MECH 6v29.002 – Model Predictive Control

L19 – Nonlinear MPC (continued)

Outline

- Nonlinear MPC
- Finding the Sublevel Set for Terminal Constraint Set
- Bounding the Linearization Error
- Inverted Pendulum Numerical Example

Nonlinear MPC (cont.)

- For now, assume that there exists $a > 0$ such that linearization error is small
- Summary:

$$J_0^*(x_0) = \min_{U_0} \sum_{k=0}^{N-1} q(x_k, u_k) + p(x_N)$$

$$p(x_N) = \frac{1}{2} x_N^T P x_N$$

$$A_K^T P A_K - P + \mu Q_K = 0$$

$$s.t.$$

$$x_{k+1} = f(x_k, u_k),\ k \in \{0,1,\ldots,N-1\}$$

$$h(x_k, u_k) \le 0, \quad k \in \{0,1,\ldots,N-1\}$$

$$x_N = \mathcal{X}_f$$

$$\mathcal{X}_f = \text{lev}_a\, p(x_N) = \left\{ x_N \in \mathbb{R}^n \mid p(x_N) \le a \right\}$$

$$x_0 = x(0)$$

$$= \left\{ x_N \in \mathbb{R}^n \mid \frac{1}{2} x_N^T P x_N \le a \right\}$$

Ellipsoid centered at the origin

- Assuming initial condition is feasible, the origin is exponentially stable under this MPC controller formulation

Finding the Sublevel Set

- How do we find $a > 0$ to define the terminal constraint such that the following is true?

$$p\left(f\left(x, Kx\right)\right) - p\left(x\right) + \frac{1}{2} x^T Q_K x \le 0 \quad \forall x \in \mathcal{X}_f = \text{lev}_a\, p\left(x_N\right) = \left\{x_N \in \mathbb{R}^n \mid p\left(x_N\right) \le a\right\}$$

- Could take a direct approach by solving the following nonlinear program.
- Choose $a > 0$ such that

$$\max_{x \in \text{lev}_a\, p(x_N)} p\left(f\left(x, Kx\right)\right) - p\left(x\right) + \frac{1}{2} x^T Q_K x \le 0$$

- Would likely want to maximize terminal set by maximizing a

$$\max_{a>0} \left(\max_{x \in \text{lev}_a\, p(x_N)} p\left(f\left(x, Kx\right)\right) - p\left(x\right) + \frac{1}{2} x^T Q_K x \le 0 \right)$$

$$\begin{aligned} s.t. \quad & \text{lev}_a\, p\left(x_N\right) \subseteq \mathcal{X} \\ & K \;\text{lev}_a\, p\left(x_N\right) \subseteq \mathcal{U} \end{aligned}$$

Could be hard to solve directly.

Finding the Sublevel Set (cont.)

- Alternatively, we could try to analytically find $a > 0$ such that

$$p\left(f\left(x, Kx\right)\right) - p\left(x\right) + \frac{1}{2} x^T Q_K x \le 0 \quad \forall x \in \mathcal{X}_f = \operatorname{lev}_a p\left(x_N\right) = \left\{x_N \in \mathbb{R}^n \mid p\left(x_N\right) \le a\right\}$$

- We have already seen that this is achieved if $\left(p\left(x\right) = \frac{1}{2} x^T P x\right)$

$$\Delta p \triangleq p\left(f\left(x, Kx\right)\right) - p\left(A_K x\right) \le \frac{\mu - 1}{2} x^T Q_K x \quad \forall x \in \operatorname{lev}_a p\left(x\right)$$

- To find when this is true, define the error between the nonlinear and linear models $e\left(x\right) = f\left(x, Kx\right) - A_K x$
- Algebraic manipulations

$$\begin{aligned} p\left(f\left(x, Kx\right)\right) = p\left(e\left(x\right) + A_K x\right) &= \frac{1}{2}\left(e\left(x\right) + A_K x\right)^T P\left(e\left(x\right) + A_K x\right) \\ &= \frac{1}{2} e\left(x\right)^T P e\left(x\right) + \frac{1}{2} x^T A_K^T P A_K x + x^T A_K^T P e\left(x\right) \end{aligned}$$

$$p\left(A_K x\right) = \frac{1}{2} x^T A_K^T P A_K x$$

$$\Delta p \triangleq p\left(f\left(x, Kx\right)\right) - p\left(A_K x\right) = \frac{1}{2} e\left(x\right)^T P e\left(x\right) + x^T A_K^T P e\left(x\right)$$

Finding the Sublevel Set (cont.)

- We need to find $a > 0$ such that

$$\Delta p = \frac{1}{2} e(x)^T P e(x) + x^T A_K^T P e(x) \leq \frac{\mu - 1}{2} x^T Q_K x \qquad \forall x \in \operatorname{lev}_a p(x)$$

- Goal is to write every term as a function of $\|x\|_2$
- Note that

$$e(x)^T P e(x) \leq \lambda_{\max}(P) \|e(x)\|_2^2 \qquad \lambda_{\min}(Q_K) \|x\|_2^2 \leq x^T Q_K x$$

- Then

$$\Delta p = \frac{1}{2} e(x)^T P e(x) + x^T A_K^T P e(x)$$

$$\Delta p \leq \frac{1}{2} \lambda_{\max}(P) \|e(x)\|_2^2 + x^T A_K^T P e(x)$$

- Note that $x^T A_K^T P e(x) \leq \|P A_K x\|_2 \|e(x)\|_2 \leq \|P A_K\|_2 \|x\|_2 \|e(x)\|_2$
- Therefore

$$\Delta p \leq \frac{1}{2} \lambda_{\max}(P) \|e(x)\|_2^2 + \|P A_K\|_2 \|x\|_2 \|e(x)\|_2$$

Finding the Sublevel Set (cont.)

- We have $$\Delta p \le \frac{1}{2}\lambda_{\max}(P)\|e(x)\|_2^2 + \|PA_K\|_2\|x\|_2\|e(x)\|_2$$
- Let's assume the error is small (we will prove this later)

$$\|e(x)\|_2 \le \frac{1}{2}c_\delta\|x\|_2^2 \qquad \forall x \in \delta\mathcal{B} = \left\{x \in \mathbb{R}^n \mid x^T x = \|x\|_2^2 \le \delta^2\right\}$$

- Then $\Delta p \le \frac{1}{8}c_\delta^2\lambda_{\max}(P)\|x\|_2^4 + \frac{1}{2}c_\delta\|PA_K\|_2\|x\|_2^3$
- Since we want $\Delta p \le \frac{\mu-1}{2}x^T Q_K x$
- And we have

$$\Delta p \le \frac{1}{8}c_\delta^2\lambda_{\max}(P)\|x\|_2^4 + \frac{1}{2}c_\delta\|PA_K\|_2\|x\|_2^3 \qquad \lambda_{\min}(Q_K)\|x\|_2^2 \le x^T Q_K x$$

- We need

$$\frac{1}{8}c_\delta^2\lambda_{\max}(P)\|x\|_2^4 + \frac{1}{2}c_\delta\|PA_K\|_2\|x\|_2^3 \le \frac{\mu-1}{2}\lambda_{\min}(Q_K)\|x\|_2^2$$

Finding the Sublevel Set (cont.)

