Acrobot : mechanics + optimization
Anybal : RL-based
manipulation : imitation-learning

Δ underactuated

$\begin{cases}\dot{x} = f(x,u) \\ y = g(x)\end{cases}$  $\begin{cases}\dot{x} = Ax+Bu \\ y = Cx+du\end{cases}$

Δ second-order nonlinear system

$\ddot{q} = f(q,\dot{q},u)$

or $\dot{x} = f(x,u) = \begin{bmatrix}\dot{q} \\ f(q,\dot{q},u)\end{bmatrix}$

Δ "control affine" nonlinear systems

$\ddot{q} = f_1(q,\dot{q}) + f_2(q,\dot{q})u$ (1)

based on this, define underactuated

def

(1) is fully actuated in $q, \dot{q}$ iff $f_2(q,\dot{q})$ is full row rank matrix

i.e. $\dim(q)=m$, $\dim(u)=n$, $f_2(q,\dot{q}) \in \mathbb{R}^{m\times n}$, $\mathrm{rank}[f_2(q,\dot{q})] = m$

controllable : can choose some inputs over time to achieve some goal
actuatable : can choose some inputs to cause acceleration

(1) is underactuated iff in $q, \dot{q}$ $\mathrm{rank}[f_2(q,\dot{q})] < m$

$\forall q, \dot{q}$, $\mathrm{rank}[f_2(q,\dot{q})] < m$
⇔ "system" is underactuated

Δ Feedback equivalence (fully-actuated)

given $\ddot{q} = f_1(q,\dot{q}) + f_2(q,\dot{q})u$

⇒ then $u = f_2^{-1}(q,\dot{q})[\ddot{q}^d - f_1(q,\dot{q})]$
⇒ $\ddot{q} = \ddot{q}^d$

give me an "acceleration", I give u an "u"

⇒ feedback equivalent to $\ddot{q} = u$ (double integrator which we hv optimal ctrl fn)

Δ feedback equivalence be broken: input saturations, state constraints, model uncertainty

Δ Manipulator Eqs

$M(q)\ddot{q} + C(q,\dot{q})\dot{q} = \tau_g(q) + Bu$ (*)

mass matrix, centrifugal coriolis terms, gravity, torque input

$M \succ 0$

⇒ $\ddot{q} = M^{-1}(q)[-C(q,\dot{q})\dot{q} + \tau_g(q) + Bu]$

$\ddot{q} = f_1(q,\dot{q}) + f_2(q,\dot{q})u$ is in this form!

Δ! whether $f_2(q,\dot{q})$ is full rank relies on "B"
as $\underbrace{M^{-1}(q)}_{\text{full rank}}\underbrace{B}_{\text{not sure!}}$

Δ Dynamic Programming

$ml\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u$

Δ Control as an optimization

• given trajectory $x(\cdot), u(\cdot)$  $\forall t\ x(t)\ t\in[0, T_{max}]$
• Assign a score (scalar reward cost)
e.g. time, any distance
• subject to constraints

e.g. minimum time for double integrator

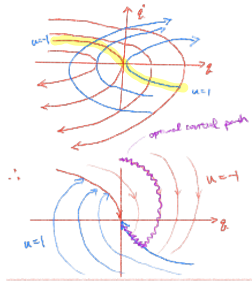
$\ddot{q} = u$

goal: drive to $q = \dot{q} = 0$ in minimum-time from any initial condition

Δ intuition: "bang-bang" policy (accelerate as much as possible) (unsmooth)

Δ intuition #2:

$\ddot{q} = u$  $u=-1$  $u=1$
$\dot{q}(t) = \dot{q}(0) - t$
$q(t) = q(0) + \dot{q}(0)t + \frac{1}{2}ut^2$




Q: How to generalize?

Δ Dynamic Programming

minimum-time ≅ shortest path problems

• weighted shortest path
• DP
discrete states $s_i \in S$
discrete actions $a_i \in A$
discrete time $s[n+1] = f(s[n], a[n])$
"edge cost" $g(s,a)$
total cost $\sum g(s,a)$
Key idea: Additive cost
e.g. time to goal $g(s,a) = \begin{cases}1 & \forall s \neq s_{goal} \\ 0 & \text{o.w.}\end{cases}$
$\int g(s,a)\,dt$

Δ DP

• DP is a recursive algorithm
• solve backwards from the goal
• cost-to-go

$J^*(s_i) = \min_{a[\cdot]} \sum_{n=0}^{\infty} g(s[n], a[n])$  — watch for convergence
$s[0] = s_i$ — hard to search

⇔ $J^*(s_i) = \min_a [g(s,a) + J^*(f(s,a))]$ — single
condition to certify optimality

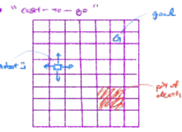
• Algorithm of DP (= value iteration)

$\hat{J}^* \leftarrow$ estimate of optimal cost-to-go

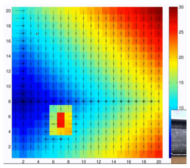
$\forall i\quad \hat{J}^*(s_i) \leftarrow \min_a [g(s,a) + \hat{J}^*(f(s,a))]$

$\hat{J}^* \longrightarrow J^* + c$

• "cost-to-go"




$g(s,a) = \begin{cases} 0 & @\text{ goal} \\ 10 & @\text{ pit} \\ 1 & @\text{ o.w.}\end{cases}$



• Caveats
- accuracy (discretization errors)
- scalability (dimensions ~5) (curse of the dimensionality)
- model is known
- cost function
- assume "full state" feedback

