# Recent Advancements in Faster and Deeper Learning

Daniel Le, Max Greene, Runhan Sun, Omkar Patil, and Warren Dixon
Department of Mechanical and Aerospace Engineering
University of Florida

AFOSR Center of Excellence Review @ Zoom

November 9th , 2021

# Overview

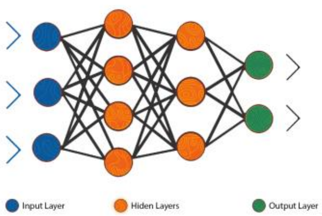
- Real-time learning and control of dynamical systems

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

- Unstructured Uncertainty
  - Deep Neural Networks to approximate model uncertainty
  - Real-time learning of DNN weights with Lyapunov-based methods

$$f(x) = W^{*T}\sigma(\Phi(x))$$

$$\Phi = \left(V_k^{*T}\phi_k \circ \ldots \circ V_1^{*T}\right)\left(V_0^{*T}x\right)$$




- Structured Uncertainty
  - Dynamics modeled as linear regression model
  - Connections between accelerated gradient methods in optimization and adaptive control

$$f(x) = Y(x)\theta^*$$

# Publications

- R. Sun, M. Greene, D. Le, Z. Bell, G. Chowdhary, and W. Dixon, "Lyapunov-based Real-Time and Iterative Adjustment of Deep Neural Networks," IEEE Control Sys. Lett. 2022, Link
- D. Le, M. Greene, W. Makumi, and W. Dixon, "Real-time Modular Deep Neural Network-Based Adaptive Control of Nonlinear Systems," IEEE Control Sys. Lett. 2022, Link
- O. Patil, D. Le, M. Greene, and W. Dixon, "Lyapunov-Derived Control and Adaptive Update Laws for Inner and Outer Layer Weights of a Deep Neural Network," (Submitted to LCSS/ACC)
- D. Le, O. Patil, P. Amy, and W. Dixon, "Integral Concurrent Learning-Based Accelerated Gradient Adaptive Control of Uncertain Euler-Lagrange Systems," (Submitted to ACC 2022)

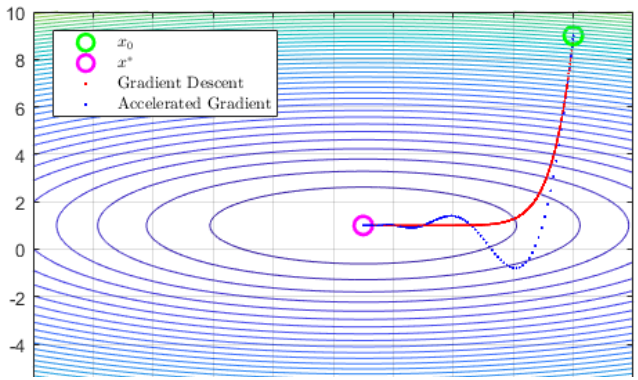
# Integral Concurrent Learning-Based Accelerated Gradient Adaptive Control of Uncertain Euler-Lagrange Systems

Daniel Le, Omkar Sudhir Patil, Patrick Amy, and  Warren Dixon

Submitted to ACC 2022

# 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 on adaptation design

$$\ddot{\hat{\theta}} + c\,(t)\,\dot{\hat{\theta}} = -k\nabla f\left(\hat{\theta}\right)$$

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

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

# Problem Formulation

- General uncertain nonlinear Euler-Lagrange dynamics

$$M\left(q\right)\ddot{q}+V_m\left(q,\dot{q}\right)\dot{q}+G\left(q\right)+F\dot{q}=\tau$$

| | |
|---|---|
| $q, \dot{q}, \ddot{q}$ | generalized position, velocity, and acceleration |
| $M$ | inertia |
| $V_m$ | centripetal coriolis |
| $G$ | generalized potential effects |
| $F$ | generalized disspation effects |
| $\tau$ | control input |

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

$$M\left(q\right)\ddot{q}+V_m\left(q,\dot{q}\right)\dot{q}+G\left(q\right)+F\dot{q}=\Psi\left(q,\dot{q},\ddot{q}\right)\theta^*$$

Unknown Parameters

Regressor Matrix

# Problem Formulation

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

- 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)

