ELEC 5530/6530
Robot Control
Lecture 11: Reinforcement Learning for Optimal Control

Instructor: Dr. Bosen Lian
Electrical and Computer Engineering
Auburn University
Spring 2026

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

$Q(x) \geq 0$

Leibniz gives Differential equivalent

Bellman Equation, in terms of the Hamiltonian function

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

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

Problem- System dynamics shows up in Hamiltonian

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

nonlinear

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

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

Converges to solution of HJB

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

- 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

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

Differential equivalent is the Bellman equation

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

Given any stabilizing FB policy $u = -Kx$

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

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

Optimal Control is

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

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

$$A_j = A - BK_j$$

j: iteration index

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

Policy Iteration

K_0

1. $0 = (A - BK_j)^T P_j + P_j (A - BK_j) + Q + K_j^T R K_j$

$\Rightarrow$ P_0

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

K_1

Still need (A, B).

Integral Reinforcement Learning

Work of Draguna Vrabie

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

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

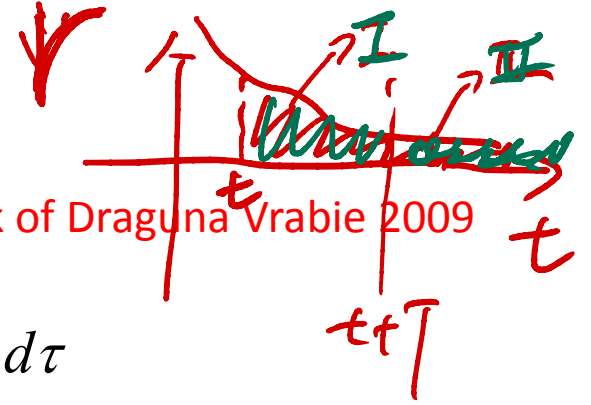
Policy iteration requires repeated solution of the CT Bellman equation

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

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

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

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

LQR Case

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

Value function $V(x(t)) = x^T(t) P x(t)$

Lemma 1 - D. Vrabie- LQR case

Lyapunov equation

$$A_c{}^T P + P A_c + L^T R L + Q = 0 \quad , \quad A_c = A - BL$$

is equivalent to Integral Reinforcement Learning form

$$x^T(t) P x(t) = \int_t^{t+T} x^T(\tau)(Q + L^T R L) x(\tau) d\tau + x^T(t+T) P x(t+T)$$

Solves Lyapunov equation without knowing A or B

Proof:

$$\frac{d(x^T P x)}{dt} = x^T (A_c{}^T P + P A_c) x = -x^T (L^T R L + Q) x$$

$$\int_t^{t+T} x^T (Q + L^T R L) x d\tau = -\int_t^{t+T} d(x^T P x) = x^T(t) P x(t) - x^T(t+T) P x(t+T)$$

$$\int_t^{t+T} \dot{\varepsilon}\, d\tau \approx \varepsilon \cdot T$$

$$\chi \rightarrow A_j^T P_j + P_j A_j + Q + K_j^T R K_j = 0$$

Bellman equation

$$\frac{dV(x(t))}{dt} = \left(\frac{\partial V}{\partial x}\right)^T \frac{\partial x}{\partial t}$$

$$0 = \left(\frac{\partial V_j}{\partial x}\right)^T (Ax + B u_j) + x^T Q x + u_j^T R u_j$$

↓ Integral $\int_t^{t+T}$

$$\int_t^{t+T} \dot{V} = V(x(t+T)) - V(x(t))$$

$$0 = V_j(x(t+T)) - V_j(x(t)) + \int_t^{t+T} (x^T Q x + u_j^T R u_j)\, d\tau$$

$$u_j = -K_j x$$

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

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

Integral Reinforcement Learning (IRL)- Draguna Vrabie

IRL Policy iteration

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 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

V_k , k iteration index

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

D. Vrabie proved convergence to the optimal value and control
Automatica 2009, Neural Networks 2009

$$f(x) = a_1 x + a_2 x^2 + a_3 x^3 + \cdots + O(x^n)$$

Nonlinear Case- Approximate Dynamic Programming

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

weight → W, basis function → $\phi(x)$

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

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

Scalar equation with vector unknowns

regression vector

Reinforcement on time interval *[t, t+T]*

Same form as standard System ID problems in Adaptive Control

Now use RLS 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 $V(x(t)) = x^T(t) P x(t)$

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

Solving the IRL Bellman Equation

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

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

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

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

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

Put together

$$W_k^T \left[\Delta\phi(x(t)) \;\; \Delta\phi(x(t+T)) \;\; \Delta\phi(x(t+2T))\right] = \left[\rho(t) \;\; \rho(t+T) \;\; \rho(t+2T)\right]$$

