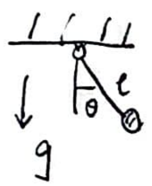
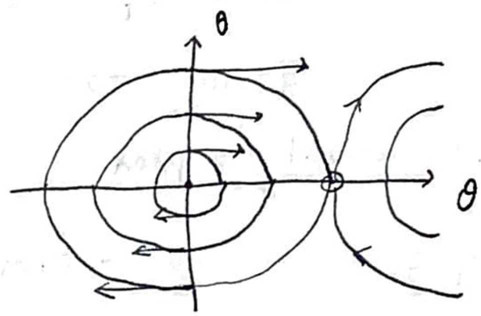
Today: Dynamic Programming

- Control as Optimization
- Ex: Double Integrator
- Dynamic Programming algorithm
- Numerical optimal control of the pendulum

Last time:





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

origin: fixed point in the sense of Lyapunov

Feedback Linearization

$$u = 2mgl\sin\theta$$

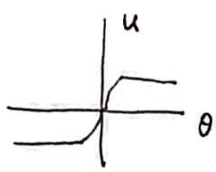
$$\Rightarrow ml^2\ddot\theta + b\dot\theta - mgl\sin\theta = 0$$

inverted gravity!

Q: Stable the unstable fixed pt

if $|u| \leq \frac{mgl}{4}$

$$u = \text{sat}_{\frac{mgl}{4}}(2mgl\sin\theta) = \begin{cases} mgl\sin\theta \\ \frac{mgl}{4} \\ -\frac{mgl}{4} \end{cases}$$



Big idea: Formulate control design as an optimization

12 Given a trajectory $x(\cdot)$ $u(\cdot)$

$x(\cdot)$ $\forall t \in [0, +\infty)$ $x(t)$

assign a score (scalar)

Potentially also set constraints

$\forall t \quad |u(t)| \leq 1 \qquad x(t_f) = \cancel{x_{goal}}\ x_{goal}$

Goal find "policy" $u = \pi(t, x)$

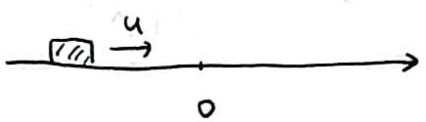
which optimizes (minimizes) the cost

ideally for all $t, x(0)$

Numerical approx.

Example: Minimum-time Problem for the double integrator

$\ddot{q} = u \qquad |u| \leq 1$



get to $\dot{q} = q = 0$ in min time

Optimal Control is "Bang-bang"

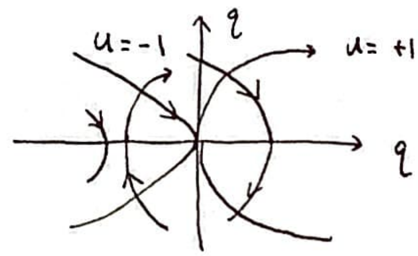
Accelerate (full throttle) then slam on the breaks

$\ddot{q} = u \qquad u = -1$

$\cancel{\ddot{q}}$

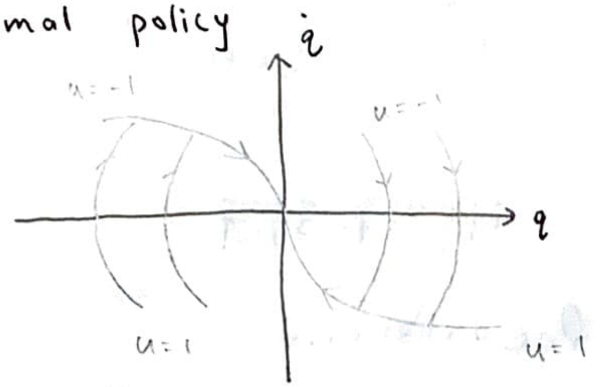
$\dot{q}(t) = q(0) - t$

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

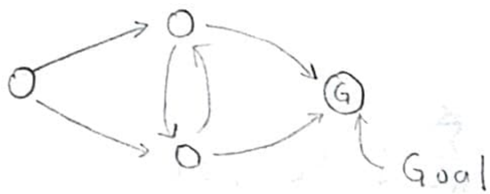


two set of curves

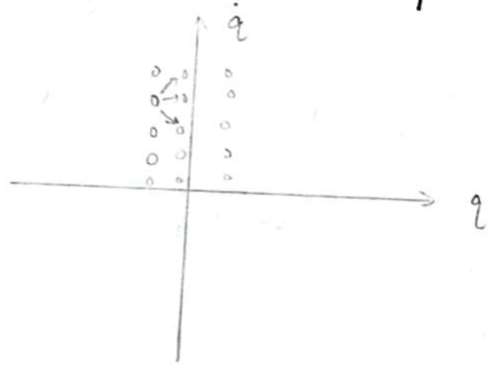
Optimal policy



Min time is closely related to shortest path problems



Shortest path via Dynamic Programming



discrete states : $s_i \in S$

" actions : $a_i \in A$

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

Cost function :

one-step cost $g(s, a)$

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

14 Key idea: Additive cost. $\int_0^\infty dt\, g(x(t), u(t))$

min-time

$$g(s, a) = \begin{cases} 1 & \text{if } s \neq s_{goal} \\ 0 & \text{otherwise} \end{cases}$$

quadratic cost.

$$g(x, u) = x^T x + u^T u.$$

may choose smooth function

Recursive form.

$$\forall i \quad J^*(s_i) = \min_{a[\cdot]} \sum_{n=0}^{\infty} g(s[n], a[n])$$

optimal!