Δ $\lim_{\Delta x, \Delta u, \Delta t \to 0}$ DP

• my minimum-time double integrator
$\ddot{q} = u$, $|u| \le 1$
optimal policy is "bang-bang"
• we want "discrete" → "continuous"

Δ optimal cost-to-go

• discrete
$s[n+1] = f(s[n], a[n])$
• cost
$\sum \ell(s[n], a[n])$
$\forall s\quad J^*(s) = \min_a [\ell(s,a) + J^*(f(s,a))]$ — cost-to-go function
∴ $\pi^*(s) = \arg\min_a [\ell(s,a) + J^*(f(s,a))]$ (optimal controller/policy)

Δ discrete v continuous

| | | |
|---|---|---|
| states | s | x |
| actions | a | u |
| time | n | t |

Δ informal derivation

$x[n+1] \approx x[n] + h\, f_c(x[n], u[n])$
euler-integration  $t_{n+1} - t_n$
$\ell_d(x,u) = h\,\ell_c(x,u)$
∴ $J^*(x) = \min_u [h\,\ell_c(x,u) + J^*(x + hf_c(x,u))]$
$J^*(x) + \frac{\partial J^*}{\partial x} h f_c(x,u)$

⇒ $0 = \min_u \left[\ell_c(x,u) + \frac{\partial J^*}{\partial x} f_c(x,u)\right]$

Hamilton-Jacobi-Bellman Eqn.
HJB

e.g. infinite-time Grid world

$J^*(x) = c$

$\frac{dJ}{dt} = \frac{\partial J}{\partial x} f(x,u)$

for $u^*$  cost-to-go

$0 = \ell(x,u^*) + \frac{\partial J}{\partial x}\frac{\partial x}{\partial t} \Rightarrow \frac{\partial J}{\partial t} = -\ell(x,u^*)$ — running cost

⇒ if you are following optimal policy your cost ↓

HJB sufficiency theorem

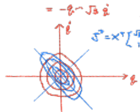
e.g. double integrator / Quadratic cost
$\ddot{q} = u$  $\ell(x,u) = q^2 + \dot{q}^2 + u^2$ (no constraints)

claim:
$\pi^*(x) = -q - \sqrt{3}\dot{q}$
$J^*(x) = \sqrt{3}q^2 + 2q\dot{q} + \sqrt{3}\dot{q}^2$  $\left(x = \begin{bmatrix} q \\ \dot{q}\end{bmatrix}\right)$

proof:

(1) $0 = \min_u \left[q^2 + \dot{q}^2 + u^2 + \frac{\partial J}{\partial x} f(x,u)\right]$
$\left(\frac{\partial J}{\partial x} f(x,u) = \left[\frac{\partial J}{\partial q}\ \frac{\partial J}{\partial \dot{q}}\right]\begin{bmatrix}\dot{q} \\ u\end{bmatrix}\right)$
$= \min_u \left[q^2 + \dot{q}^2 + u^2 + \frac{\partial J}{\partial q}\dot{q} + \frac{\partial J}{\partial \dot{q}}u\right]$
$= \min_u [0]$

(2) $u = \arg\min_u \left[u^2 + \frac{\partial J}{\partial \dot{q}}u\right]$
⇒ $\frac{\partial}{\partial u}(\cdot) = 2u + \frac{\partial J}{\partial \dot{q}} = 0$
⇒ $u = -\frac{1}{2}\frac{\partial J}{\partial \dot{q}}$
$= -\frac{1}{2}(2q + 2\sqrt{3}\dot{q})$
$= -q - \sqrt{3}\dot{q}$ ✱




$J^* = x^T\begin{bmatrix}\sqrt{3} & 1 \\ 1 & \sqrt{3}\end{bmatrix}x$

Δ LQR

• $\dot{x} = Ax + Bu$
$\ell(x,u) = x^TQx + u^TRu$
$Q \succeq 0$  $R \succ 0$

• inf horizon
$J^*(x) = x^TSx$ (cost-to-go) — can capture the dynamics
$S \succeq 0$

$\frac{\partial J}{\partial x} = 2x^TS$

$0 = \min_u \left[x^TQx + u^TRu + \frac{\partial J}{\partial x}(Ax+Bu)\right]$

⇒ $\frac{\partial [\,]}{\partial u} = 0$,  $u = -R^{-1}B^TSx$
$= -Kx$

Riccati Equation
$0 = Q - SBR^{-1}B^TS + A^TS + SA$

• insights:

$-\frac{\partial J}{\partial x} = -2x^TS$ (where we want our cost-to-go func. to decrease)

$-\frac{\partial J}{\partial x}$

$u = -R^{-1}B^TSx$
(gradient)
(underactuated B matrix to adjust Sx)
(weighting on $B^TSx$)

Δ DP so far

- Tabular (discrete → graph search)
- Linear dynamics + convex objective (LQR)

↳ here, use acrobot, cartpole, quadrotor to demonstrate DP

Δ Acrobot  Δ Cart pole:


Δ Equations of motion :

$M(q)\ddot{q} + C(q,\dot{q})\dot{q} = \tau_g(q) + Bu$

- Acrobot
$q = \begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix}$
$u = [\tau]$
$B = \begin{bmatrix}0 \\ 1\end{bmatrix}$
$|u| \le u_{max}$

- Cart pole
$q = \begin{bmatrix}x_{cart} \\ \theta_{pole}\end{bmatrix}$
$u = [f_{cart}]$
$B = \begin{bmatrix}1 \\ 0\end{bmatrix}$
$|u| \le u_{max}$
$|x| \le x_{max}$

↳ target: swing up + balance

soln
1. tabular value iteration ?
yes, not elegant
2. LQR ?
Beautiful solution for balancing (need more for swing-up)

