Reinforcement Learning with Sparse Bellman Error Extrapolation for Infinite-Horizon Approximate Optimal Tracking

Max Greene[1], Patryk Deptula[2], Scott Nivison[3], Warren Dixon[1]

[1]Dept. of Mechanical and Aerospace Engineering, Univ. of Florida
[2]The Charles Stark Draper Laboratory, Inc.
[3]Munitions Directorate, Air Force Research Laboratory

Under Review TAC, Fall 2020

Problem Formulation

Dynamical System

Given a control affine nonlinear dynamical system:

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

Control Objective (Regulation Case)

Design a controller, $u(t)$, which minimizes a cost function:

$$J(x,u) = \int_0^\infty \big(x(\tau)^T Q x(\tau) + u(\tau)^T R u(\tau)\big) d\tau$$

Cost-to-Go

Optimal value function:

$$V^*(x) = \min_{\substack{u(\tau)\epsilon U \\ \tau\epsilon\mathbb{R}_{\geq t}}} \int_t^\infty \big(x(\tau)^T Q x(\tau) + u(\tau)^T R u(\tau)\big) d\tau$$

Hamilton Jacobi Bellman Equation

Hamilton Jacobi Bellman (HJB) equation:

$$0 = \nabla_x V^*(x)\big(f(x) + g(x)u^*(x)\big) + x^T Q x + u^*(x)^T R u^*(x)$$

Optimal Controller

From Solving the HJB equation:

$$u^*(x) = -\frac{1}{2} R^{-1} g(x)^T \big(\nabla_x V^*(x)\big)^T$$

- Cannot solve HJB analytically
- Approximate the Value Function (V^*)
  - Stone Weierstrass Theorem
  - Neural Networks

Optimal Value Function and Optimal Control Policy:

$$V^*(x) = W^T\sigma(x) + \varepsilon(x) \qquad u^*(x) = -\frac{1}{2}R^{-1}g(x)^T(\nabla_x\sigma(x)^T W + \nabla_x\varepsilon(x)^T)$$

Unknown: Neural weights $\quad W \rightarrow \hat{W}_c,\ \hat{W}_a$

$\hat{W}_a$: Actor weight

$\hat{W}_c$: Critic weight

Value Function and Optimal Control Policy Approximation

$$\hat{V}\left(x,\hat{W}_c\right) = \hat{W}_c^{\,T}\sigma(x) \qquad \hat{u}\left(x,\hat{W}_a\right) = -\frac{1}{2}R^{-1}g(x)^T\left(\nabla_x\sigma(x)^T\hat{W}_a\right)$$

Bellman Error (BE): Residual from HJB

$$\hat{\delta}\left(x,\hat{W}_c,\hat{W}_a\right) \triangleq \nabla_x\hat{V}\left(x,\hat{W}_c\right)\left(f(x) + g(x)\hat{u}\left(x,\hat{W}_a\right)\right) + \hat{u}\left(x,\hat{W}_a\right)^T R\hat{u}\left(x,\hat{W}_a\right) + x^T Q x$$

Instantaneous BE: Residual from Optimality

$$\hat{\delta}_i(e,t) \triangleq \hat{\delta}\left(e_i, \widehat{W}_c(t), \widehat{W}_a(t)\right)$$

Weight Update Laws using R-MBRL

$$\dot{\widehat{W}}_c(t) = -\eta_{c1}\Gamma \frac{\omega(t)}{\rho(t)}\hat{\delta} + \eta_{c2}\frac{1}{N_j}\sum_{i=1}^{N_j}\frac{\omega_i(t)}{\rho_i(t)}\hat{\delta}_i(t)$$