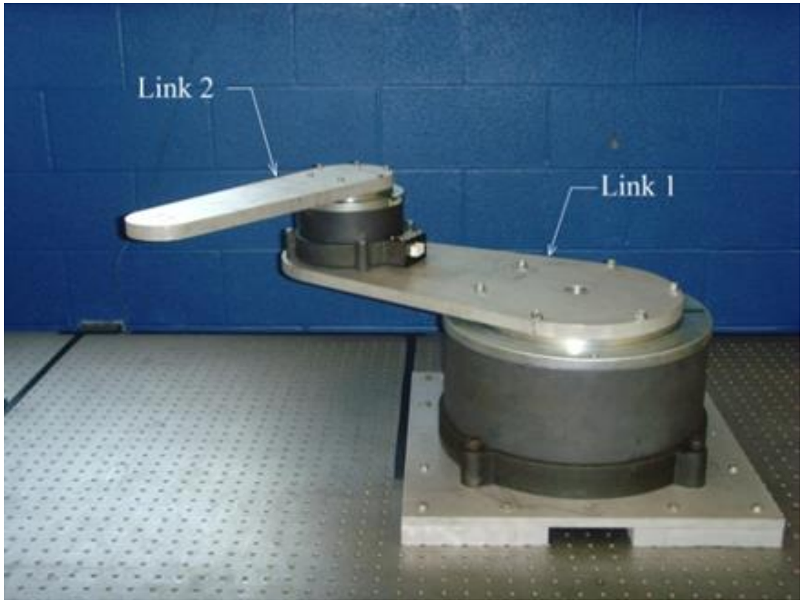
$$\dot{\nu} = \Gamma\left(Y^T r + k_1 \sum_{i=1}^{N} \Psi_{f,i}^T \Psi_{f,i} \left(\theta^* - \nu\right)\right)$$

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

**Assumption.** *There exists a time* $T \in \mathbb{R}_{>0}$ *such that* $\lambda_{min}\left(\sum_{i=1}^{N} \Psi_{f,i}^T \Psi_{f,i}\right) \geq \gamma$*, where* $\gamma \in \mathbb{R}_{>0}$ *is a user-defined parameter.*

# Simulations

- Two-link robot manipulator




$$M = \begin{bmatrix} p_1 + 2p_3c_2, & p_2 + p_3c_2 \\ p_2 + p_3c_2, & p_2 \end{bmatrix}$$

$$V_m = \begin{bmatrix} -p_3s_2\dot{q}_2, & -p_3s_2\left(\dot{q}_1 + \dot{q}_2\right) \\ p_3s_2\dot{q}_1, & 0 \end{bmatrix}$$

$$F = \begin{bmatrix} f_1, & 0 \\ 0, & f_2 \end{bmatrix}$$

- Parameters: $p_1 = 3.473,\ p_2 = 0.196,\ p_3 = 0.242,\ f_1 = 5.3,$ and $f_2 = 1.1$

- Desired trajectory $q_d \triangleq \begin{bmatrix} \cos(0.5t) \\ 2\cos(t) \end{bmatrix}$

# Simulations

- Simulation 1 – Standard Adaptive

$$\tau \triangleq kr + Y\hat{\theta} + e$$

$$\dot{\hat{\theta}} \triangleq \Gamma Y^T r$$

- Simulation 2 – ICL Adaptive

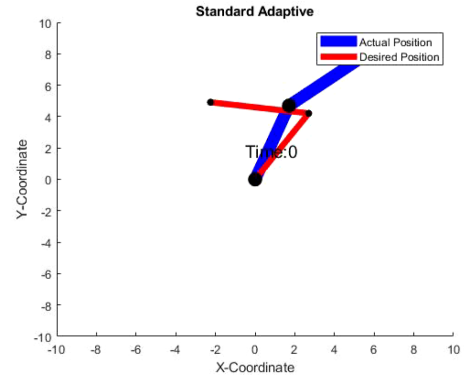
$$\dot{\hat{\theta}} \triangleq \Gamma \left( Y^T r + k_1 \sum_{i=1}^{N} \Psi_{f,i}^T \left( \tau_i - \Psi_{f,i} \hat{\theta} \right) \right)$$

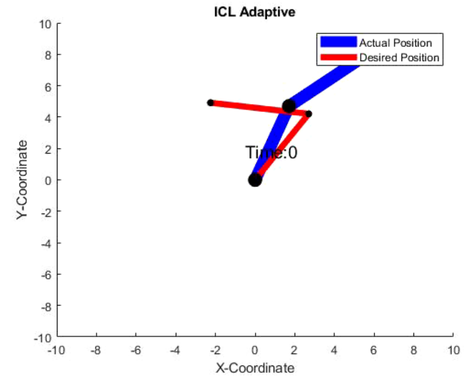
- Simulation 3 – Developed Method

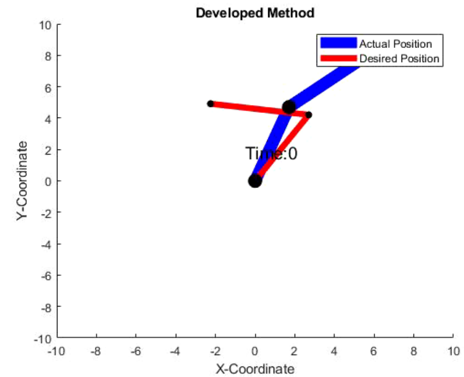
$$\tau \triangleq kr + Y\hat{\theta} + e - 2Y \left( \hat{\theta} - \nu \right)$$

