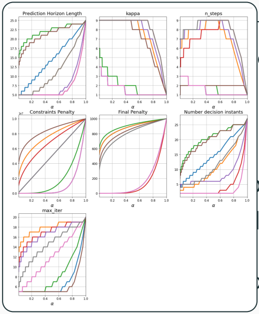




[M. Alamir. A Framework and a python-package for Real-time NMPC parameters settings. http://arxiv.org/abs/2309.17238. 2023.]

Design of NMPC setting: Principle & Difficulties

Principle



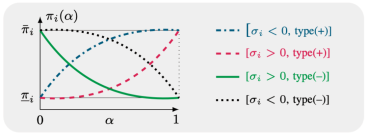
$\rightarrow$ successful or not!

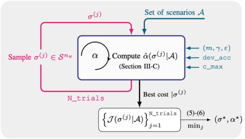
cessful on all sc in $\mathcal{A}$

bility of $\pi \leftarrow$ n_samples CL-simulations

limensional non convex set!

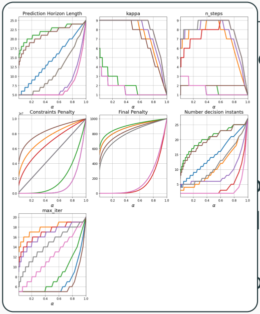
out a specific hints and heuristics.





[M. Alamir. A Framework and a python-package for Real-time NMPC parameters settings. http://arxiv.org/abs/2309.17238. 2023.]

Principle



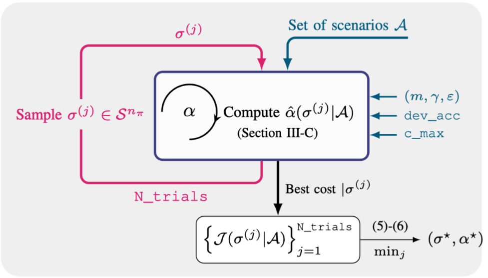
→ successfu

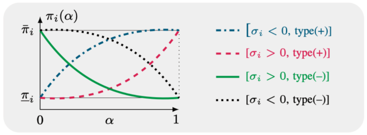
cessful on a

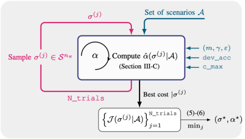
bility of π ←

limensional

out a specific hints and heuristics.



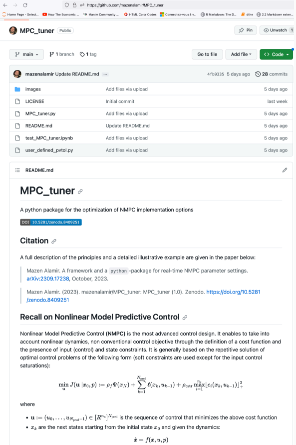




[M. Alamir. A Framework and a python-package for Real-time NMPC parameters settings. http://arxiv.org/abs/2309.17238. 2023.]

The `MPC_tuner` package

The `MPC_tunner` package: An overview



https://github.com/mazenalamir/MPC_tuner

Home Page - Select... How The Economic ... Manim Community ... HTML Color Codes Connectez-vous à v... R Markdown: The D... dthe 2.2 Markdown exten...

MPC_tuner Public

Pin Unwatch 1

main 1 branch 1 tag Go to file Add file Code

mazenalamir Update README.md ... 4fb9335 5 days ago 28 commits

images	Add files via upload	5 days ago
LICENSE	Initial commit	last week
MPC_tuner.py	Add files via upload	5 days ago
README.md	Update README.md	5 days ago
test_MPC_tuner.ipynb	Add files via upload	5 days ago
user_defined_pvtol.py	Add files via upload	5 days ago

README.md

MPC_tuner

A python package for the optimization of NMPC implementation options

DOI 10.5281/zenodo.8409251

Citation

A full description of the principles and a detailed illustrative example are given in the paper below:

> Mazen Alamir. A framework and a `python` -package for real-time NMPC parameter settings. arXiv:2309.17238, October, 2023.

> Mazen Alamir. (2023). mazenalamir/MPC_tuner: MPC_tuner (1.0). Zenodo. https://doi.org/10.5281/zenodo.8409251

Recall on Nonlinear Model Predictive Control

Nonlinear Model Predictive Control **(NMPC)** is the most advanced control design. It enables to take into account nonlinear dynamics, non conventional control objective through the definition of a cost function and the presence of input (control) and state constraints. It is generally based on the repetitive solution of optimal control problems of the following form (soft constraints are used except for the input control saturations):

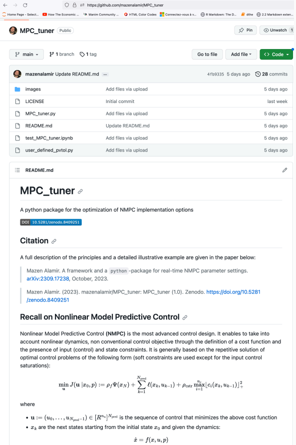
$$\min_{\mathbf{u}} J(\mathbf{u}\,|x_0,p) := \rho_f\Psi(x_N) + \sum_{k=1}^{N_{\rm pred}}\ell(x_k,u_{k-1}) + \rho_{\rm cstr}\max_{i=1}^{n_c}\lfloor c_i(x_k,u_{k-1})\rfloor_+^2$$

