ME 7247: Advanced Control Systems **Fall 2022–23**

Lecture 21: Lyapunov Stability

Tuesday November 22, 2022

Lecturer: Laurent Lessard *Scribe: Youbang Sun*

In this lecture, we study the Lyapunov approach to certifying stability of a nonlinear system. We also revisit the stability of linear systems and constrained LQR problems.

1 Revisiting MPC

Last lecture we discussed the Model Predictive Control (MPC), also known as Receding Horizon Control (RHC). The objective we considered is a constrained LQR problem with infinite horizon. For such problems, it is often difficult or intractable to find the optimal controller. MPC is a popular heuristic that often works well in practice.

The idea is to solve a finite-horizon version of the problem, apply the first action, and re-solve the finite-horizon problem at each subsequent step, again only applying the first action of the solution.

The main issues relevant to MPC are

- Optimality: The solution will generally not be optimal since we are making a finite-horizon approximation at each timestep.
- Recursive Feasibility: If the optimization problem for the current timestep is feasible, it does not necessarily imply that it will be feasible in the future.
- Stability: Even if feasibility is guaranteed, the resulting closed-loop dynamics may enter in sustained oscillations which will never go to zero.

Zero terminal state. If set the terminal state to be $\mathcal{X}_N = \{0\}$ and the initial optimization problem is feasible, then we will have recursive feasibility. However, the constraint of $\mathcal{X}_N = \{0\}$ is rather strong, and there is no guarantee the first step will be feasible.

Terminal cost. We can also set $Q_f = P$; setting the terminal cost to be the unconstrained steady-state LQR cost-to-go matrix. When the horizon is long enough, all the constraints will be eventually be slack (since x_t and u_t will be close to zero). From that point forward, the problem behaves like unconstrained LQR. If this happens, MPC will produce the true optimal solution (MPC is no longer a heuristic; it is exact!). The downside of this approach is that a very large N may be needed, and this could be computationally infeasible.

In this lecture, we will improve these two approaches to make them less restrictive.

2 Lyapunov Functions

We only talked about stability of linear systems so far in this class. Even for constrained linear systems, the previous stability results won't apply. We want to find a way to prove stability of a nonlinear dynamical system.

Generally, a nonlinear system can be defined as

$$x_{t+1} = f(x_t).$$

If the total energy of a mechanical system (kinetic + potential) is dissipating, the energy will always end up at zero. A Lyapunov function is similar to an energy function and the intuition for why it proves stability is similar to that of the dissipating energy function. However, a Lyapunov function need not represent any physical notion of energy.

Definition 2.1. A *Lyapunov function* is a function $V : \mathbb{R}^n \to \mathbb{R}$ that satisfies the properties

(i) V is positive definite: $V(0) = 0$ and $V(z) \geq 0$ if $z \neq 0$.

(ii) V is radially unbounded: If $\|z\| \to \infty$, then $V(z) \to \infty$.

(iii) V decreases along trajectories: $V(f(z)) < V(z)$ for all $z \neq 0$.

If V is a Lyapunov function for the system $x_{t+1} = f(x_t)$, then the origin is asymptotically stable. This means that $\lim_{t\to\infty} x_t = 0$.

A detailed proof of this result can be found in the Supplementary Notes.

3 Warm Up: Lyapunov Functions for Linear Systems

For a linear system $x + t + 1 = Ax_t$, the system is stable if and only if A is Schur-stable. Meanwhile Schur-stability of A is equivalent to the Lyapunov equation

$$A^\mathsf{T} PA - P + Q = 0. \tag{1}$$

having a solution $P \succ 0$ for any $Q \succ 0$.

Recall that for LQR, if we substitute the optimal action with $u_t = Kx_t$, then we have $x_{t+1} = (A + BK)x_t$, and the Riccati Equation (DARE) is written as

$$(A + BK)^\mathsf{T} P(A + BK) - P + (Q + K^\mathsf{T} RK) = 0$$

This relationship also has a connection with Lyapunov functions. In fact, we can prove stability of (1) using the following quadratic Lyapunov function.

$$V(x) = x^\mathsf{T} Px \tag{2}$$

We verify the properties that makes (2) a valid Lyapunov function.

- First property is easily satisfied, since $P \succ 0$.
- Since $P \succ 0$, we have $V(z) = z^\mathsf{T} Pz \geq \lambda_{\min}(P)\|z\|^2$, which proves the second property.

- To prove the third property, use the equality shown in (1),

$$\begin{aligned}V(f(z)) - V(z) &= z^\mathsf{T} A^\mathsf{T} P A z - z P z \\ &= z^\mathsf{T}(A^\mathsf{T} P A - P) z \\ &= -z^\mathsf{T} Q z \\ &< 0 \quad \text{if} \quad z \neq 0.\end{aligned}$$

4 Lyapunov Stability for MPC

We will prove stability of certain MPC schemes by finding an appropriate Lyapunov function. Consider the following Lyapunov function for the N-horizon MPC system.

$$\begin{aligned} V(z) \quad := \quad & \underset{\substack{u_0,\dots,u_{N-1} \\ x_0,\dots,x_N}}{\text{minimize}} && \sum_{t=0}^{N-1} (x_t^\mathsf{T} Q x_t + u_t^\mathsf{T} R u_t) \\ & \text{subject to:} && x_0 = z, \quad x_N = 0, \\ &&& x_{t+1} = A x_t + B u_t, \\ &&& x_t \in \mathcal{X}, \quad u_t \in \mathcal{U}. \end{aligned} \tag{3}$$

We can easily show that this function V satisfies the first and second properties of Definition 2.1. We now prove the third property.

