Nonlinear Network Structures for Optimal Control

Cheng Tao & Frank L. Lewis
Advanced Controls & Sensors Group
Automation & Robotics Research Institute (ARRI)
The University of Texas at Arlington

Neural Network Solution for Fixed-Final Time Optimal Control of Nonlinear Systems

Cheng Tao
Frank L. Lewis

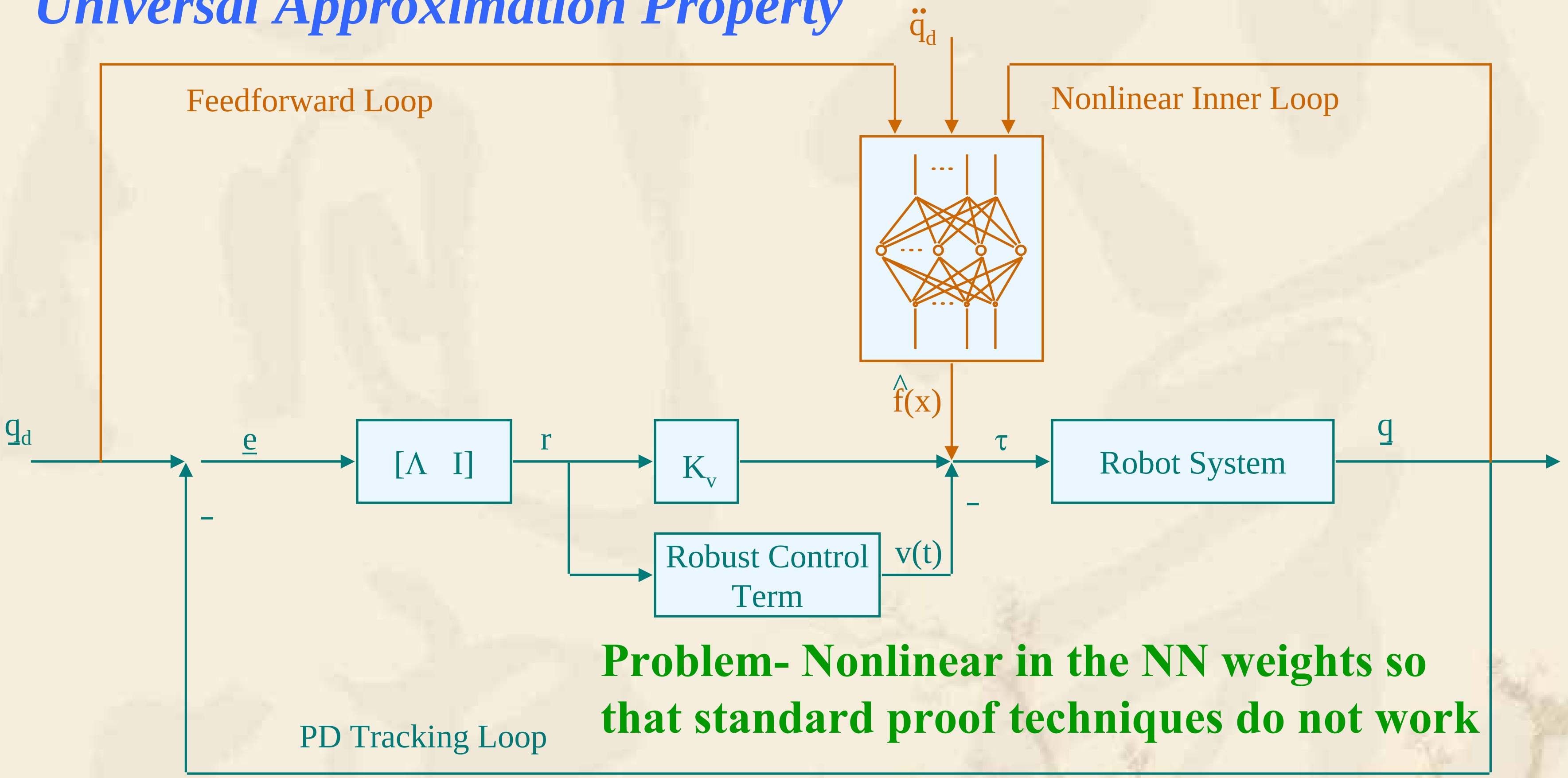
Automation & Robotics Research Institute (ARRI)
The University of Texas at Arlington

ECC 07 Kos

Neural Network Robot Controller

Universal Approximation Property

Feedback linearization



Problem- Nonlinear in the NN weights so that standard proof techniques do not work

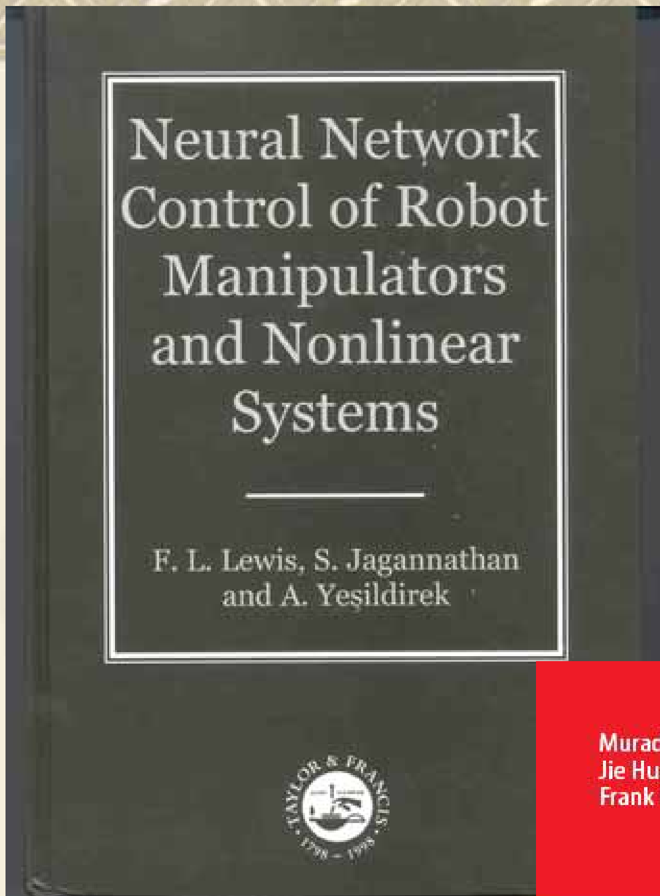
Easy to implement with a few more lines of code

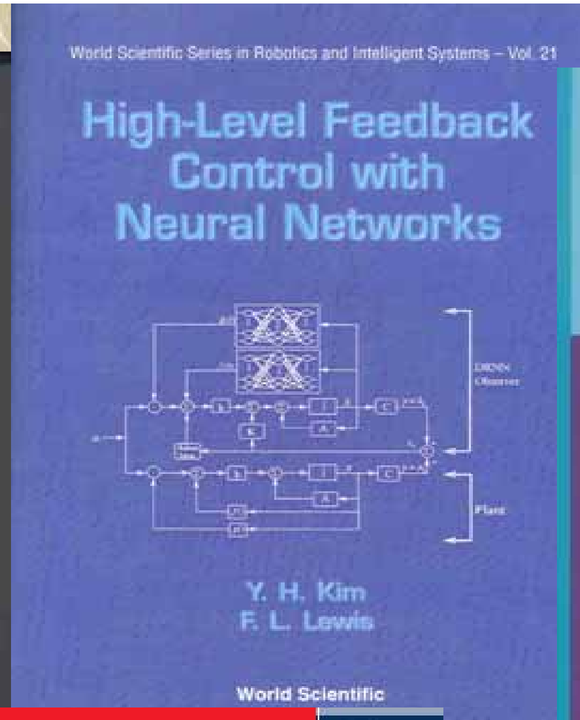
Learning feature allows for on-line updates to NN memory as dynamics change

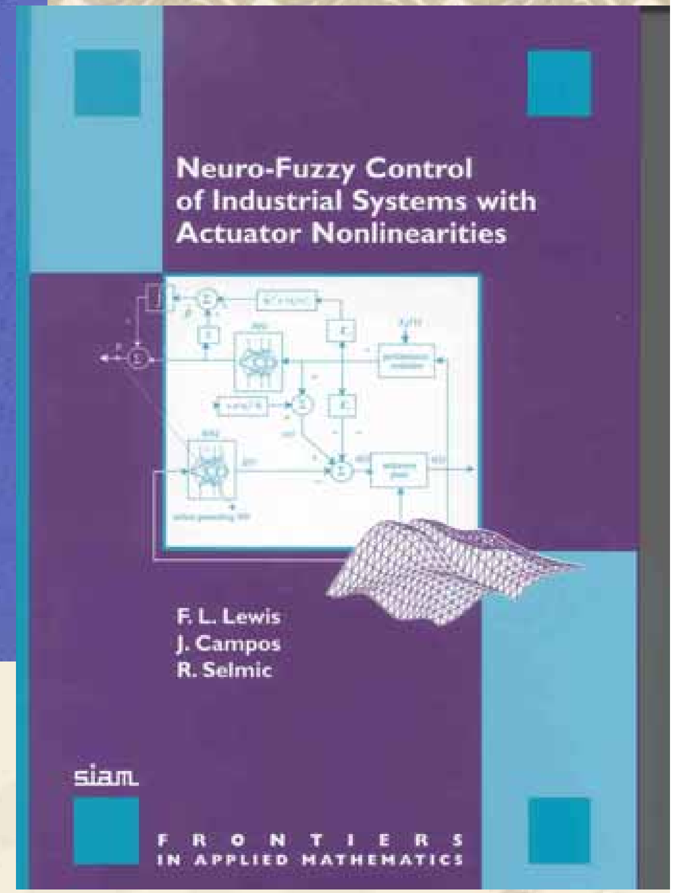
Handles unmodelled dynamics, disturbances, actuator problems such as friction

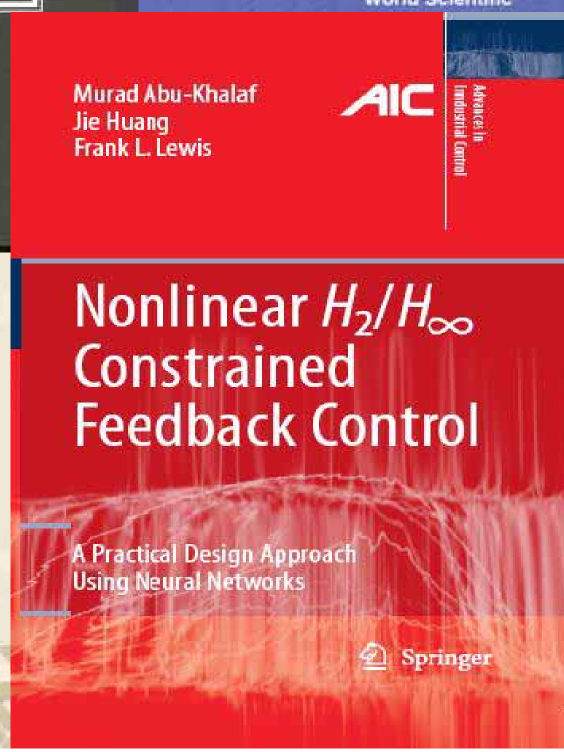
NN universal basis property means no regression matrix is needed