where

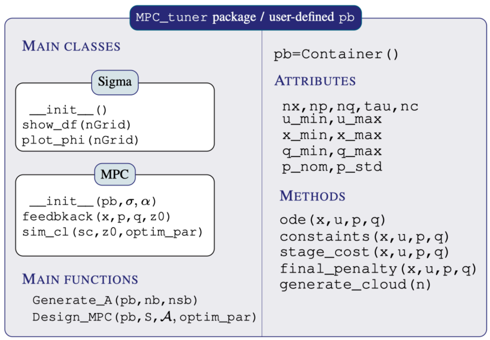
- $\mathbf{u} := (u_0,\dots,u_{N_{\rm pred}-1}) \in [R^{n_u}]^{N_{\rm pred}}$ is the sequence of control that minimizes the above cost function
- x_k are the next states starting from the initial state x_0 and given the dynamics:

$$\dot x = f(x,u,p)$$

The MPC_tunner package: An overview

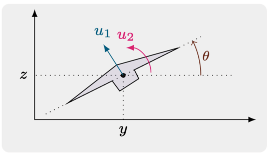


https://github.com/mazenalamir/MPC_tuner



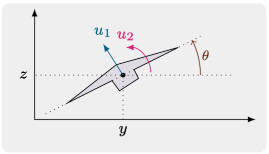
[M. Alamir. A Framework and a python-package for Real-time NMPC parameters settings. http://arxiv.org/abs/2309.17238. 2023.]

The MPC_tunner package: A use-case



$$\ddot{y} = -u_1 \sin\theta + p_1 u_2 \cos\theta$$
$$\ddot{z} = u_1 \cos\theta + p_1 u_2 \sin\theta - 1$$
$$\ddot{\theta} = p_2 u_2$$

The `MPC_tunner` package: A use-case



$$\ddot{y} = -u_1 \sin\theta + p_1 u_2 \cos\theta$$
$$\ddot{z} = u_1 \cos\theta + p_1 u_2 \sin\theta - 1$$
$$\ddot{\theta} = p_2 u_2$$

$x := (y, z, \theta, \dot{y}, \dot{z}, \dot{\theta})$

$x_d := (q_1, q_2, 0, \dots, 0)$

$\ell := \|x - x_d\|_Q^2 + \|u - u_d\|_R^2$

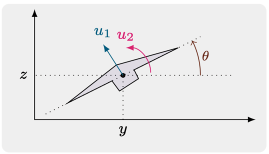
$\Psi := \rho_f \|x - x_d\|_Q$

$\gamma := 0.98,$

$u \in [-50, +50]^2$

$|\dot{\theta}| \leq 1$ and $|\theta| \leq \pi$

$\tau = 0.02$



Design parameter	min-value	max-value
N_pred	5	25
κ	1	10
ρ_f	1	10^3
rho_cstr	10^3	10^7
max_iter	5	20

Definition of Π boundaries.

$$\ddot{y} = -u_1 \sin\theta + p_1 u_2 \cos\theta$$
$$\ddot{z} = u_1 \cos\theta + p_1 u_2 \sin\theta - 1$$
$$\ddot{\theta} = p_2 u_2$$

$x := (y, z, \theta, \dot{y}, \dot{z}, \dot{\theta})$

$x_d := (q_1, q_2, 0, \dots, 0)$

$\ell := \|x - x_d\|_Q^2 + \|u - u_d\|_R^2$

$\Psi := \rho_f \|x - x_d\|_Q$

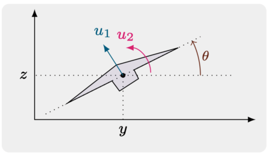
$\gamma := 0.98,$

$u \in [-50, +50]^2$

$|\dot{\theta}| \leq 1$ and $|\theta| \leq \pi$

$\tau = 0.02$

The MPC_tunner package: A use-case



$$\ddot{y} = -u_1 \sin\theta + p_1 u_2 \cos\theta$$
$$\ddot{z} = u_1 \cos\theta + p_1 u_2 \sin\theta - 1$$
$$\ddot{\theta} = p_2 u_2$$

$x := (y, z, \theta, \dot{y}, \dot{z}, \dot{\theta})$

$x_d := (q_1, q_2, 0, \dots, 0)$

$\ell := \|x - x_d\|_Q^2 + \|u - u_d\|_R^2$

$\Psi := \rho_f \|x - x_d\|_Q$

$\gamma := 0.98,$

$u \in [-50, +50]^2$

$|\dot{\theta}| \leq 1$ and $|\theta| \leq \pi$

$\tau = 0.02$

Design parameter	min-value	max-value
N_pred	5	25
κ	1	10
ρ_f	1	10^3
rho_cstr	10^3	10^7
max_iter	5	20

Definition of Π boundaries.

```
import numpy as np
from casadi import vertcat

class Container:
    def __init__(self):
        pass

pvtol = Container()
pvtol.nx, pvtol.nu = 6, 2
pvtol.n_p, pvtol.n_q = 2, 4
pvtol.u_min = np.array([-5e1, -5e1])
pvtol.u_max = np.array([+5e1, +5e1])
pvtol.x_min = np.array([-2, -2, -0.8 * np.pi, \
                         -0.1, -0.1, -0.1])
pvtol.x_max = -pvtol.x_min
pvtol.q_min = [-1, -1, 1, np.pi]
pvtol.q_max = [+1, +1, 1, np.pi]
pvtol.p_nom = pvtol.p
pvtol.p_std, pvtol.tau, pvtol.nc = 0.1, 0.02, 4
#------
```

