# Adaptation, Optimality and Constrained Systems

# Deep Fully-Connected and Residual Neural (ResNet) Network-Based Adaptive Control: A Lyapunov-Based Approach

# Fully-Connected DNN Architecture

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

Control objective : $e \triangleq x - x_d(t) \to 0$ as $t \to \infty$ where $x_d(t) \in \Omega$, a known compact set

- Fully-Connected DNN with some input $\eta$

$$\Phi(\eta, V_0, V_1, \ldots, V_k) \triangleq \left(V_k^T \phi_k \circ \ldots \circ V_1^T \phi_1\right)\left(V_0^T \eta\right)$$

- Recursive Representation

$$\Phi = V_k^T \varphi_k \qquad \varphi_j \triangleq \begin{cases} \phi_j\left(V_{j-1}^T \varphi_{j-1}\right), & j \in \{1, \ldots, k\}, \\ \eta, & j = 0. \end{cases}$$

- Universal Approximation Property

$$f(x_d) = \Phi(x_d, V_0^*, V_1^*, \ldots, V_k^*) + \varepsilon(x_d) \qquad \sup_{x_d \in \Omega} \|\varepsilon(x_d)\| \leq \overline{\varepsilon}$$

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

# Control Design

- Adaptive Feedforward DNN Term

$$\hat{\Phi} \triangleq \Phi(x_d, \hat{V}_0, \ldots, \hat{V}_k)$$

- Control Law

$$u \triangleq \dot{x}_d - \rho(\|e\|)e - k_1 e - k_s \mathrm{sgn}\,(e) - \overset{\text{DNN}}{\boxed{\hat{\Phi}}}$$

where $\|f(x) - f(x_d)\| \leq \rho\,(\|e\|)\,\|e\|$

- Let

$$\hat{\theta} \triangleq \left[\mathrm{vec}(\hat{V}_0)^T, \ldots, \mathrm{vec}(\hat{V}_k)^T\right]^T$$

$$\hat{\varphi}_j \triangleq \varphi_j(x_d, \hat{V}_0, \ldots, \hat{V}_j)$$

$$\hat{\varphi}'_j \triangleq \varphi'_j(x_d, \hat{V}_0, \ldots, \hat{V}_j)$$

# Adaptation Law

- Adaptation Law (analysis allows for ReLU activation functions)

$$\dot{\hat{\theta}} \triangleq \mathrm{proj}\left(\Gamma \Phi'^{T} e\right)$$

where $\Phi' \triangleq \frac{\partial \hat{\Phi}}{\partial \hat{\theta}}$ is computed using the chain rule as

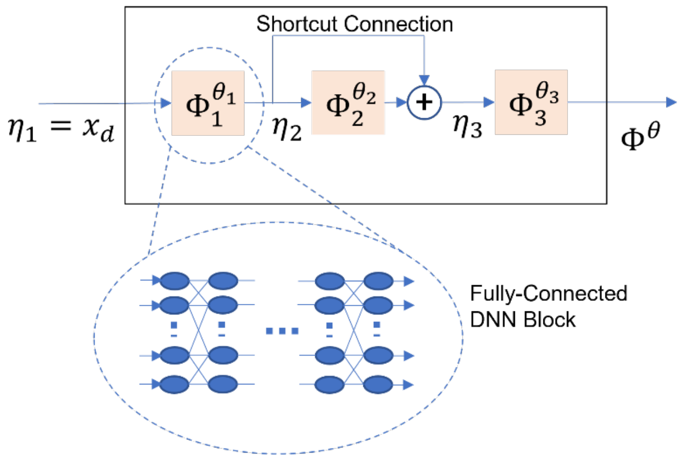
$$\frac{\partial \hat{\Phi}}{\partial \hat{\theta}} = \left[\left(\frac{\partial \hat{\Phi}}{\partial \mathrm{vec}(\hat{V}_0)}\right), \ldots, \left(\frac{\partial \hat{\Phi}}{\partial \mathrm{vec}(\hat{V}_k)}\right)\right]$$

$$\frac{\partial \hat{\Phi}}{\partial \mathrm{vec}(\hat{V}_0)} = \left(\overset{\curvearrowleft}{\prod}_{l=1}^{k} \hat{V}_l^{T} \hat{\varphi}_l'\right)\left(I_{L_1} \otimes x_d^{T}\right)$$

$$\frac{\partial \hat{\Phi}}{\partial \mathrm{vec}(\hat{V}_j)} = \left(\overset{\curvearrowleft}{\prod}_{l=j+1}^{k} \hat{V}_l^{T} \hat{\varphi}_l'\right)\left(I_{L_1} \otimes \hat{\varphi}_j^{T}\right)$$

# ResNet Architecture

- ResNets contain shortcut connections
- A ResNet can be modeled using fully-connected blocks as



$$\Phi^{\theta}(\eta_1) \triangleq \Phi_3^{\theta_3}\left(\Phi_2^{\theta_2}\left(\Phi_1^{\theta_1}(\eta_1)\right) + \Phi_1^{\theta_1}(\eta_1)\right)$$

- Each fully-connected block can be expressed using the recursive relation

$$\Phi_p = V_{k,p}^T \varphi_{k,p}$$

$$\varphi_{p,j} \triangleq \begin{cases} \phi_{p,j}\left(V_{p,j-1}^T \varphi_{p,j-1}\right), & j \in \{1,\ldots,k_p\}, \\ \eta_p, & j = 0. \end{cases}$$

where $k_p$ denotes the depth of the $p^{th}$ block

## Control Design

- Adaptive Feedforward DNN Term

$$\hat{\Phi} \triangleq \Phi(x_d, \hat{V}_0, \ldots, \hat{V}_k)$$

- Control Law

ResNet

$$u \triangleq \dot{x}_d - \rho(\|e\|)e - k_1 e - k_s \operatorname{sgn}(e) - \Phi^{\hat{\theta}}(x_d)$$

- Let

$$\hat{\theta} \triangleq \begin{bmatrix} \hat{\theta}_1^T & \hat{\theta}_2^T & \hat{\theta}_3^T \end{bmatrix}^T$$

$$\hat{\theta}_p \triangleq \left[\operatorname{vec}(\hat{V}_{p,0})^T, \ldots, \operatorname{vec}(\hat{V}_{p,k_p})^T\right]^T$$

$$\hat{\varphi}_{p,j} \triangleq \varphi_{p,j}(\hat{\eta}_p, \hat{V}_{p,0}, \ldots, \hat{V}_{p,j})$$

$$\hat{\varphi}'_{p,j} \triangleq \varphi'_{p,j}(\hat{\eta}_p, \hat{V}_{p,0}, \ldots, \hat{V}_{p,j})$$

# Adaptation Law

- Adaptation Law

$$\dot{\hat{\theta}} \triangleq \mathrm{proj}\left(\Gamma \Phi'^{T} e\right)$$

where $\Phi' \triangleq \frac{\partial \Phi^{\hat{\theta}}(x_d)}{\partial \hat{\theta}}$ can be computed using the chain rule as

$$\Phi' = \left[\begin{array}{ccc} \Phi_3' \left(\Phi_2' + I_{L_{3,0}}\right) \Lambda_1 & \Phi_3' \Lambda_2 & \Lambda_3 \end{array}\right]$$

$$\Lambda_p = \left[\begin{array}{cccc} \Lambda_{p,0} & \Lambda_{p,1} & \ldots & \Lambda_{p,k_p} \end{array}\right]$$

$$\Lambda_{p,0} = \left( \overset{\curvearrowleft}{\prod}{}_{l=1}^{k_p} \hat{V}_{p,l}^{T} \hat{\varphi}_{p,l}' \right) \left( I_{p,L_{p,1}} \otimes \hat{\eta}_p^{T} \right)$$

$$\Lambda_{p,j} = \left( \overset{\curvearrowleft}{\prod}{}_{l=j+1}^{k_p} \hat{V}_{p,l}^{T} \hat{\varphi}_{p,l}' \right) \left( I_{p,L_{p,j+1}} \otimes \hat{\varphi}_{p,j}^{T} \right)$$

$$\Phi_p' = \left( \overset{\curvearrowleft}{\prod}{}_{l=1}^{k_p} \hat{V}_{p,l}^{T} \hat{\varphi}_{p,l}' \right) \hat{V}_{p,0}^{T}$$

