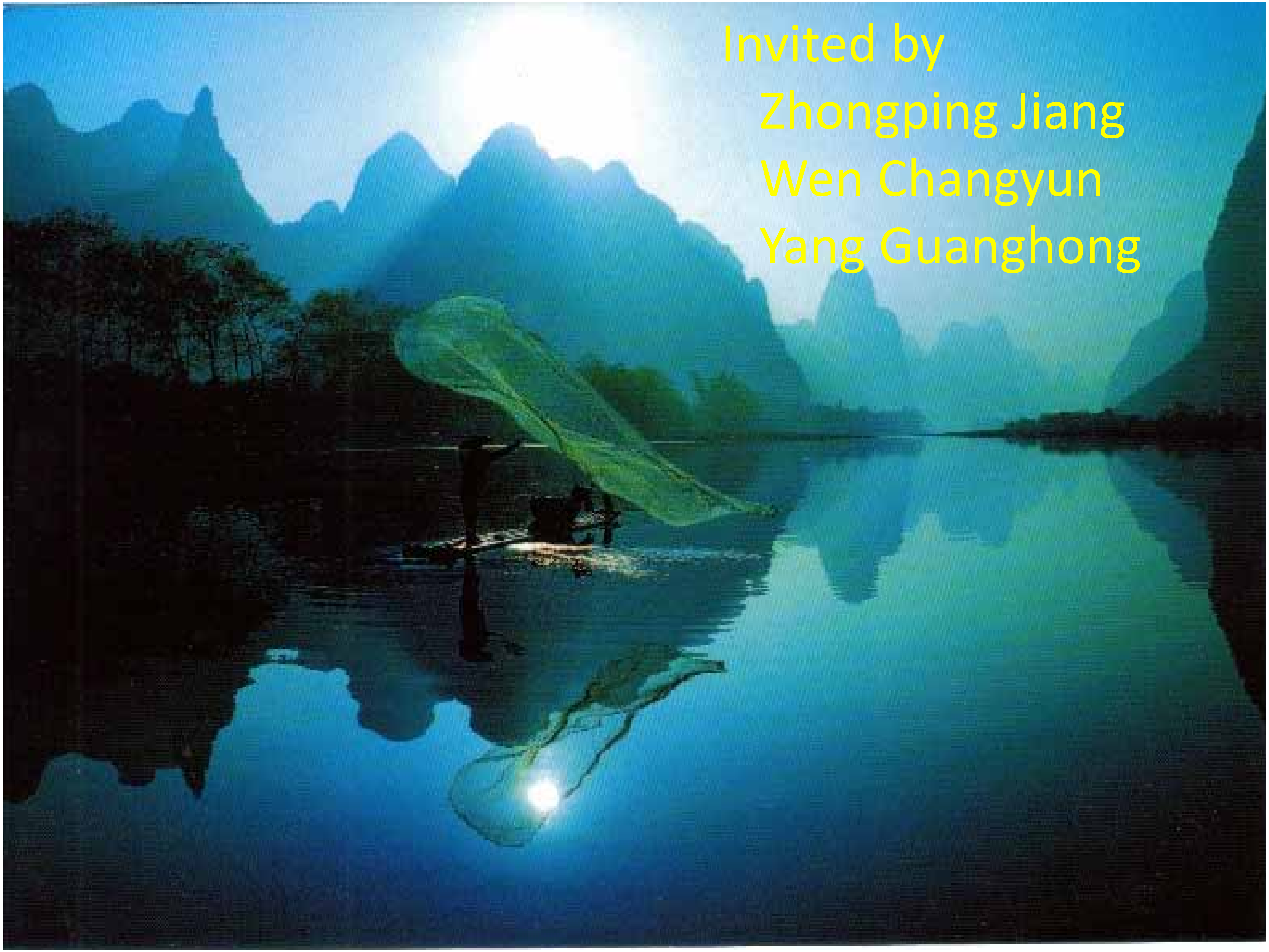
F.L. Lewis
National Academy of Inventors

Moncrief-O'Donnell Chair, UTA Research Institute (UTARI)
The University of Texas at Arlington, USA
and
Qian Ren Consulting Professor, State Key Laboratory of Synthetical Automation for Process Industries
Northeastern University, Shenyang, China

# New Developments in Integral Reinforcement Learning: Continuous-time Optimal Control and Games


Invited by
Zhongping Jiang
Wen Changyun
Yang Guanghong

**New Research Results**

Integral Reinforcement Learning for Online Optimal Control
IRL for Online Solution of Multi-player Games
Multi-Player Games on Communication Graphs
Off-Policy Learning
Experience Replay
Bio-inspired Multi-Actor Critics
Output Synchronization of Heterogeneous MAS

**Applications to:**

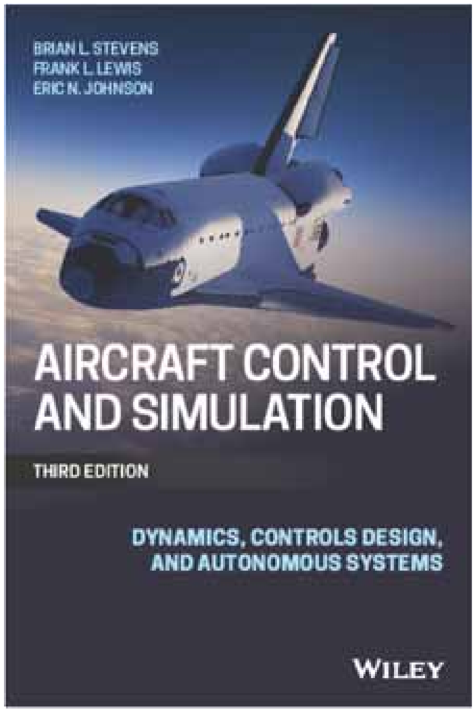
Microgrid
Robotics
Industry Process Control

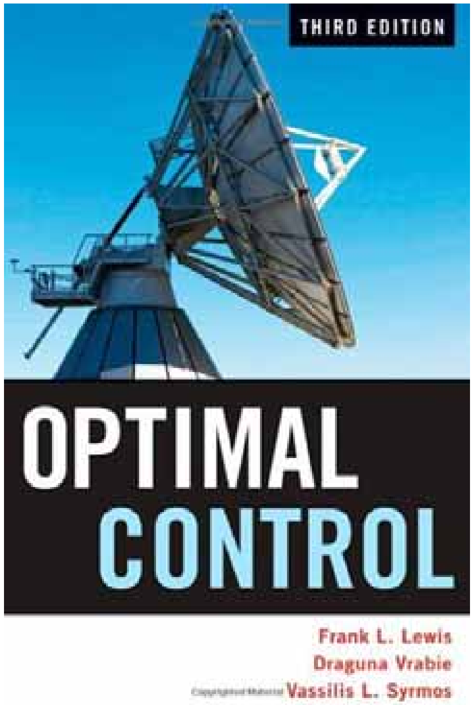
# Optimality and Games

Optimal Control is Effective for:
- Aircraft Autopilots
- Vehicle engine control
- Aerospace Vehicles
- Ship Control
- Industrial Process Control

Multi-player Games Occur in:
- Networked Systems Bandwidth Assignment
- Economics
- Control Theory disturbance rejection
- Team games
- International politics
- Sports strategy

But, optimal control and game solutions are found by
- Offline solution of Matrix Design equations
- A full dynamical model of the system is needed

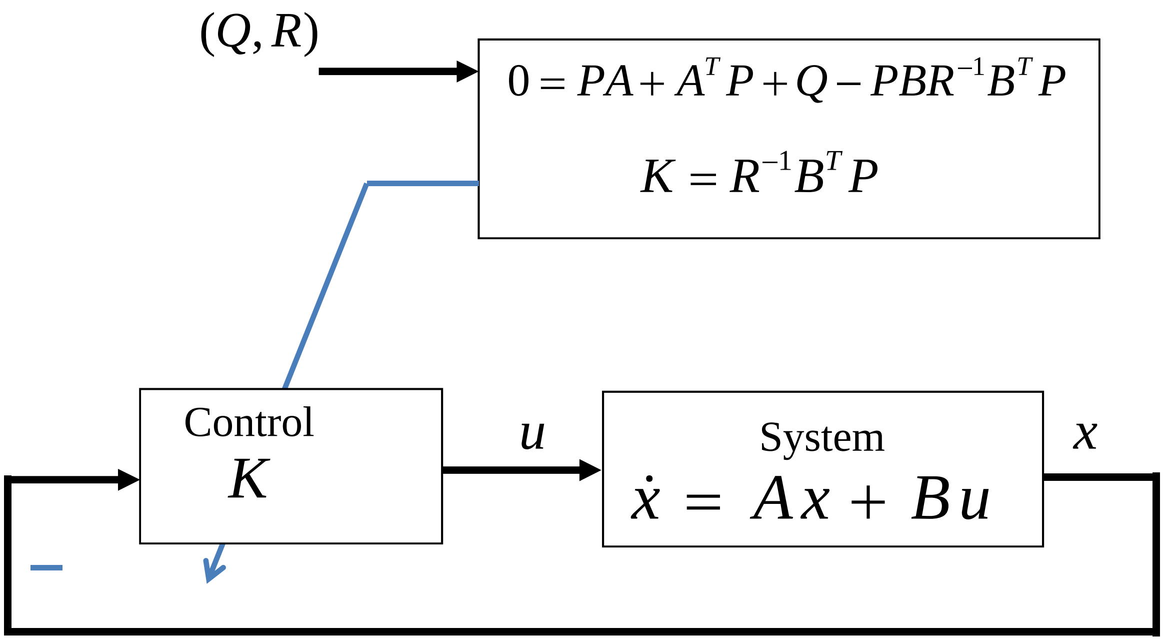


# Optimal Control- The Linear Quadratic Regulator (LQR)

User prescribed optimization criterion

$$V(x(t)) = \int_t^{\infty} (x^T Q x + u^T R u)\, d\tau$$




Off-line Design Loop
Using ARE

On-line real-time
Control Loop

An Offline Design Procedure
that requires Knowledge of system dynamics model (A,B)

System modeling is expensive, time consuming, and inaccurate

Adaptive Control is online and works for unknown systems.
Generally not Optimal

Optimal Control is off-line,
and needs to know the system dynamics to solve design eqs.

We want to find optimal control solutions
Online in real-time
Using adaptive control techniques
Without knowing the full dynamics

For nonlinear systems and general performance indices

Bring together Optimal Control and Adaptive Control

Reinforcement Learning turns out to be the key to this!

# Optimality in Biological Systems

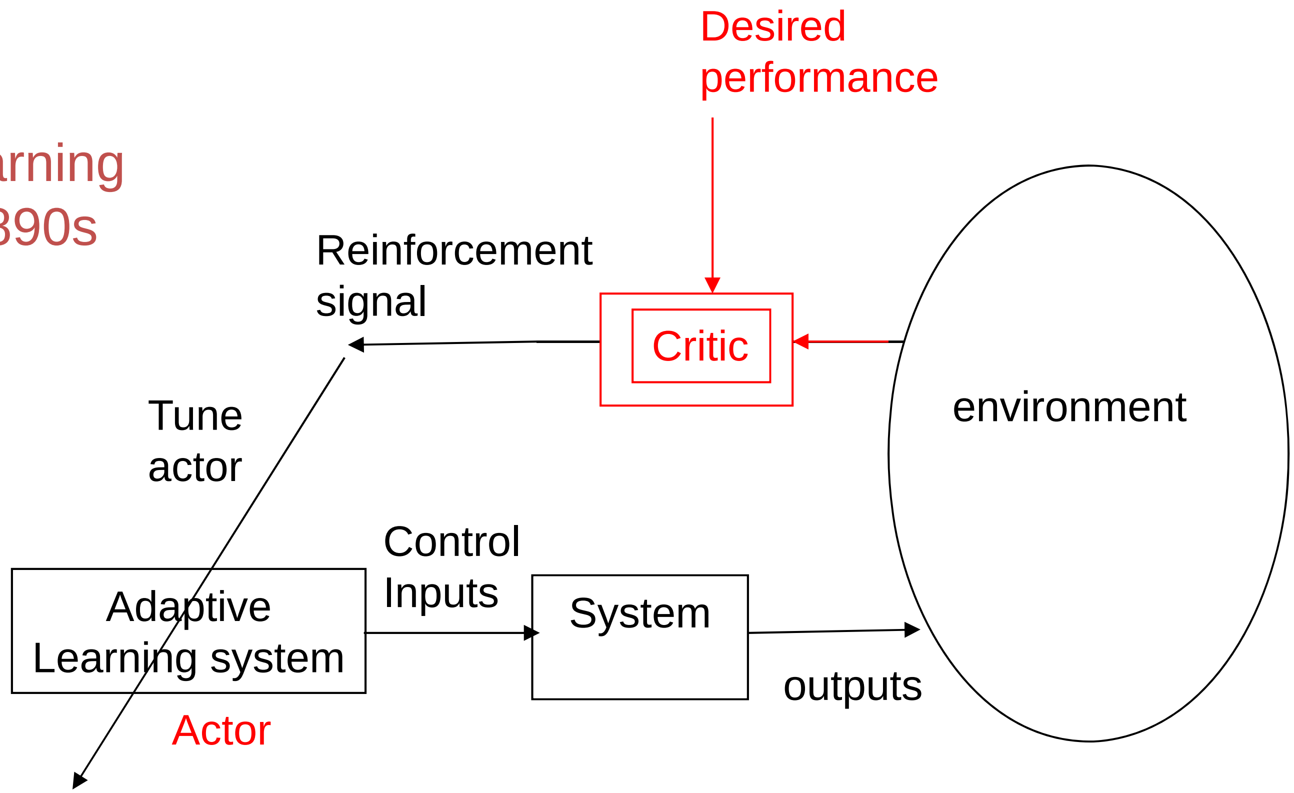
Every living organism improves its control actions based on rewards received from the environment

The resources available to living organisms are usually meager. Nature uses optimal control.

