CS 287: Advanced Robotics
Fall 2009

Lecture 4: Control 3: Optimal control---discretization (function approximation)

Pieter Abbeel

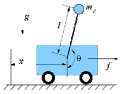
UC Berkeley EECS

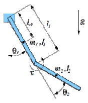
Announcement

- Tuesday Sept 15: **no** lecture

Today and forthcoming lectures

- Optimal control: provides general computational approach to tackle control problems---both under- and fully actuated.
  - Dynamic programming
    - Discretization
  - Dynamic programming for linear systems
    - Extensions to nonlinear settings:
      - Local linearization
      - Differential dynamic programming
      - Feedback linearization
  - Model predictive control (MPC)
  - Examples:





Today and Thursday

- Optimal control formalism [Tedrake, Ch. 6, Sutton and Barto Ch.1-4]
- Discrete Markov decision processes (MDPs)
  - Solution through value iteration [Tedrake Ch.6, Sutton and Barto Ch.1-4]
- Solution methods for continuous problems:
  - HJB equation [[[Tedrake, Ch. 7 (optional)]]]
  - Markov chain approximation method [Chow and Tsitsiklis, 1991; Munos and Moore, 2001] [[[Kushner and Dupuis 2001 (optional)]]]
- Continuous → discrete [Chow and Tsitsiklis, 1991; Munos and Moore, 2001] [[[Kushner and Dupuis 2001 (optional)]]]
- Error bounds:
  - Value function: Chow and Tsitsiklis; Kushner and Dupuis; function approximation [Gordon 1995; Tsitsiklis and Van Roy, 1996]
  - Value function close to optimal → resulting policy good
- Speed-ups and Accuracy/Performance improvements

Optimal control formulation

Given:

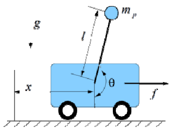
$$\text{dynamics} : \dot{x}(t) = f(x(t), u(t), t)$$

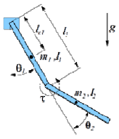
$$\text{cost function} : g(x, u, t)$$

Task: find a policy $u(t) = \pi(x, t)$ which optimizes:

$$J^\pi(x_0) = h(x(T)) + \int_0^T g(x(t), u(t), t) dt$$

Applicability: g and f often easier to specify than π





Finite horizon discrete time

- Markov decision process (MDP) (S, A, P, H, g)
  - S: set of states
  - A: set of actions
  - P: dynamics model $P(x_{t+1} = x' | x_t = x, u_t = u)$
  - H: horizon
  - g: S x A → R cost function
- Policy $\pi = (\mu_0, \mu_1, \ldots, \mu_H)$, $\mu_k : S \rightarrow A$
- Cost-to-go of a policy π: $J^\pi(x) = \mathrm{E}[\sum_{t=0}^H g(x(t), u(t)) | x_0 = x, \pi]$
- Goal: find $\pi^* \in \arg\min_{\pi \in \Pi} J^\pi$

Dynamic programming (aka value iteration)

Let $J^*_k = \min_{\mu_k,\ldots,\mu_H} \mathrm{E}[\sum_{t=k}^{H} g(x_t, u_t)]$, then we have:

$$\begin{aligned}
J^*_H(x) &= \min_u g(x(H), u(H)) \\
J^*_{H-1}(x) &= \min_u g(x,u) + \sum_{x'} P(x'|x,u) J^*_H(x') \\
&\ldots \\
J^*_k(x) &= \min_u g(x,u) + \sum_{x'} P(x'|x,u) J^*_{k+1}(x') \\
&\ldots \\
J^*_0(x) &= \min_u g(x,u) + \sum_{x'} P(x'|x,u) J^*_1(x')
\end{aligned}$$

And

$$\mu^*_k(x) = \arg\min_u g(x,u) + \sum_{x'} P(x'|x,u) J^*_{k+1}(x');$$

- Running time: O(|S|^2 |A| H) vs. naïve search over all policies would require evaluation of |A|$^{|S|H}$ policies

Discounted infinite horizon

- Markov decision process (MDP) (S, A, P, γ, g)
  - γ: discount factor
- Policy $\pi = (\mu_0, \mu_1, \ldots)$, $\mu_k : S \rightarrow A$
- Value of a policy π: $J^\pi(x) = \mathrm{E}[\sum_{t=0}^{\infty} \gamma^t g(x(t), u(t)) | x_0 = x, \pi]$
- Goal: find $\pi^* \in \arg\min_{\pi \in \Pi} V^\pi$

Discounted infinite horizon

- **Dynamic programming (DP) aka Value iteration (VI):**

For i=0,1, …

For all s ∈ S

$$J^{(i+1)}(s) \quad \leftarrow \quad \min_{u \in A} \sum_{s'} P(s'|s,u) \left(g(s,a) + \gamma J^{(i)}(s') \right)$$

