Lecture 8: Model-predictive control

Mikael Johansson
KTH - Royal Institute of Technology

Outline

- LQR: finite vs infinite horizon, receding-horizon approach
- MPC as receding-horizon LQR with constraints
- Invariant and control invariant sets
- A first design example: YALMIP implementation and experiments

Infininte-horizon LQR

$$\begin{array}{lll} \text{minimize} & \sum_{k=0}^{\infty} x_k^T Q_1 x_k + u_k^T Q_2 u_k & \\ \text{subject to} & x_{k+1} = Ax_k + Bu_k & k = 0, 1, \dots \end{array}$$

Solution is linear state feedback

$$u_t = -Lx_t = -(Q_2 + B^T PB)^{-1} B^T PAx_t \tag{1}$$

$$P = Q_1 + A^T PA - A^T PB(Q_2 + B^T PB)^{-1} BPA \tag{2}$$

Easy to compute and implement, amendable to analysis

- derived using dynamic programming
- cost-to-go is quadratic $V(x) = x^T P_\infty x$
- closed-loop stability under mild conditions

Infinite-horizon LQR with constraints

$$\begin{array}{lll} \text{minimize} & \sum_{k=0}^{\infty} x_k^T Q_1 x_k + u_k^T Q_2 u_k & \\ \text{subject to} & x_{k+1} = Ax_k + Bu_k & \\ & x_k \in X, \qquad u_k \in U & k = 0, 1, \dots \end{array}$$

Difficult to compute, implement and analyze

- cost-to-go not quadratic, hard to find and represent

Finite-horizon LQR

Finite-horizon linear-quadratic control

$$\begin{array}{ll} \text{minimize} & \sum_{k=0}^{N-1}(x_k^T Q_1 x_k + u_k^T Q_2 u_k) + x_N^T Q_f x_N \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t, \qquad\qquad t = 0, \dots, N-1 \\ & \quad x_0 = x(0) \end{array}$$

can be solved as a quadratic program (QP) or using Dynamic programming

$$\begin{aligned} L_k &= (Q_2 + B^T P_{k+1} B)^{-1} B^T P_{k+1} A \\ P_{k-1} &= Q_1 + A^T P_k A - A^T P_k B (Q_2 + B^T P_k B)^{-1} B^T P_k A \end{aligned}$$

- recovers steady-state LQR when $N \to \infty$
- can add (linear) constraints to QP, still easy to solve
- no guarantees beyond horizon N (compare exercises)

Receding-horizon LQR

t ← 0
repeat:
    measure x_t
    solve finite-horizon LQR to obtain $u_t^\star(x_t), u_{t+1}^\star(x_t), \dots, u_{t+N-1}^\star(x_t)$
    apply $u_t = u_t^\star(x_t)$
    t ← t+1

Properties:

- a feedback solution where u_t is a function of x_t
- more computations on-line (delays, reliability?)
- easy to add constraints, planning problem still a QP
- may result in unstable closed-loop system

Potential instability of receding-horizon LQR

Consider a discrete-time linear system with

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

and the receding-horizon problem given by

$$Q_1 = I, \quad Q_2 = 1, \quad Q_f = Q_1$$

The one-step optimal receding-horizon control is

$$u_t = -Lx_t = -\begin{bmatrix} 1 & 0 \end{bmatrix} x_t$$

which yields an unstable closed-loop system

$$x_{t+1} = (A - BL)x_t = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} x_t$$

Ensuring stability of receding-horizon LQR

Re-write infiite-horizon cost-to-go as

$$\begin{aligned} J(x) &= \inf_{\{u_0, u_1, \dots\}} \sum_{k=0}^{\infty} x_k^T Q_1 x_k + u_k^T Q_2 u_k = \\ &= \inf_{\{u_0, \dots, u_{N-1}\}} \sum_{k=0}^{N-1} x_k^T Q_1 x_k + u_k^T Q_2 u_k + \inf_{\{u_N, u_{N+1}, \dots\}} \sum_{k=N}^{\infty} x_k^T Q_1 x_k + u_k^T Q_2 u_k = \\ &= \inf_{\{u_0, \dots\ u_{N-1}\}} \sum_{k=0}^{N-1} x_k^T Q_1 x_k + u_k^T Q_2 u_k + v_N(x_N) = \\ &= \inf_{\{u_0, \dots\ u_{N-1}\}} \sum_{k=0}^{N-1} x_k^T Q_1 x_k + u_k^T Q_2 u_k + x_N^T P x_N \end{aligned}$$

Hence, can generate infinite-horizon optimal control by letting $Q_f = P$.

