Spring 2020

Stanford
AA 203: Optimal and Learning-based Control
Problem set 4, due on May 18

For this problem set you need to install the following software:

- `http://cvxr.com/cvx/download/` (referred to as CVX)
- `https://www.mpt3.org/` (referred to as MPT)

Problem 1: We return to the inverted pendulum swing-up problem from Problem Set 2. As you might recall, we used iterative LQR to solve the problem. However, we did not account for constraints on the control (which moves the cart horizontally). In practice, due to limitations of the motor used, we often have control constraints. In this problem, we consider we have mapped the limitations of the motor to the control we consider. The control constraints for this problem are as follows: $u \in \mathcal{U} = [-4, 3]$[1].

We will solve this problem using direct methods, relying on Sequential Convex Programming (SCP) as seen in lecture. The key idea is iteratively re-linearizing the dynamics and constructing a convex approximation of the cost function around a nominal trajectory. We will use `CVX` to solve this problem.

a) Given a nominal trajectory $(x^{(k)}, u^{(k)})$, where $x^{(k)} = (x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)})$ and $u^{(k)} = (u_1^{(k)}, u_2^{(k)}, ..., u_n^{(k)})$, write down the convex approximation to the optimal control problem for iteration $k+1$. The derived problem should have linear equality constraints obtained by linearizing the dynamics about the nominal trajectory $(x^{(k)}, u^{(k)})$.

b) Recall that linearization provides a good approximation to the nonlinear dynamics only in a small neighborhood around the nominal trajectory $x^{(k)}, u^{(k)}$. For this reason, the accuracy of the convex model may be poor if x, u deviate far from $x^{(k)}, u^{(k)}$. To ensure smooth convergence, we consider a convex trust region on the state and the input, which is imposed as an additional constraint in the convex optimization problem. We consider a box around the nominal trajectory $x^{(k)}, u^{(k)}$, which can be written as:

$$\mathcal{T} := \{x, u \mid \|x - x^{(k)}\|_\infty \le \rho, \|u - u^{(k)}\|_\infty \le \rho\}$$

Hint: Consider the following parameters, ρ= 0.5

[1]Some DC motors run with better performance (more torque) in one direction than the other, so asymmetrical constraints are definitely common in practice

c) Implement the constraints in the starter code provided in `SCP.m`, which is solved at each iteration of the algorithm.

Hint: For setting up the problem, review lecture 9 and the lecture code which can be found here: ***`github.com/StanfordASL/AA203-Examples`***

d) Run `simulate_scp.m` to run the simulations after you complete the controller. This script first performs the swing up maneuver generated by the shooting method for SCP, and then switches to the stabilizing infinite horizon LQR controller from Problem Set 2. It will generate plots of the state and control trajectories. First, run the simulations without noise (the default), and observe the response. Then, turn on noise (set line 13 of `simulate_scp.m` to true) and run the simulation with noise. Submit the plots for simulations both with and without noise.

Problem 2: Consider the second-order, discrete-time LTI system

$$x(t+1) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t).$$

We want to compute a receding horizon controller for the case where the cost is quadratic, i.e., $p(x_N) = x_N^T P x_N$, $q(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$. Assume (if not otherwise stated) that $Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $R = 0.01$, and that the system is subject to the input constraints

$$-\bar{u} \le u(k) \le \bar{u}, \quad k = 0, \dots, N-1,$$

and to the state constraints

$$\begin{bmatrix} -\bar{x} \\ -\bar{x} \end{bmatrix} \le x(k) \le \begin{bmatrix} \bar{x} \\ \bar{x} \end{bmatrix}, \quad k = 0, \dots, N.$$

Let P_∞ be the solution to the algebraic Riccati equation:

$$P_\infty = A^T P_\infty A + Q - A^T P_\infty B (B^T P_\infty B + R)^{-1} B^T P_\infty A.$$

a) Implement in CVX/MATLAB the receding horizon control strategy for this system.

b) Let $\bar{x} = 5$, $\bar{u} = 0.5$, $N = 3$, $P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $R = 10$, and $X_f = \mathbb{R}^2$. Simulate the closed-loop trajectories with initial states $x(0) = [-4.5, 2]$ and $x(0) = [-4.5, 3]$.

c) Let $\bar{x} = 10$, $\bar{u} = 1$, $N = 2$, $P = P_\infty$, and $X_f = 0$. Discretize the state space (pick a reasonable discretization step) and find the set of initial points leading to feasible closed-loop trajectories converging to the origin (i.e., the domain of attraction for the RHC policy).

d) Let $\bar{x} = 10$, $\bar{u} = 1$, $N = 6$, $P = P_\infty$, and $X_f = 0$. Discretize the state space and find the domain of attraction for the RHC policy.

