Last Time:

- Regularization + Duality
- Merit Functions + line search
- Control History

Today:

- Deterministic Optimal Control
- Pontryagin
- LQR

---

* Deterministic Optimal Control

- Continuous time:

$$\min_{\substack{x(t) \\ u(t)}} \underbrace{J(x(t), u(t))}_{\text{cost function}} = \int_{t_0}^{t_f} \underbrace{\ell(x(t), u(t))}_{\text{"stage cost"}}\, dt + \underbrace{\ell_F(x(t_f))}_{\text{"terminal cost"}}$$

$x(t), u(t)$: state + input trajectories

$$\text{s.t. } \dot{x}(t) = f(x(t), u(t)) \quad \leftarrow \text{"dynamics constraint"}$$

(possibly other constraints)

- This is an "infinite-dimensional" optimization problem
- Solutions are open-loop trajectories
- There are a handful of problems with analytic solutions but not many
- We will focus on the discrete-time setting

* Discrete time:

$$\min_{\substack{x_{1:N} \\ u_{1:N-1}}} \quad J(x_{1:N}, u_{1:N-1}) = \sum_{k=1}^{N-1} \ell(x_k, u_k) + \ell_F(x_N)$$

$$\text{s.t.} \quad x_{k+1} = f(x_k, u_k)$$

$$u_{min} \leq u_k \leq u_{max} \quad \forall k \quad \leftarrow \text{"torque limits"}$$

$$c(x_k) \geq 0 \quad \forall k \quad \leftarrow \text{obstacle/safety constraints}$$

- This is a finite-dimensional problem
- Samples x_k, u_k are often called "knot points"
- Continuous → discrete uses integration (e.g. Runge-Kutta)
- discrete → continuous using interpolation

* Pontryagin's Minimum Principle

- Also called "Maximum Principle" if you maximize a reward
- First-order necessary conditions for a deterministic optimal control problem
- In discrete time, special case of KKT conditions

- Given:

$$\min_{\substack{x_{1:N} \\ u_{1:N-1}}} \sum_{n=1}^{N-1} \ell(x_n, u_n) + \ell_F(x_N)$$

$$\text{s.t. } x_{n+1} = f(x_n, u_n)$$

- We can form the Lagrangian:

$$L = \sum_{n=1}^{N-1} \ell(x_n, u_n) + \lambda_{n+1}^T (f(x_n, u_n) - x_{n+1}) + \ell_F(x_N)$$

- This result is usually stated in terms of the "Hamiltonian"

$$H(x, u, \lambda) = \ell(x, u) + \lambda^T f(x, u)$$

- Plug H into L:

$$L = H(x_1, u_1, \lambda_2) + \left[\sum_{n=2}^{N-1} H(x_n, u_n, \lambda_{n+1}) - \lambda_n^T x_n\right] + \ell_F(x_N) - \lambda_N^T x_N$$

Note change to indexing

- Take derivatives w.r.t. x and λ:

$$\frac{\partial L}{\partial \lambda_n} = \frac{\partial H}{\partial \lambda_n} - x_{n+1} = f(x_n, u_n) - x_{n+1} = 0$$

$$\frac{\partial L}{\partial x_n} = \frac{\partial H}{\partial x_n} - \lambda_n^T = \frac{\partial \ell}{\partial x_n} + \lambda_{n+1}^T \frac{\partial f}{\partial x_n} - \lambda_n^T = 0$$

$$\frac{\partial L}{\partial x_N} = \frac{\partial \ell_F}{\partial x_N} - \lambda_N^T = 0$$

- For u we write the min explicitly to handle torque limits:

$$u_n = \underset{u}{\operatorname{argmin}} \; H(x_n, u, \lambda_{n+1})$$
$$\text{s.t. } u \in \mathcal{U}$$

shorthand for "in feasible set" e.g. $u_{min} \le u \le u_{max}$

- In Summary:

$$x_{n+1} = \nabla_\lambda H(x_n, u_n, \lambda_{n+1})$$
$$\lambda_n = \nabla_x H(x_n, u_n, \lambda_{n+1})$$
$$u_n = \underset{u}{\operatorname{argmin}} \; H(x_n, u, \lambda_{n+1})$$
$$\text{s.t. } u \in \mathcal{U}$$
$$\lambda_N = \frac{\partial l_F}{\partial x_N}$$

- These can be stated in continuous time:

$$\dot{x} = \nabla_\lambda H(x, u, \lambda)$$
$$-\dot{\lambda} = \nabla_x H(x, u, \lambda)$$
$$u = \underset{\tilde{u}}{\operatorname{argmin}} \; H(x, \tilde{u}, \lambda)$$
$$\text{s.t. } \tilde{u} \in \mathcal{U}$$
$$\lambda(t_f) = \frac{\partial l_F}{\partial x}$$

* Some Notes:

- Historically many algorithms were based on integrating ODEs forward + backward to do gradient descent
- These are called "indirect" and/or "shooting" methods
- In continuous time $\lambda(t)$ is called "costate" trajectory
- These methods have largely fallen out of favor as computers have improved.

* LQR Problem

$$\min_{\substack{x_{1:N} \\ u_{1:N-1}}} \quad J = \sum_{k=1}^{N-1} \frac{1}{2} x_k^T Q_k x_k + \frac{1}{2} u_k^T R_k u_k + \frac{1}{2} x_N^T Q_N x_N$$

$$\text{s.t.} \quad x_{k+1} = A_k x_k + B_k u_k$$

$$Q \succeq 0, \quad R > 0$$

- Can (locally) approximate many nonlinear problems
- Computationally tractable
- Many extensions e.g. infinite horizon, stochastic, etc.
- "Time invariant" if $A_k = A$, $B_k = B$, $Q_k = Q$, $R_k = R$ $\forall k$ ("time varying" otherwise).

* LQR with Indirect Shooting:

$$x_{n+1} = \nabla_\lambda H(x_n, u_n, \lambda_{n+1}) = Ax_n + Bu_n$$

$$\lambda_n = \nabla_x H(x_n, u_n, \lambda_{n+1}) = Qx_n + A^T \lambda_{n+1}$$

$$\lambda_N = Q_N x_N$$

$$u_n = \nabla_u H(x_n, u_n, \lambda_{n+1}) = 0 \Rightarrow -R^{-1} B^T \lambda_{n+1}$$

- Procedure:

1) Start with initial guess $u_{1:N-1}$
2) Simulate/rollout to get $x_{1:N}$
3) Backward pass to get λ, δu
4) Rollout with line search on δu
5) Go to 3 until convergence

* Example:

- "Double Integrator"

$$\dot{x} = \begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} q \\ \dot{q} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

($\dot{q}$: acceleration; q: position; $\dot{q}$: velocity; u: force)

- Think of this as a brick sliding on ice (no friction)

$$x_{n+1} = \underbrace{\begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix}}_{A} \begin{bmatrix} q_n \\ \dot{q}_n \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{1}{2}h^2 \\ h \end{bmatrix}}_{B} u_n$$

(h: time step)