## Stability Analysis

**Theorem.** *The designed controller and adaptation law ensure global asymptotic tracking error convergence in the sense that* $\lim_{t\to\infty} \|e(t)\| = 0$, *provided the gain condition* $\sigma_s > \overline{\varepsilon} + \overline{\Delta}$ *is satisfied.*

*Proof:*

- Candidate Lyapunov Function

$$\mathcal{V}_L(z) \triangleq \tfrac{1}{2}e^T e + \tfrac{1}{2}\tilde{\theta}^T \Gamma^{-1}\tilde{\theta}$$

$$\dot{\mathcal{V}}_L \overset{a.a.t.}{\in} e^T\left(f(x) - f(x_d) + \mathcal{O}^2\left(\left\|\tilde{\theta}\right\|\right) + \varepsilon(x_d) - \rho(\|e\|)e\right) - \sigma_e\|e\|^2$$
$$-\sigma_s e^T K[\mathrm{sgn}](e) + e^T\Phi'\tilde{\theta} - \tilde{\theta}^T\Gamma^{-1}K[\mathrm{proj}]\left(\Gamma\Phi'^T e\right)$$

$$\dot{\mathcal{V}}_L \overset{a.a.t.}{\leq} -\sigma_e\|e\|^2 + e^T\left(\mathcal{O}^2\left(\left\|\tilde{\theta}\right\|\right) + \varepsilon(x_d)\right) - \sigma_s\|e\|_1$$

- Upper-bounding yields $\dot{\mathcal{V}}_L \overset{a.a.t.}{\leq} -\sigma_e\|e\|^2$
- Invoking LaSalle-Yoshizawa theorem for nonsmooth systems yields

$$\lim_{t\to\infty}\|e(t)\| = 0$$

# Simulation Results

- System

$$f(x) = \begin{bmatrix} x_1 x_2 \tanh(x_2) + \operatorname{sech}^2(x_1) \\ \operatorname{sech}^2(x_1 + x_2) - \operatorname{sech}^2(x_2) \end{bmatrix}$$

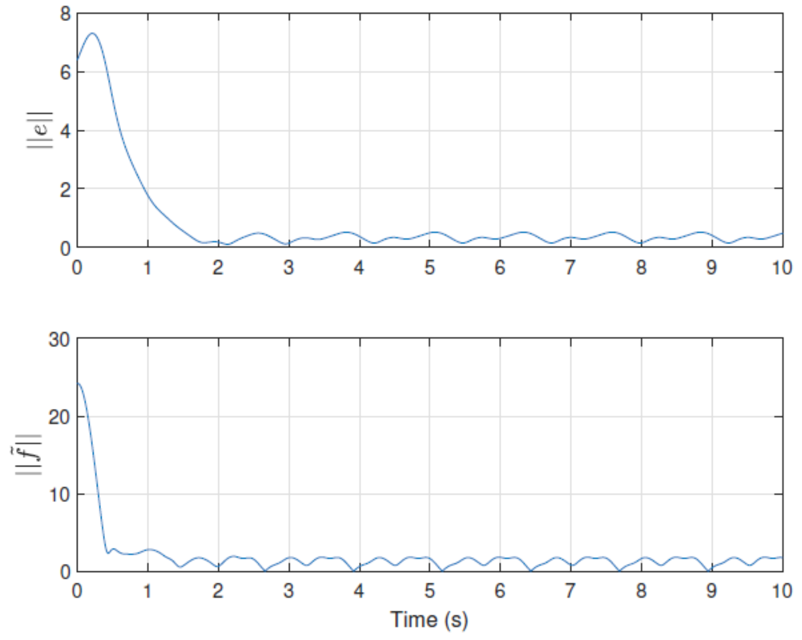
| ResNet | $k_1$ | $k_2$ | $k_3$ | Width | $\|e\|_{\mathrm{RMS}}$ | $\|\tilde{f}\|_{\mathrm{RMS}}$ |
|---|---|---|---|---|---|---|
| I | 1 | 1 | 1 | 2 | 2.400 | 5.189 |
| II | 1 | 1 | 1 | 10 | 1.625 | 4.337 |
| III | 1 | 5 | 1 | 2 | 1.687 | 3.734 |
| IV | 1 | 10 | 1 | 2 | 1.541 | 3.431 |
| V | 2 | 8 | 2 | 2 | 1.810 | 3.809 |
| VI | 3 | 6 | 3 | 2 | 1.929 | 4.053 |
| VII | 4 | 4 | 4 | 2 | 2.002 | 4.199 |

Width is the number of nodes in each hidden layer, and $k_1$, $k_2$, and $k_3$ denote the depth (i.e., number of hidden layers) of the fully-connected DNN blocks $\Phi_1$, $\Phi_2$, and $\Phi_3$, respectively.

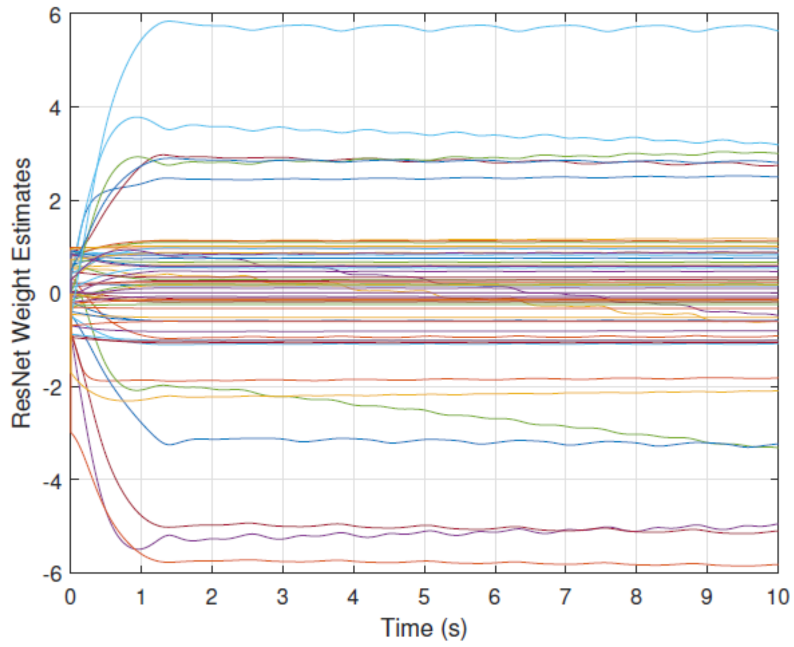
- Simulations were performed with seven ResNet configurations, each with a different depth or width. Hyperbolic tangent activation function was used.
- Based on ResNets I-IV, increasing the depth or width provided improved tracking and function approximation
- Given ResNets IV-VII with the same total depth, ResNets with deeper $\Phi_2$, i.e., shortcut connections across more layers yielded better tracking and function approximation

# Simulation Results

- ResNet VII has a slower adaptation than ResNet IV, due to vanishing gradient in $\Phi_1$ and $\Phi_3$.
- Although $\Phi_2$ is deeper in ResNet IV, the shortcut connection in ResNets circumvents vanishing gradient that occurs due to depth $\Phi_2$ .