Proof for Property 3. Suppose for x_t, function V takes minimum value when the minimization problem is solved by the following list of variables.

$$V(x_t) \to \left\{u_t^\star, \dots, u_{t+N-1}^\star, x_t^\star, \dots, x_{t+N}^\star\right\} \tag{4}$$

and all the variables satisfy the constraints in $V(x_t)$.

Then for the next time step, since we are following MPC, we have $x_{t+1} = A x_t^\star + B u_t^\star$. Now we can pick a feasible point of the minimization problem in $V(x_{t+1})$, the actual solution of the minimization task will be less of equal to value of $V(x_{t+1})$ using the feasible point.

$$\begin{aligned} V(x_{t+1}) &\le \text{cost when using } \left\{u_{t+1}^\star, \dots, u_{t+N-1}^\star, 0, x_{t+1}^\star, \dots, x_{t+N}^\star, 0\right\} \\ &= \sum_{k=t+1}^{N+t-1} (x_k^{\star\mathsf{T}} Q x_k^\star + u_k^{\star\mathsf{T}} R u_k^\star) \\ &= \underbrace{\sum_{k=t}^{N+t-1} (x_k^{\star\mathsf{T}} Q x_k^\star + u_k^{\star\mathsf{T}} R u_k^\star)}_{V(x_t)} - \underbrace{(x_t^{\star\mathsf{T}} Q x_t^\star + u_t^{\star\mathsf{T}} R u_t^\star)}_{>0} \\ &< V(x_t) \quad \text{for} \quad x_t \neq 0. \end{aligned} \tag{5}$$

Which proves the third property. ■

We have shown that V is a Lyapunov function for the MPC system, therefore the origin is asymptotically stable when using the terminal constraint $x_N = 0$.

5 Enlarging the terminal set for MPC

Suppose that instead of $x_N = 0$, we use $x_N \in \mathcal{X}_N \supseteq \{0\}$. Consider the following Lyapunov candidate for the N-horizon MPC system.

$$\begin{array}{rclll} V(z) & := & \underset{\substack{u_0,\dots,u_{N-1} \\ x_0,\dots,x_N}}{\text{minimize}} & \sum_{t=0}^{N-1} (x_t^\top Q x_t + u_t^\top R u_t) + x_N^\top P x_N & \\ & & \text{subject to:} & x_0 = z, \quad x_N \in \mathcal{X}_N, & (6) \\ & & & x_{t+1} = Ax_t + Bu_t, & \\ & & & x_t \in \mathcal{X}, \quad u_t \in \mathcal{U}. & \end{array}$$

For this problem, we need to first ensure the recursive feasibility, for which we add an additional condition that $\mathcal{X}_N$ must satisfy.

Definition 5.1 (Control Invariance)**.** A state constraint set $\mathcal{X}$ together with an input constraint set $\mathcal{U}$ and dynamical system $x_{t+1} = Ax_t + Bu_t$ is *control invariant* if for all $x \in \mathcal{X}$, we can find $u \in \mathcal{U}$ such that $Ax + Bu \in \mathcal{X}$.

We want to ensure that once x_N enters the terminal set $\mathcal{X}_N$, we can use the steady-state LQR policy $u_t = Kx_t$ and all subsequent x_t will remain in $\mathcal{X}$ forever. This requires:

- for all $x \in \mathcal{X}_N$, we have $(A + BK)x \in \mathcal{X}_N$.
- $\mathcal{X}_N \subseteq \mathcal{X}$.
- $Kx \in \mathcal{U}$ for all $x \in \mathcal{X}_N$.

Note that the set $\mathcal{X}_N = \{0\}$ is one possible choice, but there may be other, larger sets we could use, which would make the problem easier to solve (i.e. the optimization problem will have a better chance of being feasible since we are relaxing the constraints).

