1/22: Dynamics & integrators

Announcements:

- intro survey out, due Thurs (1/29)
- HW0 out, not graded
- HW1 out Thurs, due 2/12

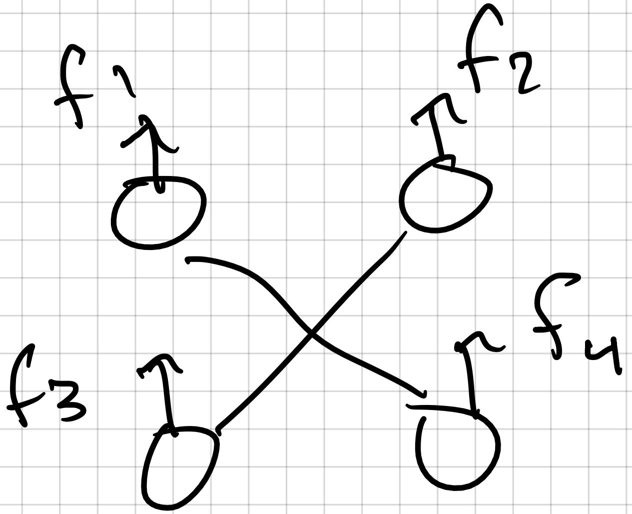
Last time:

- what is a dynamical system?
- numerical integrators

Today:

- Linear(ized) systems
- Numerical differentiation

Recall: $\quad$ <u>state</u>: $\quad x = \begin{bmatrix} p \in \mathbb{R}^3 \\ R \in SO(3) \\ \dot{p} \in \mathbb{R}^3 \\ w \in \mathbb{R}^3 \end{bmatrix}$



drone

$$u = [f_1, f_2, f_3, f_4]$$

$$\dot{x} = f(x, u)$$

$$\boxed{\dot{x} = Ax + Bu}$$

linear system

Recall: Taylor expansions

Consider x^*, u^* e.g. hover

$$f(x^* + \delta x, u^* + \delta u)$$

$$\approx \boxed{f(x^*, u^*) + \overset{=A}{\left.\frac{\partial f}{\partial x}\right|_{x^*, u^*}} \delta x + \overset{=B}{\left.\frac{\partial f}{\partial u}\right|_{x^*, u^*}} \delta u} + \cancel{H.O.T.} \to 0$$

$$\dot{\left(x^* + \delta x\right)} = f(x^* + \delta x, u^* + \delta u)$$

$$\dot{x}^* + \dot{\delta x} = f(x^*, u^*) + \dot{\delta x} \approx f(x^*, u^*) + \frac{\partial f}{\partial x}\delta x + \frac{\partial f}{\partial u}\delta u$$

$$\dot{\delta x} \approx A\delta x + B\delta u$$

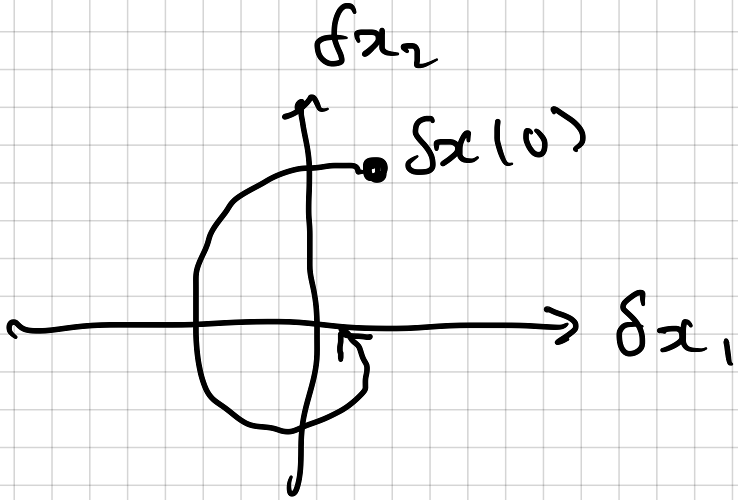
Loosely, stability means remaining near the origin.

For linear systems, we look @ asymptotic stability

<u>Def</u> (Asymptotic stability)

We say a system $\dot{x} = Ax$ is asymptotically stable if

$$\lim_{t \to \infty} x(t) = 0$$



<u>Fact:</u> $\dot{\delta x} = A\,\delta x$ is asymptotically stable iff $\mathrm{Re}\{\lambda_i(A)\} < 0 \quad \forall i$

"real parts of all eigenvalues"

Intuition: $\delta x \in \mathbb{R}$

$$\dot{\delta x} = -\lambda\,\delta x \implies \underline{\delta x(t) = \exp(-\lambda t)\cdot \delta x(0)}$$

Fact: For $\delta x_{k+1} = A\,\delta x_k$,

$$\lim_{k\to\infty} x_k = 0 \quad \text{iff} \quad |\lambda_i(A)| < 1 \quad \forall i.$$

Intuition: $\delta x_k = A\,\delta x_{k-1} = \cdots = A^k \delta x_0$

For $\delta x_k \to 0$, we need $(A)^k \to 0$,
which implies $|\lambda_i(A)| < 1$.

