Last time:

- Robust Control

Today:

- Tips + Tricks
- Historical Overview

---

* Problem Scaling:

- Units matter!
- For "Human-scale" things, meters, kg, sec. are often appropriate.
- For very large or small systems, you may want something else.
- Generally choose units so that things in your problem are O(1)
- For micro robots (insect scale) this might be mm, ms, mg.
- For spacecraft this might be 1000 Km, hours.
- Poorly scaled units lead to ill-conditioning of Jacobian matrices. MKS for a LEO satellite:

$$r \simeq 7 \times 10^6 \text{ meters}, \quad v \simeq 7 \times 10^3 \text{ m/sec.}$$

$$\dot{v} \simeq 8 \text{ m/s}^2 \Rightarrow \text{cond}(A) \simeq 9 \times 10^6$$

⇒ losing 6~7 digits of accuracy when solving $Ax = b$

* Initial Guesses:

- Since Newton's method only converges to nearest local fixed point, initial guess can matter a lot.

- If you can generate a feasible guess w.r.t. constraints, its often a good idea.

- Its often a good idea to give (nearly) dynamically feasible guesses to DIRCOL (e.g. do a rollout)

- For DDP its a good idea to start with stable "trim conditions" that won't blow up (e.g. hover or gravity compensation torques).

- For open-loop unstable systems do the backward pass of DDP first so that the forward rollout has stabilizing feedback and doesn't blow up.

* Cost Shaping

- cost and constraint functions are how we encode behaviors.

- Think carefully about what should be a cost vs. constraints.

- Non-quadratic costs are OK! Try to come up with smooth, non-negative, functions that go to zero when what you want is achieved.

- If you have a pretty good idea of what the trajectory of part of the system should look like vs. time (e.g. CoM, or a foot) you can put a quadratic tracking cost on it:

$$\sum_{n=1}^{N-1} (x_n - g(\tilde{x}_n))^T Q (x_n - g(\tilde{x}_n)) + (u_n - u_{ref})^T R (u_n - u_{ref})$$

reference trajectory

gravity compensation

- You can seed TrajOpt with results from sample-based or other planners this way.