To prove that the V given in (6) is a Lyapunov function, we proceed as in the previous case with $x_N = 0$.

$$
\begin{aligned}
V(x_{t+1}) &\le \text{cost when using } \left\{u^\star_{t+1},\dots,u^\star_{t+N-1},u_{N+t+1},x^\star_{t+1},\dots,x^\star_{t+N},x_{N+t+1}\right\} \\
&= \sum_{k=t+1}^{N+t}(x_k^{\star\top}Qx_k^\star+u_k^{\star\top}Ru_k^\star)+x_{t+N+1}^\top Px_{t+N+1} \\
&= \sum_{k=t}^{N+t-1}(x_k^{\star\top}Qx_k^\star+u_k^{\star\top}Ru_k^\star)+(x_{t+N}^{\star\top}Qx_{t+N}^\star+u_{t+N}^{\star\top}Ru_{t+N}^\star)-(x_t^{\star\top}Qx_t^\star+u_t^{\star\top}Ru_t^\star) \\
&\qquad +x_{t+N}^{\star\top}Px_{t+N}^\star-x_{t+N}^{\star\top}Px_{t+N}^\star+x_{t+N+1}^\top Px_{t+N+1} \\
&= \underbrace{\sum_{k=t}^{N+t-1}(x_k^{\star\top}Qx_k^\star+u_k^{\star\top}Ru_k^\star)+x_{t+N}^{\star\top}Px_{t+N}^\star}_{V(x_t)}-(x_t^{\star\top}Qx_t^\star+u_t^{\star\top}Ru_t^\star) \\
&\qquad -x_{t+N}^{\star\top}Px_{t+N}^\star+x_{t+N+1}^\top Px_{t+N+1}+(x_{t+N}^{\star\top}Qx_{t+N}^\star+u_{t+N}^{\star\top}Ru_{t+N}^\star) \\
&= V(x_t)-(x_t^{\star\top}Qx_t^\star+u_t^{\star\top}Ru_t^\star)-x_{t+N}^{\star\top}Px_{t+N}^\star+x_{t+N}^\top(A+BK)^\top P(A+BK)x_{t+N} \\
&\qquad +(x_{t+N}^{\star\top}Qx_{t+N}^\star+x_{t+N}^{\star\top}K^\top RKx_{t+N}^\star) \\
&= V(x_t)-\underbrace{(x_t^{\star\top}Qx_t^\star+u_t^{\star\top}Ru_t^\star)}_{>0}+x_{t+N}^{\star\top}\underbrace{\Big((A+BK)^\top P(A+BK)-P+Q+K^\top RK\Big)}_{=0}x_{t+N}^\star \\
&< V(x_t)\quad\text{for}\quad x_t\neq 0.
\end{aligned}
\tag{7}
$$

So we can relax $\mathcal{X}_N$ from 0 to any set that satisfies the invariance property we have above. Finding the good $\mathcal{X}_N$ is often hard to do in practice and is still a open research problem.

6 Explicit MPC

Can we solve the problem to optimality in general?

Recall the recursive dynamic programming previously,

$$V_t(z)=\min_u\left(z^\top Qz+u^\top Ru+V_{t+1}(Az+Bu)\right)$$

We are still solving the same problem, except now we have additional constraints.

$$V_t(z)=\begin{cases}\underset{u\in\mathcal{U},\,Az+Bu\in\mathcal{X}}{\text{minimize}}\ \left(z^\top Qz+u^\top Ru+V_{t+1}(Az+Bu)\right) & \text{if } z\in\mathcal{X}\\ +\infty & \text{if } z\notin\mathcal{X}\end{cases}\tag{8}$$

If the terminal cost is quadratic, e.g. $V_N(z)=z^\top Pz$, then V_{N-1} will not be quadratic, since it requires accounting for the constraints. However, if z is close enough to zero, the constraints will be slack, and we recover the unconstrained case and find that V_{N-1} is quadratic (with a linear policy $u=Kx$).

But when constraints are active, the inequality in the constraint becomes equality constraints. If our state and input constraints are polytopes (linear constraints), every active constraint corresponds to linear equality constraints. The result of optimizing a quadratic objective with linear constraints is again a quadratic function, so for these cases, the result is also quadratic and the optimal control is also linear in the state. However, the value function will be a *different* quadratic from the unconstrained case and the optimal policy will be a *different* K.

If we look at all possible combinations of the constraints being satisfied (there is a combinatorial explosion here; lots of constraints means a large number of possible combinations can be satisfied). The ultimate result is a piecewise-quadratic function value function with a piecewise-linear control policy.

If we can iterate another step to P_{N-2}, it gets worse. For any of the "pieces" from the previous policy, we have to check all possible combinations of constraints once again, which could further subdivide the state space. Again, we obtain a piecewise-quadratic function value function with a piecewise-linear control policy, but with possibly more pieces than in the previous step.

As we continue to iterate, there are two possible outcomes. The number of pieces could keep growing indefinitely, or it could reach a steady state after a finite number of iterations. If this limiting configuration does not have too many pieces, we can store the K's for each piece (pre-compute them), and then we can implement this using a look-up table. This is *very fast* since at each timestep, no optimization is needed. All we have to do is test the various constraints to see which are active (this determines which controller we should use), and then we look up the corresponding controller and apply it to compute our next action. This is called **Explicit MPC**. The nice thing about this approach is that it is actually *optimal*; we made no approximations in computing this controller!

In practice, if the explicit MPC solution is too complicated to store in memory, we can find an approximation. There are many heuristics to do this; it is an active area of research.

Explicit MPC is a method of choice when the system requires very fast and responsive control (there is no time to solve an optimization problem). An example is high-speed control of quadrotor drones.