Nonlinear controller allows faster & more precise motion









4 US Patents

Sponsored by
NSF- Paul Werbos
ARO- Randy Zachery

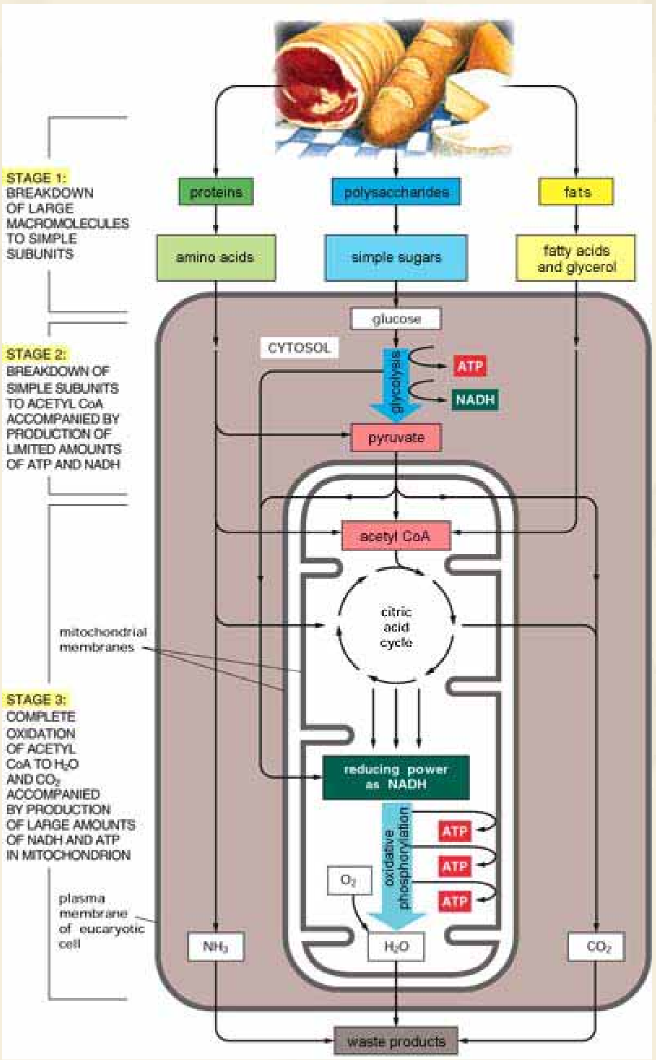
Problem- Nonlinear in the NN weights so that standard proof techniques do not work

New book by Jay Farrell and Marios Polycarpou

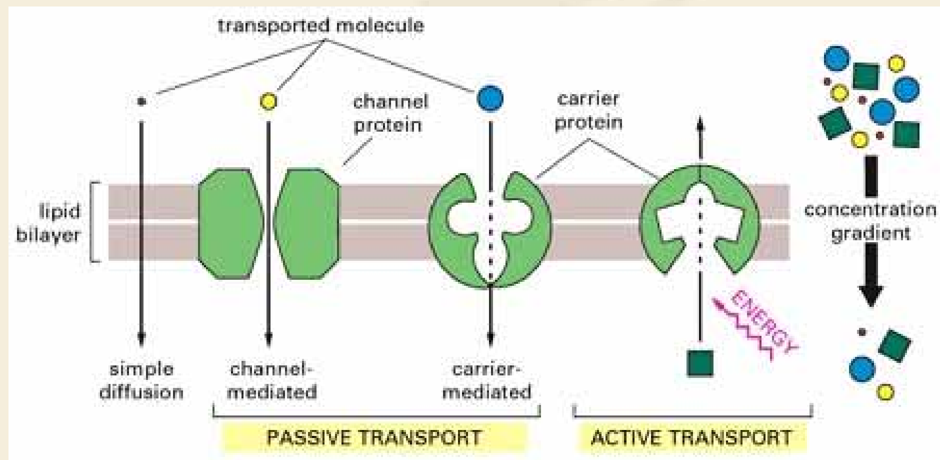
Adaptive Approximation Based Control

Optimality in Biological Systems

Cell Homeostasis



The individual cell is a complex feedback control system. It pumps ions across the cell membrane to maintain homeostatis, and has only limited energy to do so.



Permeability control of the cell membrane

http://www.accessexcellence.org/RC/VL/GG/index.html

ARRI Research Roadmap in Neural Networks

3. Approximate Dynamic Programming – 2006-

Nearly Optimal Control
Based on recursive equation for the optimal value
Usually Known system dynamics (except Q learning)
The Goal – unknown dynamics
On-line tuning
Optimal Adaptive Control

Extend adaptive control to yield OPTIMAL controllers.
No canonical form needed.

2. Neural Network Solution of Optimal Design Equations – 2002-2006

Nearly Optimal Control
Based on HJ Optimal Design Equations
Known system dynamics
Preliminary Off-line tuning

Nearly optimal solution of controls design equations.
No canonical form needed.

1. Neural Networks for Feedback Control – 1995-2002

Based on FB Control Approach
Unknown system dynamics
On-line tuning
NN- FB lin., sing. pert., backstepping, force control, dynamic inversion, etc.

Extended adaptive control to NLIP systems
No regression matrix

Objective and Significance

- Provide a tool to solve finite-horizon continuous-time optimal control problems for nonlinear systems.

- Continuous time finite horizon optimal control problems appear applications in which people use model predictive control (receding horizon control).

Outline:

1. **Fixed-Final Time Optimal Control of Nonlinear Systems Using Neural Network HJB Approach**

2. **Neural Network Solution for Finite-Final Time H-Infinity State Feedback Control**

3. **Neural Network Solution for Fixed-Final time Constrained Optimal Control**

- **This research was supported by NSF grant ECS-0140490 and ARO grant DAAD 19-02-1-0366.**

Review of Related work and Motivation

Approximate HJB solutions

Munos et. al [65]
(Gradient descent approaches)

Kim, Lewis and Dawson [47]
(NNs)

Huang and Lin [44]
(Taylor series expansion)

NN applications to an optimal control

Miller [63]
(NNs for control)

Parisini and Zoppoli [70]
(Infinite horizon)

Constrained-input optimization

Sussmann, Sontag and yang [84]

Bernstein [15]

Dolphus [33]

Abu-Khalaf, M [1]
(Infinity horizon)

Unconstrained policy iteration with finite-time horizon

Beard[11]

Background on Fixed-Final-Time HJB Optimal Control

Nonlinear dynamical system

$$\dot{x} = f(x) + g(x)u(t) \tag{1}$$

where $x \in \Re^n$, $f(x) \in \Re^n$, $g(x) \in \Re^{n \times m}$ and the input $u(t) \in R^m$

It is desired to find the control u that minimizes a generalized nonquadratic functional

$$V(x(t_0),t_0) = \phi(x(t_f),t_f) + \int_{t_0}^{t_f} [Q(x) + W(u)]dt \tag{2}$$

with $Q(x)$, $W(u)$ positive definite on Ω

Background on Fixed-Final-Time HJB Optimal Control

An infinitesimal equivalent to (2) is

$$-\frac{\partial V(x,t)}{\partial t} = L + \left(\frac{\partial V(x,t)}{\partial x}\right)^T \left(f(x) + g(x)u(t)\right) \qquad (3)$$

where $L = Q(x) + W(u)$. This is a time-varying partial differential equation with $V(x,t)$ the cost function for any given $u(t)$ and is solved backwards in time from $t = t_f$.

By setting $t_0 = t_f$ in (2) its boundary condition is

$$V(x(t_f), t_f) = \phi(x(t_f), t_f) \qquad (4)$$

Background on Fixed-Final-Time HJB Optimal Control

According to Bellman's optimality principle, the optimal cost is given by

$$-\frac{\partial V(x,t)^*}{\partial t} = \min_{u(t)}\left(L + \left(\frac{\partial V(x,t)^*}{\partial x}\right)^T \left(f(x) + g(x)u(x)\right)\right) \qquad (5)$$

which yields the optimal control

$$u^*(x) = -\frac{1}{2} R^{-1} g(x)^T \frac{\partial V(x,t)^*}{\partial x} \qquad (6)$$

where $V^*(x,t)$ is the optimal value function.

Substituting (6) into (5) yields the well-known time-varying Hamilton-Jacobi-Bellman (HJB) equation

$$\frac{\partial V(x,t)^*}{\partial t} + \frac{\partial V(x,t)^*}{\partial x} f(x) + Q(x) - \frac{1}{4}\frac{\partial V(x,t)^{*T}}{\partial x} g(x) R^{-1} g(x)^T \frac{\partial V(x,t)^*}{\partial x} = 0 \qquad (7)$$

Background on Fixed-Final-Time HJB Optimal Control

Then (5) becomes

$$HJB\left(V(x,t)^*\right) = \frac{\partial V(x,t)^*}{\partial t} + \frac{\partial V(x,t)^*}{\partial x} f(x) + Q(x)$$
$$-\frac{1}{4}\frac{\partial V(x,t)^{*T}}{\partial x} g(x) R^{-1} g(x)^T \frac{\partial V(x,t)^*}{\partial x} = 0 \qquad (8)$$

If this HJB equation can be solved for the value function $V(x,t)$, then the optimal control is

$$u^*(x) = -\frac{1}{2} R^{-1} g(x)^T \frac{\partial V(x,t)^*}{\partial x}$$

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

NN Approximation of the Cost Function $V(x,t)$

In Sandberg [78], it is shown that NNs with time-varying weights can be used to uniformly approximate continuous time-varying functions.

Using the following equation to approximate $V(x,t)$ for $t \in [t_0, t_f]$ on a compact set $\Omega \subset \Re^n$

$$V_L(x,t) = \sum_{j=1}^{L} w_j(t)\sigma_j(x) = w_L^T(t)\sigma_L(x) \qquad (9)$$

The NN weights are $w_j(t)$ and L is the number of hidden-layer neurons.

$\boldsymbol{\sigma}_L(x) \equiv [\sigma_1(x)\sigma_2(x)\ldots\sigma_L(x)]^T$ is the vector of activation function.

$\mathbf{w}_L(t) \equiv [w_1(t)w_2(t)\ldots w_L(t)]^T$ is the vector of NN weights.

Note:

The set $\sigma_j(x)$ is selected to be independent. Then without loss of generality, they can be assumed to be orthonormal, i.e. select equivalent basis functions to $\sigma_j(x)$ that are also orthonormal. The orthonormality of the set $\{\sigma_j(x)\}_1^\infty$ on Ω implies that if a function $\psi(x,t) \in L_2(\Omega)$ then

$$\psi(x,t) = \sum_{i=1}^{\infty} \langle \psi(x,t), \sigma_j(x) \rangle_\Omega \sigma_j(x)$$

where $\langle f, g \rangle_\Omega = \int_\Omega f \cdot g dx$ is inner product.

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

Note that

$$\frac{\partial V_L(x,t)}{\partial x} = \frac{\partial \boldsymbol{\sigma}_L^T(x)}{\partial x}\mathbf{w}_L(t) \equiv \nabla\boldsymbol{\sigma}_L^T(x)\mathbf{w}_L(t) \qquad (10)$$

where $\nabla\boldsymbol{\sigma}_L(x)$ is the Jacobian $\partial\boldsymbol{\sigma}_L(x)/\partial x$, and that

$$\frac{\partial V_L(x,t)}{\partial t} = \dot{\mathbf{w}}_L^T(t)\boldsymbol{\sigma}_L(x) \qquad (11)$$

Therefore approximating $V(x,t)$ by $V_L(x,t)$ uniformly in in the HJB equation (8) results in

$$\begin{aligned}&-\dot{\mathbf{w}}_L^T(t)\boldsymbol{\sigma}_L(x) - \mathbf{w}_L^T(t)\nabla\boldsymbol{\sigma}_L(x)f(x)\\ &+\frac{1}{4}\mathbf{w}_L^T(t)\boldsymbol{\sigma}_L(x)g(x)R^{-1}g^T(x)\boldsymbol{\sigma}_L^T(x)\mathbf{w}_L(t) \qquad (12)\\ &-Q(x) = e_L(x,t)\end{aligned}$$

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

or

$$HJB\left(V_L(x,t)=\sum_{j=1}^{L} w_j(t)\sigma_j(x)\right)=e_L(x,t) \qquad (13)$$

where $e_L(x,t)$ is a residual equation error. The corresponding optimal control input is

$$u^*(x)=-\frac{1}{2}R^{-1}g(x)^T\frac{\partial V(x,t)^*}{\partial x} \quad = R^{-1}g^T\nabla\boldsymbol{\sigma}_L^T(x)\mathbf{w}_L(t) \qquad (14)$$

To find the least-squares solution for $\mathbf{w}_L(t)$, the method of weighted residuals is used

$$\left\langle \frac{\partial e_L(x,t)}{\partial \dot{\mathbf{w}}_L(t)}, e_L(x,t)\right\rangle_{\Omega} = 0$$

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

$$
\begin{aligned}
&\dot{\mathbf{w}}_L(t) = \\
&-\left\langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x) \right\rangle_\Omega^{-1} \cdot \left\langle \nabla\boldsymbol{\sigma}_L(x) f(x), \boldsymbol{\sigma}_L(x) \right\rangle_\Omega \cdot \mathbf{w}_L(t) \\
&+\left\langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x) \right\rangle_\Omega^{-1} \cdot \left\langle \begin{matrix} \frac{1}{4}\mathbf{w}_L^T(t)\nabla\boldsymbol{\sigma}_L(x) g(x) R^{-1} \\ \cdot g^T(x)\nabla\boldsymbol{\sigma}_L^T \mathbf{w}_L(t), \boldsymbol{\sigma}_L(x) \end{matrix} \right\rangle_\Omega \\
&-\left\langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x) \right\rangle_\Omega^{-1} \cdot \left\langle Q(x), \boldsymbol{\sigma}_L(x) \right\rangle_\Omega
\end{aligned}
\qquad (15)
$$

Boundary condition $\quad V\left(x(t_f), t_f\right) = \phi\left(x(t_f), t_f\right) = \mathbf{w}_L^T(t_f)\boldsymbol{\sigma}_L\left(x(t_f)\right)$

Lemmas Proved:

Lemma 1: Convergence of **Approximate HJB Equation**

Lemma 2: Convergence of **NN Weights**

Lemma 3: Convergence of **Approximate Value Function**

Lemma 4: Convergence of **Value Function Gradient**

Lemma 5: Convergence of **Control Inputs**

Lemma 6: Convergence of **State Trajectory**

Lemma 1: Convergence of Approximate HJB Equation.