$W^T \Phi = \bar{P}$

$W^T = \bar{P} \Phi^T (\Phi \Phi^T)^{-1}$

Now solve by Batch least-squares

Or can use Recursive Least-Squares (RLS)

BLS

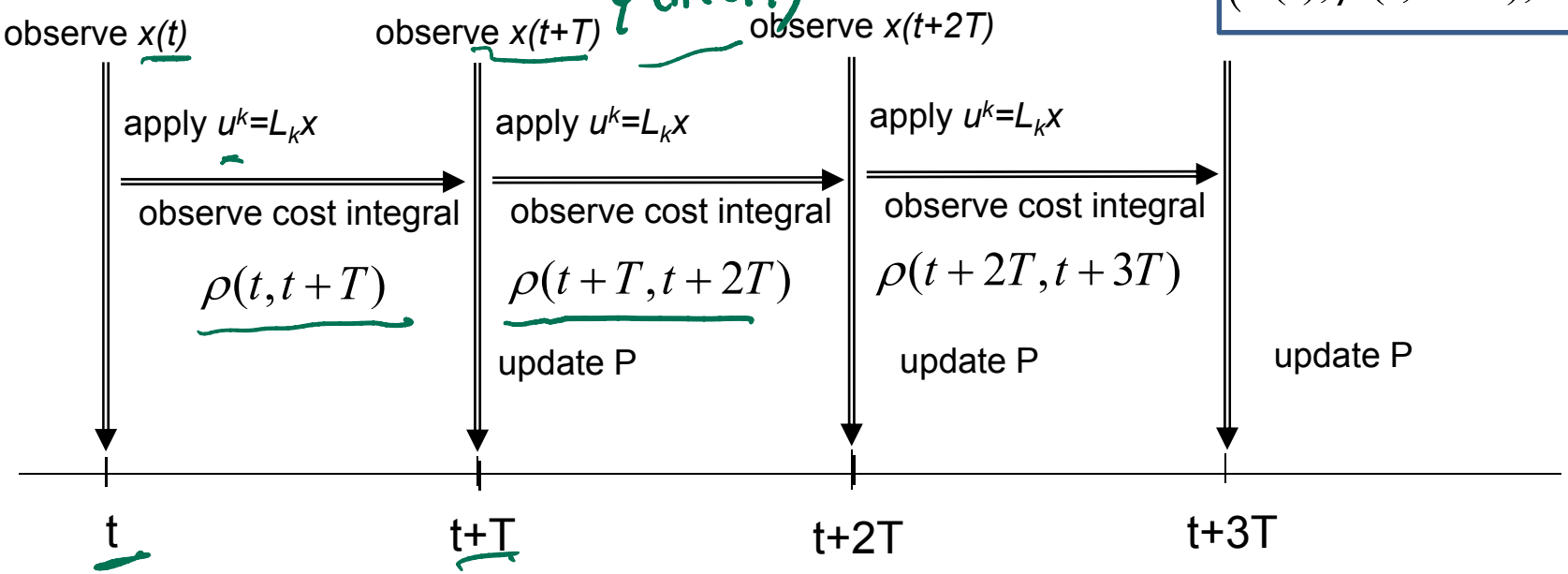
Integral Reinforcement Learning (IRL)

