16-745 Optimal Control Lecture 10

Reid Graves

February 16, 2025

1 Last Time

- Controllability- Property that system dynamics needs
- Dynamic Programming

2 Today

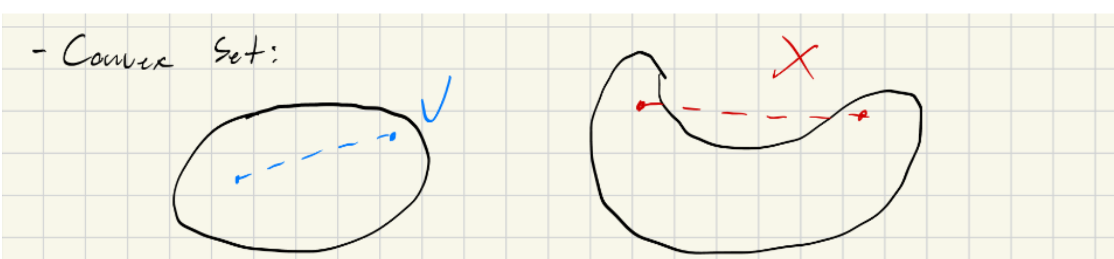
- Convexity Background
- Convex MPC (One of the big concepts of control over last 15 years)

3 Convex Model Predictive Control

- LQR is very powerful, but we often need to reason about constraints
- Often the constraints are simple (e.g. actuator limits)
- Constraints break the Riccati solution, but we can still solve the QP online
- Convex MPC has gotten popular as computers have gotten faster

3.1 Background: Convexity

- Convex set:



  - A line connecting any two points in the set is fully contained in the set
  - Standard examples:
    - Linear Subspaces- $Ax = b$
    - Half spaces/box/polytope- $Ax \leq b$

* Ellipsoids- $x^T P x \leq 1, P > 0$
* Cones- $x_1 \geq ||x_{2:N}||_2$ (Second order cone, "ice cream cone")

- Convex Function:
  - A function $f(x) : \mathbb{R}^n \to \mathbb{R}$ who's epigraph is a convex set

  - Epigraph is everything above the bowl
  - Epigraph on the right is not a convex function
  - Standard examples:
    * Linear- $f(x) = c^T x$
    * Quadratic- $f(x) = \frac{1}{2} x^T Q x + q^T x, \quad Q \geq 0$
    * Norms- $f(x) = |x|$ (Any norm)

- Convex Optimization Problem: minimize a convex function over a convex set
- Standard examples:
  - Linear Program (LP) : $f(x), c(x)$ both linear
  - Quadratic Program (QP): quadratic $f(x)$, linear $c(x)$
  - Quadratically constrained QP (QPQC): quadratic $f(x)$, ellipsoid $c(x)$
  - Second order cone program (SOCP): linear $f(x)$, cone $c(x)$
- Convex optimization problems don't have any spurious local optima that satisfy KKT conditions
- $\Rightarrow$ if you find a local KKT solution, you have the solution
- Practically, Newton's method converges really fast and reliably (5 to 10 iterations max)
- Can bound solution time for real-time control

4 Convex MPC

- Think "Constrained LQR"
- Remember from DP, if we have a cost-to-go function $V(x)$, we can get u by solving a one-step problem:

$$\begin{aligned} u_k &= \arg\min_u l(x,u) + V_{k+1}(f(x,u)) \\ &= \arg\min_u \frac{1}{2} u^T R u + (Ax + Bu)^T P_{k+1} (Ax + Bu) \end{aligned}$$

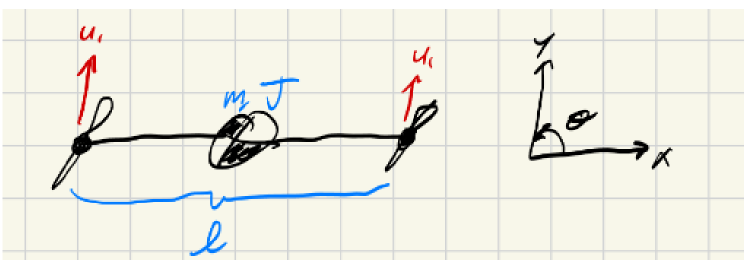
- We can add constraints on u to this one-step problem, but this will perform poorly because $V(x)$ was computed without constraints
- There's no reason I can't add more steps to the one-step problem:

$$\min_{x_{1:H},u_{1:H-1}} \sum_{k+1}^{H-1} \frac{1}{2}x_k^T Q x_k + \frac{1}{2}u_k^T R u_k + x_H^T P_H x_H$$

- $H << N$ is called "Horizon"
- With no additional constraints, MPC ("Receeding horizon horizon") exactly matches LQR for any H
- Intuition: explicit constrained optimization over first H steps gets the state close enough to the reference that the constraints are no longer active and LQR cost to go is valid farther into the future
- In general:
  - A good approximation of $V(x)$ is important for good performance
  - Better $V(x) \Rightarrow$ shorter horizon
  - Longer $H \Rightarrow$ less reliance on $V(x)$

4.1 Example

- Planar Quadrotor



$$m\ddot{x} = -(u1 + u_2)\sin(\theta)$$
$$m\ddot{y} = (u_1 + u_2)\cos(\theta)$$
$$J\ddot{(\theta)} = \frac{1}{2}l(u_2 - u_1)$$

- Linearize about hover:

$$
\begin{aligned}
&\Rightarrow u_1 = u_2 = \frac{1}{2}mg \\
&\Delta\ddot{x} \approx -g\Delta\theta \\
&\Delta\ddot{y} \approx \frac{1}{m}(\Delta u_1 + \Delta u_2) \\
&\Delta\ddot{\theta} \approx \frac{1}{J}\frac{l}{2}(\Delta u_2 - \Delta u_1)
\end{aligned}
$$

$$
\begin{bmatrix} \Delta\dot{x} \\ \Delta\dot{y} \\ \Delta\dot{\theta} \\ \Delta\ddot{x} \\ \Delta\ddot{y} \\ \Delta\ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -g & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \\ \Delta\dot{x} \\ \Delta\dot{y} \\ \Delta\dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{M} & \frac{1}{m} \\ -\frac{l}{2J} & \frac{l}{2j} \end{bmatrix} \begin{bmatrix} \Delta u_1 \\ \Delta u_2 \end{bmatrix}
$$

- MPC cost function

$$
J = \sum_{k=1}^{H-1} \frac{1}{2}(x_k - x_{ref})^T Q (x_k - x_{ref}) + \frac{1}{2}\Delta u_k^T R \Delta u_k + \frac{1}{2}(x_k - x_{ref})^T P_H (x_k - x_{ref})
$$

- Due to linearization, the model can break of going outside the region where the linearization is valid. To deal with this, can add constraint to keep control within valid region. i.e.- keeping θ to a small enough values.