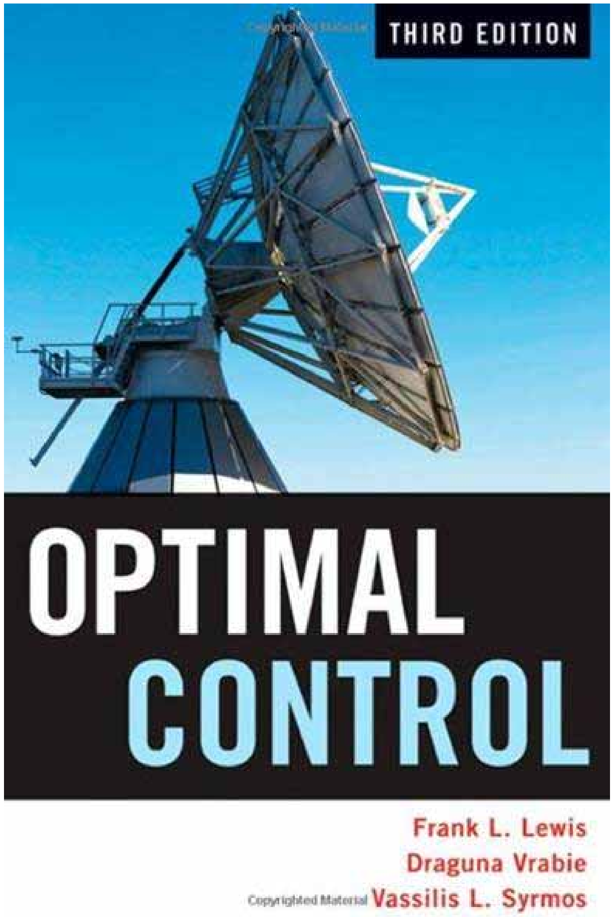
We want OPTIMAL performance
- ADP- Approximate Dynamic Programming

Actor-Critic Learning

Reinforcement learning
Ivan Pavlov 1890s



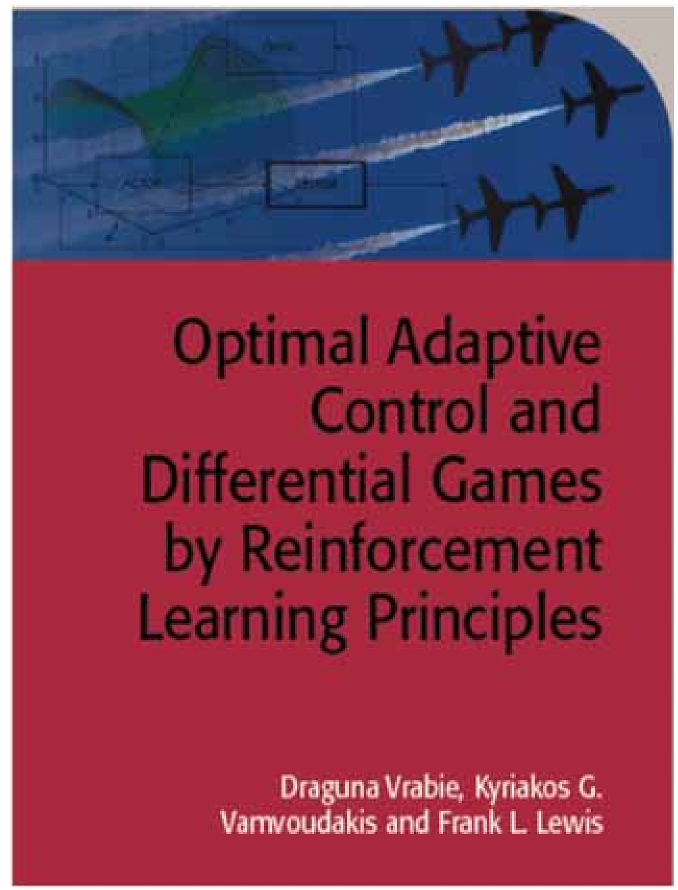
# Books



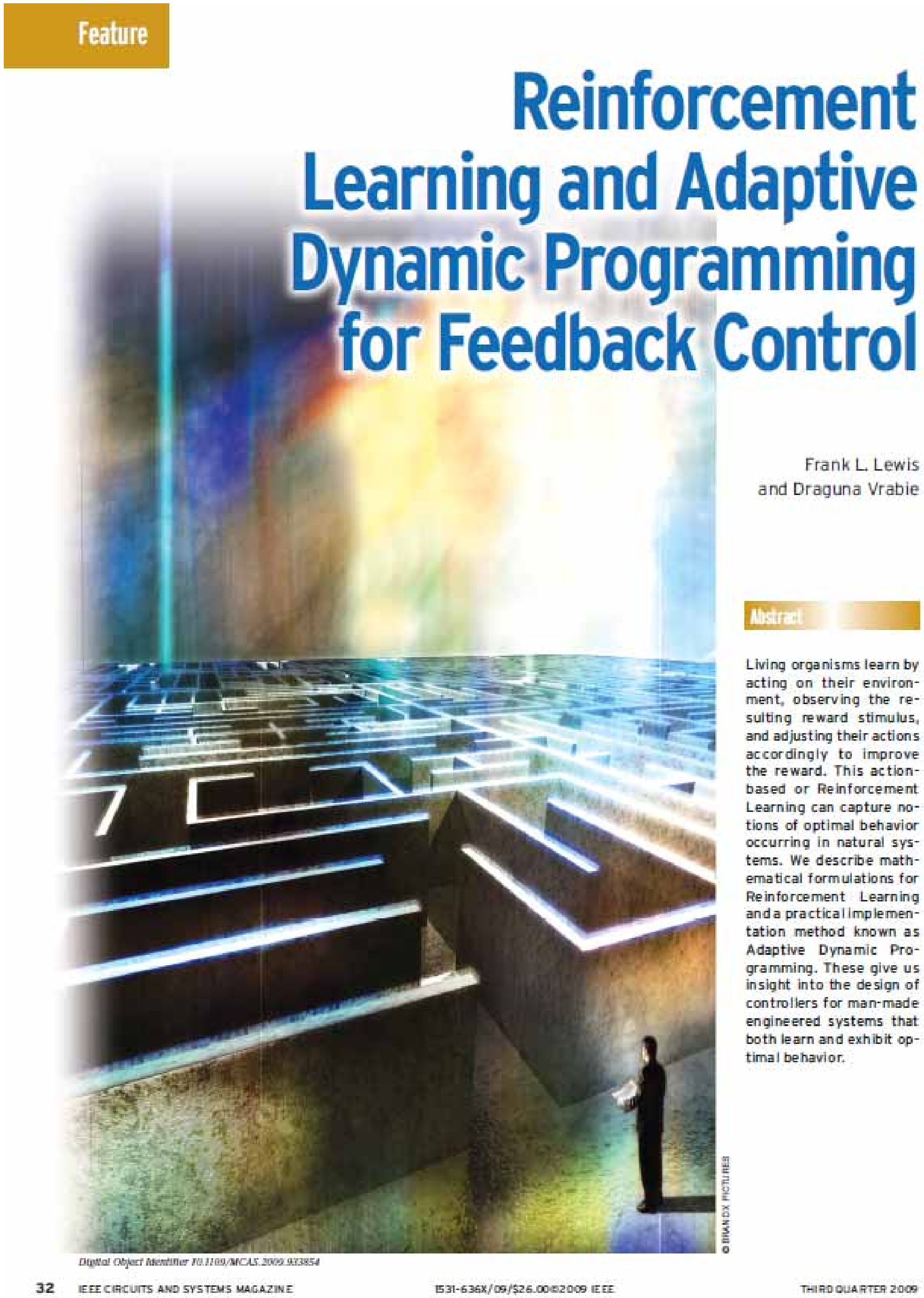

F.L. Lewis, D. Vrabie, and V. Syrmos,
*Optimal Control*, third edition, John Wiley and Sons, New York, 2012.

New Chapters on:

Reinforcement Learning
Differential Games




D. Vrabie, K. Vamvoudakis, and F.L. Lewis, *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*, IET Press, 2012.



Feature

# Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control

Frank L. Lewis
and Draguna Vrabie

Abstract

Living organisms learn by acting on their environment, observing the resulting reward stimulus, and adjusting their actions accordingly to improve the reward. This action-based or Reinforcement Learning can capture notions of optimal behavior occurring in natural systems. We describe mathematical formulations for Reinforcement Learning and a practical implementation method known as Adaptive Dynamic Programming. These give us insight into the design of controllers for man-made engineered systems that both learn and exhibit optimal behavior.

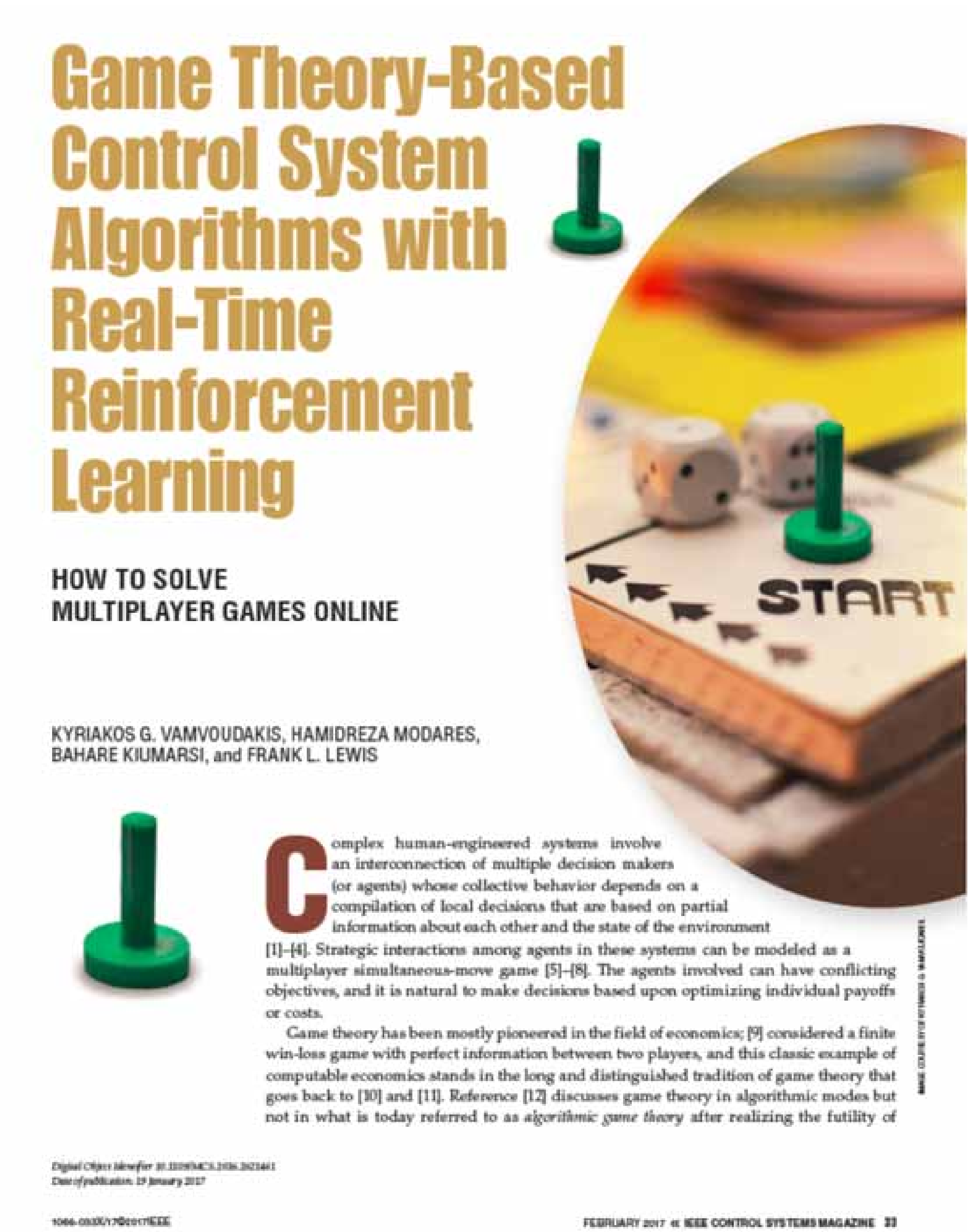


Digital Object Identifier 10.1109/MCAS.2009.933854



F.L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," IEEE Circuits & Systems Magazine, Invited Feature Article, pp. 32-50, Third Quarter 2009.

IEEE Control Systems Magazine, F. Lewis, D. Vrabie, and K. Vamvoudakis, "Reinforcement learning and feedback Control," Dec. 2012



# Game Theory-Based Control System Algorithms with Real-Time Reinforcement Learning

## HOW TO SOLVE MULTIPLAYER GAMES ONLINE

KYRIAKOS G. VAMVOUDAKIS, HAMIDREZA MODARES, BAHARE KIUMARSI, and FRANK L. LEWIS

Complex human-engineered systems involve an interconnection of multiple decision makers (or agents) whose collective behavior depends on a compilation of local decisions that are based on partial information about each other and the state of the environment [1]–[4]. Strategic interactions among agents in these systems can be modeled as a multiplayer simultaneous-move game [5]–[8]. The agents involved can have conflicting objectives, and it is natural to make decisions based upon optimizing individual payoffs or costs.

Game theory has been mostly pioneered in the field of economics; [9] considered a finite win-loss game with perfect information between two players, and this classic example of computable economics stands in the long and distinguished tradition of game theory that goes back to [10] and [11]. Reference [12] discusses game theory in algorithmic modes but not in what is today referred to as *algorithmic game theory* after realizing the futility of

*Digital Object Identifier 10.1109/MCS.2016.2621461*
*Date of publication: 19 January 2017*

1066-033X/17©2017IEEE



Multi-player Game Solutions
IEEE Control Systems Magazine, Feb. 2017

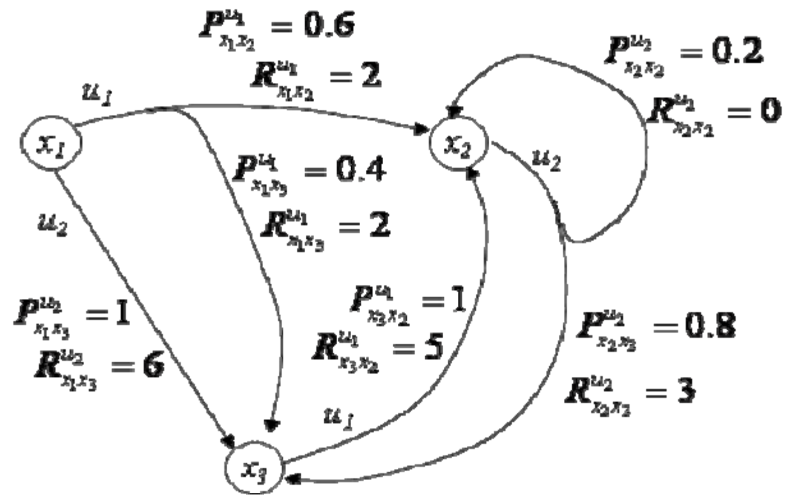
Bahare Kiumarsi, K. Vamvoudakis, H. Modares, and F.L. Lewis, “Optimal and Autonomous Control Using Reinforcement Learning: A Survey,” IEEE Trans. Neural Networks and Learning Systems, to appear 2018.