On-Trajectory Point

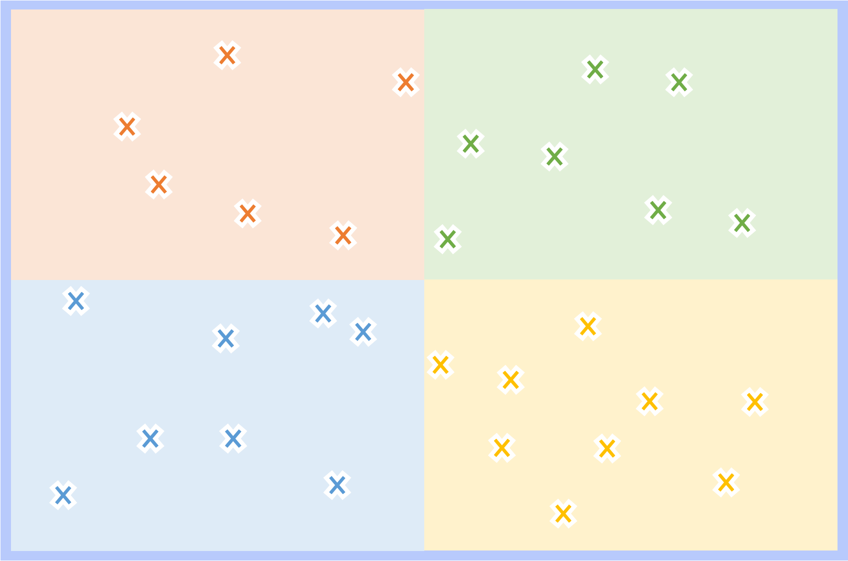
Off-Trajectory Points Sparse Terms

$$\dot{\Gamma}(t) = \left(\lambda\Gamma(t) - \frac{\eta_{c1}\Gamma(t)\omega(t)\omega(t)^T\Gamma(t)}{\rho(t)} - \Gamma(t)\eta_{c2}\left(\frac{1}{N_j}\sum_{i=1}^{N_j}\frac{\omega_i(t)\omega_i^T(t)}{\rho_i(t)}\hat{\delta}_i(t)\right)\Gamma(t)\right)\mathbf{1}_{\{\underline{\Gamma}\le\|\Gamma\|\le\overline{\Gamma}\}}$$

$$\dot{\widehat{W}}_a(t) = -\eta_{c1}\left(\widehat{W}_a(t) - \widehat{W}_c(t)\right) - \eta_{a2}\widehat{W}_a(t) + \frac{\eta_{c1}G_\sigma^T(t)\widehat{W}_a(t)\omega(t)^T}{4\rho(t)}\widehat{W}_c(t) + \left(\frac{\eta_{c2}}{4N_j}\sum_{i=1}^{N_j}\frac{G_{i\sigma}^T\widehat{W}_a(t)\omega_i(t)}{\rho_i(t)}\hat{\delta}_i(t)\right)\widehat{W}_c(t)$$

Segmentation

- Separate operating domain
- Bellman error extrapolation contained to segment
- Smaller history stack
- Switches depending on region
- Introduces discontinuities



Segmentation

Simulation Results

- Linear Quadratic Tracking (LQT)

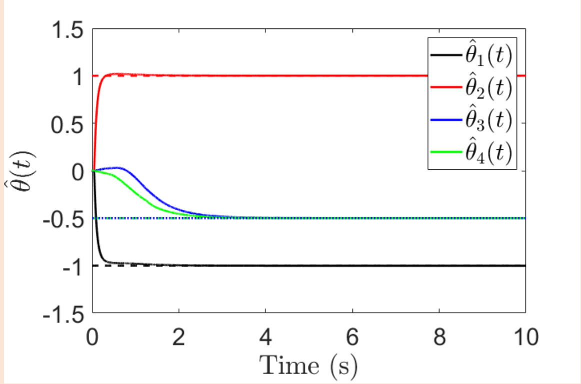
$$\dot{x} = \begin{bmatrix} -x_1 + x_2 \\ -\frac{1}{2}x_1 - \frac{1}{2}x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

$$x_d = \begin{bmatrix} 4\sin(t) \\ 4\cos(t) + 4\sin(t) \end{bmatrix}$$

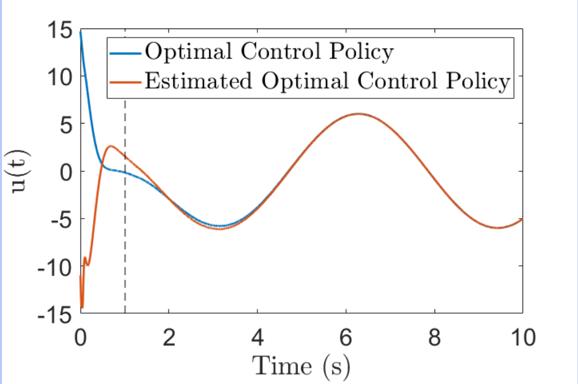
  - Analytical solution known
  - Non-sparse basis outside of box
  - $\sigma(\zeta) = [e_1^2, e_1 e_2, e_1 x_{d1}, e_1 x_{d2}, e_2^2, e_2 x_{d1}, e_2 x_{d2}]^T$
  - Sparse basis inside of box
  - $\sigma(\zeta) = [e_1^2, e_1 e_2, 0, 0, e_2^2, e_2 x_{d1}, e_2 x_{d2}]^T$
  - Dynamics approximated with neural network