- **Facts:**

$J^{(i)} \rightarrow J^*$ for $i \rightarrow \infty$

There is an optimal stationary policy: $\pi^* = (\mu^*, \mu^*, \ldots)$ which satisfies:

$$\mu^*(x) = \arg\min_u g(x,u) + \gamma \sum_{x'} P(x'|x,u) J^*(x)$$

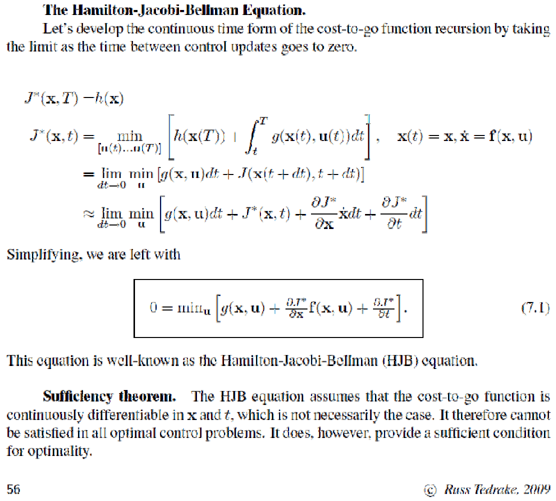
Continuous time and state-action space

- Hamilton-Jacobi-Bellman equation / approach:
  - Continuous equivalent of discrete case we already discussed

  → We will see 2 slides.
- Variational / Markov chain approximation method:
  - Numerically solve a continuous problem by directly approximating the continuous MDP with a discrete MDP

  → We will study this approach in detail.

Hamilton-Jacobi-Bellman (HJB) [*]



The Hamilton-Jacobi-Bellman Equation.
Let's develop the continuous time form of the cost-to-go function recursion by taking the limit as the time between control updates goes to zero.

$$J^*(\mathbf{x},T) - h(\mathbf{x})$$

$$\begin{aligned} J^*(\mathbf{x},t) &= \min_{[\mathbf{u}(t)\ldots\mathbf{u}(T)]}\left[h(\mathbf{x}(T)) + \int_t^T g(\mathbf{x}(t),\mathbf{u}(t))dt\right], \quad \mathbf{x}(t)=\mathbf{x}, \dot{\mathbf{x}}=\mathbf{f}(\mathbf{x},\mathbf{u}) \\ &= \lim_{dt\to 0}\min_{\mathbf{u}}\left[g(\mathbf{x},\mathbf{u})dt + J(\mathbf{x}(t+dt),t+dt)\right] \\ &\approx \lim_{dt\to 0}\min_{\mathbf{u}}\left[g(\mathbf{x},\mathbf{u})dt + J^*(\mathbf{x},t) + \frac{\partial J^*}{\partial \mathbf{x}}\dot{\mathbf{x}}dt + \frac{\partial J^*}{\partial t}dt\right] \end{aligned}$$

Simplifying, we are left with

$$\boxed{0 = \min_{\mathbf{u}}\left[g(\mathbf{x},\mathbf{u}) + \tfrac{\partial J^*}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u}) + \tfrac{\partial J^*}{\partial t}\right].} \qquad (7.1)$$

This equation is well-known as the Hamilton-Jacobi-Bellman (HJB) equation.

Sufficiency theorem. The HJB equation assumes that the cost-to-go function is continuously differentiable in $\mathbf{x}$ and t, which is not necessarily the case. It therefore cannot be satisfied in all optimal control problems. It does, however, provide a sufficient condition for optimality.

56 © *Russ Tedrake, 2009*

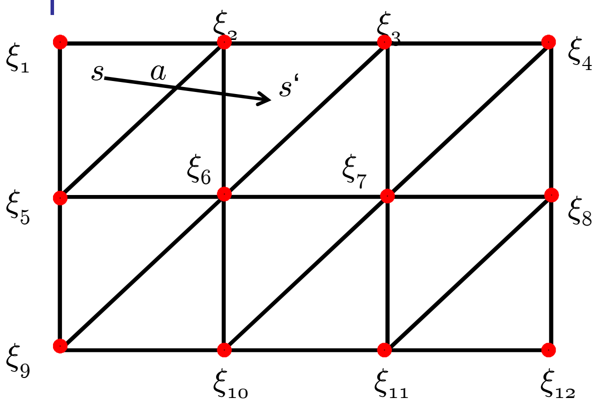
Hamilton-Jacobi-Bellman (HJB) [*]