# RL for Markov Decision Processes $(X,U,P,R)$

$X$= states, $U$= controls

P= Probability of going to state x' from state x given that the control is u

R= Expected reward on going to state x' from state x given that the control is u




*Expected Value* of a policy $\pi(x,u)$

$$V_k^\pi(x) = E_\pi\{J_{k,T} \mid x_k = x\} = E_\pi\{\sum_{i=k}^{k+T}\gamma^{i-k} r_i \mid x_k = x\}$$

*Optimal control problem*

determine a policy $\pi(x,u)$ to minimize the expected future cost

*optimal policy* $\pi^*(x,u) = \arg\min_\pi V_k^\pi(s) = \arg\min_\pi E_\pi\{\sum_{i=k}^{k+T}\gamma^{i-k} r_i \mid x_k = x\}$.

*optimal value* $V_k^*(x) = \min_\pi V_k^\pi(x) = \min_\pi E_\pi\{\sum_{i=k}^{k+T}\gamma^{i-k} r_i \mid x_k = x\}$.

## Policy Iteration

Policy evaluation by Bellman eq. $V_j(x) = \sum_u \pi_j(x,u)\sum_{x'} P^u_{xx'}\left[R^u_{xx'} + \gamma V_j(x')\right]$ *for all* $x \in X$ .

Policy Improvement $\pi_{j+1}(x,u) = \arg\min_u \sum_{x'} P^u_{xx'}\left[R^u_{xx'} + \gamma V_j(x')\right]$ *for all* $x \in X$ .

Policy Evaluation equation is a system of *N* simultaneous linear equations, one for each state.

Policy Improvement makes $V^{\pi'}(x) \le V^\pi(x)$


R.S. Sutton and A.G. Barto, Reinforcement Learning– An Introduction, MIT Press, Cambridge, Massachusetts, 1998.
D.P. Bertsekas and J. N. Tsitsiklis, Neuro-Dynamic Programming, Athena Scientific, MA, 1996.
W.B. Powell, Approximate Dynamic Programming: Solving the Curses of Dimensionality, Wiley, New York, 2009.

## Discrete-Time Systems Optimal Adaptive Control

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

cost $V_h(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(x_i, u_i)$ Example $r(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$

Difference eq equivalent $V_h(x_k) = r(x_k, u_k) + \gamma \sum_{i=k+1}^{\infty} \gamma^{i-(k+1)} r(x_i, u_i)$

Bellman equation $V_h(x_k) = x_k^T Q x_k + u_k^T R u_k + \gamma V_h(x_{k+1})$

Hamiltonian $H(x_k, \nabla V(x_k), u_k) = r(x_k, u_k) + \gamma V_h(x_{k+1}) - V_h(x_k)$

System dynamics does not appear

## Continuous-time Systems Nonlinear Optimal Regulator

Nonlinear System dynamics $\dot{x} = f(x,u) = f(x) + g(x)u$

Cost/value $V(x(t)) = \int_t^{\infty} r(x,u)\, dt = \int_t^{\infty} (Q(x) + u^T R u)\, dt$

Leibniz gives
Differential equivalent

Bellman Equation, in terms of the Hamiltonian function

$$H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \dot{x} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \big( f(x) + g(x)u \big) + r(x,u) = 0$$

**STABILITY ?!**

Problem- System dynamics
shows up in Hamiltonian

# RL ADP has been developed for Discrete-Time Systems

## Discrete-Time System Hamiltonian Function

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

$$H(x_k, \nabla V(x_k), h) = r(x_k, h(x_k)) + \gamma V_h(x_{k+1}) - V_h(x_k)$$

- Directly leads to temporal difference techniques
- System dynamics does not occur
- Two occurrences of value allow APPROXIMATE DYNAMIC PROGRAMMING methods

## Continuous-Time System Hamiltonian Function

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

$$H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \dot{x} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u)$$

Leads to off-line solutions if system dynamics is known
Hard to do on-line learning

- How to define temporal difference?
- System dynamics DOES occur
- Only ONE occurrence of value gradient

How can one do Policy Iteration for Unknown Continuous-Time Systems?
What is Value Iteration for Continuous-Time systems?
How can one do ADP for CT Systems?

Discrete-Time Systems
Adaptive (Approximate) Dynamic Programming

## Four ADP Methods proposed by Paul Werbos

Critic NN to approximate:

Heuristic dynamic programming

Value Iteration

Value $V(x_k)$

AD Heuristic dynamic programming
(Watkins Q Learning)

Q function $Q(x_k, u_k)$

Dual heuristic programming

Gradient $\dfrac{\partial V}{\partial x}$

AD Dual heuristic programming

Gradients $\dfrac{\partial Q}{\partial x}, \ \dfrac{\partial Q}{\partial u}$

Action NN to approximate the Control

Bertsekas- Neurodynamic Programming

Barto & Bradtke- Q-learning proof (Imposed a settling time)

# CT Systems- Derivation of Nonlinear Optimal Regulator

To find online methods for optimal control

Focus on these two equations

Nonlinear System dynamics $\dot{x} = f(x,u) = f(x) + g(x)u$

Cost/value $V(x(t)) = \int_t^\infty r(x,u)\,dt = \int_t^\infty (Q(x) + u^T R u)\,dt$

Leibniz gives
Differential equivalent

Bellman Equation, in terms of the Hamiltonian function

$$\boxed{H(x,\frac{\partial V}{\partial x},u) = \dot{V} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \dot{x} + r(x,u) = \left(\frac{\partial V}{\partial x}\right)^T \big(f(x) + g(x)u\big) + r(x,u) = 0}$$

Stationarity condition $\frac{\partial H}{\partial u} = 0$

Problem- System dynamics
shows up in Hamiltonian

Stationary Control Policy $\boxed{u = h(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V}{\partial x}}$

HJB equation $0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \frac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$ , $V(0) = 0$

Off-line solution
HJB hard to solve. May not have smooth solution.
Dynamics must be known

# CT Policy Iteration – a Reinforcement Learning Technique

Given any admissible *policy* $u(x) = h(x)$

The cost is given by solving the CT Bellman equation

$$0 = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u)$$

Scalar equation

Utility $r(x,u) = Q(x) + u^T R u$

## Policy Iteration Solution

Pick stabilizing initial control policy $h_0(x)$

Policy Evaluation - Find cost, Bellman eq.

$$0 = \left(\frac{\partial V_j}{\partial x}\right)^T f(x, h_j(x)) + r(x, h_j(x))$$

$$V_j(0) = 0$$

Policy improvement - Update control

$$h_{j+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_j}{\partial x}$$

- Convergence proved by Leake and Liu 1967,
  Saridis 1979 if Lyapunov eq. solved exactly
- Beard & Saridis used Galerkin Integrals to solve Lyapunov eq.
- Abu Khalaf & Lewis used NN to approx. V for nonlinear systems and proved convergence

Full system dynamics must be known

Off-line solution

Converges to solution of HJB

$$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$$

M. Abu-Khalaf, F.L. Lewis, and J. Huang, “Policy iterations on the Hamilton-Jacobi-Isaacs equation for H-infinity state feedback control with input saturation,” IEEE Trans. Automatic Control, vol. 51, no. 12, pp. 1989-1995, Dec. 2006.

# Policy Iterations for the Linear Quadratic Regulator

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

Cost $V(x(t)) = \int_t^{\infty} (x^T Qx + u^T Ru)\, d\tau \qquad = x^T(t)Px(t)$

Differential equivalent is the Bellman equation

$$0 = H(x, \frac{\partial V}{\partial x}, u) = \dot{V} + x^T Qx + u^T Ru = 2\left(\frac{\partial V}{\partial x}\right)^T \dot{x} + x^T Qx + u^T Ru = 2x^T P(Ax + Bu) + x^T Qx + u^T Ru$$

Given any stabilizing FB policy $u = -Kx$

The cost value is found by solving **Lyapunov equation = Bellman equation**

$$0 = (A - BK)^T P + P(A - BK) + Q + K^T RK$$

Optimal Control is

$$u = -R^{-1} B^T Px = -Kx$$

Algebraic Riccati equation

$$0 = PA + A^T P + Q - PBR^{-1} B^T P$$

Full system dynamics must be known
Off-line solution

# LQR Policy iteration = Kleinman algorithm

1. For a given control policy $u = -K_j x$ solve for the cost:

$$0 = A_j^T P_j + P_j A_j + Q + K_j^T R K_j$$

Bellman eq. = Lyapunov eq.

Matrix equation

$$A_j = A - BK_j$$

2. Improve policy:

$$K_{j+1} = R^{-1} B^T P_j$$

- If started with a stabilizing control policy $K_0$ the matrix $P_j$ monotonically converges to the unique positive definite solution of the Riccati equation.
- Every iteration step will return a stabilizing controller.
- The system has to be known.

OFF-LINE DESIGN
MUST SOLVE LYAPUNOV EQUATION AT EACH STEP.

Kleinman 1968

# Integral Reinforcement Learning

Work of Draguna Vrabie

$$\dot{x} = f(x) + g(x)u$$

Can Avoid knowledge of drift term *f(x)*

Policy iteration requires repeated solution of the CT Bellman equation

$$0 = \dot{V} + r(x,u(x)) = \left(\frac{\partial V}{\partial x}\right)^T \dot{x} + r(x,u(x)) = \left(\frac{\partial V}{\partial x}\right)^T f(x,u(x)) + Q(x) + u^T Ru \equiv H(x, \frac{\partial V}{\partial x}, u(x))$$

This can be done online without knowing *f(x)*
using measurements of *x(t), u(t)* along the system trajectories

D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," Automatica, vol. 45, pp. 477-484, 2009.

# Integral Reinforcement Learning

Work of Draguna Vrabie 2009

value $$V(x(t)) = \int_t^{\infty} r(x,u)\, d\tau = \int_t^{t+T} r(x,u)\, d\tau + \int_{t+T}^{\infty} r(x,u)\, d\tau$$

Key Idea= US Patent

**Lemma 1 – Draguna Vrabie**

$$0 = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u), \quad V(0) = 0$$

Bad Bellman Equation

Is equivalent to Integral reinf. form (IRL) for the CT Bellman eq.

$$V(x(t)) = \int_t^{t+T} r(x,u)\, d\tau \quad + \quad V(x(t+T)), \quad V(0) = 0$$

Good Bellman Equation

Solves Bellman equation without knowing *f(x,u)*

---

Allows definition of temporal difference error for CT systems

$$e(t) = -V(x(t)) + \int_t^{t+T} r(x,u)\, d\tau \quad + \quad V(x(t+T))$$

# Integral Reinforcement Learning (IRL)- Draguna Vrabie

## IRL Policy iteration

Policy evaluation- IRL Bellman Equation

Cost update $$\underline{V_k}(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad \underline{V_k}(x(t+T))$$

CT Bellman eq.

*f(x)* and *g(x)* do not appear

Equivalent to $$0 = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u)$$

Solves Bellman eq. (nonlinear Lyapunov eq.) without knowing system dynamics

Policy improvement

Control gain update $$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$$

*g(x)* needed for control update

Initial stabilizing control is needed

Converges to solution to HJB eq. $$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T gR^{-1}g^T \frac{dV^*}{dx}$$

D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," Automatica, vol. 45, pp. 477-484, 2009.

## CT Policy Iteration – How to implement online?
## Linear Systems Quadratic Cost- LQR

Value function is quadratic $V(x(t)) = x^T(t)Px(t)$

Policy evaluation- solve IRL Bellman Equation

$$x^T(t)P_k x(t) = \int_t^{t+T} x^T(\tau)(Q + K_k{}^T R K_k)x(\tau)\,d\tau \quad + \quad x^T(t+T)P_k x(t+T)$$

$$x^T(t)P_k x(t) - x^T(t+T)P_k x(t+T) = \int_t^{t+T} x^T(\tau)(Q + K_k{}^T R K_k)x(\tau)\,d\tau$$

$$\begin{bmatrix} x^1(t) & x^2(t) \end{bmatrix} \begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} x^1(t) \\ x^2(t) \end{bmatrix} - \begin{bmatrix} x^1(t+T) & x^2(t+T) \end{bmatrix} \begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} x^1(t+T) \\ x^2(t+T) \end{bmatrix}$$

$$= \begin{bmatrix} p_{11} & p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} (x^1)^2 \\ 2x^1x^2 \\ (x^2)^2 \end{bmatrix}_{(t)} - \begin{bmatrix} p_{11} & p_{12} & p_{22} \end{bmatrix} \begin{bmatrix} (x^1)^2 \\ 2x^1x^2 \\ (x^2)^2 \end{bmatrix}_{(t+T)}$$

Quadratic basis set

$$= \bar{p}_k^T \left[ \bar{x}(t) - \bar{x}(t+T) \right]$$

$$\bar{p}_k{}^T \phi(t) \equiv \quad \bar{p}_k{}^T \left[ \bar{x}(t) - \bar{x}(t+T) \right] = \int_t^{t+T} x(\tau)^T (Q + L_k{}^T R L_k) x(\tau) d\tau \equiv \rho(t, t+T)$$

Same form as standard System ID problems

# Approximate Dynamic Programming Implementation

Value Function Approximation (VFA) to Solve Bellman Equation
– Paul Werbos (ADP), Dimitri Bertsekas (NDP)

**Optimal Control and Adaptive Control come together On this slide. Because of RL**

$$V_k(x(t)) = \int_t^{t+T} \left( Q(x) + u_k{}^T R u_k \right) dt + V_k(x(t+T))$$

Approximate value by Weierstrass Approximator Network $V = W^T \phi(x)$

$$W_k^T \phi(x(t)) = \int_t^{t+T} \left( Q(x) + u_k{}^T R u_k \right) dt + W_k^T \phi(x(t+T))$$

$$W_k^T \underbrace{\left[ \phi(x(t)) - \phi(x(t+T)) \right]}_{\text{regression vector}} = \underbrace{\int_t^{t+T} \left( Q(x) + u_k{}^T R u_k \right) dt}_{\text{Reinforcement on time interval } [t,\, t+T]}$$

