1/19 RECITATION

- linear systems
- taylor series

Dynamic System

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

(x: state, u: control)

$$x = \begin{bmatrix} q \\ v \end{bmatrix} \begin{matrix} \text{- configuration} \\ \text{- velocity} \end{matrix}$$

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

Dynamic systems can be represented as continuous time Ordinary Differential Equations (ODES), or in discrete time as difference equations.

1st-order ODEs

$$\ddot{p} = \frac{F}{M}$$ - 2nd order ODE

$$x = \begin{bmatrix} p \\ \dot{p} \end{bmatrix}, \text{ so } \underbrace{\begin{bmatrix} \dot{p} \\ \ddot{p} \end{bmatrix}}_{\dot{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \underbrace{\begin{bmatrix} p \\ \dot{p} \end{bmatrix}}_{x} + \begin{bmatrix} 0 \\ F/M \end{bmatrix}$$

Any order ODE can be represented as a first order ODE in state space, where we make up a new state to contain all of the orders of the variable. In mechanics/robotics, we often write down equations of motion that are second derivatives, and converting these to first order ODEs is easy. To convert an N-order ODE in p to a 1st order ode, we just need to create a state x that contains the 0 - (N-1)th derivatives of the value.

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

$$\dot{x} = A(t)\,x + B(t)\,u$$

Is it linear? When we say the dynamics are "linear", we really mean "linear in x and u". This means the output of the function is a linear combination of x's and u's. All equations linear in x and u can be written in a form where the matrices A and B do not have any x or u. If you can write down the ODE this way, it must be linear in x and u. This works for continuous time and discrete time equations.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} e^t & \sin(t) \\ \cos t & t^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} t! & e^t \\ t^3 & t^2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

A B

No x's or u's in there to be linear

We have not mentioned time yet, the equation does not have to be linear in t, the one listed to the left is a linear time varying (LTV) system, where the matrices A and B vary with t. This is still linear and easy for us to deal with.

When talking $f(x, u)$

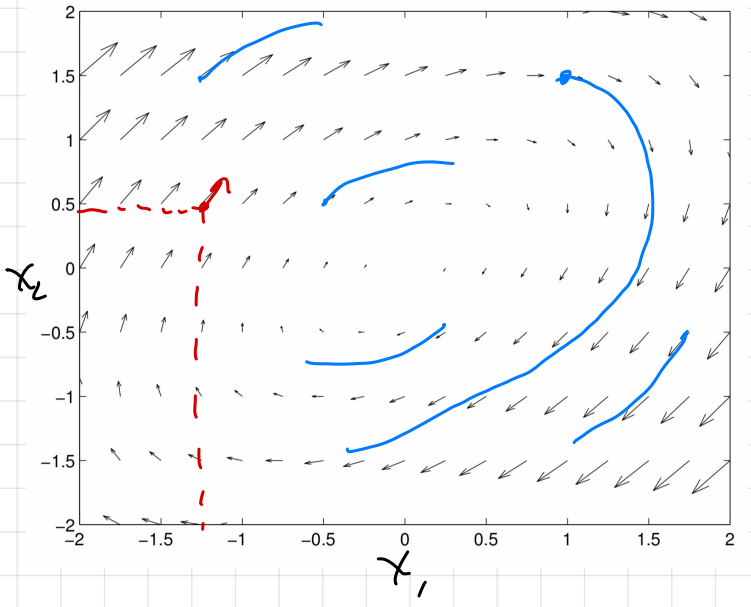
Zac "It's linear!" ⇒ "It's linear in x and u"