- Can also derive HJB equation for the stochastic setting. Keywords for finding out more: Controlled diffusions / diffusion jump processes.
  - For special cases, can assist in finding / verifying analytical solutions
  - However, for most cases, need to resort to numerical solution methods for the corresponding PDE --- or directly approximate the control problem with a Markov chain
- References:
  - Tedrake Ch. 7; Bertsekas, "Dynamic Programming and Optimal Control."
  - Oksendal, "Stochastic Differential Equations: An Introduction with Applications"
  - Oksendal and Sulem, "Applied Stochastic Control of Jump Diffusions"
  - Michael Steele, "Stochastic Calculus and Financial Applications"
  - Markov chain approximations: Kushner and Dupuis, 1992/2001

Markov chain approximation (“discretization”)

- Original MDP (S, A, P, R, γ)
- Discretized MDP:
  - Grid the state-space: the vertices are the discrete states.
  - Reduce the action space to a finite set.
    - Sometimes not needed:
      - When Bellman back-up can be computed exactly over the continuous action space
      - When we know only certain controls are part of the optimal policy (e.g., when we know the problem has a “bang-bang” optimal solution)
  - Transition function remains to be resolved!

Discretization: example 1



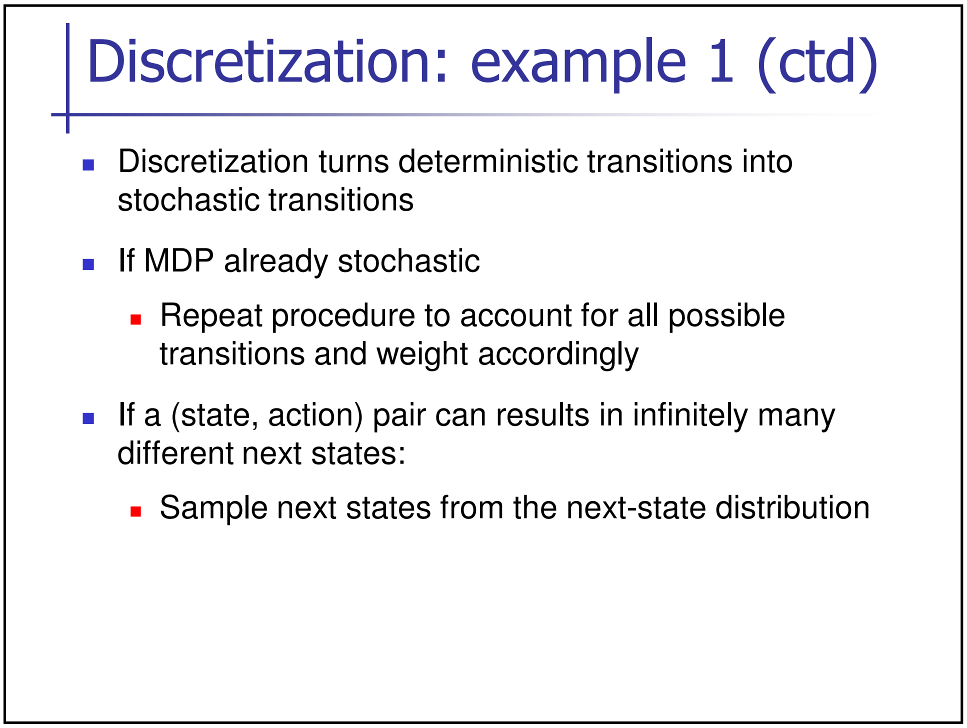
Discrete states: { ξ_1 , …, ξ_{12} }

$P(\xi_2|s,a) = p_A;$
$P(\xi_3|s,a) = p_B;$
$P(\xi_6|s,a) = p_C;$
s.t. $s' = p_A\xi_2 + p_B\xi_3 + p_C\xi_6$

- Results in discrete MDP, which we know how to solve.
- Policy when in “continuous state”:

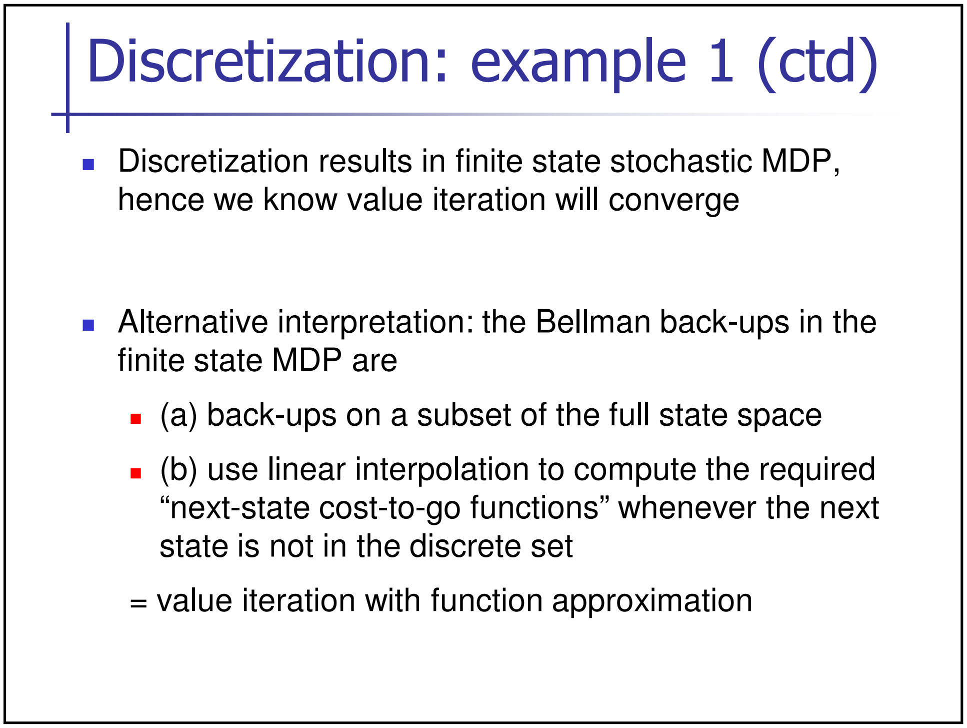
$$\pi(s) = \arg\min_a g(s,a) + \gamma \sum_{s'} P(s'|s,a) \sum_i P(\xi_i; s') J(\xi_i)$$

- Note: need not be triangular. [See also: Munos and Moore, 2001.]



Discretization: example 1 (ctd)

- Discretization turns deterministic transitions into stochastic transitions
- If MDP already stochastic
  - Repeat procedure to account for all possible transitions and weight accordingly
- If a (state, action) pair can results in infinitely many different next states:
  - Sample next states from the next-state distribution

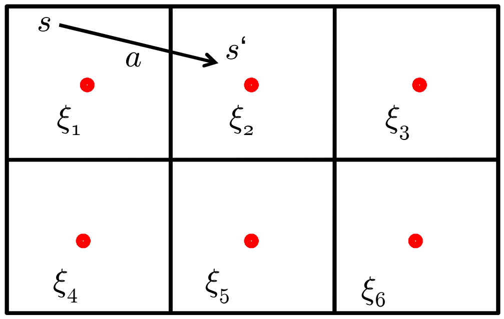


Discretization: example 1 (ctd)

- Discretization results in finite state stochastic MDP, hence we know value iteration will converge
- Alternative interpretation: the Bellman back-ups in the finite state MDP are
  - (a) back-ups on a subset of the full state space
  - (b) use linear interpolation to compute the required “next-state cost-to-go functions” whenever the next state is not in the discrete set

  = value iteration with function approximation

Discretization: example 2



Discrete states: { ξ_1 , ..., ξ_6 }

$P(\xi_2|s,a) = 1;$

Similarly define transition probabilities for all ξ_i

- Results in discrete MDP, which we know how to solve.
- Policy when in “continuous state”:

$$\pi(s) = \arg\min_a g(s,a) + \gamma \sum_{s'} P(s'|s,a) \sum_i P(\xi_i; s') J(\xi_i)$$

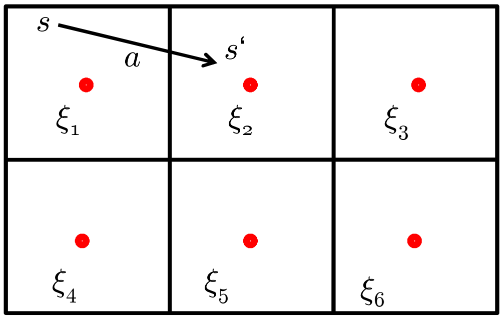
- This is nearest neighbor; could also use weighted combination of nearest neighbors.

Discretization: example 2 (ctd)

- Discretization results in finite state (stochastic) MDP, hence we know value iteration will converge
- Alternative interpretation: the Bellman back-ups in the finite state MDP are
  - (a) back-ups on a subset of the full state space
  - (b) use nearest neighbor interpolation to compute the required “next-state cost-to-go functions” whenever the next state is not in the discrete set

  = value iteration with function approximation

Discretization: example 3



Discrete states: { ξ_1 , …, ξ_6 }

$$P(\xi_i|\xi_j,u) = \frac{\int_{s\in\xi_j} P(s'|s,u)1\{s'\in\xi_i\}ds}{\int_{s\in\xi_j} P(s'|s,u)ds}$$

After entering a region, the state gets uniformly reset to any state from that region.

[Chow and Tsitsiklis, 1991]

Discretization: example 3 (ctd)

- Discretization results in a similar MDP as for example 2
  - Main difference: transition probabilities are computed based upon a region rather than the discrete states

Continuous time

- One might want to discretize time in a variable way such that one discrete time transition roughly corresponds to a transition into neighboring grid points/regions
- Discounting: $$\exp(-\beta\delta t)$$
  δt depends on the state and action

See, e.g., Munos and Moore, 2001 for details.