Solve Bellman Equation - Solves Lyapunov eq. without knowing dynamics

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

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

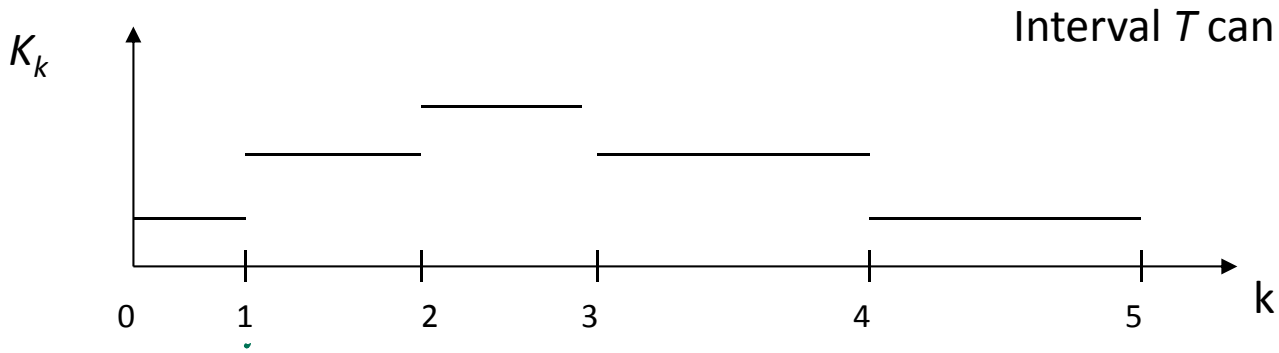
A is not needed anywhere

update control gain

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

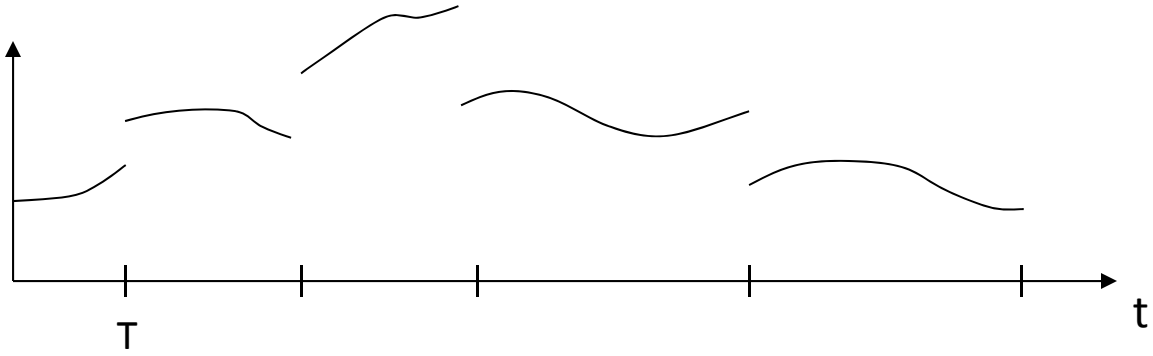
This is a data-based approach that uses measurements of *x(t), u(t)*
Instead of the plant dynamical model.

Gain update (Policy)