The user-defined file (part 1)

```
#------
def p_vtol(x, u, p):
    xdot = vertcat(...)
    return xdot
#------
def constraints(x, u, p, q):
    d_theta_dt_max, theta_max = q[2], q[3]
    c = vertcat(x[5]/d_theta_dt_max-1,...)
    return c
#------
def stage_cost(x, u, p, q):
    ...
    return ell
#------
def final_penalty(x, p, q):
    ...
    ef = 0
    for i in range(6):
        ef += q_xf[i] * (x[i] - xd[i]) ** 2
    return ef
#------
def generate_cloud(nSamples=None):
    x_min, x_max = pvtol.x_min, pvtol.x_max
    ...
    A_sc = Container()
    A_sc.x0, A_sc.p, A_sc.q = X, P, Q
    return A_sc
#------
pvtol.ode = p_vtol
pvtol.constraints = constraints
pvtol.final_penalty = final_penalty
pvtol.stage_cost = stage_cost
pvtol.generate_cloud = generate_cloud
```

The user-defined file (part 2)

The MPC_tunner package: A use-case

```python
from user_defined_pvtol import pvtol
from pyMPC import Design_MPC, Sigma, generate_A, OptimPar

# Generate the set of scenarios
nb, nsb = 30, 10
A = generate_A(pvtol, nb, nsb)

# Generate the set of candidate sigma's
N_trials = 100
S = [Sigma() for _ in range(N_trials)]

# Set the Design meta-parameters and run the Design
optim_par = OptimPar(gam=0.98, c_max=0.1,
                     dev_acc=1.0, T=0.5)

R_design_log = Design_MPC(pvtol, S, A, optim_par)
```

$\gamma := 0.98,$

$u \in [-50, +50]^2$

$|\dot{\theta}| \leq 1$ and $|\theta| \leq \pi$

$\tau = 0.02$

```python
pvtol.q_min = [-1, -1, 1, np.pi]
pvtol.q_max = [+1, +1, 1, np.pi]
pvtol.p_nom = pvtol.p
pvtol.p_std, pvtol.tau, pvtol.nc = 0.1, 0.02, 4
#------
```

The user-defined file (part 1)

```python
#------
def p_vtol(x, u, p):
    xdot = vertcat(...)
    return xdot
#------
def constraints(x, u, p, q):
    d_theta_dt_max, theta_max = q[2], q[3]
    c = vertcat(x[5]/d_theta_dt_max-1,...)
    return c
#------
def stage_cost(x, u, p, q):
    ...
    return ell
#------
def final_penalty(x, p, q):
    ...
    ef = 0
    for i in range(6):
        ef += q_xf[i] * (x[i] - xd[i]) ** 2
    return ef
#------
def generate_cloud(nSamples=None):
    x_min, x_max = pvtol.x_min, pvtol.x_max
    ...
    A_sc = Container()
    A_sc.x0, A_sc.p, A_sc.q = X, P, Q
    return A_sc
#------
pvtol.ode = p_vtol
pvtol.constraints = constraints
pvtol.final_penalty = final_penalty
pvtol.stage_cost = stage_cost
pvtol.generate_cloud = generate_cloud
```

The user-defined file (part 2)

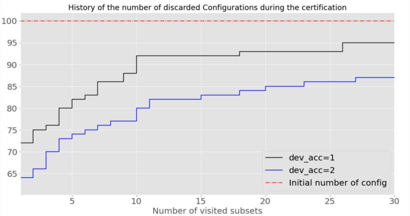
Results: Admissible design settings

dev_acc = 1

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
1	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
2	2	5	2	2	7.937212e+06	1.243896	0.02	0.04	8	2437.847089	0.250
3	8	7	5	2	1.250875e+06	2.951172	0.02	0.16	18	3401.306977	0.500
4	9	6	7	7	9.170123e+06	8.804688	0.02	0.18	6	3755.852208	0.500

dev_acc = 2

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	7	17	4	12	8.409123e+06	16.609375	0.02	0.14	5	3210.056159	0.250
1	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
2	9	20	2	19	8.409123e+06	820.515021	0.02	0.18	8	3766.654430	0.250
3	6	17	3	16	6.124112e+06	3.778133	0.02	0.12	5	3284.618189	0.375
4	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
5	9	5	8	4	6.259375e+05	2.951172	0.02	0.18	18	3126.158301	0.500
6	8	12	2	11	7.515030e+04	1.054939	0.02	0.16	17	2891.563097	0.375
7	7	9	5	2	2.442162e+06	15.537363	0.02	0.14	19	3583.879784	0.625
8	9	5	8	6	9.350676e+06	38.215650	0.02	0.18	18	4611.172897	0.625
9	2	12	2	2	8.846257e+06	1.146503	0.02	0.04	7	3408.564392	0.375
10	6	6	6	2	5.282910e+05	1.146503	0.02	0.12	18	3355.846137	0.375
11	9	18	2	13	3.536180e+06	125.875000	0.02	0.18	12	3250.814430	0.125
12	6	19	2	2	7.430228e+06	2.951172	0.02	0.12	14	3360.845813	0.125



Results of two different runs of the `Design_MPC` for two targeted device `dev_acc=1` and `dev_acc=2`.
Both use **100 sampled candidate design settings**.

