NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Nonlinear MPC and stability

Cyber-Physical Control Systems

B. Guerreiro

2021

Course plan

Week	Subject	Assignment
1	M0 Course details and introduction	
2	M1.1 Discrete-time systems state model representation	P1 start
3	M1.2 Optimization and optimal control	HW1
4	M2.1 Introduction to model predictive control (MPC)	
5	M2.2 Constrained MPC design	HW2
6	M3.1 Nonlinear MPC design	
7	M3.2 Feasibility and Stability analysis of MPC design	HW3, P1 due
8	M4.1 Decentralized MPC design	P2 start
9	M4.2 Distributed MPC design	
10	M4.3 Networked control systems	HW4
11	Discussion and feedback of draft P2 paper	Draft P2 due
12	M5.1 Hybrid dynamic systems	
13	M5.2 MPC for hybrid dynamic systems	HW5
14	Final project 2 presentations and discussion	P2 due

Course plan

Week	Subject	Assignment
1	M0 Course details and introduction	
2	M1.1 Discrete-time systems state model representation	P1 start
3	M1.2 Optimization and optimal control	HW1
4	M2.1 Introduction to model predictive control (MPC)	
5	M2.2 Constrained MPC design	HW2
6	M3.1 Nonlinear MPC design	
7	M3.2 Feasibility and Stability analysis of MPC design	HW3, P1 due
8	M4.1 Decentralized MPC design	P2 start
9	M4.2 Distributed MPC design	
10	M4.3 Networked control systems	HW4
11	Discussion and feedback of draft P2 paper	Draft P2 due
12	M5.1 Hybrid dynamic systems	
13	M5.2 MPC for hybrid dynamic systems	HW5
14	Final project 2 presentations and discussion	P2 due

Outline

Outline

Nonlinear optimal control

Nonlinear MPC

Basic feasibility and stability notions

Feasibility and stability of Linear MPC

Feasibility and stability of Nonlinear MPC

Outline

Outline

Nonlinear optimal control

Nonlinear MPC

Basic feasibility and stability notions

Feasibility and stability of Linear MPC

Feasibility and stability of Nonlinear MPC

Outline

Outline

Problem definition

Problem (Nonlinear optimal control problem)

Find the optimal control sequence u_k^* for $k = 0, \ldots, N-1$ that drives the system along a state trajectory x_k^* for $k = 0, \ldots, N$ according to its system dynamics and within the input and state constraints, while minimizing the performance index J_0. That is,

$$
\begin{aligned}
\min_{u_0,\ldots,u_{N-1}} \quad & J_0 = p(x_N) + \sum_{k=0}^{N-1} q(x_k, u_k) \\
s.t. \quad & x_{k+1} = f(x_k, u_k) \ , \ \forall_{k=0,\ldots,N-1} \ , \ x_0 = z_0 \\
& x_k \in \mathcal{X} \ , \ u_k \in \mathcal{U} \ , \ \forall_{k=0,\ldots,N-1} \\
& x_N \in \mathcal{X}_f
\end{aligned}
$$

Reformulating the optimization problem

- **Idea:** what if we include the equality constraint in the cost function?
- **Simple method:** for each k, use the inner product of the Lagrange multiplier $\lambda_k \in \mathbb{R}^n$ by the respective equality constraint.
- Resulting unconstrained optimization problem:

$$\min_{u} \quad J_0' := p(x_N) + \sum_{k=0}^{N-1} [q(x_k, u_k) + \lambda_{k+1}^T (f(x_k, u_k) - x_{k+1})]$$

- Define the Hamiltonian (consider a quadratic cost, for simplicity):

$$\begin{aligned} H(x_k, u_k) &= q(x_k, u_k) + \lambda_{k+1}^T f(x_k, u_k) \\ &= \frac{1}{2} x_k^T Q x_k + \frac{1}{2} u_k^T R u_k + \lambda_{k+1}^T f(x_k, u_k) \end{aligned}$$

Reformulating the cost functional

- New cost functional

$$J_0' = p(x_N) + \sum_{k=0}^{N-1} [H(x_k, u_k) - \lambda_{k+1}^T x_{k+1}]$$
$$= p(x_N) - \lambda_N^T x_N + \sum_{k=1}^{N-1} [H(x_k, u_k) - \lambda_k^T x_k] + H(x_0, u_0)$$

- First-order approximation for increments of J_0':

$$dJ_0' = (p_{x_N} - \lambda_N^T) dx_N + H_{u_0} du_0 + \sum_{k=1}^{N-1} [(H_{x_k} - \lambda_k^T) dx_k + H_{u_k} du_k] + \sum_{k=1}^{N} (H_{\lambda_k} - x_k^T) d\lambda_k$$

Optimality conditions

- Relevant derivatives:

$$H_{x_k} := \frac{dH(x_k, u_k)}{dx_k^T} = q_{x_k} + \lambda_{k+1}^T f_{x_k} \; , \qquad p_{x_N} := \frac{dp(x_N)}{dx_N^T} \; , \qquad f_{x_k} := \frac{df(x_k)}{dx_k^T}$$

$$H_{u_k} := \frac{dH(x_k, u_k)}{du_k^T} = q_{u_k} + \lambda_{k+1}^T f_{u_k} \; , \quad H_{\lambda_{k+1}} := \frac{dH(x_k, u_k)}{d\lambda_{k+1}^T} \; , \quad f_{u_k} := \frac{df(x_k)}{du_k^T}$$

- Necessary conditions of optimality ($dJ_0' = 0$):

$$(H_{\lambda_{k+1}} - x_{k+1}^T) d\lambda_{k+1} = 0 \; , \; \forall_{k=0,\ldots,N-1}$$

$$(H_{x_k} - \lambda_k^T) dx_k = 0 \; , \; \forall_{k=1,\ldots,N-1} \qquad (p_{x_N} - \lambda_N^T) dx_N = 0$$

$$H_{u_k} du_k = 0 \; , \; \forall_{k=0,\ldots,N-1} \qquad H_{x_0} dx_0 = 0$$

Optimal controller

- State equation (from $(H_{\lambda_{k+1}} - x_{k+1}^T)d\lambda_{k+1} = 0$):

$$x_{k+1} = H_{\lambda_{k+1}}^T = f(x_k, u_k)$$

- Co-state equation (from $(H_{x_k} - \lambda_k^T)dx_k = 0$):

$$\lambda_k = H_{x_k}^T = Qx_k + f_{x_k}^T \lambda_{k+1}$$

- Stationary condition (from $H_{u_k} du_k = 0$):

$$0 = H_{u_k}^T = Ru_k + f_{u_k}^T \lambda_{k+1} \quad \Leftrightarrow \quad u_k = -R^{-1} f_{u_k}^T \lambda_{k+1}$$

- Boundary conditions:

$$(P_N x_N - \lambda_N^T)dx_N = 0 \qquad (Q_0 x_0)^T dx_0 = 0$$

Finding the solution

- Resulting control law:

$$u_k = -R^{-1} f_{u_k} \lambda_{k+1}$$

- The recursive form of the optimal controller is

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) = f(x_k, -R^{-1} f_{u_k} \lambda_{k+1}) \qquad &, x_0 = z_0 \\ \lambda_k &= Q x_k + f_{x_k}^T \lambda_{k+1} \qquad &, \lambda_N = P x_N \end{aligned}$$

- Usually not possible to find closed form solution.
- Approach also known as Pontryagin maximum principle.
- Numeric solutions can be used.

