Biologically Inspired Adaptive Optimal Control and Learning

Tao Bian
Advisor: Zhong-Ping Jiang

Control and Networks (CAN) Lab,
Department of Electrical and Computer Engineering,
Tandon School of Engineering,
New York University, Brooklyn, NY 11201

April 13, 2017

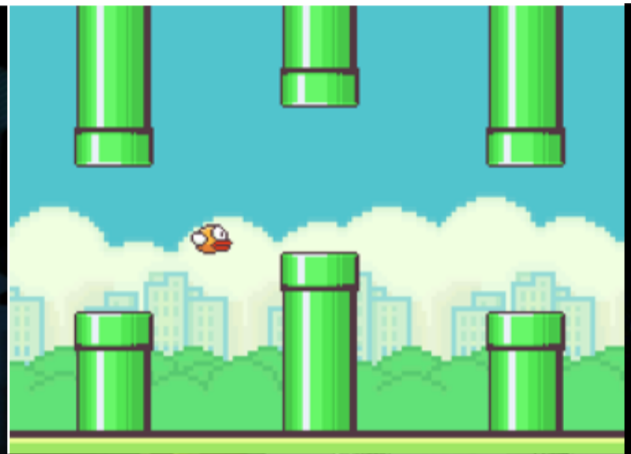
NYU TANDON SCHOOL OF ENGINEERING

Outline

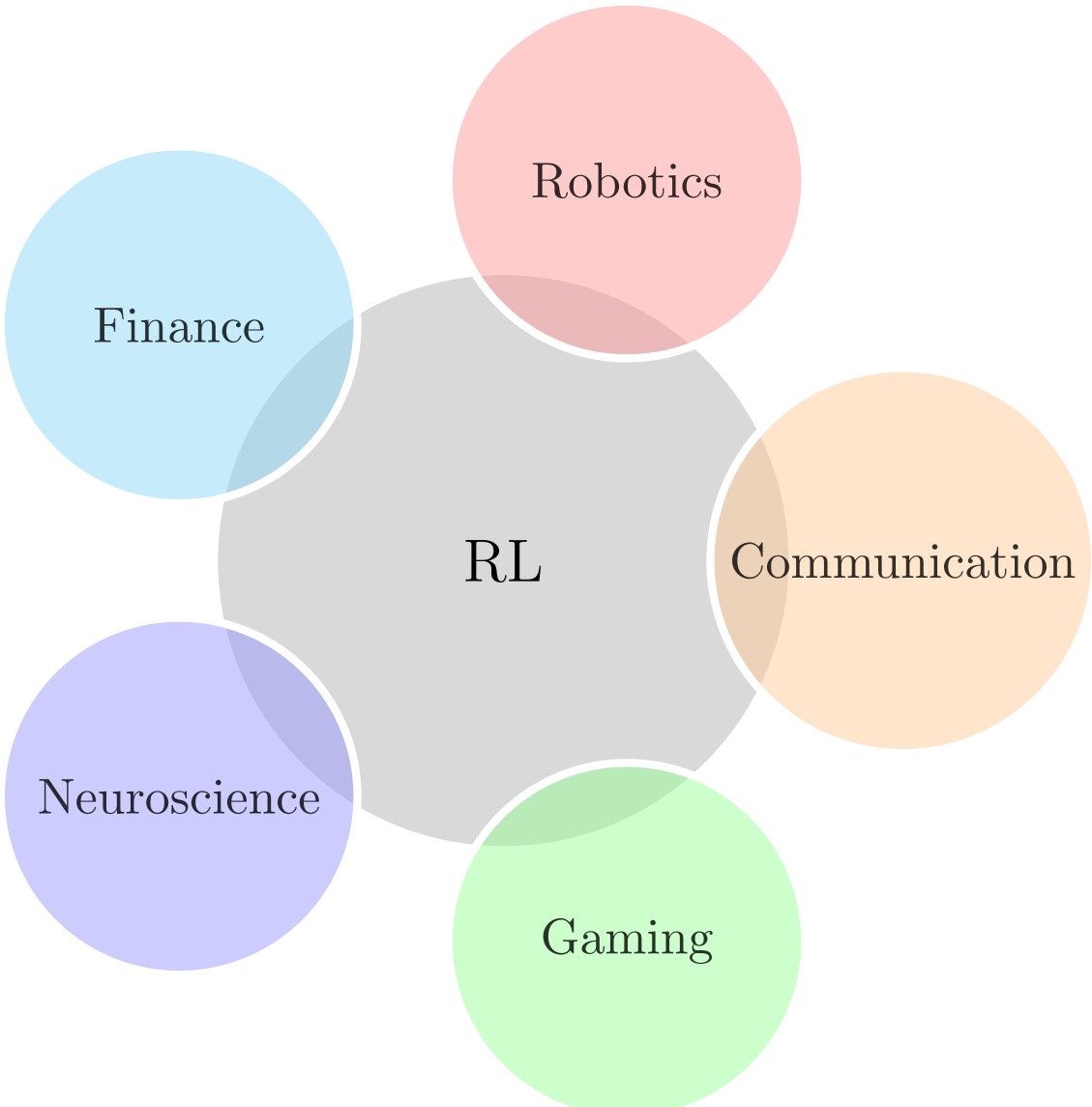
Outline

Outline

RL is everywhere



RL is everywhere



What is RL?

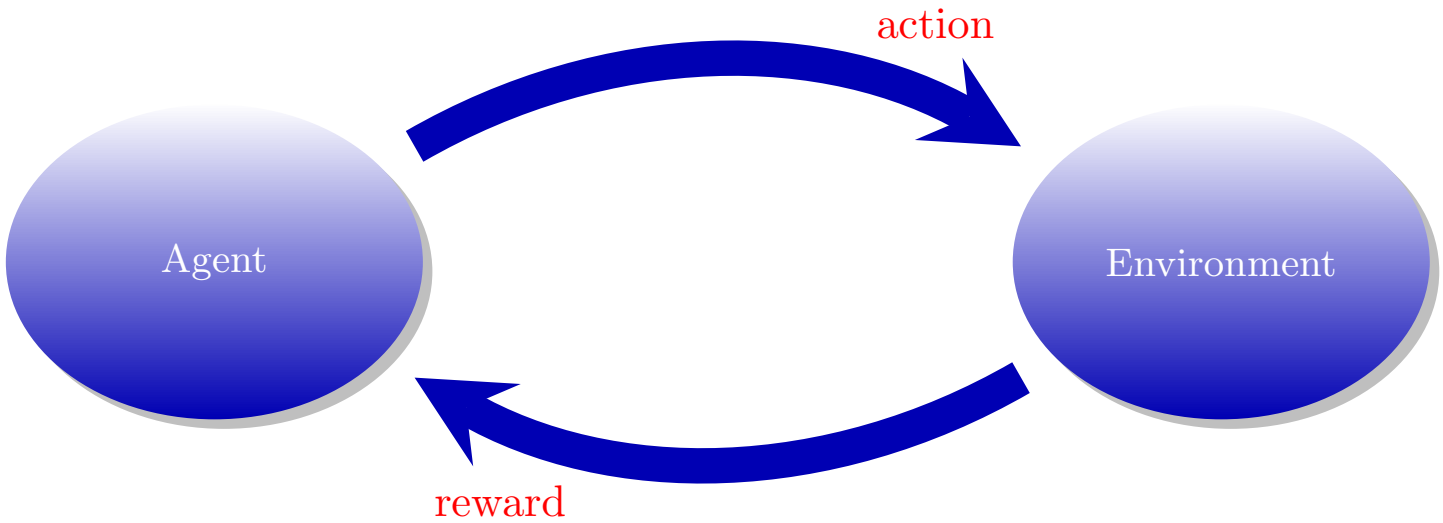


Figure: Reinforcement Learning.

- "Reinforcement" is a form of learning in which "an element, X, of the behavior" of an object, M, is caused to become more (or less !) "prominent" in the future behavior of M as a result of the "activation" of a special entity or process, Z, called the "reinforcement operator !") within a (relatively) short time after an occurrence of X. – Minsky. Theory of neural-analog reinforcement systems and its application to the brain model problem, 1954.
- Reinforcement learning is learning what to do–how to map situations to actions–so as to maximize a numerical reward signal. – Sutton and Barto. Reinforcement Learning: An Introduction, 1998.

From RL to ADP

Drawbacks of RL:

- Not applicable to dynamical systems (continuous-state-action space).

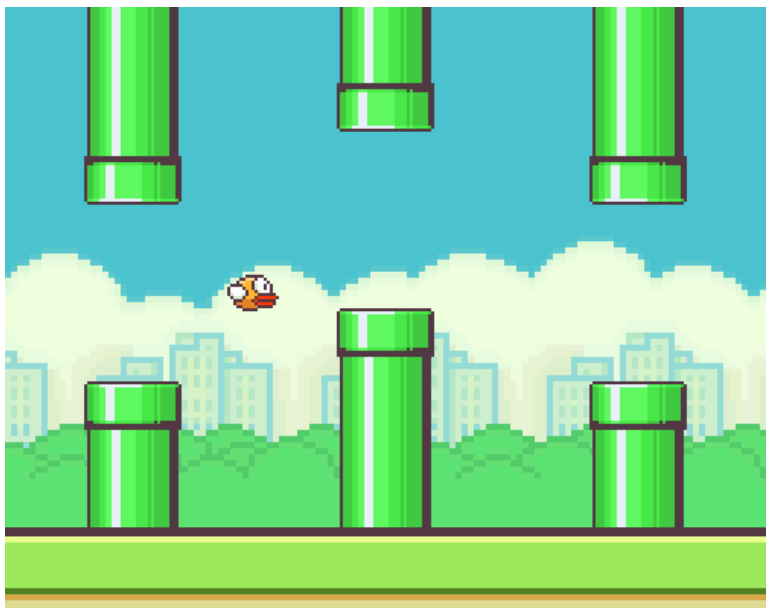


Figure: Play flappy bird in RL.

- The stability issue is usually overlooked.

From RL to ADP

Drawbacks of RL:

- Not applicable to dynamical systems (continuous-state-action space).

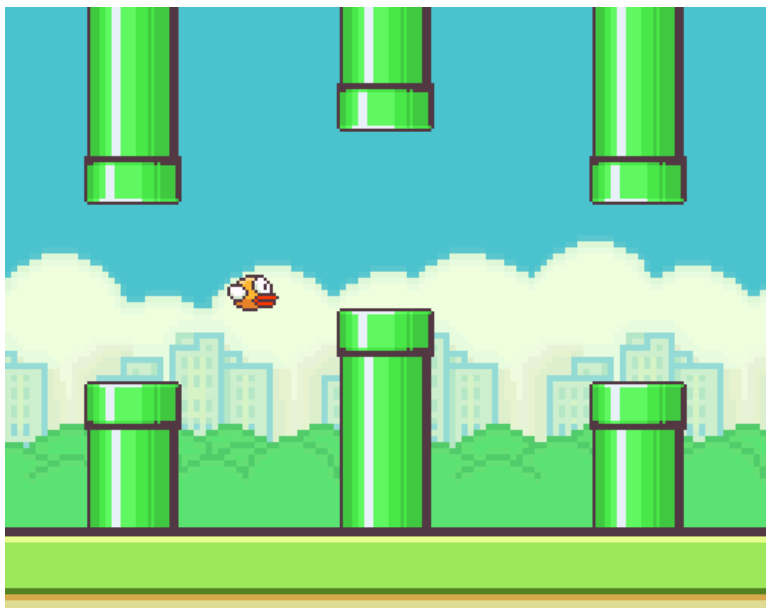
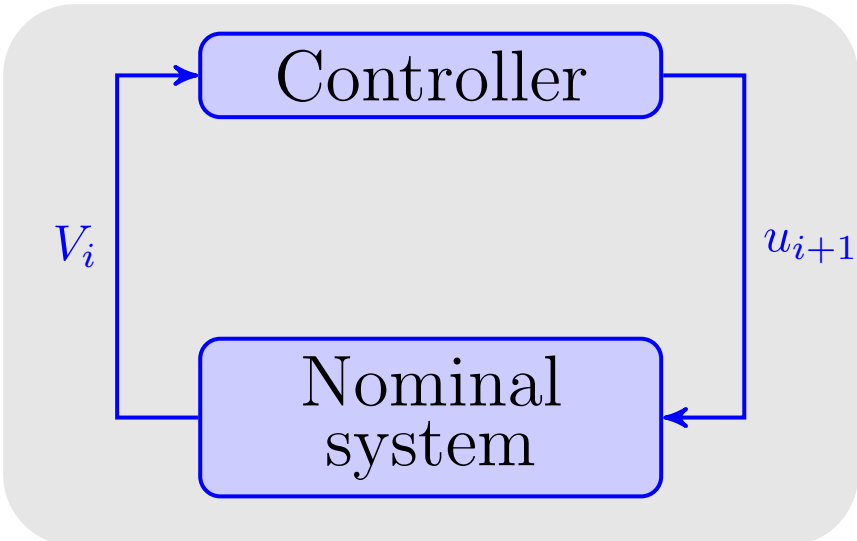


Figure: Play flappy bird in RL.

- The stability issue is usually overlooked.

A solution:



Adaptive dynamic programming (ADP) aims at finding a stabilizing optimal control policy for dynamical systems via online learning.

Outline

Bellman's equation and DP operator

Discrete-time, discrete-state-action space system:

$$x_{t+1} = f(x_t, a_t), \quad x_t \in \mathcal{S}, \quad a_t \in \mathcal{A}.$$

Objective: Find π^* to minimize

$$J(x_0; \pi) = \sum_{t=0}^{\infty} \alpha^t r(x_t, a_t),$$

where $\pi = \{a_0, a_1, \cdots\}$.

[1] under supremum norm.

Bellman's equation and DP operator

Discrete-time, discrete-state-action space system:

$$x_{t+1} = f(x_t, a_t), \quad x_t \in \mathcal{S}, \quad a_t \in \mathcal{A}.$$

Objective: Find π^* to minimize

$$J(x_0; \pi) = \sum_{t=0}^{\infty} \alpha^t r(x_t, a_t),$$

where $\pi = \{a_0, a_1, \cdots\}$.

Bellman's equation: $V^*(x_0) = \inf_\pi J(x_0; \pi)$,

$$V^*(x) = \mathcal{T}(V^*)(x) := \inf_{a \in \mathcal{A}} \{r(x, a) + \alpha V^*(f(x, a))\}.$$

[1] under supremum norm.

Bellman's equation and DP operator

Discrete-time, discrete-state-action space system:

$$x_{t+1} = f(x_t, a_t), \quad x_t \in \mathcal{S}, \quad a_t \in \mathcal{A}.$$

Objective: Find π^* to minimize

$$J(x_0; \pi) = \sum_{t=0}^{\infty} \alpha^t r(x_t, a_t),$$

where $\pi = \{a_0, a_1, \cdots\}$.