The rest of this lecture:

How do we actually compute
$A = \frac{\partial f}{\partial x}$, $B = \frac{\partial f}{\partial u}$

Aside: In this class, we'll try to make few assumptions on the system dynamics.

$$x_{k+1} = f(x_k, u_k)$$

- physics + numerical integration
- automata
- physics simulation
- learned dynamics

Q: How do we approximate / compute these Jacobians of dynamics?

How do we compute $\frac{\partial f}{\partial x}$, $f : \mathbb{R}^n \to \mathbb{R}^m$

$$f(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{bmatrix} \qquad \frac{\partial f}{\partial x} = \begin{bmatrix} \nabla_x f_1(x)^T \\ \vdots \\ \nabla_x f_m(x)^T \end{bmatrix}$$

$$\underset{\in \mathbb{R}^m}{\partial f} = \frac{\partial f}{\partial x} \underset{\in \mathbb{R}^n}{\partial x}$$

m

1. Analytic 2. Finite diff 3. Autodiff

Finite differencing

Only do this in the worst case!

<u>Idea:</u> $f(x+\delta x) = f(x) + \left.\frac{\partial f}{\partial x}\right|_x \delta x + H.O.T.$

$\delta x = \varepsilon \underline{e_i}$

$e_1 = [1 \; 0 \; 0 \; \dots \; 0]$

$e_n = [0 \; 0 \; \dots \; 1]$

$$\frac{\partial f}{\partial x}\delta x = \begin{bmatrix} \nabla f_1^T \\ \vdots \\ \nabla f_m \end{bmatrix} \varepsilon \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \varepsilon j_1$$

calls: $n+1$

$$\frac{\partial f}{\partial x} = \underbrace{\begin{bmatrix} \underline{j_1} & \cdots & \underline{j_n} \end{bmatrix}}$$

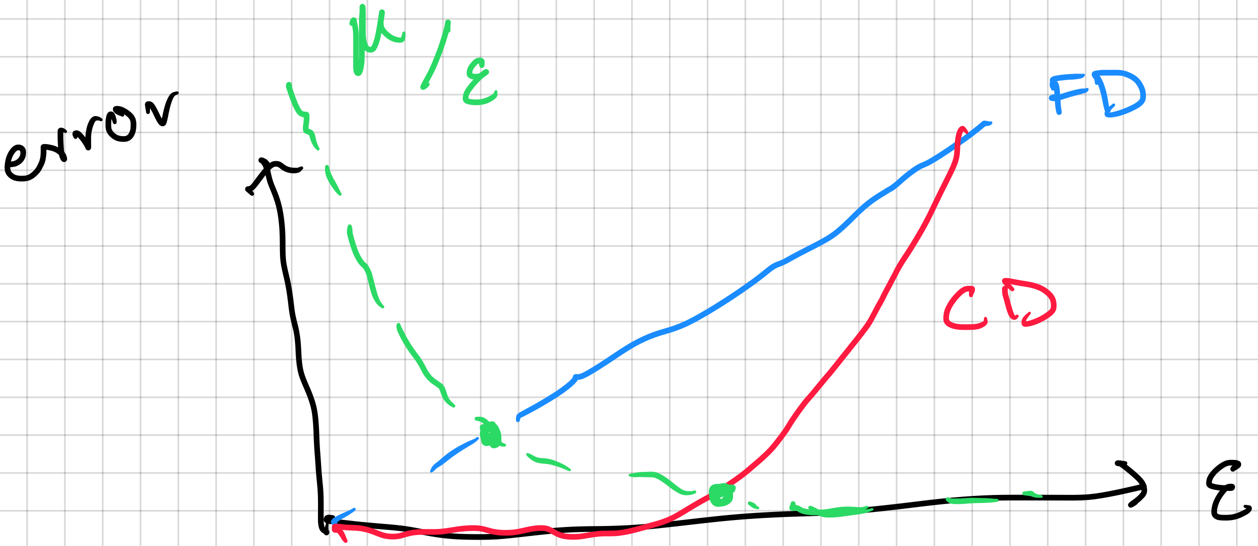
$$\underbrace{\boxed{\frac{f(x+\varepsilon e_i) - f(x)}{\varepsilon} \approx j_i}}_{\text{apply for each column}}$$

we can trade off compute for acc.

central difference

$$\boxed{j_i \approx \frac{f(x+\varepsilon e_i) - f(x-\varepsilon e_i)}{2\varepsilon}}$$

error: $O(\varepsilon^2)$, requires $2n$ calls



μ: machine precision (smallest # you can represent)

$$j_i = \frac{f(x+\varepsilon e_i) - f(x) + \mu}{\varepsilon}$$

Another idea: automatic differentiation

```
def f(x: Array):
    ...
    return y
```

$$h(x) = f_N(\cdots f_2(f_1(x))\cdots)$$

$y_1 = f_1(x)$ $\quad \frac{\partial f_1}{\partial x}$

$y_2 = f_2(y_1)$ $\quad \vdots$

$\vdots$

$h(x)$ " $y_N = f_N(y_{N-1})$ $\quad \frac{\partial f_N}{\partial y_{N-1}}$

$$\overset{w^T}{} \frac{\partial h}{\partial x} = \frac{\partial f_N}{\partial y_{N-1}} \cdot \frac{\partial f_{N-1}}{\partial y_{N-1}} \cdots \frac{\partial f_1}{\partial x} \cdot v$$

why not?

- memory
- compute

Jv . Jacobian-vector

$w^T J$ vector-Jacobian