Δ LQR for nonlinear systems

$\dot{x} = f(x,u) \Leftrightarrow \dot{x} = Ax + Bu$

⇒ Choose nominal $x_0, u_0$ (want a fixed pt)

⇒ $\dot{x} \approx f(x_0,u_0) + \underbrace{\frac{\partial f}{\partial x}\Big|_{x=x_0, u=u_0}}_{A}(x-x_0) + \underbrace{\frac{\partial f}{\partial u}\Big|_{x=x_0, u=u_0}}_{B}(u-u_0)$

let
$\bar{x} = x - x_0$
$\dot{\bar{x}} = \dot{x} - \dot{x}_0$
$\bar{u} = u - u_0$

⇒ $\dot{\bar{x}} = A\bar{x} + B\bar{u}$

e.g. simple pendulum

• $ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = \tau$
$x = \begin{bmatrix}\theta \\ \dot{\theta}\end{bmatrix}$  $u = [\tau]$

• linearize about $\theta = \pi$, $\dot{\theta} = 0$, $u = 0$

• $\dot{x} = \begin{bmatrix}\dot{\theta} \\ \frac{1}{ml^2}[\tau - b\dot{\theta} - mgl\sin\theta]\end{bmatrix}$

• check fixed pt ✓
$\dot{x}_0 = \begin{bmatrix}0 \\ \frac{1}{ml^2}[0 - b\cdot 0 - mgl\sin\pi]\end{bmatrix} = \begin{bmatrix}0 \\ 0\end{bmatrix}$

• linearization
$\frac{\partial f}{\partial x} = \begin{bmatrix}0 & 1 \\ -\frac{g}{l}\cos\theta & -\frac{b}{ml^2}\end{bmatrix}\Bigg|_{\theta=\pi} = \begin{bmatrix}0 & 1 \\ \frac{g}{l} & -\frac{b}{ml^2}\end{bmatrix}$

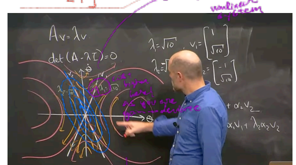
$\frac{\partial f}{\partial u} = \begin{bmatrix}0 \\ \frac{1}{ml^2}\end{bmatrix}$

let $m = l = 1$  $g = 10$

$A = \begin{bmatrix}0 & 1 \\ 10 & 0\end{bmatrix}$

$B = \begin{bmatrix}0 \\ 1\end{bmatrix}$

orange: approximate of a nonlinear system




red: nonlinear system

• LQR Design

$\begin{cases}\dot{\bar{x}} = A\bar{x} + B\bar{u} \\ J = \int_0^\infty \bar{x}^TQ\bar{x} + \bar{u}^TR\bar{u}\,dt \\ [K,S] = \mathrm{lqr}(A,B,Q,R) \\ u = -Kx = -R^{-1}B^TSx \\ J^* = \bar{x}^TS\bar{x}\end{cases}$

⇒ Stable

$\dot{x} = \underbrace{(A - BK)}_{\text{Hurwitz}}x$

Δ Is (A,B) controllable?

underactuated but controllable

Δ controllable: given $x(0)$
can I find $u(t)$ $\forall t \in [t_0, t_f]$
s.t. $x(t_f) = 0$

Is (A,B) stablizable

Δ stablizable: given $x(0)$
can I find $u(t)$ $\forall t \in [t_0, \infty)$
s.t. $\lim_{t\to\infty} x(t) = 0$
(no control input)

Δ Nonlinear Dynamics

(kinetic energy)
$T = \frac{1}{2}ml^2\dot{\theta}^2$  $v_{rm} = l\dot{\theta}$
(potential energy)
$U = -mgl\cos\theta$

Lagrangian mechanics

$L = K - U$
$K = \frac{1}{2}mv^2$
$= \frac{1}{2}m\dot{s}^2$
$= \frac{1}{2}ml^2\dot{\theta}^2$  $\theta = \frac{s}{l}$
$u = mgy$
$= -mgl\cos\theta$
∴ $L = \frac{1}{2}ml^2\dot{\theta}^2 + mgl\cos\theta$

Euler-Lagrange equation
$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} = \frac{\partial L}{\partial \theta}$

$L = \frac{1}{2}ml^2\dot{\theta}^2 + mgl\cos\theta$

$\frac{\partial L}{\partial \theta} = ml^2\ddot{\theta} + mgl\sin\theta = Q$ — generalized force

$Q = -b\dot{\theta} + u$ (friction, torque)

∴ $ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u$
⇕
$M(q)\ddot{q} + C(q,\dot{q})\dot{q} = \tau_g(q) + Bu$
⇕
$\ddot{q} = f(\theta, \dot{\theta}, u)$
⇕
$\theta(t) = ?$

Δ Nonlinear Dynamics Questions

• What is $\lim_{t\to\infty}\theta(t)$ ?

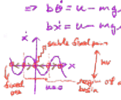
• Will my robot fall down ?

$ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u$  (N·m)
$kg\frac{m^2}{s}\frac{1}{s}$  $kg\,m^2\frac{1}{s^2}$  $kg\frac{m^2}{s^2}$

case: $b\dot{\theta} \gg ml^2$

$b\sqrt{\frac{l}{g}} \gg ml^2$ (heavily damped regime)

⇒ linear approximation!
⇒ $b\dot{\theta} = u - mgl\sin\theta$
$b\dot{x} = u - mgl\sin x$  $x \in \mathbb{R}$




Δ Stability

• local stability — do not leave a ball
- in the sense of Lyapunov (i.s.L.)
$\begin{cases}\epsilon, \delta: \text{small positive constants} \\ \forall\epsilon, \exists\delta \text{ s.t. } \|x(0) - x^*\| < \delta \\ \Rightarrow \forall t\ \|x(t) - x^*\| < \epsilon\end{cases}$
- locally attractive — get to the region
$\left\{\lim_{t\to\infty} x(t) \longrightarrow x^*\right.$
- asymptotically stable — i.s.L + attractive
{ isL + attractive
- exponentially stable — get to the region w/ rate
$\begin{cases}\forall t \\ \|x(t) - x^*\| < Ce^{-\alpha t} \\ C, \alpha > 0\end{cases}$

Δ Simple RNN

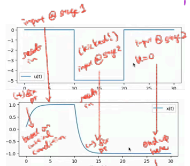
Autapse

$\tanh(\sum w_i u)$

±b affect

$\dot{x} = -x + \tanh(wx + b)$

$\dot{x} = -x + \tanh(wx)$




Δ 2nd-order system

$ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u$

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

$x = \begin{bmatrix}\theta \\ \dot{\theta}\end{bmatrix}$

$\dot{x} = \begin{bmatrix}\dot{\theta} \\ \frac{1}{ml^2}[u - b\dot{\theta} - mgl\sin\theta]\end{bmatrix}$

$b = 0$
$u = 0$

homoclinic
damped ones

✱ Essence of controls for underactuated robotics:

we hv all these dynamics
→ plug w/ u, s.t. phase portraits behave as we want!!!

## △ Lyapunov Analysis

recall DP:
- Tabular — easy to explode
- LQR — only for linear case
- Approximate DP (NN) — works quite well (but solve a different DP, e.g. discounting)

→ all are trying to get "cost-to-go" function $J^*(x)$

△ now Lyapunov (easy to find) ⟷ optimal value (hard to find)
good enough — very good

might replace the original optimal val.

△ **Example**: stability analysis of simple pendulum

$ml\ddot{\theta} + mgl\sin\theta = -b\dot{\theta}$

△ $\theta(t)$ ⟶ hard to solve, cannot be analysis

△ Lyapunov instead !!!

△ $E = K + U$

$= \frac{1}{2}ml^2\dot{\theta}^2 - mgl\cos\theta$

△ $\frac{d}{dt}E(x)$ , $x = \begin{bmatrix}\theta \\ \dot{\theta}\end{bmatrix}$

$= \frac{\partial E}{\partial x}\frac{\partial x}{\partial t}$

$= \frac{\partial E}{\partial \theta}\dot{\theta} + \frac{\partial E}{\partial \dot{\theta}}\ddot{\theta}$

$= mgl\sin\theta\,\dot{\theta} + ml^2\dot{\theta}\ddot{\theta}$

$= \dot{\theta}(mgl\sin\theta + ml^2\ddot{\theta})$

$= -b\dot{\theta}^2 \le 0$ if $b > 0$

### △ General Energy function

△ given $\dot{x} = f(x)$ no $u$

→ want to prove stability of $x^* = 0$

→ construct a differentiable function $V(x)$, s.t.

$$\begin{cases} V(0)=0, & V(x)>0,\ x\neq 0 & \text{PD} \\ \dot{V}(0)=0, & \dot{V}(x)\le 0,\ x\neq 0 & \text{NSD}\end{cases}$$

sufficient condition

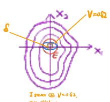
→ then, $x^*$ is stable i.s.L.

△ i.s.L def

$\forall \epsilon > 0$, $\exists \delta > 0$

s.t. $\|x(0) - x^*\| < \delta$

$\forall t\ \|x(t) - x^*\| < \epsilon$




e.g. pendulum

$V(x) = E + mgl$

### △ Asymptotically Stable

$$\begin{cases} V(0)=0, & V(x)>0,\ x\neq 0 & \text{PD} \\ \dot{V}(0)=0, & \dot{V}(x)<0,\ x\neq 0 & \text{NSD}\end{cases}$$

"<" not "≤"

otherwise $\dot{V}(x) = 0$, → limit cycles.

### △ Global Stability

Global Asymptotically stability (GAS)

$$\begin{cases} V(0)=0, & V(x)>0,\ x\neq 0 & \text{PD} \\ \dot{V}(0)=0, & \dot{V}(x)\le 0,\ x\neq 0 & \text{NSD}\end{cases}$$

+

$\lim_{|x|\to\infty} V(x) = \infty$ "radially unbounded"

### △ Regional Stability

$$\begin{cases} V(0)=0, & V(x)>0,\ x\neq 0 & \text{PD} \\ \dot{V}(0)=0, & \dot{V}(x)\le 0,\ x\neq 0 & \text{NSD}\end{cases}$$

$\forall x \in D \subset \mathbb{R}^n$

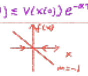
### △ Exponential Stability

$$\begin{cases} V(0)=0, & V(x)>0,\ x\neq 0 & \text{PD} \\ \dot{V}(0)=0, & \dot{V}(x)\le -\alpha V(x),\ x\neq 0 & \text{NSD}\end{cases}$$

⇕

$V(x(t)) \le V(x(0))e^{-\alpha t}$

e.g. $\dot{x} = -x$



$V(x) = x^2$

$\dot{V}(x) = \frac{\partial V}{\partial x}\dot{x}$

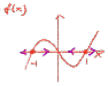
$= 2x(-x)$

$= -2x^2 < 0$ — { i.S.L ✓, asymptotic ✓

$\le 2V(x)$ — { exponential ✓

$\lim_{|x|\to\infty} V(x) = \infty$ — { global stable

e.g. $\dot{x} = -x + x^3 = f(x)$



$x^* = 0$ a f.p.