Let $V_L(x,t) = \sum_{j=1}^{L} w_j^T(t)\sigma_j(x)$ satisfy $\langle HJB(V_L(x,t)), \boldsymbol{\sigma}_L(x)\rangle_\Omega = 0$ and $\langle V_L(t_f), \boldsymbol{\sigma}_L\rangle_\Omega = 0$

Let $V(x,t) = \sum_{j=1}^{\infty} c_j^T(t)\sigma_j(x)$ and $\mathbf{c}_L(t) \equiv [c_1(t)c_2(t)\ldots c_L(t)]^T$ satisfy

$HJB(V(x,t)) = 0$ and $V(x,t_f) = \phi(x(t_f),t_f)$

Then $\left|HJB(V_L(x,t))\right| \to 0$ on Ω as L increases.

Outline of proof of Lemma 1

1. Calculate $\left\langle HJB(V_L(x,t)), \sigma_j(x) \right\rangle_\Omega$ using eq. (13).

2. Apply $\left\langle \sigma_k, \sigma_j \right\rangle_\Omega = 0$ if $k \neq j$ to simplify $\left\langle HJB(V_L(x,t)), \sigma_j(x) \right\rangle_\Omega$.

3. Prove $\left| HJB(V_L(x,t)) \right| \to 0$ as L increases,

   Here $\left| HJB(V_L(x,t)) \right| = \left| \sum_{j=1}^{\infty} \left\langle HJB(V_L(x,t)), \sigma_j(x) \right\rangle_\Omega \sigma_j(x) \right|$.

Optimal Algorithm Based on NN Approximation

A mesh of points over the integration region can be introduced on Ω_0

$$A = \left\lfloor \boldsymbol{\sigma}_L(x)|_{x_1} \; \ldots\ldots \boldsymbol{\sigma}_L(x)|_{x_p} \right\rfloor^T$$

$$B = \left\lfloor \boldsymbol{\sigma}_L(x) f(x)|_{x_1} \; \ldots\ldots \boldsymbol{\sigma}_L(x) f(x)|_{x_p} \right\rfloor^T$$

$$C = \begin{bmatrix} \frac{1}{4}\left(\nabla\boldsymbol{\sigma}_L(x) g(x) R^{-1} g^T(x) \nabla\boldsymbol{\sigma}_L^T(x)\right)|_{x_1} \; \ldots\ldots \\ \frac{1}{4}\left(\nabla\boldsymbol{\sigma}_L(x) g(x) R^{-1} g^T(x) \nabla\boldsymbol{\sigma}_L^T(x)\right)|_{x_p} \end{bmatrix}^T$$

$$D = \left\lfloor Q(x)|_{x_1} \ldots Q(x)|_{x_p} \right\rfloor^T$$

where p in x_p represents the number of points of the mesh.

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

$$-A^T A \cdot \dot{\mathbf{w}}_L(t) - A^T B \cdot \mathbf{w}_L(t) + A^T \mathbf{w}_L^T(t) C \mathbf{w}_L(t) - A^T D = 0 \qquad (16)$$

then

$$\begin{aligned}\dot{\mathbf{w}}_L(t) &= -(A^T A)^{-1} \mathbf{w}_L(t) A^T B \\ &+ (A^T A)^{-1} A^T \mathbf{w}_L^T(t) C \mathbf{w}_L(t) - (A^T A)^{-1} A^T D\end{aligned} \qquad (17)$$

This is a nonlinear ODE that can easily be integrated backwards using final condition $\mathbf{w}_L(t_f)$ to find the least-squares optimal NN weights.

Numerical Examples

a) Linear System

$$\dot{x}_1 = 2x_1 + 3x_2 + u_1$$
$$\dot{x}_2 = 5x_1 + 6x_2 + 2u_2 \qquad (18)$$

To find a nearly optimal time-varying controller, the following smooth function is used to approximate the value function of the system

$$V(x_1, x_2) = w_1 x_1^2 + w_2 x_1 x_2 + w_3 x_2^2 \qquad (19)$$

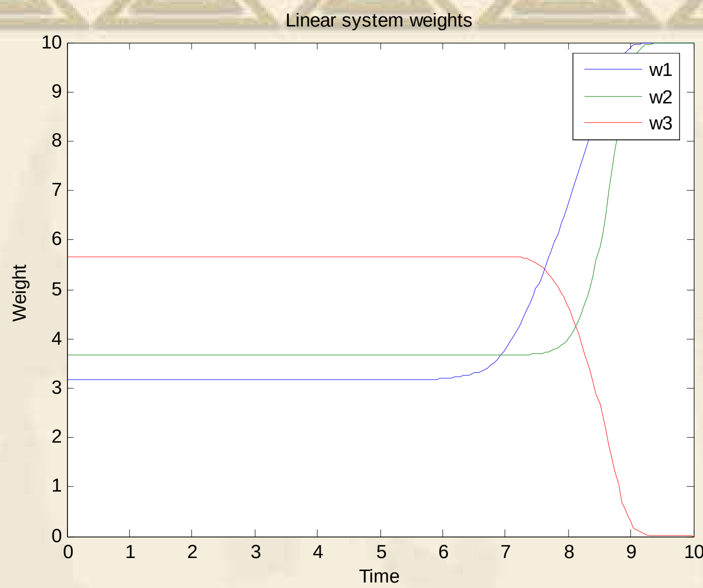


Fig. 1 Linear System Weights

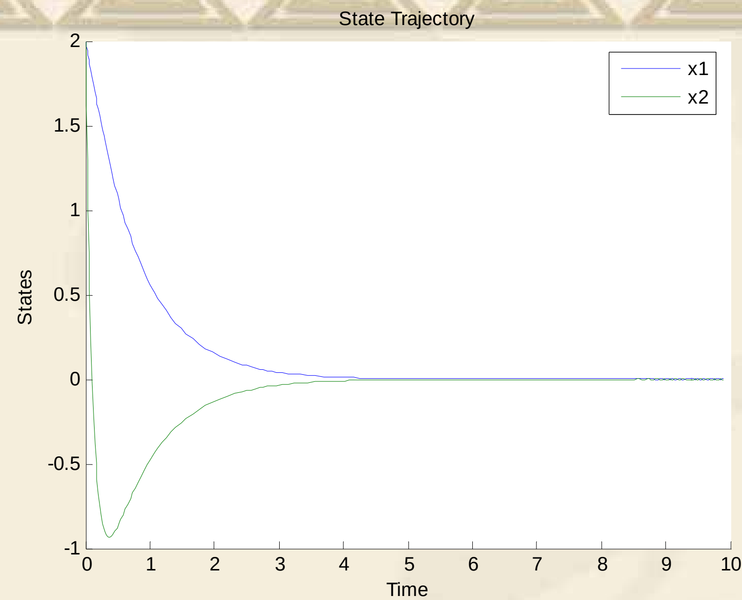


Fig. 2 State Trajectory of Linear System

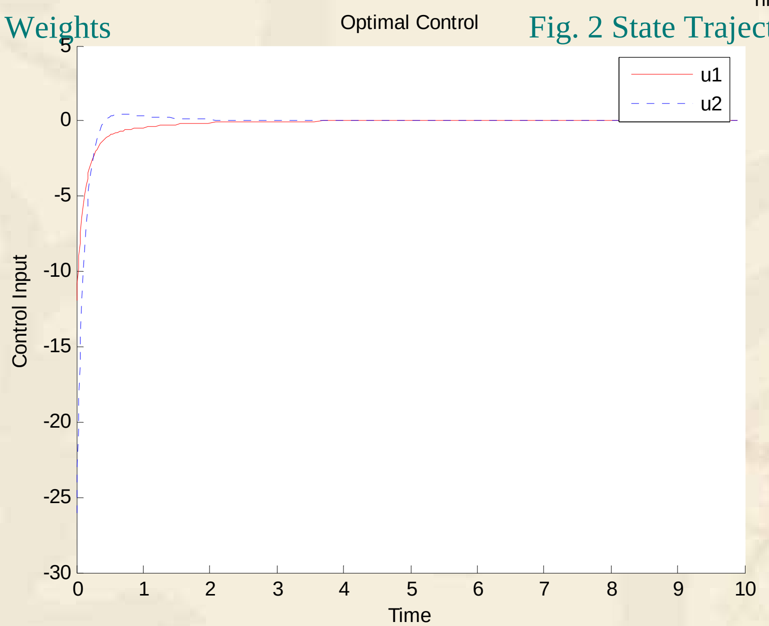


Fig. 3 Optimal NN Control Law

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

b) Nonlinear Chained System

$$\begin{aligned}\dot{x}_1 &= u_1 \\ \dot{x}_2 &= u_2 \\ \dot{x}_3 &= x_1 u_2\end{aligned} \qquad (20)$$

Smooth approximating function

$$\begin{aligned}V(x_1, x_2, x_3) &= w_1 x_1^2 + w_2 x_2^2 + w_3 x_3^2 + w_4 x_1 x_2 + w_5 x_1 x_3 + w_6 x_2 x_3 \\ &+ w_7 x_1^4 + w_8 x_2^4 + w_9 x_3^4 + w_{10} x_1^2 x_2^2 + w_{11} x_1^2 x_3^2 + w_{12} x_2^2 x_3^2 + w_{13} x_1^2 x_2 x_3 \\ &+ w_{14} x_1 x_2^2 x_3 + w_{15} x_1 x_2 x_3^2 + w_{16} x_1^3 x_2 + w_{17} x_1^3 x_3 + w_{18} x_1 x_2^3 + w_{19} x_1 x_3^3 \\ &+ w_{20} x_2 x_3^3 + w_{21} x_2^3 x_3\end{aligned} \qquad (21)$$