Note: Numerical methods research refers to this connection between time and space as the CFL (Courant Friedrichs Levy) condition. Googling for this term will give you more background info.

!! 1 nearest neighbor tends to be especially sensitive to having the correct match [Indeed, with a mismatch between time and space 1 nearest neighbor might end up mapping many states to only transition to themselves no matter which action is taken.]

Example: Double integrator---minimum time

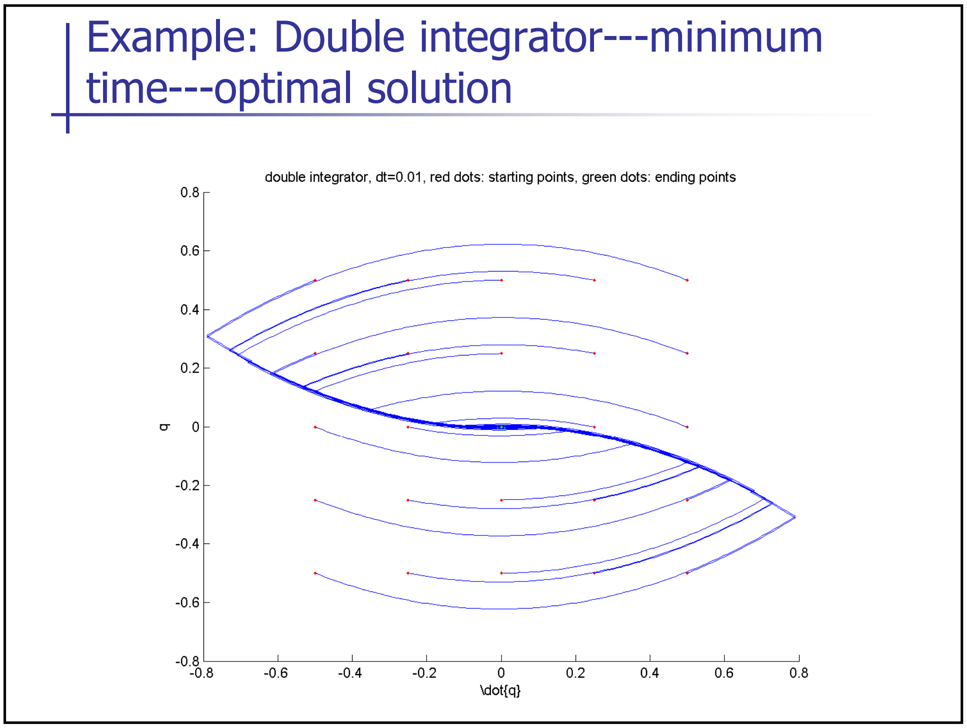
- Continuous time: $\ddot{q} = u, \quad \forall t : u(t) \in [-1, +1]$
- Objective: reach origin in minimum time

$$g(q, \dot{q}, u) = \left[\begin{array}{ll} 0 & \text{if } q = \dot{q} = 0 \\ 1 & \text{otherwise} \end{array} \right.$$

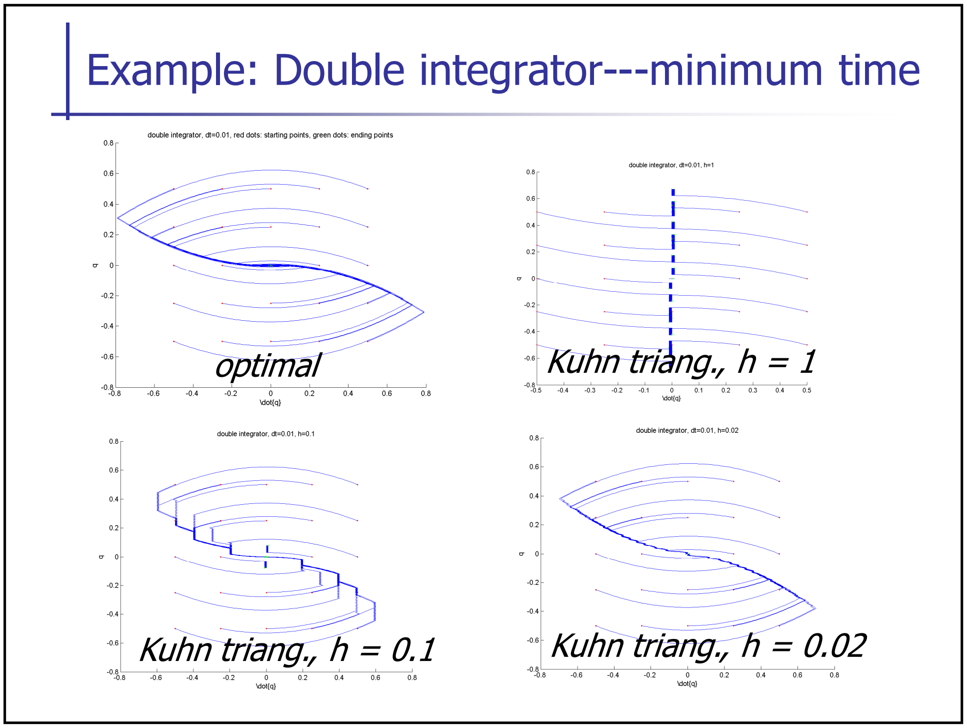
- Can be solved analytically: optimal policy is bang-bang: the control system should accelerate maximally towards the origin until a critical point at which it should hit the brakes in order to come perfectly to rest at the origin. This results in:

$$u = \left[\begin{array}{ll} 1 & \text{if } \dot{q} \leq -\text{sign}(q)\sqrt{2\text{sign}(q)q} \\ -1 & \text{otherwise} \end{array} \right.$$

[See Tedrake 6.6.3 for further details.]



Example: Double integrator---minimum time---optimal solution



Example: Double integrator---minimum time

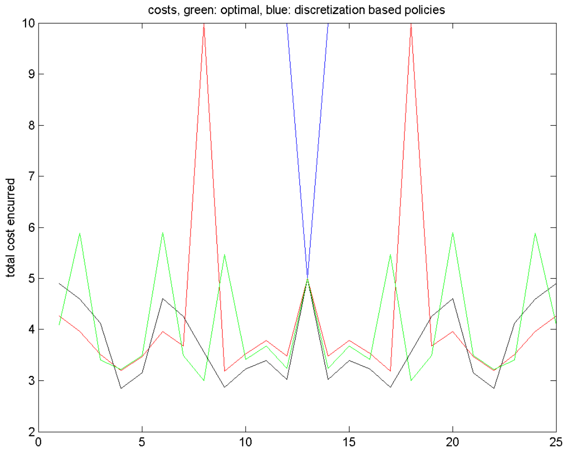
optimal

Kuhn triang., h = 1

Kuhn triang., h = 0.1

Kuhn triang., h = 0.02

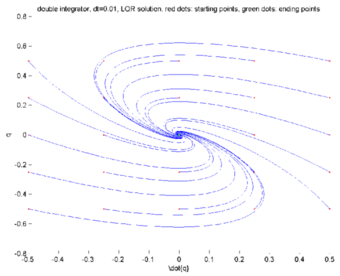
Resulting cost, Kuhn triang.



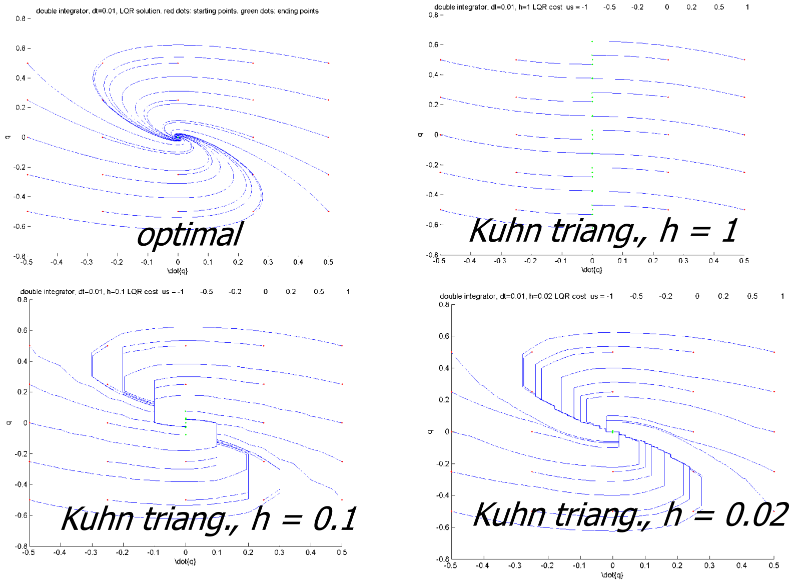
Green = continuous time optimal policy for mintime problem
For simulation we used: dt = 0.01; and goal area = within .01 of zero for q and \dot{q}.
This results in the continuous time optimal policy not being exactly optimal for the discrete time case.