$x \in (-1, 1)$ R.O.A.

$V(x) = x^2$

$\dot{V}(x) = 2x(-x + x^3)$

$= 2x^2(x^2 - 1)$

$= \begin{cases} 0 & x = 0, +1, -1 \\ >0 & |x| > 1 \\ <0 & |x| < 1,\ x \neq 0\end{cases}$

• sublevel set of $V$ ⟶ invariant set

$V(x) \le \rho$ (inside the R.O.A. of $x^*$)

### △ General form of R.O.A.

if $V(x) > 0$, $\dot{V}(x) < 0$

$\forall x \in \{x \mid V(x) < \rho,\ \rho > 0\}$ — invariant set

then $V(x(0)) < \rho$

⇒ $\lim_{t\to\infty} V \to 0$ , $x \to 0$

and $\{x \mid V(x) < \rho\}$ is inside R.O.A.

### △ LaSalle's Theorem

### △ Lyapunov ⟶ Cost-to-go funcs.

HJB:

$0 = \min_u \left[\ell(x,u) + \frac{\partial J^*}{\partial x} f(x,u)\right]$

$u^* = \pi^*(x)$

⇒ $0 = \ell(x, u^*) + \frac{\partial J^*}{\partial x} f(x, u^*)$

$= \ell(x, u^*) + \frac{\partial J^*}{\partial t}$

⇒ $\dot{J}^*(x) = -\ell(x, u^*)$

"cost-to-go" function's derivative has to be decreasing !!!

⇕ relaxation

$\dot{V}(x) < 0$ ⟶ way more easy !!!

HJB discrete

$J^*(x) = \min_u \left[\ell(x,u) + J^*(f(x,u))\right]$

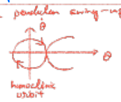
$\Delta J^*(x) = J^*(f(x, u^*)) - J^*(x)$

⇕ $= -\ell(x, u^*)$

$\Delta V(x) \le 0$

### △ Lyapunov-based controller

e.g. pendulum swing-up




• $E^d = mgl$

$V(x) = \frac{1}{2}(E(x) - E^d(x))^2$

$= \frac{1}{2}\tilde{E}^2(x)$

• $ml^2\ddot{\theta} + mgl\sin\theta = u$

$\dot{E} = u\dot{\theta}$

$\dot{\tilde{E}} = \dot{E} - \dot{E}^d = u\dot{\theta}$

• let $u = -k\dot{\theta}\tilde{E}$ , $k > 0$

making

$\dot{\tilde{E}} = -k\dot{\theta}^2\tilde{E}$

### △ Promise: find V(x) should be easier

- Global Lyapunov analysis for the pendulum
- input: pendulum dynamics, parametric family of trig/poly for Lyapunov — polynomial over $1, \cos\theta, \sin\theta, \dot{\theta}$

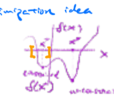
output: coefficient of the polynomials + certificate that Lyapunov conditions satisfied $\forall x$ ?

→ Lyapunov analysis w/ convex optimization

### △ Some basic optimization idea

$\min_x f(x)$

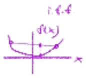
s.t. $g_i(x) \le 0$




• convex optimization

i.f.f. $f(x)$ is convex function

$\{g_i(x)\}$ is a convex set



LP → QP → SOCP ⟶ SDP

easy to solve ⟶ hard to solve

### △ SDP

linear objective, linear constraints

PSD matrix constraints.

### △ Q! How do we compute Lyapunov func?

1. Parameterize Lyapunov candidate

let's say

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

$V(x) = \sum_i \alpha_i \phi_i(x) = \alpha^T\phi(x)$ — nonlinear basis

2. Use convex optimization search over $\alpha$ to satisfy Lyapunov condition.

find $\alpha$: $V(0) = 0$, $\forall i$ $V(x_i) > \epsilon x_i^T x_i$

⇒ $V(x_i) > \epsilon x^T x$ linear constraint in $\alpha$

$\dot{V}(0) = 0$, $\forall i$ $\dot{V}(x_i) = \left.\frac{\partial V}{\partial x}\right|_{x = x_i} f(x_i)$

$= \alpha^T \frac{\partial \phi}{\partial x} f(x_i)$

$< -\epsilon x_i^T x$ — linear constraint

How do we make sure for all x?

⇕

$\min_\alpha \|\dot{V}(x) + 1\|_1$

### ▲ Robust stability analysis

$\dot{x} = Ax$, $A \in [\bar{A}, A^+]$ uncertain

⇒ $A = \sum \beta_i A_i$ $\beta_i \ge 0$, $\sum\beta_i = 1$

⇕

find $P \succ 0$ s.t. $\forall i$ $PA_i + A_i^T P < 0$

⇒ implies $P(\sum\beta_i A_i) + (\sum\beta_i A_i)^T P < 0$

↑

### ▲ $\dot{x} = Ax$ is it stable?

$$\begin{cases} V(x) = x^T P x, & P \succ 0 \\ \dot{V}(x) = 2x^T PAx \\ \quad = x^T PAx + xA^T Px \end{cases}$$

Lyapunov fun linear system

$\dot{V}(x) < 0$ if $(PA + A^T P) \prec 0$

↕ pick $Q \succ 0$ & $PA + A^T P = -Q$ (Lyapunov eq.)