$\text{card}(\mathcal{A}) = 270 > N_{\text{certification}}(\eta = 0.05, \delta = 10^{-3}) = 264$

Results: Admissible design settings

dev_acc = 1

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
1	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
2	2	5	2	2	7.937212e+06	1.243896	0.02	0.04	8	2437.847089	0.250
3	8	7	5	2	1.250875e+06	2.951172	0.02	0.16	18	3401.306977	0.500
4	9	6	7	7	9.170123e+06	8.804688	0.02	0.18	6	3755.852208	0.500

dev_acc = 2

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	7	17	4	12	8.409123e+06	16.609375	0.02	0.14	5	3210.056159	0.250
1	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
2	9	20	2	19	8.409123e+06	820.515021	0.02	0.18	8	3766.654430	0.250
3	6	17	3	16	6.124112e+06	3.778133	0.02	0.12	5	3284.618189	0.375
4	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
5	9	5	8	4	6.259375e+05	2.951172	0.02	0.18	18	3126.158301	0.500
6	8	12	2	11	7.515030e+04	1.054939	0.02	0.16	17	2891.563097	0.375
7	7	9	5	2	2.442162e+06	15.537363	0.02	0.14	19	3583.879784	0.625
8	9	5	8	6	9.350676e+06	38.215650	0.02	0.18	18	4611.172897	0.625
9	2	12	2	2	8.846257e+06	1.146503	0.02	0.04	7	3408.564392	0.375
10	6	6	6	2	5.282910e+05	1.146503	0.02	0.12	18	3355.846137	0.375
11	9	18	2	13	3.536180e+06	125.875000	0.02	0.18	12	3250.814430	0.125
12	6	19	2	2	7.430228e+06	2.951172	0.02	0.12	14	3360.845813	0.125

Design parameter	min-value	max-value
N_pred	5	25
κ	1	10
ρ_f	1	10^3
rho_cstr	10^3	10^7
max_iter	5	20

Small portion of configurations is eligible

5% dev_acc=1
13% dev_acc=2

Results of two different runs of the Design_MPC for two targeted device dev_acc=1 and dev_acc=2.
Both use **100 sampled candidate design settings**.

$\text{card}(\mathcal{A}) = 270 > N_{\text{certification}}(\eta = 0.05, \delta = 10^{-3}) = 264$

Results: Admissible design settings

dev_acc = 1

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
1	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
2	2	5	2	2	7.937212e+06	1.243896	0.02	0.04	8	2437.847089	0.250
3	8	7	5	2	1.250875e+06	2.951172	0.02	0.16	18	3401.306977	0.500
4	9	6	7	7	9.170123e+06	8.804688	0.02	0.18	6	3755.852208	0.500

dev_acc = 2

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	7	17	4	12	8.409123e+06	16.609375	0.02	0.14	5	3210.056159	0.250
1	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
2	9	20	2	19	8.409123e+06	820.515021	0.02	0.18	8	3766.654430	0.250
3	6	17	3	16	6.124112e+06	3.778133	0.02	0.12	5	3284.618189	0.375
4	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
5	9	5	8	4	6.259375e+05	2.951172	0.02	0.18	18	3126.158301	0.500
6	8	12	2	11	7.515030e+04	1.054939	0.02	0.16	17	2891.563097	0.375
7	7	9	5	2	2.442162e+06	15.537363	0.02	0.14	19	3583.879784	0.625
8	9	5	8	6	9.350676e+06	38.215650	0.02	0.18	18	4611.172897	0.625
9	2	12	2	2	8.846257e+06	1.146503	0.02	0.04	7	3408.564392	0.375
10	6	6	6	2	5.282910e+05	1.146503	0.02	0.12	18	3355.846137	0.375
11	9	18	2	13	3.536180e+06	125.875000	0.02	0.18	12	3250.814430	0.125
12	6	19	2	2	7.430228e+06	2.951172	0.02	0.12	14	3360.845813	0.125

Results of two different runs of the `Design_MPC` for two targeted device `dev_acc=1` and `dev_acc=2`.
Both use **100 sampled candidate design settings**.

$\text{card}(\mathcal{A}) = 270 > N_{\text{certification}}(\eta = 0.05, \delta = 10^{-3}) = 264$

Design parameter	min-value	max-value
N_pred	5	25
κ	1	10
ρ_f	1	10^3
rho_cstr	10^3	10^7
max_iter	5	20

High values of ρ_f lead to unfeasibility

Recall that random sampling of

$$\rho_f \in [1, 1000]$$

is applied but only rather small values appear in the resulting admissible settings.

Results: Admissible design settings

dev_acc = 1

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
1	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
2	2	5	2	2	7.937212e+06	1.243896	0.02	0.04	8	2437.847089	0.250
3	8	7	5	2	1.250875e+06	2.951172	0.02	0.16	18	3401.306977	0.500
4	9	6	7	7	9.170123e+06	8.804688	0.02	0.18	6	3755.852208	0.500

dev_acc = 2

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	7	17	4	12	8.409123e+06	16.609375	0.02	0.14	5	3210.056159	0.250
1	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
2	9	20	2	19	8.409123e+06	820.515021	0.02	0.18	8	3766.654430	0.250
3	6	17	3	16	6.124112e+06	3.778133	0.02	0.12	5	3284.618189	0.375
4	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
5	9	5	8	4	6.259375e+05	2.951172	0.02	0.18	18	3126.158301	0.500
6	8	12	2	11	7.515030e+04	1.054939	0.02	0.16	17	2891.563097	0.375
7	7	9	5	2	2.442162e+06	15.537363	0.02	0.14	19	3583.879784	0.625
8	9	5	8	6	9.350676e+06	38.215650	0.02	0.18	18	4611.172897	0.625
9	2	12	2	2	8.846257e+06	1.146503	0.02	0.04	7	3408.564392	0.375
10	6	6	6	2	5.282910e+05	1.146503	0.02	0.12	18	3355.846137	0.375
11	9	18	2	13	3.536180e+06	125.875000	0.02	0.18	12	3250.814430	0.125
12	6	19	2	2	7.430228e+06	2.951172	0.02	0.12	14	3360.845813	0.125