Linear Dynamical System

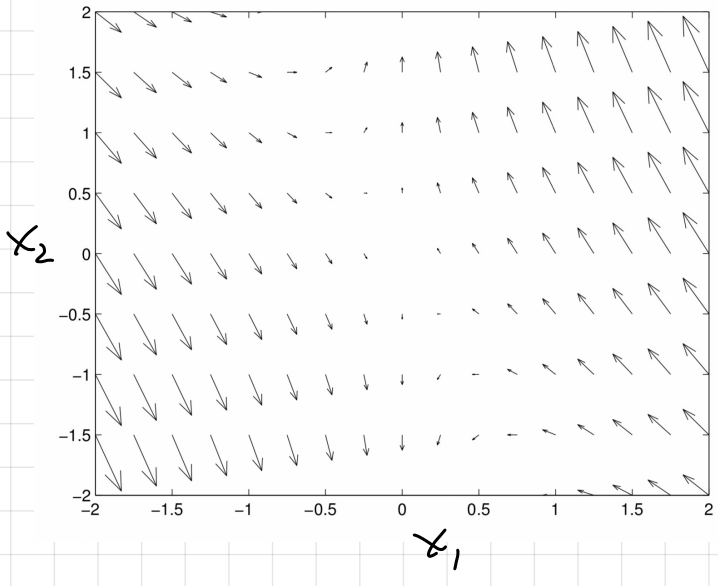
We make a big deal about this because linear ODEs are the only ones we can absolutely solve. Most ODEs are unsolvable, even for simple well-studied systems like a double pendulum. Linear dynamics also make our lives much easier when it comes time to control these systems with optimal control.

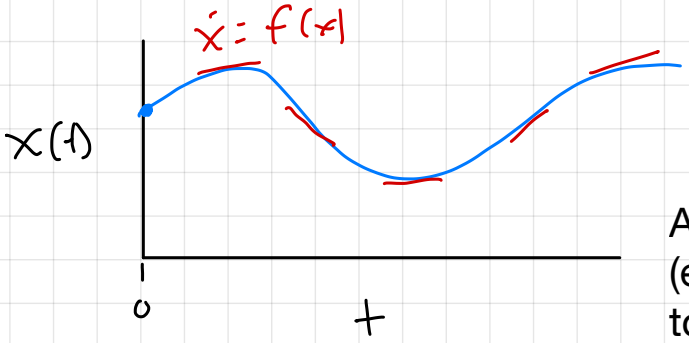
$$\dot{x} = Ax \qquad x \in \mathbb{R}^2$$

$$\dot{x} = \begin{bmatrix} -0.5 & 1 \\ -1 & 0.5 \end{bmatrix} x$$



$$\dot{x} = \begin{bmatrix} -1 & 0 \\ 2 & 1 \end{bmatrix} x$$



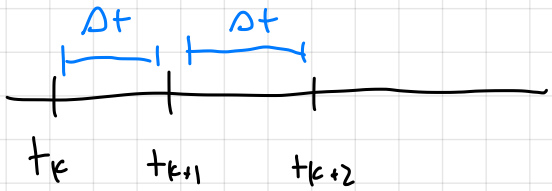


Any linear ODE in this form, can be solved for (exactly) with a matrix exponential. All we have to specify is an initial condition and the amount of time that has passed. By choosing dt to be our timestep, we can discretize any linear ODE into a linear difference equation using this matrix exponential.

$$\dot{x} = Ax$$

$$x(t+\Delta t) = \left(e^{A\Delta t}\right) x(t)$$

↑ matrix exponential



$$x_{k+1} = \underbrace{\left(e^{A\Delta t}\right)}_{A_d} x_k$$

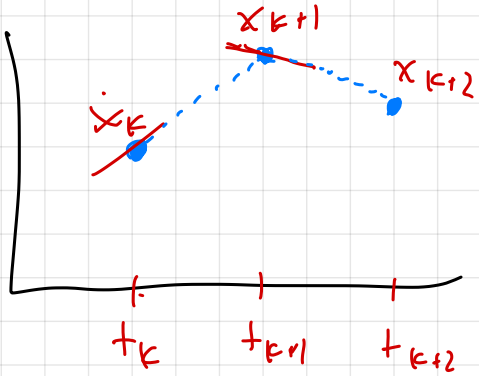
Nonlinear ODEs almost never have closed-form solutions. However, we can usually just solve them numerically using any numerical integrator (Euler, midpoint, RK4, ode45 in matlab, etc). These methods call the ODE a bunch and use the values to build an approximate solution to the ODE.