- We need

$$\frac{1}{8}c_\delta^2\lambda_{\max}(P)\|x\|_2^4+\frac{1}{2}c_\delta\|PA_K\|_2\|x\|_2^3\le\frac{\mu-1}{2}\lambda_{\min}(Q_K)\|x\|_2^2$$

- Since $\|x\|_2>0\ \ \forall x\neq 0$

$$\frac{1}{8}c_\delta^2\lambda_{\max}(P)\|x\|_2^2+\frac{1}{2}c_\delta\|PA_K\|_2\|x\|_2\le\frac{\mu-1}{2}\lambda_{\min}(Q_K)$$

- We currently have the condition that $x\in\delta\mathcal{B}=\left\{x\in\mathbb{R}^n \mid x^Tx=\|x\|_2^2\le\delta^2\right\}$
- We want to impose the condition

$$x\in\mathcal{X}_f=\operatorname{lev}_a p(x)=\left\{x\in\mathbb{R}^n \mid p(x)\le a\right\}=\left\{x\in\mathbb{R}^n \mid \frac{1}{2}x^TPx\le a\right\}$$

- Since $\frac{1}{2}\lambda_{\min}(P)\|x\|_2^2\le\frac{1}{2}x^TPx$

$$x\in\mathcal{X}_f\Rightarrow\frac{1}{2}\lambda_{\min}(P)\|x\|_2^2\le a\ \ \Rightarrow\|x\|_2\le\sqrt{\frac{2a}{\lambda_{\min}(P)}}$$

Finding the Sublevel Set (cont.)

- Now we can finally choose $a > 0$ such that x being in the terminal constraint set results in

$$\|x\|_2 \leq \sqrt{\frac{2a}{\lambda_{\min}(P)}} \leq \delta$$

Since $x \in \delta\mathcal{B} = \left\{x \in \mathbb{R}^n \mid x^T x = \|x\|_2^2 \leq \delta^2\right\}$ is required for the linearization error to be small

- and $\frac{1}{8}c_\delta^2 \lambda_{\max}(P)\|x\|_2^2 + \frac{1}{2}c_\delta \|PA_K\|_2 \|x\|_2 \leq \frac{\mu - 1}{2}\lambda_{\min}(Q_K)$

- which is satisfied if the following condition is satisfied

$$\frac{1}{8}c_\delta^2 \lambda_{\max}(P)\frac{2a}{\lambda_{\min}(P)} + \frac{1}{2}c_\delta \|PA_K\|_2 \sqrt{\frac{2a}{\lambda_{\min}(P)}} \leq \frac{\mu - 1}{2}\lambda_{\min}(Q_K)$$

Bounding the Linearization Error

- Now we can derive/verify our assumption that the error was small

$$\left\|e(x)\right\|_2 \le \frac{1}{2} c_\delta \left\|x\right\|_2^2 \qquad \forall x \in \delta\mathcal{B} = \left\{x \in \mathbb{R}^n \mid x^T x = \left\|x\right\|_2^2 \le \delta^2\right\}$$

- Linearization error was defined as $e(x) = f(x, Kx) - A_K x$
- By definition, no linearization error at the origin

$$e(0) = f(0,0) - A_K 0 = 0$$

- Derivative of linearization error is

$$\frac{d}{dx} e(x) = \frac{\partial}{\partial x} f(x, Kx) + \frac{\partial}{\partial u} f(x, Kx) \frac{du}{dx} - A_K \qquad u = Kx$$

$$\frac{d}{dx} e(x) = \frac{\partial}{\partial x} f(x, Kx) + \frac{\partial}{\partial u} f(x, Kx) K - A_K \qquad \frac{du}{dx} = K$$

- Derivative of linearization error is zero at the origin

$$\frac{d}{dx} e(0) = A + BK - A_K = 0 \qquad A = \left.\frac{\partial f}{\partial x}\right|_{x,u=0} \qquad B = \left.\frac{\partial f}{\partial u}\right|_{x,u=0}$$

Bounding the Linearization Error (cont.)

- Will use result from Mean Value Theorem to bound linearization error (Proposition A.11 of [1])
- Mean Value Theorem for Vector Functions

If $f:\mathbb{R}^n \to \mathbb{R}^m$ has continuous partial derivatives at each point of $x \in \mathbb{R}^n$,

then for any $x, y \in \mathbb{R}^n$,

$$f(y) = f(x) + \frac{\partial}{\partial x} f(x)(y-x) + \int_0^1 (1-s)(y-x)^T \frac{\partial^2}{\partial x^2} f\big(x + s(y-x)\big)(y-x)\,ds$$

- Think about $f(y) = e(x) \quad f(x) = e(0)$
- Since $e(0) = 0 \qquad \frac{d}{dx} e(0) = 0$

$$\Downarrow$$

$$e(x) = 0 + 0 + \int_0^1 (1-s)\, x^T \frac{\partial^2}{\partial x^2} e(sx)\, x\,ds$$

Watch out for the change in variable x

[1] J. Rawlings, D. Mayne, M. Diehl. "Model Predictive Control: Theory, Computation, and Design," Nob Hill Publishing, 2nd Edition, 2019.

Bounding the Linearization Error (cont.)

- Now we can use the fact that *f* is twice continuously differentiable to bound the linearization error
- Since *f* is twice continuously differentiable

for any $\delta>0$, there exists $c_\delta > 0$ such that

$$\left\| \frac{\partial^2}{\partial x^2} e(x) \right\|_2 \le c_\delta, \quad \forall x \in \delta\mathcal{B} = \left\{ x \in \mathbb{R}^n \mid x^T x = \|x\|_2^2 \le \delta^2 \right\}$$

- Therefore,

$$e(x) = \int_0^1 (1-s) x^T \frac{\partial^2}{\partial x^2} e(sx) x ds \quad \Rightarrow \quad \|e(x)\|_2 \le \left\| \int_0^1 (1-s) x^T c_\delta x ds \right\|_2$$

Note that s is between 0 and 1

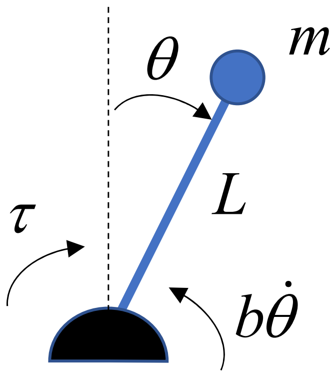
$$\|e(x)\|_2 \le c_\delta \int_0^1 (1-s) ds \left\| x^T x \right\|_2$$

This is exactly what we used when we assumed the error was small

$$\|e(x)\|_2 \le \frac{c_\delta}{2} \|x\|_2^2 \quad \forall x \in \delta\mathcal{B}$$

Inverted Pendulum Example

- Consider the inverted pendulum



- Dynamic model: $I\ddot{\theta} = \sum M$

$$mL^2\ddot{\theta} = \tau - b\dot{\theta} + mgL\sin\theta$$

$$\ddot{\theta} = \frac{1}{mL^2}\tau - \frac{b}{mL^2}\dot{\theta} + \frac{g}{L}\sin\theta$$

- State-space model:

$$x_1 = \theta \qquad \dot{x}_1 = x_2$$

$$x_2 = \dot{\theta} \qquad \dot{x}_2 = \frac{g}{L}\sin x_1 - \frac{b}{mL^2}x_2 + \frac{1}{mL^2}u$$

$$u = \tau$$

- Forward-Euler discretization $\dot{x} \approx \dfrac{x^+ - x}{\Delta t}$

$$x_1^+ = x_1 + \Delta t x_2$$

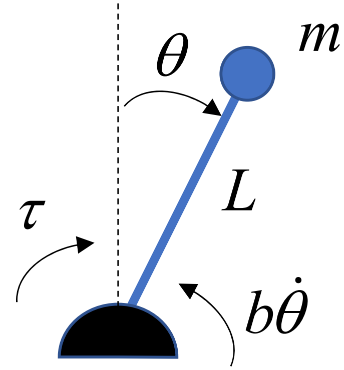
$$x_2^+ = \Delta t\frac{g}{L}\sin x_1 + \left(1 - \frac{\Delta t b}{mL^2}\right)x_2 + \frac{\Delta t}{mL^2}u$$

$$\Rightarrow \quad x_{k+1} = f\left(x_k, u_k\right)$$

Inverted Pendulum Example (cont.)