e) Let $\bar{x} = 10$, $\bar{u} = 1$, $N = 2$, $P = P_\infty$, and $X_f = \mathbb{R}^2$. Discretize the state space and find the domain of attraction for the RHC policy.

f) Let $\bar{x} = 10$, $\bar{u} = 1$, $N = 6$, $P = P_\infty$, and $X_f = \mathbb{R}^2$. Discretize the state space and find the domain of attraction for the RHC policy.

g) Discuss and compare the results in parts c), d), e), and f).

h) Consider the case $\bar{x} = 10$, $\bar{u} = 1$, $P = P_\infty$, and $X_f = 0$. Consider several different values of N and discuss, by presenting simulation experiments, how the trajectory and its cost are affected by the choice of N (you should use an appropriate initial condition).

Problem 3: Consider the discrete-time LTI system $x(t+1) = Ax(t) + Bu(t)$, with

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0.9 & 1 \\ 0 & 0.2 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},$$

and state and control constraints

$$\begin{bmatrix} -5 \\ -5 \\ -5 \end{bmatrix} \leq x(t) \leq \begin{bmatrix} 5 \\ 5 \\ 5 \end{bmatrix}, \qquad -0.5 \leq u(t) \leq 0.5.$$

Using the Matlab-based Multi Parametric Toolbox (MPT), compute and plot the control invariant set for the system. *Hint: look at the example at* *`https://www.mpt3.org/UI/Invariance`.*

Problem 4: Consider the discrete-time LTI system $x(t+1) = Ax(t) + Bu(t)$, with

$$A = \begin{bmatrix} 0.99 & 1 \\ 0 & 0.99 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

and state and control constraints

$$\begin{bmatrix} -5 \\ -5 \end{bmatrix} \leq x(t) \leq \begin{bmatrix} 5 \\ 5 \end{bmatrix}, \qquad -0.5 \leq u(t) \leq 0.5.$$

We wish to synthesize a controller to stabilize the system to the origin while minimizing a quadratic cost function

$$J = x(t_f)^T P x(t_f) + \sum_{t=1}^{t_f-1} \Big(x(t)^T Q x(t) + u(t)^T R u(t) \Big).$$

a) Consider the weights

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad R = 1.$$

Propose a final condition X_f and a final cost P that, together, guarantee asymptotic stability and persistent feasibility of the closed-loop system. Plot the region X_f.

b) Implement an online MPC controller with MPT. (*Hint: have a look at the tutorial at* `https://www.mpt3.org/UI/RegulationProblem`.) Make sure to include the terminal penalty and the terminal set computed in Part 1 (*Hint: see* `https://www.mpt3.org/UI/Filters`). Simulate the performance of the controller for $x_0 = [-4.7,\, 2]$ with an horizon $N = 4$. Plot the resulting trajectory.

c) Convert the online controller from Part 2 to an explicit (offline) MPC controller. Simulate the performance of the controller for the same initial condition $x_0 = [-4.7,\, 2]$ with an horizon $N = 4$. Plot the resulting trajectory.

d) Compare the execution time (both setup time and time required to compute the closed-loop trajectory) for the online controller and the explicit controller.

e) One can show that, for an LTI system with a quadratic cost function, the optimal policy is piecewise affine. Plot the partition of the state space for the controller computed in Part 3, i.e., the regions within the state space such that, within one region, the optimal control law is affine (*Hint: use* `expmpc.partition.plot()` *to plot*).

Problem 5: Consider the continuous-time system

$$\dot{y}(t) + a y(t) = b u(t) \tag{1}$$

We want to control this system, but we do not know the true plant parameters (a, b). You will use *model-reference adaptive control (MRAC)* to match the behaviour of the true plant with that of the reference model

$$\dot{y}_m(t) + a_m y_m(t) = b_m r(t) \tag{2}$$

where (a_m, b_m) are *known* constant parameters, and $r(t)$ is a chosen bounded external reference signal.

a) Consider the control law

$$u(t) = k_r(t) r(t) + k_y(t) y(t) \tag{3}$$

where $k_r(t)$ and $k_y(t)$ are time-varying feedback gains. Write out the differential equation for the resulting closed-loop dynamics. Use this to verify that, if we knew (a, b), the following constant control gains

$$\begin{aligned} k_r^* &:= \frac{b_m}{b} \\ k_y^* &:= \frac{a - a_m}{b} \end{aligned} \tag{4}$$

would make the true plant dynamics perfectly match the reference model.

b) When we do not know (a, b), we need to adaptively update our controller over time. Specifically, we want an *adaptation law* for $k_r(t)$ and $k_y(t)$ to make $y(t)$ tend towards $y_m(t)$ asymptotically. For this, we define the tracking error $e(t) := y(t) - y_m(t)$ and the parameter errors

$$\begin{aligned} \delta_r(t) &:= k_r(t) - k_r^* \\ \delta_y(t) &:= k_y(t) - k_y^* \end{aligned} \tag{5}$$

Determine the differential equation governing the dynamics of $e(t)$, in terms of e, any of its derivatives, y, r, δ_y, δ_r, and suitable constants.