Bellman's equation: $V^*(x_0) = \inf_\pi J(x_0; \pi)$,

$$V^*(x) = \mathcal{T}(V^*)(x) := \inf_{a \in \mathcal{A}} \{r(x, a) + \alpha V^*(f(x, a))\}.$$

Value iteration (VI):

$$V_{k+1} = \mathcal{T}(V_k), \quad V_k \to V^*.$$

$\mathcal{T}$ is contractive[1] and monotone.

[1] under supremum norm.

History of value iteration

- 1957 VI for Markov decision processes (MDPs) (Bellman)
- 1960 The name of VI was introduced (Howard)
- 1995 VI for discrete-time, continuous-state-space linear systems (Lancaster & Rodman)
- 2015 VI for discrete-time, continuous-state-space nonlinear systems (Bertsekas)

History of value iteration

- 1957 VI for Markov decision processes (MDPs) (Bellman)
- 1960 The name of VI was introduced (Howard)
- 1995 VI for discrete-time, continuous-state-space linear systems (Lancaster & Rodman)
- 2015 VI for discrete-time, continuous-state-space nonlinear systems (Bertsekas)

- VI has been widely used in RL:
  - Watkins's Q-learning (Watkins, 1989);
  - TD-learning (Sutton, 1984).

History of value iteration

- 1957 VI for Markov decision processes (MDPs) (Bellman)
- 1960 The name of VI was introduced (Howard)
- 1995 VI for discrete-time, continuous-state-space linear systems (Lancaster & Rodman)
- 2015 VI for discrete-time, continuous-state-space nonlinear systems (Bertsekas)

- VI has been widely used in RL:
  - Watkins's Q-learning (Watkins, 1989);
  - TD-learning (Sutton, 1984).
- VI for continuous-time, continuous-state-space systems?

Contributions

- A new framework of VI for continuous-time, continuous-state-space systems.

Contributions

- A new framework of VI for continuous-time, continuous-state-space systems.
- VI-based adaptive optimal control design for continuous-time systems.

Contributions

- A new framework of VI for continuous-time, continuous-state-space systems.
- VI-based adaptive optimal control design for continuous-time systems.
- Model-free learning and control in biological motor systems.

Outline

Outline of continuous-time VI & ADP design

- Continuous-time DP operator ($\mathcal{T}$)?
- ADP in continuous-state-action space: $\mathcal{S} = \mathbb{R}^n$.
- Stability and robustness?

Outline of continuous-time VI & ADP design

- Continuous-time DP operator ($\mathcal{T}$)?
  - $\mathcal{T}$ exists
  - $\mathcal{T}$ is not necessarily a contraction mapping (noncontractive model)
- ADP in continuous-state-action space: $\mathcal{S} = \mathbb{R}^n$.

- Stability and robustness?

Outline of continuous-time VI & ADP design

- Continuous-time DP operator ($\mathcal{T}$)?
  - $\mathcal{T}$ exists
  - $\mathcal{T}$ is not necessarily a contraction mapping (noncontractive model)
- ADP in continuous-state-action space: $\mathcal{S} = \mathbb{R}^n$.
  - Basis function approximation $\Longrightarrow$ semi-global.
  - Linear systems $\Longrightarrow$ global.
- Stability and robustness?

Outline of continuous-time VI & ADP design

- Continuous-time DP operator ($\mathcal{T}$)?
  - $\mathcal{T}$ exists
  - $\mathcal{T}$ is not necessarily a contraction mapping (noncontractive model)
- ADP in continuous-state-action space: $\mathcal{S} = \mathbb{R}^n$.
  - Basis function approximation $\Longrightarrow$ semi-global.
  - Linear systems $\Longrightarrow$ global.
- Stability and robustness?
  - ADP + Lyapunov stability theory and small-gain theory.

Outline

Optimal control design for linear systems

Find a controller u^* which minimizes $\mathcal{J}$:

$$\text{System model:} \quad \dot{x} = Ax + Bu, \quad x(0) = x_0,$$

$$\text{Cost:} \quad \mathcal{J}(x_0; u) = \int_0^\infty (x^T Q x + u^T R u) dt.$$

Optimal control design for linear systems

Find a controller u^* which minimizes $\mathcal{J}$:

$$\text{System model:} \quad \dot{x} = Ax + Bu, \quad x(0) = x_0,$$

$$\text{Cost:} \quad \mathcal{J}(x_0; u) = \int_0^\infty (x^T Q x + u^T R u) dt.$$

- $u^* = -K^* x$;
- $V^*(x_0) = \inf_u \mathcal{J} = x_0^T P^* x_0$, $P^* = P^{*T} > 0$.

Algebric Riccati equation:

$$0 = \mathcal{R}(P^*).$$

where

$$\mathcal{R}(P^*) := A^T P^* + P^* A - P^* B R^{-1} B^T P^* + Q.$$

Continuous-time VI

How to solve $0 = \mathcal{R}(P^*)$?

Continuous-time VI

How to solve $0 = \mathcal{R}(P^*)$?

Step 1. $\dot{P} = \mathcal{R}(P)$: $\lim_{t\to\infty} P(t) = P^*$ (Willems, 1971), where $P(0) = P^T(0) > 0$.

Continuous-time VI

How to solve $0 = \mathcal{R}(P^*)$?

Step 1. $\dot{P} = \mathcal{R}(P)$: $\lim_{t\to\infty} P(t) = P^*$ (Willems, 1971), where $P(0) = P^T(0) > 0$.
Step 2. $P_{k+1} = P_k + \epsilon_k \mathcal{R}(P_k) + Z_k$ (Kushner and Yin, 2003)

$$\epsilon_k : \epsilon_k > 0,\ \lim_{k\to\infty} \epsilon_k = 0,\ \sum_{k=0}^{\infty} \epsilon_k = \infty,$$

$$Z_k : \{B_q\}_{q=0}^{\infty},\ B_q \subseteq B_{q+1},\ \lim_{q\to\infty} B_q = \{P \in \mathbb{R}^{n\times n} : P^T = P \geq 0\}.$$

VI for linear continuous-time systems

Algorithm 1 Value iteration algorithm

Choose $P_0 = P_0^T > 0$. $k, q \leftarrow 0$.
loop
  $\tilde{P}_{k+1} \leftarrow P_k + \epsilon_k(A^T P_k + P_k A - P_k B R^{-1} B^T P_k + Q)$
  if $\tilde{P}_{k+1} \notin B_q$ **then**
    $P_{k+1} \leftarrow P_0$. $q \leftarrow q + 1$
  else if $|\tilde{P}_{k+1} - P_k|/\epsilon_k < \varepsilon$ **then return** P_k
  else $P_{k+1} \leftarrow \tilde{P}_{k+1}$
  $k \leftarrow k + 1$

- Convergence[2]: $P_k \to P^*$.
- Stability: If $Q > \varepsilon I_n$, then $A - B\hat{K}^*$ is Hurwitz, where $\hat{K}^* = R^{-1}B^T\hat{P}^*$, and $\hat{P}^*$ is obtained from the VI algorithm.

[2] T. Bian and Z. P. Jiang, *Automatica* 71, 348–360, 2016.

An example of continuous-time VI

Illustration of VI in second-order systems ($x \in \mathbb{R}^2$) $\Longrightarrow$ $P_k \in \mathbb{R}^{2\times 2}$: $\left[\begin{smallmatrix} x & y \\ y & z \end{smallmatrix}\right] \Rightarrow (x, y, z)$.

$$\begin{bmatrix} x & y \\ y & z \end{bmatrix} > 0 \Longrightarrow \{(x, y, z) | x > 0, xz - y^2 > 0\},$$

$$B_q: \ 0 < \begin{bmatrix} x & y \\ y & z \end{bmatrix} < qI \Longrightarrow \{(x, y, z) | 0 < x < q, (q - x)(q - z) - y^2 > 0, xz - y^2 > 0\}.$$

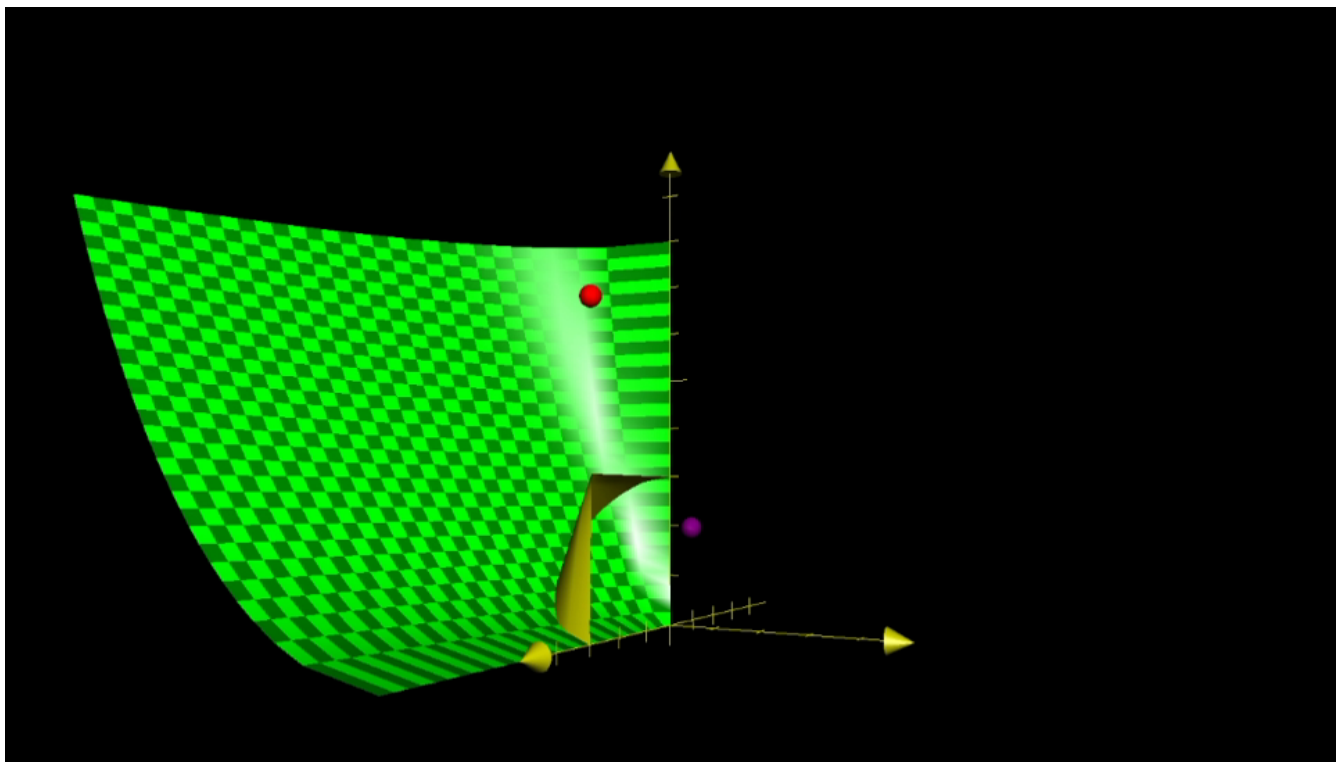


Figure: VI for a 2-d system.

VI-based ADP for linear systems

$$
\begin{aligned}
P_{k+1} &= P_k + \epsilon_k \mathcal{R}(P_k) + Z_k \\
&= P_k + \epsilon_k (A^T P_k + P_k A - P_k^T B R^{-1} B^T P_k + Q) + Z_k.
\end{aligned}
$$

VI-based ADP for linear systems

$$
\begin{aligned}
P_{k+1} &= P_k + \epsilon_k \mathcal{R}(P_k) + Z_k \\
&= P_k + \epsilon_k (A^T P_k + P_k A - P_k^T B R^{-1} B^T P_k + Q) + Z_k.
\end{aligned}
$$

$$
\frac{d}{dt}(x^T P_k x) = x^T (P_k A + A^T P_k) x + x^T P_k B u + u^T B^T P_k x.
$$

VI-based ADP for linear systems

$$P_{k+1} = P_k + \epsilon_k \mathcal{R}(P_k) + Z_k$$
$$= P_k + \epsilon_k (A^T P_k + P_k A - P_k^T B R^{-1} B^T P_k + Q) + Z_k.$$
$$\frac{d}{dt}(x^T P_k x) = x^T (P_k A + A^T P_k) x + x^T P_k B u + u^T B^T P_k x.$$

Denote $H_k = A^T P_k + P_k A$, $K_k = R^{-1} B^T P_k$.

Continuous-time VI-based ADP algorithm

Choose $P_0 = P_0^T > 0$. $k, q \leftarrow 0$.
Apply an input u to the system.
loop
  $P_k \Rightarrow (H_k, K_k)$.
  $\tilde{P}_{k+1} \leftarrow P_k + \epsilon_k (H_k + Q - K_k^T R K_k)$
  if $\tilde{P}_{k+1} \notin B_q$ **then** $P_{k+1} \leftarrow P_0$. $q \leftarrow q + 1$
  else if $|\tilde{P}_{k+1} - P_k|/\epsilon_k < \varepsilon$ **then**
    return P_k
  else $P_{k+1} \leftarrow \tilde{P}_{k+1}$
  $k \leftarrow k + 1$

$$\text{rank}([I_{xx}, 2I_{xu}]) = \frac{n(n+1)}{2} + nm,$$

where

$$I_{xu} = \left[\int_{t_0}^{t_1} x \otimes R u ds, \cdots, \int_{t_{l-1}}^{t_l} x \otimes R u ds \right]^T,$$
$$I_{xx} = \left[\int_{t_0}^{t_1} \bar{x} ds, \int_{t_1}^{t_2} \bar{x} ds, \cdots, \int_{t_{l-1}}^{t_l} \bar{x} ds \right]^T,$$
$$\bar{x} = [x_1^2, 2x_1 x_2, \cdots, 2x_{n-1} x_n, x_n^2].$$

Outline

Optimal control for continuous-time nonlinear systems

Nonlinear system:

$$\dot{x} = f(x, u), \quad \mathcal{J}(x_0; u) = \int_0^\infty r(x, u)dt, \quad x(0) = x_0.$$

Objective: $V^*(x_0) = \inf_u \mathcal{J}(x_0; u)$, $\mu^*(x_0) = \arg\inf_u \mathcal{J}(x_0; u)$.

[3] Suppose $V^* \in C^1$ and $\mu^* \in C^0$.

Optimal control for continuous-time nonlinear systems

Nonlinear system:

$$\dot{x} = f(x,u), \quad \mathcal{J}(x_0;u) = \int_0^\infty r(x,u)dt, \quad x(0) = x_0.$$

Objective: $V^*(x_0) = \inf_u \mathcal{J}(x_0;u)$, $\mu^*(x_0) = \arg\inf_u \mathcal{J}(x_0;u)$.