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

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

Regression vector must be PE

Relates to choice of reinforcement interval T

Implementation

Policy evaluation
Need to solve online

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

Add a new state= Integral Reinforcement

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

This is the controller dynamics or memory

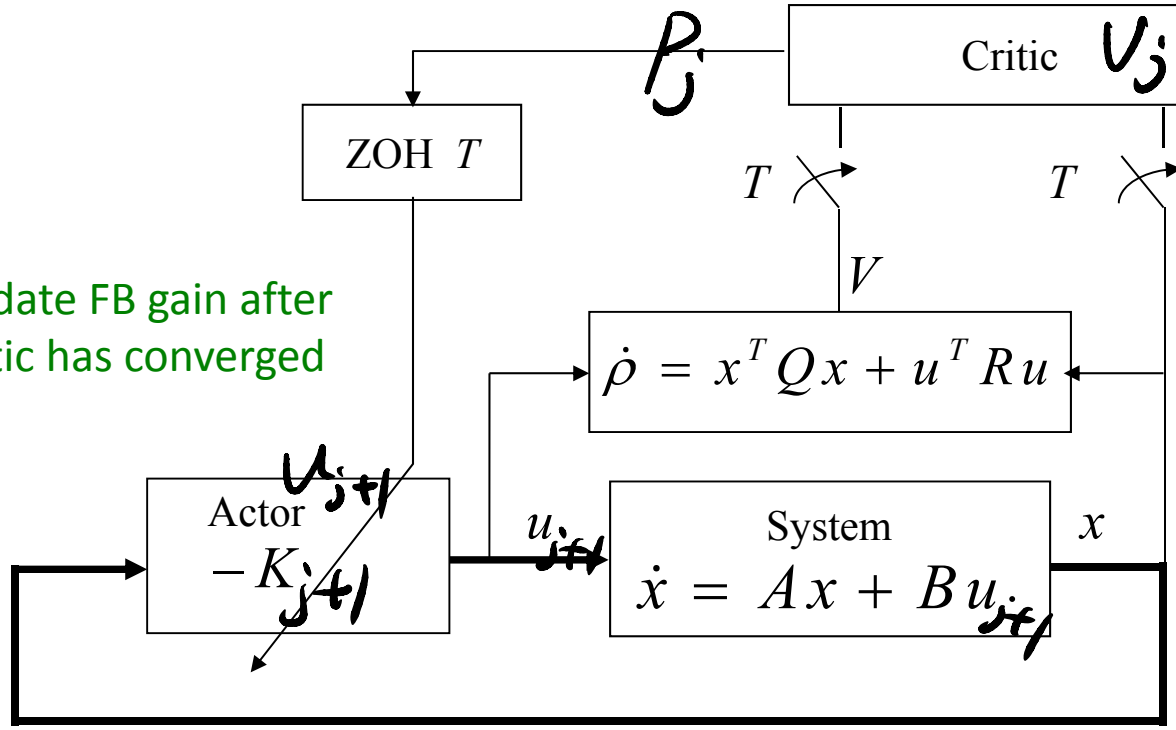
Draguna Vrabie

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



Dynamic Control System w/ MEMORY

Update FB gain after
Critic has converged

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

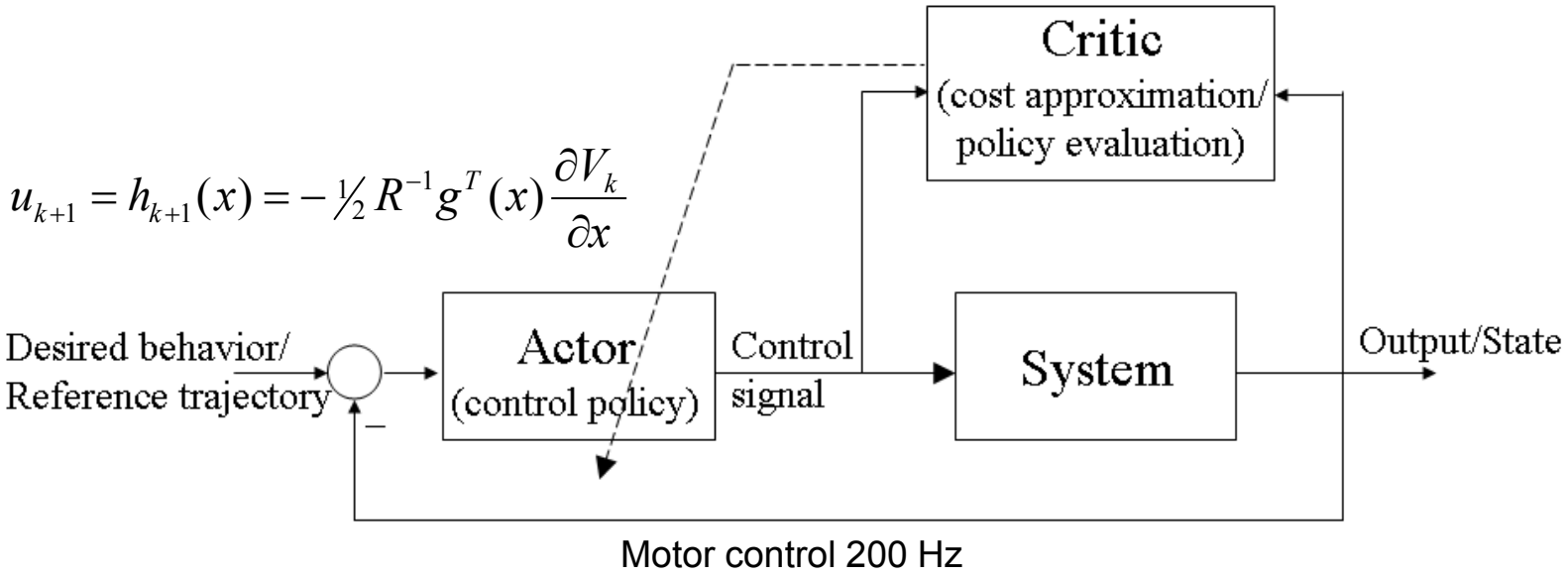
Optimal Adaptive IRL for CT systems

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



A new structure of adaptive controllers

The Bottom Line about Integral Reinforcement Learning

Off-line ARE Solution

The Optimal Control Solution

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

Algebraic Riccati equation

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

Full system dynamics must be known
Off-line solution

Data-driven On-line ARE Solution

ARE solution can be found online in real-time by using the IRL Algorithm without knowing A matrix

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

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

Iterate for *k=0,1,2,….*

CT Bellman eq.