- Linearization about the origin:

$$\frac{\partial f}{\partial x} = \begin{bmatrix} 1 & \Delta t \\ \Delta t \dfrac{g}{L} \cos x_1 & 1 - \dfrac{\Delta t b}{mL^2} \end{bmatrix} \qquad \frac{\partial f}{\partial u} = \begin{bmatrix} 0 \\ \dfrac{\Delta t}{mL^2} \end{bmatrix}$$



$$x_1^+ = x_1 + \Delta t x_2$$

$$x_2^+ = \Delta t \frac{g}{L} \sin x_1 + \left(1 - \frac{\Delta t b}{mL^2}\right) x_2 + \frac{\Delta t}{mL^2} u$$

$$A = \left.\frac{\partial f}{\partial x}\right|_{x,u=0} = \begin{bmatrix} 1 & \Delta t \\ \Delta t \dfrac{g}{L} & 1 - \dfrac{\Delta t b}{mL^2} \end{bmatrix}$$

$$B = \left.\frac{\partial f}{\partial u}\right|_{x,u=0} = \begin{bmatrix} 0 \\ \dfrac{\Delta t}{mL^2} \end{bmatrix}$$

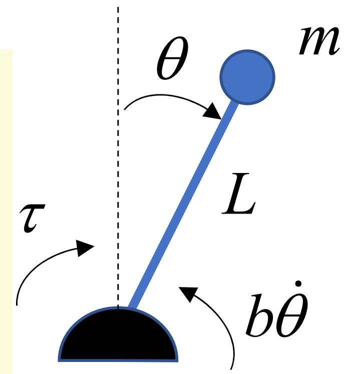
Inverted Pendulum Example (cont.)

- Parameters:

```
%% System Definition
% System dimensions
nx = 2; % Number of states
nu = 1; % Number of inputs (controllable)

% Model parameters
g = 9.81;  % Gravity [m/s^2]
L = 1;     % Length of pendulum
m = 2;     % Mass of pendulum
b = 3;     % Friction coeefficient
dt = 0.1;  % Discrete time step size
thetaMaxSS = pi/2; % Maximum angle that torque can hold at steady-state [rad]
tauMax = abs(m*g*L*sin(thetaMaxSS));

% Nonlinear discrete time model
f = @(x,u) [x(1) + dt*x(2); (1-dt*b/(m*L^2))*x(2) + dt*g/L*sin(x(1)) + dt/(m*L^2)*u(1)];
% Linear discrete time model
A = [1 dt; dt*g/L 1-dt*b/(m*L^2)];
B = [0; dt/(m*L^2)];
f_lin = @(x,u) [A*x + B*u];
```



- Constraints

$$-5\ rad \le x_1 \le 5\ rad$$

$$-10\ rad \le x_2 \le 10\ rad/s$$

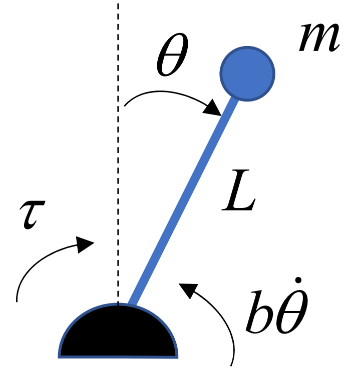
$$-\tau_{\max} \le u \le \tau_{\max}$$

Inverted Pendulum Example (cont.)

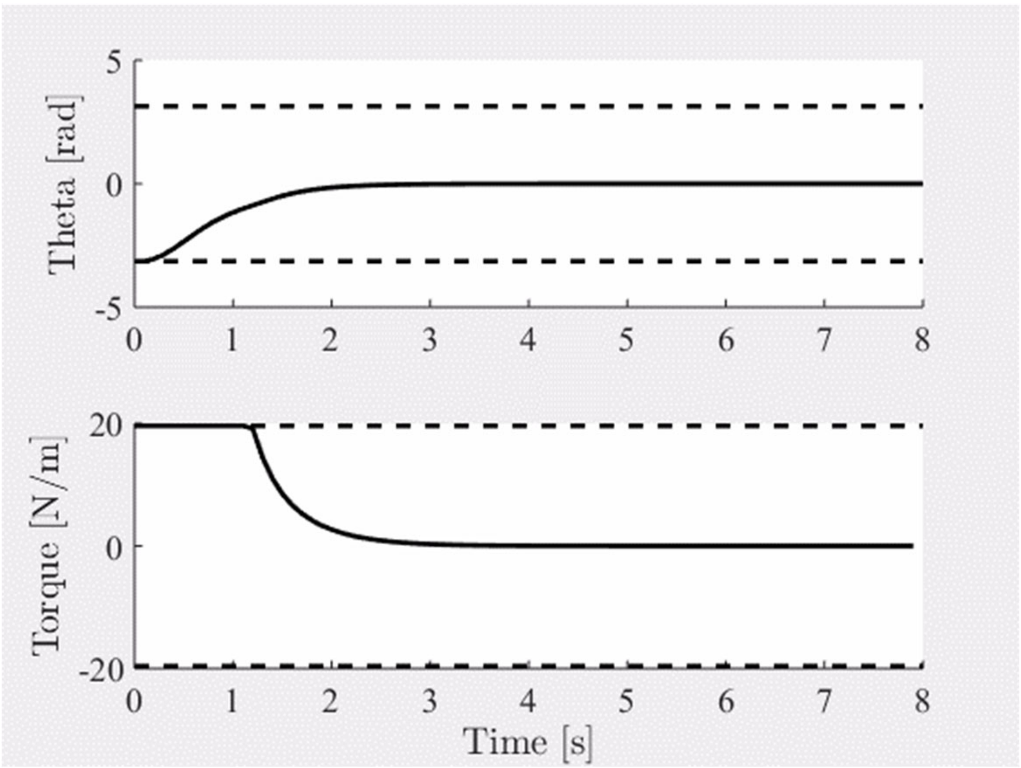
- Control design:
  - Stage cost: $q(x_k, u_k) = \frac{1}{2}\left(x_k^T Q x_k + u_k^T R u_k\right)$

    $$Q = I_2,\ R = 1$$

  - LQR Design for candidate static feedback control law $u_k = K x_k$




```
%% LQR design
Q = eye(nx);
R = eye(nu);
[K,~,~] = dlqr(A,B,Q,R);
K = -K; % For positive u = Kx
% Closed-loop linear dynamics
AK = A + B * K;
% Closed-loop stage cost
QK = A + K'*R*K;
```



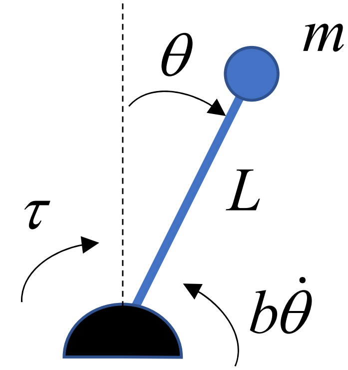
  - LQR applied in closed-loop to the nonlinear system starting at different initial angles
  - Surprisingly stabilizes from most initial conditions but not all

Inverted Pendulum Example (cont.)