Interpretation: terminal penalty $x_N^T Q_f x_N$ should reflect future cost-to-go.

Stability of receding-horizon LQR

Theorem. Let (A, B) reachable, $Q_2 \succ 0$ and $Q_1 \succeq 0$ with $(A, Q_1^{1/2})$ observable. Then, if $Q_f = P$ where

$$P = A^T PA + Q_1 - (B^T PA)^T (Q_2 + B^T PB)^{-1} B^T PA$$

the receding-horizon control results in an asymptotically stable closed-loop system for all values of $N \geq 1$. Moreover, the control is a linear feedback $u_t = -Lx_t$ where L satisfies

$$L = (Q_2 + B^T PB)^{-1} B^T PA$$

Stability of receding-horizon LQR

Proof. Letting $P_N = P$ in the Riccati recursion gives $P_{N-k} = P$, for all $k = 1, 2, \ldots, N$. The optimal control is thus identical to the infinite-horizon optimal control $u_t = -Lx_t$.

From last lecture, we know that under the given conditions, the associated closed-loop system is guaranteed to be asymptotically stable.

Note. Next lecture, we will show that also some other choices of Q_f guarantee closed-loop stability.

Attempting the same trick for constrained LQR

Re-write infinite-horizon constrained LQR problem

$$\begin{array}{lll} \text{minimize} & \sum_{t=0}^{\infty} x_t^T Q_1 x_t + u_t^T Q_2 u_t & \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t & t = 0, 1, \ldots \\ & x_t \in X \qquad u_t \in U & t = 0, 1, \ldots \end{array}$$

as

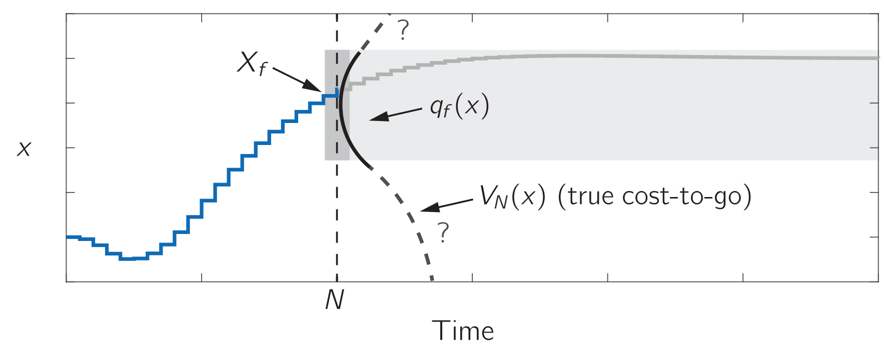
$$\begin{array}{lll} \text{minimize} & \sum_{t=0}^{N-1} x_t^T Q_1 x_t + u_t^T Q_2 u_t + v_N(x_N) & \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t & t = 0, 1, \ldots, N-1 \\ & x_t \in X \qquad u_t \in U & t = 0, 1, \ldots, N-1 \end{array}$$

Challenge. No simple expression for cost-to-go $v_N(x)$.

A way forward

Ensure that x_N is such that $u = -Lx$ will satisfy constraints for all time.

– cost-to-go is quadratic, $v_N(x) = x^T Px$ where P solves ARE



Good approximation when N is large; can be made to work also for small N.

Model predictive control

Basic MPC formulation: receding-horizon LQR with constraints.

Uses finite-horizon constrained optimal control problem

$$\begin{array}{lll} \text{minimize} & \sum_{k=0}^{N-1} q(x_t, u_t) + q_f(x_N) & \\ \text{subject to} & x_{t+1} = Ax_t + Bu_t & t = 0, \dots, N-1 \\ & x_t \in X, & t = 0, \dots, N \\ & u_t \in U & t = 0, \dots, N-1 \\ & x_N \in X_f & \\ & x_0 = \text{current plant state} & \end{array}$$

where $x_t \in X$ models state constraints and $u_t \in U$ control constraints.