Simulation Results

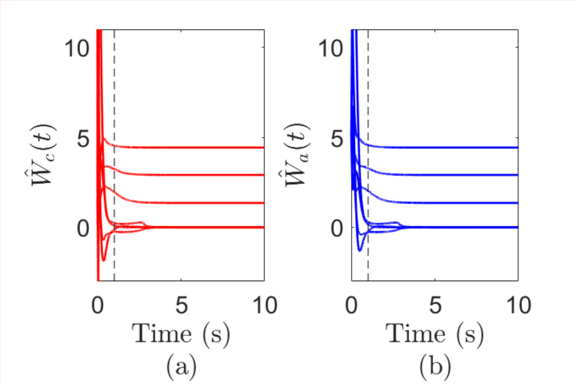
NN System ID Weights



Control Policy



Critic/Actor Weights



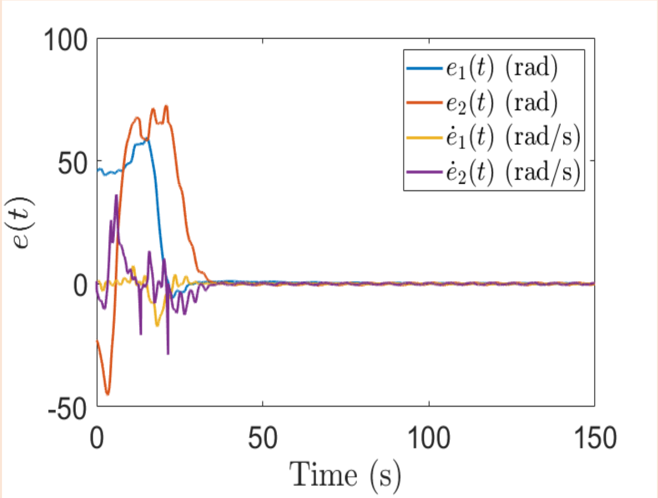
Simulation Results

- Standard Model-Based ADP
- SS Model-Based ADP

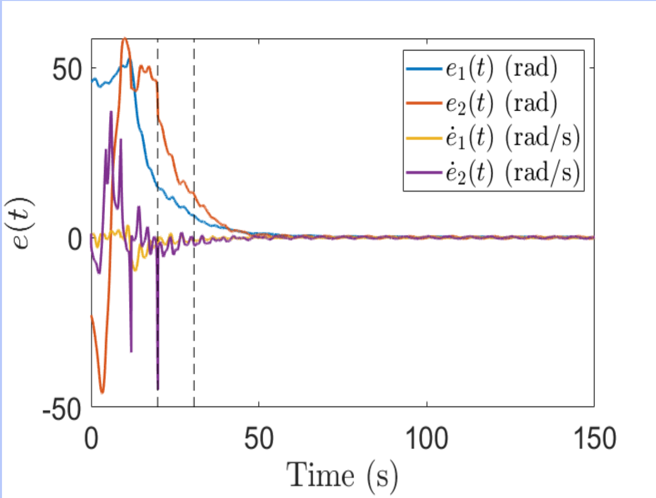
Median Computation Time (10 trials) (s)	120.40	25.90
Integral of Error ($\int_0^{150} \|e(\tau)\| \, d\tau$) (rad·s)	33.72	27.97
5% Rise Time (s)	33.33	44.29
RMS Steady State Error (s)	$6.92 \cdot 10^{-3}$	$5.57 \cdot 10^{-3}$

Simulation Results

Standard Model-Based ADP



SS Model-Based ADP



Model-based Reinforcement Learning for Optimal Feedback Control of Switched Systems

MAX Greene[1], MOAD ABUDIA[2], RUSHIKESH KAMALAPURKAR[2], WARREN DIXON[1]

[1]Dept. of Mech. and Aerospace Engineering, Univ. of Florida
[2]Dept. of Mech. and Aerospace Engineering, Oklahoma State Univ.

To Appear, Conf. on Decision and Control (CDC), December 2020

- Theorem 1: Subsystem Stability Analysis
  - $V_{L,i}(r_i,t) = V_i^*(x) + \frac{1}{2}\widetilde{W}_{c,i}^{\ T}\Gamma_i^{-1}\widetilde{W}_{c,i} + \frac{1}{2}\widetilde{W}_{a,i}^{\ T}\widetilde{W}_{a,i}$
  - $\dot{V}_{L,i}(r_i,t) \leq \frac{\Lambda_i}{\alpha_{2,i}} V_{L,i}\ (r_i,t) + l_i$
  - System state (x), weight estimation errors $(\widetilde{W}_c, \widetilde{W}_a)$, and control policy $u(t)$ is Uniformly Ultimately Bounded
  - Exponential convergence to a region $V_{L,i}(r_i,t) \leq \frac{2 l_i \alpha_{2,i}^3}{\Lambda_i \alpha_{1,i}^2}$.