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

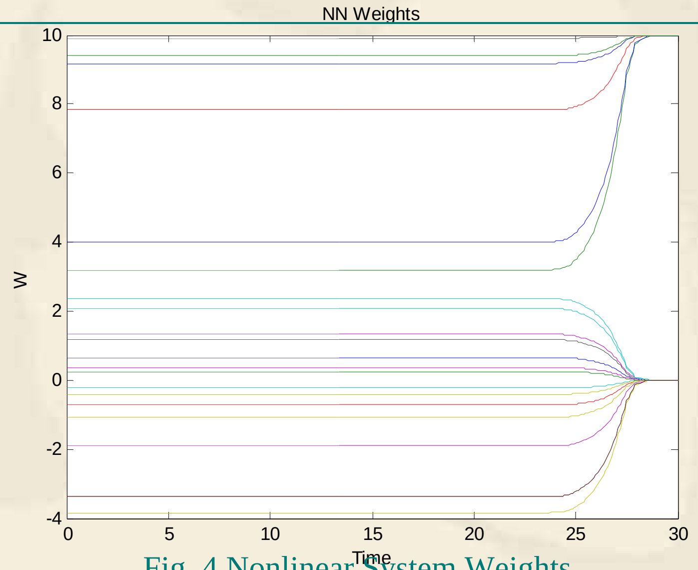


Fig. 4 Nonlinear System Weights

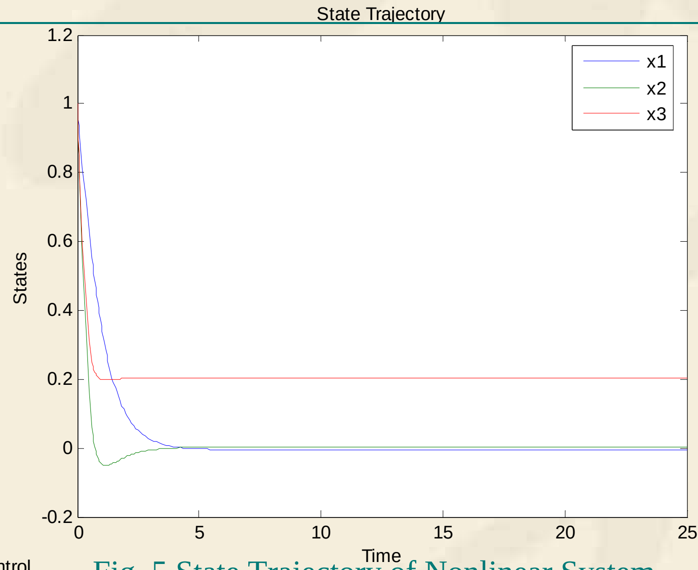


Fig. 5 State Trajectory of Nonlinear System

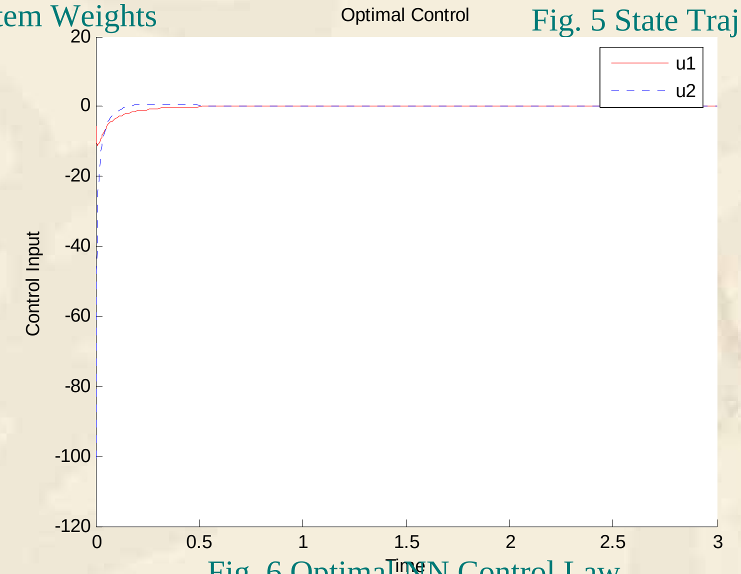


Fig. 6 Optimal NN Control Law

The University of Texas ARLINGTON

ARRI

Fixed-Final-Time HJI Optimal Control

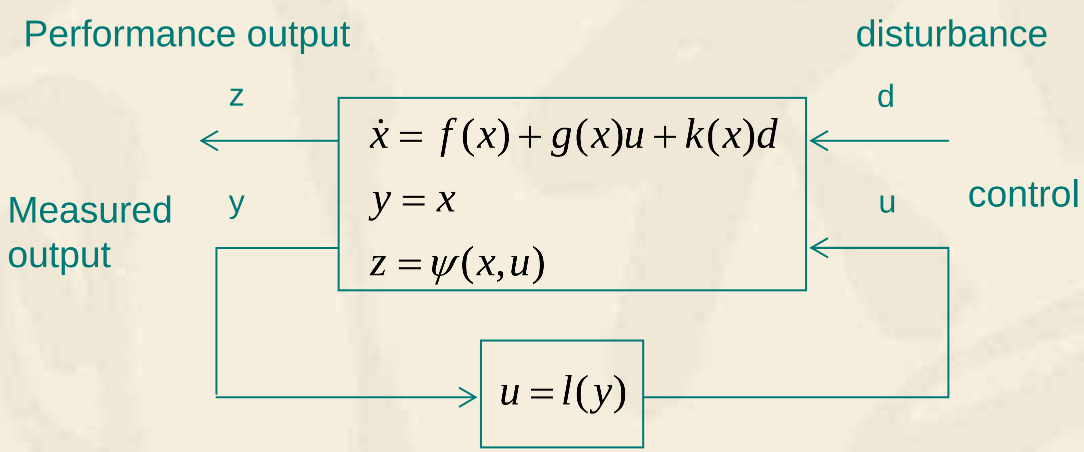
1. Based on solving a related Hamilton-Jacobi-Isaacs equation of the corresponding finite-horizon zero-sum game.

2. The game value function is approximated by a neural network with time-varying weights.

3. It is shown that the neural network approximation converges uniformly to the game-value function and the resulting nearly optimal constrained feedback controller provides closed-loop stability and bounded L_2 gain.

H-Infinity Control Using Neural Networks

System



where

$$\|z\|^2 = h^T h + \|u\|^2$$

L_2 Gain Problem

Find control *u(t)* so that

$$\frac{\int_0^\infty \|z(t)\|^2 dt}{\int_0^\infty \|d(t)\|^2 dt} = \frac{\int_0^\infty (h^T h + \|u\|^2) dt}{\int_0^\infty \|d(t)\|^2 dt} \le \gamma^2$$

For all L_2 disturbances
And a prescribed gain γ^2

Zero-Sum differential Nash game

Hamiltonian function

$$H(x,p,u,d)=\frac{\partial V^T(x,t)}{\partial x}\left(f(x)+g(x)u(t)+k(x)d(t)\right)+h^T(x)h(x)+\|u(t)\|^2-\gamma^2\|d(t)\|^2$$

Minimizing the Hamiltonian of the optimal control problem with regard to u and d gives

$$g^T(x)\frac{\partial V(x,t)}{\partial x}+2u^*(t)=0$$

$$k^T(x)\frac{\partial V(x,t)}{\partial x}-2\gamma^2 d^*(t)=0$$

$$u^*(t)=-\frac{1}{2}g(x)^T\frac{dV(x,t)}{dx}$$

$$d^*(t)=\frac{1}{2\gamma^2}k(x)^T\frac{dV(x,t)}{dx}$$

Hamilton-Jacobi-Isaacs equation

$$\frac{\partial V(x,t)}{\partial t}+\frac{\partial V^T(x,t)}{\partial x}\left(f(x)+g(x)u^*(t)+k(x)d^*(t)\right)$$
$$+h^T(x)h(x)+\left\|u^*(t)\right\|^2-\gamma^2\left\|d^*(t)\right\|^2=0$$

So

$$HJI\left(V^*(x,t)\right)=\frac{\partial V^*(x,t)}{\partial t}+\frac{\partial V^{*T}(x,t)}{\partial x}f$$
$$-\frac{\partial V^{*T}(x,t)}{\partial x}\hat{g}(x)\hat{g}(x)^T\frac{\partial V^*(x,t)}{\partial x}+h^T(x)h(x)=0$$

Boundary condition $V\left(x(t_f),t_f\right)=\phi\left(x(t_f),t_f\right)$

Here $\hat{g}(x)\hat{g}(x)^T=\frac{1}{4}g(x)g(x)^T-\frac{1}{4\gamma^2}k(x)k(x)^T$

Cannot solve HJI !!

Infinite Horizon Case

Successive Solution- Algorithm 1:

Let γ be prescribed and fixed.

Murad Abu Khalaf

u_0 *a stabilizing control with region of asymptotic stability* Ω_0

1. *Outer loop- update control*

   Initial disturbance $d^0 = 0$

   2. *Inner loop- update disturbance*

   Solve Value Equation

Consistency equation For Value Function →

$$\frac{\partial (V^i{}_j)}{\partial x}^T \left(f + gu_j + kd\right) + h^T h + 2\int_0^{u_j} \phi^{-T}(v)\,dv - \gamma^2 (d^i)^T d^i = 0$$

Inner loop update disturbance

$$d^{i+1} = \frac{1}{2\gamma^2} k^T(x) \frac{\partial V^i{}_j}{\partial x}$$

go to 2.

Iterate i until convergence to $d^\infty, V^\infty{}_j$ *with RAS* $\Omega^\infty{}_j$

Outer loop update control action

$$u_{j+1} = -\tfrac{1}{2}\phi\left(g^T(x)\frac{\partial V^\infty{}_j}{\partial x}\right)$$

Go to 1.

Iterate j until convergence to $u_\infty, V^\infty{}_\infty$ *, with RAS* $\Omega^\infty{}_\infty$

CT Policy Iteration for H-Infinity Control

Results for this Algorithm

Murad Abu Khalaf

The algorithm converges to $V^*(\Omega_0), \Omega_0, u^*(\Omega_0), d^*(\Omega_0)$

the optimal solution on the RAS Ω_0

Sometimes the algorithm converges to the optimal HJI solution V^*, Ω^*, u^*, d^*

For this to occur it is required that $\Omega^* \subseteq \Omega_0$

For every iteration on the disturbance d^i one has

$V^i{}_j \leq V^{i+1}{}_j$ the value function increases

$\Omega^i{}_j \supseteq \Omega^{i+1}{}_j$ the RAS decreases

Converges to available storage

For every iteration on the control u_j one has

$V^\infty{}_j \geq V^\infty{}_{j+1}$ the value function decreases

$\Omega^\infty{}_j \subseteq \Omega^\infty{}_{j+1}$ the RAS does not decrease

Hamilton-Jacobi-Isaacs equation

$$HJI\left(V^*(x,t)\right)=\frac{\partial V^*(x,t)}{\partial t}+\frac{\partial V^{*T}(x,t)}{\partial x}f-\frac{\partial V^{*T}(x,t)}{\partial x}\hat{g}(x)\hat{g}(x)^T\frac{\partial V^*(x,t)}{\partial x}+h^T(x)h(x)=0$$

Boundary condition $V(x(t_f),t_f)=\phi(x(t_f),t_f)$

Here $\hat{g}(x)\hat{g}(x)^T=\frac{1}{4}g(x)g(x)^T-\frac{1}{4\gamma^2}k(x)k(x)^T$

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

NN Approximation of the Cost Function $V(x,t)$

In Sandberg [78], it is shown that NNs with time-varying weights can be used to uniformly approximate continuous time-varying functions.

$$V_L(x,t) = \sum_{j=1}^{L} w_j(t)\sigma_j(x) = w_L^T(t)\sigma_L(x)$$

The NN weights are $w_j(t)$ and L is the number of hidden-layer neurons.

$\boldsymbol{\sigma}_L(x) \equiv [\sigma_1(x)\sigma_2(x)\ldots\sigma_L(x)]^T$ is the vector of activation function.

$\mathbf{w}_L(t) \equiv [w_1(t)w_2(t)\ldots w_L(t)]^T$ is the vector of NN weights.