~~cost~~ ~~to~~

s.t. $s[0] = s_i$

cost-to-go

$$= \min_a \Big[\underbrace{g(s_i, a)}_{\text{one-step}} + J^*(f(s_i, a)) \Big]$$

single time step.

one-step + cost-to-go from the next state

$$\pi^*(s_i) = \arg\min_a \left[g(s_i, a) + J^*(f(s_i, a)) \right]$$

$\hat{J}^*(s_i)$ ← "guess" initialize as random

$$\forall i \quad \hat{J}^*(s_i) \Leftarrow \min_a \left(g(s_i, a) + \hat{J}^*(f(s_i, a)) \right)$$

"value iteration" or "dynammic programming" (infinite horizon)

☆ Guaranteed to converge to J^* (up to a constant)

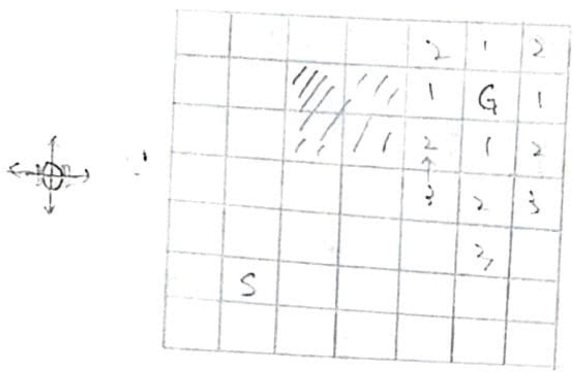
different by a constant

Optimization can be done in batch

not point wise update, update entire states in one iteration ?

Even asynchronous → distributed Version

Grid world



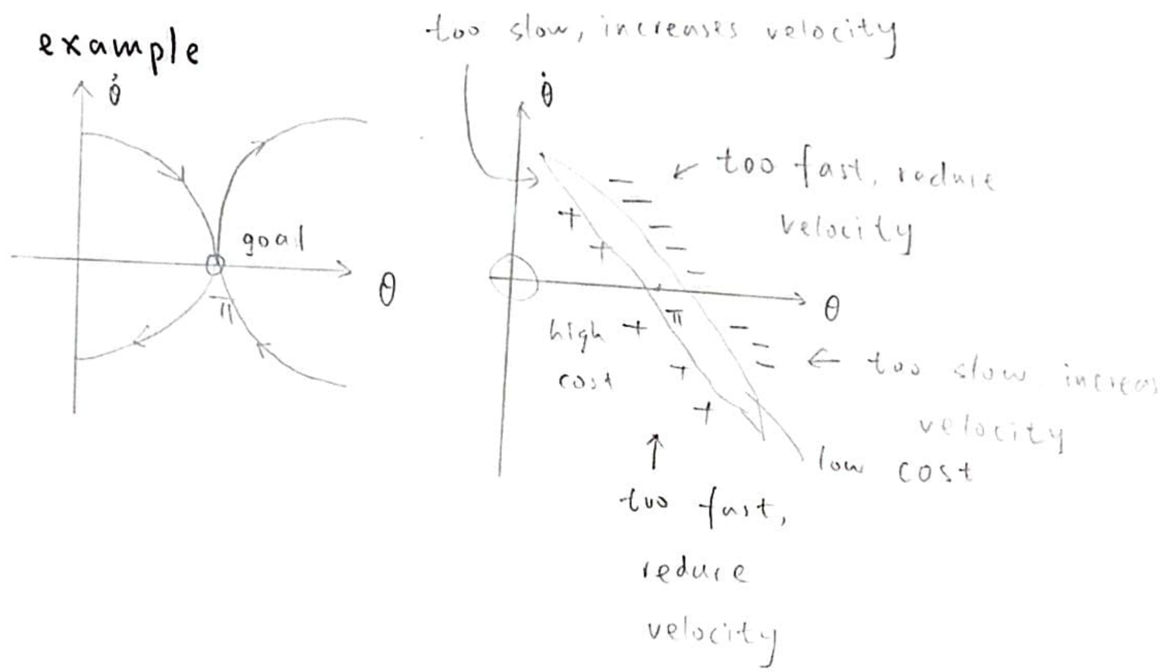
Optimal cost is unique

optimal policy is not unique

☆ J^* is unique

☆ π^* is NOT unique

Pendulum example



Limitations:

- Accuracy for continuous systems (discretization error)
- Scale (curse of dimensionality)

16 Deeper

- Assumes full state information