Theorem 2

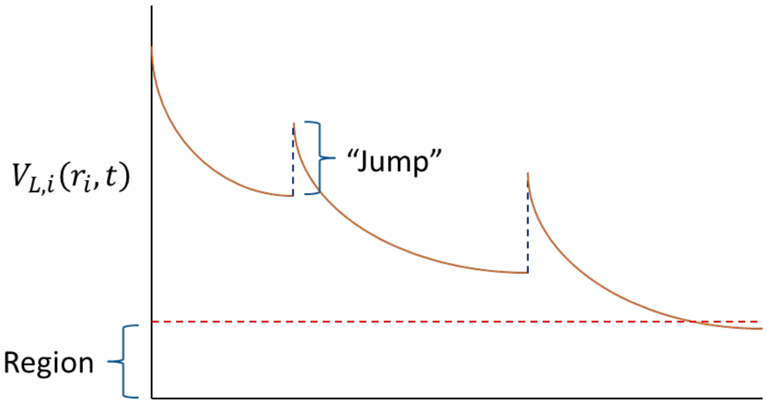
- When switching from $i = 1 \rightarrow 2$, there is a jump between the multiple Lyapunov functions.

$$V_{L,1}(r_1, t) = V_1^*(x) + \frac{1}{2}\tilde{W}_{c,1}{}^T \Gamma_1^{-1} \tilde{W}_{c,1} + \frac{1}{2}\tilde{W}_{a,1}{}^T \tilde{W}_{a,1}$$

$$V_{L,2}(r_2, t) = V_2^*(x) + \frac{1}{2}\tilde{W}_{c,2}{}^T \Gamma_2^{-1} \tilde{W}_{c,2} + \frac{1}{2}\tilde{W}_{a,2}{}^T \tilde{W}_{a,2}$$

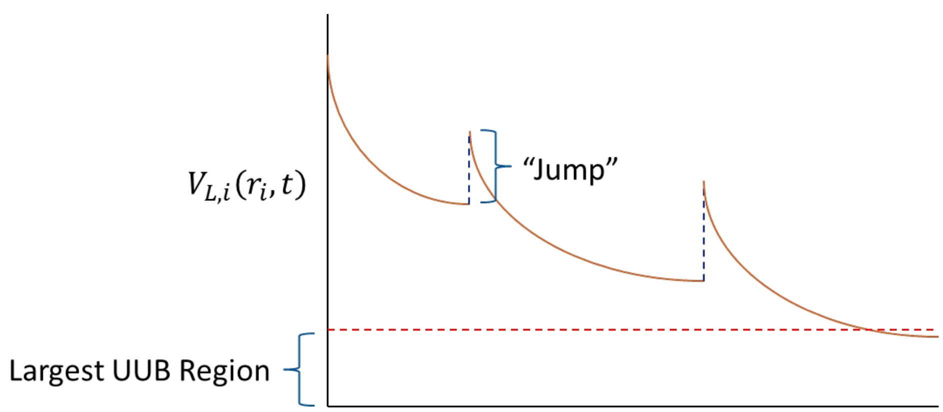
Scales by const. due to quadratic value fcn. assumption

Switching causes discrete jumps in these values



Theorem 2:

The system consisting of a family of subsystems, each with control affine dynamics and a properly designed dwell-time, τ, ensures that x, $\tilde{W}_{c,i}$ and $\tilde{W}_{a,i}$ $\forall i$ will converge to a neighborhood of the origin in the sense that $V_{L,i}(r_i, t) \leq V_{L,B}$ for all $t \geq T$; where $V_{L,B} \in \mathbb{R}$ is the maximum ultimate bound for all subsystems, and $T \in \mathbb{R}$ is the time required to reach the ultimate bound $V_{L,B}$; provided a minimum dwell-time τ^* is satisfied.



Switched System ADP

- F-16 longitudinal dynamics
  - [Stevens, Lewis, Johnson, 2016]

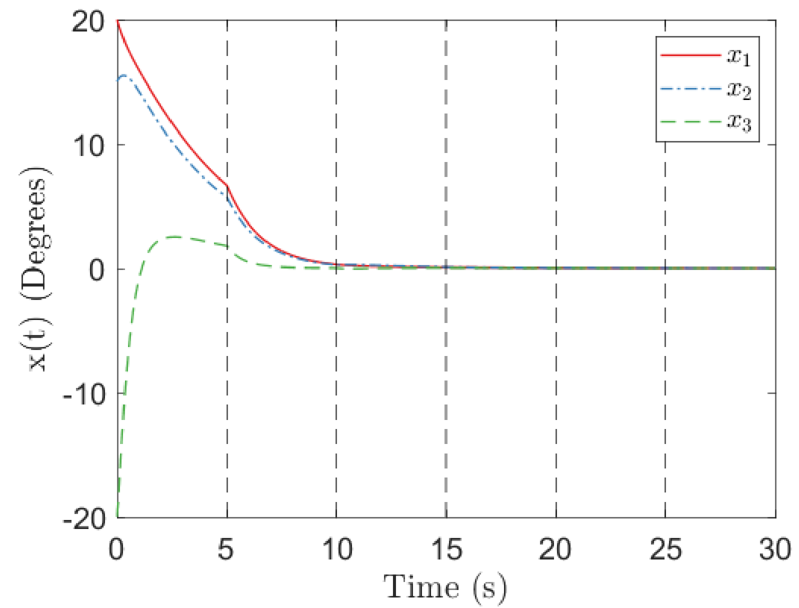
Explore further connection with Ben Dickenson (AFRL/RW), regarding reconfigurable aircraft munition that extend wings, retract wings