find $\alpha$ s.t. $P = [\dots]$

⇕ $P \succ 0$, $PA + A^T P \preceq 0$

find $\alpha, \beta$ $P \succ 0$ $P_2 \succ 0$

s.t. $PA + A^T P = -P_2$

↑

$V(x) = \sum \alpha_i \phi(x)$ , $\alpha_i \ge 0$

★ $V(x) = \phi^T(x) G \phi(x)$ , $G \succ 0$

$\dot{V}(x) = \frac{\partial V}{\partial x} f(x)$

→ construct $-\phi^T(x) G_2 \phi(x)$

→ set $\dot{V}(x) = -\phi^T G_2 \phi$

### △ DP sum

$\frac{d}{dt}x = f(x(t), u(t), t)\,dt$

• $J(x(t), u(t), t_0, t_f)$

$= Q(x(t_f), t_f) + \int_{t_0}^{t_f} \ell(x(\tau), u(\tau))\,d\tau$

• $V(x(t_0), t_0, t_f)$ (cost-to-go)

$= \min_{u(t)} J(x(t), u(t), t_0, t_f)$

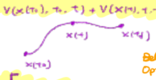
• $-\frac{\partial V}{\partial t} = \min_{u(t)} \left(\left(\frac{\partial V}{\partial x}\right)^T f(x(t), u(t)) + \ell(x(t), u(t))\right)$

⇔ $\min_{u(t)} \left(\left(\frac{\partial J^*}{\partial x}\right)^T f(x(t), u(t)) + \ell(x(t), u(t))\right)$

Hamilton-Jacobi-Bellman (HJB)

• $V(x(t_0), t_0, t_f)$

$= V(x(t_0), t_0, t) + V(x(t), t, t_f)$




Bellman Optimality

⇔ $J^*(s_i) = \min_a [\ell(s,a) + J^*(f(s,a))]$

proof

total derivative

$\frac{d}{dt}V(x(t), t, t_f) = \frac{\partial V}{\partial t} + \left(\frac{\partial V}{\partial x}\right)^T \frac{\partial x}{\partial t}$

$= \frac{d}{dt}\min_{u(t)} J(x(t), u(t), t, t_f)$

$= \frac{d}{dt}\min_{u(t)} Q(x(t_f), t_f) + \int_t^{t_f} \ell(x(\tau), u(\tau))\,d\tau$

$= \min_{u(t)} \frac{d}{dt}\left[Q(x(t_f), t_f) + \int_t^{t_f} \ell(x(\tau), u(\tau))\,d\tau\right]$

$= \min_{u(t)}\ 0 + \frac{d}{dt}\left(\left[\mathcal{L}(x(\tau), u(\tau))\right]_t^{t_f}\right)$

$= \min_{u(t)} \frac{d}{dt}\left(\mathcal{L}(x(t_f), u(t_f)) - \mathcal{L}(x(t), u(t))\right)$

$= \min_{u(t)} -\ell(x(t), u(t))$

⇕

$\frac{\partial V}{\partial t} + \left(\frac{\partial V}{\partial x}\right)^T \frac{\partial x}{\partial t} = \min_{u(t)} -\ell(x(t), u(t))$

⇕

$-\frac{\partial V}{\partial t} = \min_{u(t)} \left(\left(\frac{\partial V}{\partial x}\right)^T \frac{\partial x}{\partial t} + \ell(x,u)\right)$

$= \min_{u(t)} \left(\frac{\partial V}{\partial x}^T f(x,u) + \ell(x,u)\right)$

★ Get $V$ ⟶ Get $u$
- two point bounded value problem
- shooting method

### △ Discrete-Time HJB

• $x_{k+1} = F(x_k, u_k)$

• $J(x_0, \{u_k\}_{k=0}^{n}, n)$

$= Q(x_n, t_n) + \sum_{k=0}^{n} \ell(x_k, u_k)$

• $V(x_0, n)$

$= \min_{\{u_k\}_{k=0}^{n}} J(x_0, \{u_k\}_{k=0}^{n}, n)$

• $V(x_0, n)$

$= V(x_0, k) + V(x_k, n-k)\ \forall k \in [0, n]$

• $V(x_k, n)$

$= \min_{u_k}\left[\ell(x_k, u_k) + V(x_{k+1}, n-1)\right]$, $x_{k+1} = F(x_k, u_k)$

⇔ vs $J^*(s) = \min_a [\ell(s,a) + J^*(f(s,a))]$

Discrete HJB

⇕

$V(x_k, n)$

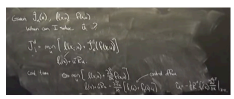
$= \min_{u_k}\left(\ell(x_k, u_k) + V(F(x_k, u_k), n-1)\right)$

⇕

$V(x) = \min_u \left(\ell(x,u) + V(F(x,u))\right)$

↓

$\pi(x) = \arg\min_u \left(\ell(x,u) + V(F(x,u))\right)$




- use optimal $u$: recall CT LQR: $u^* = -K^*x = -R^{-1}B^T S^* x$; DT LQR: $u^* = -K^* x = -(R + B^T S^* B)^{-1} B^T S^* A x$
- instead of sampling $u_i$, only sample $x_i \in X_s \subset X$

⇒ $J_i^d = x_i^T Q x_i + \hat{u}_i^T R \hat{u}_i + \hat{J}_\alpha(Ax_i + Bu_i)$

w/ optimal $u$ w/ current $\hat{S}$

### △ DP (cont'd)

key idea
1) LQR (linear optimal control)
2) local linearization
3) $Q = \begin{bmatrix} 10 & & & \\ & 10 & & \\ & & 1 & \\ & & & 1\end{bmatrix}$ $R = [1]$

@ balancing

Q: How do we swing up?

"value iteration"

⇒ Approximate DP

### △ Value iteration on a mesh for Pendulum Swing-up

△ mesh $q$ 51 bins — odd for "0"

$\dot{q}$ 51 bins

$u$ 9 bins

△ Acrobot

$51^4$ bins

9 bins — explode !!!