A quadratic program (thus, efficiently solved) if

- q and q_f are quadratic and convex, and
- the constraints are described by linear inequalities (and equalities)

The model predictive control algorithm

t ← 0
repeat:
  measure x_t
  solve finite-horizon constrained LQR problem on previous slide
  extract optimal solution $U^\star = \left(u_t^\star(x_t), u_{t+1}^\star(x_t), \dots, u_{t+N-1}^\star(x_t)\right)$
  apply $u_t = u_t^\star(x_t)$
  t ← t+1

Observations:

- A receding-horizon control that accounts for constraints
- Needs to solve a QP in every sample
- Has many tuning parameters (weights, horizons, constraints)

A few words about invariant sets

Definition. The set $\mathcal{I} \subseteq \mathbb{R}^n$ is *(positively) invariant* under $x_{t+1} = f(x_t)$ if

$$x_t \in \mathcal{I} \Rightarrow x_{t+k} \in \mathcal{I} \text{ for all } k \geq 0 \dots$$

Proposition. Assume that a polyhedral constraint set

$$X = \{x \mid Hx \leq h\}$$

is given. The largest invariant set contained in X under the dynamics $x_{t+1} = Ax_t$ is the polyhedron

$$\begin{bmatrix} H \\ HA \\ HA^2 \\ \vdots \end{bmatrix} x \leq \begin{bmatrix} h \\ h \\ h \\ \vdots \end{bmatrix}$$

Once we note that all inequalities in $HA^k \leq h$ are redundant, we can stop.

A few words about invariant sets

Example. Consider a linear system

$$x_{t+1} = Ax_t + Bu_t$$

with state constraints $\|x_t\|_\infty \leq 1$ and $|u_t| \leq u_{\max}$ for all t. What is the largest invariant set $\mathcal{I}$ for which $u = -Lx$ satisfies all constraints?

The constraints require that x satisfies

$$\underbrace{\begin{bmatrix} I \\ -I \\ L \\ -L \end{bmatrix}}_{H} x \leq \underbrace{\begin{bmatrix} \mathbf{1} \\ \mathbf{1} \\ u_{\max} \\ u_{\max} \end{bmatrix}}_{h}$$

The procedure in the previous slide now yields $\mathcal{I}$.

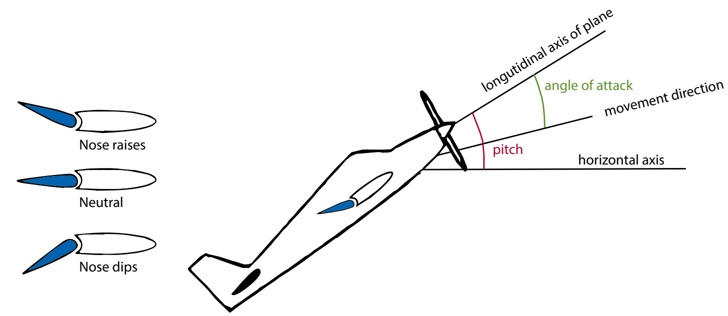
MPC example: Cessna citation aircraft

Linearized model at 5000 m altitude, 128.2m/sec.

$$\dot{x} = \begin{bmatrix} -1.2822 & 0 & 0.98 & 0 \\ 0 & 0 & 1 & 0 \\ -5.4293 & 0 & -1.8366 & 0 \\ -128.2 & 128.2 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} -0.3 \\ 0 \\ -17 \\ 0 \end{bmatrix} u$$

States: x_1: angle of attack, x_2: pitch angle, x_3: pitch rate, x_4: altitude.
Control: u: elevator surface angle.

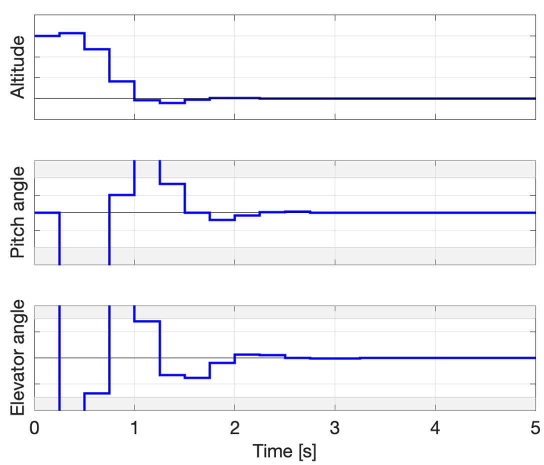
MPC example: terminology and control problem