Hamilton-Jacobi-Bellman (HJB) equation[3]:

$$0 = \inf_{v\in\mathbb{R}^m} \{\partial_x V^*(x) f(x,v) + r(x,v)\}, \quad V^*(0) = 0,$$

$$\mu^*(x) = \arg\inf_{v\in\mathbb{R}^m} \{\partial_x V^*(x) f(x,v) + r(x,v)\}, \quad \forall x \in \mathbb{R}^n.$$

[3] Suppose $V^* \in C^1$ and $\mu^* \in C^0$.

Optimal control for continuous-time nonlinear systems

Nonlinear system:

$$\dot{x} = f(x,u), \quad \mathcal{J}(x_0;u) = \int_0^\infty r(x,u)dt, \quad x(0) = x_0.$$

Objective: $V^*(x_0) = \inf_u \mathcal{J}(x_0;u)$, $\mu^*(x_0) = \arg\inf_u \mathcal{J}(x_0;u)$.

Hamilton-Jacobi-Bellman (HJB) equation[3]:

$$0 = \inf_{v\in\mathbb{R}^m} \{\partial_x V^*(x) f(x,v) + r(x,v)\}, \quad V^*(0) = 0,$$
$$\mu^*(x) = \arg\inf_{v\in\mathbb{R}^m} \{\partial_x V^*(x) f(x,v) + r(x,v)\}, \quad \forall x \in \mathbb{R}^n.$$

How to solve the HJB equation using VI and ADP?

[3] Suppose $V^* \in C^1$ and $\mu^* \in C^0$.

DP operator for continuous-time systems

Continuous-time DP operator[4]:

$$\mathcal{T}_\tau(V) = \inf_u \mathcal{T}^u_{[s_f-\tau, s_f)}(V),$$

where

$$\mathcal{T}^u_{[s_0,s_f)}(V)(\xi) = V(x(s_f)) + \int_{s_0}^{s_f} r(x,u)ds, \quad x(s_0) = \xi \in \mathbb{R}^n.$$

[4] $\mathcal{T}_\tau$ is well defined, in the sense that if $0 \leq V(\xi) < \infty$ for all $\xi \in \mathbb{R}^n$, then $0 \leq \mathcal{T}_\tau(V)(\xi) < \infty$

DP operator for continuous-time systems

Continuous-time DP operator[4]:

$$\mathcal{T}_\tau(V) = \inf_u \mathcal{T}^u_{[s_f-\tau, s_f)}(V),$$

where

$$\mathcal{T}^u_{[s_0,s_f)}(V)(\xi) = V(x(s_f)) + \int_{s_0}^{s_f} r(x,u)ds, \quad x(s_0) = \xi \in \mathbb{R}^n.$$

Value iteration:

$$V_{k+1} = \mathcal{T}_{\tau_k}(V_k), \quad \tau_k > 0, \quad \sum_{k=0}^{\infty} \tau_k = \infty.$$

$\mathcal{T}_\tau$ is not necessarily a contraction mapping.

[4] $\mathcal{T}_\tau$ is well defined, in the sense that if $0 \le V(\xi) < \infty$ for all $\xi \in \mathbb{R}^n$, then $0 \le \mathcal{T}_\tau(V)(\xi) < \infty$

DP operator for continuous-time systems

Continuous-time DP operator[4]:

$$\mathcal{T}_\tau(V) = \inf_u \mathcal{T}^u_{[s_f-\tau, s_f)}(V),$$

where

$$\mathcal{T}^u_{[s_0,s_f)}(V)(\xi) = V(x(s_f)) + \int_{s_0}^{s_f} r(x,u)ds, \quad x(s_0) = \xi \in \mathbb{R}^n.$$

Value iteration:

$$V_{k+1} = \mathcal{T}_{\tau_k}(V_k), \quad \tau_k > 0, \quad \sum_{k=0}^{\infty} \tau_k = \infty.$$

$\mathcal{T}_\tau$ is not necessarily a contraction mapping.
But $\mathcal{T}_\tau$ is monotone:

$$V_1 \leq V_2 \Longrightarrow \mathcal{T}_\tau(V_1) \leq \mathcal{T}_\tau(V_2).$$

[4] $\mathcal{T}_\tau$ is well defined, in the sense that if $0 \leq V(\xi) < \infty$ for all $\xi \in \mathbb{R}^n$, then $0 \leq \mathcal{T}_\tau(V)(\xi) < \infty$

VI for general continuous-time nonlinear systems

Pointwise convergence[5] : if $V_0 \in \mathcal{C}^1$ is proper and positive semidefinite, then

$$V_k \to V^*.$$

[5] T. Bian and Z. P. Jiang, submitted to *IEEE TAC*, 2017.

VI for general continuous-time nonlinear systems

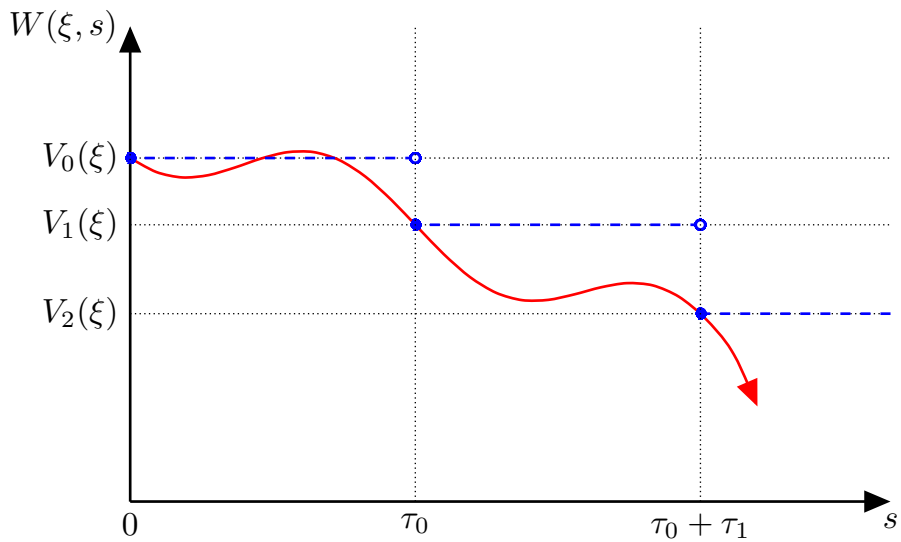
Pointwise convergence[5] : if $V_0 \in \mathcal{C}^1$ is proper and positive semidefinite, then

$$V_k \to V^*.$$

Finite-horizon HJB equation:

$$\partial_s W(\xi, s) = \inf_{u \in \mathbb{R}^m} \{\partial_x W(\xi, s) f(\xi, u) + r(\xi, u)\}.$$

$\lim_{s \to \infty} W(\cdot, s) = V^*(\cdot)$.



[5] T. Bian and Z. P. Jiang, submitted to *IEEE TAC*, 2017.

VI for general continuous-time nonlinear systems

Pointwise convergence[5] : if $V_0 \in \mathcal{C}^1$ is proper and positive semidefinite, then

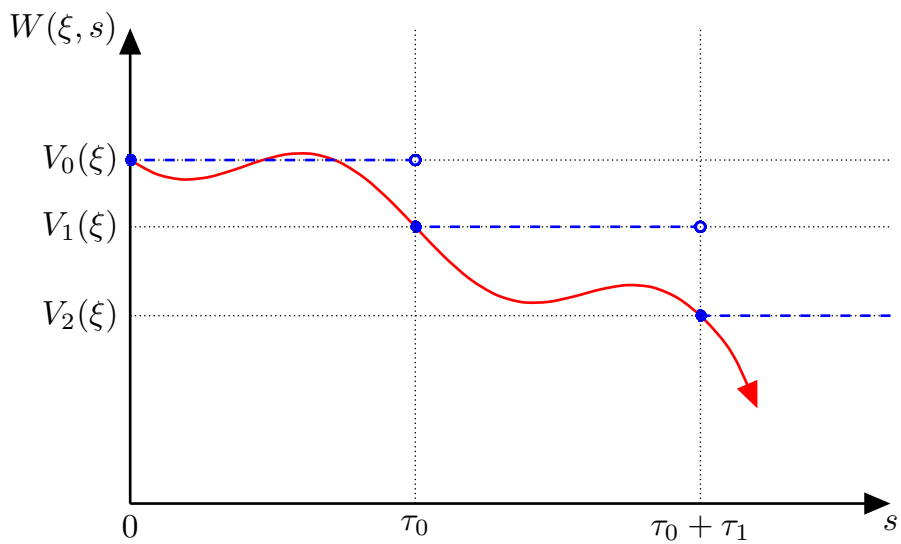
$$V_k \to V^*.$$

Finite-horizon HJB equation:

$$\partial_s W(\xi, s) = \inf_{u \in \mathbb{R}^m} \{\partial_x W(\xi, s) f(\xi, u) + r(\xi, u)\}.$$

$\lim_{s\to\infty} W(\cdot, s) = V^*(\cdot)$.

How to conduct ADP learning?



[5] T. Bian and Z. P. Jiang, submitted to *IEEE TAC*, 2017.

VI-based ADP for nonlinear systems

For any $x \in \mathbb{R}^n$,

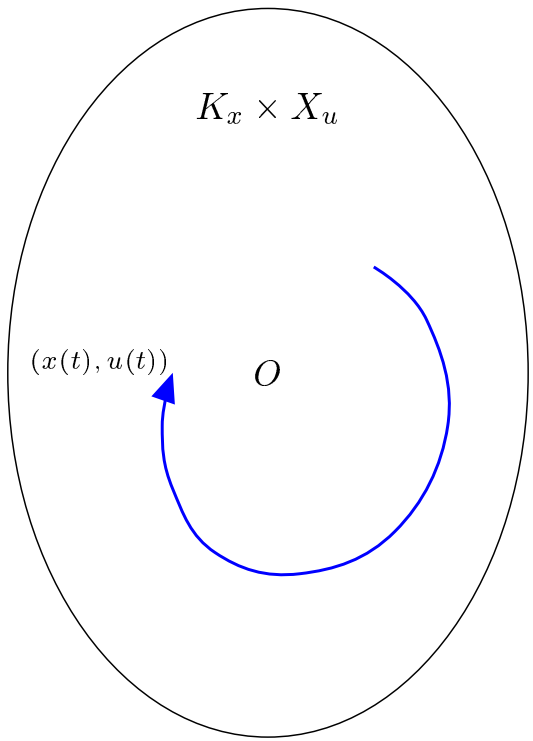
$$\partial_s W(x,s) = \inf_{u \in \mathbb{R}^m} \{\partial_x W(x,s) f(x,u) + r(x,u)\}, \quad \lim_{s \to \infty} W(x,s) = V^*(x).$$

Basis function approximation:

$$\sum_{i=1}^{\infty} w_i(s)\phi_i(x) = W(x,s),$$

$$\sum_{i=1}^{\infty} c_i(s)\psi_i(x,u) = \partial_x W(x,s) f(x,u),$$

on $K_x \times K_u \times [0, s_f]$.
(x,u) is obtained from the online trajectory.



VI-based ADP for nonlinear systems

For any $x \in \mathbb{R}^n$,

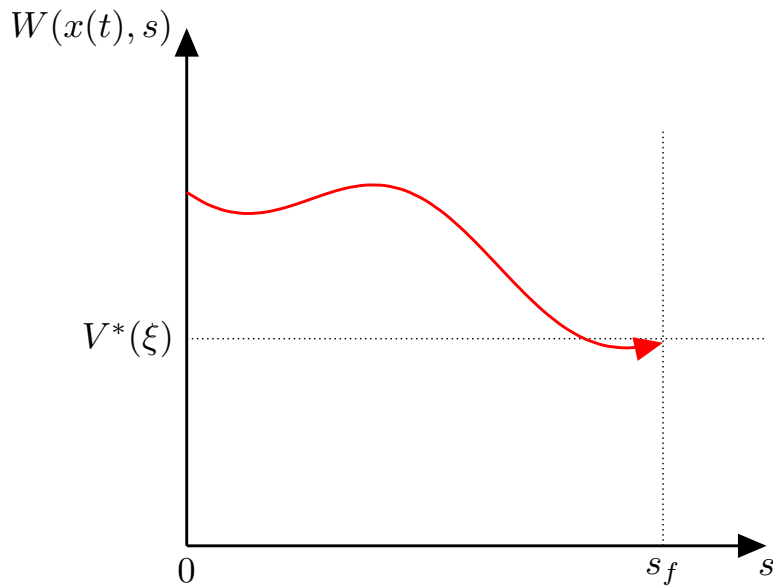
$$\partial_s W(x,s) = \inf_{u \in \mathbb{R}^m} \{\partial_x W(x,s) f(x,u) + r(x,u)\}, \quad \lim_{s \to \infty} W(x,s) = V^*(x).$$

Basis function approximation:

$$\sum_{i=1}^{\infty} w_i(s)\phi_i(x) = W(x,s),$$

$$\sum_{i=1}^{\infty} c_i(s)\psi_i(x,u) = \partial_x W(x,s) f(x,u),$$

on $K_x \times K_u \times [0, s_f]$.
(x,u) is obtained from the online trajectory.