Scalar equation with vector unknowns

**Same form as standard System ID problems in Adaptive Control**

Now use RLS or batch least-squares along the trajectory to get new weights $W_k$

Then find updated FB

$$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x} = -\tfrac{1}{2} R^{-1} g^T(x) \left[ \frac{\partial \phi(x(t))}{\partial x(t)} \right]^T W_k$$

**Direct Optimal Adaptive Control for Partially Unknown CT Systems**

# Solving the IRL Bellman Equation- LQR case

LQR case $V(x(t)) = x^T(t) P x(t)$

Solve for value function parameters $\begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix}$ $W^T = [\, p_{11} \quad p_{12} \quad p_{22} \,]$

Need data from 3 time intervals to get 3 equations to solve for 3 unknowns

$$W_k^T \left[ \phi(x(t)) - \phi(x(t+T)) \right] = \int_t^{t+T} \left( Q(x) + u_k{}^T R u_k \right) dt$$

$$W_k^T \left[ \phi(x(t+T)) - \phi(x(t+2T)) \right] = \int_{t+T}^{t+2T} \left( Q(x) + u_k{}^T R u_k \right) dt$$

$$W_k^T \left[ \phi(x(t+2T)) - \phi(x(t+3T)) \right] = \int_{t+2T}^{t+3T} \left( Q(x) + u_k{}^T R u_k \right) dt$$

Now solve by Batch least-squares

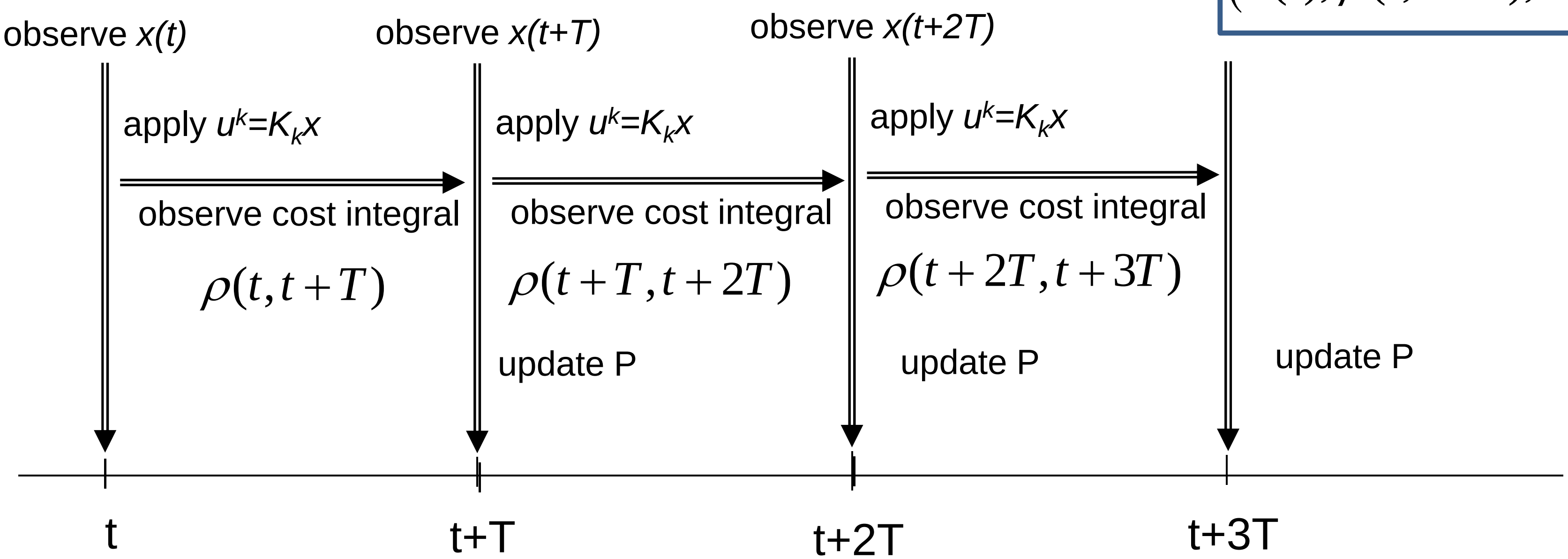
# Integral Reinforcement Learning (IRL)

Solve Bellman Equation - Solves Lyapunov eq. without knowing dynamics

$$W_k^T\left[\phi(x(t))-\phi(x(t+T))\right]=\int_t^{t+T} x(\tau)^T(Q+K_k^T R K_k)x(\tau)d\tau=\rho(t,t+T)$$

Data set at time *[t,t+T)*

$$\left(x(t),\rho(t,t+T),x(t+T)\right)$$




Do RLS until convergence to $P_k$

Or use batch least-squares

A is not needed anywhere

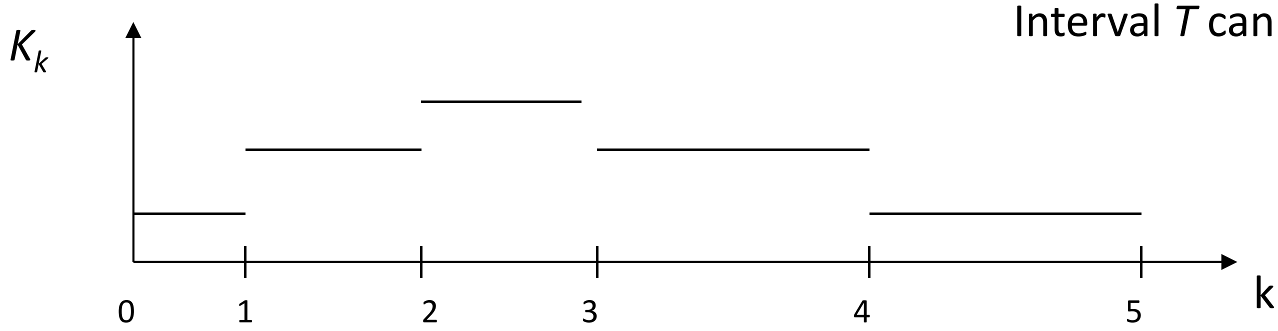
This is a data-based approach that uses measurements of *x(t), u(t)*
Instead of the plant dynamical model.

update control gain

$$K_{k+1}=R^{-1}B^T P_k$$

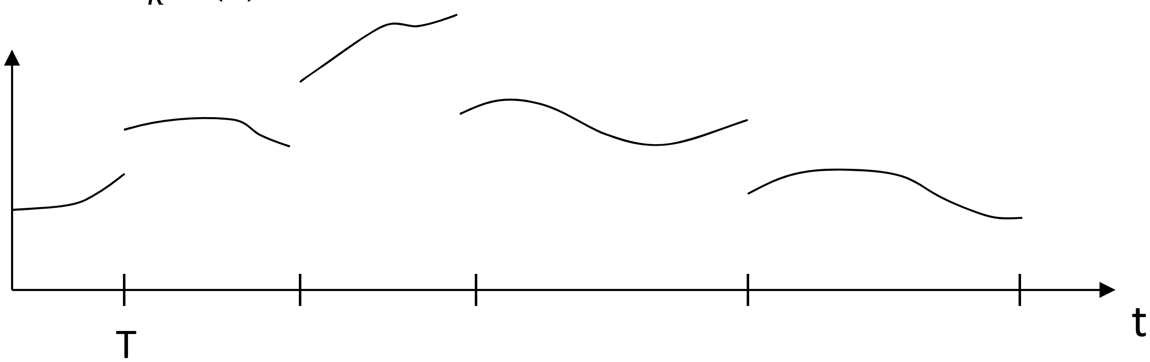
Gain update (Policy)

Interval $T$ can vary




Control

$$u_k(t) = -K_k x(t)$$




Reinforcement Intervals T need not be the same

They can be selected on-line in real time

## Continuous-time control with discrete gain updates

# Persistence of Excitation

$$W_k^T \underbrace{\left[\phi(x(t)) - \phi(x(t+T))\right]}_{\text{Regression vector must be PE}} = \int_t^{t+T} \left(Q(x) + u_k{}^T R u_k\right) dt$$

Relates to choice of reinforcement interval T

# Implementation

Policy evaluation
Need to solve online

$$W_k^T\left[\phi(\mathrm{x}(t))-\phi(x(t+T))\right]=\int_t^{t+T} x(\tau)^T(Q+K_k^T R K_k)x(\tau)d\tau=\rho(t,t+T)$$

Add a new state= Integral Reinforcement

$$\dot{\rho} = x^T Q x + u^T R u$$

This is the controller dynamics or memory

Draguna Vrabie

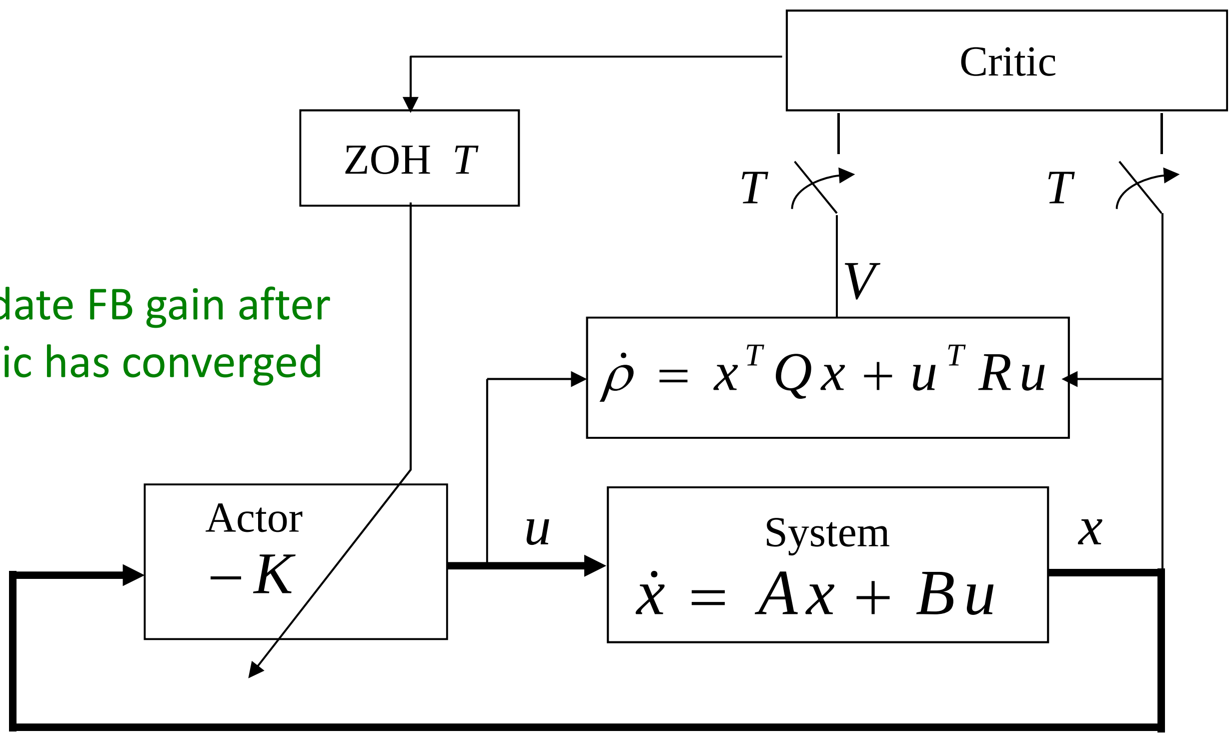
# Direct Optimal Adaptive Controller

Solves Riccati Equation Online without knowing A matrix

**CT time Actor-Critic Structure**

Run RLS or use batch L.S.
To identify value of current control

Update FB gain after
Critic has converged




Dynamic
Control
System
w/ MEMORY

A hybrid continuous/discrete dynamic controller
whose internal state is the observed cost over the interval

