Advanced Model Predictive Control

Recitation 2
Introduction to Code Framework & Nominal Nonlinear MPC

Alexandre Didier and Jérôme Sieber

ETH Zurich

Fall 2021

MATLAB Code Framework for AMPC 2021

For all recitations we will use a unified MATLAB code framework, which consists of the following parts:

- **Parameter File:** `rec_02.m` defines system, control, and simulation parameters; `get_params.m` loads these parameters to the main file.
- **Main File:** `main.m` initializes and simulates the complete control loop. You will only execute this file.
- **System Class:** `System.m` implements the dynamics, constraints, and disturbance descriptions of a segway system.
- **Control Class:** `Controller.m` is an abstract parent class of all controllers implemented in the recitations.
- **Parameter Estimator Class:** `Parameter_Estimator.m` is an abstract parent class of all parameter estimators implemented in the recitations.

```
code/
├── controllers/
│   ├── MPC.m
│   └── Nonlinear_MPC.m
├── estimators/
├── parameters/
│   ├── get_params.m
│   └── rec_02.m
├── plotting/
├── Controller.m
├── Parameter_Estimator.m
├── System.m
├── main.m
└── setup.m
```

Main File

- This file performs the following actions:
  - load system and control parameters from parameter file,
  - initialize the system and controller objects,
  - simulate a defined number of closed-loop trajectories with a defined number of time steps,
  - plot the results of the closed-loop simulations.
- Execute only this file.

MATLAB Pseudocode:

```
% get parameters and define ...
    system & controller
params = get_params('rec_02');
sys = System(params.sys);
ctrl = MPC(sys, params.ctrl);

% control loop
x(1) = params.sim.x_0;
for i=1:nrTraj
    for j=1:nrSteps
        u = ctrl.solve();
        x(j+1) = sys.step(x(j),u);
    end
end

% plot results
plot(x); plot(u);
```

System Class

- `System` implements a linear and a nonlinear model of a segway.
- The class defines the interfaces between the system and any controller derived from the `Controller` class.
- You should not modify this class.
- However, feel free to go through the code and play with it.

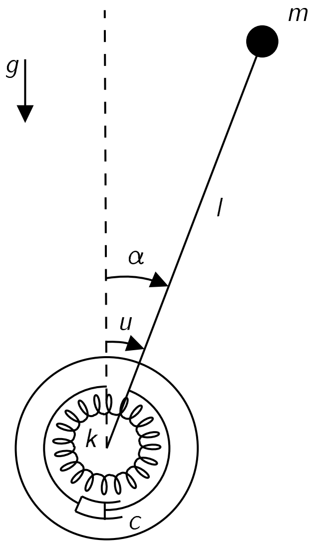
MATLAB Pseudocode:

```
classdef System
%System class for segway system
properties
   % class variables
end

methods
   % class methods
   function obj = System(params)
        %Class Constructor
   end
   function x1 = step(obj,x,u)
        %advance system from ...
              state x with input u
   end
   function update_params(obj, ...
        params)
        %update system parameters
   end
end
end
```

Segway System


[freepik.com]



We only consider the rotational dynamics of the segway and discretize the dynamics using Euler forward:

$$\begin{bmatrix} \alpha(k+1) \\ \dot{\alpha}(k+1) \end{bmatrix} = \begin{bmatrix} \alpha(k) + \delta t \cdot \dot{\alpha}(k) \\ \dot{\alpha}(k) + \delta t \left[-k\alpha(k) - c\dot{\alpha}(k) + {}^{g}/{}_{l} \cdot \sin\alpha(k) + u(k) \right] \end{bmatrix}$$

Controller Class

- All controllers you will implement in the recitations inherit from this class.
- We provide you with a nominal MPC implementation as a reference.
- All controller classes need to define the property `prob`.
- The `solve` method works for both Yalmip and CasADi `prob` objects.

MATLAB Pseudocode:

```
classdef Controller
%Parent controller class
properties
  % class variables
  prob % optimizer/solver object
end

methods
  % class methods
  function obj = ...
        Controller(params)
    %Class Constructor
  end
  function [u, out, info] = ...
        solve(obj, x, vars, verbose)
    %solve optimization ...
        problem with initial ...
        state x and auxilliary ...
        variables vars
  end
end
end
```

Parameter Estimator Class

- All parameter estimators you will implement in the recitations inherit from this class.
- We will use this class only in the second part of the course.

MATLAB Pseudocode:

```
classdef Parameter_Estimator
%Parent parameter estimator class
properties
  % class variables
end

methods
  % class methods
  function obj = ...
        Parameter_Estimator(sys)
        %Class Constructor
  end
end
end
```

Nominal Nonlinear MPC

Nominal Nonlinear MPC Problem:

$$
\begin{aligned}
\min_{x,u} \quad & l_f(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i) \\
\text{s.t.} \quad & \forall i = 0, \dots, N-1 \\
& x_{i+1} = f(x_i, u_i) \\
& x_i \in \mathcal{X}, \;\; u_i \in \mathcal{U} \\
& x_N \in \mathcal{X}_f, \;\; x_0 = x(k)
\end{aligned}
$$

Homework (PS2): Make yourself familiar with the code base. Then, implement the nominal nonlinear MPC problem in the provided `Nonlinear_MPC.m` file. Use the following choices of cost function, dynamics, constraints, and terminal ingredients:

$$
\begin{aligned}
l(x,u) &= x^\top Q\, x + u^\top R\, u \\
f(x,u) &= \begin{bmatrix} x_1 + \delta t \cdot x_2 \\ x_2 + \delta t \left[-k x_1 - c x_2 + {}^{g}/_{l} \cdot \sin x_1 + u\right] \end{bmatrix} \\
\mathcal{X} &= \{x \mid A_x x \le b_x\} \\
\mathcal{U} &= \{u \mid A_u u \le b_u\} \\
l_f(x) &= 0 \\
\mathcal{X}_f &= \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\}
\end{aligned}
$$

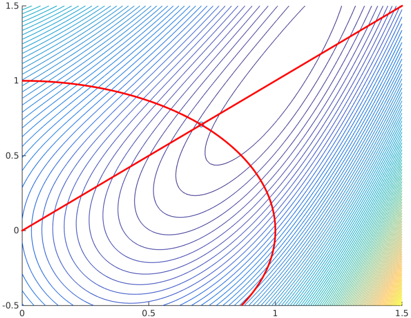
Nonlinear Programming using CasADi

Nonlinear Programming (NLP) Example:

$$\min_{x,y} \quad (1-x)^2 + \left(y - x^2\right)^2$$

$$\text{s.t.} \quad x^2 + y^2 = 1$$

$$x \leq y$$



MATLAB Code:

```
% initialize Opti stack
prob = casadi.Opti();

% define decision variables
x = prob.variable();
y = prob.variable();

% objective
prob.minimize((1-x)^2+(y-x^2)^2);

% constraints
prob.subject_to(x^2+y^2==1);
prob.subject_to(y>=x);

% solve NLP
prob.solver('ipopt');
sol = prob.solve();
```

[Source: CasADi Opti Stack Documentation]

Robustness of Nominal Nonlinear MPC

Homework (PS2): Consider now the same nonlinear segway system but with additive disturbances.

1. Run the cell labelled "Exercise 3a" in `main.m` and observe how the initial state and the disturbance affect the feasibilty of the closed-loop trajectories.
2. Run the cell labelled "Exercise 3b" in `main.m` with different choices of initial states and disturbance sizes. Observe how these two parameters affect the closed-loop trajectories and the cost decrease.