Normalized tracking and function approximation errors with ResNet IV



Weight Estimates of ResNet IV

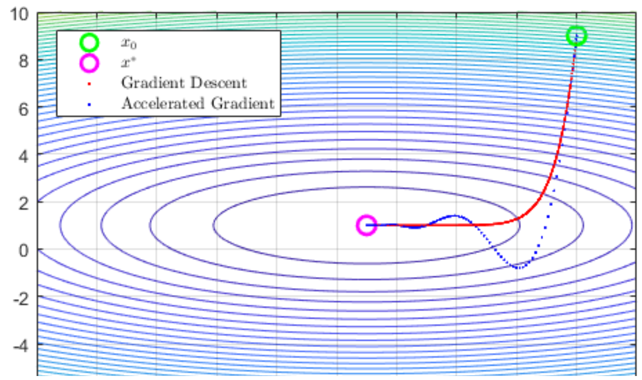
## Conclusion

- First result on Lyapunov-derived weight adaptation laws for ResNets
- ResNets with shortcut connections across more layers were found to yield better tracking and function approximation
- In future work, recurrent residual neural network architectures can be explored

## Accelerated Gradient Decent for Adaptive Control

## Motivation

- Nesterov's Accelerated Gradient
  - Add "momentum" to the update law by adding the current step a weighted version of the previous step



- GD converges in 5574 iterations
- NAG converges in 447 iterations

- Connections to continuous time analogues*
  - Dynamical systems perspective and analysis
  - Insights and heuristics on adaptation design

$$\underbrace{\ddot{\theta} + \beta\dot{\theta} = -\frac{\gamma\beta}{\mathcal{N}}\nabla f(\theta)}$$ **

$$\dot{\nu} = -\frac{\gamma}{\mathcal{N}}\nabla f(\theta)$$

$$\dot{\theta} = -\beta(\theta - \nu)\mathcal{N}$$

Technical Challenges

- Can not naively implement in closed-loop control
- Convergence of parameter estimations

*Su.Boyd.Candes, 2019
Wibisono.Wilson.Jordan, 2016
Wilson.Recht.Jordan, 2021
**Gaudio. Annaswamy.et2021

## Problem Formulation

- Control objective
  - Trajectory tracking
  - Real-time parameter estimation

**Assumption.** *The uncertain dynamics are linear-in-the-parameters (LIP) and can be expressed as*

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F\dot{q} = \underbrace{\boxed{\Psi(q,\dot{q},\ddot{q})}}_{\text{Regressor Matrix}}\underbrace{\boxed{\theta^*}}_{\text{Unknown Parameters}}$$

- Tracking and filtered tracking errors

$$e \triangleq q_d - q$$

$$r \triangleq \dot{e} + \alpha e$$

- Parameter estimation errors

$$\tilde{\theta} \triangleq \theta^* - \hat{\theta}$$

## Control Design

- Control Input $$\tau \triangleq kr + Y\hat{\theta} + e - 2Y\left(\hat{\theta} - \nu\right)$$
- Higher-order adaptation laws (implementable form)

$$\dot{\nu} \triangleq \Gamma\left(Y^T r + k_1 \sum_{i=1}^{N} \Psi_{f,i}^T\left(\tau_{f,i} - \Psi_{f,i}^T \nu\right)\right)$$

$$\dot{\hat{\theta}} \triangleq -\Gamma\left(k_2\left(\hat{\theta} - \nu\right) - k_1 \sum_{i=1}^{N} \Psi_{f,i}^T\left(\tau_{f,i} - \Psi_{f,i}^T \nu\right)\right)$$

- Higher-order adaptation laws (analysis form)

$$\dot{\nu} = \Gamma\left(Y^T r + k_1 \sum_{i=1}^{N} \Psi_{f,i}^T \Psi_{f,i}\left(\theta^* - \nu\right)\right)$$

$$\dot{\hat{\theta}} = -\Gamma\left(k_2\left(\hat{\theta} - \nu\right) - k_1 \sum_{i=1}^{N} \Psi_{f,i}^T \Psi_{f,i}\left(\theta^* - \nu\right)\right)$$

**Theorem.** *Consider a general Euler-Lagrange system that satisfies the stated properties. Let the stated assumptions hold. The controller and ICL-based higher-order adaptive update laws ensure the equilibrium point* $z = 0_{2n+2m}$ *is globally exponentially stable, i.e.,*

$$\|z(t)\| \le \frac{c_2}{c_1}\|z(0)\| \exp(\lambda T) \exp(-\lambda t),$$

## Simulations

- Simulation 1 – Standard Adaptive

$$\tau \triangleq kr + Y\hat{\theta} + e$$
$$\dot{\hat{\theta}} \triangleq \Gamma Y^T r$$

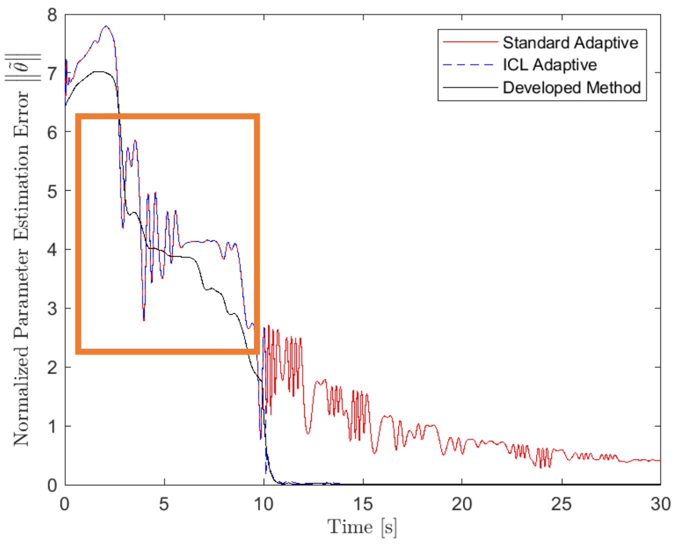
- Simulation 2 – ICL Adaptive

$$\dot{\hat{\theta}} \triangleq \Gamma\left(Y^T r + k_1 \sum_{i=1}^{N} \Psi_{f,i}^T\left(\tau_i - \Psi_{f,i}\hat{\theta}\right)\right)$$

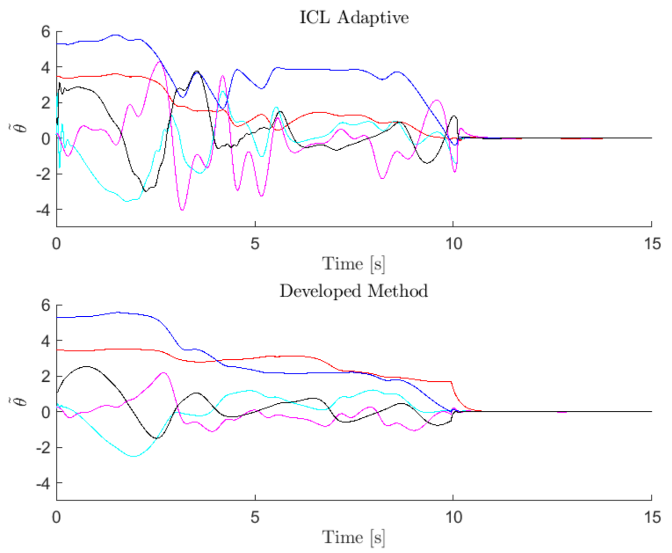
- Simulation 3 – Developed Method

$$\tau \triangleq kr + Y\hat{\theta} + e - 2Y\left(\hat{\theta} - \nu\right)$$
$$\dot{\nu} \triangleq \Gamma\left(Y^T r + k_1 \sum_{i=1}^{N} \Psi_{f,i}^T\left(\tau_{f,i} - \Psi_{f,i}^T\nu\right)\right)$$
$$\dot{\hat{\theta}} \triangleq -\Gamma\left(k_2\left(\hat{\theta} - \nu\right) - k_1 \sum_{i=1}^{N} \Psi_{f,i}^T\left(\tau_{f,i} - \Psi_{f,i}^T\nu\right)\right)$$

# Simulations



Evolution of the normalized parameter estimation error trajectories for each simulation. The red line represents the simulation using the standard adaptive method. The blue line represents the simulation using the ICL adaptive method. The black line represents the simulation using the developed method.