- Control design:
  - Lyapunov equation for terminal cost

$$A_K^T P A_K - P + \mu Q_K = 0 \quad \Rightarrow \quad p(x_N) = \frac{1}{2} x_N^T P x_N$$

```
%% Lyapunov Equation
mu = 10; % Must be greater than 1
P = dlyap(AK',mu*Q); % Remember A' since Matlab uses A*X*A' - X + Q = 0
```



  - Finding a sublevel set for terminal constraint

$$x_N \in \mathcal{X}_f = \text{lev}_a\, p(x_N) = \left\{ x_N \in \mathbb{R}^n \mid \frac{1}{2} x^T P x \le a \right\}$$

    - Analyze linearization error

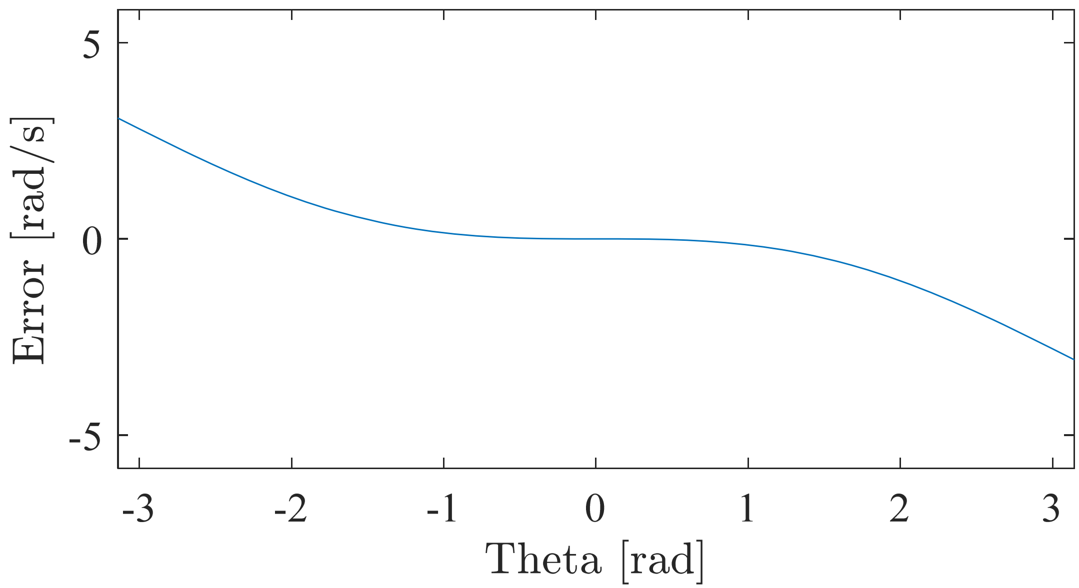
$$e(x) = f(x, Kx) - A_K x$$

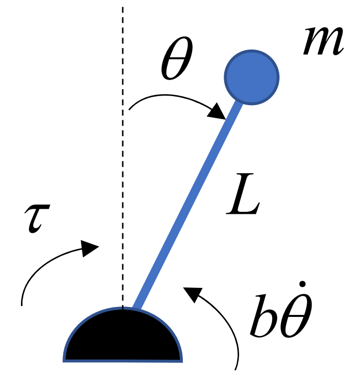
    - By comparing nonlinear and linear model, we should get

$$e(x) = \begin{bmatrix} 0 \\ \Delta t \frac{g}{L} \sin x_1 - \Delta t \frac{g}{L} x_1 \end{bmatrix}$$

Inverted Pendulum Example (cont.)

- Control design:
  - Plot the linearization error $e(x)=\begin{bmatrix} 0 \\ \Delta t \dfrac{g}{L}\sin x_1 - \Delta t \dfrac{g}{L} x_1 \end{bmatrix}$

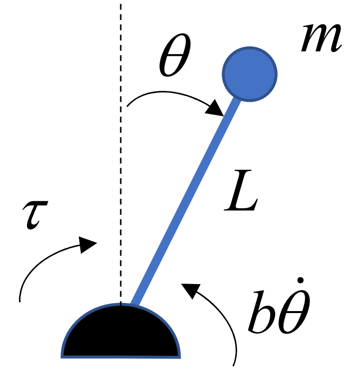




  - Confirm that $e(0)=0$ $\quad \dfrac{d}{dx}e(0)=0$

Inverted Pendulum Example (cont.)

- Control design:
  - Plot the second derivative of the linearization error



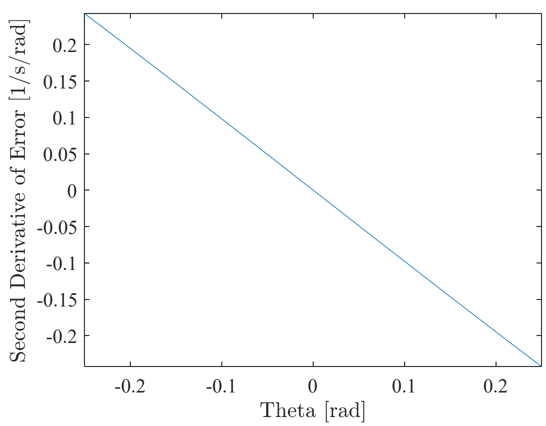
$$e(x)=\begin{bmatrix} 0 \\ \Delta t\frac{g}{L}\sin x_1-\Delta t\frac{g}{L}x_1 \end{bmatrix} \qquad \frac{d^2}{dx^2}e(x)=\begin{bmatrix} 0 \\ -\Delta t\frac{g}{L}\sin x_1 \end{bmatrix}$$