Reinforcement interval T can be selected on line on the fly – can change

# Optimal Adaptive IRL for CT systems

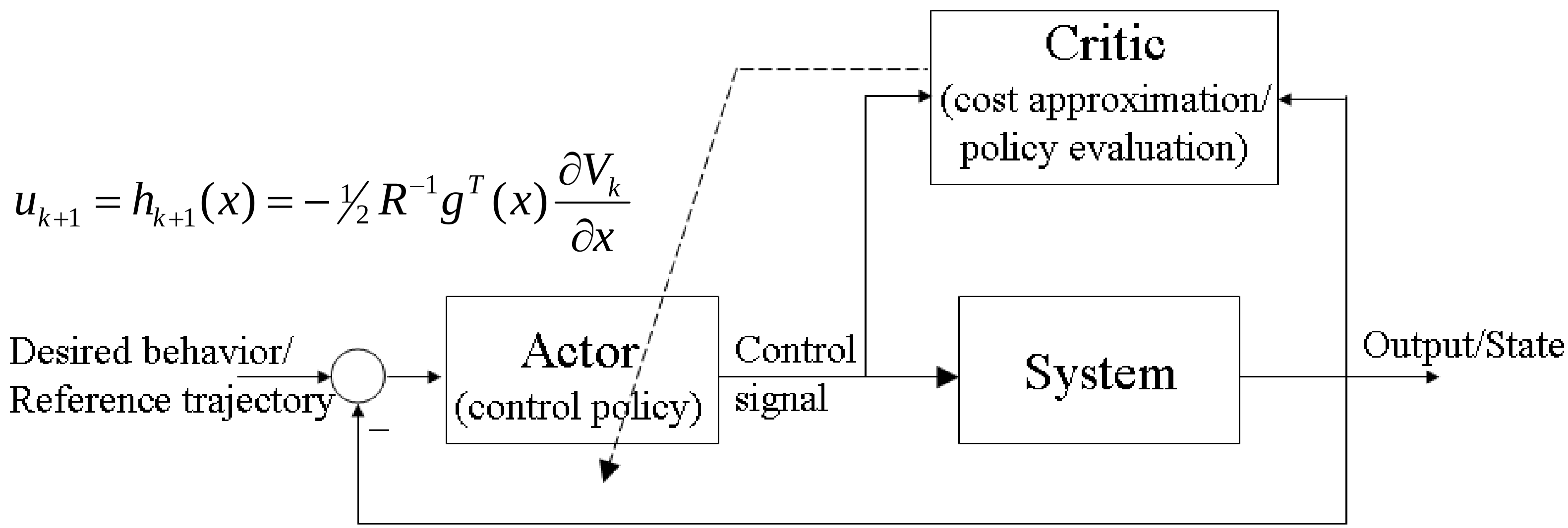
D. Vrabie, 2009

## Actor / Critic structure for CT Systems

Reinforcement learning $$V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

Theta waves 4-8 Hz

$$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$$

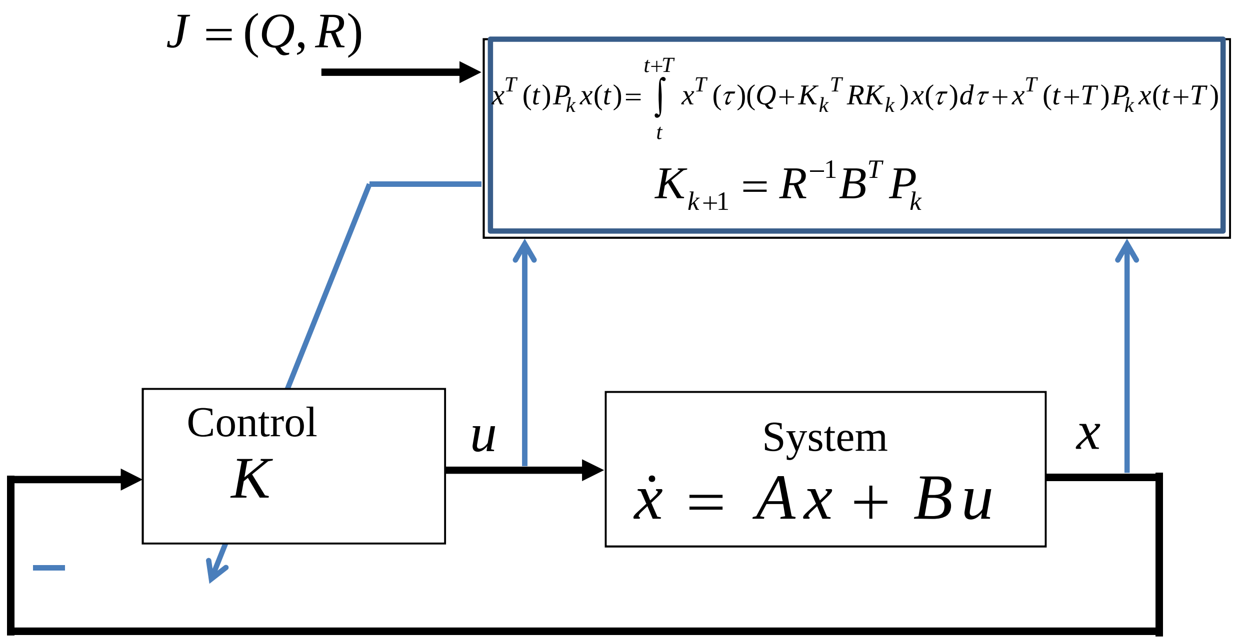



Motor control 200 Hz

A new structure of adaptive controllers

# Data-driven Online Adaptive Optimal Control
# DDO

User prescribed optimization criterion




On-line Performance Loop

On-line Control Loop

Data set at time *[t,t+T)*

$$\big(x(t), \rho(t, t+T), x(t+T)\big)$$

An Online Supervisory Control Procedure
that requires no Knowledge of system dynamics model A

Automatically tunes the control gains in real time to optimize a user given cost function
Uses measured data *(u(t),x(t))* along system trajectories

Simulation 1- F-16 aircraft pitch rate controller

$$\dot{x} = \begin{bmatrix} -1.01887 & 0.90506 & -0.00215 \\ 0.82225 & -1.07741 & -0.17555 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

Stevens and Lewis 2003

$$x = [\alpha \quad q \quad \delta_e]$$

$$Q = I, \quad R = I$$

ARE $$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

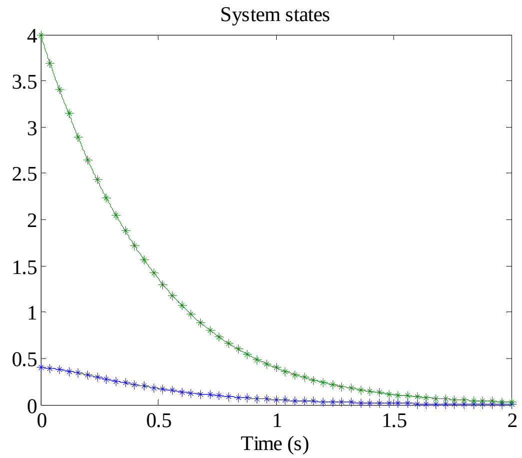
Select quadratic NN basis set for VFA

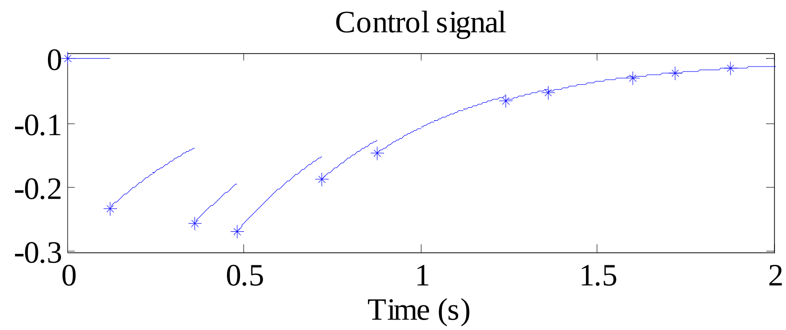
Exact solution $$W_1^* = [p_{11} \quad 2p_{12} \quad 2p_{13} \quad p_{22} \quad 2p_{23} \quad p_{33}]^T$$

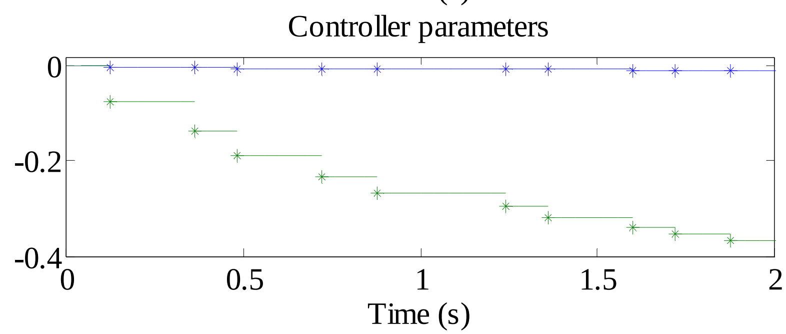
$$= [1.4245 \quad 1.1682 \quad -0.1352 \quad 1.4349 \quad -0.1501 \quad 0.4329]^T$$

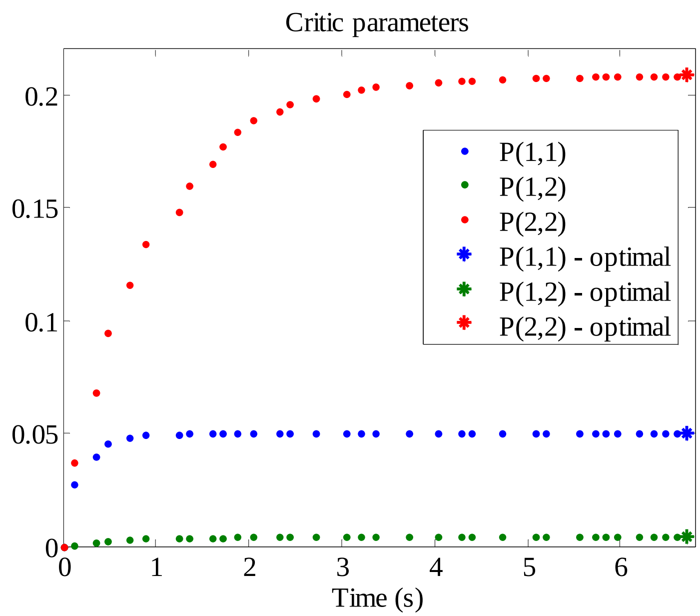
Simulations on: F-16 autopilot

A matrix not needed




Converge to SS Riccati equation soln

Solves ARE online without knowing A

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

A. Al-Tamimi, D. Vrabie, Youyi Wang

## Simulation 2: Load Frequency Control of Electric Power system

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

$$x(t) = [\Delta f(t) \quad \Delta P_g(t) \quad \Delta X_g(t) \quad \Delta E(t)]^T$$

Frequency
Generator output
Governor position
Integral control

$$A = \begin{bmatrix} -1/T_p & K_p/T_p & 0 & 0 \\ 0 & -1/T_T & 1/T_T & 0 \\ -1/RT_G & 0 & -1/T_G & -1/T_G \\ K_E & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1/T_G \\ 0 \end{bmatrix}$$

ARE

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

ARE solution using full dynamics model (A,B)

$$P_{ARE} = \begin{bmatrix} 0.4750 & 0.4766 & 0.0601 & 0.4751 \\ 0.4766 & 0.7831 & 0.1237 & 0.3829 \\ 0.0601 & 0.1237 & 0.0513 & 0.0298 \\ 0.4751 & 0.3829 & 0.0298 & 2.3370 \end{bmatrix}.$$

$$0 = PA + A^T P + Q - PBR^{-1}B^T P$$

Solves ARE online without knowing A

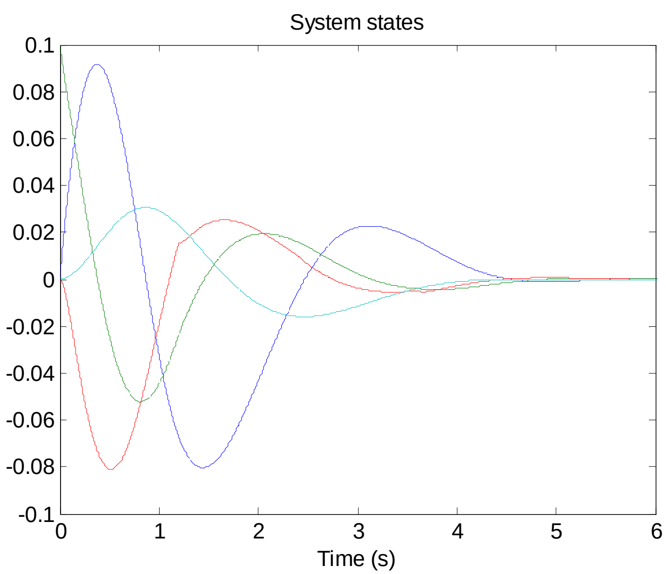
$$P_{ARE} = \begin{bmatrix} 0.4750 & 0.4766 & 0.0601 & 0.4751 \\ 0.4766 & 0.7831 & 0.1237 & 0.3829 \\ 0.0601 & 0.1237 & 0.0513 & 0.0298 \\ 0.4751 & 0.3829 & 0.0298 & 2.3370 \end{bmatrix}.$$

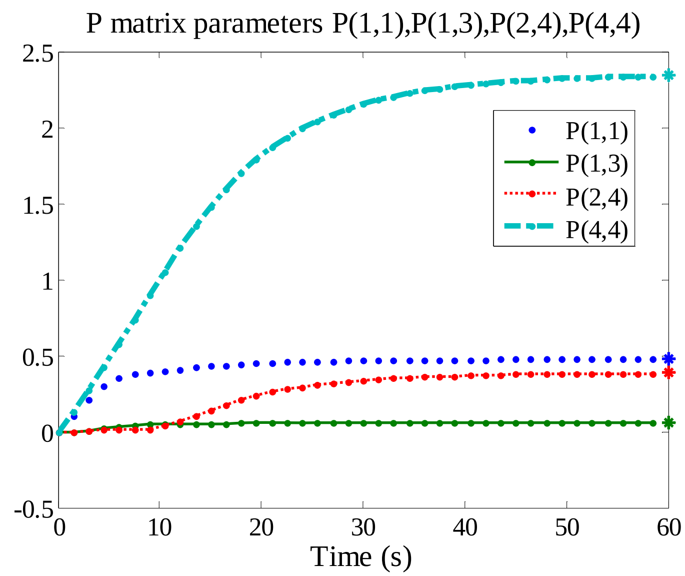
$$P_{critic\ NN} = \begin{bmatrix} 0.4802 & 0.4768 & 0.0603 & 0.4754 \\ 0.4768 & 0.7887 & 0.1239 & 0.3834 \\ 0.0603 & 0.1239 & 0.0567 & 0.0300 \\ 0.4754 & 0.3843 & 0.0300 & 2.3433 \end{bmatrix}.$$

IRL period of T= 0.1s.

Fifteen data points $\big(x(t), x(t+T), \rho(t:t+T)\big)$

Hence, the value estimate was updated every 1.5s.



# Optimal Control Design Allows a Lot of Design Freedom

## The Power of Optimal Design

Once you can do optimal design that minimizes a performance index, many sorts of designs are immediately possible.