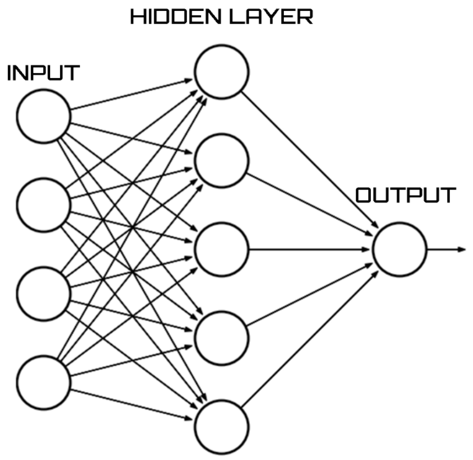
(top): Parameter estimation error using the ICL adaptive method. (bottom): Parameter estimation error using the developed method.

## Neural Network Extension

Dynamics

- Unknown/unstructured model uncertainty (i.e., does not satisfy the LIP assumption)

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



**Neural Network Model**

$$f(x_d) = W^{*T}\sigma\left(V^{*T}\overline{x}_d\right) + \varepsilon(x_d), \forall x_d \in \Omega$$

$W^*$ denotes the ideal output-layer weights
$V^*$ denotes the ideal hidden-layer weights

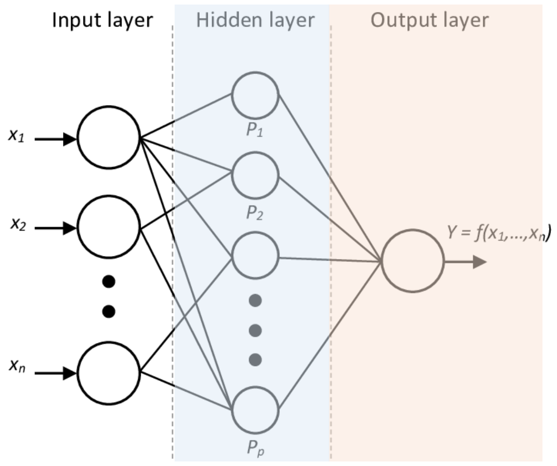
**Assumption.** *The ideal NN weights can be bounded as* $\|W^*\|_F \leq \overline{W}$ *and* $\|V^*\|_F \leq \overline{V}$, *where* $\overline{W}, \overline{V} \in \mathbb{R}_{>0}$ *are known constants.*

## Controller and Adaptation Laws

Higher-order output-layer weight adaptatic

$$\dot{\hat{\omega}} \triangleq \mathrm{proj}\left(\Gamma_\omega \sigma\left(\hat{V}^T \overline{x}_d\right) e^T\right)$$

$$\dot{\hat{W}} \triangleq -\mathrm{proj}\left(\Gamma_\omega \Gamma_W \tilde{W}\right)$$



Higher-order hidden-layer weight adaptation laws

$$\dot{\hat{\nu}} \triangleq \mathrm{proj}\left(\Gamma_\nu \overline{x}_d e^T \hat{W}^T \sigma'\left(\hat{V}^T \overline{x}_d\right)\right)$$

$$\dot{\hat{V}} \triangleq -\mathrm{proj}\left(\Gamma_\nu \Gamma_V \tilde{V}\right)$$

Control input

$$u \triangleq \dot{x}_d - k_e e - k_s \mathrm{sgn}\left(e\right) - \rho\left(\|e\|\right) e - \hat{W}^T \sigma\left(\hat{V}^T \overline{x}_d\right) + \mu$$

$$\mu \triangleq 2\tilde{W}^T \sigma\left(\hat{V}^T \overline{x}_d\right) + 2\hat{W}^T \sigma'\left(\hat{V}^T \overline{x}_d\right) \tilde{V}^T \overline{x}_d$$

## Sketch Proof

**Theorem.** *Consider a general uncertain nonlinear system that satisfies the stated assumptions. The control input and higher-order weight adaptation laws in ensure global asymptotic tracking in the sense that* $\lim_{t\to\infty}\|e(t)\| = 0$, $\lim_{t\to\infty}\left\|vec\left(\tilde{W}\right)\right\| = 0$, *and* $\lim_{t\to\infty}\left\|vec\left(\tilde{V}\right)\right\| = 0$, *provided* $k_s > c_1$ *and* $k_e > c_2$.

## Lyapunov Function

$$V_L(z) \triangleq \frac{1}{2}e^T e + \frac{1}{2}\mathrm{vec}\left(\tilde{W}\right)^T\left(I_{L+1}\otimes\Gamma_\omega^{-1}\right)\mathrm{vec}\left(\tilde{W}\right) + \frac{1}{2}\mathrm{vec}\left(\tilde{\omega}^*\right)^T\left(I_{L+1}\otimes\Gamma_\omega^{-1}\right)\mathrm{vec}\left(\tilde{\omega}^*\right)$$
$$+\frac{1}{2}\mathrm{vec}\left(\tilde{V}\right)^T\left(I_{n+1}\otimes\Gamma_\nu^{-1}\right)\mathrm{vec}\left(\tilde{V}\right) + \frac{1}{2}\mathrm{vec}\left(\tilde{\nu}^*\right)^T\left(I_{n+1}\otimes\Gamma_\nu^{-1}\right)\mathrm{vec}\left(\tilde{\nu}^*\right)$$

$$\dot{V}_L = e^T\mathrm{K}\left[\dot{e}\right] + \mathrm{vec}\left(\tilde{W}\right)^T\mathrm{vec}\left(\Gamma_\omega^{-1}\mathrm{K}\left[\dot{\hat{W}}\right]\right) + \mathrm{vec}\left(\tilde{V}\right)^T\mathrm{vec}\left(\Gamma_\nu^{-1}\mathrm{K}\left[\dot{\hat{V}}\right]\right)$$
$$+\mathrm{vec}\left(-\tilde{W}^* - 2\tilde{W}\right)^T\mathrm{vec}\left(\Gamma_\omega^{-1}\mathrm{K}\left[\dot{\hat{\omega}}\right]\right) + \mathrm{vec}\left(-\tilde{V}^* - 2\tilde{V}\right)^T\mathrm{vec}\left(\Gamma_\nu^{-1}\mathrm{K}\left[\dot{\hat{\nu}}\right]\right)$$

$$\cdots \quad \dot{V}_L \overset{\text{a.e.}}{\leq} -\lambda\|e\|^2 - \lambda_W\left\|\tilde{W}\right\|_F^2 - \lambda_V\left\|\tilde{V}\right\|_F^2$$

Invoke LaSalle-Yoshizawa theorem extension for nonsmooth systems

## Simulations

Dynamics: $f(x) = \begin{bmatrix} -x_1 + x_2 \\ -\frac{1}{2}x_1 - \frac{1}{2}x_2\left(1 - (\cos(2x_1) + 2)^2\right) \end{bmatrix}$

Desired trajectory: $x_d(t) = \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}$