Results of two different runs of the `Design_MPC` for two targeted device `dev_acc=1` and `dev_acc=2`.
Both use **100 sampled candidate design settings**.

$\text{card}(\mathcal{A}) = 270 > N_{\text{certification}}(\eta = 0.05, \delta = 10^{-3}) = 264$

Design parameter	min-value	max-value
`N_pred`	5	25
κ	1	10
ρ_f	1	10^3
`rho_cstr`	10^3	10^7
`max_iter`	5	20

High values of ρ_f lead to unfeasibility

Recall that random sampling of

$$\rho_f \in [1, 1000]$$

is applied but only rather small values appear in the resulting admissible settings.

High values of ρ_{cstr} are necessary

Only high values of ρ_{cstr} appear in the admissible settings

Results: Admissible design settings

dev_acc = 1

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
1	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
2	2	5	2	2	7.937212e+06	1.243896	0.02	0.04	8	2437.847089	0.250
3	8	7	5	2	1.250875e+06	2.951172	0.02	0.16	18	3401.306977	0.500
4	9	6	7	7	9.170123e+06	8.804688	0.02	0.18	6	3755.852208	0.500

dev_acc = 2

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	7	17	4	12	8.409123e+06	16.609375	0.02	0.14	5	3210.056159	0.250
1	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
2	9	20	2	19	8.409123e+06	820.515021	0.02	0.18	8	3766.654430	0.250
3	6	17	3	16	6.124112e+06	3.778133	0.02	0.12	5	3284.618189	0.375
4	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
5	9	5	8	4	6.259375e+05	2.951172	0.02	0.18	18	3126.158301	0.500
6	8	12	2	11	7.515030e+04	1.054939	0.02	0.16	17	2891.563097	0.375
7	7	9	5	2	2.442162e+06	15.537363	0.02	0.14	19	3583.879784	0.625
8	9	5	8	6	9.350676e+06	38.215650	0.02	0.18	18	4611.172897	0.625
9	2	12	2	2	8.846257e+06	1.146503	0.02	0.04	7	3408.564392	0.375
10	6	6	6	2	5.282910e+05	1.146503	0.02	0.12	18	3355.846137	0.375
11	9	18	2	13	3.536180e+06	125.875000	0.02	0.18	12	3250.814430	0.125
12	6	19	2	2	7.430228e+06	2.951172	0.02	0.12	14	3360.845813	0.125

Results of two different runs of the `Design_MPC` for two targeted device `dev_acc=1` and `dev_acc=2`.
Both use **100 sampled candidate design settings**.

$\text{card}(\mathcal{A}) = 270 > N_{\text{certification}}(\eta = 0.05, \delta = 10^{-3}) = 264$

Design parameter	min-value	max-value
N_pred	5	25
κ	1	10
ρ_f	1	10^3
rho_cstr	10^3	10^7
max_iter	5	20

High values of ρ_f lead to unfeasibility

Recall that random sampling of

$$\rho_f \in [1, 1000]$$

is applied but only rather small values appear in the resulting admissible settings.

High values of ρ_{cstr} are necessary

Only high values of ρ_{cstr} appear in the admissible settings

$\Rightarrow$ **The scenarios are constraints-challenging**

Results: Admissible design settings

dev_acc = 1

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
1	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
2	2	5	2	2	7.937212e+06	1.243896	0.02	0.04	8	2437.847089	0.250
3	8	7	5	2	1.250875e+06	2.951172	0.02	0.16	18	3401.306977	0.500
4	9	6	7	7	9.170123e+06	8.804688	0.02	0.18	6	3755.852208	0.500

dev_acc = 2

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	7	17	4	12	8.409123e+06	16.609375	0.02	0.14	5	3210.056159	0.250
1	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
2	9	20	2	19	8.409123e+06	820.515021	0.02	0.18	8	3766.654430	0.250
3	6	17	3	16	6.124112e+06	3.778133	0.02	0.12	5	3284.618189	0.375
4	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
5	9	5	8	4	6.259375e+05	2.951172	0.02	0.18	18	3126.158301	0.500
6	8	12	2	11	7.515030e+04	1.054939	0.02	0.16	17	2891.563097	0.375
7	7	9	5	2	2.442162e+06	15.537363	0.02	0.14	19	3583.879784	0.625
8	9	5	8	6	9.350676e+06	38.215650	0.02	0.18	18	4611.172897	0.625
9	2	12	2	2	8.846257e+06	1.146503	0.02	0.04	7	3408.564392	0.375
10	6	6	6	2	5.282910e+05	1.146503	0.02	0.12	18	3355.846137	0.375
11	9	18	2	13	3.536180e+06	125.875000	0.02	0.18	12	3250.814430	0.125
12	6	19	2	2	7.430228e+06	2.951172	0.02	0.12	14	3360.845813	0.125

Results of two different runs of the Design_MPC for two targeted device dev_acc=1 and dev_acc=2.
Both use **100 sampled candidate design settings**.