Objective: manipulate elevators to control pitch and altitude.
Constraints: elevator magnitude $\pm 15°$,
elevator rate-of-change $\pm 30°/s$
pitch angle $\pm 20°$

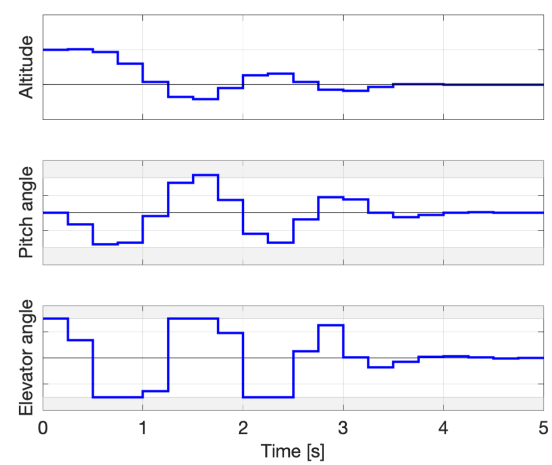
Linear design: altitude change of 10 meters

Use $Q_1 = I$, $Q_2 = 10$. Good altitude response, but violates constraints.



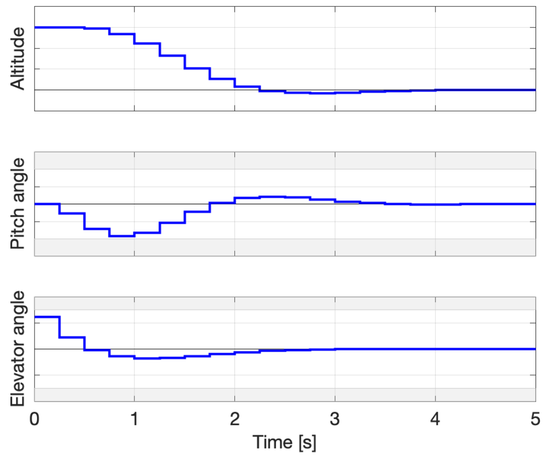
Linear design: altitude change of 10 meters

Poor performance when actuator limits included in simulations.



Linear design: altitude change of 10 meters

Letting $Q_2 = 10^4$ gives slow response, but no constraints are violated.



Unsatisfactory to detune controller to satisfy constraints!

MPC code in YALMIP

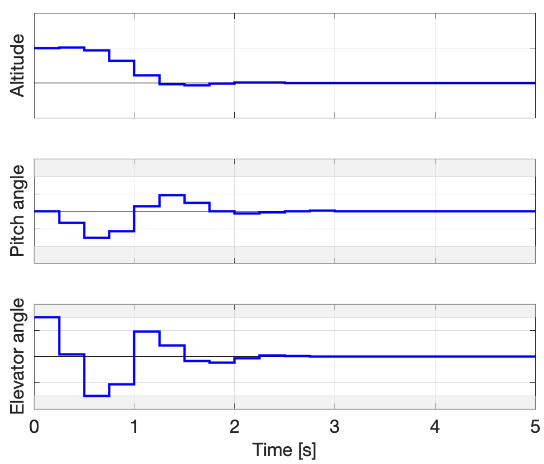
Basic code for defining MPC controller in YALMIP (Matlab)

```
% Introduce decision variables
for t=1:N,
    x{t}=sdpvar(n,1); u{t}=sdpvar(m,1);
end;

% Define objective and contstraints
obj=0; cons=[];
for t=1:N-1,
    obj=obj+x{t}'*Q1*x{t}+u{t}'*Q2*u{t};
    cons=[cons, x{t+1}==A*x{t}+B*u{t}]; % dynamics
    cons=[cons, ymin<= C*x{t} <= ymax]; % state constraints
    cons=[cons, umin<= u{t} <=umax];    % control constraints
end;
obj=obj+x{N}'*Qf*x{N};                  % terminal cost
cons=[cons, MN*x{N} <= mN];             % terminal constraint

% Construct controller object
controller=optimizer(cons, obj, [], x{1}, u{1});
```

Many extensions later...

MPC design

Model predictive control recrectly for actuator constraints.
($Q_1 = I$, $Q_2 = 10$, $N = 10$, $X_N = \mathbf{R}^n$, $Q_f = 0$.



Adding rate-of-change constraint

Rate-of-change constraints

$$|u_t - u_{t-1}| \leq \delta_{\max} \Leftrightarrow u_t - u_{t-1} \leq \delta_{\max} \wedge -u_t + u_{t-1} \leq \delta_{\max}$$

readily included in QP (but does not fit standard form for MPC).