Simulations:

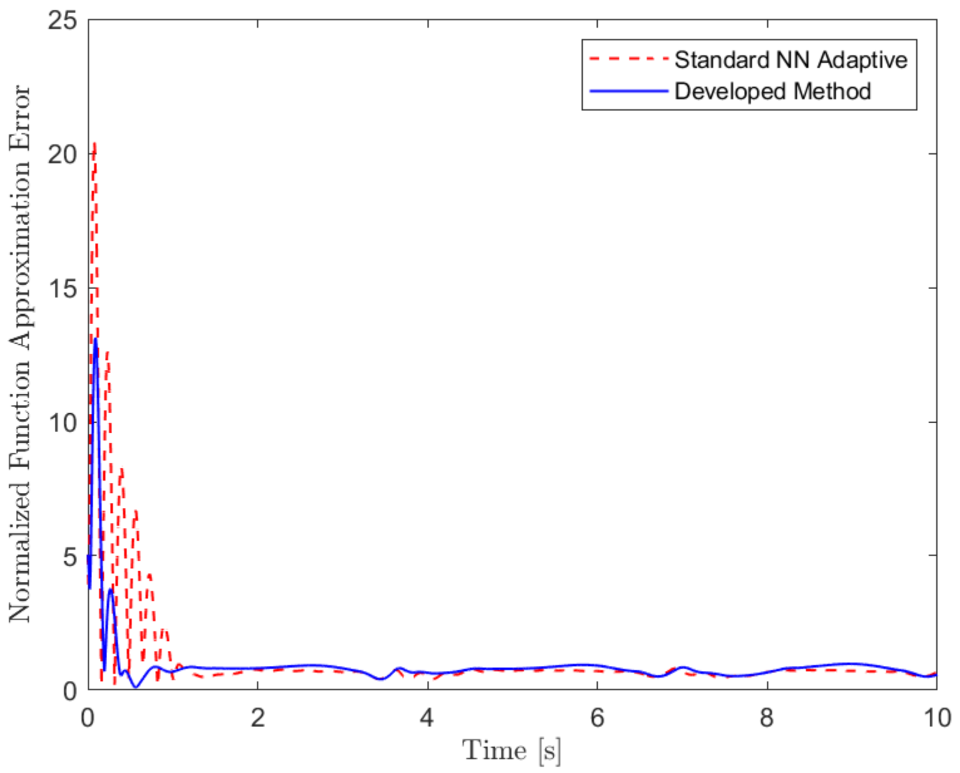
1. Developed method
2. Standard gradient-based NN adaptive controller

$$\dot{\hat{W}} \triangleq \Gamma_W \hat{\sigma} e^T \qquad \dot{\hat{V}} \triangleq \Gamma_V \overline{x}_d e^T \hat{W}^T \hat{\sigma}' \qquad u \triangleq \dot{x}_d - \hat{W}^T \hat{\sigma} - k_e e - k_s \mathrm{sgn}(e)$$

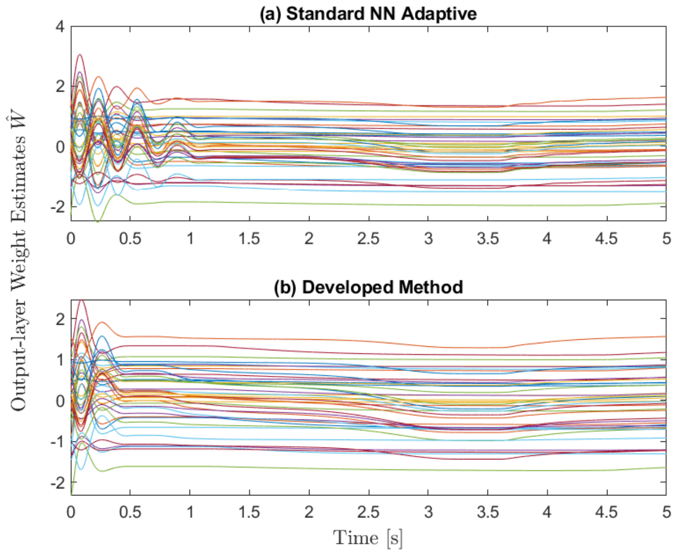
Table 1: Simulation Parameters

| Adaptation Law | $\Gamma_\omega$ | $\Gamma_\nu$ | $\Gamma_W$ | $\Gamma_V$ | $k_e$ | $k_s$ |
|---|---|---|---|---|---|---|
| Standard NN Adaptive | - | - | $50I_{21}$ | $50I_3$ | 5 | 0.5 |
| Developed Method | $50I_{21}$ | $50I_3$ | $0.9I_{21}$ | $0.9I_3$ | 5 | 0.5 |

# Simulations



Function approximation error for each simulation



Output-layer weight estimates using the developed method and the standard NN adaptive controller

## Switched Approximate Dynamic Programming and Applications to Hierarchical Reinforcement Learning

## Outline

- Model-Based Reinforcement Learning for Optimal Feedback Control of Switched Systems
  - Time-Based Switched ADP
  - Facilitate Trajectory Tracking
  - Switched Multiple Lyapunov UUB Stability Theorem

- Hierarchical Reinforcement Learning-based Supervisory Control of Unknown Nonlinear Systems
  - Builds on Switched ADP Result
  - Switch Between Multiple Control Policies
  - Optimal Value Function-Based Hierarchical Policy

# Switched ADP

## Dynamical System

Given a control affine nonlinear dynamical system:

$$\dot{\zeta} = F(\zeta) + G(\zeta)\mu$$

## Control Objective

Design a controller, $\mu$, which minimizes a cost function:

$$J(\zeta, \mu) = \min_{\mu(\tau)\epsilon U} \int_0^\infty Q\big(\zeta(\tau)\big) + \mu(\tau)^T R\mu(\tau)\ d\tau$$

## Cost-to-Go

Optimal value function:

$$V^*(x) = \min_{\mu(\tau)\epsilon U} \int_t^\infty Q\big(\zeta(\tau)\big) + \mu(\tau)^T R\mu(\tau)\ d\tau$$

## Switched ADP

**Optimal Value Function and Optimal Control Policy:**

$$V^*(\zeta) = W^T\sigma(\zeta) + \varepsilon(\zeta) \qquad \mu^*(\zeta) = -\frac{1}{2}R^{-1}G(\zeta)^T\left(\nabla_\zeta\sigma(\zeta)^T W + \nabla_\zeta\varepsilon(\zeta)^T\right)$$