	Dynamic Model
Mode 1, Unaltered Model	$\dot{x} = \begin{bmatrix} -1 & 0.9 & -0.002 \\ 0.8 & -1.1 & -0.2 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$
Mode 2, Altered Model	$\dot{x} = \begin{bmatrix} -0.8 & 0.2 & -0.01 \\ 0.6 & -1.3 & -0.1 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$
Mode 3, Altered Model	$\dot{x} = \begin{bmatrix} -1 & 0.5 & -0.02 \\ 0.9 & -0.8 & -0.4 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$

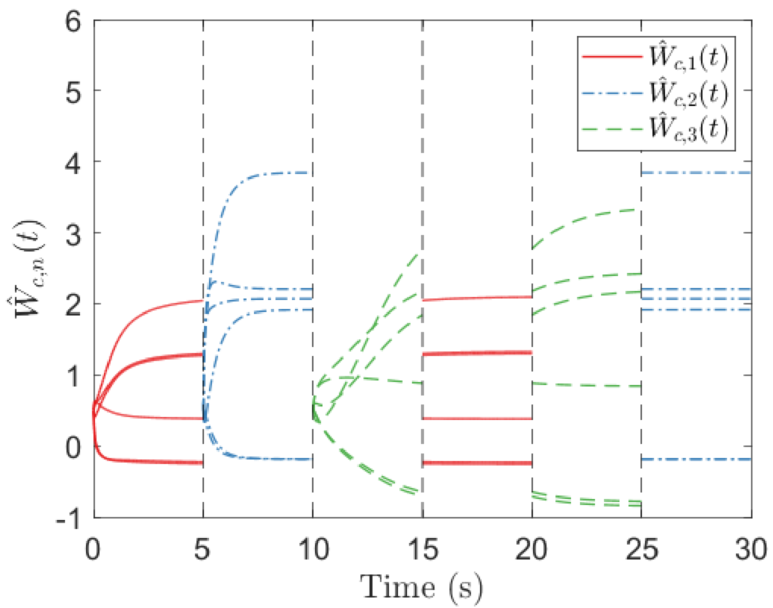
Switched System ADP

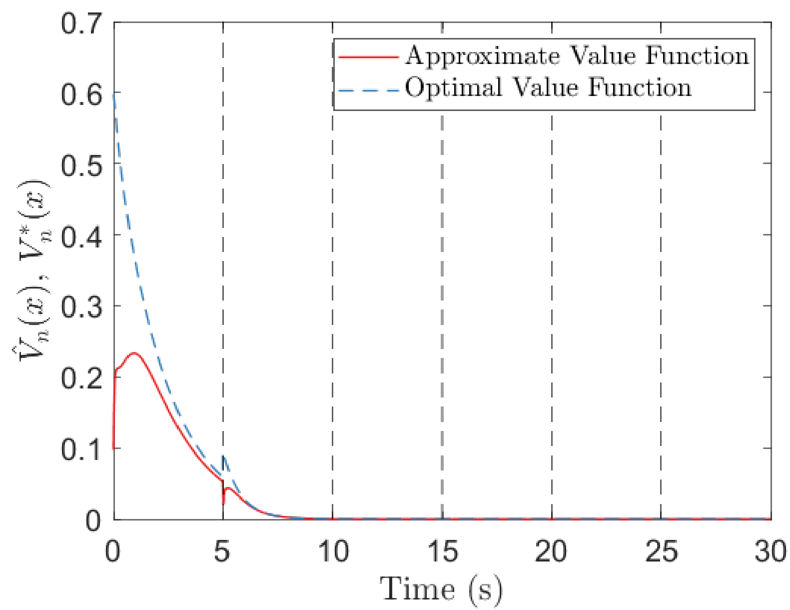
- Switch between multiple dynamical systems
  - Arbitrary switching sequence
  - Satisfies minimum dwell-time condition