Example 3.1

Problem (Example 3.1 - Simple nonlinear scalar system)

Find the optimal control sequence u_k^* *for* $k \in \{0, \ldots, N-1\}$ *that drives the system along a state trajectory* x_k^* *for* $k \in \{0, \ldots, N\}$ *according to the system dynamics,* $x_{k+1} = a x_k + b u_k + c$*, with initial state* $x_0 = r_0$ *and free final state, to minimize the performance index*

$$J_0 = \frac{1}{2}(x_N - r_N)^2 + \sum_{k=0}^{N-1} \frac{r}{2} u_k^2 \, .$$

Note: use the Hamiltonian approach (Pontryagin max. principle.)

Example 3.1

Example 3.1 - Simple nonlinear scalar system (part 1)

Define the Hamiltonian
$H(x_k, u_k) = q(u_k) + \lambda_{k+1}^T f(x_k, u_k) = \frac{r}{2}u_k^2 + \lambda_{k+1}(ax_k + bu_k + c)$, which results in the new cost functional

$$J_0' = \frac{1}{2}(x_N - r_N)^2 + \sum_{k=0}^{N-1}\left[\frac{r}{2}u_k^2 + \lambda_{k+1}\left(ax_k + bu_k + c - x_{k+1}\right)\right]$$

Noting that

$$H_{x_k} = q_{x_k} + \lambda_{k+1} f_{x_k} = a\lambda_{k+1} \qquad H^k_{\lambda_{k+1}} = f(x_k, u_k) = ax_k + bu_k + c$$

$$H_{u_k} = q_{u_k} + \lambda_{k+1} f^k_{u_k} = ru_k + b\lambda_{k+1} \qquad p^N_{x_N} = x_N - r_N$$

Example 3.1

Example 3.1 - Simple nonlinear scalar system (part 2)

The first order conditions of optimality are

$$0 = H_{u_k} = ru_k + b\lambda_{k+1} \qquad x_{k+1} = H_{\lambda_{k+1}} = ax_k + bu_k + c$$
$$\lambda_k = H_{x_k} = a\lambda_{k+1} \qquad \lambda_N = p_{x_N} = x_N - r_N$$

From the stationary condition, we have that $u_k = -\frac{b}{r}\lambda_{k+1}$, and considering $\gamma = \frac{b^2}{r}$ the state equation is

$$x_{k+1} = ax_k - \gamma\lambda_{k+1} + c$$

Example 3.1

Example 3.1 - Simple nonlinear scalar system (part 3)

Thus, the costate recursion is

$$\begin{aligned}
\lambda_N &= x_N - r_N \\
\lambda_{N-1} &= a(x_N - r_N) \\
\lambda_{N-2} &= a^2(x_N - r_N) \\
&\vdots \\
\lambda_1 &= a^{N-1}(x_N - r_N)
\end{aligned}$$

yielding the state equation

$$x_{k+1} = ax_k - \gamma a^{N-k-1}(x_N - r_N) + c$$

Example 3.1

Example 3.1 - Simple nonlinear scalar system (part 4)

and the state recursion is

$$
\begin{aligned}
x_1 &= ax_0 - \gamma\lambda_1 + c = ax_0 - \gamma a^{N-1}(x_N - r_N) + c \\
x_2 &= ax_1 - \gamma\lambda_2 + c = a^2x_0 - \gamma a^{N}(1 + a^{-2})(x_N - r_N) + c(1 + a) \\
x_3 &= ax_2 - \gamma\lambda_3 + c = a^3x_0 - \gamma a^{N+1}(1 + a^{-2} + a^{-4})(x_N - r_N) + c(1 + a + a^2) \\
&\vdots \\
x_k &= a^k x_0 - \gamma a^{N+k-2}(x_N - r_N)\sum_{i=0}^{k-1} a^{-2i} + c\sum_{i=0}^{k-1} a^i \\
&= a^k x_0 - \gamma a^{N+k-2}\frac{1 - a^{-2k}}{1 - a^{-2}}(x_N - r_N) + c\frac{1 - a^k}{1 - a}
\end{aligned}
$$

Example 3.1

Example 3.1 - Simple nonlinear scalar system (part 5)

The state recursion for the final state is then

$$
\begin{aligned}
x_N &= a^N x_0 - \gamma a^{2(N-1)} \frac{1-a^{-2N}}{1-a^{-2}} (x_N - r_N) + c\frac{1-a^N}{1-a} \\
&= a^N x_0 - \Lambda(x_N - r_N) + C
\end{aligned}
$$

with

$$
\Lambda = \gamma a^{2(N-1)} \frac{1-a^{-2N}}{1-a^{-2}} \quad , \quad C = c\frac{1-a^N}{1-a} \ .
$$

Example 3.1

Example 3.1 - Simple nonlinear scalar system (part 6)

Thus, the final state x_N can now be computed as

$$x_N = \frac{a^N x_0 + \Lambda r_N + C}{1 + \Lambda}$$

yielding the costate and optimal control law

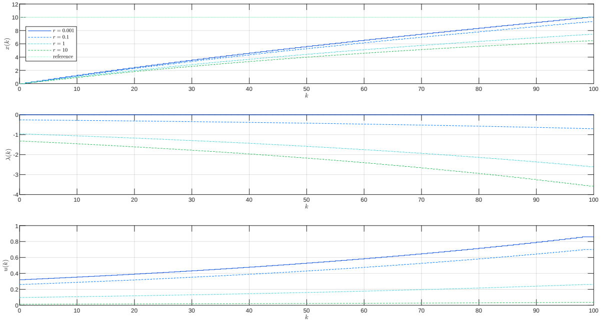
$$\lambda_{k+1} = a^{N-k-1}(x_N - r_N) = \frac{a^{N-k-1}}{1+\Lambda}(a^N x_0 - r_N + C)$$

$$u_k^* = -\frac{b\, a^{N-k-1}}{r(1+\Lambda)}(a^N x_0 - r_N + C)$$

Example 3.1

Example 3.1 - Simple nonlinear scalar system (part 7)

Matlab simulation results, with $a = 0.99$, $b = 0.1$, $c = 0.1$, $x_0 = 0$, $r_N = 10$, and r varying between 0.001 and 10:



Solution via batch approach

- Consider the state and control sequences, X, and U.
- The equivalent optimization problem for a quadratic cost can be reformulated as

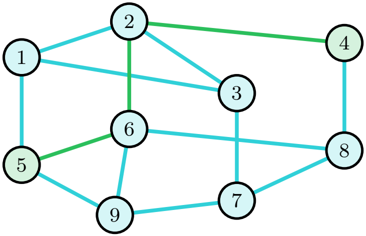
$$\begin{aligned} \min_{U,X} \quad & J_0 = X^T \bar{Q} X + U^T \bar{R} U \\ \text{s.t.} \quad & F(X,U) = 0 \\ & G(X,U) \leq 0 \end{aligned}$$

  where $\bar{Q} = \operatorname{diag}(Q, Q, \ldots, Q, P)$ and $\bar{R} = \operatorname{diag}(R, R, \ldots, R)$.
- The state sequence is now a part of the optimization variables.
- Constraints $F(X,U) = 0$ and $G(X,U) \leq 0$ may be nonlinear.

Dynamic programming approach

- Principle of optimality: Any tail of an optimal trajectory is optimal too.
- General cost-to-go from time i to horizon N:

$$J_i = J_{i \to N} = p(x_N) + \sum_{k=i}^{N-1} q(x_k, u_k)$$



- Optimal cost-to-go for nonlinear problem is

$$\begin{aligned} J_i^* = &\min_{u_i} (q(x_i, u_i) + J_{i+1}^*) \\ &s.t. \quad x_{i+1} = f(x_i, u_i) \\ &s.t. \quad g(x_i, u_i) \leq 0 \end{aligned}$$

- A recursive solution for the optimal control problem can be found.
- The curse of dimensionality: augmenting the state dimensions influences exponentially the complexity of the problem.