Unknown: Neural weights $\quad \widehat{W}_c, \widehat{W}_a \to W$

$\widehat{W}_c$: Critic weight

$\widehat{W}_a$: Actor weight

**Value Function and Optimal Control Policy Approximation**

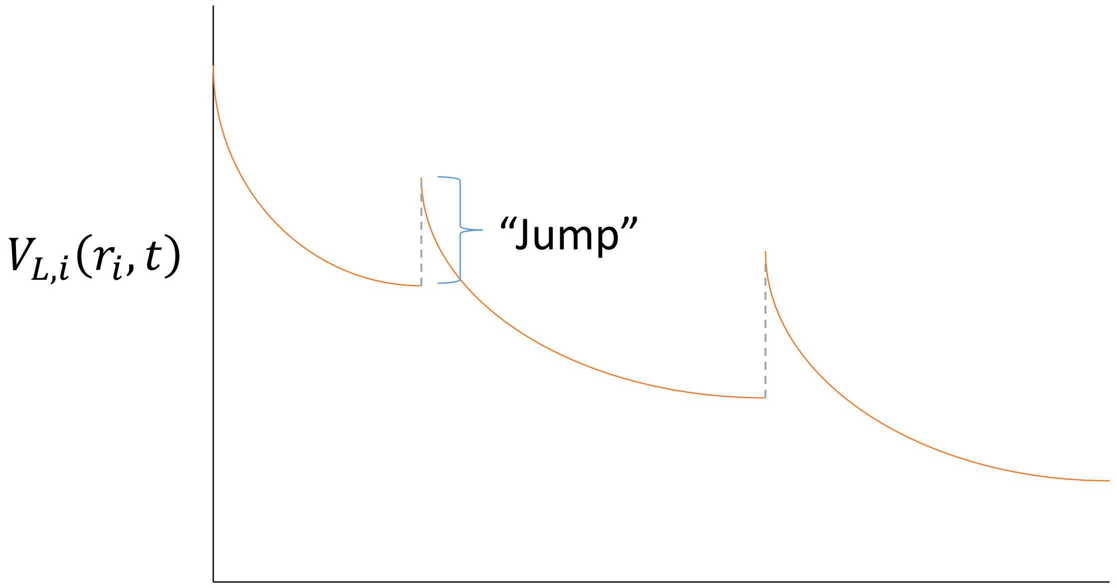
$$\hat{V}\left(\zeta, \widehat{W}_c\right) = \widehat{W}_c^{\,T}\sigma(\zeta) \qquad \hat{\mu}\left(\zeta, \widehat{W}_a\right) = -\frac{1}{2}R^{-1}G(\zeta)^T\left(\nabla_\zeta\sigma(\zeta)^T\widehat{W}_a\right)$$

**Bellman Error (BE): Residual from HJB**

$$\hat{\delta}\left(\zeta, \widehat{W}_c, \widehat{W}_a\right) \triangleq Q(\zeta) + \hat{\mu}\left(\zeta, \widehat{W}_a\right)^T R\hat{\mu}\left(\zeta, \widehat{W}_a\right) + \nabla_\zeta\hat{V}\left(\zeta, \widehat{W}_c\right)\left(\hat{F}_i(\zeta) + G(\zeta)\hat{\mu}\left(\zeta, \widehat{W}_a\right)\right)$$

## Switched ADP

- ADP Subsystem Stability is Well-Understood
- Switched ADP is More Complicated



- Each subsystem must remain active until it has decayed past its subsequent jump.

# Switched ADP

- **Initial Result:**
  - Analyze Subsystems Separately

$$V_{L,i}(r_i, t) = V_i^*(e, t) + \frac{1}{2}\widetilde{W}_{c,i}^{\,T}\Gamma_i^{-1}\widetilde{W}_{c,i} + \frac{1}{2}\widetilde{W}_{a,i}^{\,T}\widetilde{W}_{a,i}$$

  - Switched Lyapunov-Based Analysis

- **Problem #1:** Unknown Value Function $V_i^*$
  - Common Lyapunov Function?
- **Problem #2:** Lyapunov Function Decay Rate
  - Assumptions on quadratic bound on Lyapunov function required for exponential stability
- **Problem #3:** Subsystem State Vector $r_i$
  - "Discontinuous" States (i.e., $r_i(t) \neq r_{i+1}(t)$)?

## Switched UUB Theorem (Informal)

*Given that each subsystem is UUB in the sense of Theorem 4.18 of Khalil and let* $t_\sigma = \{t_0, t_1, t_2, \dots\}$ *represents a switching sequence. If the minimum dwell-time condition*

$$\tau(t_i) \geq \begin{cases} \dfrac{\alpha_{2,\sigma(t_i)}(\|r(t_i)\|) - \alpha_{1,\sigma(t_i^-)}(\|r(t_i)\|)}{\kappa} & V_{\sigma(t_i)}(r(t_i), t_i) > \bar{\alpha} \\ > 0 & V_{\sigma(t_i)}(r(t_i), t_i) \leq \bar{\alpha} \end{cases}$$

*is satisfied, then the trajectories of the switched system converge to a bounded region given by* $\lim_{t\to\infty} r(t) \leq \max_{p\in P} \alpha_{1,p}^{-1}(\bar{\alpha})$.

# Switched ADP

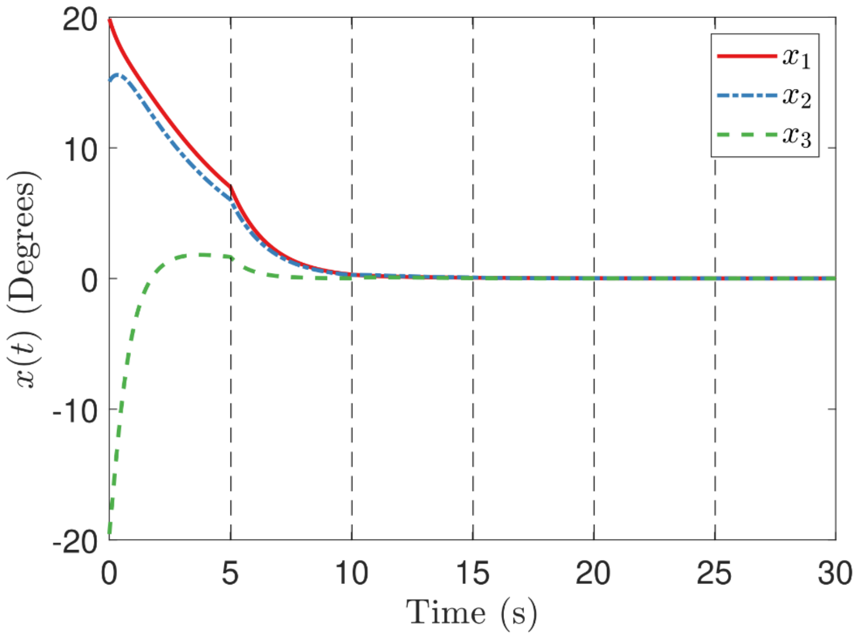
- F-16 Longitudinal Dynamics
  - [Stevens, Lewis, Johnson, 2016]

| | 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 | $\dot{x} = \begin{bmatrix} -1 & 0.5 & -0.02 \\ 0.9 & -0.8 & -0.4 \\ 0 & 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$ |