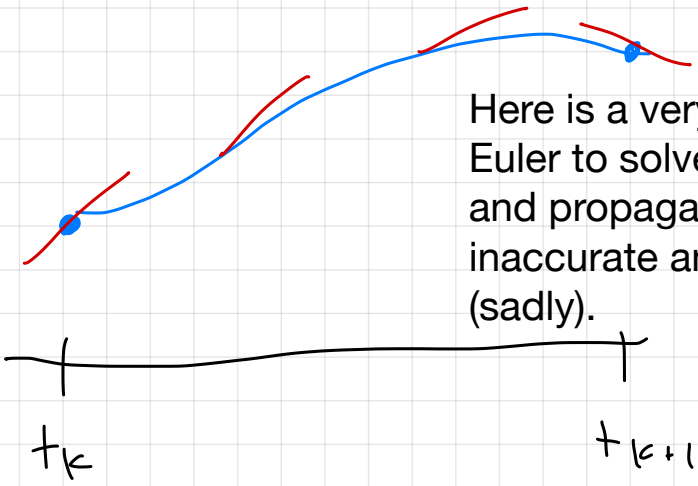
Linear	Nonlinear
$\dot{x} = Ax$	$\dot{x} = f(x)$
Solve w/ e^{At} in closed form	numerically solve w/ euler, rk4, midpoint, (ODE45 in matlab)

Numerically Solve ODE w/ Euler

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

$$x_{k+1} = x_k + \dot{x}_k \cdot \Delta t$$





Here is a very simple example, using forward Euler to solve an ODE by just taking the slope and propagating it a timestep. This is both inaccurate and very common in robotics (sadly).

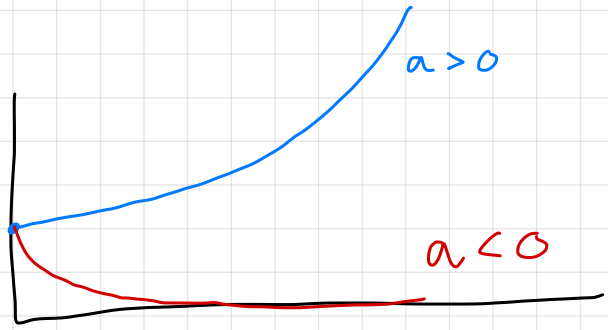
Stability

Continuous $\dot{x} = Ax$

all(Real(eigs(A)) < 0) == stable

$\dot{x} = ax, \quad x \in \mathbb{R}$

$x_{k+1} = e^{a\,dt}\, x$

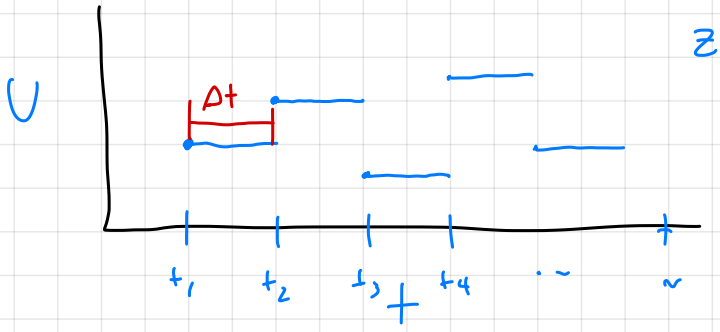


discrete $x_{k+1} = A_d\, x_k$

all(norm(eigs(A)) < 1 : stable

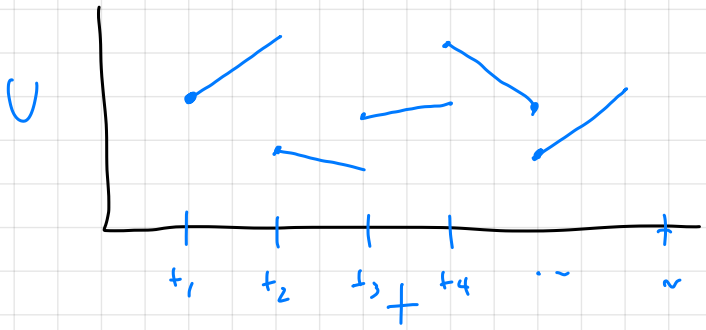
How to discretize forced linear ODE's

$$\dot{x} = [A]x + [B]u$$



Zero-order-hold

ZOH



A forced linear ODE is just one where we have a u in there. We can also solve these in closed form with a matrix exponential, but first let's be specific about what this control signal should look like between time steps. A zero-order hold (ZOH) is very common, where we assume the control does not change in between time steps, but simply gets a new value at each control call. We can also use higher order holds, like a first-order hold (FOH) where we specify a value and a first derivative. All of these result in linear ODE's (even higher order holds), and we can still solve them with our matrix exponential.