$\text{card}(\mathcal{A}) = 270 > N_{\text{certification}}(\eta = 0.05, \delta = 10^{-3}) = 264$

Design parameter	min-value	max-value
N_pred	5	25
κ	1	10
ρ_f	1	10^3
rho_cstr	10^3	10^7
max_iter	5	20

Impact of dev_acc

Increasing dev_acc increases the number of eligible configurations!

Results: Admissible design settings

dev_acc = 1

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
1	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
2	2	5	2	2	7.937212e+06	1.243896	0.02	0.04	8	2437.847089	0.250
3	8	7	5	2	1.250875e+06	2.951172	0.02	0.16	18	3401.306977	0.500
4	9	6	7	7	9.170123e+06	8.804688	0.02	0.18	6	3755.852208	0.500

dev_acc = 2

	kappa	N_pred	n_steps	n_ctr	rho_cstr	rho_final	tau	tau_u	max_iter	cost	alpha
0	7	17	4	12	8.409123e+06	16.609375	0.02	0.14	5	3210.056159	0.250
1	9	7	3	8	9.646822e+06	178.800537	0.02	0.18	11	7859.487689	0.750
2	9	20	2	19	8.409123e+06	820.515021	0.02	0.18	8	3766.654430	0.250
3	6	17	3	16	6.124112e+06	3.778133	0.02	0.12	5	3284.618189	0.375
4	4	6	2	4	9.102911e+06	24.259781	0.02	0.08	16	4404.438416	0.625
5	9	5	8	4	6.259375e+05	2.951172	0.02	0.18	18	3126.158301	0.500
6	8	12	2	11	7.515030e+04	1.054939	0.02	0.16	17	2891.563097	0.375
7	7	9	5	2	2.442162e+06	15.537363	0.02	0.14	19	3583.879784	0.625
8	9	5	8	6	9.350676e+06	38.215650	0.02	0.18	18	4611.172897	0.625
9	2	12	2	2	8.846257e+06	1.146503	0.02	0.04	7	3408.564392	0.375
10	6	6	6	2	5.282910e+05	1.146503	0.02	0.12	18	3355.846137	0.375
11	9	18	2	13	3.536180e+06	125.875000	0.02	0.18	12	3250.814430	0.125
12	6	19	2	2	7.430228e+06	2.951172	0.02	0.12	14	3360.845813	0.125

Results of two different runs of the `Design_MPC` for two targeted device `dev_acc=1` and `dev_acc=2`.
Both use **100 sampled candidate design settings**.

$$\text{card}(\mathcal{A}) = 270 > N_{\text{certification}}(\eta = 0.05, \delta = 10^{-3}) = 264$$

Design parameter	min-value	max-value
`N_pred`	5	25
κ	1	10
ρ_f	1	10^3
`rho_cstr`	10^3	10^7
`max_iter`	5	20

Impact of `dev_acc`

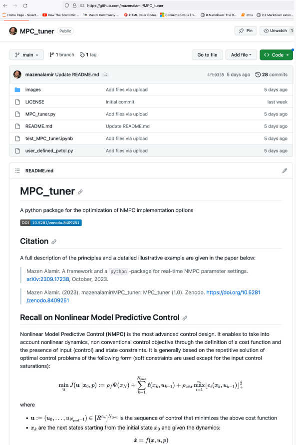
Increasing `dev_acc` increases the number of eligible configurations!

Large eligible possibilities of prediction horizons

$N_{\text{pred}} \in [0.2, 1.26]$ `dev_acc=1`
$N_{\text{pred}} \in [0.48, 3.24]$ `dev_acc=2`

Conclusion and future extensions

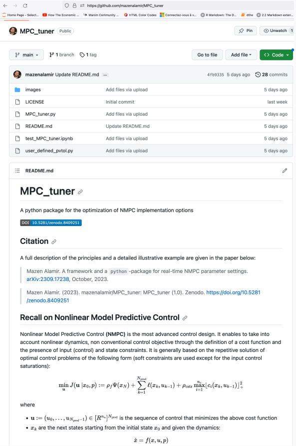
https://github.com/mazenalamir/MPC_tuner



✓ Design of NMPC setting: **A Relevant problem**!

The MPC_tuner package: Conclusion and future extensions

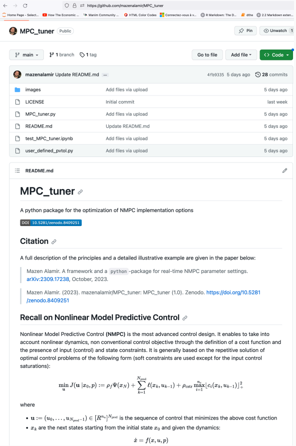
https://github.com/mazenalamir/MPC_tuner



✓ Design of NMPC setting: **A Relevant problem**!