Note that

$$\frac{\partial V_L(x,t)}{\partial x} = \frac{\partial \boldsymbol{\sigma}_L^T(x)}{\partial x}\mathbf{w}_L(t) \equiv \nabla \boldsymbol{\sigma}_L^T(x)\mathbf{w}_L(t)$$

$$\nabla \boldsymbol{\sigma}_L(x) = \quad \partial \boldsymbol{\sigma}_L(x)/\partial x$$

$$\frac{\partial V_L(x,t)}{\partial t} = \dot{\mathbf{w}}_L^T(t)\boldsymbol{\sigma}_L(x)$$

HJI becomes

$$-\dot{\mathbf{w}}_L^T(t)\boldsymbol{\sigma}_L(x) - \mathbf{w}_L^T(t)\nabla\boldsymbol{\sigma}_L(x)\, f(x)$$
$$+\frac{1}{4}\mathbf{w}_L^T(t)\boldsymbol{\sigma}_L(x)\, g(x) R^{-1} g^T(x)\boldsymbol{\sigma}_L^T(x)\mathbf{w}_L(t) - Q(x) = e_L(x,t)$$

This turns a PDE into an ODE for the NN weights

Lemmas Proved:

As number of NN hidden layer neurons $L \to \infty$

Lemma 1: Convergence of **Approximate HJI Equation**

Lemma 2: Convergence of **NN Weights**

Lemma 3: Convergence of **Approximate Value Function**

Lemma 4: Convergence of **Value Function Gradient**

(Lemmas 3–4:) Conv. in Sobolev space

Lemma 5: Convergence of **Control Inputs**

Lemma 6: Convergence of **State Trajectory**

Stability for enough hidden layer neurons

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

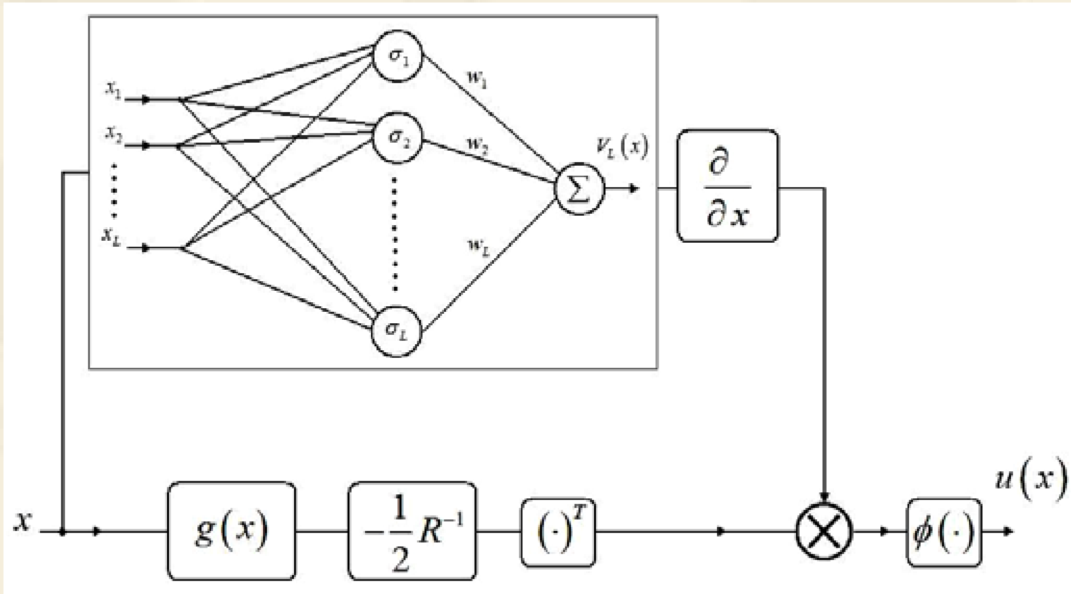
$$HJB\left(V_L(x,t)=\sum_{j=1}^{L} w_j(t)\sigma_j(x)\right)=e_L(x,t)$$

The corresponding optimal control input is

$$u^*(x)=-\frac{1}{2}R^{-1}g(x)^T\frac{\partial V(x,t)^*}{\partial x} \quad = R^{-1}g^T\nabla\boldsymbol{\sigma}_L^T(x)\mathbf{w}_L(t)$$

To find the least-squares solution for $\mathbf{w}_L(t)$, the method of weighted residuals is used

$$\left\langle \frac{\partial e_L(x,t)}{\partial \dot{\mathbf{w}}_L(t)}, e_L(x,t)\right\rangle_\Omega = 0$$



Neural-network-based nearly optimal FEEDBACK control law.

This uses preliminary off-line tuning to solve HJI equation using NN.
Dynamics must be known.

NN Least-Squares Approximate HJI Solution

$$
\begin{aligned}
&\dot{\mathbf{w}}_L(t) = \\
&-\langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x) \rangle_\Omega^{-1} \cdot \langle \nabla \boldsymbol{\sigma}_L(x) f(x), \boldsymbol{\sigma}_L(x) \rangle_\Omega \cdot \mathbf{w}_L(t) \\
&+ \langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x) \rangle_\Omega^{-1} \cdot \langle \mathbf{w}_L^T(t) \nabla \boldsymbol{\sigma}_L(x) \hat{g}(x) \hat{g}^T(x) \nabla \boldsymbol{\sigma}_L^T \mathbf{w}_L(t), \boldsymbol{\sigma}_L(x) \rangle_\Omega \\
&- \langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x) \rangle_\Omega^{-1} \cdot \langle h^T h, \boldsymbol{\sigma}_L(x) \rangle_\Omega \cdot \mathbf{w}_L(t)
\end{aligned}
$$

Boundary condition $V(x(t_f), t_f) = \phi(x(t_f), t_f.)$

NN approx has converted a PDE into an ODE for the NN weights
c.f. Mech Eng assumed mode shapes method for flexible systems

Optimal Algorithm Based on NN Approximation

A mesh of points over the integration region can be introduced on Ω_0

$$A = \left[\boldsymbol{\sigma}_L(x)|_{x_1} \;......\boldsymbol{\sigma}_L(x)|_{x_p}\right]^T$$

$$B = \left[\boldsymbol{\sigma}_L(x) f(x)|_{x_1} \;......\boldsymbol{\sigma}_L(x) f(x)|_{x_p}\right]^T$$

$$C = \left[\left(\nabla\boldsymbol{\sigma}_L(x)\hat{g}(x)\hat{g}^T(x)\nabla\boldsymbol{\sigma}_L^T(x)\right)|_{x_1} \;......\left(\nabla\boldsymbol{\sigma}_L(x)\hat{g}(x)\hat{g}^T(x)\nabla\boldsymbol{\sigma}_L^T(x)\right)|_{x_p}\right]^T$$

$$D = \left[\mathrm{h}^T\mathrm{h}\,|_{x_1} \;......\mathrm{h}^T\mathrm{h}\,|_{x_p}\right]^T$$

where p represents the number of points of the mesh.

Neural Network Solution for Finite-Horizon H-Infinity State Feedback Control

$$\dot{\mathbf{w}}_L(t) = -\left(A^T A\right)^{-1} \mathbf{w}_L(t) A^T B + \left(A^T A\right)^{-1} A^T \mathbf{w}_L^T(t) C \mathbf{w}_L(t) - \left(A^T A\right)^{-1} A^T D$$

This is a nonlinear ODE that can easily be integrated backwards using final condition $\mathbf{w}_L\left(t_f\right)$ to find the least-squares optimal NN weights.

The University of Texas ARLINGTON

ARRI

Numerical Examples

a) Linear System

$$\dot{x}_1 = 2x_1 + 3x_2 + u_1$$
$$\dot{x}_2 = 5x_1 + 6x_2 + 2u_2 \quad (18)$$

To find a nearly optimal time-varying controller, the following smooth function is used to approximate the value function of the system

$$V(x_1, x_2) = w_1 x_1^2 + w_2 x_1 x_2 + w_3 x_2^2 \quad (19)$$

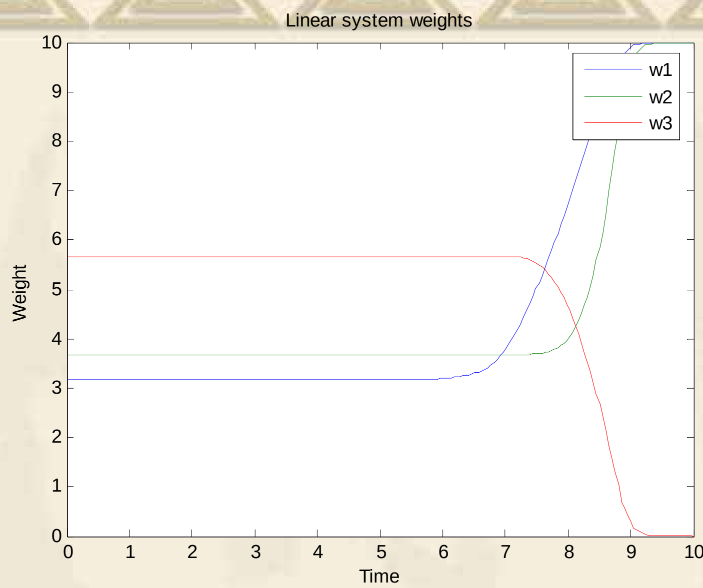


Fig. 1 Linear System Weights

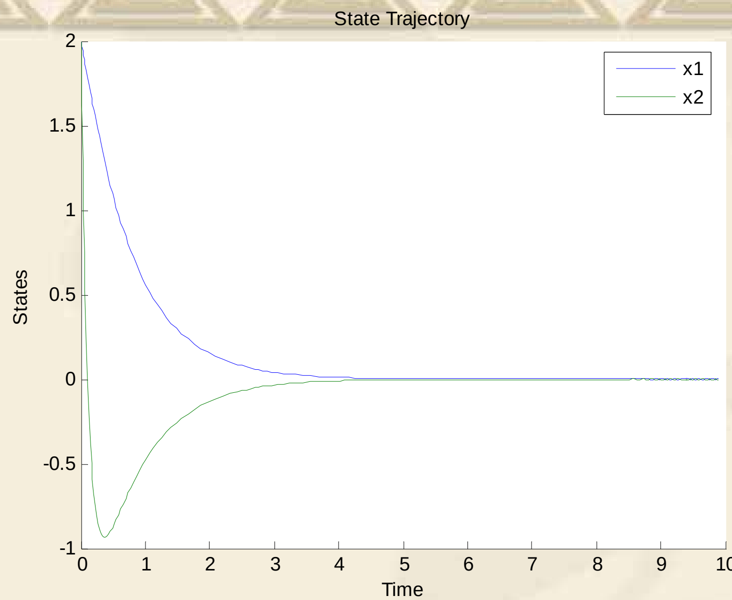


Fig. 2 State Trajectory of Linear System

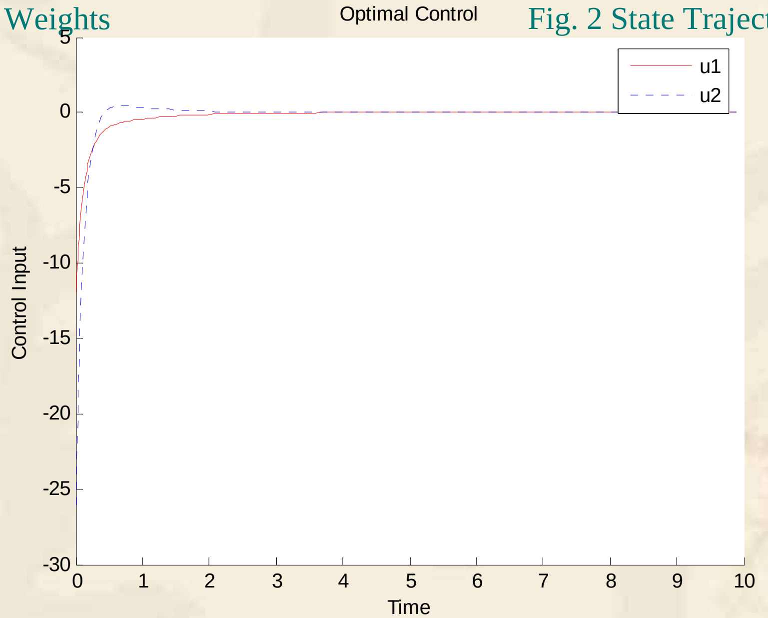


Fig. 3 Optimal NN Control Law

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

b) Nonlinear Chained System

$$\begin{aligned}\dot{x}_1 &= u_1 \\ \dot{x}_2 &= u_2 \\ \dot{x}_3 &= x_1 u_2\end{aligned} \qquad (20)$$

Smooth approximating function

$$\begin{aligned} V(x_1, x_2, x_3) &= w_1 x_1^2 + w_2 x_2^2 + w_3 x_3^2 + w_4 x_1 x_2 + w_5 x_1 x_3 + w_6 x_2 x_3 \\ &+ w_7 x_1^4 + w_8 x_2^4 + w_9 x_3^4 + w_{10} x_1^2 x_2^2 + w_{11} x_1^2 x_3^2 + w_{12} x_2^2 x_3^2 + w_{13} x_1^2 x_2 x_3 \\ &+ w_{14} x_1 x_2^2 x_3 + w_{15} x_1 x_2 x_3^2 + w_{16} x_1^3 x_2 + w_{17} x_1^3 x_3 + w_{18} x_1 x_2^3 + w_{19} x_1 x_3^3 \\ &+ w_{20} x_2 x_3^3 + w_{21} x_2^3 x_3 \end{aligned} \qquad (21)$$