VI-based ADP for nonlinear systems

VI-based ADP algorithm:

$$\hat{c}_k = \left(\sum_{j=0}^{M} \Theta_j^T \Theta_j \right)^{-1} \sum_{j=0}^{M} \Theta_j^T (\Phi(x_{j+1}) - \Phi(x_j)) \hat{w}_k,$$

$$\hat{w}_{k+1} = \hat{w}_k + h_k \left(\sum_{j=0}^{M} \Phi^T(x_j) \Phi(x_j) \right)^{-1} \sum_{j=0}^{M} \Phi^T(x_j) \inf_{u \in K_u} A_N(x_j, u, \hat{c}_k),$$

$$A_N(x, u, c) := \sum_{i=1}^{N} c_i(s) \psi_i(x, u) + r(x, u).$$

VI-based ADP for nonlinear systems

VI-based ADP algorithm:

$$\hat{c}_k = \left(\sum_{j=0}^{M} \Theta_j^T \Theta_j\right)^{-1} \sum_{j=0}^{M} \Theta_j^T (\Phi(x_{j+1}) - \Phi(x_j))\hat{w}_k,$$

$$\hat{w}_{k+1} = \hat{w}_k + h_k \left(\sum_{j=0}^{M} \Phi^T(x_j)\Phi(x_j)\right)^{-1} \sum_{j=0}^{M} \Phi^T(x_j) \inf_{u \in K_u} A_N(x_j, u, \hat{c}_k),$$

$$A_N(x, u, c) := \sum_{i=1}^{N} c_i(s)\psi_i(x, u) + r(x, u).$$

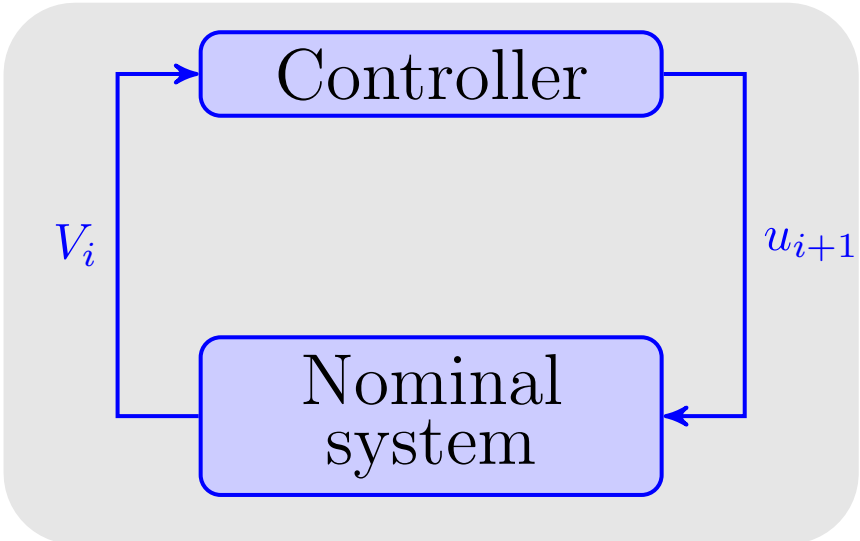
Convergence: If $\frac{1}{M}\sum_{j=0}^{M} \Phi^T(x_j)\Phi(x_j) > \gamma I_N$, $\frac{1}{M}\sum_{j=0}^{M} \Theta_j^T\Theta_j > \gamma I_N$, then

$$\sum_{i=1}^{N} \hat{w}_i(s_f)\phi_i \to V^*, \quad \arg\inf_{u \in K_u} A_N(x, u, \hat{c}_k) \to \mu^*(x), \quad \text{on } K_x.$$

Stability: The closed-loop system is semi-globally practically asymptotically stable at the origin.

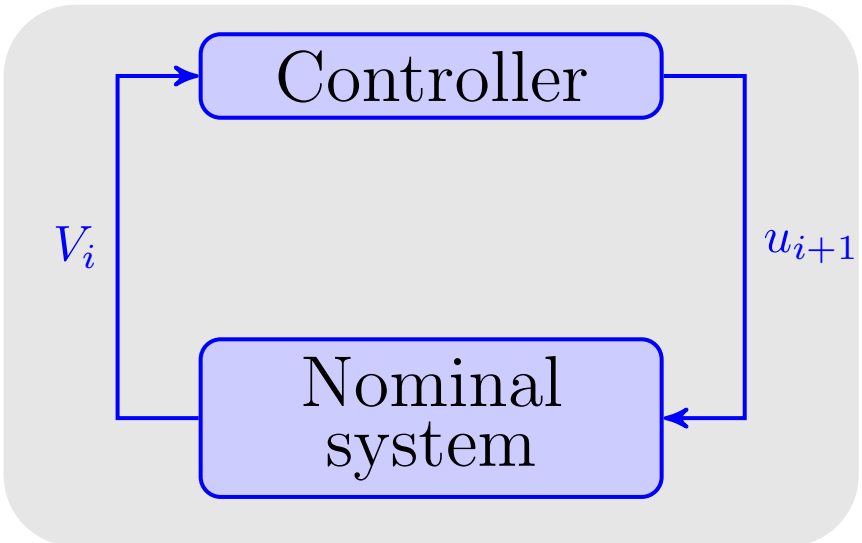
Outline

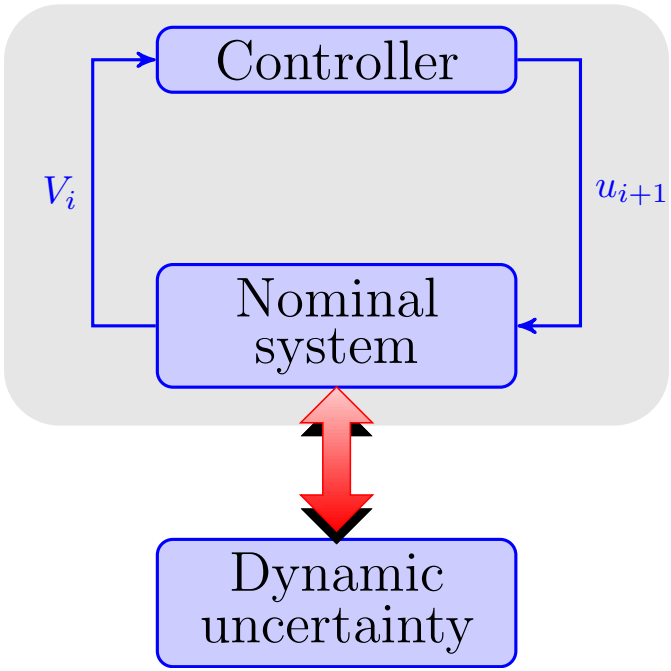
Why robust ADP?



- Robustness?
- Globally asymptotically stable?

Why robust ADP?



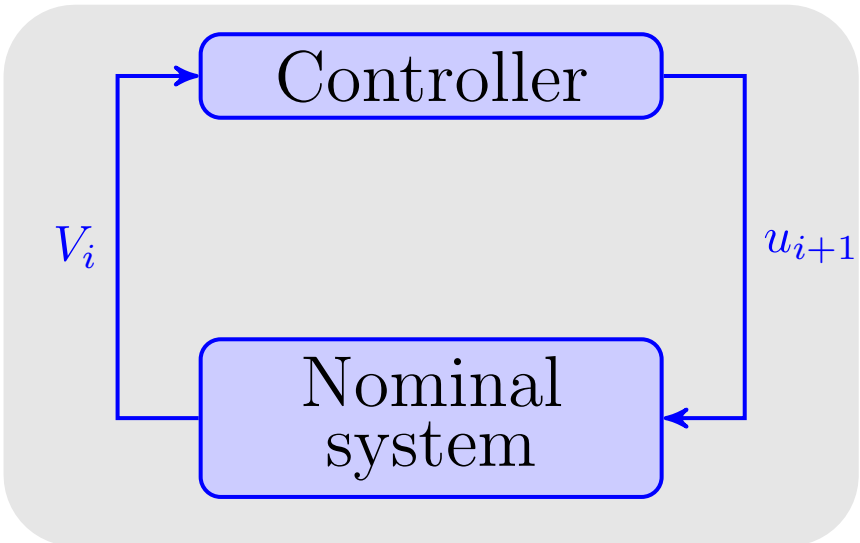


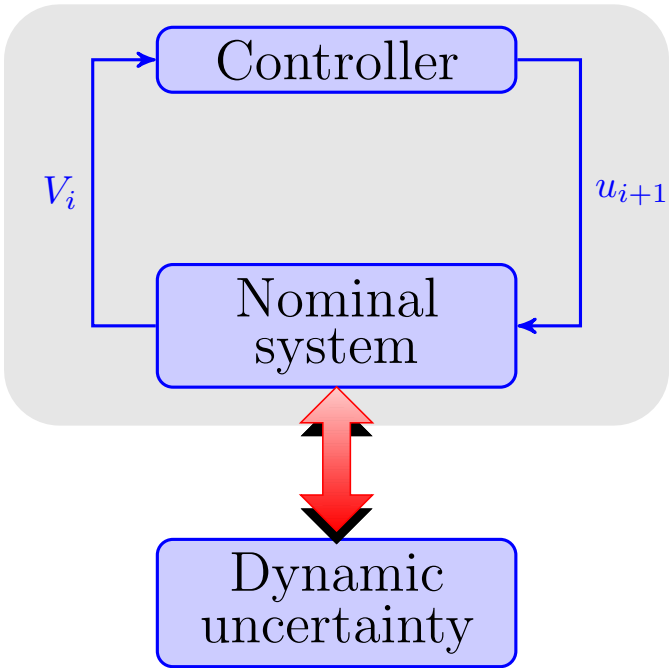
- Robustness?
- Globally asymptotically stable?

What is dynamic uncertainty?

1. A physical component.
2. Mismatch between the simplified model and the actual system dynamics.

Why robust ADP?





- Robustness?
- Globally asymptotically stable?

Robust optimal controller: Optimal in the absence of the dynamic uncertainty, and stabilizing when dynamic uncertainty occurs.

Problem formulation

Strict-feedback system (Krstić, et.al., 1995; Jiang and Praly, 1998):

$$dx = Axdt + B\left((z + \Delta_1(\varsigma, y, v))dt + \sum_{i=1}^{q} A_{0i}xdw_i\right),$$

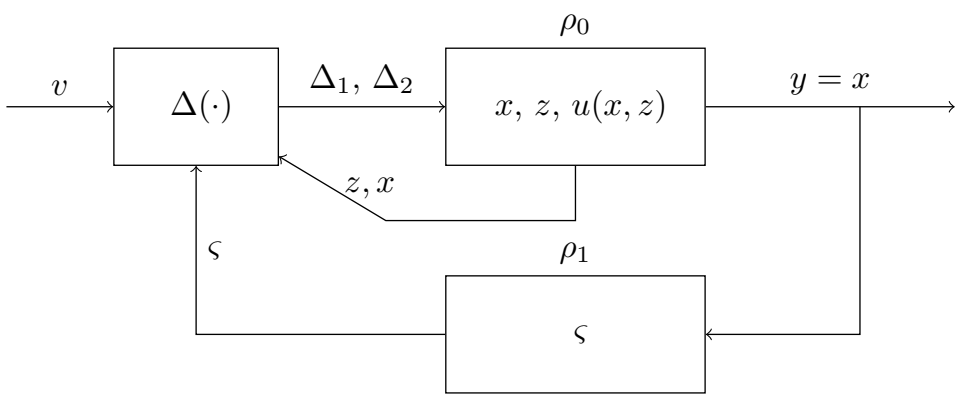
$$dz = G\left((u + \Delta_2(\varsigma, y, z, v))dt + \sum_{i=1}^{q}(E_{0i}x + F_{0i}z)dw_i\right),$$

$$d\varsigma = f(\varsigma, x)dt + \sum_{i=1}^{q} g_i(\varsigma, x)dw_i,$$

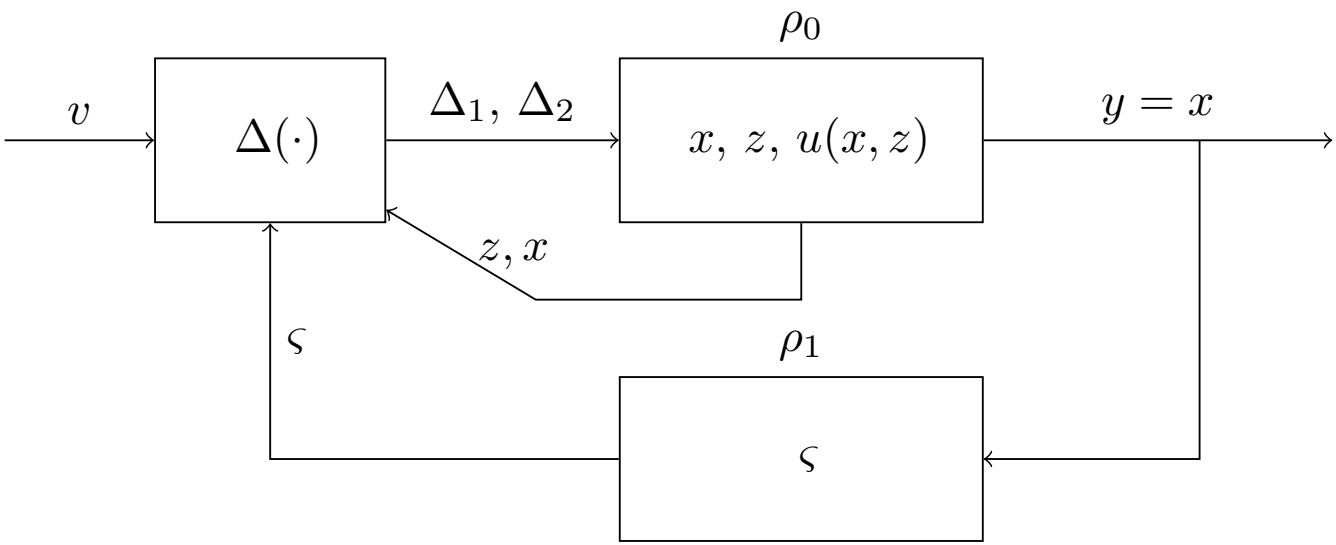
$$y = x,$$

where $(x, z, \varsigma) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$, $y \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, and $v \in \mathbb{R}^m$ are the state, output, input, and external disturbance of the system, respectively.

- Assume $\rho_1 < \infty$.
- Objective: Assign the linear L^2 gain from v to y.



Key results



1. ρ_0 can be made arbitrarily small, by stochastic gain assignment[6];
2. The closed-loop system is stochastic input-to-output stable w.r.t. v;
3. Two-phase robust ADP design[7].

