Last Time:

- Infinite-Horizon LQR
- Controllability
- Dynamic Programming

Today:

- Convexity Background
- Convex MPC

---

* Finally: What are the Lagrange Multipliers?

- Recall Riccati derivation from QP:

$$\lambda_n = P_n x_n$$

- From DP:

$$V_n(x) = \frac{1}{2} x^T P_n x$$

$$\Rightarrow \quad \lambda_n = \nabla_x V_n(x)$$

- Dynamics multipliers are cost-to-go gradients!
- Carries over to nonlinear setting (not just LQR)

---

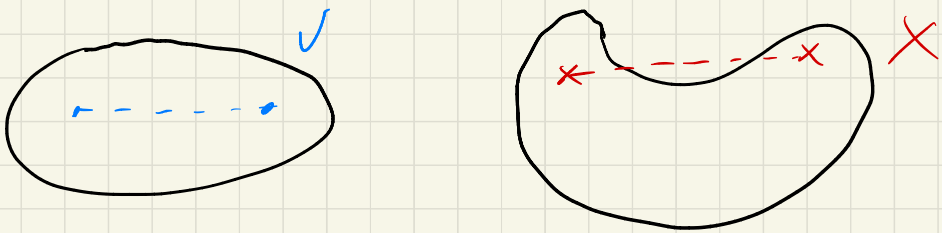
* Convex Model-Predictive Control

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

---

* Background: Convexity

- Convex Set:

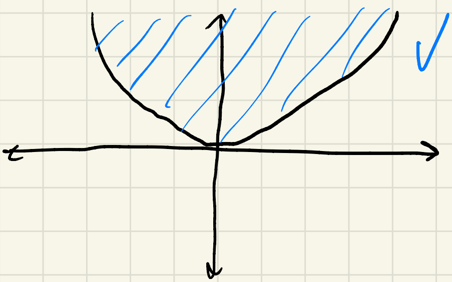


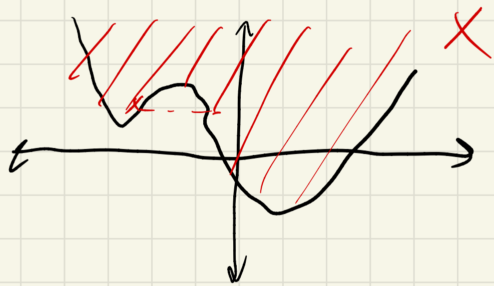
- A line connecting any two points in the set is also also contained in the set.

- Standard Examples:
  - Linear subspace $(Ax = b)$
  - Half space / Box / Polytope $(Ax \le b)$
  - Ellipsoids $(x^T P x \le 1, \ P > 0)$
  - Cones $\underbrace{(\|x_{2:n}\|_2 \le x_1)}_{\text{"second-order cone"}}$

    (standard ice cream cone)

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





- Standard examples:
  - Linear $f(x) = c^T x$
  - Quadratic $f(x) = \frac{1}{2} x^T Q x + q^T x$, $Q > 0$
  - Norms $f(x) = |x|$ (any norm)

- Convex optimization problem: minimizing a convex function over a convex set.

- Standard Examples:

Linear Program (LP): Linear $f(x)$, Linear $C(x)$

Quadric Program (QP): Quadratic $f(x)$, "

Quadratically-Constrained QP (QCQP): " , ellipsoidal $C(x)$

Second-Order Cone Program (SOCP): linear $f(x)$, cone $C(x)$

- Convex problems don't have any spurious local optima that satisfy KKT.

  ⇒ if you find a local KKT solution, you have the global optimum.

- Practically, Newton's method converges really fast and reliably (5~10 iterations max).

  ⇒ can bound solution time for real-time control.

* Convex MPC

- Think about this as "constrained LQR"

- Remember from DP, if we have a cost-to-go function we can get u by solving a one-step problem:

$$u_n = \underset{u}{\operatorname{argmin}} \; l(x_n, u) + V_{n+1}(f(x_n, u))$$

$$= \underset{u}{\operatorname{argmin}} \; \tfrac{1}{2} u^T R u + \tfrac{1}{2}(A_n x_n + B_n u)^T P_{n+1} (A_n x_n + B_n u)$$

- 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

$$\min_{\substack{x_{1:H} \\ u_{1:H-1}}} \sum_{n=1}^{H-1} \tfrac{1}{2} x_n^T Q_n x_n + \tfrac{1}{2} u_n^T R_n u_n + \underbrace{x_H^T P_H x}_{\text{LQR cost-to-go}}$$

$$\text{s.t.} \quad x_{n+1} = A_n x_n + B_n u_n$$

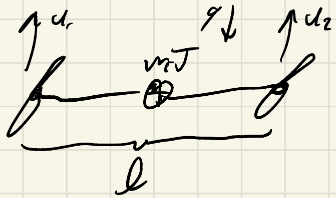
$$x_n \in \mathcal{X} \qquad (\mathcal{X}, \mathcal{U} \text{ convex})$$

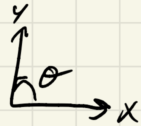
$$\underbrace{u_n \in \mathcal{U}}_{\text{e.g. } u_{min} \le u \le u_{max}}$$

- $H \in \mathbb{N}$ is called "Horizon"
- With no additional constraints, MPC ("receding 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 solution / 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 $V(x)$

* Example :

- Planar Quadrotor





$$m\ddot{x} = -(u_1 + u_2)\sin(\theta)$$

$$m\ddot{y} = (u_1 + u_2)\cos(\theta) - mg$$

$$J\ddot{\theta} = \tfrac{1}{2}\ell(u_2 - u_1)$$

- Linearize about hover :

$$\Rightarrow \quad u_1 = u_2 = \tfrac{1}{2}mg$$

$$\Rightarrow \begin{cases} \Delta\ddot{x} \approx -g\,\Delta\theta \\ \Delta\ddot{y} \approx \frac{1}{m}(\Delta u_1 + \Delta u_2) \\ \Delta\ddot{\theta} \approx \frac{1}{J}\frac{\ell}{2}(\Delta u_2 - \Delta u_1) \end{cases}$$

$$\underbrace{\begin{bmatrix} \Delta\dot{x} \\ \Delta\dot{y} \\ \Delta\dot{\theta} \\ \Delta\ddot{x} \\ \Delta\ddot{y} \\ \Delta\ddot{\theta} \end{bmatrix}}_{\dot{x}} = \underbrace{\left[\begin{array}{c|c} 0 & I \\ \hline \begin{matrix} 0 & 0 & -g \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix} & 0 \end{array}\right]}_{A} \underbrace{\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\theta \\ \Delta\dot{x} \\ \Delta\dot{y} \\ \Delta\dot{\theta} \end{bmatrix}}_{x} + \underbrace{\left[\begin{array}{c} 0 \\ \hline \begin{matrix} 0 & 0 \\ 1/m & 1/m \\ -\ell/2J & \ell/2J \end{matrix} \end{array}\right]}_{B} \underbrace{\begin{bmatrix} \Delta u_1 \\ \Delta u_2 \end{bmatrix}}_{u}$$

- MPC Cost Function:

$$J = \sum_{n=1}^{H-1} \frac{1}{2}(x_n - x_{ref})^T Q (x_n - x_{ref}) + \frac{1}{2}\delta u_n^T R \delta u_n$$
$$+ \frac{1}{2}(x_H - x_{ref})^T P_H (x_n - x_{ref})$$