We will consider the adaptation law for k_r and k_y described by

$$\begin{aligned} \dot{k}_r(t) &= -\operatorname{sign}(b)\gamma e(t) r(t) \\ \dot{k}_y(t) &= -\operatorname{sign}(b)\gamma e(t) y(t) \end{aligned} \tag{6}$$

where $\gamma \in \mathbb{R}_{>0}$ is a chosen constant *adaptation gain*. For this adaptation law, we must at least know the sign of b, which indicates in what direction the input $u(t)$ "pushes" the output $y(t)$ in Eq. (1). For example, when modeling a car, you could reasonably assume that an increased braking force slows down the car.

To show that tracking error and parameter errors are stabilized by our chosen control law and adaptation law, we will use Lyapunov theory. While previously we applied Lyapunov theory to discrete-time systems in the context of MPC, we are now dealing with continuous-time systems.

Theorem 1 (Lyapunov): *Consider the continuous-time system $\dot{x} = f(x, t)$, where $x = 0$ is an equilibrium point, i.e., $f(0, t) \equiv 0$. If there exists a continuously differentiable scalar function $V(x, t)$ such that*

- *V is positive definite in x, and*
- *$\dot{V}$ is negative semi-definite in x,*

then $x = 0$ is a stable point in the sense of Lyapunov, i.e., $\|x(t)\|$ remains bounded as long as $\|x(0)\|$ is bounded.

c) Now, consider the state $x := (e, \delta_r, \delta_y)$ and the Lyapunov function candidate

$$V(x) = \frac{1}{2}e^2 + \frac{|b|}{2\gamma}(\delta_r^2 + \delta_y^2) \tag{7}$$

Show that $\dot{V} = -a_m e^2$. Based on Lyapunov theory, what can you say about $e(t)$, $\delta_r(t)$, and $\delta_y(t)$ for all $t \in [0, \infty)$ if $a_m > 0$? Furthermore, apply Barbalat's lemma to $\dot{V}$ to make a stronger statement about $e(t)$ than you originally did with just Lyapunov theory.

Theorem 2 (Barbalat's Lemma): *If a differentiable function $g(t)$ has a finite limit as $t \to \infty$, and if $\dot{g}(t)$ is uniformly continuous, then $\dot{g}(t) \to 0$ as $t \to \infty$.*

Hint: To prove that a function is uniformly continuous, it suffices to show that its derivative is bounded. Lipschitz continuity and thus uniform continuity follow from this.

With a given control law and adaptation law, MRAC proceeds as follows. First, we choose a reference signal $r(t)$ to excite the reference output $y_m(t)$ and construct the input signal $u(t)$, which is used to excite the true model. The output $y(t)$ is then observed and fed back into the control law, and the tracking error $e(t)$ is fed into the adaptation law.

d) Apply MRAC to the unstable plant

$$\dot{y}(t) - y(t) = 3u(t) \tag{8}$$

That is, simulate an adaptive controller for this system that does not have access to the true model parameters $(a, b) = (-1, 3)$. The desired reference model is

$$\dot{y}_m(t) + 4y_m(t) = 4r(t) \tag{9}$$

i.e., $(a_m, b_m) = (4, 4)$. Use an adaptation gain of $\gamma = 2$, and zero initial conditions for y, y_m, k_r, and k_y. Plot both $y(t)$ and $y_m(t)$ over time in one figure, and $k_r(t)$, k_r^*, $k_y(t)$, and k_y^* over time in another figure for $r(t) = 4$. Then repeat this for $r(t) = 4\sin(3t)$. That is, you should have four figures in total. What do you notice about the trends for different reference signals? Why do you think this occurs?

Hint: To do the simulation in MATLAB, form a system of ODEs for either (y, y_m, k_r, k_y) or $(y, e, \delta_r, \delta_y)$, then use `ode45()`.

Learning goals for this problem set:

Problem 1: To learn how to implement direct methods, leveraging Sequential Convex Programming, using CVX

Problem 2: To gain experience with "tuning" MPC controllers using CVX.

Problem 3: To gain experience with control invariant sets.

Problem 4: To gain experience with MPT and implicit/explicit MPC.

Problem 5: To explore the theoretical underpinnings of MRAC, and observe its behaviour on an example system in simulation.