Nonlinear Fixed-Final-Time HJB Solution by NN Least-Squares Approximation

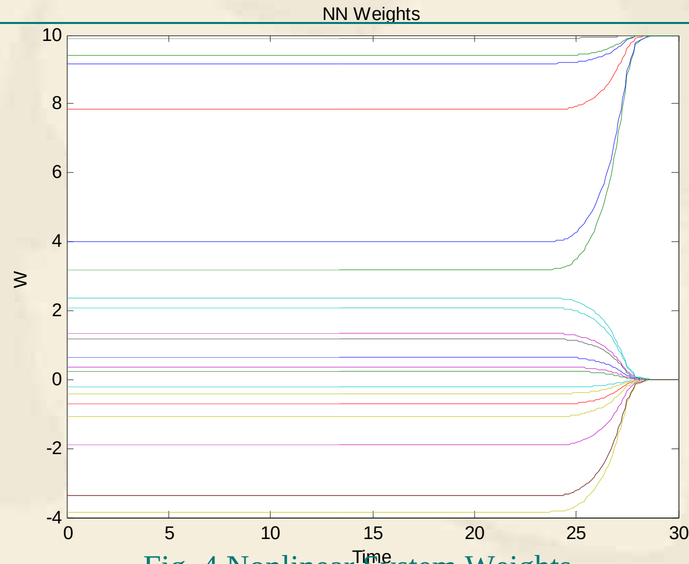


Fig. 4 Nonlinear System Weights

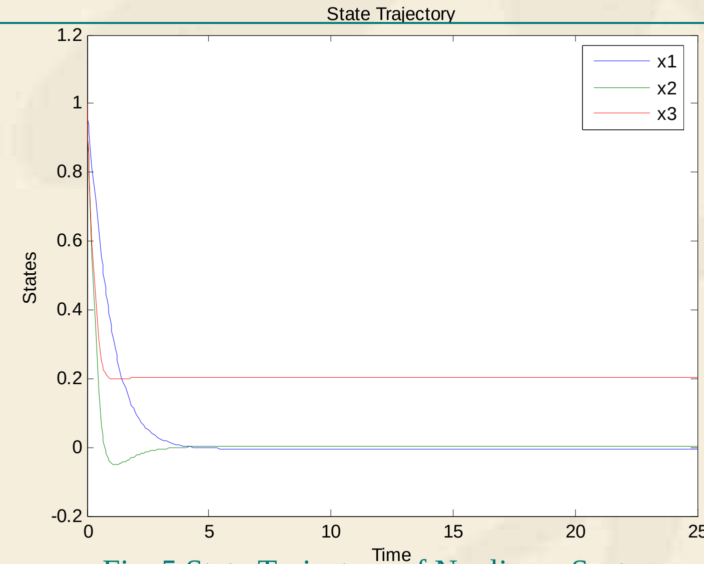


Fig. 5 State Trajectory of Nonlinear System

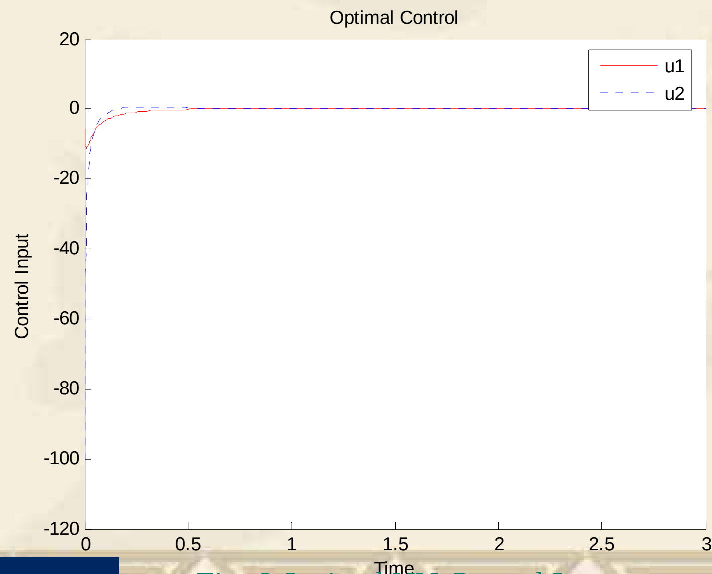


Fig. 6 Optimal NN Control Law

Fix using time-varying transformations– Zhihua Qu

Simulation-Benchmark Problem

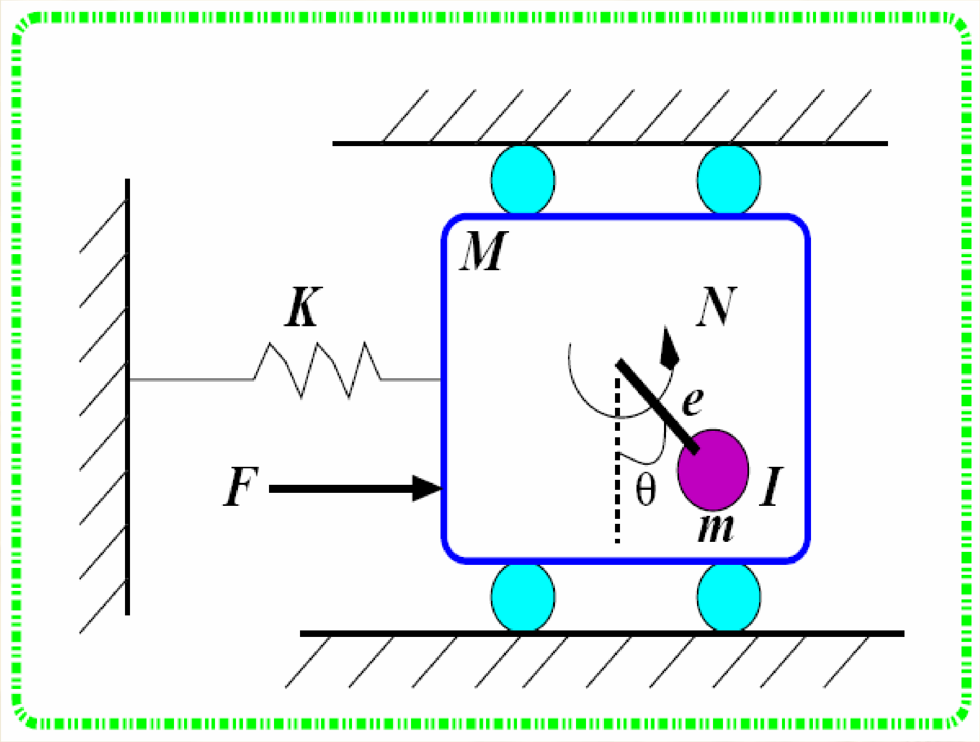


Fig. 8 Rotational actuator to control a translational oscillator.

Neural Network Solution for Finite-Horizon H-Infinity State Feedback Control

$$\dot{x} = f(x) + g(x)u + k(x)d(t) \qquad |u| \le 2$$

$$z^T z = x_1^2 + 0.1x_2^2 + 0.1x_3^2 + 0.1x_4^2 + \|u\|_q^2$$

$$\varepsilon = \frac{me}{\sqrt{(I + me^2)(M + m)}} = 0.2 \qquad \gamma = 10$$

$$f = \begin{bmatrix} x_2 & \frac{-x_1 + \varepsilon x_4^2 \sin x_3}{1 - \varepsilon^2 \cos^2 x_3} & x_4 & \frac{\varepsilon \cos x_3 (x_1 - \varepsilon x_4^2 \sin x_3)}{1 - \varepsilon^2 \cos^2 x_3} \end{bmatrix}^T$$

$$g = \begin{bmatrix} 0 & \frac{-\varepsilon \cos x_3}{1 - \varepsilon^2 \cos^2 x_3} & 0 & \frac{1}{1 - \varepsilon^2 \cos^2 x_3} \end{bmatrix}^T$$

$$k = \begin{bmatrix} 0 & \frac{1}{1 - \varepsilon^2 \cos^2 x_3} & 0 & \frac{-\varepsilon \cos x_3}{1 - \varepsilon^2 \cos^2 x_3} \end{bmatrix}^T$$

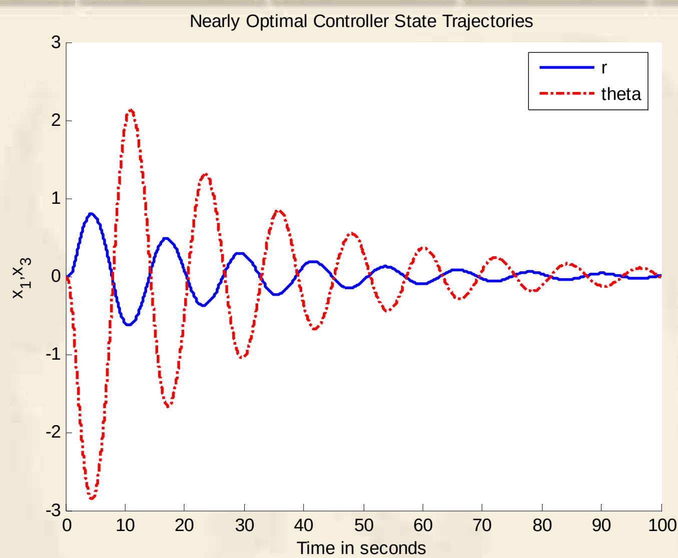


Fig. 9 r , θ State Trajectories

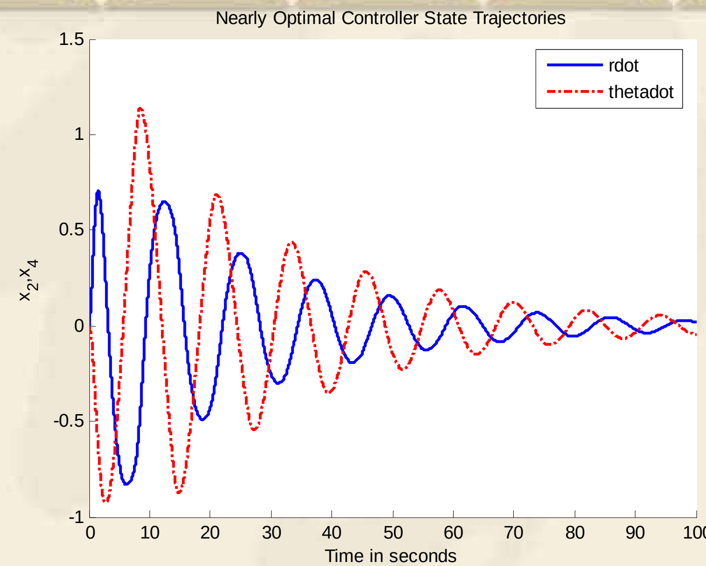


Fig. 10 $\dot{r}$, $\dot{\theta}$ State Trajectories

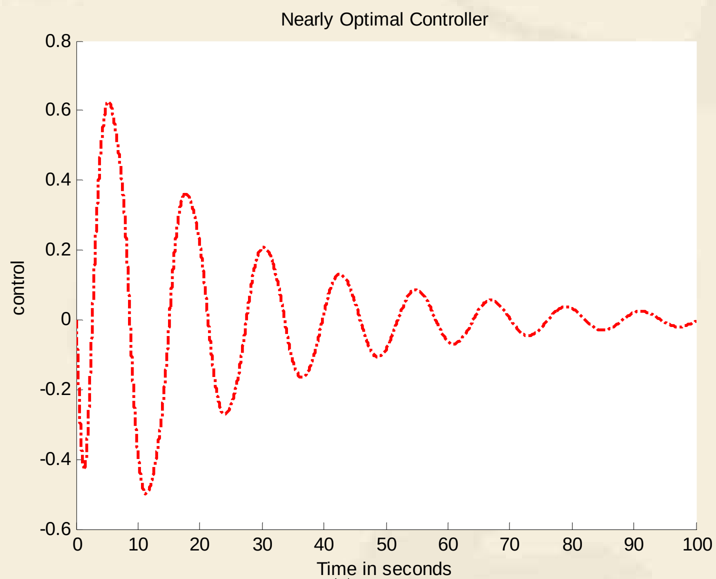


Fig. 11 $u(t)$ Control Input

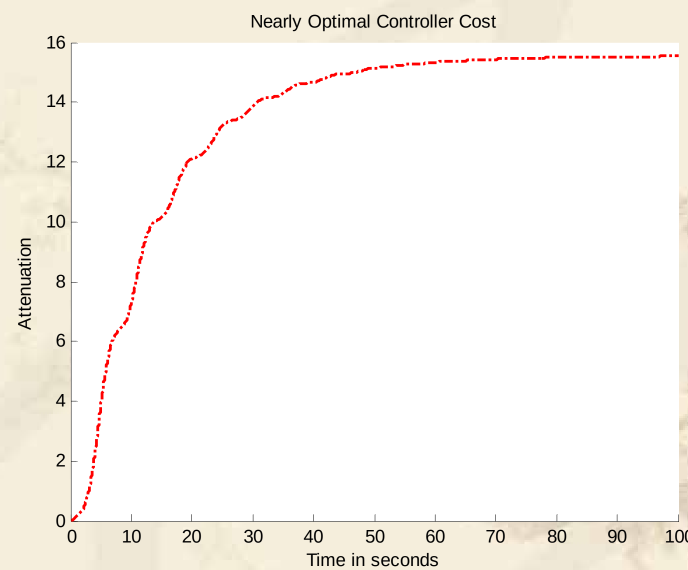


Fig. 12 Disturbance Attenuation

Neural Network Solution for Fixed-Final time Constrained Optimal Control

When the control input is constrained by a saturated function $\boldsymbol{\varphi}(\cdot)$ .To guarantee bounded controls, [1][46] introduced a generalized nonquadratic functional