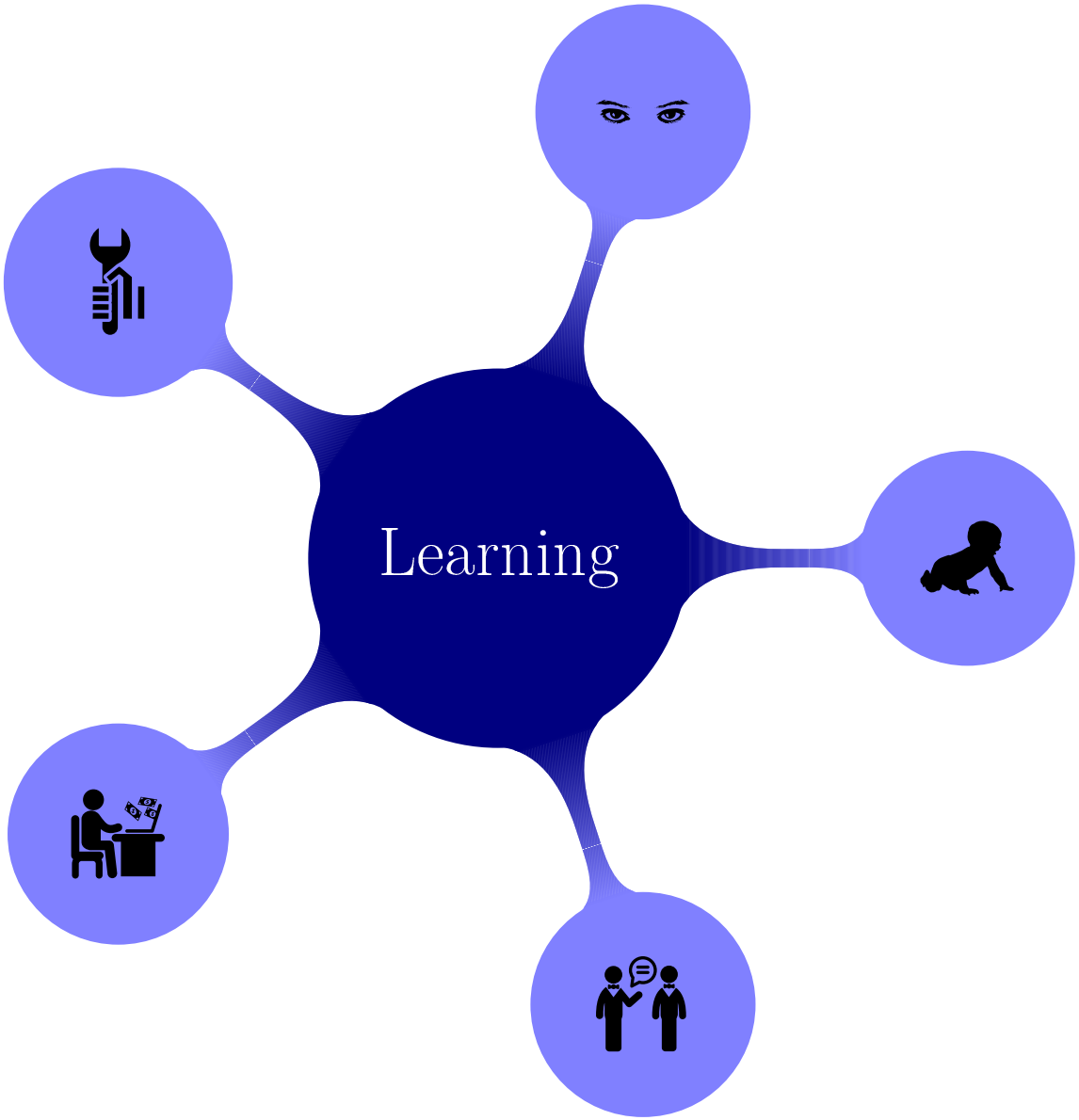
[6] T. Bian and Z. P. Jiang, , *IEEE TAC* 62, 1946–1951, 2017

[7] T. Bian and Z. P. Jiang, submitted to *Systems & Control Letters*, 2016.

Outline

What is human sensorimotor control?

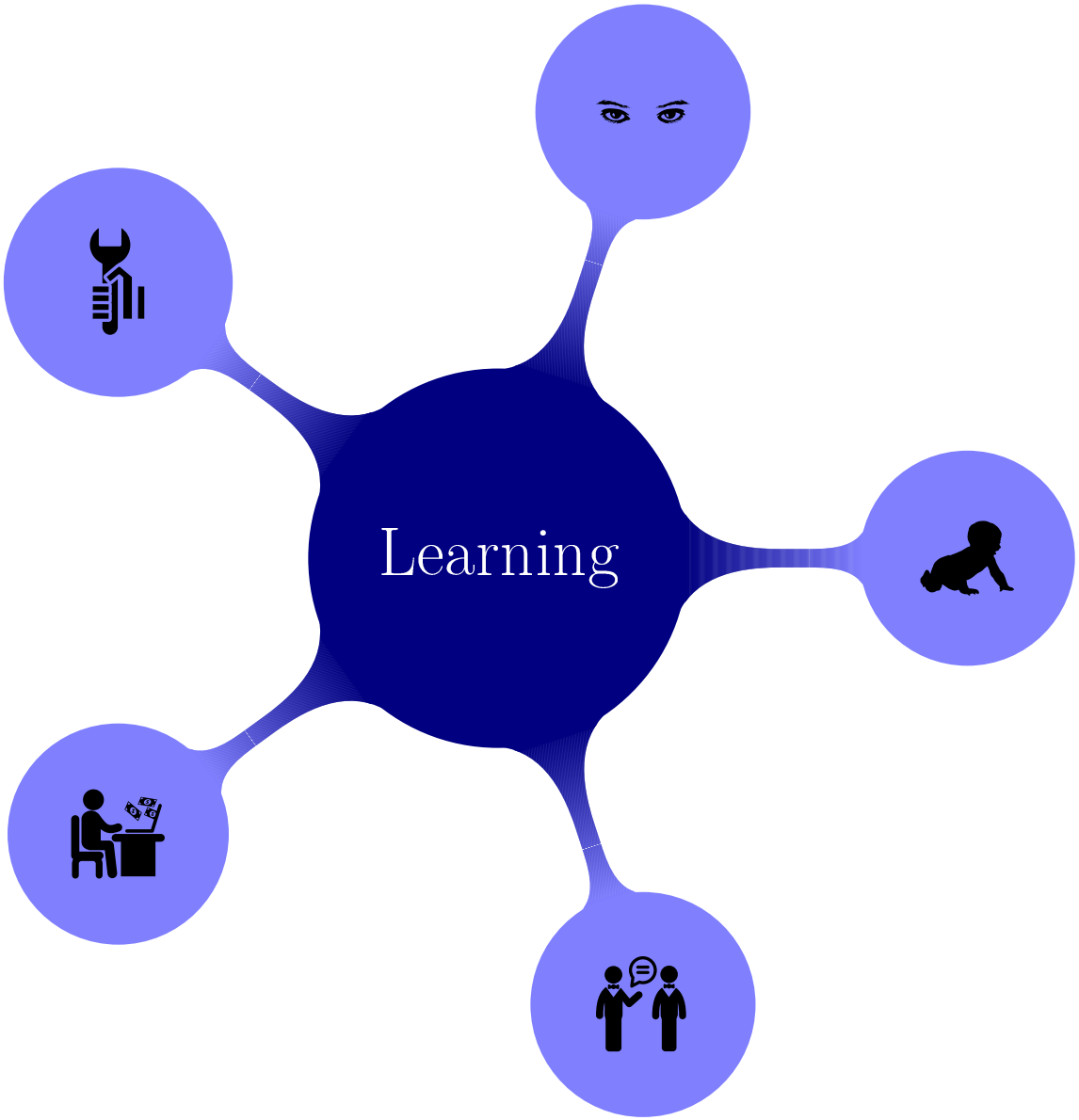
We learn to perform different tasks.



What is human sensorimotor control?

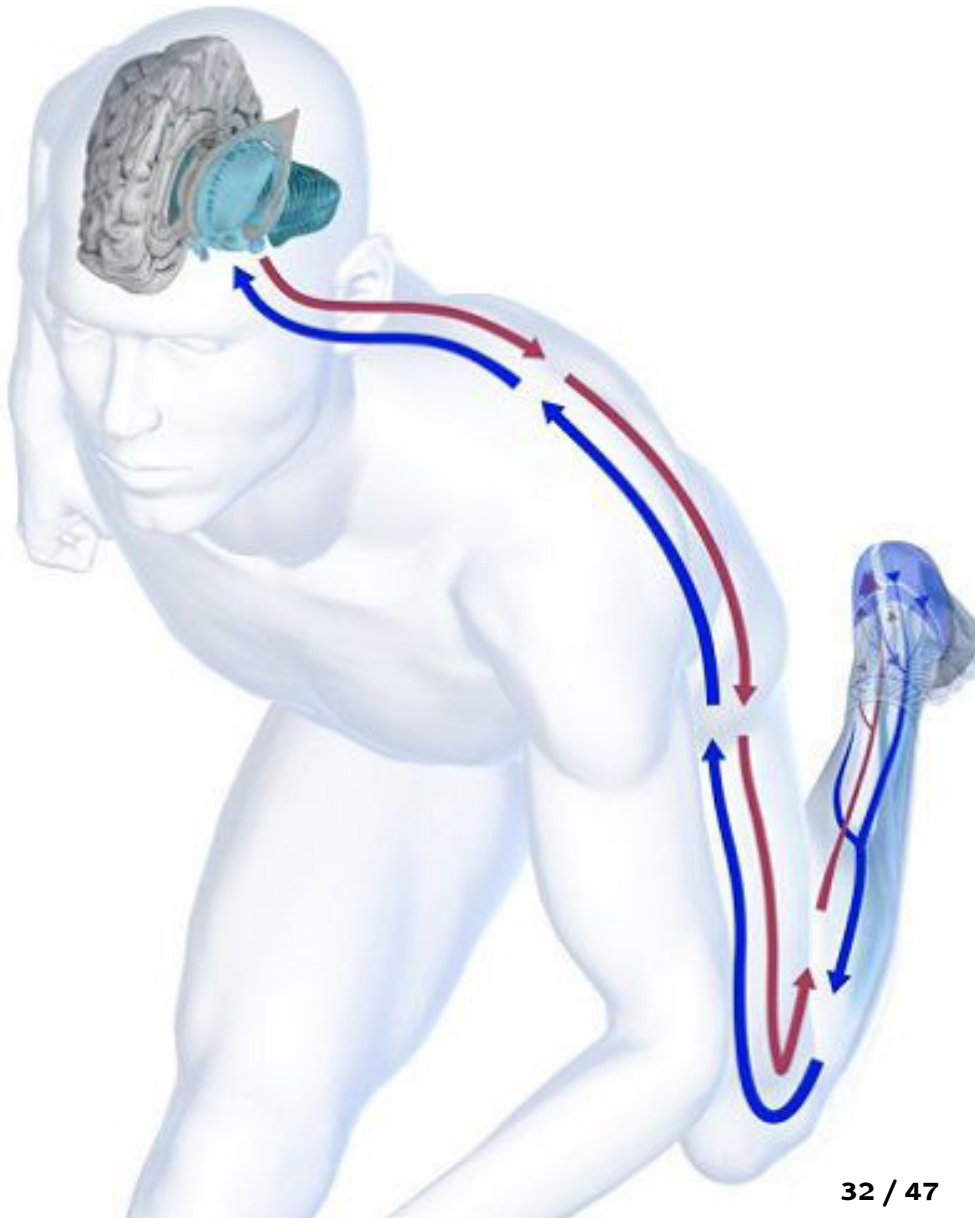
We learn to perform different tasks.

How do we control and learn our motor movements?



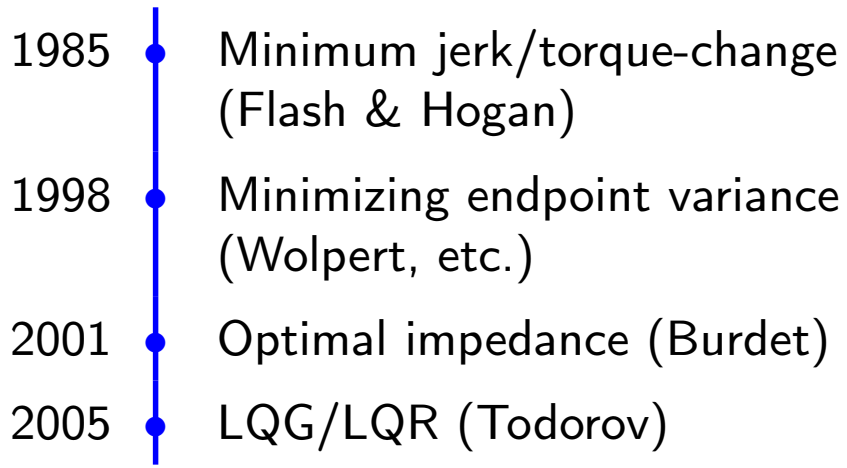
Human sensorimotor control

- Two key components: the central nervous system (CNS) and the limbs (muscles).
- Motor control is a feedback control problem.



Computational mechanisms in human motor control

- The CNS uses optimal control policy.
- Many models have been developed to support this theory.
- System identification + model-based optimal control.



Human arm movement model

Human arm movement model (Liu and Todorov, 2007):

$$
\begin{aligned}
dp &= v dt, \\
m dv &= (a - bv + f) dt, \\
\tau da &= (u - a) dt + d\zeta,
\end{aligned}
$$

where $d\zeta$ is the signal-dependent noise; f is generated by the divergent field (DF).

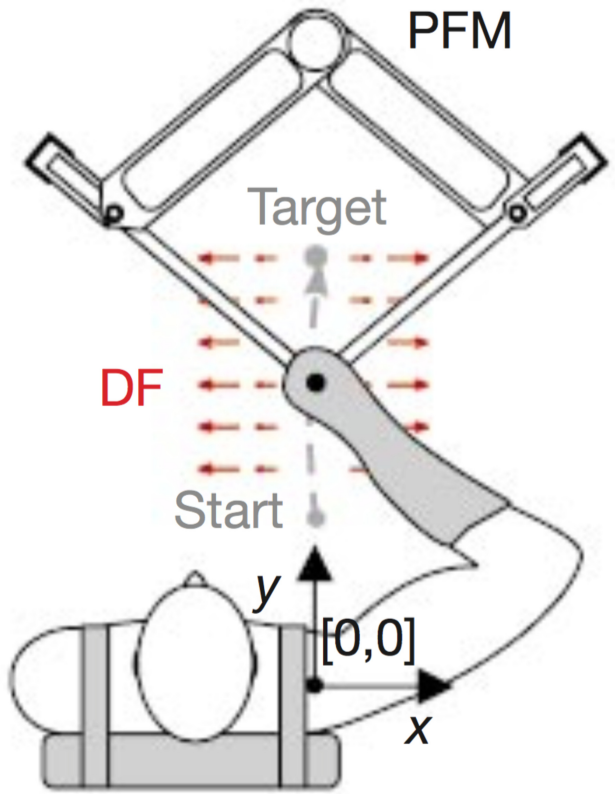
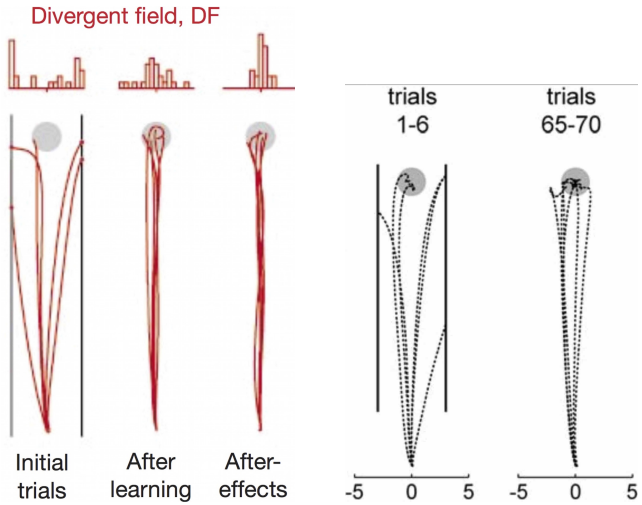
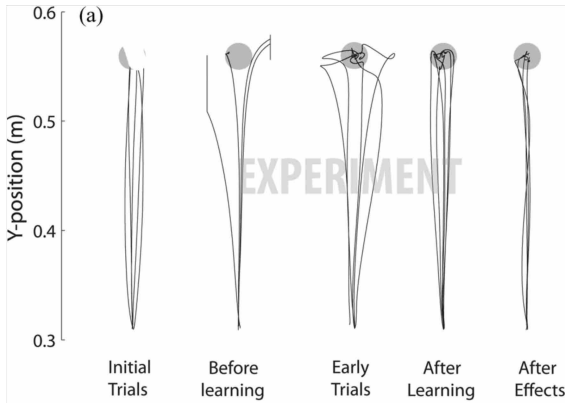


Figure: Experiment setup (Burdet2001).