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

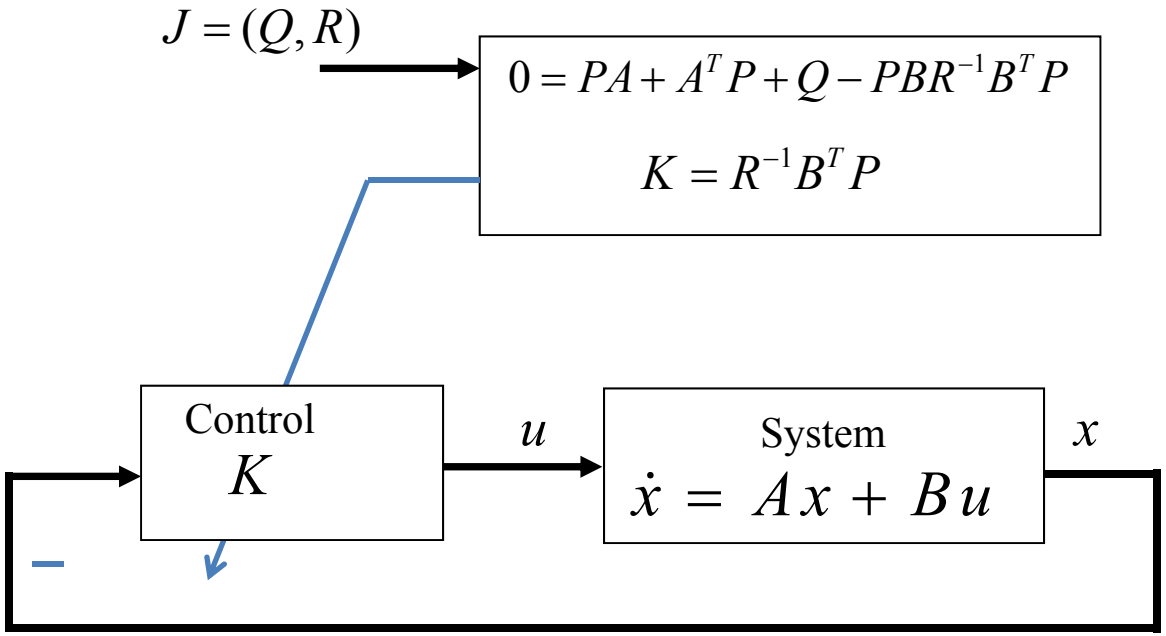
$$K_{k+1} = R^{-1}B^T P_k$$

$$u_{k+1}(t) = -K_{k+1}x(t)$$

Only B is needed
On line solution in real time
Uses data measurements along system trajectory

Optimal Control- The Linear Quadratic Regulator (LQR)