Cost-to-go for nonlinear optimal control

- Optimal cost-to-go at time N:

$$J_N^*(x_N) := p(x_N) \quad \text{with} g(x_N) \leq 0$$

- Recursion at time $N-1$, with $P_N = P$:

$$\begin{aligned} J_{N-1}^*(x_{N-1}) := \min_{u_{N-1}} \quad & q(x_{N-1}, u_{N-1}) + J_N^*(x_N) \\ \text{s.t.} \quad & x_N = f(x_{N-1}, u_{N-1}) \\ & g(x_{N-1}, u_{N-1}) \leq 0 \end{aligned}$$

$$\vdots$$

$$\begin{aligned} J_0^*(x_0) := \min_{u_0} \quad & q(x_0, u_0) + J_1^*(x_1) \\ \text{s.t.} \quad & x_1 = f(x_0, u_0) \\ & g(x_0, u_0) \leq 0 \quad , \quad x_0 = z_0 \end{aligned}$$

Outline

Nonlinear MPC

Ingredients:

- Cost function
- Model, input, and state constraint
- Receding horizon

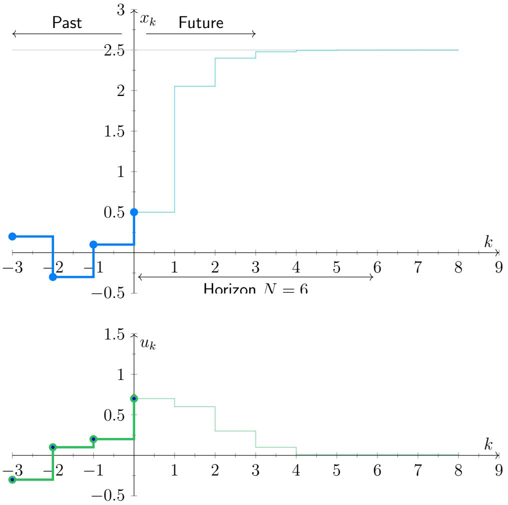
Solve optimization problem at each instant

$$\min_{u_i,\ldots,u_{i+N-1}} \quad J_i(x_i) = p(x_N) + \sum_{k=i}^{i+N-1} q(x_k, u_k)$$

$$s.t. \quad x_{k+1} = f(x_k, u_k) \ \forall_{k=i,\ldots,i+N-1}$$

$$f_{eq}(x_k, u_k) = 0$$

$$f_{in}(x_k, u_k) \leq 0$$



Nonlinear MPC

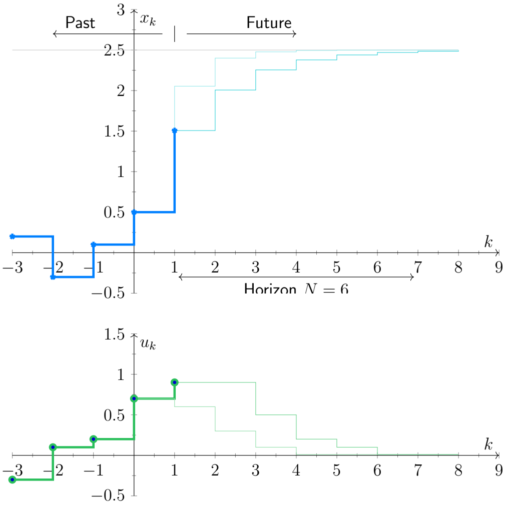
Ingredients:

- Cost function
- Model, input, and state constraint
- Receding horizon

Solve optimization problem at each instant

$$\min_{u_i,\dots,u_{i+N-1}} \quad J_i(x_i) = p(x_N) + \sum_{k=i}^{i+N-1} q(x_k, u_k)$$

$$\begin{aligned} s.t. \quad & x_{k+1} = f(x_k, u_k)\ \forall_{k=i,\dots,i+N-1} \\ & f_{eq}(x_k, u_k) = 0 \\ & f_{in}(x_k, u_k) \leq 0 \end{aligned}$$



Nonlinear MPC

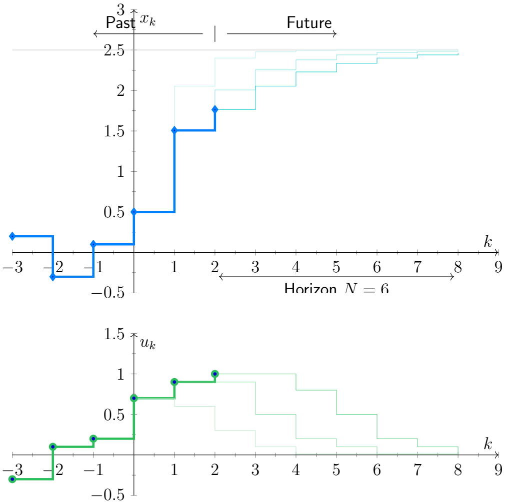
Ingredients:

- Cost function
- Model, input, and state constraint
- Receding horizon

Solve optimization problem at each instant

$$\min_{u_i,\dots,u_{i+N-1}} \quad J_i(x_i) = p(x_N) + \sum_{k=i}^{i+N-1} q(x_k, u_k)$$

$$\begin{aligned} s.t. \quad & x_{k+1} = f(x_k, u_k)\ \forall_{k=i,\dots,i+N-1} \\ & f_{eq}(x_k, u_k) = 0 \\ & f_{in}(x_k, u_k) \le 0 \end{aligned}$$



Nonlinear MPC

- Nonlinear predictive control problem:

$$\begin{aligned} \min_{U} \quad & J_0(x_0, U) = p(x_N) + \sum_{k=0}^{N-1} q(x_k, u_k) \\ \text{s.t.} \quad & x_{k+1} = f(x_k, u_k) \ , \ \forall_{k=0,\ldots,N-1} \ , \ x_0 = x_{t_k} \\ & x_k \in \mathcal{X} \ , \ u_k \in \mathcal{U} \ , \ \forall_{k=0,\ldots,N-1} \ , \ x_N \in \mathcal{X}_f \end{aligned}$$

- No equivalent batch formulation.
- Solve NLP optimization problem numerically.
- Obtain optimal control sequence: $U^* = \{u_0^*, u_1^*, \ldots, u_{N-1}^*\}$
- Receding horizon control policy: $u(t_k) = u_0^*$.

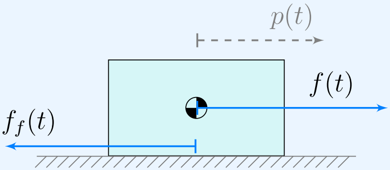
Receding horizon control algorithm

1. Consider the current state $x(t_k)$ as the initial condition x_0;
2. Obtain the optimal control sequence U^* solving the nonlinear optimal control problem;
3. If problem unfeasible, terminate algorithm;
4. Apply the first control action from U^* to the system, $u(t_k) = u_0^*$;
5. Repeat from 1 in the next time instant, t_{k+1}.

Example 3.2

Problem (Example 3.2: MPC for nonlinear car)

Consider a car evolving on an horizontal plane, where the only forces acting on the vehicle are the one applied by the engine and the air friction.



Compute the nonlinear model-based predictive controller considering a quadratic cost, with $N = 5$, $P = Q =$ *diag*$(10, 0.1)$, $R = 0.1$ *and initial condition* $x(0) = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$, *with model parameters* $\beta = 0.5$, $m = 1$, *and* $T_s = 0.1$.