$$W(u)=2\int_0^u \boldsymbol{\varphi}^{-T}(v)R\,dv$$

$$\boldsymbol{\varphi}(v)=[\phi(v_1)\cdots\phi(v_m)]^T \qquad (22)$$

$$\boldsymbol{\varphi}^{-1}(u)=[\phi^{-1}(u_1)\cdots\phi^{-1}(u_m)]$$

where $v\in\Re^m$, $\boldsymbol{\varphi}\in\Re^m$, and $\boldsymbol{\varphi}(\cdot)$ is a bounded one-to-one function that belongs to $C^p(p\geq 1)$ and $L_2(\Omega)$.

Moreover, it is a monotonic odd function with its first derivative bounded by a constant M .

Neural Network Solution for Fixed-Final time Constrained Optimal Control

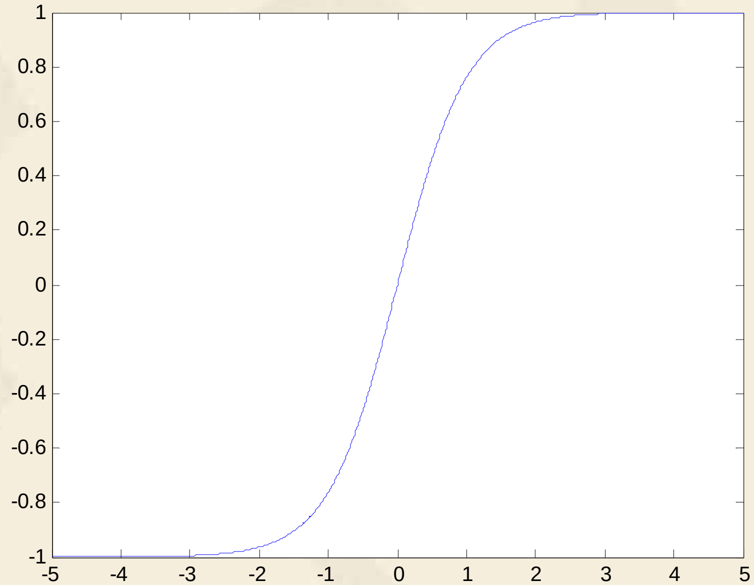


Fig. 13 Hyperbolic tangent function

$$y = \tanh(x)$$

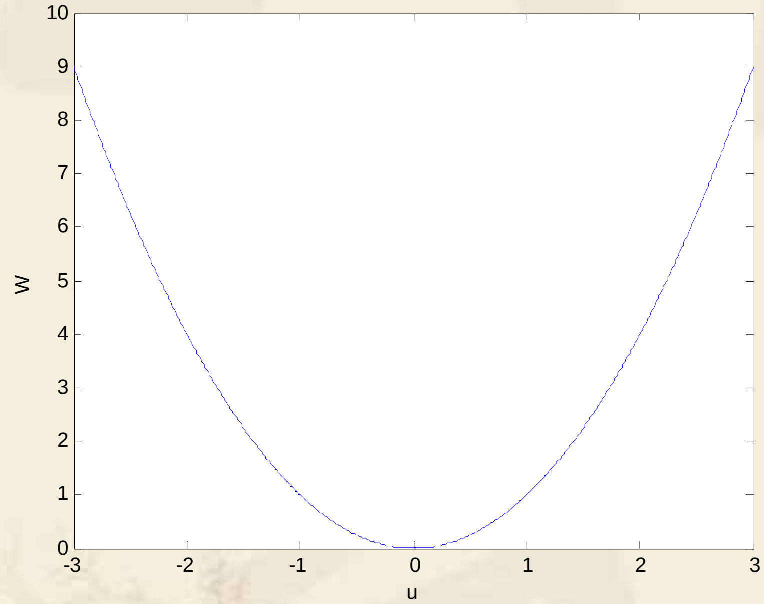


Fig. 14 Nonquadratic Cost

$$W(u) = 2\int_0^u A \cdot \tanh^{-T}(v/A)R\,dv$$

Neural Network Solution for Fixed-Final time Constrained Optimal Control

When (22) is used, (5) becomes

$$-\frac{\partial V(x,t)^*}{\partial t}=\min_{u(t)}\left(Q(x)+2\int_0^u \boldsymbol{\varphi}^{-T}(v)R\,dv+\frac{\partial V(x,t)^{*T}}{\partial x}\left(f(x,t)+g(x)u(x)\right)\right)$$

Minimizing the Hamiltonian of the optimal control problem with regard to u gives

$$g^T(x)\frac{\partial V(x,t)^*}{\partial x}+2\boldsymbol{\varphi}^{-1}(u^*)=0$$

so

$$u(x)^*=-\boldsymbol{\varphi}\left(\frac{1}{2}R^{-1}g(x)^T\frac{\partial V(x,t)^*}{\partial x}\right)\qquad u\in U\subset\Re^m \qquad (23)$$

HJB equation

$$HJB\left(V(x,t)^*\right) = \frac{\partial V(x,t)^*}{\partial t} + \frac{\partial V(x,t)^*}{\partial x}^T f(x)$$
$$+ 2\int_0^u \phi^{-T}(v) R\,dv - \frac{\partial V(x,t)^*}{\partial x}^T \cdot g(x) \cdot \varphi\left(\frac{1}{2} R^{-1} g(x)^T \frac{\partial V(x,t)^*}{\partial x}\right) + Q(x) = 0 \qquad (24)$$

If this HJB equation can be solved for the value function $V(x,t)$, then (24) gives the optimal constrained control.

Neural Network Solution for Fixed-Final time Constrained Optimal Control

So that

$$
\begin{aligned}
\dot{\mathbf{w}}_L(t) &= -\langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x)\rangle_\Omega^{-1} \cdot \langle \nabla\boldsymbol{\sigma}_L(x) f(x), \boldsymbol{\sigma}_L(x)\rangle_\Omega \cdot \mathbf{w}_L(t) \\
&- \langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x)\rangle_\Omega^{-1} \left\langle 2\int_0^u \boldsymbol{\varphi}^{-T}(v) R\,dv, \boldsymbol{\sigma}_L(x)\right\rangle_\Omega \\
&+ \langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x)\rangle_\Omega^{-1} \cdot \left\langle \mathbf{w}_L^T(t) \nabla\boldsymbol{\sigma}_L(x) \cdot g(x) \cdot \boldsymbol{\varphi}\left(\frac{1}{2} R^{-1} g^T(x) \nabla\boldsymbol{\sigma}_L^T(x) \mathbf{w}_L(t)\right), \boldsymbol{\sigma}_L(x)\right\rangle_\Omega \\
&- \langle \boldsymbol{\sigma}_L(x), \boldsymbol{\sigma}_L(x)\rangle_\Omega^{-1} \cdot \langle Q(x), \boldsymbol{\sigma}_L(x)\rangle_\Omega
\end{aligned}
\tag{25}
$$

Optimal Algorithm Based on NN Approximation

(12) can be converted to

$$-A^T A\dot{\mathbf{w}}_L(t) - A^T B\mathbf{w}_L(t) - A^T C + A^T D\mathbf{w}_L(t) - A^T E = 0 \qquad (26)$$

then

$$\begin{aligned}\dot{\mathbf{w}}_L(t) &= -\left(A^T A\right)^{-1} A^T B\mathbf{w}_L(t) - \left(A^T A\right)^{-1} A^T \\ &+ \left(A^T A\right)^{-1} A^T D\mathbf{w}_L(t) - \left(A^T A\right)^{-1} A^T E\end{aligned} \qquad (27)$$

This is a nonlinear ODE that can easily be integrated backwards using final condition $\mathbf{w}_L(t_f)$ to find the least-squares optimal NN weights.

Numerical Examples

a) Linear System

$$\dot{x}_1 = 2x_1 + x_2 + x_3$$
$$\dot{x}_2 = x_1 - x_2 + u_2 \qquad |u_1| \le 5 \qquad |u_2| \le 20 \qquad (28)$$
$$\dot{x}_3 = x_3 + u_1$$

To find a nearly optimal time-varying controller, the following smooth function is used to approximate the value function of the system

$$V(x_1, x_2) = w_1 x_1^2 + w_2 x_2^2 + w_3 x_3^2 + w_4 x_1 x_2 + w_5 x_1 x_3 + w_6 x_2 x_3 \qquad (29)$$

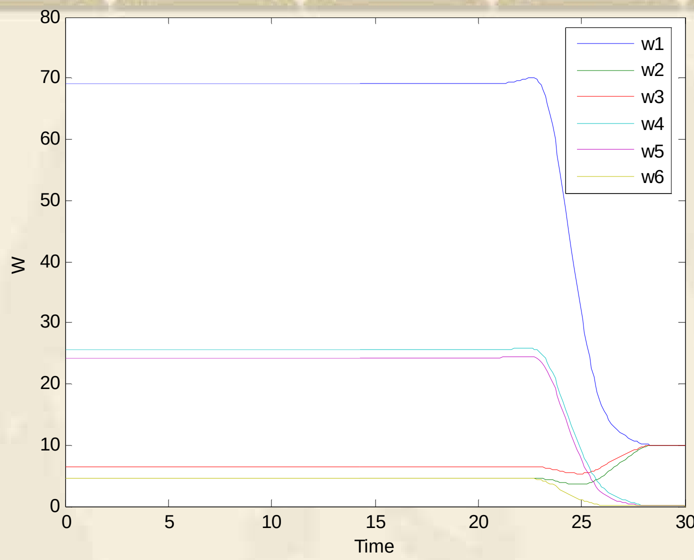


Fig. 15 Constrained Linear System Weights

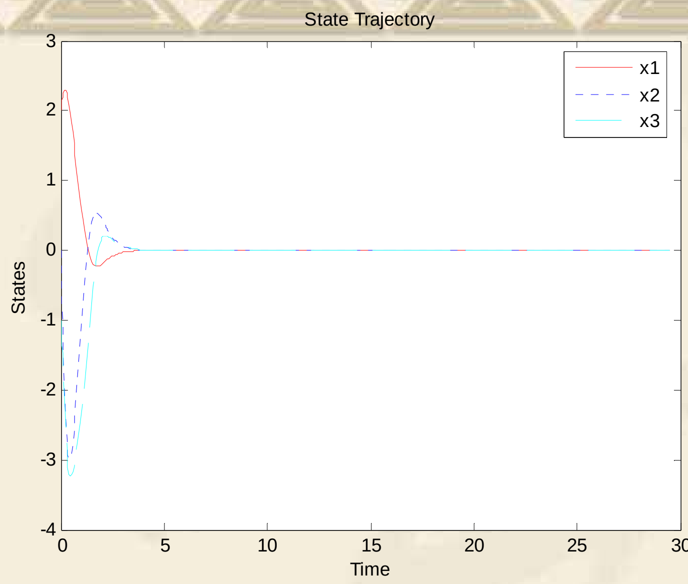


Fig. 16 State Trajectory of Linear System with Bounds

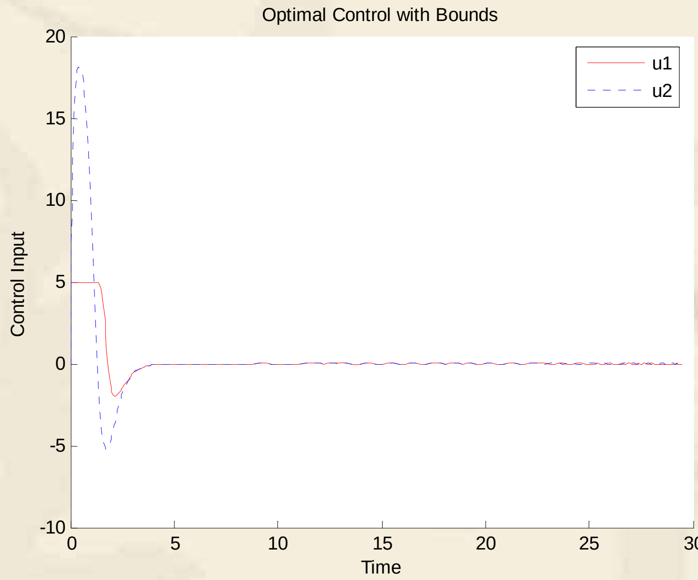


Fig. 17 Optimal NN Control Law with Bounds

Neural Network Solution for Fixed-Final time Constrained Optimal Control

b) Nonlinear Chained System

$$\begin{aligned} \dot{x}_1 &= u_1 \\ \dot{x}_2 &= u_2 \\ \dot{x}_3 &= x_1 u_2 \end{aligned} \qquad |u_1| \le 1 \qquad |u_2| \le 2 \qquad (30)$$