  - Choose

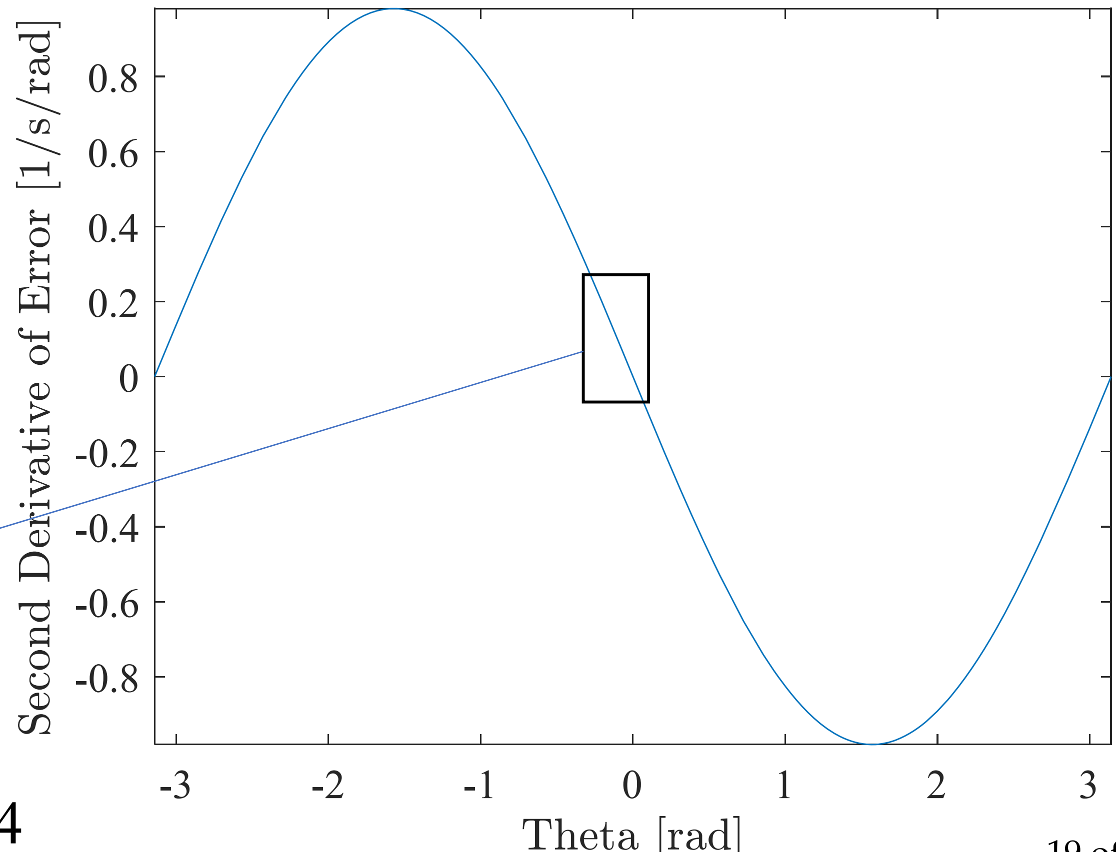
$\delta>0$ and $c_\delta>0$ such that

$$\left\|\frac{\partial^2}{\partial x^2}e(x)\right\|_2 \le c_\delta, \quad \forall x\in\delta\mathcal{B}$$

```
delta = 0.25;
c_delta = abs(dt*g/L*sin(delta));
```



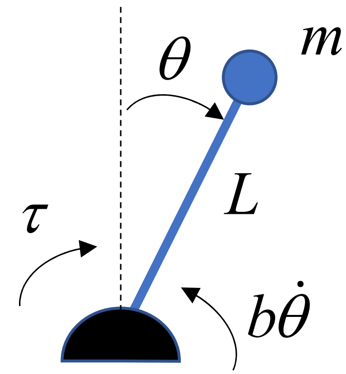
$c_\delta=0.24$



Inverted Pendulum Example (cont.)

- Control design:
  - Find $a > 0$ that satisfies

$$\frac{1}{8}c_\delta^2 \lambda_{\max}(P)\frac{2a}{\lambda_{\min}(P)} + \frac{1}{2}c_\delta \left\|PA_K\right\|_2 \sqrt{\frac{2a}{\lambda_{\min}(P)}} \le \frac{\mu-1}{2}\lambda_{\min}(Q_K)$$



and $\sqrt{\dfrac{2a}{\lambda_{\min}(P)}} \le \delta$

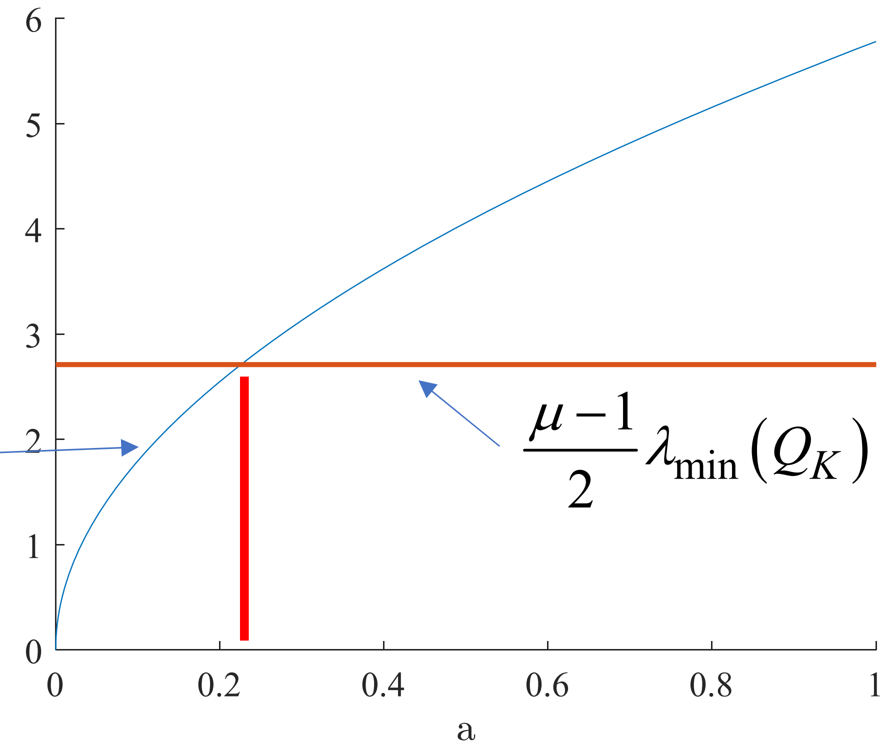
```
lambdaMinP = min(eig(P));
lambdaMaxP = max(eig(P));
lambdaMinQK = min(eig(QK));
normPAK = norm(P*AK);

c1 = 1/4*c_delta^2*lambdaMaxP/lambdaMinP;
c2 = sqrt(2)/2*c_delta*normPAK/sqrt(lambdaMinP);

f_a = @(a) c1*a(1) + c2*sqrt(a(1));
```

```
f_aRoot = @(a) f_a(a) - (mu-1)/2*lambdaMinQK;

a = fzero(f_aRoot,1e0);
r = sqrt(2*a/lambdaMinP);
```



$$0.1838 = r = \sqrt{\frac{2a}{\lambda_{\min}(P)}} \le \delta = 0.25$$

Inverted Pendulum Example (cont.)

- Control design:

$$J_0^*(x_0) = \min_{U_0} \sum_{k=0}^{N-1} q(x_k, u_k) + p(x_N) \qquad p(x_N) = \frac{1}{2} x_N^T P x_N$$

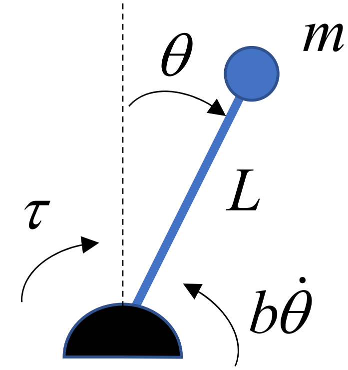
$$s.t.$$

$$x_{k+1} = f(x_k, u_k), \; k \in \{0,1,\ldots,N-1\}$$

$$h(x_k, u_k) \le 0, \quad k \in \{0,1,\ldots,N-1\}$$

$$x_N = \mathcal{X}_f \qquad \mathcal{X}_f = \mathrm{lev}_a\, p(x_N) == \left\{ x_N \in \mathbb{R}^n \,|\, \frac{1}{2} x_N^T P x_N \le a \right\}$$

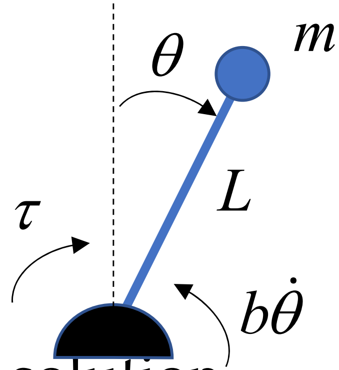
$$x_0 = x(0)$$




```
% Prediction horizon
N = 80;
% Optimization variables
u_ = sdpvar(repmat(nu,1,N),repmat(1,1,N));
x_ = sdpvar(repmat(nx,1,N+1),repmat(1,1,N+1));
constraints = [];
objective = 0;
for k = 1:N
    objective = objective + 1/2*x_{k}'*Q*x_{k} + 1/2*u_{k}'*R*u_{k};
    constraints = [constraints, x_{k+1} == f(x_{k},u_{k})];
    constraints = [constraints, -tauMax <= u_{k} <= tauMax];
    constraints = [constraints, -5 <= x_{k}(1) <= 5];
    constraints = [constraints, -10 <= x_{k}(2) <= 10];
end
constraints = [constraints, x_{N+1}'*P*x_{N+1} <= 2*a];
objective = objective + 1/2*x_{N+1}'*P*x_{N+1};
```

Inverted Pendulum Example (cont.)