Example 3.2

Example 3.2 - MPC for nonlinear car (part 1)

Recalling the Example 1.2 and considering the state $\mathbf{x}(t) = \begin{bmatrix} p(t) \\ v(t) \end{bmatrix}$, the input $u(t) = f(t)$ and the output $y(t) = p(t)$, results in the nonlinear state space model

$$\dot{x}(t) = f_c(x(t), u(t)) \quad , \quad x(0) = x_{ini}$$
$$y(t) = h_c(x(t), u(t))$$

where, considering $\beta = 0.5$, $m = 1$, and $T_s = 0.1$.

$$f_c(x(t), u(t)) = \begin{bmatrix} v(t) \\ -\frac{\beta}{m} v(t)^2 + \frac{1}{m} f(t) \end{bmatrix} \quad , \quad x_0 = \begin{bmatrix} p_0 \\ v_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$
$$h_c(x(t), u(t)) = Cx(t) \quad , \quad C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

Example 3.2

Example 3.2 - MPC for nonlinear car (part 2)

A simple discretization of the nonlinear system can be obtained using the forward Euler method, yielding

$$\begin{aligned} x(k+1) &= f(x(k), u(k)) \quad , \quad x(0) = x_{ini} \\ y(k) &= Cx(k) \end{aligned}$$

where

$$f(x(k), u(k)) = x(k) + T_s f_c(x_c(k), u(k))$$

Example 3.2

Example 3.2 - MPC for nonlinear car (part 3)

Using the batch approach, the equivalent optimization problem is

$$\min_{U,X} \quad J_0 = X^T\bar{Q}X + U^T\bar{R}U = \begin{bmatrix} X \\ U \end{bmatrix}^T \begin{bmatrix} \bar{Q} & 0 \\ 0 & \bar{R} \end{bmatrix} \begin{bmatrix} X \\ U \end{bmatrix} \qquad s.t. \quad F(X,U) = 0$$

where $\bar{Q} = \mathrm{diag}\,(Q,Q,Q,Q,Q,P)$ and $\bar{R} = \mathrm{diag}\,(R,R,R,R,R) = 0.1I_5$, while the state and input sequences are $X = \begin{bmatrix} x_0^T & \cdots & x_N^T \end{bmatrix}^T$, $U = \begin{bmatrix} u_0^T & \cdots & u_{N-1}^T \end{bmatrix}^T$. Regarding the batch equality constraint, we can write

$$F(X,U) = \begin{bmatrix} x_0 - x_{ini} \\ f(x_0,u_0) - x_1 \\ \dots \\ f(x_{N-1},u_{N-1}) - x_N \end{bmatrix}$$

Example 3.2

Example 3.2 - MPC for nonlinear car (part 4)

To solve this problem in each instant, we will use the function `fmincon`:

```
fun = @(XU) cost(XU,Qb,Rb);
nonlcon = @(XU) statecon(XU,x0,N,nx,nu);
XU_opt = fmincon(fun,XU0,[],[],[],[],[],[],nonlcon);
u_mpc = XU(nx*N+1+1:nx*N+nu);
```

For the definition of the cost in the function handler `fun`, linked to the function `cost`, we can use the code

```
function J = cost(XU,Qb,Rb)
    J = XU'*blkdiag(Qb,Rb)*XU;
end
```

Example 3.2

Example 3.2 - MPC for nonlinear car (part 5)

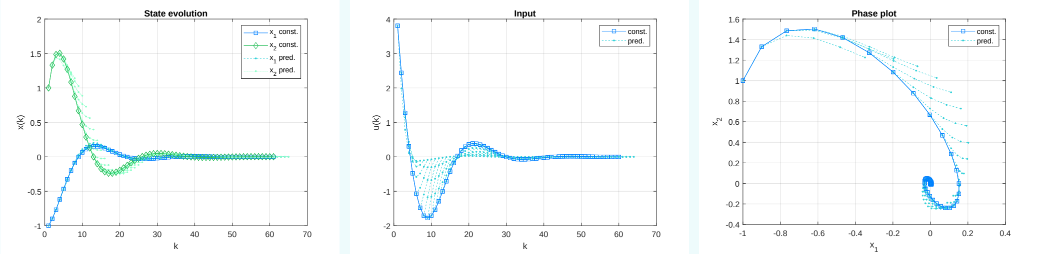
To define the nonlinear equality constraints we can use

```
function [c,ceq] = statecon(XU,x0,N,nx,nu)
    X = reshape( XU(1 : nx*(N+1))               , nx,[]);
    U = reshape( XU(nx*(N+1)+1:nx*(N+1)+nu*N)    , nu,[]);
    Ceq(:,1) = X(:,1) - x0;
    for k = 1:N
        Ceq(:,k+1) = f_car(X(:,k),U(:,k),MP) - X(:,k+1);
    end
    ceq = reshape(Ceq,[],1); c = [];
end
function xp = f_car(x,u)
    xdot = [x(2);u(1)-0.5*x(2)^2]; xp = x + 0.1*xdot;
end
```

Example 3.2

Example 3.2 - MPC for nonlinear car (part 6)

Simulation results for 60 sampling intervals:



Summary 3.1

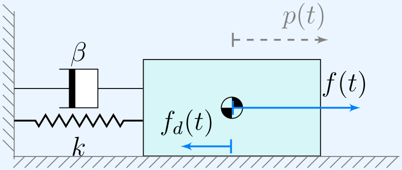
The main learning outcomes of this class are:

- Define nonlinear optimal control problems
- Know solution approaches to nonlinear optimal control problems
- implement and solve nonlinear MPC problems

Exercise 3.3

Problem (Exercise 3.3: MPC for nonlinear mass-spring-dumper)

Consider a nonlinear mass-spring-damper system shown below, where the position and velocity of the cart are denoted by $p(t)$ *and* $v(t)$*, respectively.*



Compute the nonlinear model-based predictive controller considering a quadratic cost, with $N = 5$, $P = Q =$ *diag*$(10, 0.1)$, $R = 0.1$ *and initial condition* $x(0) = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$*, with* $T_s = 0.1$.

Course plan

Week	Subject	Assignment
1	M0 Course details and introduction	
2	M1.1 Discrete-time systems state model representation	P1 start
3	M1.2 Optimization and optimal control	HW1
4	M2.1 Introduction to model predictive control (MPC)	
5	M2.2 Constrained MPC design	HW2
6	M3.1 Nonlinear MPC design	
7	M3.2 Feasibility and Stability analysis of MPC design	HW3, P1 due
8	M4.1 Decentralized MPC design	P2 start
9	M4.2 Distributed MPC design	
10	M4.3 Networked control systems	HW4
11	Discussion and feedback of draft P2 paper	Draft P2 due
12	M5.1 Hybrid dynamic systems	
13	M5.2 MPC for hybrid dynamic systems	HW5
14	Final project 2 presentations and discussion	P2 due

Course plan

Week	Subject	Assignment
1	M0 Course details and introduction	
2	M1.1 Discrete-time systems state model representation	P1 start
3	M1.2 Optimization and optimal control	HW1
4	M2.1 Introduction to model predictive control (MPC)	
5	M2.2 Constrained MPC design	HW2
6	M3.1 Nonlinear MPC design	
7	M3.2 Feasibility and Stability analysis of MPC design	HW3, P1 due
8	M4.1 Decentralized MPC design	P2 start
9	M4.2 Distributed MPC design	
10	M4.3 Networked control systems	HW4
11	Discussion and feedback of draft P2 paper	Draft P2 due
12	M5.1 Hybrid dynamic systems	
13	M5.2 MPC for hybrid dynamic systems	HW5
14	Final project 2 presentations and discussion	P2 due