- Switching Sequence
  - {1,2,3,1,3,2}



Switched System ADP





Lyapunov-Based Real-Time and Iterative Adjustment of Deep Neural Networks

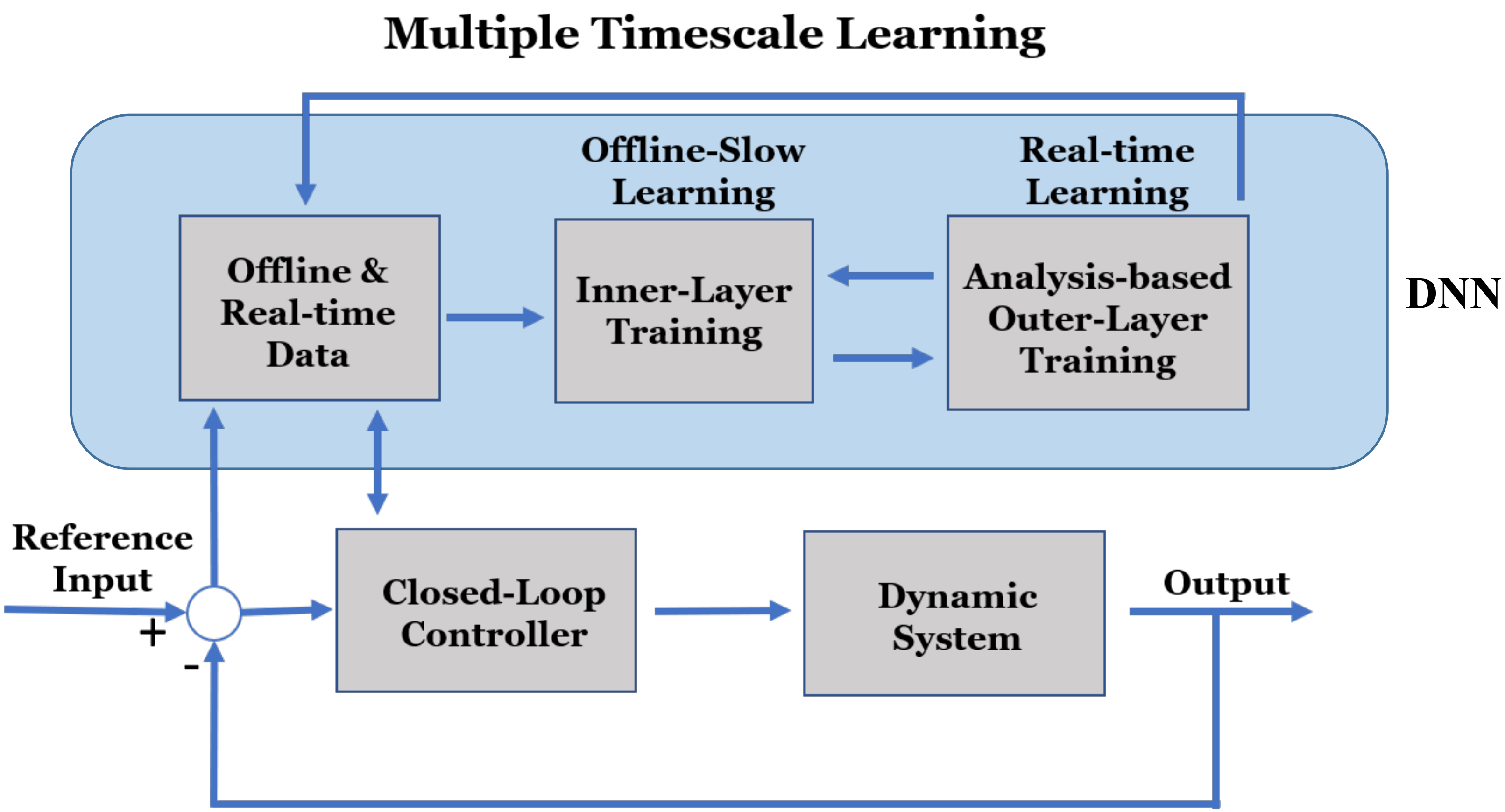
R. Sun, M. L. Greene, D. M. Le, Z. I. Bell, G. Chowdhary, W. E. Dixon