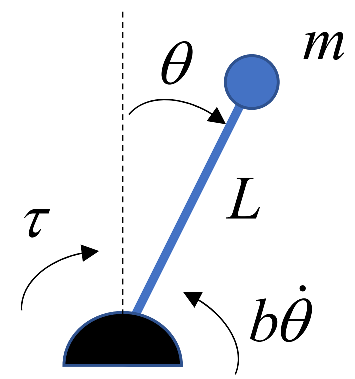
- Nonlinear MPC implementation:
  - Used IPOPT (could use fmincon)
  - Nonlinear programs are much harder to solve
  - If you type opts.ipopt, you will see 100+ settings you can configure (most you should not change)
  - No guarantee that you will find the globally optimal solution
    - No guarantee that you will find a feasible local solution
  - Warm-starting is extremely valuable
    - Give a good initial guess of the optimal solution to start the solver close to a local minimum
    - Change your initial guess, could get completely different solution (different local minimum)



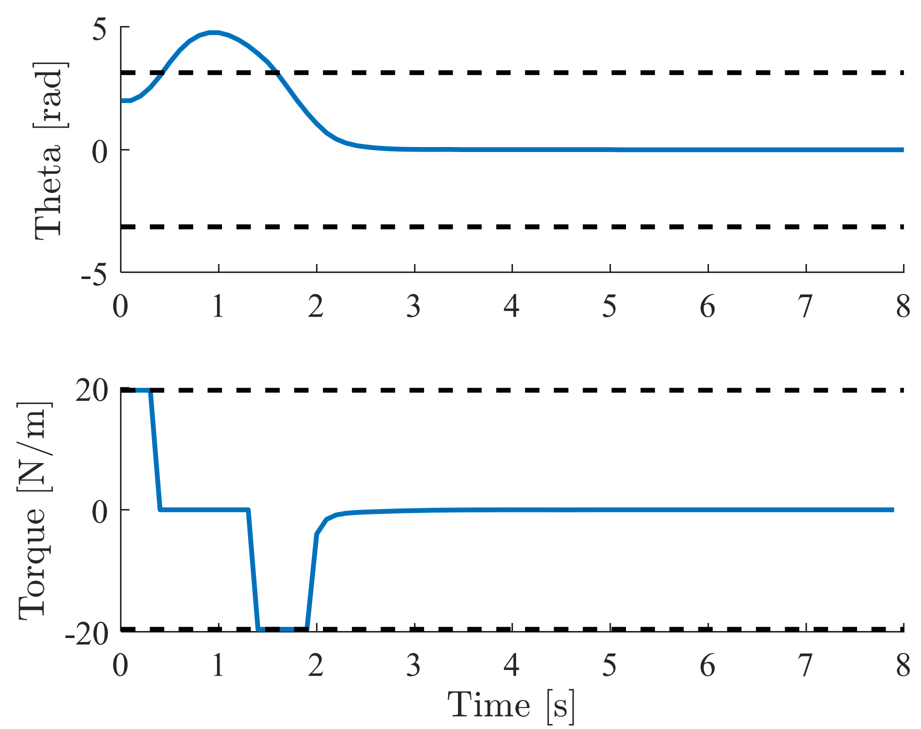

```
opts = sdpsettings('solver','ipopt','ipopt.max_iter',10000,'usex0',1,'debug',0,'verbose',2);
controller = optimizer(constraints,objective,opts,{x_{1}},[u_,x_]);
```

Inverted Pendulum Example (cont.)

- Starting from an initial condition in the range where LQR did not work $x_0 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$
- Warm-start
  - Designed a simple control strategy to use as an initial guess of a solution

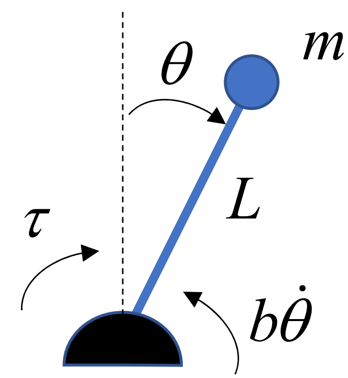



```
sim_length = N;
x0 = [2;0];
x = x0;
u = [];
tCalcs = [];
for k_step = 1:sim_length
    if k_step <= 4
        u_k = tauMax;
    elseif (5 <= k_step) && (k_step <= 14)
        u_k = 0;
    elseif (15 <= k_step) && (k_step <= 20)
        u_k = -tauMax;
    else
        u_k = K*x(:,end);
    end
    u_k = max(u_k,-tauMax);
    u_k = min(u_k,tauMax);
    x = [x f(x(:,end),u_k)];
    u = [u u_k];
end
```

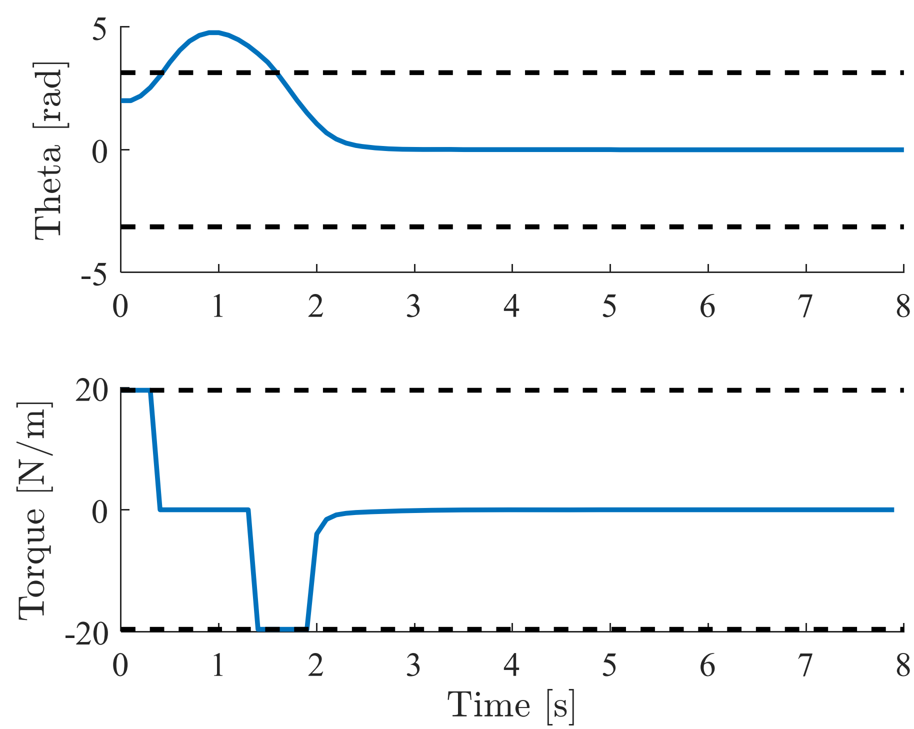


Inverted Pendulum Example (cont.)