User prescribed optimization criterion



Off-line Design Loop

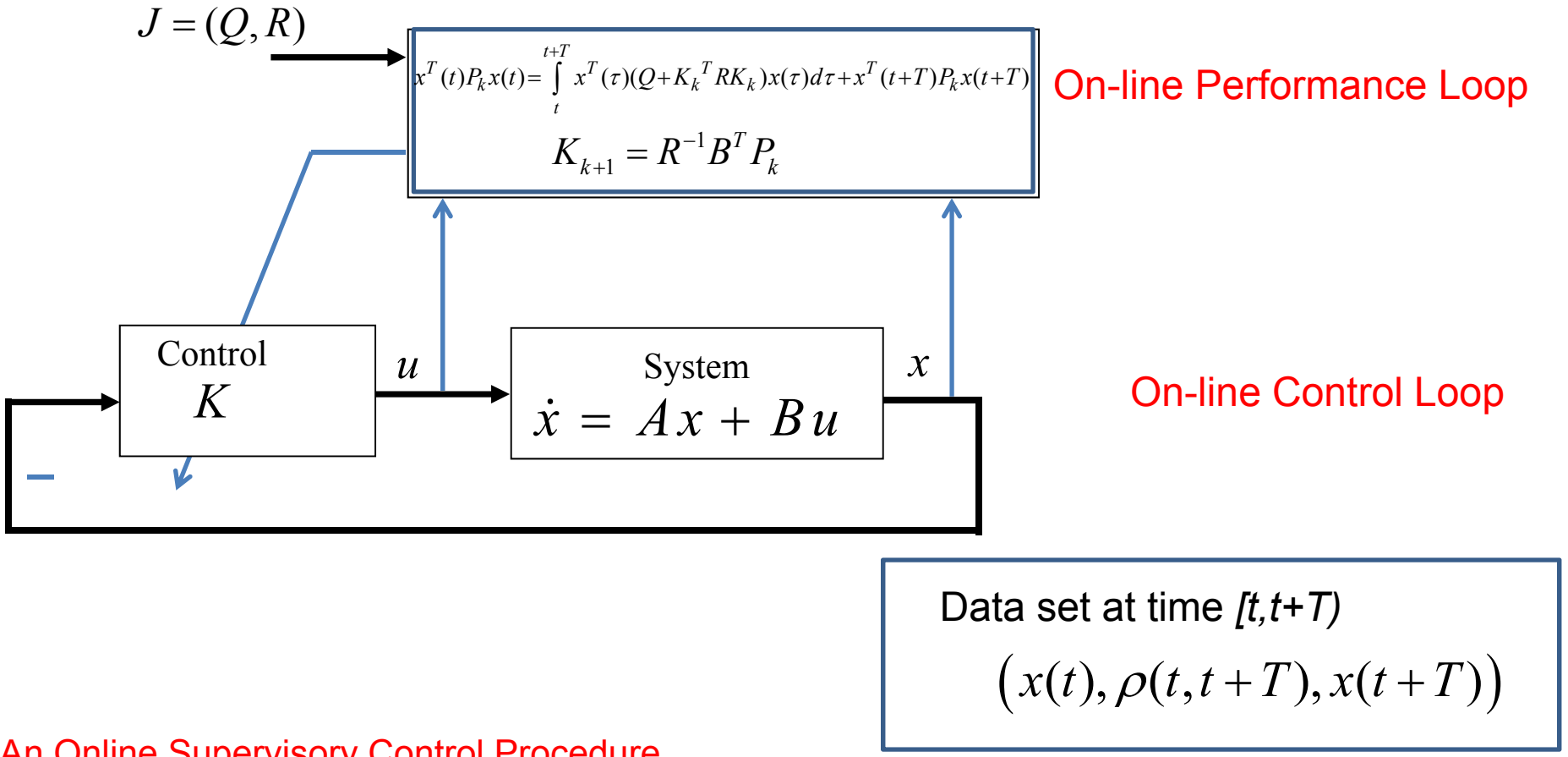
On-line Control Loop

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

System modeling is expensive, time consuming, and inaccurate

Data-driven Online Adaptive Optimal Control
DDO

User prescribed optimization criterion



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

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

$Q = I, \quad R = I$

$K = R^{-1} B^T P$

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

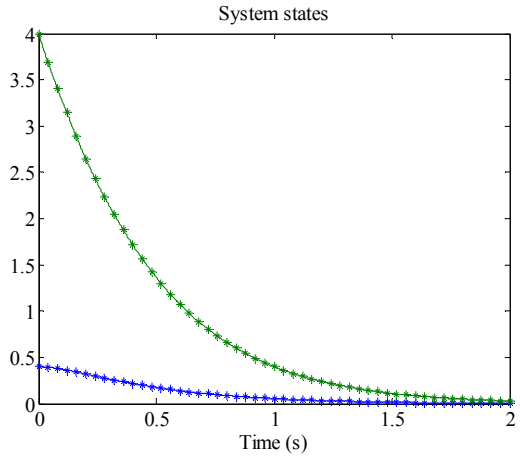
Select quadratic NN basis set for VFA

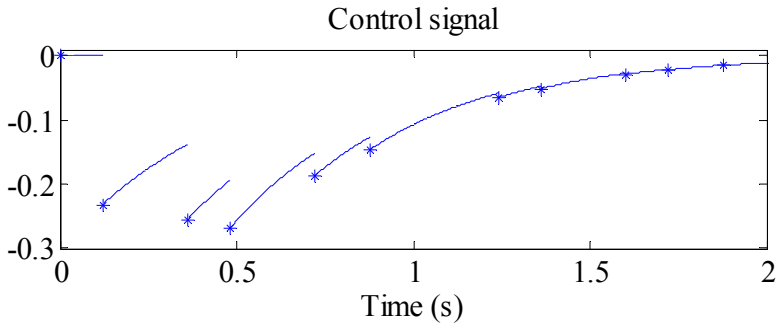
Exact solution $\quad W_1^* = [p_{11} \quad 2p_{12} \quad 2p_{13} \quad p_{22} \quad 2p_{23} \quad p_{33}]^T$

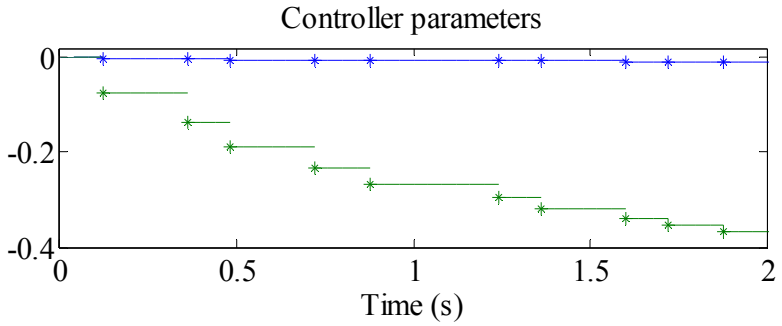
$= [1.4245 \quad 1.1682 \quad -0.1352 \quad 1.4349 \quad -0.1501 \quad 0.4329]^T$