[1]Univ. of Florida, [1]AFRL/RW, [2]Univ. of Illinois Urbana-Champagne

Under Review, IEEE Control Systems Letters

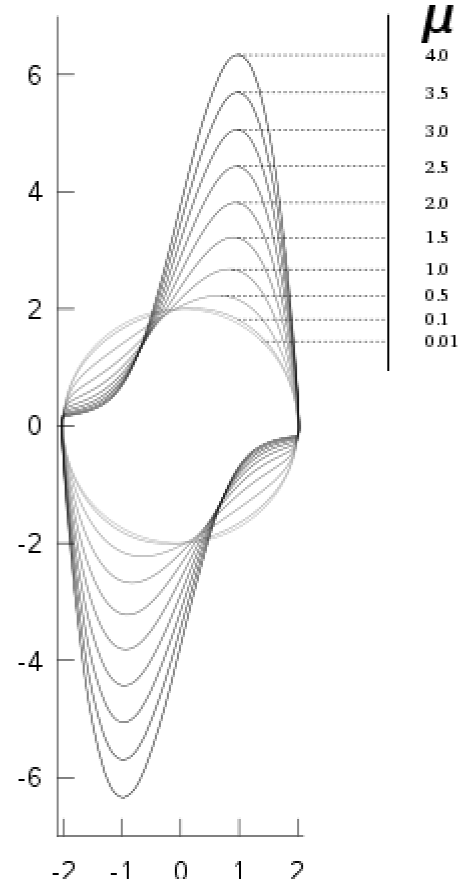
UF UNIVERSITY of FLORIDA Duke UNIVERSITY TEXAS The University of Texas at Austin UC SANTA CRUZ

DNN-Based Adaptive Control



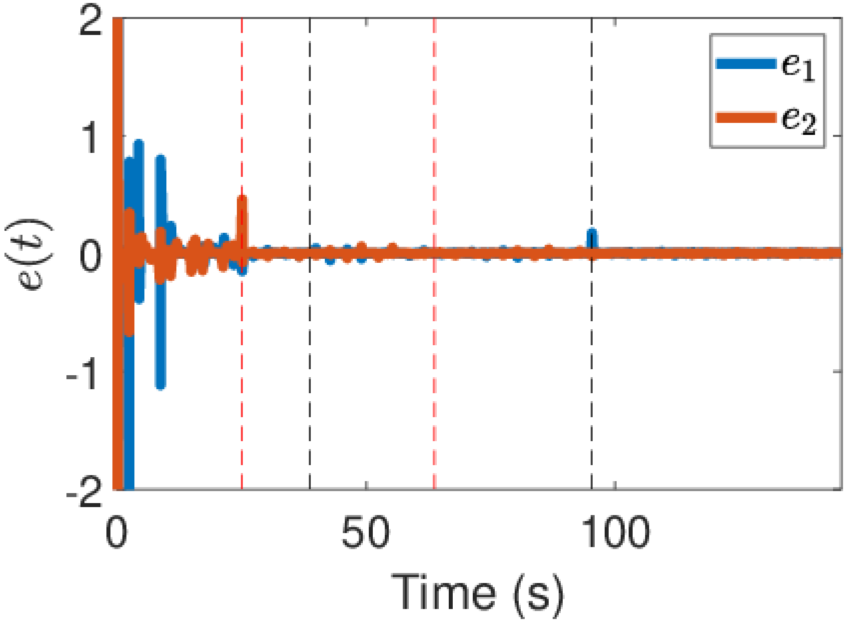
DNN-Based Adaptive Control

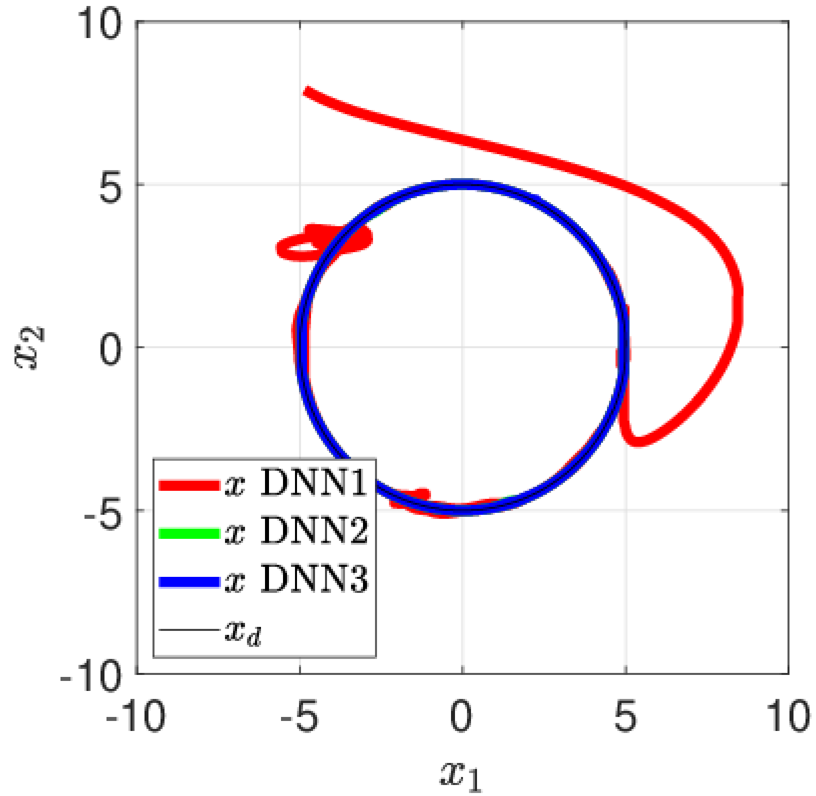
- Van der Pol Oscillator
- Trained with 600s of simulation data
- Transient response is fast relative to the overall timescale