- Starting from an initial condition in the range where LQR did not work $x_0 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$
- Warm-start
  - Assign this solution to the yalmip variables
  - And check that this solution satisfies the constraints of the MPC formulation

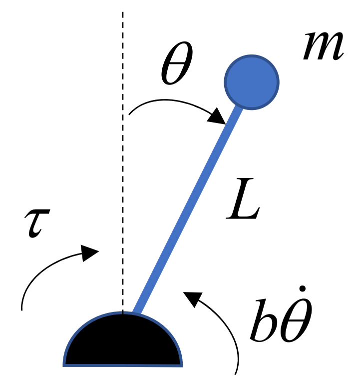



```
for k = 1:N
    assign(x_{k},x(:,k));
    assign(u_{k},u(k));
end
assign(x_{k+1},x(:,k+1));
check(constraints)
```

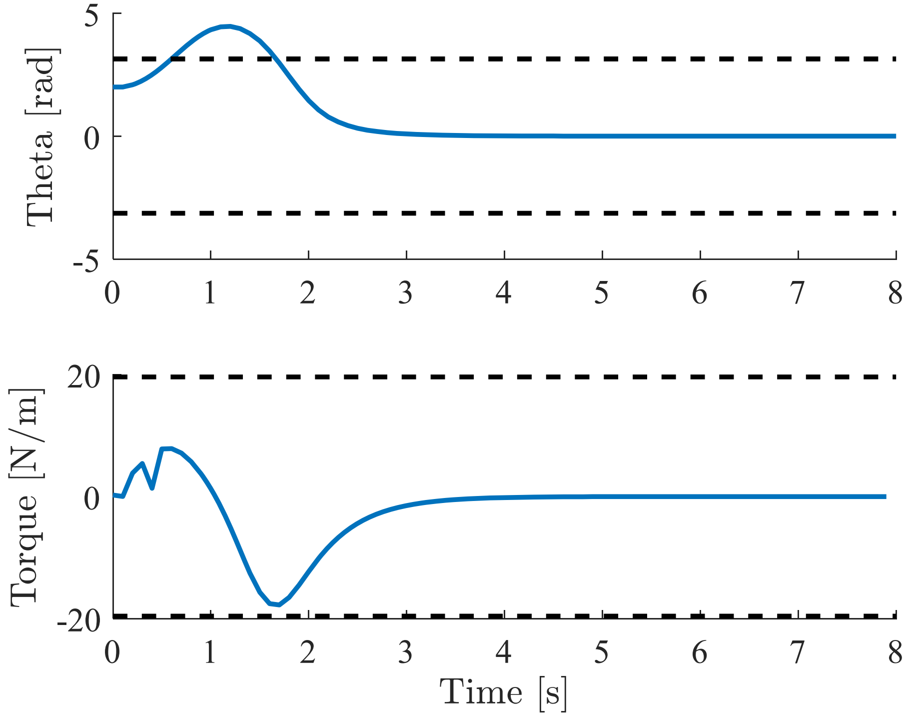


Inverted Pendulum Example (cont.)

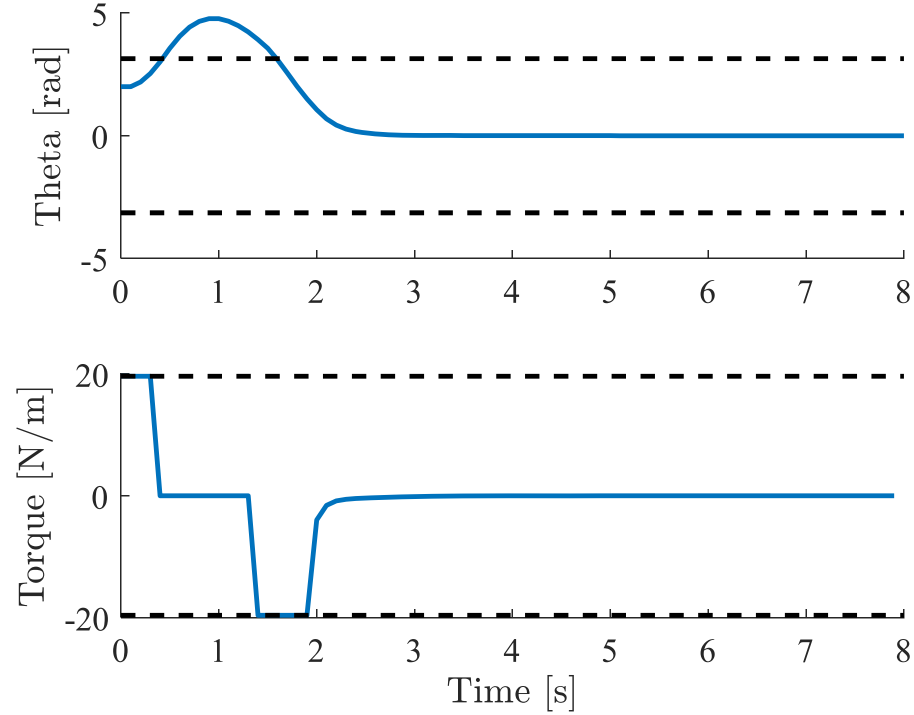
- Starting from an initial condition in the range where LQR did not work $x_0 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$
- MPC Solution