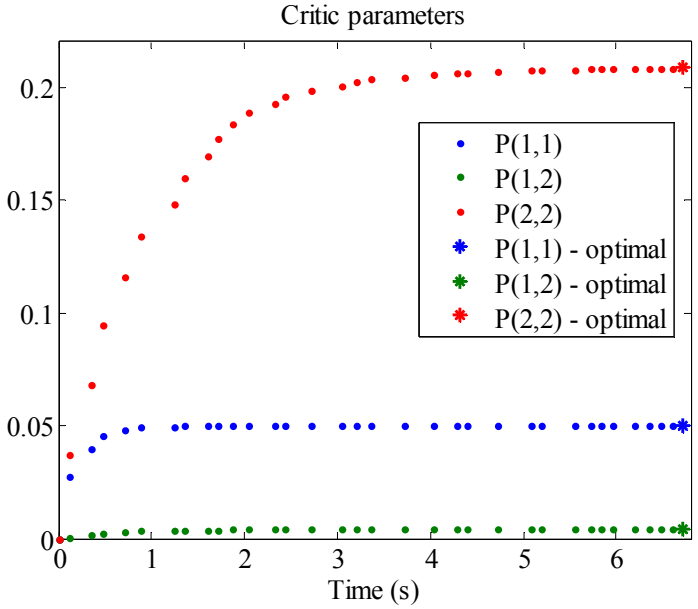
Simulations on: F-16 autopilot

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

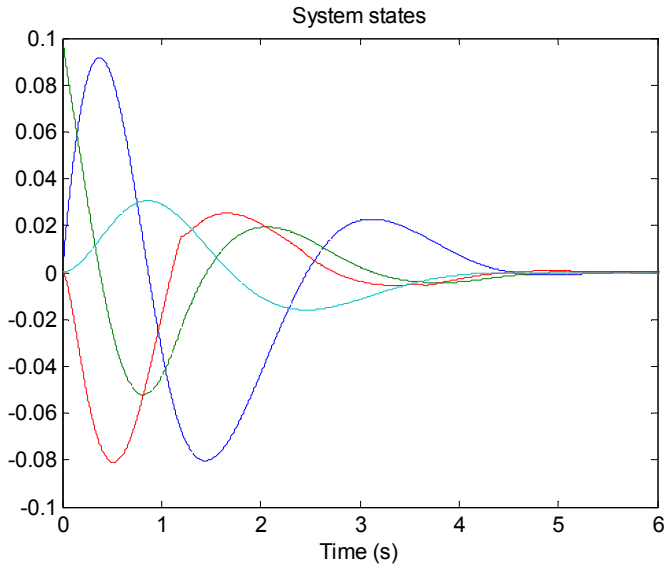
$$P_{ARE} = \begin{bmatrix} 0.4750 & 0.4766 & 0.0601 & 0.4751 \\ 0.4766 & 0.7831 & 0.1237 & 0.3829 \\ 0.0601 & 0.1237 & 0.0513 & 0.0298 \\ 0.4751 & 0.3829 & 0.0298 & 2.3370 \end{bmatrix}.$$

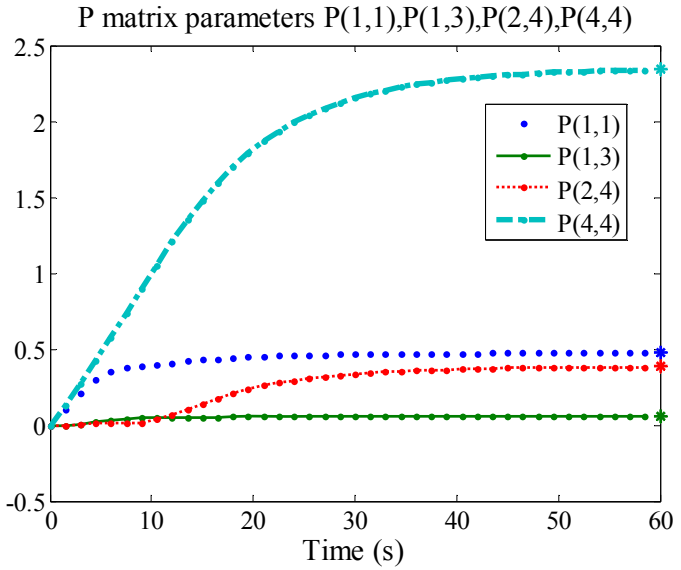
$$P_{critic\ NN} = \begin{bmatrix} 0.4802 & 0.4768 & 0.0603 & 0.4754 \\ 0.4768 & 0.7887 & 0.1239 & 0.3834 \\ 0.0603 & 0.1239 & 0.0567 & 0.0300 \\ 0.4754 & 0.3843 & 0.0300 & 2.3433 \end{bmatrix}.$$

IRL period of T= 0.1s.