DNN-Based Adaptive Control

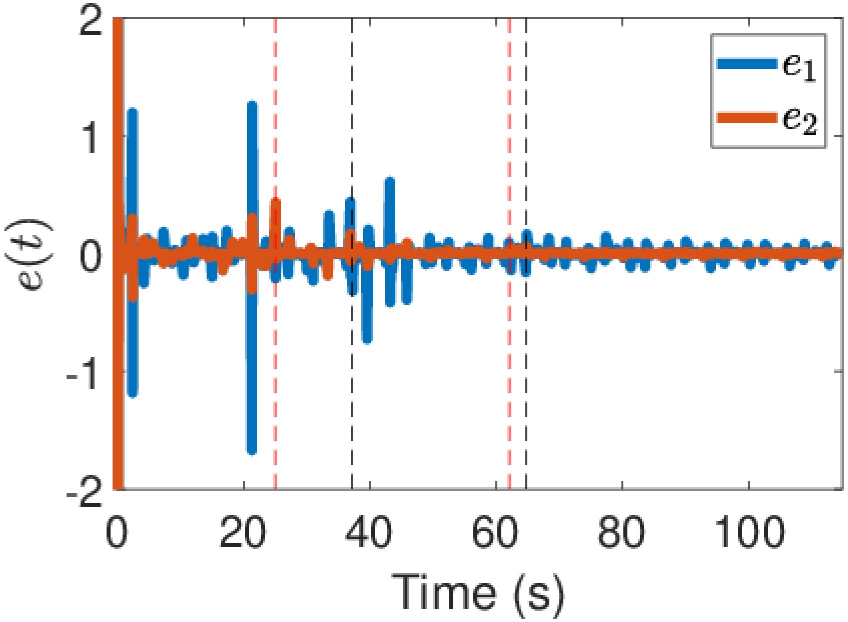
Trained on identical dynamics

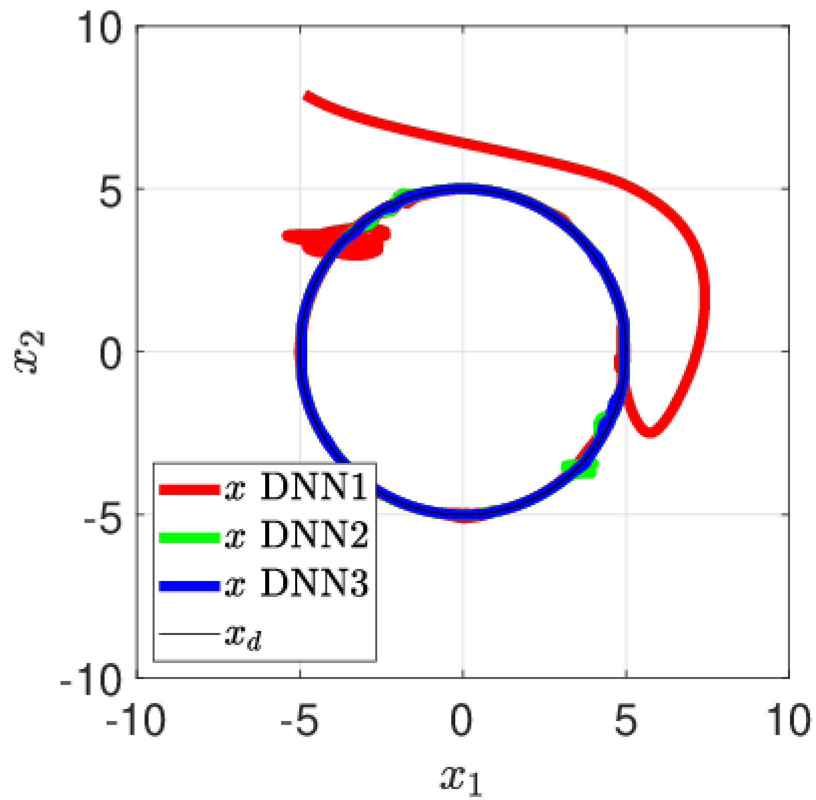




DNN-Based Adaptive Control

Trained on similar dynamics (different coefficients) - transfer learning





DNN-Based Adaptive Control

No offline training. Inner-layer DNN weights are randomly initialized.