Any of the discarded settings can be an intuitive choice of someone!

https://github.com/mazenalamir/MPC_tuner



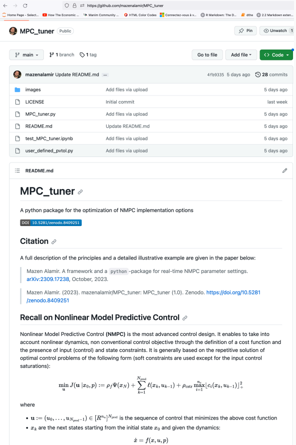
- ✓ Design of NMPC setting: **A Relevant problem**!

  Any of the discarded settings can be an intuitive choice of someone!

- ✓ The freely available `MPC_tuner` is a **first step**!

The MPC_tuner package: Conclusion and future extensions

https://github.com/mazenalamir/MPC_tuner



✓ Design of NMPC setting: **A Relevant problem**!

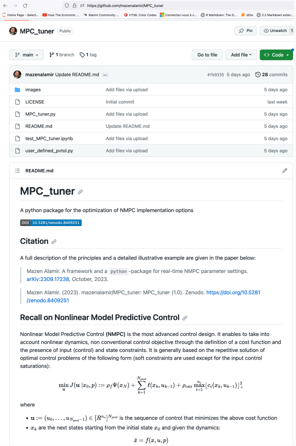
Any of the discarded settings can be an intuitive choice of someone!

✓ The freely available `MPC_tuner` is a **first step**!

Any comment or suggestion is more than welcome!

The MPC_tuner package: Conclusion and future extensions

https://github.com/mazenalamir/MPC_tuner



https://github.com/mazenalamir/MPC_tuner

MPC_tuner Public

main · 1 branch · 1 tag · Go to file · Add file · Code

mazenalamir Update README.md · 4fb9335 5 days ago · 28 commits

images	Add files via upload	5 days ago
LICENSE	Initial commit	last week
MPC_tuner.py	Add files via upload	5 days ago
README.md	Update README.md	5 days ago
test_MPC_tuner.ipynb	Add files via upload	5 days ago
user_defined_pvtol.py	Add files via upload	5 days ago

README.md

MPC_tuner

A python package for the optimization of NMPC implementation options

DOI 10.5281/zenodo.8409251

Citation

A full description of the principles and a detailed illustrative example are given in the paper below:

> Mazen Alamir. A framework and a `python` -package for real-time NMPC parameter settings. arXiv:2309.17238, October, 2023.

> Mazen Alamir. (2023). mazenalamir/MPC_tuner: MPC_tuner (1.0). Zenodo. https://doi.org/10.5281/zenodo.8409251

Recall on Nonlinear Model Predictive Control

Nonlinear Model Predictive Control **(NMPC)** is the most advanced control design. It enables to take into account nonlinear dynamics, non conventional control objective through the definition of a cost function and the presence of input (control) and state constraints. It is generally based on the repetitive solution of optimal control problems of the following form (soft constraints are used except for the input control saturations):

$$\min_{\mathbf{u}} J(\mathbf{u}\,|x_0,p) := \rho_f\Psi(x_N) + \sum_{k=1}^{N_{\text{pred}}}\ell(x_k,u_{k-1}) + \rho_{\text{cstr}}\max_{i=1}^{n_c}\lfloor c_i(x_k,u_{k-1})\rfloor_+^2$$

where

- $\mathbf{u} := (u_0,\dots,u_{N_{\text{pred}}-1}) \in [R^{n_u}]^{N_{\text{pred}}}$ is the sequence of control that minimizes the above cost function
- x_k are the next states starting from the initial state x_0 and given the dynamics:

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

- ✓ Design of NMPC setting: **A Relevant problem**!

  Any of the discarded settings can be an intuitive choice of someone!

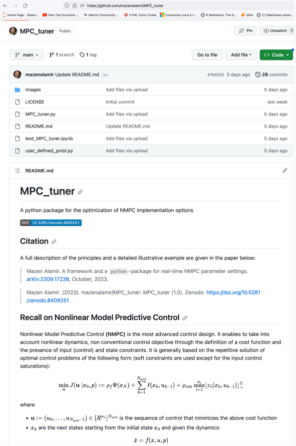
- ✓ The freely available `MPC_tuner` is a **first step**!

  Any comment or suggestion is more than welcome!

- ✓ **Extension 1**: Include fast-gradient option.

The MPC_tuner package: Conclusion and future extensions

https://github.com/mazenalamir/MPC_tuner



✓ Design of NMPC setting: **A Relevant problem**!

Any of the discarded settings can be an intuitive choice of someone!

✓ The freely available MPC_tuner is a **first step**!

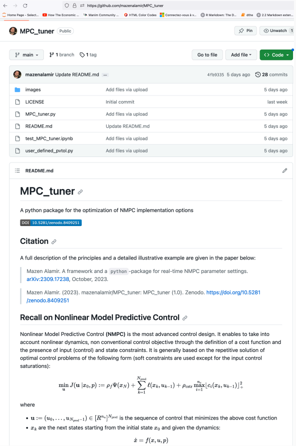
Any comment or suggestion is more than welcome!

✓ **Extension 1**: Include fast-gradient option.

✓ **Extension 2**: Include other solvers inside CasADi

The `MPC_tuner` package: Conclusion and future extensions

https://github.com/mazenalamir/MPC_tuner



- ✓ Design of NMPC setting: **A Relevant problem**!

  Any of the discarded settings can be an intuitive choice of someone!

- ✓ The freely available `MPC_tuner` is a **first step**!