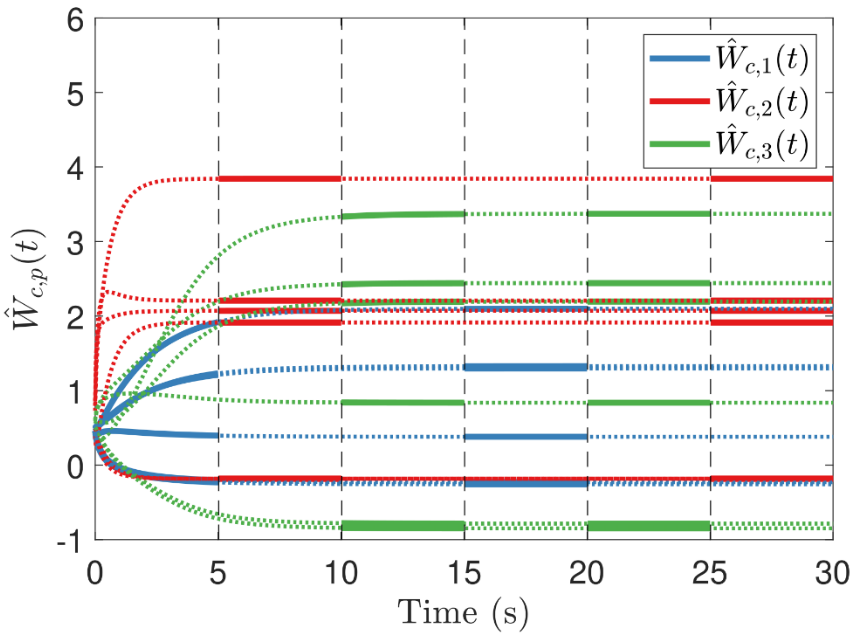
# Switched ADP

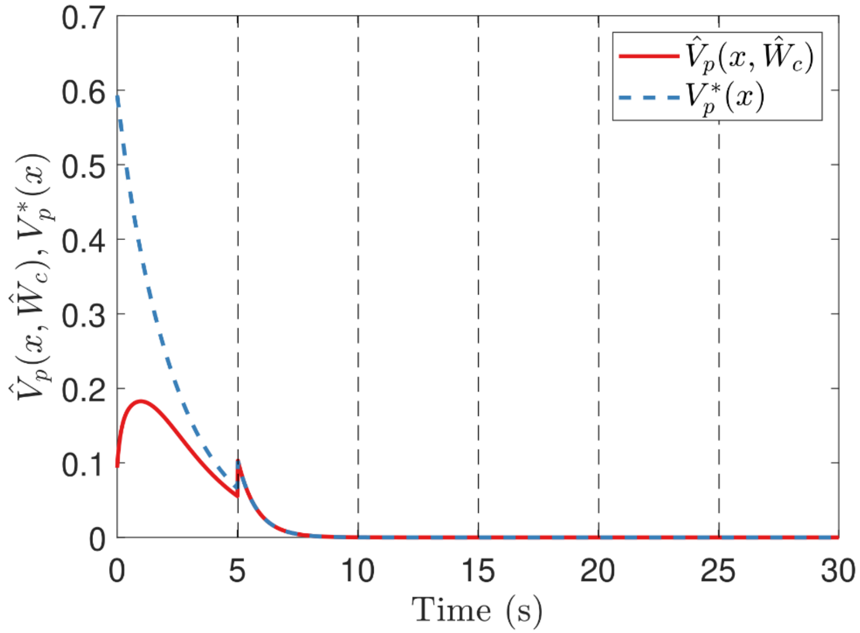
- Switch between multiple dynamical systems
    - Arbitrary switching sequence
    - Satisfies minimum dwell-time condition

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



# Switched ADP

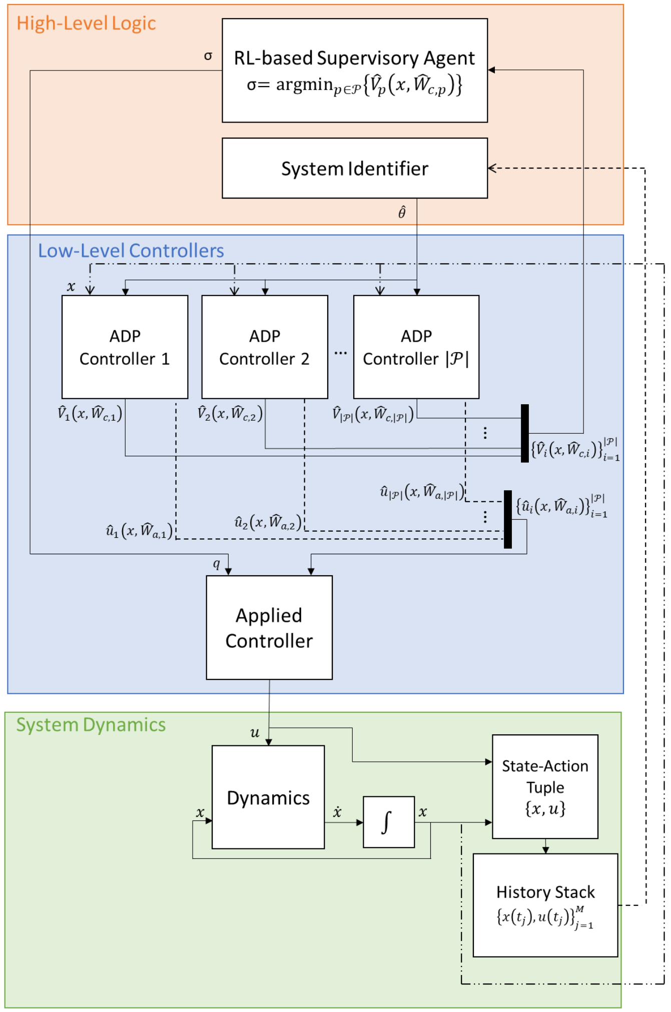




## Hierarchical Reinforcement Learning-based Supervisory Control of Unknown Nonlinear Systems

# Hierarchical ADP

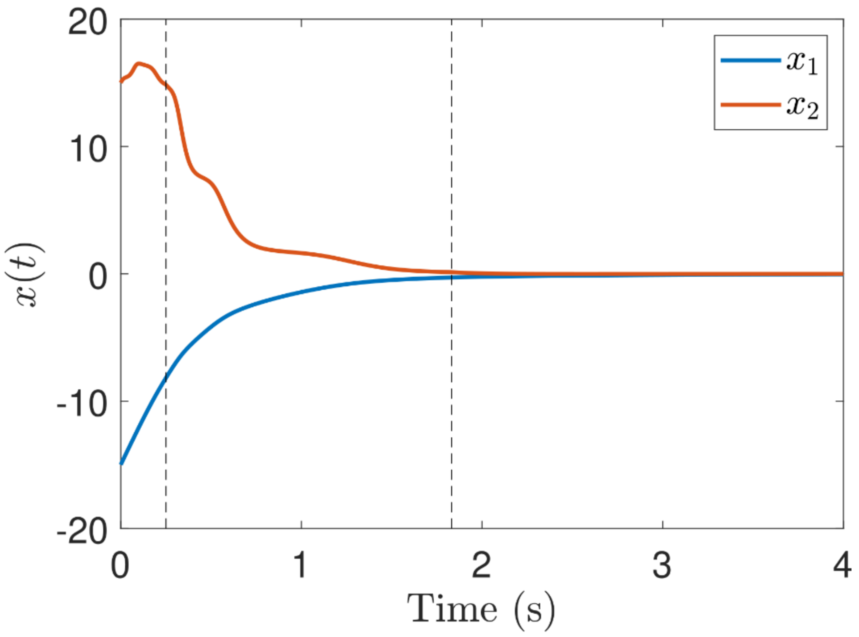
- Hierarchical ADP
  - Hierarchical Agent
    - Switching Logic
    - Approximation of Optimal Value Function
  - ADP Sub-Policies
    - Each ADP Policy Learning Separately
    - 1 Control Policy is Selected by HRL Agent
  - System Identification
    - Approximation of Drift Dynamics
    - Approximation used to Update Model-Based ADP Weights