Example: Double integrator---quadratic cost

- Continuous time: $\ddot{q} = u$
- In discrete time:
$$\begin{aligned} q_{t+1} &= q_t + \dot{q}_t \delta t \\ \dot{q}_{t+1} &= \dot{q}_t + u \delta_t \end{aligned}$$
- Cost function: $g(q, \dot{q}, u) = q^2 + u^2$



Example: Double integrator---quadratic cost



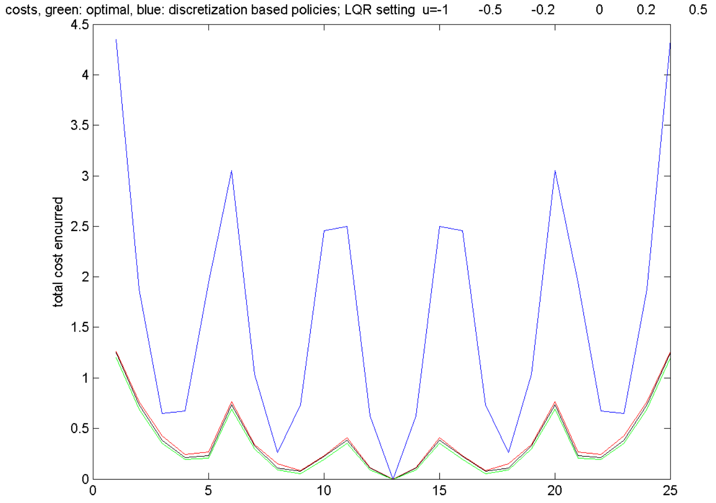
optimal

Kuhn triang., h = 1

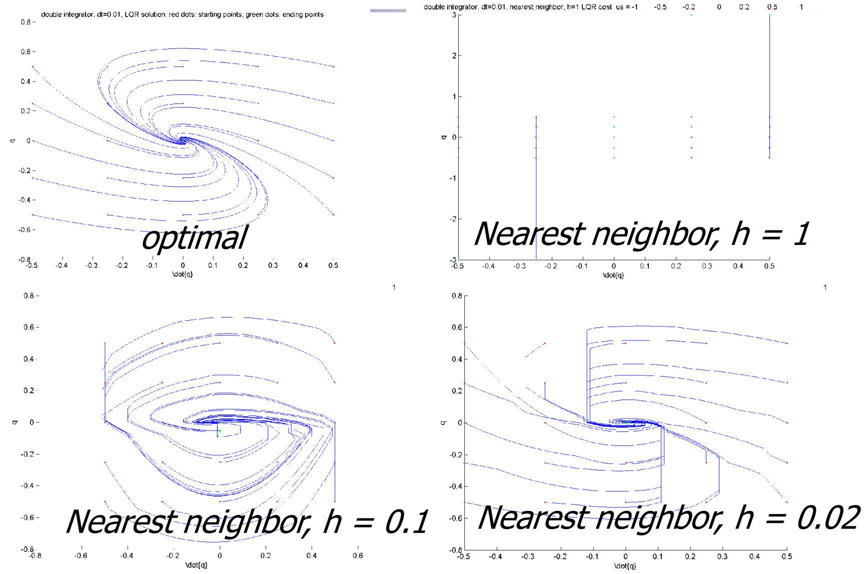
Kuhn triang., h = 0.1

Kuhn triang., h = 0.02

Resulting cost, Kuhn



Example: Double integrator---quadratic cost



double integrator, dt=0.01, LQR solution. red dots: starting points, green dots: ending points

double integrator, dt=0.01, nearest neighbor, h=1 LQR cost us = -1 -0.5 -0.2 0 0.2 0.5 1