ADP-based motor learning

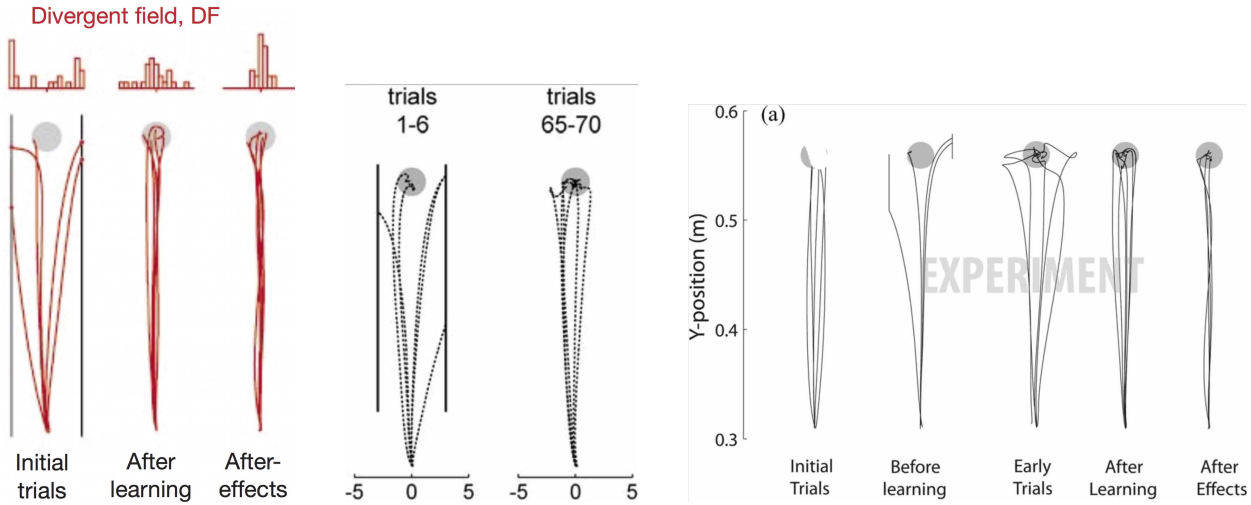
Experiment results (Burdet 2001; Franklin 2003; Zhou, et.al., 2012):



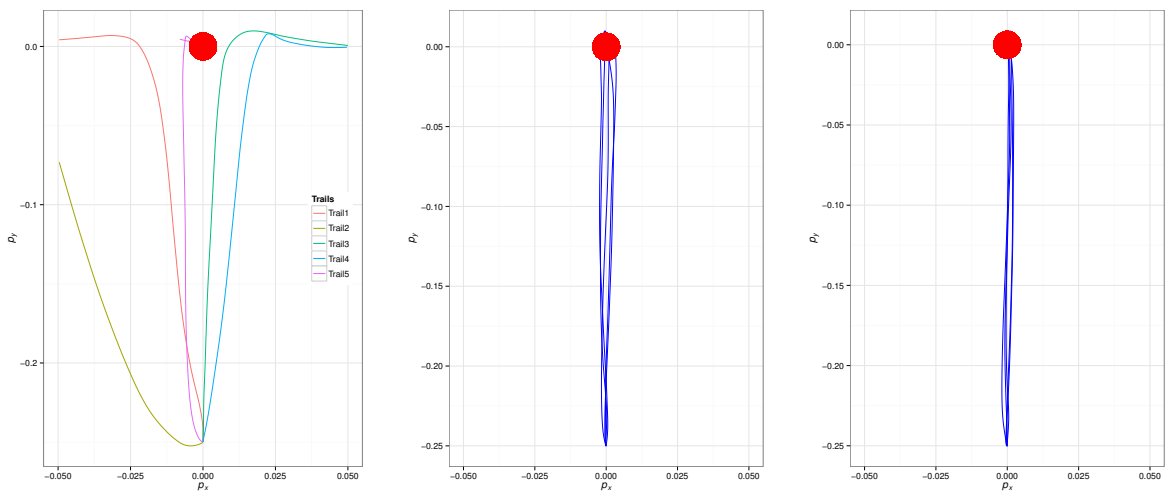


ADP-based motor learning

Experiment results (Burdet 2001; Franklin 2003; Zhou, et.al., 2012):



ADP simulation[8]:



(a) First five trials in the DF.

(b) Five independent trials after ADP learning in the DF.

(c) Five independent trials after ADP learning in the NF.

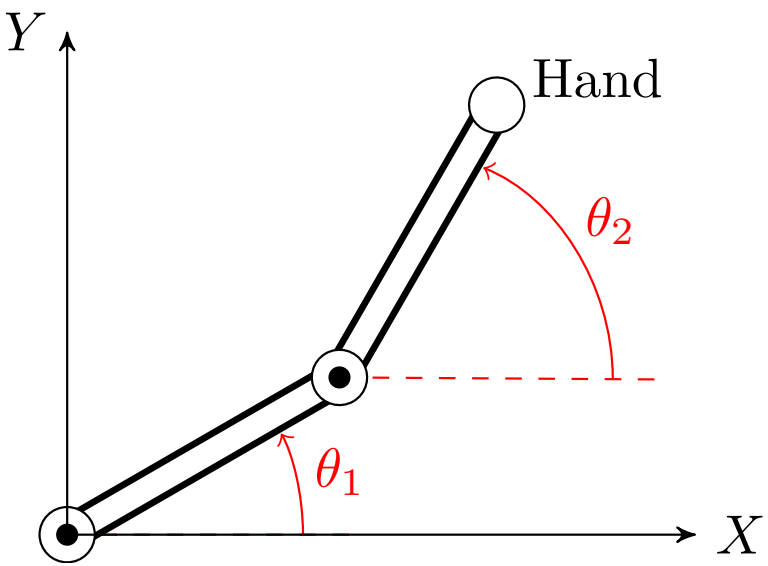
[8] T. Bian and Z. P. Jiang. In *Proc. of WCICA 2016*. (Best biomedical paper award)

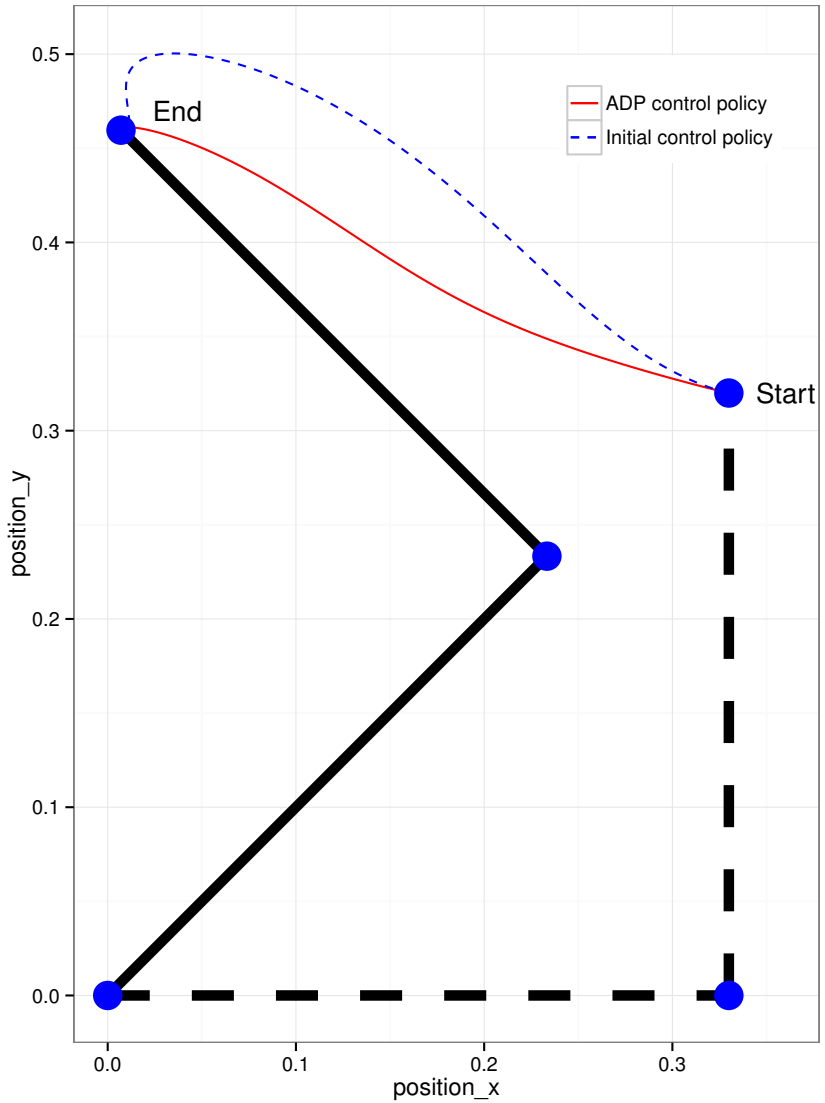
Nonlinear human sensorimotor control

Human sensorimotor control–nonlinear model:

$$T_s = D_s(\theta)\ddot{\theta} + C_s(\theta, \dot{\theta})\dot{\theta}.$$

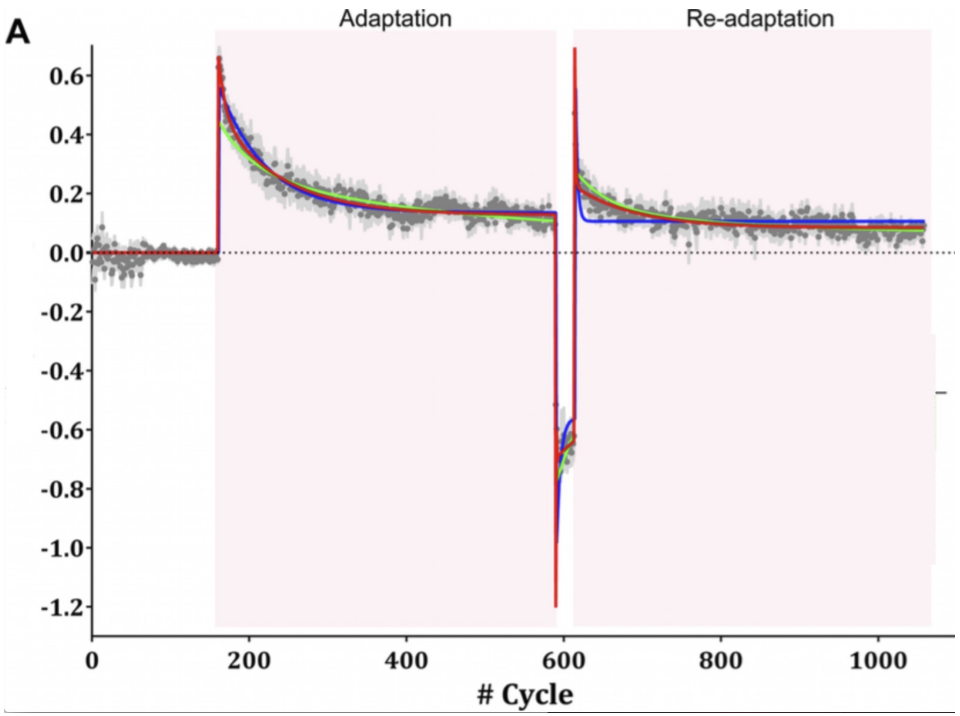
$$x = (\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2), \quad u \in \mathbb{R}^2.$$





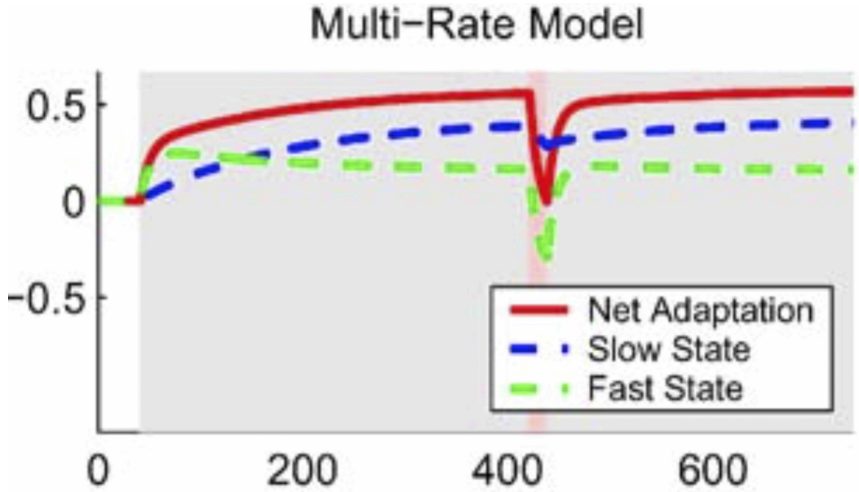
Savings

Savings: Prior learning to speed subsequent relearning even after behavioral manifestations of the prior learning have been washed out.



ADP learning and savings[9]

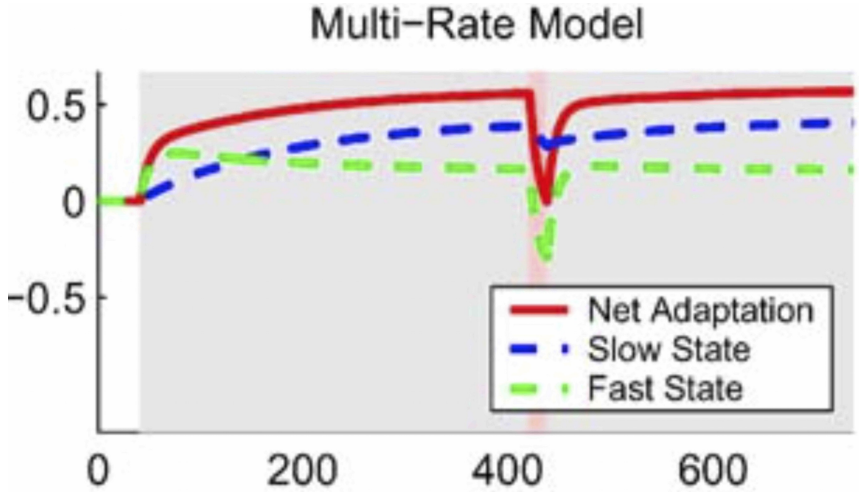
Multi-rate model (Smith, 2006).