Course plan

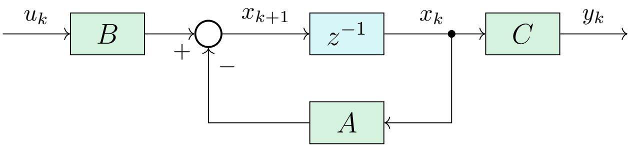
Week	Subject	Assignment
1	M0 Course details and introduction	
2	M1.1 Discrete-time systems state model representation	P1 start
3	M1.2 Optimization and optimal control	HW1
4	M2.1 Introduction to model predictive control (MPC)	
5	M2.2 Constrained MPC design	HW2
6	M3.1 Nonlinear MPC design	
7	M3.2 Feasibility and Stability analysis of MPC design	HW3, P1 due
8	M4.1 Decentralized MPC design	P2 start
9	M4.2 Distributed MPC design	
10	M4.3 Networked control systems	HW4
11	Discussion and feedback of draft P2 paper	Draft P2 due
12	M5.1 Hybrid dynamic systems	
13	M5.2 MPC for hybrid dynamic systems	HW5
14	Final project 2 presentations and discussion	P2 due

Outline

Challenges of this module

- Consider a model and respective MPC problem formulation.
- Feasibility:
  - Is the resulting optimization problem possible to solve?
  - If OK in the first iteration, is it feasible in the next iterations?
  - Under which conditions?
  - Do we have to compromise on performance?
- Stability:
  - Will the solution converge to the desired goal?
  - If OK in the first iteration, will it converge in the next iterations?
  - Under which conditions?
  - Do we have to compromise on performance?

Discrete-time linear dynamic systems



- LTI state-space non-autonomous form:

  $$x_{k+1} = Ax_k + Bu_k \quad , x_0 = x_{ini}$$

  where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$.
- Solution: $x_k = \phi(k)x_0 + \sum_{m=0}^{k-1} \phi(k-1-m)Bu_m$
- Without input the system is said to be autonomous, being defined as

  $$x_{k+1} = Ax_k \quad , x_0 = x_{ini}$$

- Autonomous solution: $x_k = \phi(k,0)x_0 = \phi(k-0)x_0 = \phi(k)x_0$

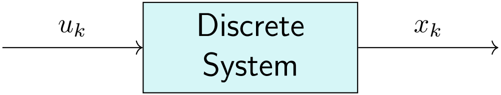
Lyapunov stability of linear systems

Definition ([1])

The discrete-time linear system, for every initial condition $x_{k_0} = x_0 \in \mathbb{R}^n$, is:

1. marginally stable in the Lyapunov/internal sense if the homogeneous state response $x_k = \phi(k, k_0)x_0$, for all $k_0 \geq 0$, is uniformly bounded;
2. asymptotically stable if, in addition, the homogeneous response satisfies $x_k \to 0$ as $k \to \infty$;
3. exponentially stable if, in addition, there exist constants $c > 0$, $\lambda < 1$ such that the homogeneous state response satisfies $\|x_k\| \leq c\lambda^{k-k_0}\|x_0\|$, for all $k_0 \geq 0$;
4. unstable if it is not marginally stable in the Lyapunov sense.

Discrete-time nonlinear dynamic systems



- Nonlinear state-space non-autonomous form:

$$x_{k+1} = f(x_k, u_k) \quad , x_0 = x_{ini}$$

  - $x_k \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state vector and respective admissible set.
  - $u_k \in \mathcal{U} \subseteq \mathbb{R}^m$ the control vector and respective admissible set.
- Solution: $x_k = \phi(k; x_0, U) = \phi(k; x_0, u_0, \ldots, u_{k-1})$
- Without input the system is said to be autonomous, being defined as

$$x_{k+1} = f(x_k) \quad , x_0 = x_{ini}$$

- Autonomous solution: $x_k = \phi(k; x_0)$

Stability of nonlinear systems

Definition ([3])

A discrete-time nonlinear autonomous system described above, for every initial condition $x_{k_0} = x_0 \in \mathbb{R}^n$, $k_0 \geq 0$, is said to be:

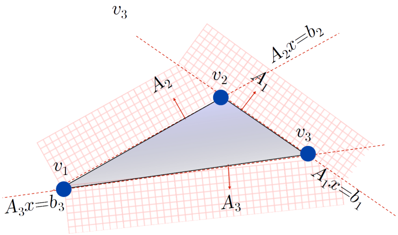
1. the origin is locally stable if for all $\varepsilon > 0$ there exists a $\delta > 0$ such that $\|x\| < \delta$ implies $\|\phi(k, k_0; x_0)\| < \varepsilon$ for all $k_0 \geq 0$;
2. globally attractive if $\|\phi(k, k_0; x_0)\| \to 0$ as $k \to \infty$ for all $x_0 \in \mathbb{R}^n$;
3. globally asymptotically stable if it is locally stable and globally attractive;
4. the origin is unstable if it is not locally stable.

Using polyhedra as sets

- Convex polyhedron: intersection of a finite set of half-spaces in $\mathbb{R}^n$.
- Convex polytope: bounded convex polyhedron.
- Hyperplane ($\mathcal{H}$-)representation:

$$\mathcal{P} = \{x \in \mathbb{R}^n : Hx \leq h\}$$

- Vertex ($\mathcal{V}$-)representation for polytopes:



$$\mathcal{P} = \{x \in \mathbb{R}^n : x = \sum_{i=1}^{q} \alpha_i v_i\}$$

where $v_i \in \mathbb{R}^n$ are the vertices of the polytope, $\alpha_i \geq 0$, and $\sum_{i=1}^{q} \alpha_i = 1$.

- Convex hull: transformation from $\mathcal{V}$ to $\mathcal{H}$ representation.
- Vertex enumeration: transformation from $\mathcal{H}$ to $\mathcal{V}$ representation.

Controlable sets

- Consider a nonlinear system $x_{k+1} = f(x_k, u_k)$ subject to $x_k \in \mathcal{X}$ and $u_k \in \mathcal{U}$, for all $k \geq 0$.
- The precursor set, Pre$(\mathcal{S})$, of set $\mathcal{S}$ is defined as

$$\text{Pre}(\mathcal{S}) = \{x \in \mathbb{R}^n : \exists u \in \mathcal{U} \; s.t. \; f(x,u) \in \mathcal{S}\}$$

- The successor set, Suc$(\mathcal{S})$, can also be similarly defined.
- The N-step controllable set, $\mathcal{K}_N(\mathcal{S})$, of a target set $\mathcal{S} \in \mathcal{X}$ is defined for the above system with the recursion

$$\mathcal{K}_j(\mathcal{S}) = \text{Pre}(\mathcal{K}_{j-1}(\mathcal{S})) \cap \mathcal{X} \;, \quad \mathcal{K}_0(\mathcal{S}) = \mathcal{S} \;, \quad j \in \{1, \ldots, N\}$$

- The maximal controllable set, $\mathcal{K}_{max}(\mathcal{S})$, of a target set $\mathcal{S} \in \mathcal{X}$ is defined for the above system as the union of all N-step controllable sets $\mathcal{K}_N(\mathcal{S})$ contained in $\mathcal{X}$, for $N \in \mathbb{N}$.
- Similar definitions can me made for autonomous systems.

Example 3.3

Problem (Example 3.3: computing the controllable set)

Consider the autonomous linear system $x_{k+1} = Ax_k$, *with* $A = \begin{bmatrix} 0.5 & 0 \\ 1 & -0.5 \end{bmatrix}$,
subject to the state constraint $x_k \in \mathcal{X}$, *where*
$\mathcal{X} = \{x \in \mathbb{R}^2 : -x_{max} \leq x \leq x_{max}\}$ *and* $x_{max} = \begin{bmatrix} 10 & 10 \end{bmatrix}^T$.
Compute the one-step controllable set to $\mathcal{X}$.