Minimum energy

$$J = \frac{1}{2}\int_0^\infty x^T Q x + u^T R u \, dt$$

Minimum fuel

$$J = \frac{1}{2}\int_0^\infty x^T Q x + \rho |u| \, dt$$

Minimum time

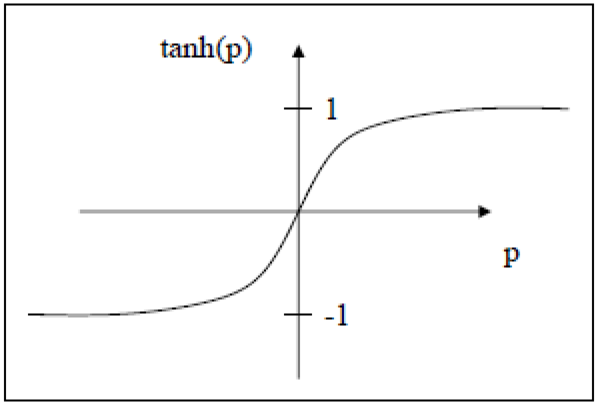
$$J = \int_0^T 1 \, dt = T$$

Constrained control inputs

$$J = \frac{1}{2}\int_0^\infty \left( Q(x) + \int_0^u \sigma^{-1}(v) dv \right) dt$$

Approximate minimum time with smooth control inputs

$$J = \frac{1}{2}\int_0^\infty \left( \tanh(x^T Q x) + \rho \int_0^u \sigma^{-1}(v) dv \right) dt$$



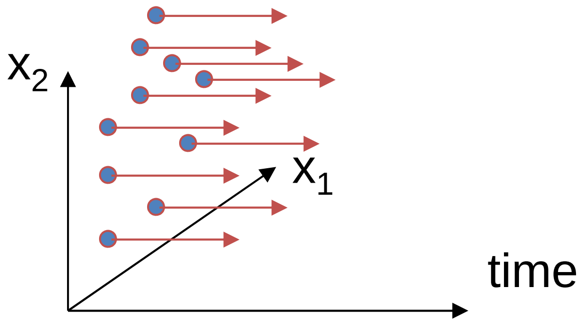
# Issues with Nonlinear ADP

**Selection of NN Training Set**

LS local smooth solution for Critic NN update

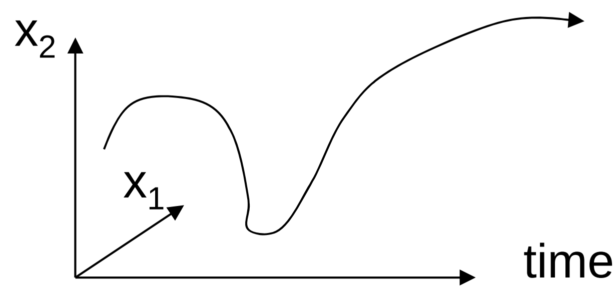
$$0 = \left(\frac{\partial V}{\partial x}\right)^T f(x,u) + r(x,u) \equiv H(x, \frac{\partial V}{\partial x}, u), \quad V(0) = 0$$

$$V(x(t)) = \int_t^{t+T} r(x,u)\, d\tau \quad + \quad V(x(t+T)), \quad V(0) = 0$$




Integral over a region of state-space
Approximate using a set of points

Batch LS




Take sample points along a single trajectory

Recursive Least-Squares RLS

Set of points over a region vs. points along a trajectory

For Linear systems- these are the same

For Nonlinear systems
  Persistence of excitation is needed to solve for the weights
  But EXPLORATION is needed to identify the complete value function
    - PE Versus Exploration

# IRL Value Iteration - Draguna Vrabie

## IRL Policy iteration

Initial stabilizing control is needed

Policy evaluation- IRL Bellman Equation

Cost update
$$V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

CT PI Bellman eq.
= Lyapunov eq.

Policy improvement

Control gain update
$$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$$

Converges to solution to HJB eq.
$$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T g R^{-1} g^T \frac{dV^*}{dx}$$

---

## IRL Value iteration

Initial stabilizing control is NOT needed

Value evaluation- IRL Bellman Equation

Cost update
$$V_{k+1}(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

CT VI Bellman eq.

Policy improvement

Control gain update
$$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_{k+1}}{\partial x}$$

**Converges if T is small enough**

# Kung Tz  500 BC
# Confucius

孔子

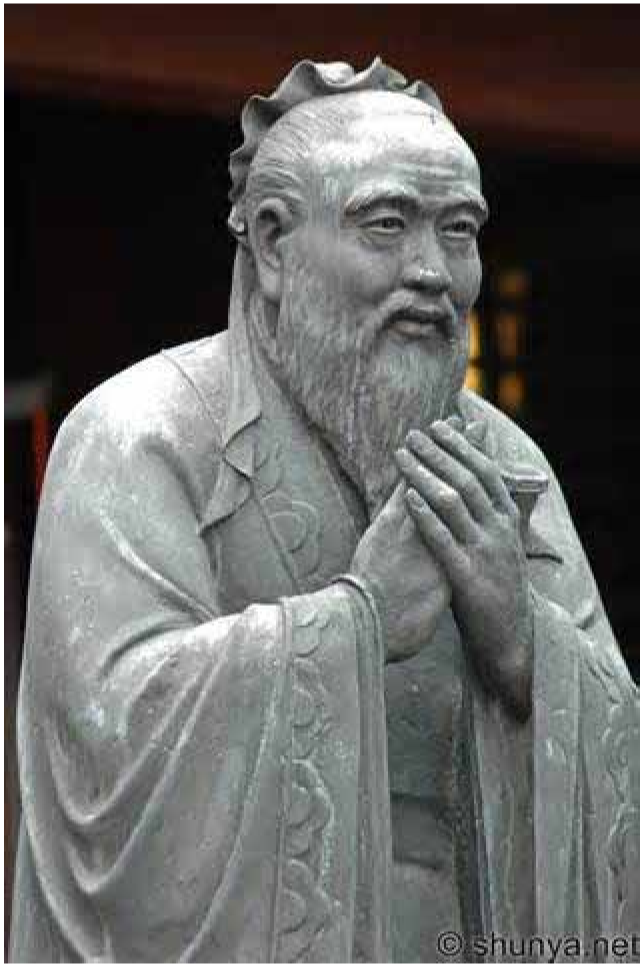
Man's relations to
- Family
- Friends
- Society
- Nation
- Emperor
- Ancestors

Tian xia da tong
Harmony under heaven

Archery
Chariot driving

Music
Rites and Rituals

Poetry
Mathematics



# Optimal Adaptive IRL for CT systems

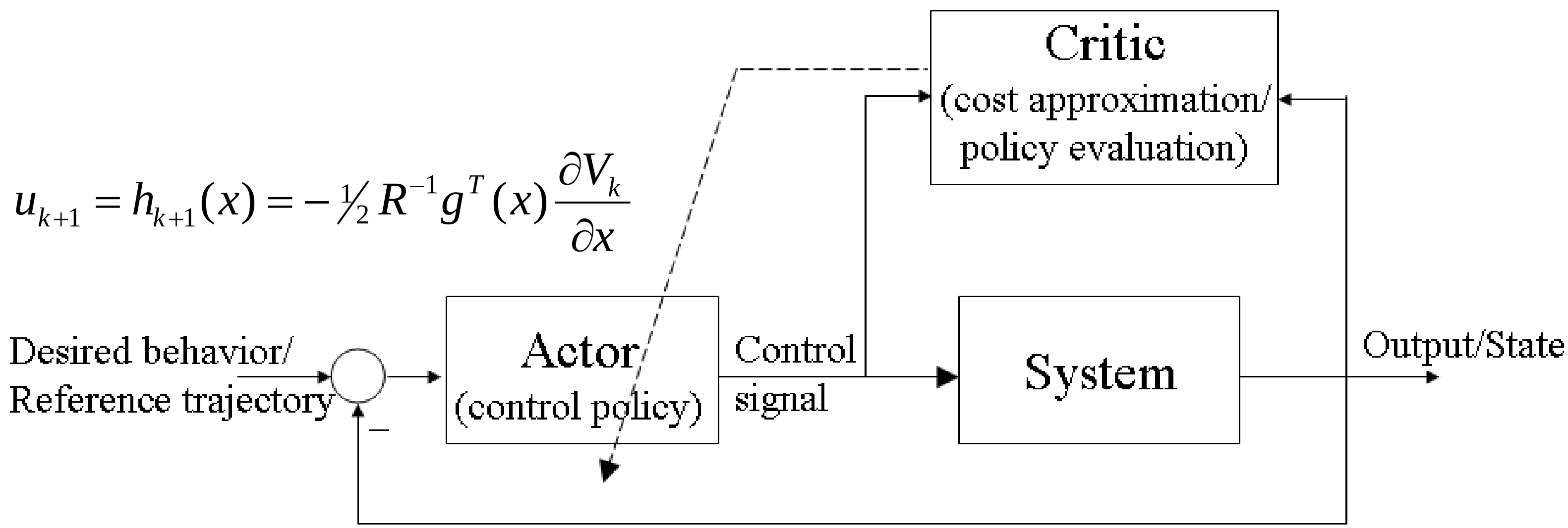
D. Vrabie, 2009

## Actor / Critic structure for CT Systems

Reinforcement learning
$$V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

Theta waves 4-8 Hz

$$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$$




Motor control 200 Hz

A new structure of adaptive controllers

# Oscillation is a fundamental property of neural tissue

**Brain has multiple adaptive clocks with different timescales**

*gamma rhythms* 30-100 Hz, hippocampus and neocortex
high cognitive activity.
- consolidation of memory
- spatial mapping of the environment – place cells

The high frequency processing is due to the large amounts of sensorial data to be processed

*theta rhythm*, Hippocampus, Thalamus, 4-10 Hz
sensory processing, memory and voluntary control of movement.

Spinal cord

Motor control 200 Hz

D. Vrabie, F.L. Lewis, D. Levine, “Neural Network-Based Adaptive Optimal Controller- A Continuous-Time Formulation -,” Proc. Int. Conf. Intelligent Control, Shanghai, Sept. 2008.

D. Vrabie and F.L. Lewis, “Neural network approach to continuous-time direct adaptive optimal control for partially-unknown nonlinear systems,” Neural Networks, vol. 22, no. 3, pp. 237-246, Apr. 2009.

Deliberative evaluation

Limbic system

control

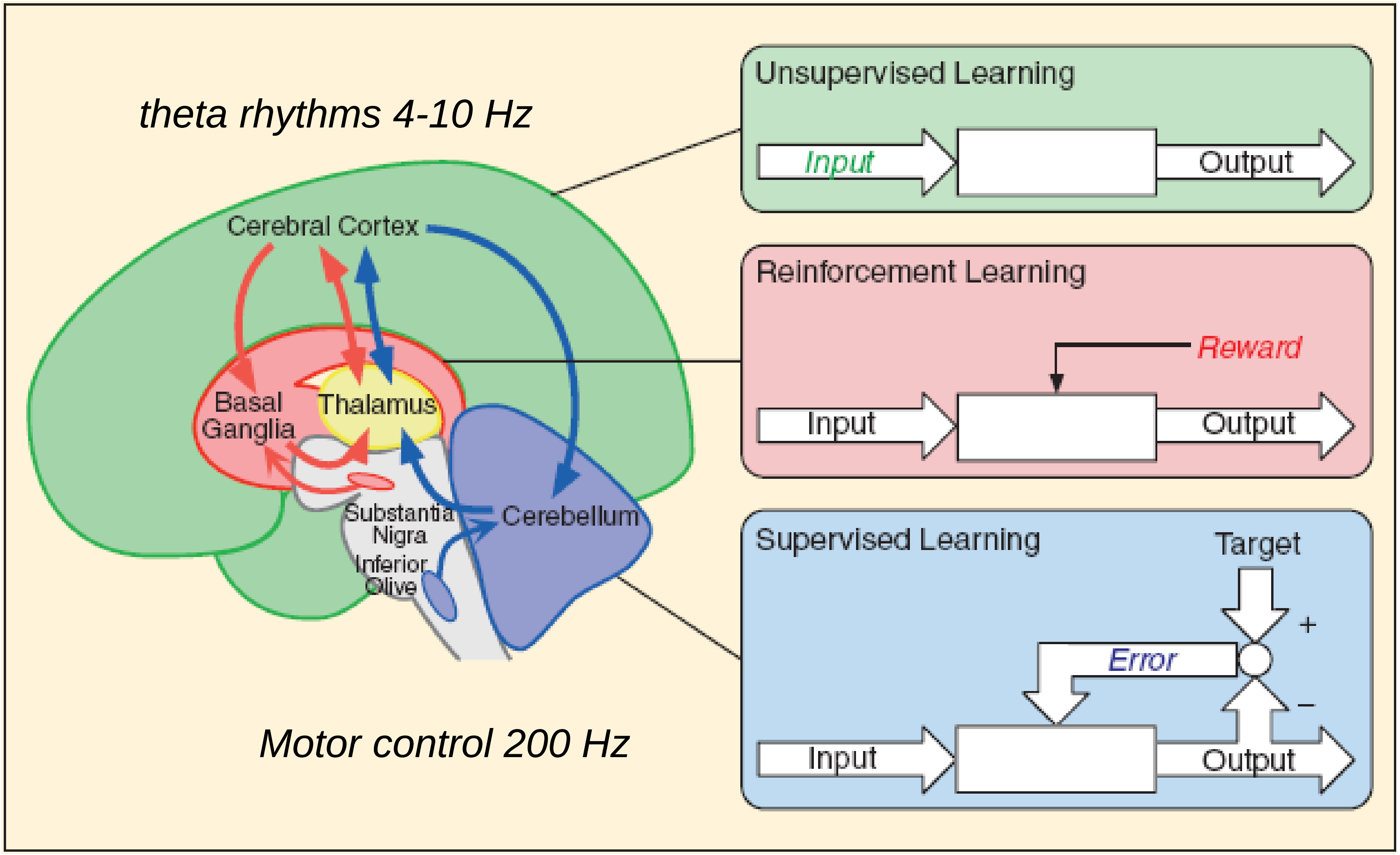
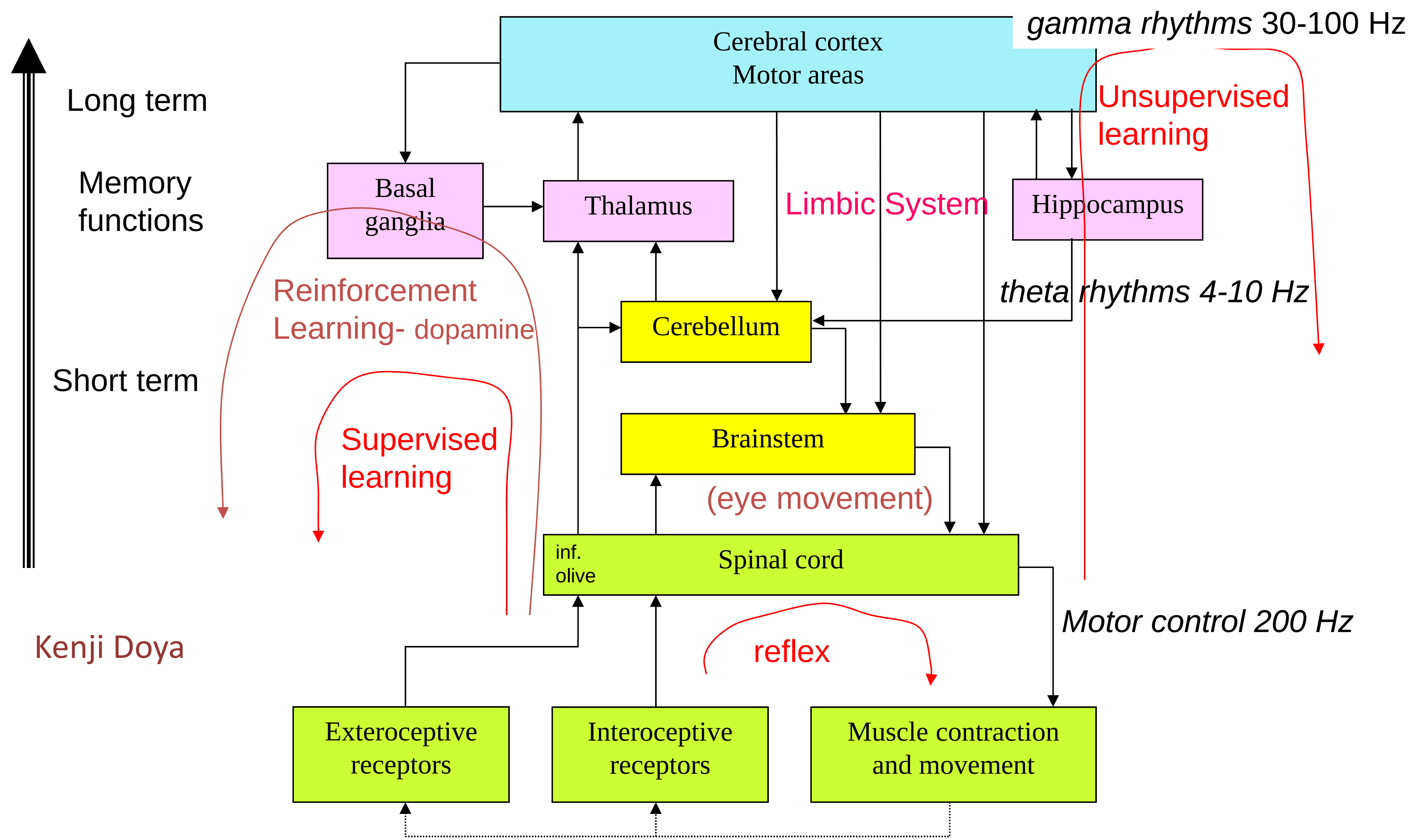