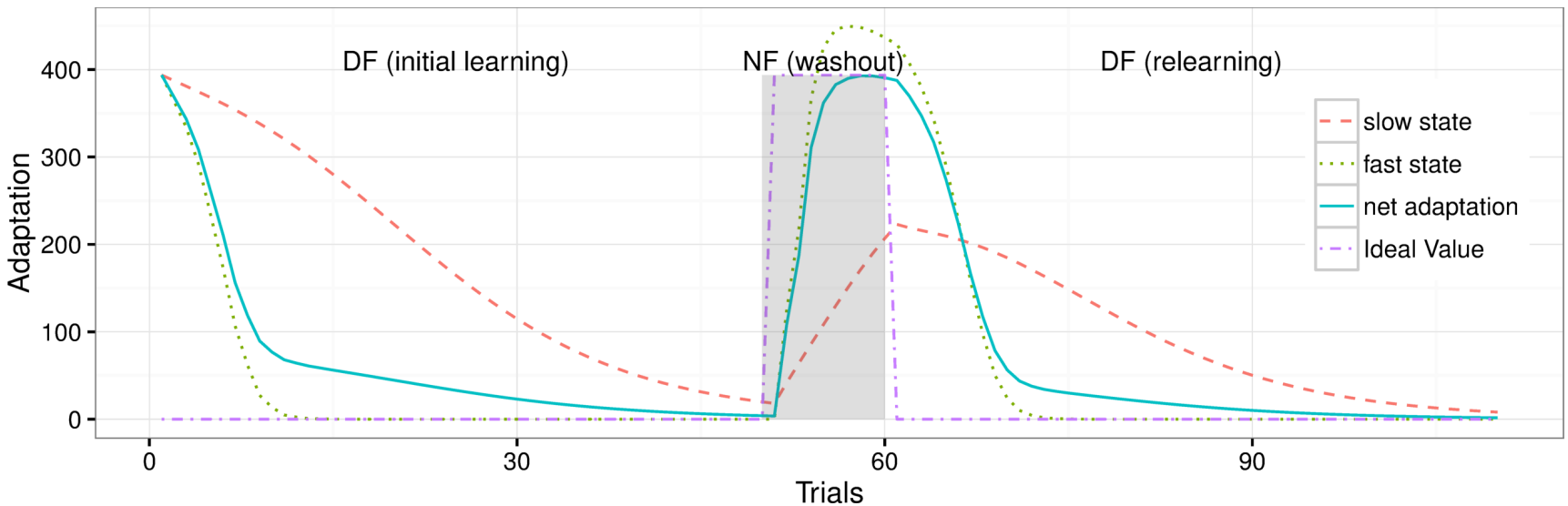
[9] T. Bian, D. M. Wolpert, and Z. P. Jiang. In *SfN 2016*.

ADP learning and savings[9]

Multi-rate model (Smith, 2006).



Multi-rate model based on ADP learning:



[9] T. Bian, D. M. Wolpert, and Z. P. Jiang. In *SfN 2016*.

ADP for video games

Outline

1. Policy iteration (PI)-based ADP for nonaffine systems.
   - T. Bian, Y. Jiang, and Z. P. Jiang, "Adaptive dynamic programming and optimal control of nonlinear nonaffine systems," *Automatica*, vol. 50, pp. 2624–2632, 10 2014.

1. Policy iteration (PI)-based ADP for nonaffine systems.
   - T. Bian, Y. Jiang, and Z. P. Jiang, "Adaptive dynamic programming and optimal control of nonlinear nonaffine systems," *Automatica*, vol. 50, pp. 2624–2632, 10 2014.
2. ADP for continuous-time stochastic optimal control.
   - T. Bian and Z. P. Jiang, "Adaptive dynamic programming for stochastic systems with state and control dependent noise," *IEEE Transactions on Automatic Control*, vol. 61, no 12, pp. 4170–4175, 2016.
   - T. Bian and Z. P. Jiang, Stochastic adaptive dynamic programming for robust optimal control design. In K. G. Vamvoudakis and S. Jagannathan (Eds.), *Control of Complex Systems: Theory and Applications*, chapter 7, pp. 211–245. Butterworth-Heinemann, Cambridge, MA, 2016.

1. Policy iteration (PI)-based ADP for nonaffine systems.
   - T. Bian, Y. Jiang, and Z. P. Jiang, "Adaptive dynamic programming and optimal control of nonlinear nonaffine systems," *Automatica*, vol. 50, pp. 2624–2632, 10 2014.
2. ADP for continuous-time stochastic optimal control.
   - T. Bian and Z. P. Jiang, "Adaptive dynamic programming for stochastic systems with state and control dependent noise," *IEEE Transactions on Automatic Control*, vol. 61, no 12, pp. 4170–4175, 2016.
   - T. Bian and Z. P. Jiang, Stochastic adaptive dynamic programming for robust optimal control design. In K. G. Vamvoudakis and S. Jagannathan (Eds.), *Control of Complex Systems: Theory and Applications*, chapter 7, pp. 211–245. Butterworth-Heinemann, Cambridge, MA, 2016.
3. Stochastic robust nonlinear control design.
   - T. Bian and Z. P. Jiang, "A tool for the global stabilization of stochastic nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 62, no 4, pp. 1946–1951, 2017.
   - T. Bian and Z. P. Jiang, "New results in global stabilization for stochastic nonlinear systems," *Control Theory and Technology*, vol. 14, no 1, pp. 57–67, 2016.

1. Policy iteration (PI)-based ADP for nonaffine systems.
   - T. Bian, Y. Jiang, and Z. P. Jiang, "Adaptive dynamic programming and optimal control of nonlinear nonaffine systems," *Automatica*, vol. 50, pp. 2624–2632, 10 2014.
2. ADP for continuous-time stochastic optimal control.
   - T. Bian and Z. P. Jiang, "Adaptive dynamic programming for stochastic systems with state and control dependent noise," *IEEE Transactions on Automatic Control*, vol. 61, no 12, pp. 4170–4175, 2016.
   - T. Bian and Z. P. Jiang, Stochastic adaptive dynamic programming for robust optimal control design. In K. G. Vamvoudakis and S. Jagannathan (Eds.), *Control of Complex Systems: Theory and Applications*, chapter 7, pp. 211–245. Butterworth-Heinemann, Cambridge, MA, 2016.
3. Stochastic robust nonlinear control design.
   - T. Bian and Z. P. Jiang, "A tool for the global stabilization of stochastic nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 62, no 4, pp. 1946–1951, 2017.
   - T. Bian and Z. P. Jiang, "New results in global stabilization for stochastic nonlinear systems," *Control Theory and Technology*, vol. 14, no 1, pp. 57–67, 2016.
4. ADP-based decentralized optimal control, with application to power systems.
   - T. Bian, Y. Jiang, and Z. P. Jiang, "Decentralized adaptive optimal control of large-scale systems with application to power systems," *IEEE Transactions on Industrial Electronics*, vol. 62, pp. 2439–2447, April 2015.

Outline

Conclusions

Main contributions:

- Developed the first continuous-time value iteration (VI).
- Proposed a new data-driven control design based on VI and adaptive dynamic programming (ADP).
- Application to human biological motor control.

Conclusions

Main contributions:

- Developed the first continuous-time value iteration (VI).
- Proposed a new data-driven control design based on VI and adaptive dynamic programming (ADP).
- Application to human biological motor control.

Other contributions:

- Developed ADP for nonaffine systems on the basis of policy iteration (PI).
- Investigated the stochastic optimal control problems using continuous-time ADP.
- Studied stochastic robust nonlinear control design.
- Applied decentralized adaptive optimal control to power systems.

Future works

- More efficient numerical methods to solve HJB equation (using on-line data).
- Global ADP for general nonlinear systems.
- ADP for nonlinear stochastic systems.

Acknowledgement

- Control and Networks (CAN) Lab:
  Ashish Gupta; Eric Mauro; Ye Lin; Manuel Serrano Rebuelta...
- Collaborators:
  Dr. Yu Jiang (The MathWorks, Inc.)
  Prof. D. M. Wolpert (The University of Cambridge)
- Center for Advanced Technology in Telecommunications (CATT)
- National Science Foundation (NSF)

Partial list of publications

Accepted journal papers:

T. Bian, Y. Jiang, and Z. P. Jiang (2014). Adaptive dynamic programming and optimal control of nonlinear nonaffine systems. *Automatica 50*(10), 2624–2632.

T. Bian, Y. Jiang, and Z. P. Jiang (2015). Decentralized adaptive optimal control of large-scale systems with application to power systems. *IEEE Transactions on Industrial Electronics 62*(4), 2439–2447.

T. Bian, Y. Jiang, and Z. P. Jiang (2016b). Adaptive dynamic programming for stochastic systems with state and control dependent noise. *IEEE Transactions on Automatic Control 61*(12), 4170–4175.

T. Bian and Z. P. Jiang (2016e). Value iteration and adaptive dynamic programming for data-driven adaptive optimal control design. *Automatica 71*, 348–360.

T. Bian and Z. P. Jiang (2016a). New results in global stabilization for stochastic nonlinear systems. *Control Theory and Technology 14*(1), 57–67.

T. Bian and Z. P. Jiang (2017). A tool for the global stabilization of stochastic nonlinear systems. *IEEE Transactions on Automatic Control 62*(4), 1946–1951.

Book Chapter:

T. Bian and Z. P. Jiang (2016c). Stochastic adaptive dynamic programming for robust optimal control design. In K. G. Vamvoudakis and S. Jagannathan (Eds.), *Control of Complex Systems: Theory and Applications*. Chapter 7, Cambridge, MA: Butterworth-Heinemann.

Thank you for your attention!

Consider the following linearized model (Wang and Hill, 1993):

$$\begin{aligned}\dot{\delta} &= \omega,\\ \dot{\omega} &= -\frac{D}{2H}\omega + \frac{\omega_0}{2H}(P_{m0} - P_e),\\ \dot{P}_e &= -\frac{1}{T'_{d0}}P_e + \frac{1}{T'_{d0}}v_f.\end{aligned}$$

We have the state-space equation with

$$x = \begin{bmatrix}\delta - \delta_0\\ \omega\\ P_e - P_{m0}\end{bmatrix}, \quad u = v_f - P_{m0}, \quad A = \begin{bmatrix}0 & 1 & 0\\ 0 & -\frac{D}{2H} & -\frac{\omega_0}{2H}\\ 0 & 0 & -\frac{1}{T'_{d0}}\end{bmatrix}, \quad B = \begin{bmatrix}0\\ 0\\ \frac{1}{T'_{d0}}\end{bmatrix}.$$

The operating point is $\delta_0 = 47^\circ$ and $P_{m0} = 0.45$ p.u.

Weight matrices:

$$Q = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}, \quad R = 1.$$

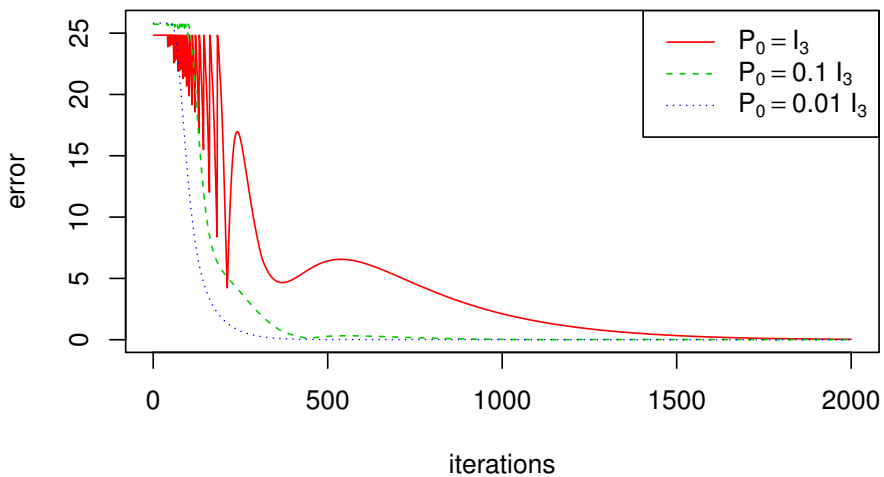


Figure: Error plot of $|P_k - P^*|$ for different P_0.

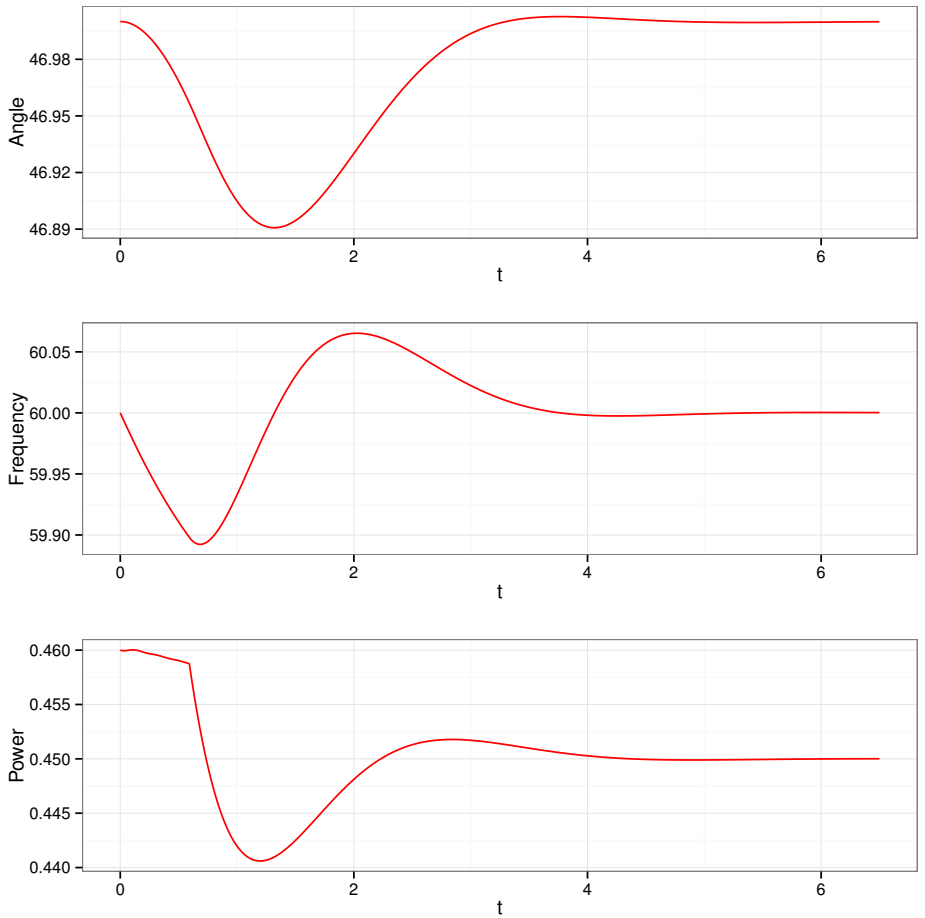


Figure: Trajectories of the generator.