for Δt

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

$$\dot{u} = 0$$

⇓ equivalent

$$\begin{bmatrix}\dot{x}\\ \dot{u}\end{bmatrix} = \begin{bmatrix}A & B\\ 0 & 0\end{bmatrix}\begin{bmatrix}x\\ u\end{bmatrix}$$

$$\begin{bmatrix}x_{k+1}\\ u_{k+1}\end{bmatrix} = \left(e^{\begin{bmatrix}A & B\\ 0 & 0\end{bmatrix}\Delta t}\right)\begin{bmatrix}x_k\\ u_k\end{bmatrix}$$

$$\dot{z} = Dz$$

$$\begin{bmatrix}x_{k+1}\\ u_{k+1}\end{bmatrix} = \begin{bmatrix}A_d & B_d\\ 0 & I\end{bmatrix}\begin{bmatrix}x_k\\ u_k\end{bmatrix}, \quad x_{k+1} = A_d x_k + B_d u_k$$

$$z_{k+1} = e^{D\Delta t} z$$

$$\begin{bmatrix}A_d & B_d\\ 0 & I\end{bmatrix} = e^{\begin{bmatrix}A & B\\ 0 & 0\end{bmatrix}\Delta t}$$

Discretization of Linearized Dynamics

In order to discretize the continuous system (14), the matrix exponential will be used. For a generic homogeneous linear ODE of the form $\dot{x} = Ax$, the solution for x after a time δt, can be expressed using the matrix exponential and the initial condition:[13,14]

$$\dot{x} = Ax, \tag{15}$$
$$x(t_0 + \delta t) = \exp(A \cdot \delta t)x(t_0). \tag{16}$$

For a forced affine ODE, where the control input and affine forcing term are assumed constant over a time step, the state can simply be augmented with these terms,

$$\begin{bmatrix} \dot{x} \\ \dot{u} \\ \dot{d} \end{bmatrix} = \begin{bmatrix} Ax + Bu + d \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} A & B & I_n \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ u \\ d \end{bmatrix}, \tag{17}$$

and this system can be discretized with a sample time of δt in the same way as (16)

$$\begin{bmatrix} x_{t+1} \\ u_{t+1} \\ d_{t+1} \end{bmatrix} = \exp\left(\begin{bmatrix} A & B & I_n \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \delta t \right) \begin{bmatrix} x_t \\ u_t \\ d_t \end{bmatrix}. \tag{18}$$

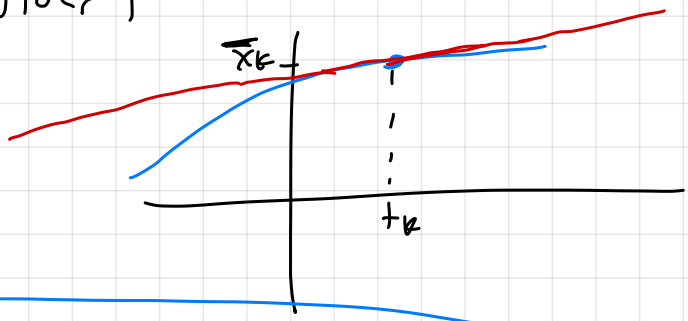
Finally, we obtain in the following difference equation,

$$x_{t+1} = A_d x_t + B_d u_t + D_d d_t, \tag{19}$$

where the transition matrices come from the matrix exponential,

$$\begin{bmatrix} A_d & B_d & D_d \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} = \exp\left(\begin{bmatrix} A & B & I_n \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \cdot \delta t \right). \tag{20}$$

Taylor Series (1st order)



Lin about $\bar{x}$

① $f(x) \approx f(\bar{x}) + \left(\frac{\partial f}{\partial x} \Big|_{\bar{x}} \right) (x - \bar{x})$

$x = \bar{x} + \Delta x$

② $f(\bar{x} + \Delta x) \approx f(\bar{x}) + \left(\frac{\partial f}{\partial x} \Big|_{\bar{x}} \right) \Delta x$