optimal

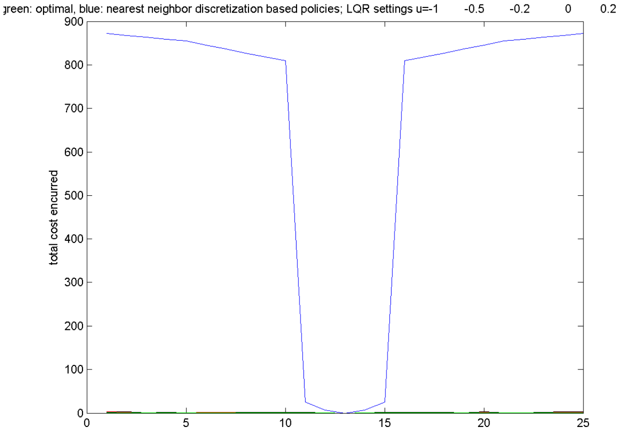
Nearest neighbor, h = 1

Nearest neighbor, h = 0.1

Nearest neighbor, h = 0.02

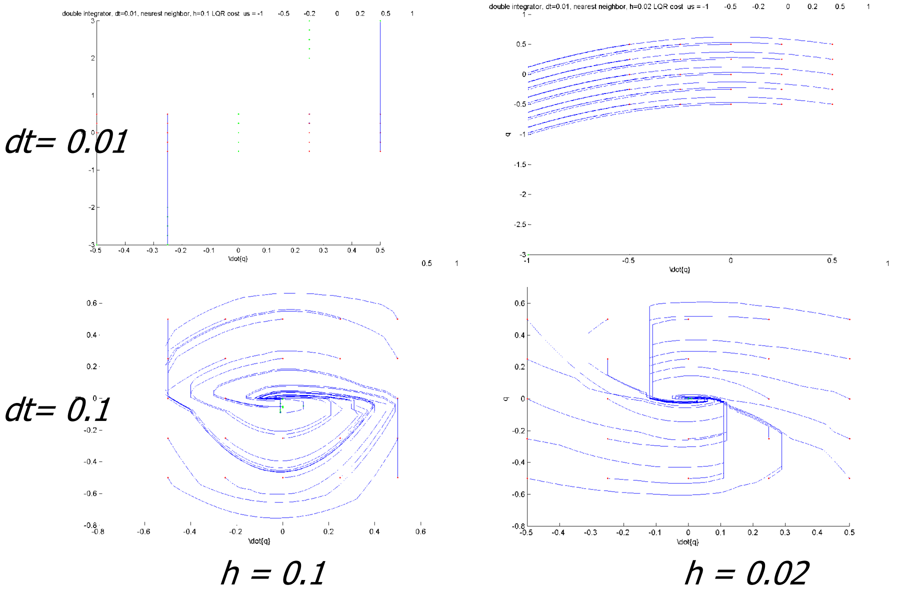
dt=0.1

Resulting cost, nearest neighbor



green: optimal, blue: nearest neighbor discretization based policies; LQR settings u=-1 -0.5 -0.2 0 0.2

Nearest neighbor quickly degrades when time and space scale are mismatched



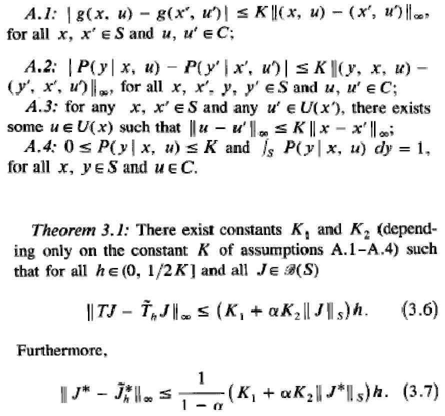
Discretization guarantees

- Typical guarantees:
  - Assume: smoothness of cost function, transition model
  - For h → 0, the discretized value function will approach the true value function
- Combine with:
  - Greedy policy w.r.t. value function V which is close to V* is a policy that attains value close to V*

Discretization proof techniques

- Chow and Tsitsiklis, 1991:
  - Show that one discretized back-up is close to one "complete" back-up + then show sequence of back-ups is also close
- Kushner and Dupuis, 2001:
  - Show that sample paths in discrete stochastic MDP approach sample paths in continuous (deterministic) MDP [also proofs for stochastic continuous, bit more complex]
- Function approximation based proof
  - Applies more generally to solving large-scale MDPs
  - Great descriptions: Gordon, 1995; Tsitsiklis and Van Roy, 1996

Example result (Chow and Tsitsiklis,1991)



A.1: $|g(x, u) - g(x', u')| \leq K\|(x, u) - (x', u')\|_\infty$, for all $x, x' \in S$ and $u, u' \in C$;

A.2: $|P(y \mid x, u) - P(y' \mid x', u')| \leq K\|(y, x, u) - (y', x', u')\|_\infty$, for all $x, x', y, y' \in S$ and $u, u' \in C$;

A.3: for any $x, x' \in S$ and any $u' \in U(x')$, there exists some $u \in U(x)$ such that $\|u - u'\|_\infty \leq K\|x - x'\|_\infty$;

A.4: $0 \leq P(y \mid x, u) \leq K$ and $\int_S P(y \mid x, u)\, dy = 1$, for all $x, y \in S$ and $u \in C$.

Theorem 3.1: There exist constants K_1 and K_2 (depending only on the constant K of assumptions A.1–A.4) such that for all $h \in (0, 1/2K]$ and all $J \in \mathscr{B}(S)$

$$\|TJ - \tilde{T}_h J\|_\infty \leq (K_1 + \alpha K_2 \|J\|_S)h. \qquad (3.6)$$

Furthermore,

$$\|J^* - \tilde{J}_h^*\|_\infty \leq \frac{1}{1-\alpha}(K_1 + \alpha K_2 \|J^*\|_S)h. \quad (3.7)$$

Function approximation

- General idea
  - Value iteration back-up on some states → V_{i+1}
  - Fit parameterized function to V_{i+1}

Discretization as function approximation

- Nearest neighbor discretization = piecewise constant
- Piecewise linear over “triangles” discretization