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

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

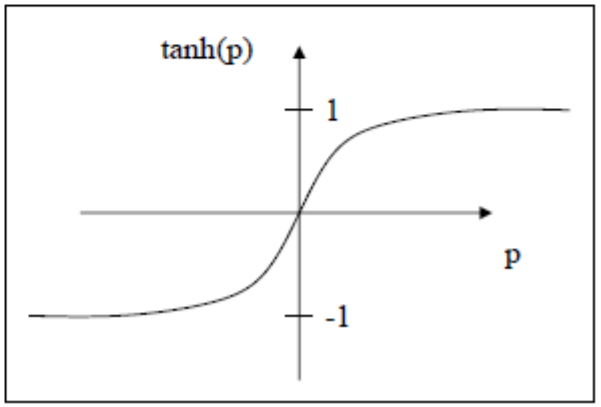
$$J = \int_0^T 1 \, dt = T$$

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

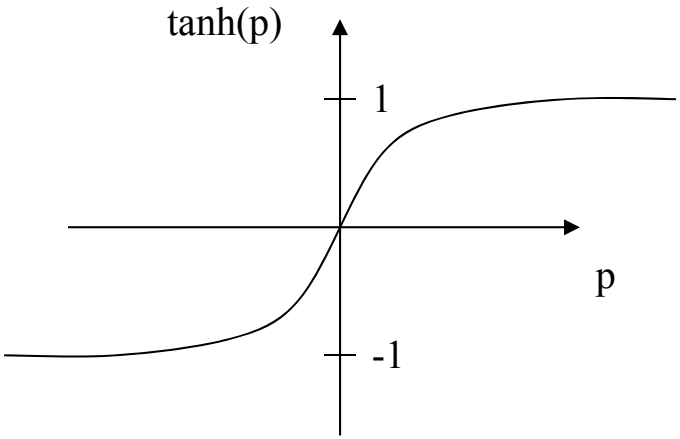
$$J = \frac{1}{2}\int_0^\infty \left(\tanh(x^T Q x) + \rho \int_0^u \sigma^{-1}(v) dv \right) dt$$



Murad Abu-Khalaf

Optimal Control for Constrained Input Systems

Control constrained by saturation function $\sigma(.)$



Encode constraint into Value function

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

$$\|u\|_q^2 = 2\int_0^u \sigma^{-T}(v)dv$$

(Used by Lyshevsky for H_2 control)

This is a quasi-norm

Weaker than a norm –
homogeneity property is replaced by the weaker symmetry property $\|x\|_q = \|-x\|_q$

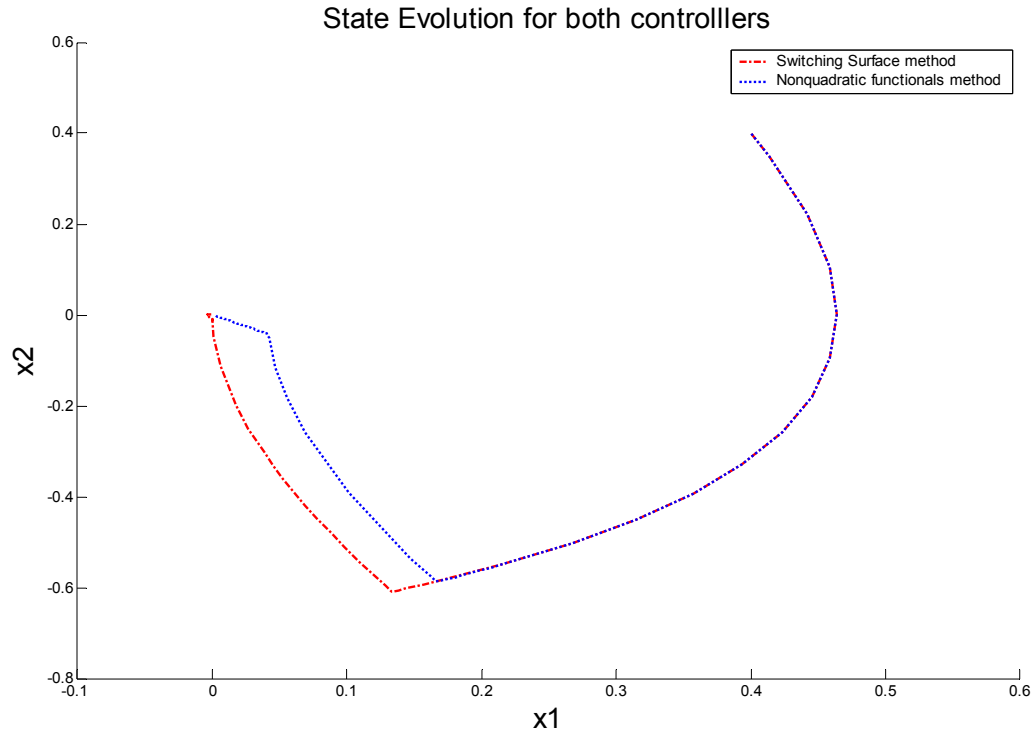
Then $u = -\sigma\left(R^{-1} g(x)^T \frac{\partial V}{\partial x} \right)$ Is BOUNDED

Murad Abu-Khalaf

Near Minimum-Time Control

Encode into Value Function

$$V = \int_0^{\infty} \left[\tanh(x^T Q x) + 2 \int_0^{u} \left(\sigma^{-1}(\mu) \right)^T R d\mu \right] dt$$



Do not need to find switching surface