Selecting the smooth approximating function

$$\begin{aligned} V(x_1, x_2, x_3) &= w_1 x_1^2 + w_2 x_2^2 + w_3 x_3^2 + w_4 x_1 x_2 + w_5 x_1 x_3 + w_6 x_2 x_3 \\ &+ w_7 x_1^4 + w_8 x_2^4 + w_9 x_3^4 + w_{10} x_1^2 x_2^2 + w_{11} x_1^2 x_3^2 + w_{12} x_2^2 x_3^2 + w_{13} x_1^2 x_2 x_3 \\ &+ w_{14} x_1 x_2^2 x_3 + w_{15} x_1 x_2 x_3^2 + w_{16} x_1^3 x_2 + w_{17} x_1^3 x_3 + w_{18} x_1 x_2^3 + w_{19} x_1 x_3^3 \\ &+ w_{20} x_2 x_3^3 + w_{21} x_2^3 x_3 \end{aligned} \qquad (31)$$

Neural Network Solution for Fixed-Final time Constrained Optimal Control

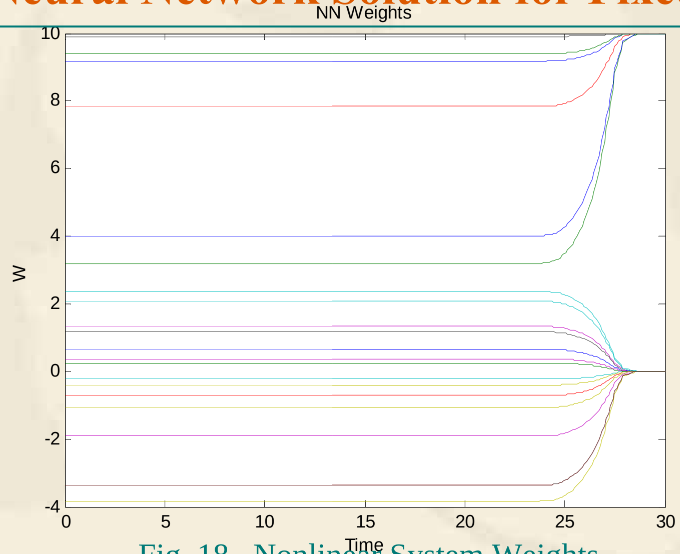


Fig. 18 Nonlinear System Weights

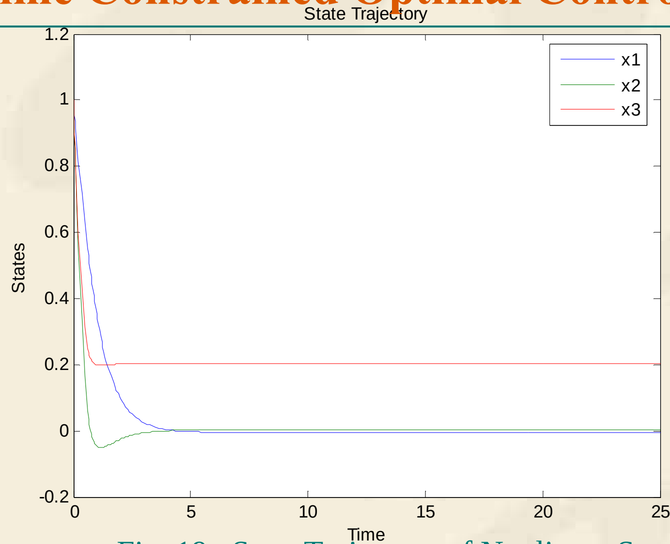


Fig. 19 State Trajectory of Nonlinear System

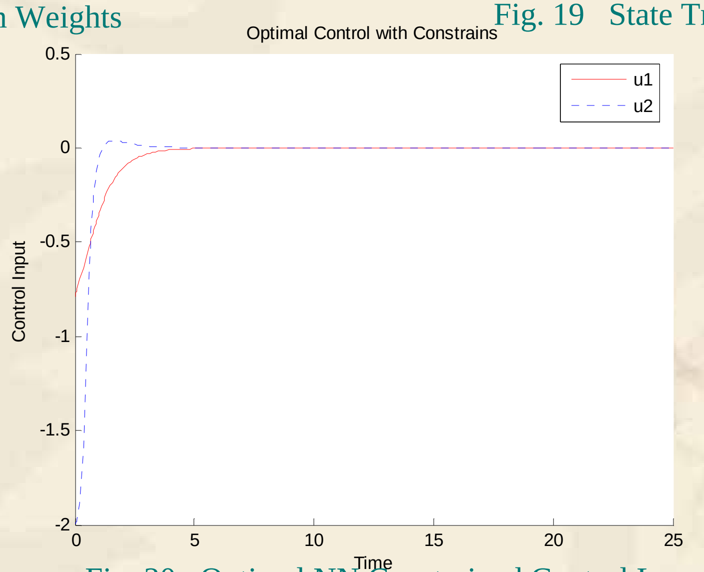


Fig. 20 Optimal NN Constrained Control Law

The University of Texas ARLINGTON

ARRI

C) Simulation-Benchmark Problem

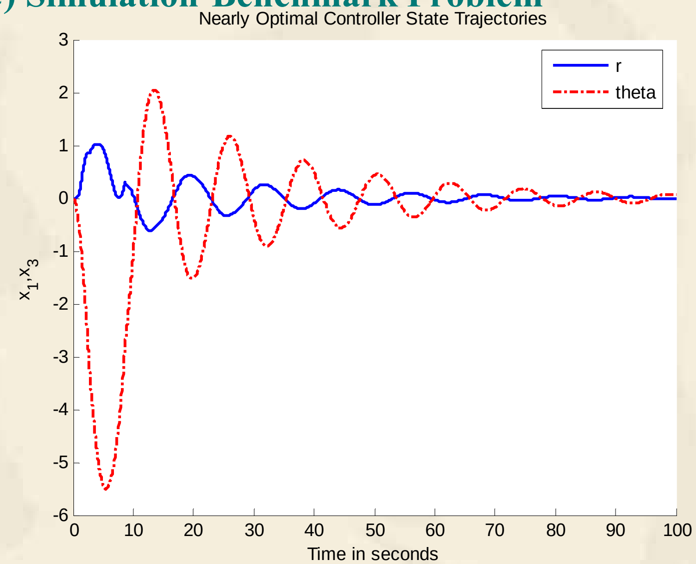


Fig. 21 r θ State Trajectories

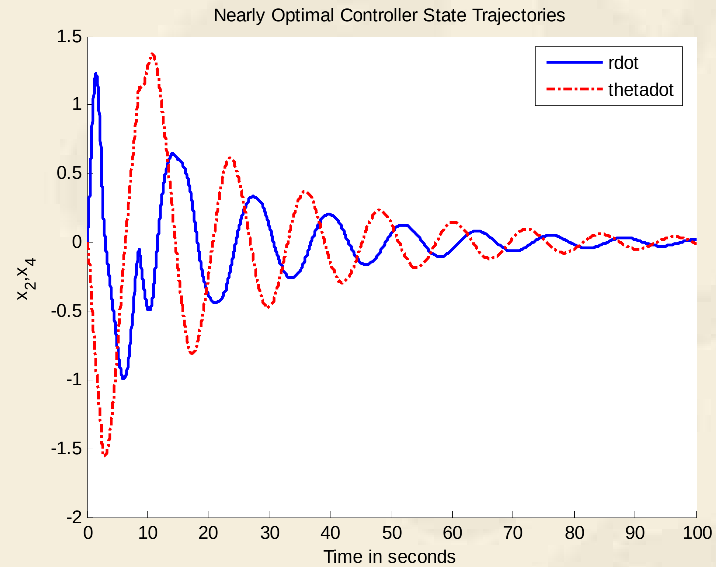


Fig. 22 $\dot{r}$ $\dot{\theta}$ State Trajectories

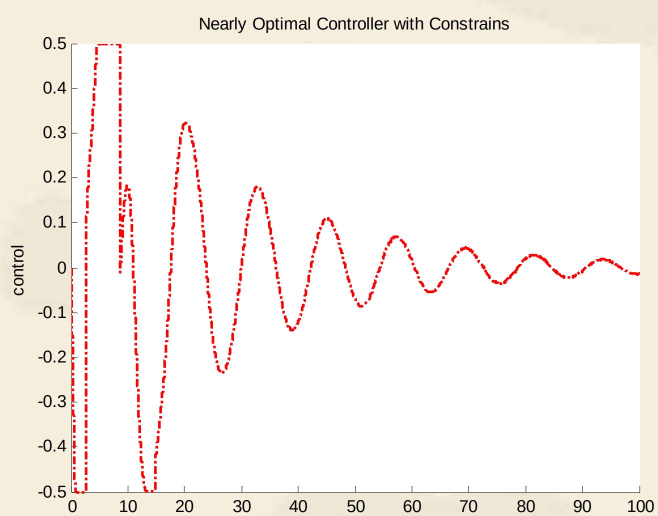


Fig. 23 $u(t)$ Control Input

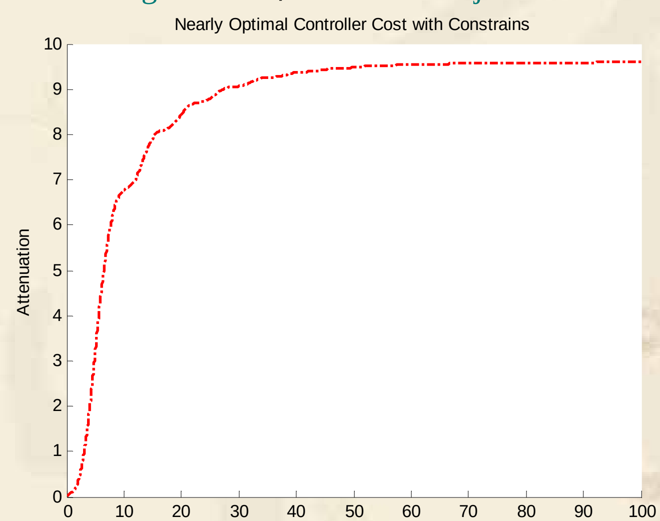


Fig. 24 Disturbance Attenuation

Overview of the Method

- Neural networks are used to approximately solve the finite-horizon optimal state feedback control problem
- The method is based on solving a related Hamilton-Jacobi equation of the corresponding finite-horizon problem
- Transform the problem into solving an ODE equation backwards in time.
- Neural network approximation converges uniformly to the function and the resulting controller provides closed-loop stability.
- The result is a nearly exact feedback controller with time-varying coefficients.
- No policy iteration needed.