### △ NN! Approximating Value Functions

$\hat{J}_\alpha(x)$ — parameters + bias

- value iteration update

• recall:

$\forall s_i \in S$

$\hat{J}(s_i) = \min_a \left[\ell(s_i, a) + \hat{J}(f(s_i, a))\right]$

$s[n+1] = f(s[n], a[n])$

$\sum_{n=0}^{\infty} \ell(s[n], a[n])$

• Draw samples

$x_i \in X_s \subset X$

$u_j \in U_s \subset U$

$\forall ij$ $\dot{x}_{ij} = f(x_i, u_j)$ next state samples

$\ell_{ij} = \ell(x_i, u_j)$

desired $J$ @ $i$ — $J_i^d = \min_j [\ell_{ij} + \hat{J}_\alpha(x'_{ij})]$

⇒ $\min_\alpha$ loss $\sum_i \left[\hat{J}_\alpha(x_i) - J_i^d\right]^2$

⇕

$\min_\alpha$ loss $\sum_i \left[\hat{J}_\alpha(x_i) - \min_j\left[\ell_{ij} + \hat{J}_\beta(x'_{ij})\right]\right]^2$ "target network"

"here we do the training based on the samples. To hv a better sampling, RL kicks in, & guide the state space, to a subset, that is closer to the optima."

Q: when does it work? does it converge

↓ △ linear approximator

△ works @ special case: linear NN

• $\hat{J}_\alpha(x) = \alpha^T\phi(x)$ — nonlinear basis "input features" (fixed)

$= \sum_j \alpha_j \phi_j(x)$

⇒ $\min_\alpha \sum_i [\alpha^T\phi(x) - J_i^d]^2$

linear least squares has closed-form sol^n

⇒ SGD is guaranteed to converge to global minima

### △ Back to LQR; connection to approximator; w/ discrete LQR

$J^*(x) = x^T \underset{NN}{S^*} x$ ⟶ linear approximator

loss $\sum_i [x_i^T \hat{S} x_i - J_i^d]^2$

$\hat{S} \leftarrow \hat{S} - \eta \frac{\partial loss}{\partial S}$

Based on this, loss explodes ↑

why? LQR tries to solve inf horizon, cost cannot go to inf.

How does normal LQR avoid inf?

$x \to 0$, $u \to 0$ (exact 0)

How about here, when we do sampling? popularly seen in RL

- discounting $\sum_{n=0}^{\infty} \gamma^n \ell(x[n], u[n])$ $0 \le \gamma \le 1$

optimal: LQR$(\sqrt{\gamma}A, B, Q, \frac{1}{\gamma}R)$ — pretend that is more stable

$x[n+1] = Ax[n] + Bu[n]$

$|eig(A)| \le 1$.

$\llcorner \|FA\| \le |A|$

## △ Let's continue on Lyapunov

Recall:

$V_\alpha(x) > 0$

$\dot V_\alpha(x) < 0$

- Linear neural network
  - $\forall x_i \longrightarrow$ Linear Program
- Questions: from samples
  - $\forall x_i \longrightarrow \forall x \in \mathbb{R}^n$

e.g. $\dot x = Ax$

$$\begin{cases} V(x) = x^T P x & V(x) > 0 \quad P \succ 0 \\ \dot V(x) = x^T P A x + x^T A^T P x < 0 & (PA + A^T P) \prec 0 \end{cases}$$

SDP:

find $P \succeq 0$ $P_2 \succeq 0$ — Linear Matrix Inequality

$PA + A^T P = -P_2$ (linear constraints)

## △ Sums of squares

- given a function $f(x)$, is $f(x) \ge 0$ $\forall x$?

let $f(x) = \phi^T(x) P \phi(x)$ $P \succeq 0$

$= \sum \alpha_i \phi_i^2(x)$ $\alpha_i \ge 0$

sums of squares (SOS) decomposition of the problem

then we can know $f(x) \ge 0$ $\forall x$

- special case Polynomials

$P_\alpha(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots$

$= \phi^T(x) P \phi(x)$ $P \succ 0$

basis function

e.g. $P_\alpha(x) = \alpha^T \begin{bmatrix} 1 \\ x^2 \end{bmatrix}$ ← gets

$\therefore \phi(x) = \begin{bmatrix} 1 \\ x \end{bmatrix}$ → expand $\phi^T P \phi$

simplest basis

$m(x) \equiv$ monomial basis

eg $1, x, x^2, x^3, \dots$

$1, x, y, xy, x^2, y^2$

e.g. $2 - 4x + 5x^2$

$= \begin{bmatrix} 1 & x \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix} \begin{bmatrix} 1 \\ x \end{bmatrix}$

$\Updownarrow$

$$\begin{cases} \text{find } P \succeq 0 \\ \text{s.t. } P_{11} = 2 \quad 2P_{12} = -4 \quad P_{22} = 5 \end{cases}$$

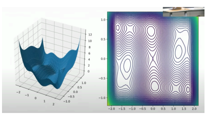
SDP again!

- linear objective
- linear constraints
- PSD constraints

SOS $P_\alpha(x)$ is SOS → formulate SDP & solve $\forall x \in \mathbb{R}^n$ $P_\alpha(x) \ge 0$

## △ Six-hump camel

$p(x, y) = 4x^2 + xy - 4y^2 - 2.1x^4 + 4y^4 + \frac{x^6}{6}$



SOS kick in!

$\min_\gamma -\gamma$ s.t. $p(x, y) - \gamma$ is SOS

$\gamma$ & $P_2$

?