Example 3.3: computing the controllable set (part 1)

Notice by the definitions above, that the one-step controllable set to $\mathcal{X}$ is given by $\mathcal{K}_1(\mathcal{X}) = \text{Pre}(\mathcal{X}) \cap \mathcal{X}$. As such, we will compute first the predecessor $\text{Pre}(\mathcal{X})$ and then intersect the result with $\mathcal{X}$.

Example 3.3

Example 3.3: computing the controllable set (part 2)

The inequalities of the state constraint set can be redefined as $Hx \leq h$ using the procedure we have described in Module 4, where

$$H = \begin{bmatrix} -I_2 \\ I_2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad , \quad h = \begin{bmatrix} x_{max} \\ x_{max} \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \end{bmatrix}$$

By the definition of predecessor,

$$\begin{aligned} \text{Pre}(\mathcal{X}) &= \{x \in \mathbb{R}^2 : Ax \in \mathcal{X}\} \\ &= \{x \in \mathbb{R}^2 : HAx \leq h\} \end{aligned}$$

Example 3.3

Example 3.3: computing the controllable set (part 3)

which result in

$$\text{Pre}(\mathcal{X}) = \left\{ x : \begin{bmatrix} -A \\ A \end{bmatrix} x \leq \begin{bmatrix} x_{max} \\ x_{max} \end{bmatrix} \right\} = \left\{ x : \begin{bmatrix} -0.5 & 0 \\ -1 & 0.5 \\ 0.5 & 0 \\ 1 & -0.5 \end{bmatrix} x \leq \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \right\}$$

The previous expression might have redundant inequalities, which can be removed using the algorithms provided in [2] and in the course documentation, resulting in

$$\text{Pre}(\mathcal{X}) = \left\{ x : \begin{bmatrix} -1 & 0 \\ -1 & -0.5 \\ 1 & 0 \\ 1 & -0.5 \end{bmatrix} x \leq \begin{bmatrix} 20 \\ 10 \\ 20 \\ 10 \end{bmatrix} \right\}$$

Example 3.3

Example 3.3: computing the controllable set (part 4)

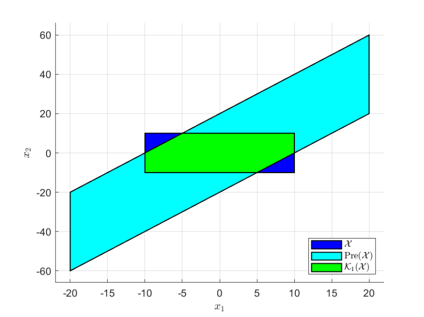
Finally, the on-step controllable set $\mathcal{K}_1(\mathcal{X})$ is computed with the intersection of Pre$(\mathcal{X})$ with $\mathcal{X}$, obtained simply by appending the constraints of both sets, simplifying the constrains in case of repeated or redundant inequalities. This results in

$$\mathcal{K}_1(\mathcal{X}) = \left\{ x : \begin{bmatrix} -A \\ A \\ H \end{bmatrix} x \leq \begin{bmatrix} x_{max} \\ x_{max} \\ h \end{bmatrix} \right\} = \left\{ x : \begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 1 \\ -1 & -0.5 \\ 1 & -0.5 \end{bmatrix} x \leq \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \right\}$$

Example 3.3

Example 3.3: computing the controllable set (part 5)

This type of computations can also be done in Matlab, using the opensource MPT toolbox (available here), as shown in the figure below.



Operations on sets

- For a constrained autonomous linear systems, the predecessor of the state constraint set is computed by:

  $$\text{Pre}(\mathcal{X}) = \mathcal{X} \circ A$$

  where $\circ$ denotes the convex hull, which if the set $\mathcal{X}$ is defined by the convex hull of its vertices, $\mathcal{X} = \text{convh}(V)$, is simply given by $\mathcal{X} \circ A = \text{convh}(VA)$.
- For a constrained linear systems, the predecessor of the state constraint set is computed by:

  $$\text{Pre}(\mathcal{X}) = (\mathcal{X} \oplus (-B \circ \mathcal{U})) \circ A$$

- The notation $\oplus$ stands for the Minkowski sum, which is given by

  $$\mathcal{P} \oplus \mathcal{Q} = \{p + q \in \mathbb{R}^n : p \in \mathcal{P}, q \in \mathcal{Q}\}$$

Invariant sets

- A set $\mathcal{O} \in \mathcal{X}$ is said to be positive invariant for the constrained autonomous nonlinear system if $x_0 \in \mathcal{O}$ implies that $x_k \in \mathcal{O}$ for all $k > 0$.
- A set $\mathcal{O}_{max} \in \mathcal{X}$ is said to be the maximal positive invariant set for the constrained autonomous nonlinear system if it is invariant and contains all the invariant sets contained in $\mathcal{X}$.
- A set $\mathcal{C} \in \mathcal{X}$ is said to be control invariant for the constrained nonlinear system if $x_k \in \mathcal{C}$ implies that there exists $u_k \in \mathcal{U}$ such that $f(x_k, u_k) \in \mathcal{C}$ for all $k > 0$.
- A set $\mathcal{C}_{max} \in \mathcal{X}$ is said to be the maximal control invariant set for the constrained nonlinear system if it is control invariant and contains all the control invariant sets contained in $\mathcal{X}$.

Computation of control invariant sets

Algorithm 1: Maximal control invariant set

Data: f, $\mathcal{X}$, $\mathcal{U}$
Result: $\mathcal{C}_{max}$
$\Omega_0 \leftarrow \mathcal{X}$; $i \leftarrow -1$;
repeat
  $i \leftarrow i + 1$;
  $\Omega_{i+1} \leftarrow \mathsf{Pre}(\Omega_i) \cap \Omega_i$;
until $\Omega_{i+1} = \Omega_i$;
$\mathcal{C}_{max} \leftarrow \Omega_i$;

- The set $\mathcal{C}_{max}$ is finitely determined if and only if there is $i \in \mathbb{N}$ such that, in the above algorithm, $\Omega_{i+1} = \Omega_i$.
- The $\mathcal{C}_{max}$ determinedness index is the smallest $i \in \mathbb{N}$ such that $\Omega_{i+1} = \Omega_i$.

Computation of maximal controllable set

Algorithm 2: Maximal controllable set

Data: f, $\mathcal{X}$, $\mathcal{U}$

Result: $\mathcal{K}_{max}(\mathcal{O})$

$\mathcal{K}_0 \leftarrow \mathcal{O}$, where $\mathcal{O}$ is a control invariant set; $i \leftarrow -1$;

repeat

$\quad i \leftarrow i + 1$;

$\quad \mathcal{K}_{i+1} \leftarrow \text{Pre}(\mathcal{K}_i) \cap \mathcal{X}$;

until $\mathcal{K}_{i+1} = \mathcal{K}_i$;

$\mathcal{K}_{max}(\mathcal{O}) \leftarrow \mathcal{K}_k$;

- The set $\mathcal{K}_{max}(\mathcal{O})$ is finitely determined if and only if there is $i \in \mathbb{N}$ such that, in the above algorithm, $\mathcal{K}_{i+1} = \mathcal{K}_i$.
- The $\mathcal{K}_{max}(\mathcal{O})$ determinedness index, $\bar{N}$, is the lowest i such that $\mathcal{K}_{i+1} = \mathcal{K}_i$.