Table: Error for different P_0.

P_0	$0.01I_3$	$0.1I_3$	I_3
$\lvert\hat{P}^* - P^*\rvert$ $(\times 10^{-4})$	3.57	9.43	6.10
CPU time (s)	0.935	1.098	3.382
# iterations	2778	3247	10094

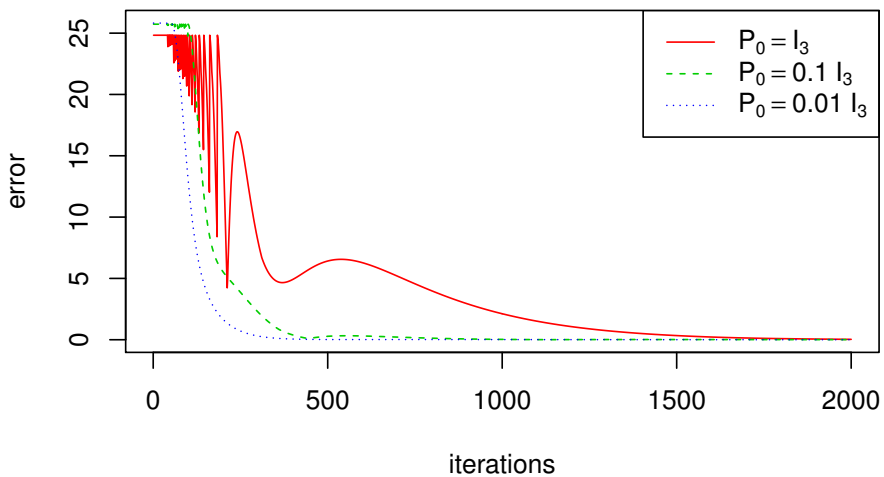


Figure: Error plot of $|P_k - P^*|$ for different P_0.

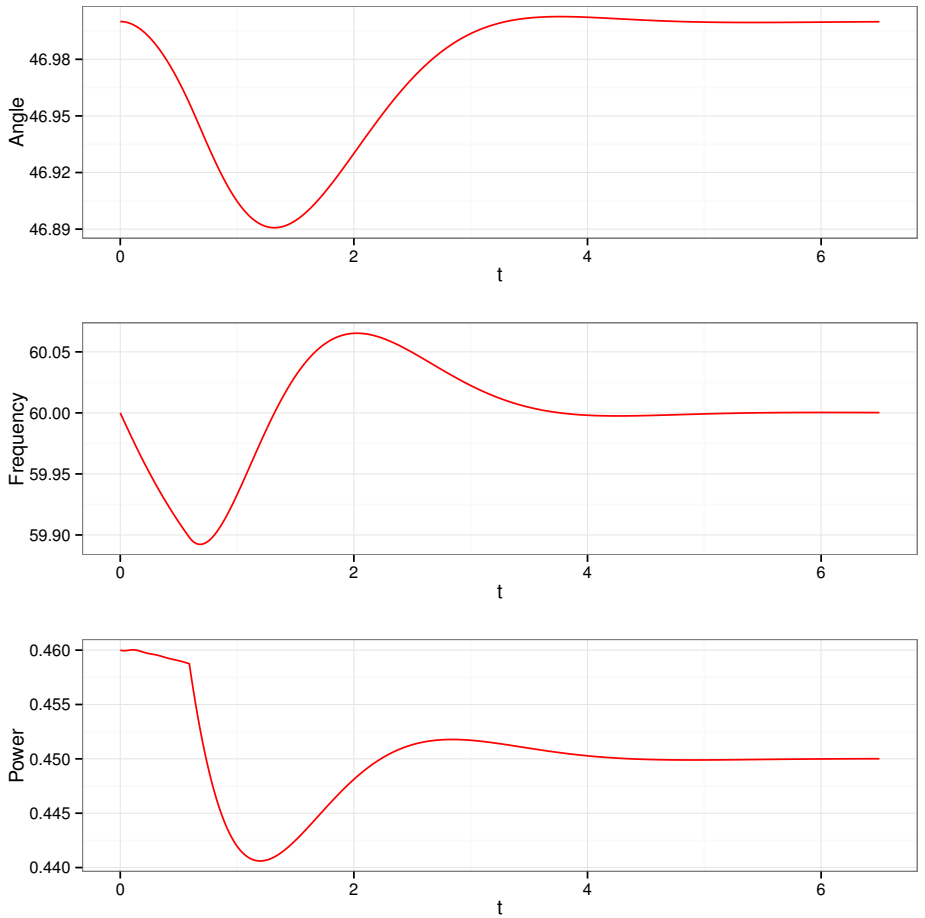


Figure: Trajectories of the generator.

Consider the following nonlinear polynomial system:

$$\dot{x}_1 = -\frac{1}{2}x_1^3 - x_1 - 2x_2,$$
$$\dot{x}_2 = \frac{1}{8}x_2^3 - x_2 + \frac{1}{2}u^3.$$
$$\mathcal{J} = \int_0^\infty \left(x_1^4 + 2(x_1 + x_2)^2 + \frac{3}{4}u^4\right) dt.$$

The obtained cost and control policy are

$$\hat{V} = 1.0004x_1^2 + 0.0008x_1x_2 + 1.0009x_2^2,$$
$$\hat{\mu} = -0.0013x_1 - 1.0000x_2.$$

Optimal solution:

$$V^*(x_1, x_2) = x_1^2 + x_2^2, \quad \mu^*(x_1, x_2) = -x_2.$$

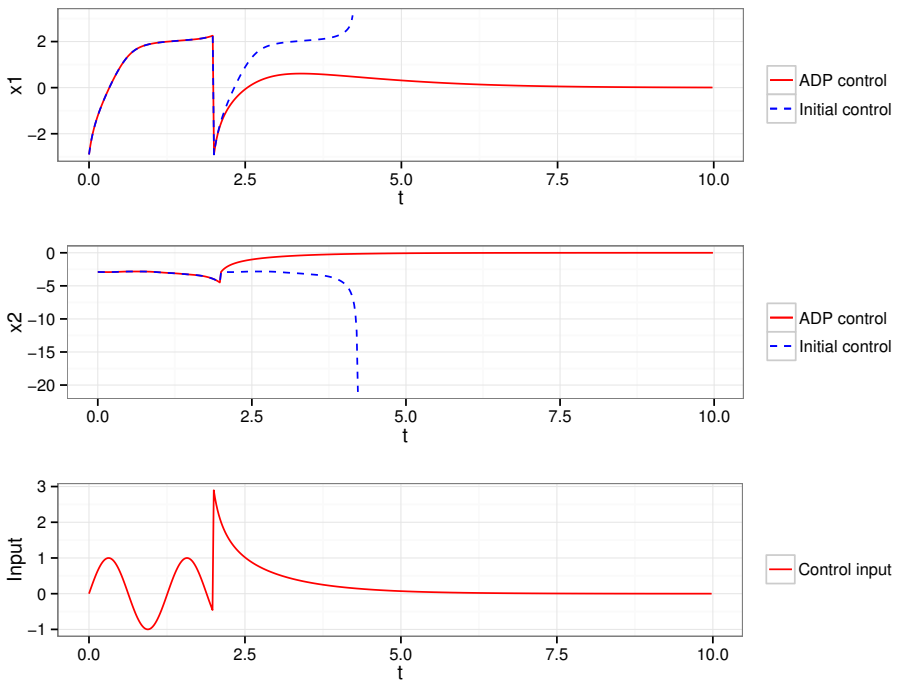


Figure: Plots of the closed-loop solutions x_1 and x_2.

DMRE:

$$\begin{aligned}\dot{P} &= A^T P + PA - PBR^{-1}B^T P + Q \\ &= (A - BK)^T P + P(A - BK) + K^T RK + Q \qquad K = R^{-1}B^T P \\ &= (A - BK^*)^T P + P(A - BK^*) + (K^*)^T RK^* + Q - (K - K^*)^T R(K - K^*) \\ \Delta\dot{P} &= (A - BK^*)^T \Delta P + \Delta P(A - BK^*) - (\Delta P)^T BR^{-1}B^T \Delta P.\end{aligned}$$

$\Delta P = P - P^*$.

Policy evaluation:

$$\begin{aligned}\dot{\hat{P}} &= (A - BK^*)^T \hat{P} + \hat{P}(A - BK^*) + (K^*)^T RK^* + Q \qquad \hat{P} \to P^* \\ \Delta\dot{\hat{P}} &= (A - BK^*)^T \Delta\hat{P} + \Delta\hat{P}(A - BK^*).\end{aligned}$$

$\Delta\hat{P} = \hat{P} - P^*$.

Small-gain theory: If $\dot{x} = f(x)$ is AS at 0 and has finite gain, and $\gamma(\cdot)$ has a "small" gain, then $\dot{x} = f(x) + \gamma(x)$ is AS at 0.

Then,

$$\dot{P} = A^T P + PA - PBR^{-1}B^T P + Q + \gamma(\Delta P)$$

converges to P^* under small-gain condition.

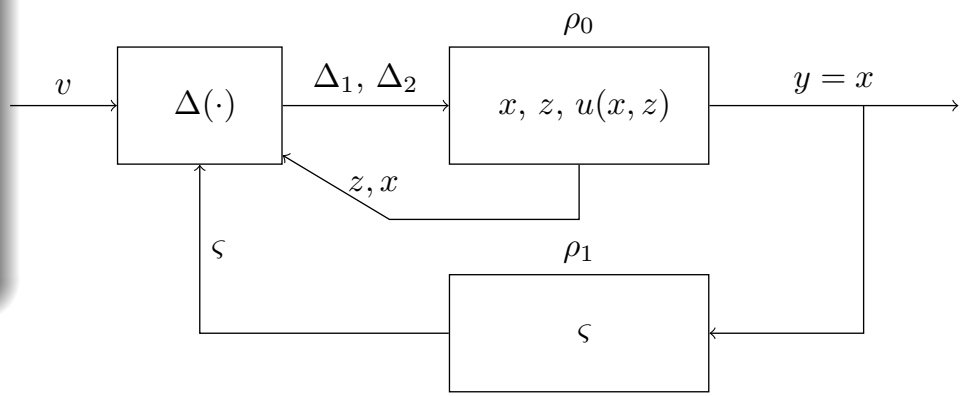
Gain of (x, z) subsystem

Lemma 1

We can design a robust optimal controller, such that the closed-loop (x, z)-subsystem admits a linear L^2 gain from Δ to x:

$$\rho_0 \leq \sqrt{a_1\gamma_1^2 + a_2\gamma_2^2}, \quad a_1, a_2 \geq 0,$$

where $\gamma_i > 0$, $i = 1, 2$.



ρ_0 can be made arbitrarily small by choosing small γ_i.

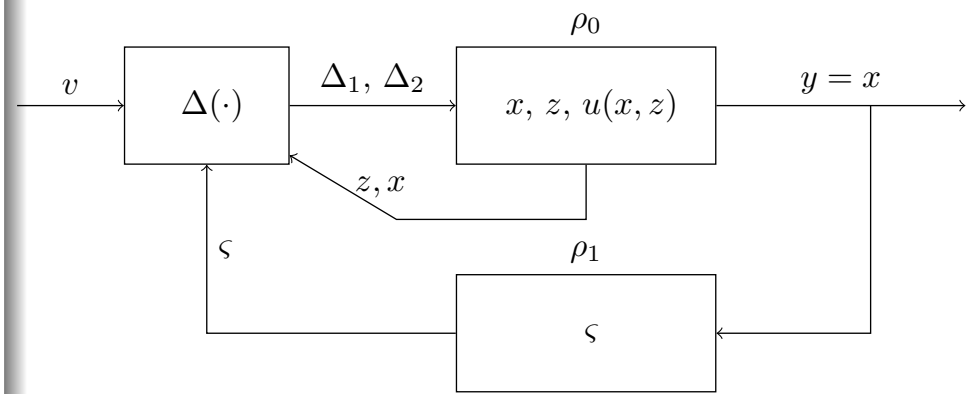
Gain assignment for the whole system:

Theorem 1

If $\rho_0^2\rho_1^2 C_1 + \rho_0^2 C_2 < 1$, then we can design a robust optimal controller, such that the interconnected system

- is GAS at the origin when $v = 0$; and
- admits a linear L^2 gain from v to y .

$$\rho = \sqrt{\frac{(1+\rho_1^2)\rho_0^2 C_3}{1 - C_1\rho_0^2\rho_1^2 - C_2\rho_0^2}}.$$



- ς subsystem undermines the robustness of the interconnected system;
- ρ can be made arbitrarily small by choosing ρ_0.

Phase 1:

$$dx = Axdt + B\left((z+\Delta_1)dt + \sum_{i=1}^{q} A_{0i}xdw_i\right),\ \mathcal{J}^{(1)} = \int_0^\infty (x^TQx + \lambda_1|z|^2 - \gamma_1|\Delta_1|^2)dt.$$

Phase 2: Denote $\xi = z - z^*$.

$$d\xi = (\bar{F}\xi + \bar{E}x)dt + G\left((u+\bar{\Delta}_2)dt + \sum_{i=1}^{q}(\bar{E}_{0i}x + F_{0i}z)dw_i\right),\ \mathcal{J}^{(2)} = \int_0^\infty (\xi^TW\xi + \lambda_2|u|^2 - \gamma_2|\bar{\Delta}_2$$

In each phase, we use VI-based RADP:

$$\text{Phase 1}(x) \Longrightarrow (\hat{J}^{(1)*}, \hat{z}^*) \Longrightarrow \text{Phase 2}(\hat{\xi}) \Longrightarrow (\hat{J}^{(2)*}, \hat{u}^*).$$

Gain assignment

There exist $\hat{u}^*$ derived from the RADP Algorithm, such that the interconnected system admits the following L^2 gain from v to x:

$$\hat{\rho} = \sqrt{\frac{(1+\alpha_3)\hat{\rho}_0^2 C_3}{1 - C_1\alpha_3\hat{\rho}_0^2 - C_2\hat{\rho}_0^2}}, \quad \hat{\rho}_0 > \rho_0.$$