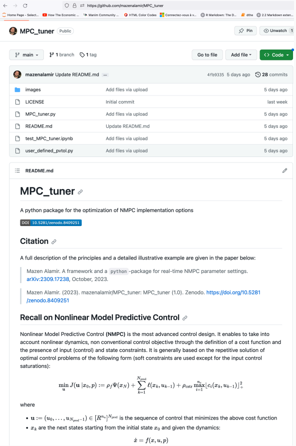
  Any comment or suggestion is more than welcome!

- ✓ **Extension 1**: Include fast-gradient option.

- ✓ **Extension 2**: Include other solvers inside `CasADi`

  The current implementation uses exclusively IPOPT

https://github.com/mazenalamir/MPC_tuner



MPC_tuner Public

main · 1 branch · 1 tag · Go to file · Add file · Code

mazenalamir Update README.md · 4fb9335 5 days ago · 28 commits

images	Add files via upload	5 days ago
LICENSE	Initial commit	last week
MPC_tuner.py	Add files via upload	5 days ago
README.md	Update README.md	5 days ago
test_MPC_tuner.ipynb	Add files via upload	5 days ago
user_defined_pvtol.py	Add files via upload	5 days ago

README.md

MPC_tuner

A python package for the optimization of NMPC implementation options

DOI 10.5281/zenodo.8409251

Citation

A full description of the principles and a detailed illustrative example are given in the paper below:

> Mazen Alamir. A framework and a `python` -package for real-time NMPC parameter settings. arXiv:2309.17238, October, 2023.

> Mazen Alamir. (2023). mazenalamir/MPC_tuner: MPC_tuner (1.0). Zenodo. https://doi.org/10.5281/zenodo.8409251

Recall on Nonlinear Model Predictive Control

Nonlinear Model Predictive Control **(NMPC)** is the most advanced control design. It enables to take into account nonlinear dynamics, non conventional control objective through the definition of a cost function and the presence of input (control) and state constraints. It is generally based on the repetitive solution of optimal control problems of the following form (soft constraints are used except for the input control saturations):

$$\min_{\mathbf{u}} J(\mathbf{u}\,|x_0,p) := \rho_f\Psi(x_N) + \sum_{k=1}^{N_{pred}} \ell(x_k,u_{k-1}) + \rho_{cstr}\max_{i=1}^{n_c}\lfloor c_i(x_k,u_{k-1})\rfloor_+^2$$

where

- $\mathbf{u} := (u_0,\dots,u_{N_{pred}-1}) \in [R^{n_u}]^{N_{pred}}$ is the sequence of control that minimizes the above cost function
- x_k are the next states starting from the initial state x_0 and given the dynamics:

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

✓ Design of NMPC setting: **A Relevant problem**!

Any of the discarded settings can be an intuitive choice of someone!

✓ The freely available `MPC_tuner` is a **first step**!

Any comment or suggestion is more than welcome!

✓ **Extension 1**: Include fast-gradient option.

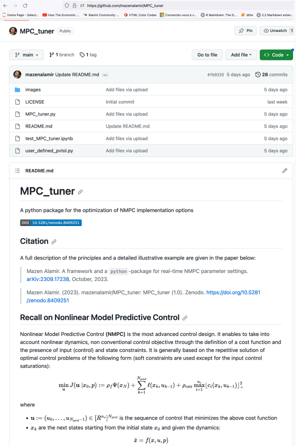
✓ **Extension 2**: Include other solvers inside `CasADi`

The current implementation uses exclusively IPOPT

✓ **Extension 3**: Include time varying weighting and penalties on the state excursion.

The MPC_tuner package: Conclusion and future extensions

https://github.com/mazenalamir/MPC_tuner



- ✓ Design of NMPC setting: **A Relevant problem**!

  Any of the discarded settings can be an intuitive choice of someone!

- ✓ The freely available `MPC_tuner` is a **first step**!

  Any comment or suggestion is more than welcome!

- ✓ **Extension 1**: Include fast-gradient option.

- ✓ **Extension 2**: Include other solvers inside `CasADi`

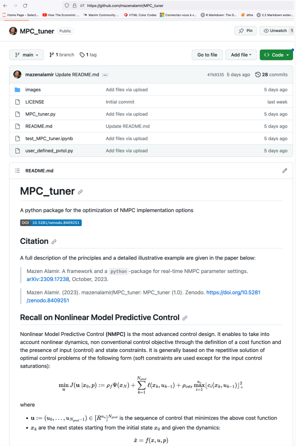
  The current implementation uses exclusively IpOpt

- ✓ **Extension 3**: Include time varying weighting and penalties on the state excursion.

- ✓ **Extension 4**: Accelerate the heuristic using **ML**!

The MPC_tuner package: Conclusion and future extensions

https://github.com/mazenalamir/MPC_tuner



✓ Design of NMPC setting: **A Relevant problem**!

Any of the discarded settings can be an intuitive choice of someone!

✓ The freely available `MPC_tuner` is a **first step**!

Any comment or suggestion is more than welcome!

✓ **Extension 1**: Include fast-gradient option.

✓ **Extension 2**: Include other solvers inside `CasADi`

The current implementation uses exclusively IpOpt

✓ **Extension 3**: Include time varying weighting and penalties on the state excursion.

✓ **Extension 4**: Accelerate the heuristic using **ML**!

Each example needed $\approx$ 1hr under `MacBook Pro,2.4GHz intel Corei9`

...


Thank you!