Example 3.4

Problem (Example 3.4: computing the maximal controllable set)

Consider the linear system

$$x_{k+1} = Ax_k + Bu_k \text{ , with } A = \begin{bmatrix} 1.5 & 0 \\ 1 & -1.5 \end{bmatrix} \text{ and } B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

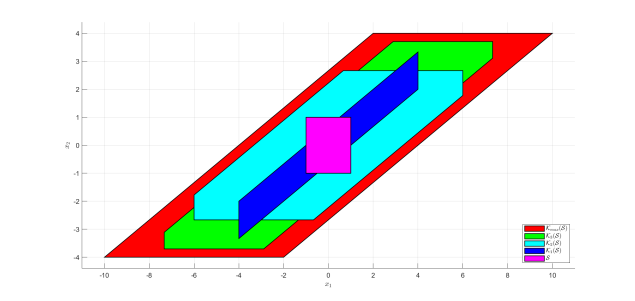
subject to the constraints $x_k \in \mathcal{X} = \{x \in \mathbb{R}^2 : -x_{max} \leq x \leq x_{max}\}$ *and* $u_k \in \mathcal{U} = \{u \in \mathbb{R} : -u_{max} \leq x \leq u_{max}\}$, *where* $x_{max} = \begin{bmatrix} 10 & 10 \end{bmatrix}^T$ *and* $u_{max} = 5$. *Compute the maximal controllable set and the determinedness index, if any, considering the target set*

$$\mathcal{S} = \left\{ x \in \mathbb{R}^2 : -\begin{bmatrix} 1 \\ 1 \end{bmatrix} \leq x \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}.$$

Example 3.4

Example 3.4: computing the maximal controllable set

This type of computations can be done in Matlab, using the opensource MPT toolbox (available here), as shown in the figure below, whereas the computations can be found in Exercise Book. The computed determinedness index is 35.



Outline

Linear MPC problem

- Linear MPC problem, $\mathcal{P}_N(x_0)$:

$$J_0^*(x_{t_k}) = \min_U \quad J_0(x_0, U) = p(x_N) + \sum_{k=0}^{N-1} q(x_k, u_k)$$

$$\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k \ , \ \forall_{k=0,\ldots,N-1} \ , \ x_0 = x_{t_k}$$

$$x_k \in \mathcal{X} \ , \ u_k \in \mathcal{U} \ , \ \forall_{k=0,\ldots,N-1} \ , \ x_N \in \mathcal{X}_f$$

- RHC law: $u_{t_k} = f_{RHC}(x_{t_k}) = u_0^*(x_{t_k})$
- Resulting closed-loop (autonomous) system:

$$x_{t_{k+1}} = Ax_{t_k} + Bf_{RHC}(x_{t_k}) = f_{cl}(x_{t_k})$$

Set of feasible states

- Batch approach, starting from i to N, with $U_i = \{u_i, \ldots, u_{N-1}\}$:
  - Set of feasible input sequences:

$$\mathcal{U}_i = \{U_i \in \mathbb{R}^{Nm} | x_k \in \mathcal{X}, u_k \in \mathcal{U}, x_{k+1} = f(x_k, u_k), \text{ for all } k = i, \ldots, N-1 \text{ and } x_N \in \mathcal{X}_f\}$$

  - Set of feasible initial conditions for the MPC problem $\mathcal{P}_N(x_0)$:

$$\mathcal{X}_i = \{x \in \mathbb{R}^n | \exists U_i \text{ such that } x \in \mathcal{X} \text{ and } U_i \in \mathcal{U}_i\}$$

- Recursive approach:

$$\mathcal{X}_N = \mathcal{X}_f$$
$$\mathcal{X}_i = \text{Pre}(\mathcal{X}_{i+1}) \cap \mathcal{X}\ , \quad i = N-1, \ldots, 0$$

- Note that the algorithm to compute $\mathcal{X}_i$ is similar to that of the N-step controllable set defined above, with target set $\mathcal{X}_f$.

Linear MPC – feasibility

Corollary (Cor. 12.1 in [2])

Consider the RHC law defined above with $N \geq 1$*. If there exist* $i \in [1, N]$ *such that* $\mathcal{X}_i$ *is a control invariant set for the considered linear system, then the RHC is persistently feasible for all cost functions.*

Corollary (Cor. 12.2 in [2])

Consider the RHC law defined above. If N *is greater than the determinedness index* $\bar{N}$ *of the maximal controllable set* $\mathcal{K}_{max}(\mathcal{X}_f)$ *for the considered linear system, then the RHC is persistently feasible.*

- **Important**: persistent feasibility does not imply convergence!

Linear MPC – stability of the origin

Theorem (Stability of the origin [2])

A discrete-time linear system, the linear MPC control law, and the respective closed loop system described above. Assume that

(A1) *The stage and terminal costs, $q(x,u)$ and $p(x)$, are positive definite and continuous.*

(A2) *The sets $\mathcal{X}$, $\mathcal{X}_f$, and $\mathcal{U}$ contain the origin and are closed.*

(A3) *The set $\mathcal{X}_f$ is control invariant and $\mathcal{X}_f \subseteq \mathcal{X}$.*

(A4) *For all $x \in \mathcal{X}_f$, there exists $u \in \mathcal{U}$ such that $Ax + Bu \in \mathcal{X}_f$ and*

$$p(Ax + Bu) - p(x) \leq -q(x,u)$$

Then, the origin of the closed loop system is asymptotically stable in $\mathcal{X}_0$ (the domain of attraction).

Example 3.5

Problem (Example 3.5: MPC analysis for double integrator)

Consider double pendulum time-invariant discrete linear system described by in Example 4.3, and the respective regulation MPC control law, with $Q = I$, $R = 0.01$, *and* P *the resulting matrix of an infinite horizon LQR. Consider also the input constraint* $-1 \leq u(k) \leq 1$ *and the state constraint* $-x_{max} \leq x(k) \leq x_{max}$ *for all* $k \geq 0$, *where* $x_{max} = \begin{bmatrix} 10 & 10 \end{bmatrix}^T$. *Analyze the closed loop, in particular the region of attraction,* $\mathcal{X}_0$, *when:*

(a) *The horizon is* $N = 2$ *and the terminal constraint is* $\mathcal{X}_f = 0$.

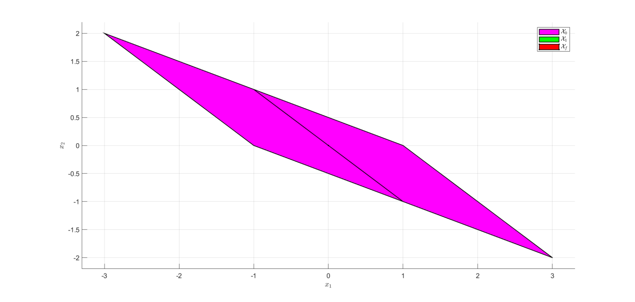
(b) *The horizon is* $N = 2$ *and the terminal constraint is* $\mathcal{X}_f$ *is the positive invariante set resulting from using the LQR control law.*

(c) *No terminal constraint, and the horizon is* $N = 6$.

Example 3.5

Example 3.5: MPC analysis for double integrator (part 1)

All cases result in recursive feasibility and stability.
Case (a): The terminal set is $\mathcal{X}_f = \{x \in \mathbb{R}^2 : x = 0\}$, which using the algorithm to compute the i-th feasibility set, $\mathcal{X}_i$ for the MPC problem results in the MPC feasibility set $\mathcal{X}_0$ presented below (computations in Exercise book).