Alternatively, re-write prediction model as

$$\begin{bmatrix} x_{t+1} \\ u_t \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \begin{bmatrix} x_t \\ u_{t-1} \end{bmatrix} + \begin{bmatrix} B \\ I \end{bmatrix} (u_t - u_{t-1})$$

i.e. consider $\Delta u_t = u_t - u_{t-1}$ as input to augmented system

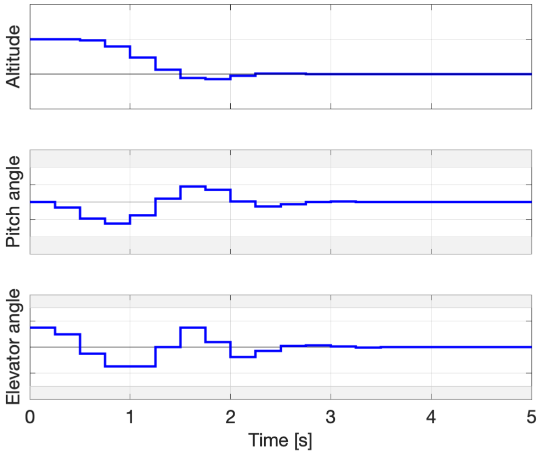
$$\bar{x}_{t+1} = \bar{A}\bar{x}_t + \bar{B}\Delta u_t$$

Then

- rate-of-change constraints on u_t are magnitude constraints on Δu_t
- magnitude constraints on u_t are state constraints in augmented system

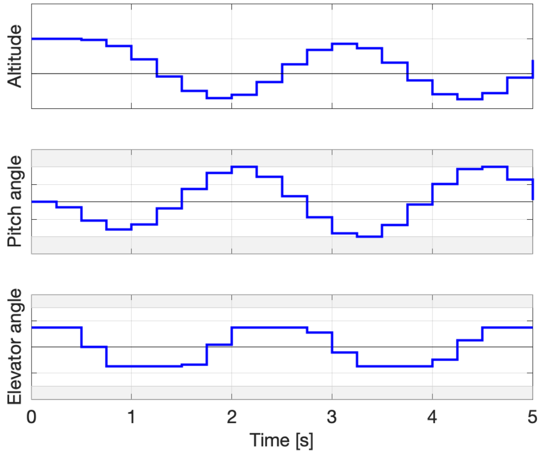
MPC design with rate-constraints on actuators

Works perfectly!



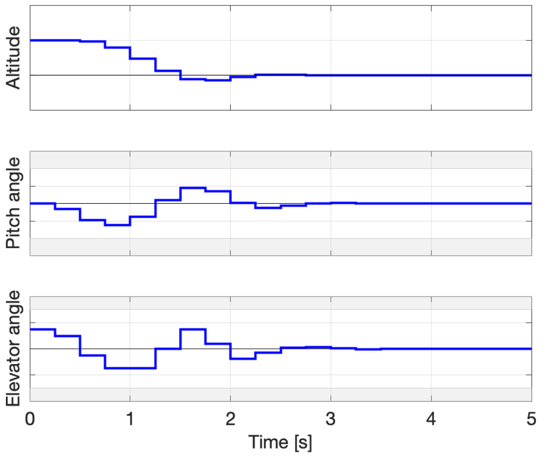
MPC design: some things can still go wrong

Setting $N = 4$ gives unstable behaviour.



MPC design with rate-constraints on actuators

Instability fixed by adding terminal cost, terminal constraint.



However, in a 40 meter descent, the MPC problem becomes infeasible!

MPC: some things can still go wrong

A few things can still go wrong

- performance may be suboptimal, compared to infinite-horizon LQR
  - remidied by long horizons (but computational cost increases!)
- closed-loop may become unstable
  - can be fixed with appropriate terminal sets, terminal constraints
- the optimization problem may become infeasible
  - improved by proper terminal set, horizon length
  - must still understand which initial states are beyond our control
  - may need to accept (occasional) constraint violations

We will address these challenges in the next lectures!

Summary

- LQR: batch vs. DP, finite vs infinite horizon, receding-horizon approach
- MPC as receding-horizon LQR with constraints
- A first motivation for terminal set and terminal constraints
- Experience from a first design example