Nonlinear MPC solution



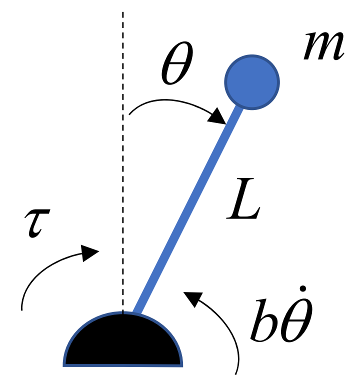
Warm start solution



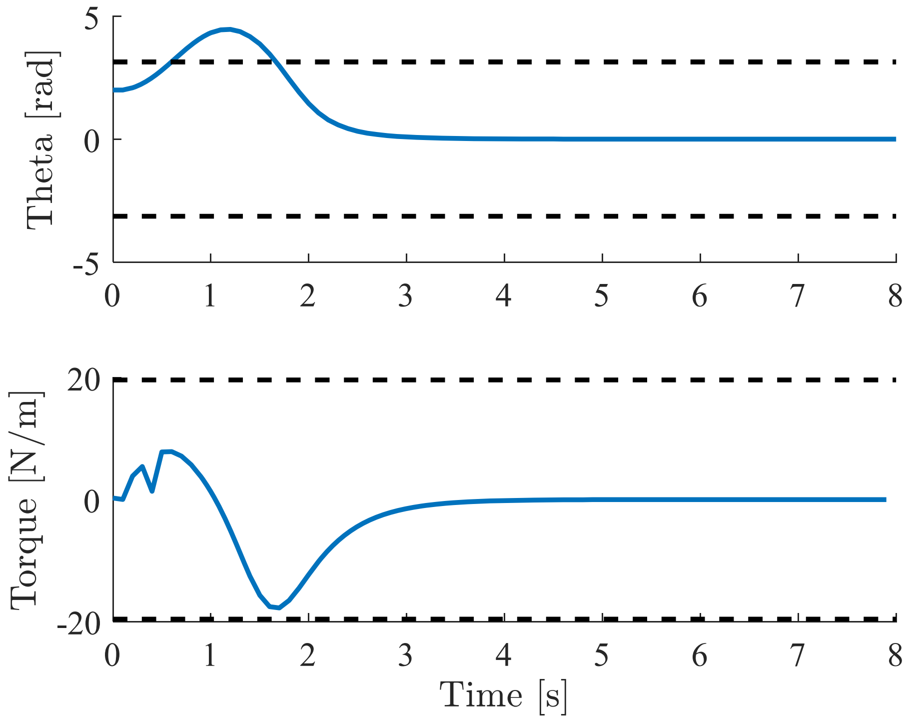
Significantly less use of the actuator

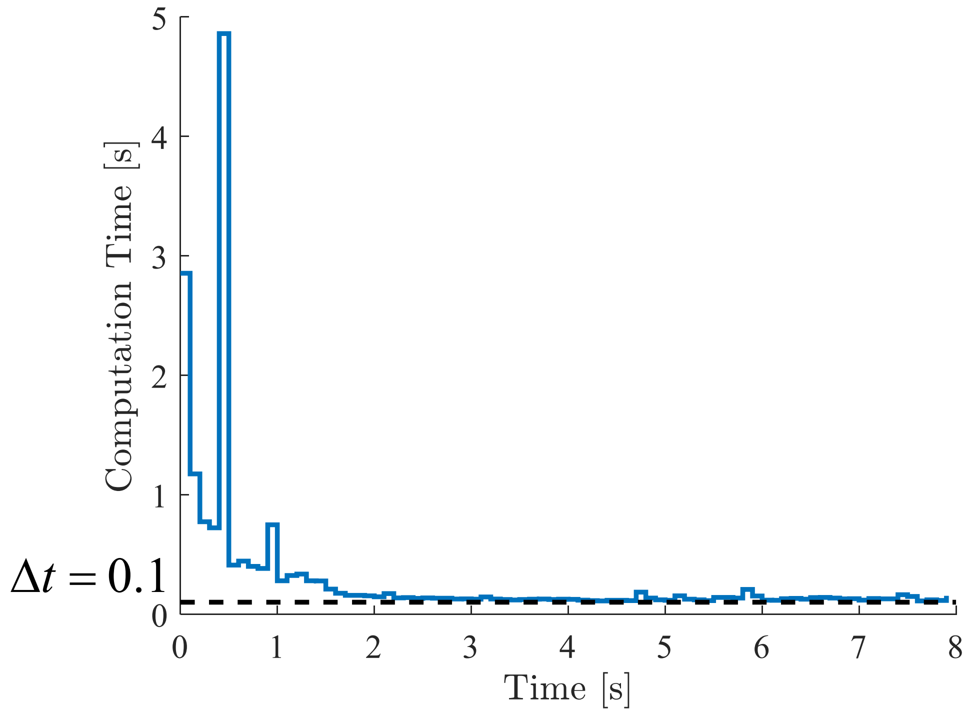
Inverted Pendulum Example (cont.)

- Starting from an initial condition in the range where LQR did not work $x_0 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$
- MPC Solution



Nonlinear MPC solution

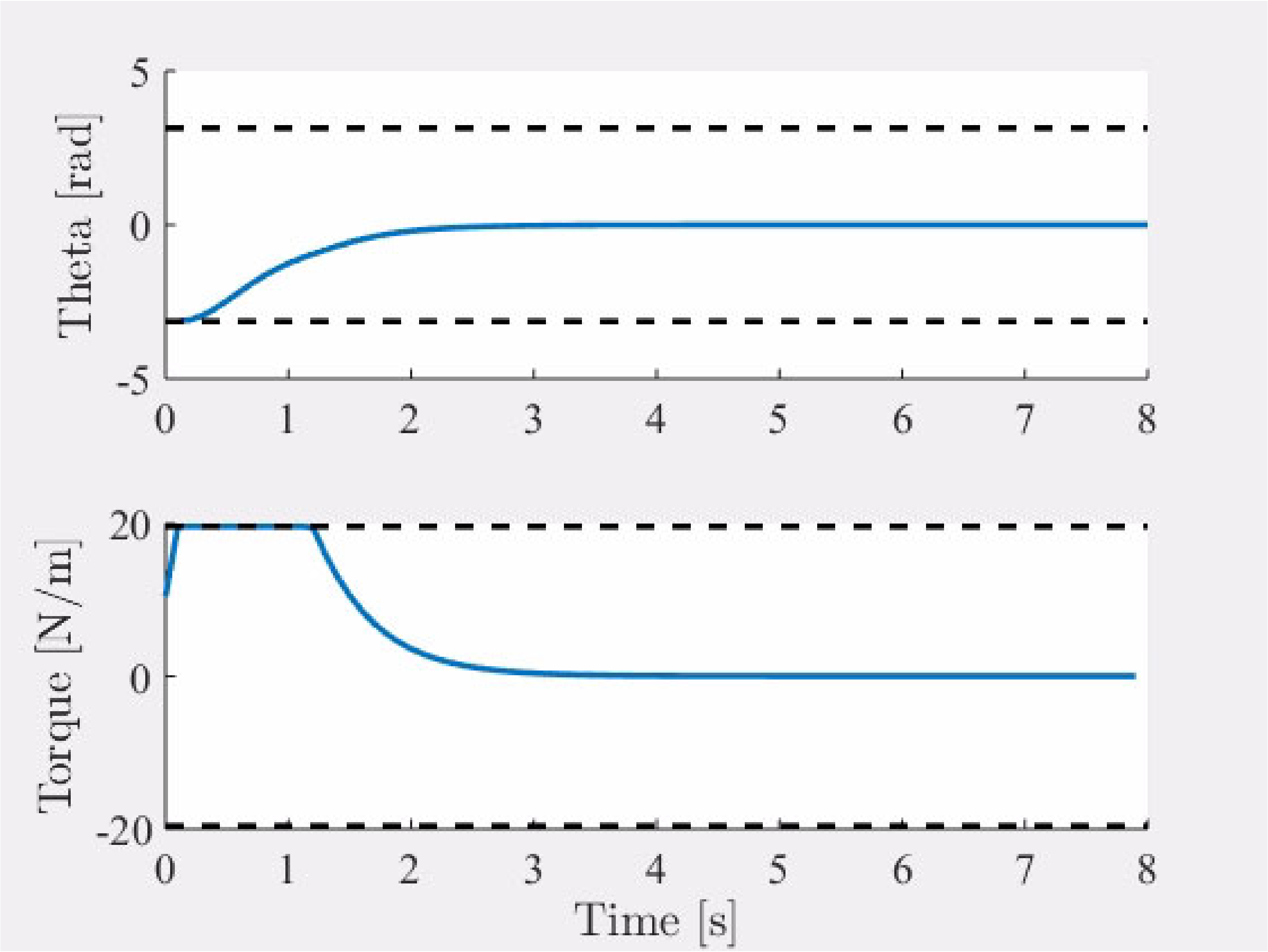


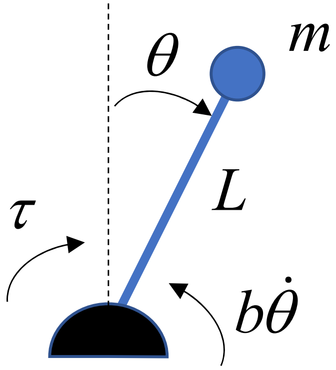


Could not be implemented in real-time!

Inverted Pendulum Example (cont.)

- Range of initial conditions





Inverted Pendulum Example (cont.)

- Range of initial conditions