Example 3.6

Example 3.6: MPC analysis for double integrator (part 2)

Case (b): The terminal set $\mathcal{X}_f$ can be obtained by solving the algebraic Riccati equation detailed in Module 2, from which we can compute the respective gain, K_∞, and the resulting closed loop system, $x_{k+1} = (A - BK_\infty)x_k$. Computing the invariant set to this system, we can find $\mathcal{X}_f$, in this case, given by

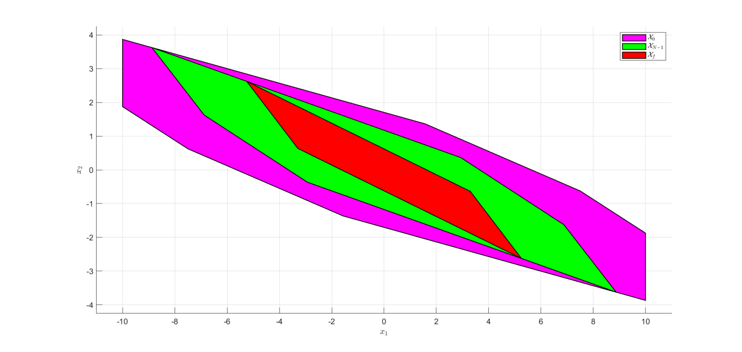
$$\mathcal{O}_\infty = \left\{ x : \begin{bmatrix} 0.37 & 0.37 \\ -0.37 & -0.37 \\ -0.61 & -1.61 \\ 0.61 & 1.61 \end{bmatrix} x \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\}$$

which using the algorithm for computation of the i-th feasibility set, $\mathcal{X}_i$ for the MPC problem defined in the slides.

Example 3.6

Example 3.6: MPC analysis for double integrator (part 3)

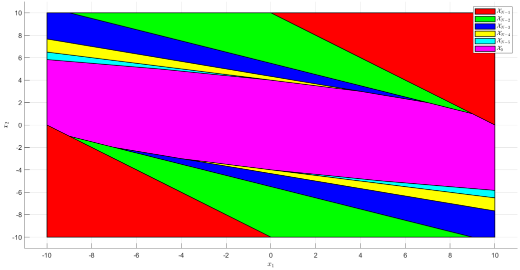
This results in the MPC feasibility set $\mathcal{X}_0$ presented below



Example 3.6

Example 3.6: MPC analysis for double integrator (part 4)

Case (c): The terminal set is $\mathcal{X}_f = \{x \in \mathbb{R}^2\}$ and $N = 6$, which using the algorithm for computation of the i-th feasibility set, $\mathcal{X}_i$ for the MPC problem results in the MPC feasibility set $\mathcal{X}_0$ presented below.



Outline

Lyapunov function

Definition ($\mathcal{K}_\infty$ function)

A function $\sigma : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ belongs to a class $\mathcal{K}_\infty$ if it is continuous, zero at zero, strictly increasing, and unbounded ($\sigma(s) \to \infty$ as $s \to \infty$).

Definition (Lyapunov function)

Consider a discrete-time nonlinear autonomous system described above and a positive invariant set $\mathcal{X} \in \mathbb{R}^n$. A function $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is said to be a Lyapunov function for the system in $\mathcal{X}$ if there are functions $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ and a continuous positive definite function α_3 such that, for any $x \in \mathcal{X}$,

$$\alpha_1(\|x\|) \leq V(x) \leq \alpha_2(\|x\|)$$
$$V(f(x)) - V(x) \leq -\alpha_3(\|x\|)$$

Lyapunov asymptotic stability

Theorem ([3])

Consider a discrete-time nonlinear autonomous system described above and a positive invariant set $\mathcal{X} \in \mathbb{R}^n$.

1. *The origin is asymptotically stable if there exists a Lyapunov function defined in $\mathcal{X}$;*
2. *the origin is exponentially stable if, in addition, $\alpha_i(\|x\|) = c_i \|x\|^a$, where $a, c_i \in \mathbb{R}_{>0}$, $i = 1, 2, 3$.*
3. *the origin is globally asymptotically/exponentially stable if, in addition, $\mathcal{X} = \mathbb{R}^n$;*

Nonlinear MPC problem

- Nonlinear predictive control problem, $\mathcal{P}_N(x_0)$:

$$
\begin{aligned}
J_0^*(x_0) = \min_U \quad & J_0(x_0, U) = p(x_N) + \sum_{k=0}^{N-1} q(x_k, u_k) \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k)\ ,\ \forall_{k=0,\dots,N-1}\ ,\ x_0 = x_{t_k} \\
& x_k \in \mathcal{X}\ ,\ u_k \in \mathcal{U}\ ,\ \forall_{k=0,\dots,N-1}\ ,\ x_N \in \mathcal{X}_f
\end{aligned}
$$

- RHC law: $u_{t_k} = f_{RHC}(x_{t_k}) = u_0^*(x_{t_k})$
- Resulting closed-loop (autonomous) system:

$$x_{t_{k+1}} = f(x_{t_k}, f_{RHC}(x_{t_k})) = f_{cl}(x_{t_k})$$

Nonlinear MPC – feasibility

Proposition (Existence of solution [3])

Consider the discrete-time nonlinear system and the nonlinear MPC problem defined above. Assume that

(A1) *The stage and terminal costs, $q(x, u)$ and $p(x)$, are positive definite and, with the model function $f(x, u)$, all are continuous and zero at zero.*

(A2) *The sets $\mathcal{X}$ is closed, $\mathcal{U}$ and $\mathcal{X}_f$ are compact, and all contain the origin.*

Then,

(i) *The cost function $J_0(x, U)$ is continuous for all $x \in \mathcal{X}$ and $U \in \mathcal{U}_N$.*

(ii) *For each $x \in \mathcal{X}_0$, the control constraint set $U \in \mathcal{U}_N$ is compact.*

(iii) *For each $x \in \mathcal{X}_0$, a solution to the MPC problem exists.*

Nonlinear MPC – stability of the origin

Theorem (Stability of the origin [3])

Consider the discrete-time nonlinear system, the nonlinear MPC control law, and the respective closed loop system described above. Assume that

(A1) *The stage and terminal costs, $q(x,u)$ and $p(x)$, are positive definite and, with the model function $f(x,u)$, all are continuous and zero at zero.*

(A2) *The sets $\mathcal{X}$ is closed, $\mathcal{U}$ and $\mathcal{X}_f$ are compact, and all contain the origin.*

(A3) *There are $\mathcal{K}_\infty$ functions $\alpha_q(.)$, $\alpha_p(.)$, and $\alpha_J(.)$ such that*
$q(x,u) \geq \alpha_q(\|x\|)$ *and* $J_N^*(x) \leq \alpha_J(\|x\|)$, *for all* $x \in \mathcal{X}_0$ *and* $u \in \mathcal{U}$,
$p(x) \leq \alpha_p(\|x\|)$ *for all* $x \in \mathcal{X}_f$.

(A4) *For all $x \in \mathcal{X}_f$, there exists $u \in \mathcal{U}$ such that $f(x,u) \in \mathcal{X}_f$ and*

$$p(f(x,u)) - p(x) \leq -q(x,u)$$

Then, the origin of the closed loop system is asymptotically stable in $\mathcal{X}_0$.

Summary 3.2 and tunning of MPC

- To ensure feasibility and stability, we need to carefully design:
  - Cost functional: level and final costs.
  - State and input constraints.
  - Final state constraint set.

- Tunning guidelines:
  - **Weights**: More aggressive response with higher ratio $\|Q\|_F \,/\, \|R\|_F$.
  - **Constraints**: smaller limits on Δu leads to less aggressive responses.
  - **Horizon**: large N least to improved optimality, but more complexity.
  - Possible to have a smaller N for actuation (N_u) than for prediction.
  - Having soft-constraints on the state might improve feasibility.

Further reading

João P. Hespanha.
Linear System Theory
Princeton, 2nd Edition, 2018.

F. Borrelli, A. Bemporad, M. Morari.
Predictive Control for Linear and Hybrid Systems
Cambridge, 2017. [Chapter 7]

J. Rawlings, D. Mayne, M. Diehl.
Model Predictive Control: Theory, Comp., and Design
Nob Hill, 2017. [Chapters 1 and 2]