$f(x) = NN^T(x) NN(x)$

## △ Back to Lyapunov

- Choose $V(x)$ to be polynomial
- assume $\dot x = f(x)$ restricted to polynomial
  - ⇒ $\dot V(x) = \frac{\partial V}{\partial x} f(x)$ is still polynomial

$$\begin{cases} V(x) > 0 \longrightarrow V(x) \text{ is SOS} \\ \dot V(x) < 0 \longrightarrow -\dot V(x) \text{ is SOS} \end{cases}$$

New criterions!

e.g. $\dot x_1 = -x_1 - 2x_2^2$

$\dot x_2 = -x_2 - x_1 x_2 - 2x_2^3$

$V(x) = x_1^2 + 2x_2^2$

$-\dot V(x)$ is S.O.S? $\forall x$

→ $-\dot V(x) = m^T(x) P_x m(x)$ — decision variable

inversely: find Lyapunov function

$V(x) = (S_{2,2} + 2x_0x_1S_{1,0} + 2x_0S_{2,1} + x_0^2S_{1,1} + 2x_1S_{2,0} + x_1^2S_{0,0})$

Solution:

$V(x) = (1.000000x_0^2 + 2.000057x_1^2)$

## △ R.O.A again

so basically:

- oracle answers T/F $\forall x \in \mathbb{R}^n$ ↓ $P(x) \ge 0$
- New Q: $\forall x \in D \subseteq \mathbb{R}^n$ is $p(x) \ge 0$

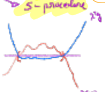
How:

define $D = \{x \mid g(x) \le 0\}$ — polynomial

$$\begin{cases} \text{find}_\alpha \text{ s.t. } p(x) + \lambda_\alpha^T(x) g(x) \text{ is SOS} \\ \lambda_\alpha(x) \text{ is SOS} \end{cases}$$

adding penalty / lagrange — multiplier

→ S-procedure



- if $p(x) - \lambda g$ is SOS, $p(x)$ is defo SOS
- if $g(x) \ge 0$ outside the region, $p(x) - \lambda g$ ↑ positive

$g(x) \le 0 \Rightarrow p(x) \ge 0$

## △ Trajectory Optimization

SO FAR:

- Dynamic Programming / optimal control
  - tabular (mesh-based)
  - LQR
  - NN (function approximator) (need sampling distributions)
- Lyapunov (simpler DP)
  - sums of squares opt. (10~20 dim) (more w/ SOS approximation)

for higher dimension?

$\forall x \longrightarrow$ NO

just

$x[0] = x_0$ (1 initial condition)

→ motivation

## △ Traj opt w/ linear discrete

Choose finite horizon $N$

$$\min_{x[\cdot], u[\cdot]} \sum_{n=0}^{N-1} l(x[n], u[n]) + \underbrace{l_f(x[N])}_{\text{terminal cost}}$$

$0 \sim N$, $0 \sim N-1$

s.t.

$$\forall n \in [0, N-1] \begin{cases} x[n+1] = Ax[n] + Bu[n] \\ x[0] = x_0 \\ -1 \le u[n] \le 1 \end{cases}$$

→ QP (L-MPC)

Choose

$l(x, u) = x^T Q x + u^T R u$, $Q \succeq 0$, $R \succ 0$

but why? this is LQR!

- LQR v QP
  - inf horizon — finite horizon
  - no constraints — can add constraints!
- horizon $N \longrightarrow$ limiting? affecting sol?

idea: choose $l_f(x) = x^T S x$ — LQR

$\underbrace{\sum_{n=0}^{N-1} l(x, u)}_{\text{constraints imposed on here}} + \underbrace{l_f(x)}_{N \sim \inf}$

- can restrict your state!

$\min_{x[\cdot], u[\cdot]} \sum_{n=0}^{N-1} |u[n]|^2$

s.t. $x[0] = x_0$ $x[N] = x_f$

$x[n+1] = Ax[n] + Bu[n]$ $\forall n$

$|u[n]| \le 1$

- min time?

$\ddot q = u$, $|u| \le 1$

set $N$ big → inv feasible.

then do $N-1$ recursively

## △ Categories

- Direct transcription
  - decision variables $x[\cdot]$, $u[\cdot]$
  - constraints $x[n+1] = Ax[n] + Bu[n]$
- Direct "shooting"
  - decision variables $x[0]$, $u[\cdot]$
  - constraints (linear) $x[n] = A^n x_0 + \sum A^{n-1} B u[n]$ — linear in $u[k]$

## △ MPC

- estimate current state $\hat x$
- solve linear traj opt w/ $x[0] = \hat x$
- execute $u[0]$
- repeat — "Recursive Feasibility"

★ when ever we are using it in field, we are actually tracking (w/ predefined traj) $x_f \longrightarrow 0$, where we are optimizing $u$ w.r.t. error state.

## △ Nonlinear Traj. Opt.

$x[n+1] = f(x[n], u[n])$ (possibly non-convex)

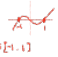
⇒ transcript / direct } could work: NOT CONVEX

SQP (second-order method): 1. take big steps (convergence rate) 2. native to handle constraints

## △ R.O.A. using SOS + S-procedure

e.g. $\dot x = -x + x^3$

$V(x) = x^2$



we know R.O.A is $\in [-1, 1]$

$\therefore V(x) \le 1$

⇒ find$_\alpha$ s.t. $-\dot V(x) + \lambda_\alpha(x)(1 - V(x))$ is SOS & $\lambda_\alpha(x)$ is SOS

↑

$D = \{x \mid g(x) = 0\}$

find$_\alpha$ $p(x) + \lambda_\alpha^T(x) g(x)$ is SOS

$\lambda_\alpha(x)$ can be arbitrary.