**Figure 1.** *Learning-oriented specialization of the cerebellum, the basal ganglia, and the cerebral cortex [1], [2]. The cerebellum is specialized for supervised learning based on the error signal encoded in the climbing fibers from the inferior olive. The basal ganglia are specialized for reinforcement learning based on the reward signal encoded in the dopaminergic fibers from the substantia nigra. The cerebral cortex is specialized for unsupervised learning based on the statistical properties of the input signal.*

Doya, Kimura, Kawato 2001

## Summary of Motor Control in the Human Nervous System

picture by E. Stingu
D. Vrabie




Hierarchy of multiple parallel loops

# Synchronous Real-time Data-driven Optimal Control

# Optimal Adaptive
# Integral Reinforcement Learning for CT systems

D. Vrabie, 2009

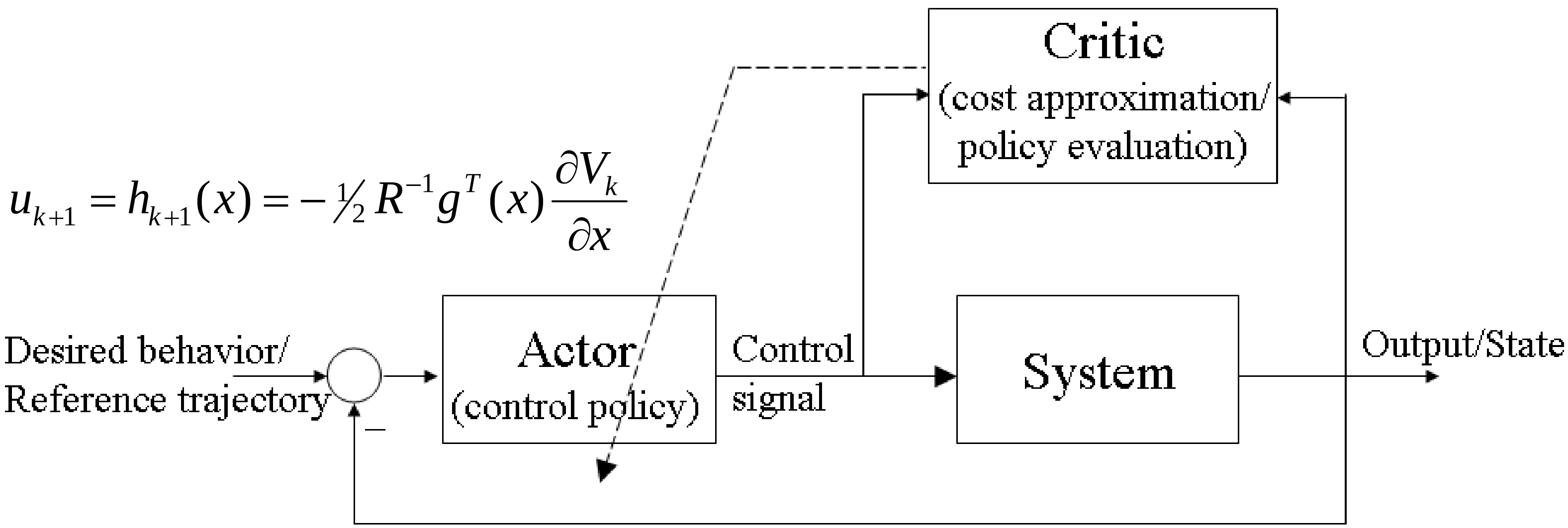
Policy Iteration gives the structure needed for online optimal solution

## Actor / Critic structure for CT Systems

$$V_k(x(t)) = \int_t^{t+T} r(x,u_k)\,dt \quad + \quad V_k(x(t+T))$$

Theta waves 4-8 Hz

$$u_{k+1} = h_{k+1}(x) = -\tfrac{1}{2} R^{-1} g^T(x) \frac{\partial V_k}{\partial x}$$




Motor control 200 Hz

A new structure of adaptive controllers

# Synchronous Online Solution of Optimal Control for Nonlinear Systems

Kyriakos Vamvoudakis

## Critic Network

Take VFA as $V(x) = \hat{W}_1^T \phi_1(x) + \varepsilon(x)$ , $\nabla V(x) = \nabla \phi_1^T \hat{W}_1$

Then IRL Bellman eq $V(x(t)) = \int_{t-T}^{t} \left( Q(x) + u_k^T R u_k \right) dt + V(x(t+T))$

becomes $\hat{W}_1^T \phi(x(t-T)) = \int_{t-T}^{t} \left( Q(x) + u_k^T R u_k \right) dt + \hat{W}_1^T \phi(x(t))$

## Action Network for Control Approximation

$$u(x) = -\tfrac{1}{2} R^{-1} g^T(x) \nabla \phi_1^T \hat{W}_2,$$

Define $\Delta\phi(x(t)) \equiv \phi(x(t)) - \phi(x(t-T))$

Bellman eq becomes $\Delta\phi(x(t))^T \hat{W}_1 + \int_{t-T}^{t} \left( Q(x) + \frac{1}{4} \hat{W}_2^T \overline{D}_1 \hat{W}_2 \right) = 0$

K.G. Vamvoudakis and F.L. Lewis, “Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem,” Automatica, vol. 46, no. 5, pp. 878-888, May 2010.

# Data-driven Online Synchronous Policy Iteration using IRL

Does not need to know *f(x)*

Vamvoudakis & Vrabie

**Theorem (Vamvoudakis & Vrabie)- Online Learning of Nonlinear Optimal Control**

Let $\Delta\phi(x(t)) \equiv \phi(x(t)) - \phi(x(t-T))$ be PE. Tune critic NN weights as

$$\dot{\hat{W}}_1 = -a_1 \frac{\Delta\phi(x(t))}{\left(1+\Delta\phi(x(t))^T \Delta\phi(x(t))\right)^2}\left(\Delta\phi(x(t))^T \hat{W}_1 + \int_{t-T}^{t}\left(Q(x)+\frac{1}{4}\hat{W}_2{}^T \overline{D}_1 \hat{W}_2\right)d\tau\right)$$

Learning the Value

Tune actor NN weights as

$$\dot{\hat{W}}_2 = -a_2\left(F_2\hat{W}_2 - F_1\Delta\phi(x(t))^T \hat{W}_1\right) - \tfrac{1}{4}a_2 \overline{D}_1(x)\hat{W}_2 \frac{\Delta\phi(x(t))^T}{\left(1+\Delta\phi(x(t))^T \Delta\phi(x(t))\right)^2}\hat{W}_1$$

Learning the control policy

Then there exists an $N_0$ such that, for the number of hidden layer units $N > N_0$

the closed-loop system state, the critic NN error $\tilde{W}_1 = W_1 - \hat{W}_1$

and the actor NN error $\tilde{W}_2 = W_1 - \hat{W}_2$ are UUB bounded.

Data set at time *[t,t+T)*

$$\left(x(t), \rho(t-T,t), x(t-T)\right)$$

Lyapunov energy-based Proof:

$$L(t) = V(x) + \frac{1}{2}tr(\tilde{W}_1^T a_1^{-1}\tilde{W}_1) + \frac{1}{2}tr(\tilde{W}_2^T a_2^{-1}\tilde{W}_2).$$

V(x)= Unknown solution to HJB eq.

$$0 = \left(\frac{dV}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV}{dx}\right)^T gR^{-1}g^T \frac{dV}{dx}$$

Guarantees stability

$$\tilde{W}_1 = W_1 - \hat{W}_1$$

$$\tilde{W}_2 = W_1 - \hat{W}_2$$

$W_1$= Unknown LS solution to Bellman equation for given N

$$H(x, W_1, u) = W_1^T \nabla\phi_1(f + gu) + Q(x) + u^T Ru = \varepsilon_H$$

## Synchronous Online Solution of Optimal Control for Nonlinear Systems

K.G. Vamvoudakis and F.L. Lewis, “Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem,” Automatica, vol. 46, no. 5, pp. 878-888, May 2010.

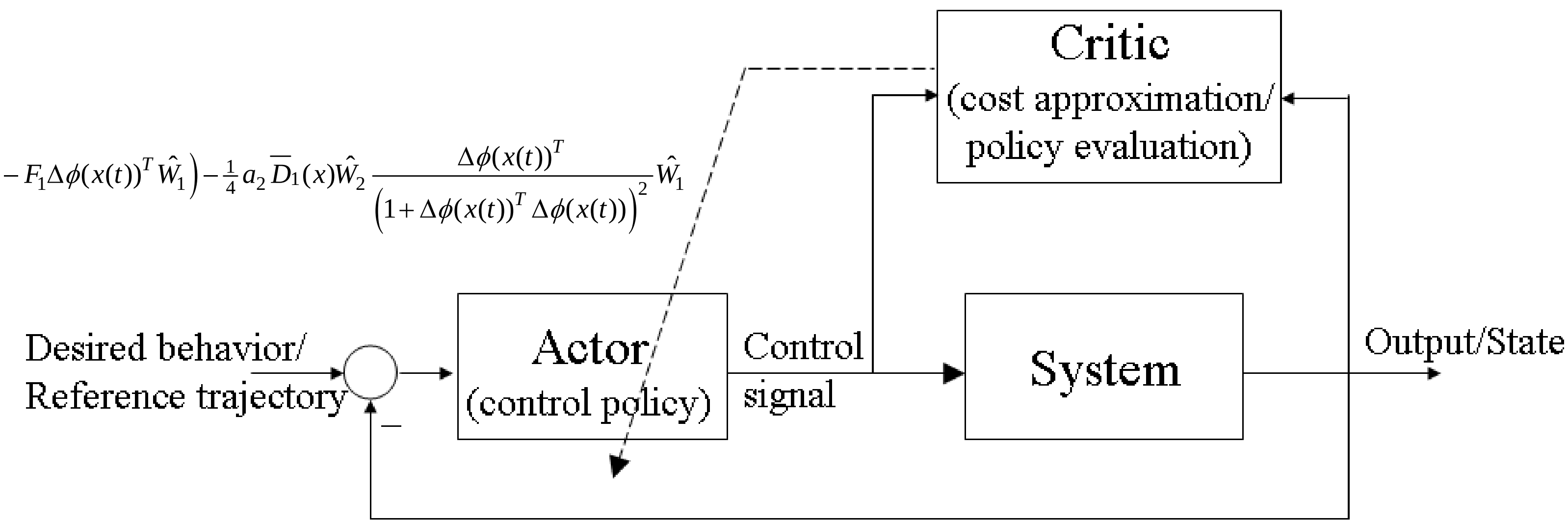
A new form of Adaptive Control with TWO tunable networks

# Adaptive Critic structure

Reinforcement learning

$$\dot{\hat W}_1 = -a_1 \frac{\Delta\phi(x(t))}{\left(1+\Delta\phi(x(t))^T\Delta\phi(x(t))\right)^2}\left(\Delta\phi(x(t))^T\hat W_1 + \int_{t-T}^{t}\left(Q(x)+\frac{1}{4}\hat W_2^{\,T}\bar D_1 \hat W_2\right)d\tau\right)$$

$$\dot{\hat W}_2 = -a_2\left(F_2\hat W_2 - F_1\Delta\phi(x(t))^T\hat W_1\right) - \tfrac{1}{4} a_2 \bar D_1(x)\hat W_2 \frac{\Delta\phi(x(t))^T}{\left(1+\Delta\phi(x(t))^T\Delta\phi(x(t))\right)^2}\hat W_1$$