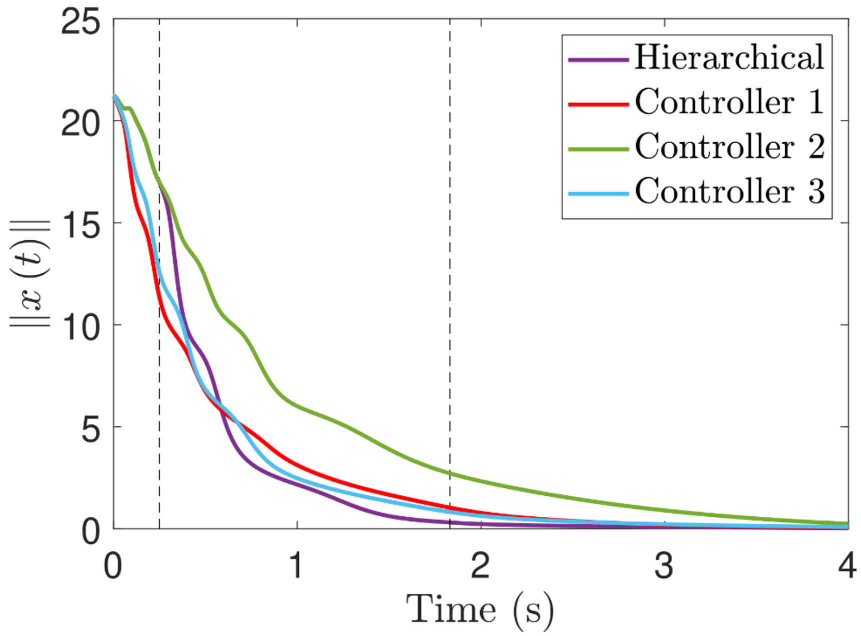

High-Level Logic
RL-based Supervisory Agent
$\sigma = \text{argmin}_{p\in\mathcal{P}}\{\hat{V}_p(x,\widehat{W}_{c,p})\}$
$\sigma$
System Identifier
$\hat{\theta}$
Low-Level Controllers
$x$
ADP Controller 1
ADP Controller 2
...
ADP Controller $|\mathcal{P}|$
$\hat{V}_1(x,\widehat{W}_{c,1})$
$\hat{V}_2(x,\widehat{W}_{c,2})$
$\hat{V}_{|\mathcal{P}|}(x,\widehat{W}_{c,|\mathcal{P}|})$
$\{\hat{V}_i(x,\widehat{W}_{c,i})\}_{i=1}^{|\mathcal{P}|}$
$\hat{u}_{|\mathcal{P}|}(x,\widehat{W}_{a,|\mathcal{P}|})$
$\{\hat{u}_i(x,\widehat{W}_{a,i})\}_{i=1}^{|\mathcal{P}|}$
$\hat{u}_1(x,\widehat{W}_{a,1})$
$\hat{u}_2(x,\widehat{W}_{a,2})$
$q$
Applied Controller
System Dynamics
$u$
Dynamics
$x$
$\dot{x}$
$\int$
$x$
State-Action Tuple $\{x,u\}$
History Stack $\{x(t_j),u(t_j)\}_{j=1}^{M}$

## Hierarchical ADP Stability Analysis (Sketch)

1. Each ADP Subsystem Learns Separately
2. Simultaneous System Identification
3. Each Subsystem is UUB in the Sense of Khalil Thm. 4.18
4. HRL Agent Switches Active Policy
   - Based on Optimal Value Function Appx. & Dwell-Time
5. Leverage Switched ADP Stability Result
6. *The trajectories of the switched system converge to a bounded region given by* $\lim_{t\to\infty} r(t) \leq \max_{p\in P} \alpha_{1,p}^{-1}(\bar{\alpha})$

# Hierarchical ADP





# Hierarchical ADP

- Simulation Performance
  - Total Cost
    - Cost of Implementing Each Controller
    - **HRL Costs Less by 37%**
  - Rise Time
    - Time to Reach 1% of the Initial State Error
    - **HRL Faster by 30%**

| Controller | Total Cost | 99% Rise Time (s) |
|---|---|---|
| HRL Controller | 1073 | 2.08 |
| Controller 1 | 2683 | 2.97 |
| Controller 2 | 1701 | 4.11 |
| Controller 3 | 1940 | 3.07 |

## High Order Control Barrier Functions

# High Order Control Barrier Functions

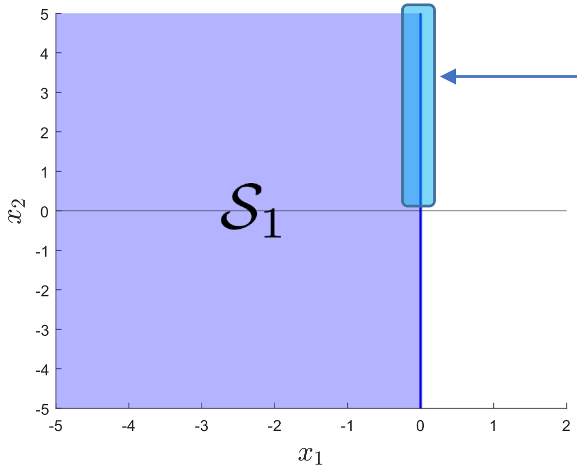
$$\dot{x} \in F(x, u) \qquad \mathcal{S}_1 \triangleq \{x \in \mathbb{R}^n : B_1(x) \leq 0\}$$

- Previously, we have developed control barrier functions (CBF) for situations where safety-ensuring control inputs exist:

$$K_c(x) \triangleq \{u \in \mathbb{R}^m : \Gamma_1(x, u) \leq -\gamma_1(x)\} \qquad \Gamma_1(x, u) \triangleq \sup_{f \in F(x,u)} \langle \nabla B_1(x), f \rangle$$

- High-order CBFs are used when safety cannot be assured in certain regions of the safe set
  - Often occurs because barrier function does not depend on a state whose dynamics depend on the control input

$$\dot{x}_1 = x_2 \qquad B_1(x) \triangleq x_1$$
$$\dot{x}_2 = u \qquad \mathcal{S}_1 = \{x \in \mathbb{R}^2 : x_1 \leq 0\}$$



Trajectories starting with positive velocity near the boundary will always exit the safe set

# High Order Control Barrier Functions

- Assuming $\Gamma_i$ doesn't depend on the control input, we define a new CBF

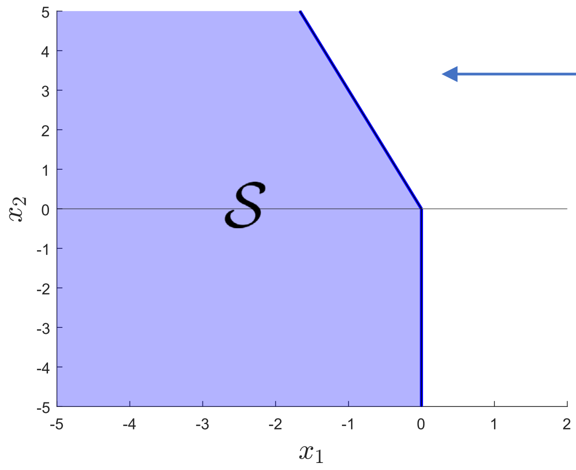
$$B_{i+1}(x) \triangleq \Gamma_i(x) + \gamma_i(x) + \epsilon_i$$

  - Defines a safe set $\mathcal{S}_{i+1} = \{x \in \mathbb{R}^n : \Gamma_i(x) \leq -\gamma_i(x) - \epsilon_i\}$ where $K_i(x) = \{u \in \mathbb{R}^m : \Gamma_i(x) \leq -\gamma_i(x)\}$ is nonempty
  - The parameter $\epsilon_i > 0$ provides some robustness and is needed for theoretical reasons

$$B_2(x) = x_2 + K_b x_1 + \epsilon_1$$

$$\mathcal{S}_2 = \{x \in \mathbb{R}^2 : B_2(x) \leq 0\}$$

$$\mathcal{S} = \mathcal{S}_1 \cap \mathcal{S}_2$$



Problem states are removed from the safe set

## High Order Control Barrier Functions

- Recursively define multiple CBFs until for some $k > 1$ the following set is nonempty

$$K_k(x) \triangleq \{u \in \mathbb{R}^m : \Gamma_k(x,u) \leq -\gamma_k(x)\}$$

in which case the set $\mathcal{S} \triangleq \cap_{i=1}^{k} \mathcal{S}_i$ can be made forward invariant using any control law $\kappa(x) \in K_k(x)$

- Our results apply to a more general class of dynamics than current HOCBF formulations
  - Apply to problems with additional HOCBFs and traditional CBFs