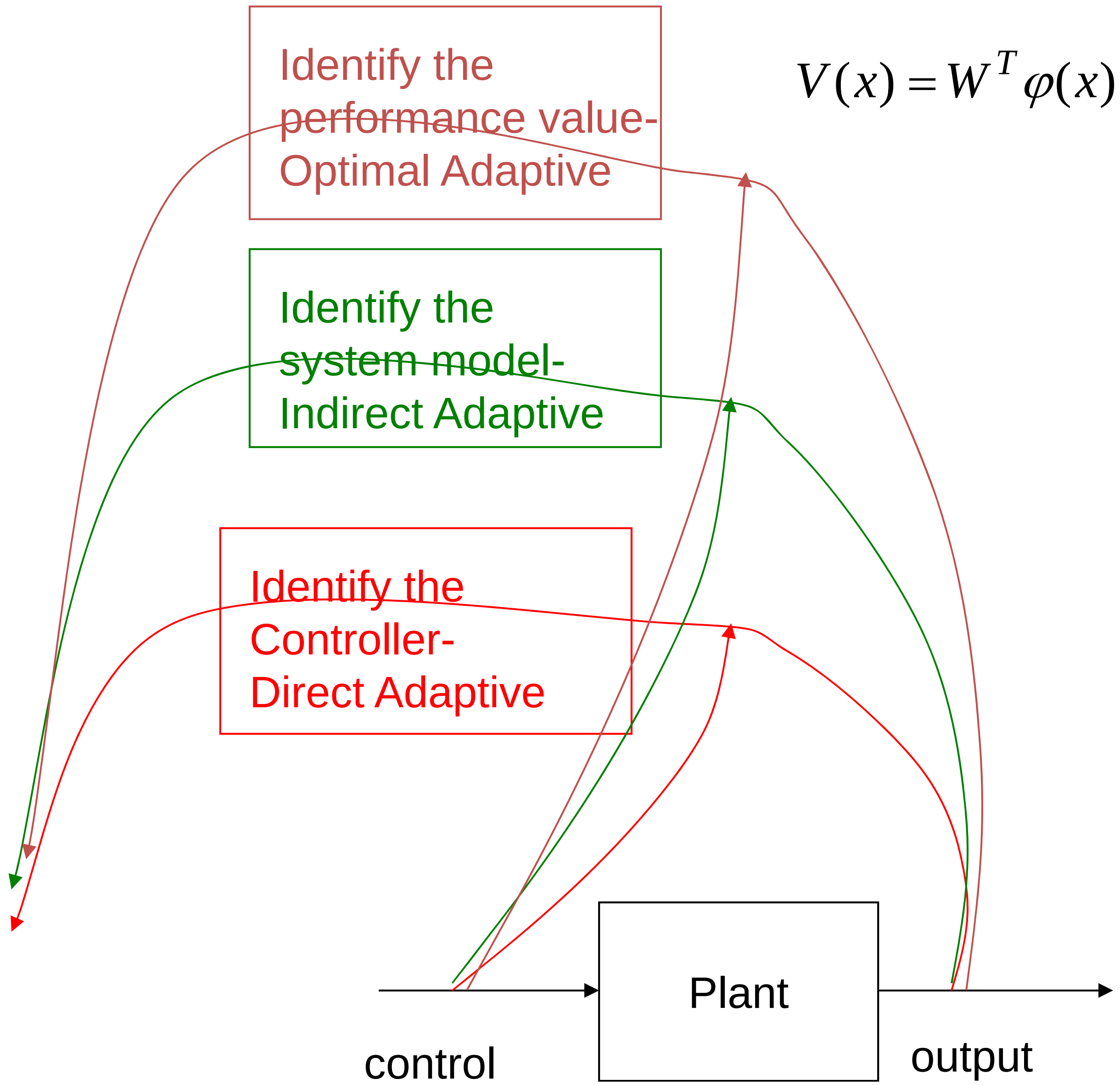

Two Learning Networks

Tune them Simultaneously

A new structure of adaptive controllers

# A New Class of Adaptive Control

$$V(x) = W^T \varphi(x)$$



## Simulation 1- F-16 aircraft pitch rate controller

$$\dot{x}=\begin{bmatrix}-1.01887 & 0.90506 & -0.00215\\ 0.82225 & -1.07741 & -0.17555\\ 0 & 0 & -1\end{bmatrix}x+\begin{bmatrix}0\\0\\1\end{bmatrix}u$$

$$Q=I,\quad R=I$$

Stevens and Lewis 2003

$$x=[\alpha \quad q \quad \delta_e]$$

Solves ARE online

$$0=PA+A^TP+Q-PBR^{-1}B^TP$$

Select quadratic NN basis set for VFA

Exact solution $W_1^* = [\mathrm{p}_{11} \quad 2\mathrm{p}_{12} \quad 2\mathrm{p}_{13} \quad \mathrm{p}_{22} \quad 2\mathrm{p}_{23} \quad \mathrm{p}_{33}]^T$

$$=[1.4245 \quad 1.1682 \quad -0.1352 \quad 1.4349 \quad -0.1501 \quad 0.4329]^T$$

---

Must add probing noise to get PE

$$u(x)=-\tfrac{1}{2}R^{-1}g^T(x)\nabla\phi_1^T\hat{W}_2+n(t)$$

(exponentially decay *n(t)*)

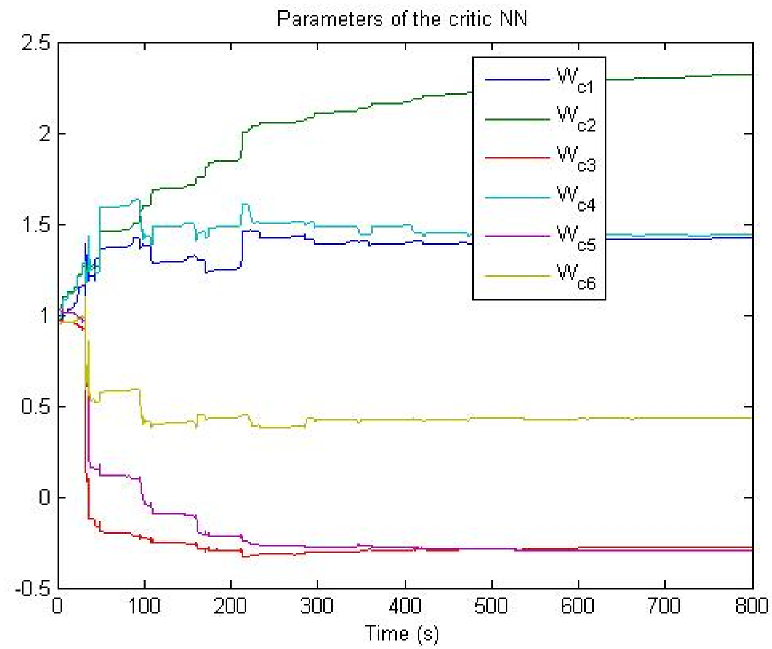
Algorithm converges to

$$\hat{W}_1(t_f)=[1.4279 \quad 1.1612 \quad -0.1366 \quad 1.4462 \quad -0.1480 \quad 0.4317]^T.$$

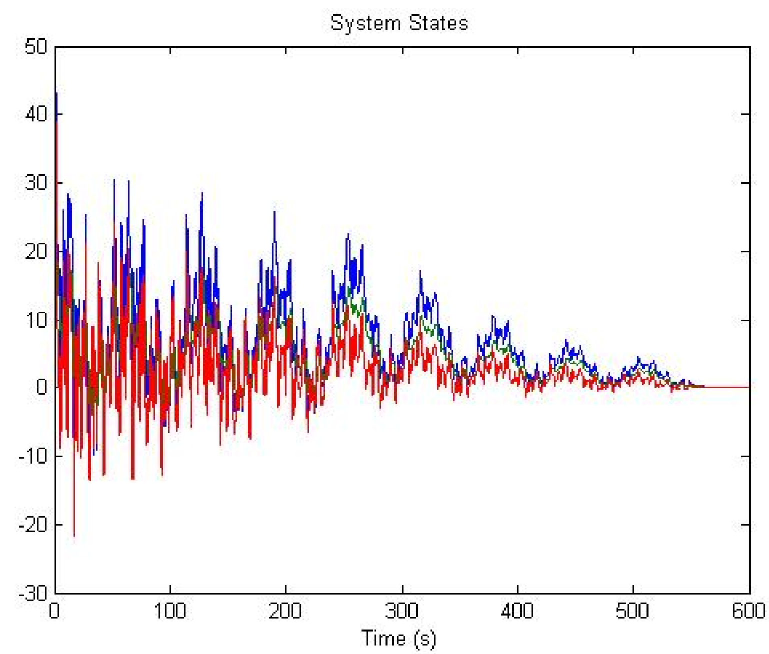
$$\hat{W}_2(t_f)=[1.4279 \quad 1.1612 \quad -0.1366 \quad 1.4462 \quad -0.1480 \quad 0.4317]^T$$

$$\hat{u}_2(x)=-\tfrac{1}{2}R^{-1}B^TPx=-\tfrac{1}{2}R^{-1}\begin{bmatrix}0\\0\\1\end{bmatrix}^T\begin{bmatrix}2x_1 & 0 & 0\\ x_2 & x_1 & 0\\ x_3 & 0 & x_1\\ 0 & 2x_2 & 0\\ 0 & x_3 & x_2\\ 0 & 0 & 2x_3\end{bmatrix}^T\begin{bmatrix}1.4279\\1.1612\\-0.1366\\1.4462\\-0.1480\\0.4317\end{bmatrix}$$

Critic NN parameters- Converge to ARE solution




System states



Nevistic V. and Primbs J. A. (1996)
Converse optimal

## Simulation 2. – Nonlinear System

$$\dot{x} = f(x) + g(x)u, \qquad x \in R^2$$

$$f(x) = \begin{bmatrix} -x_1 + x_2 \\ -0.5x_1 - 0.5x_2(1-(\cos(2x_1)+2)^2) \end{bmatrix}$$

$$g(x) = \begin{bmatrix} 0 \\ \cos(2x_1)+2 \end{bmatrix}.$$

$$Q = I, \quad R = I$$

Solves HJB equation online

$$0 = \left(\frac{dV^*}{dx}\right)^T f + Q(x) - \tfrac{1}{4}\left(\frac{dV^*}{dx}\right)^T gR^{-1}g^T \frac{dV^*}{dx}$$

Optimal Value $\quad V^*(x) = \dfrac{1}{2}x_1^{\,2} + x_2^{\,2}$

Optimal control $\quad u^*(x) = -(\cos(2x_1)+2)x_2.$

---

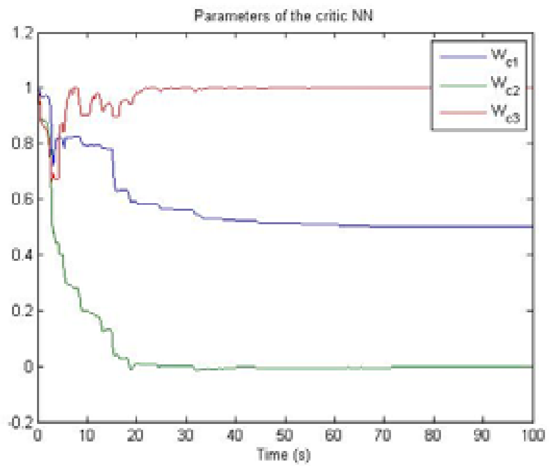
Select VFA basis set $\quad \phi_1(x) = [\; x_1^{\,2} \quad x_1x_2 \quad x_2^{\,2}\,]^T,$

Algorithm converges to

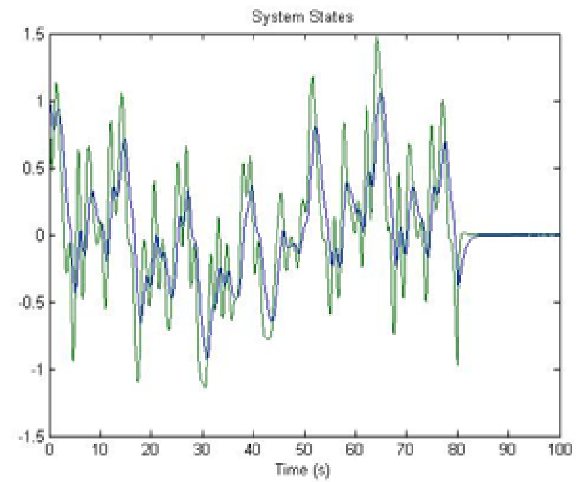
$$\hat{W}_1(t_f) = [0.5017 \quad -0.0020 \quad 1.0008]^T.$$

$$\hat{W}_2(t_f) = [0.5017 \quad -0.0020 \quad 1.0008]^T.$$

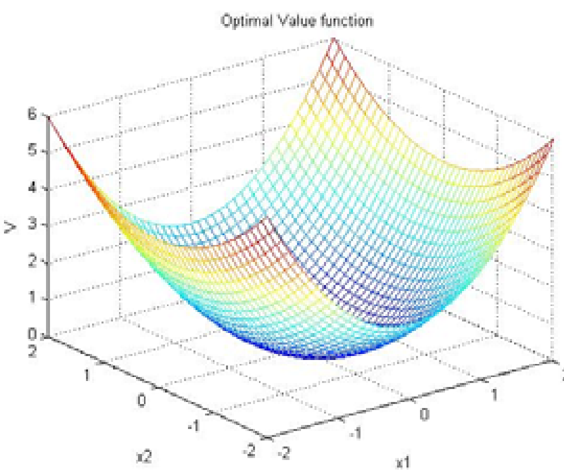
$$\hat{u}_2(x) = -\tfrac{1}{2}R^{-1}\begin{bmatrix} 0 \\ \cos(2x_1)+2 \end{bmatrix}^T \begin{bmatrix} 2x_1 & 0 \\ x_2 & x_1 \\ 0 & 2x_2 \end{bmatrix}^T \begin{bmatrix} 0.5017 \\ -0.0020 \\ 1.0008 \end{bmatrix}$$



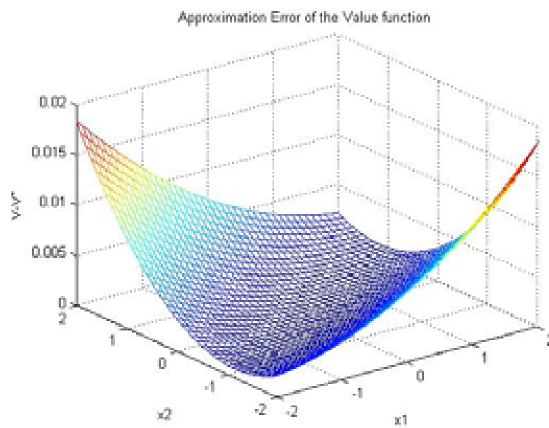

Critic NN parameters



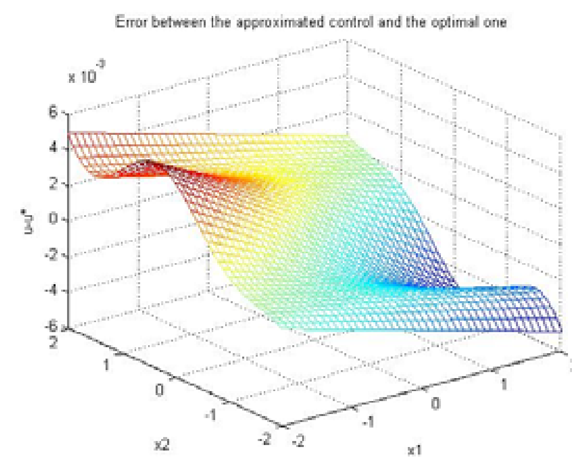

states




Optimal value fn.




Value fn. approx. error




Control approx error