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

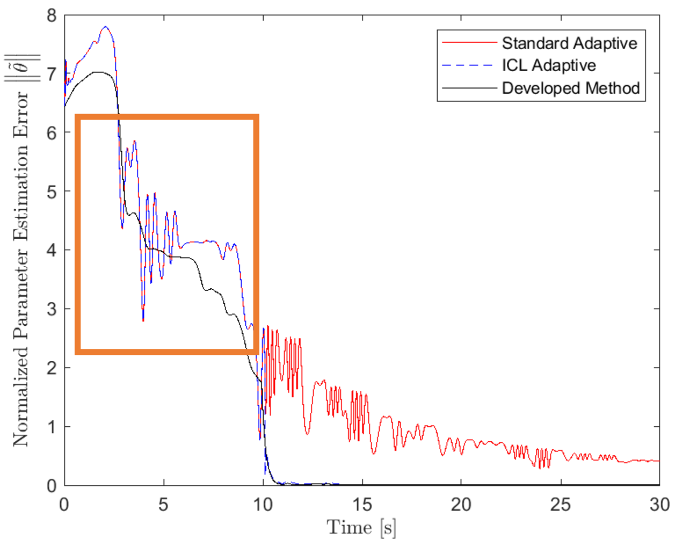
$$\dot{\hat{\theta}} \triangleq -\Gamma \left( k_2 \left( \hat{\theta} - \nu \right) - k_1 \sum_{i=1}^{N} \Psi_{f,i}^T \left( \tau_{f,i} - \Psi_{f,i}^T \nu \right) \right)$$



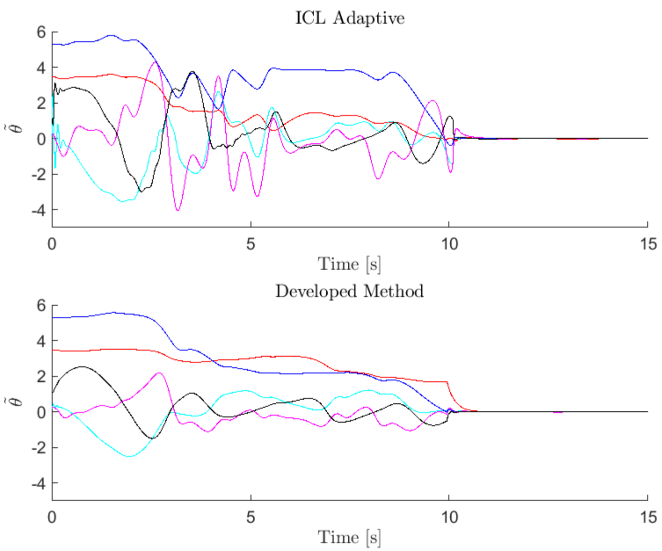



# Simulations




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.

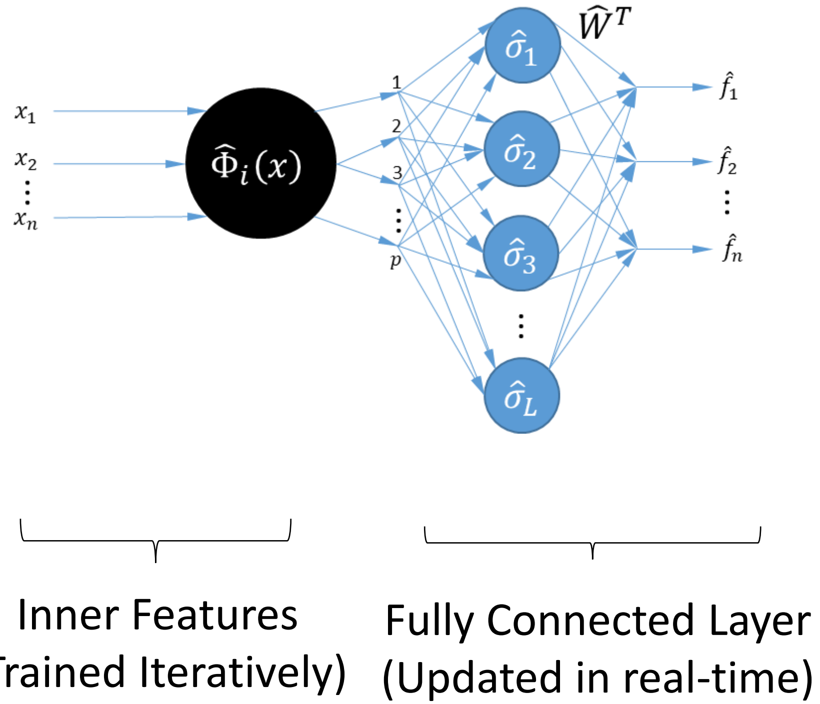
# Iterative Deep Neural Network System Identification

Max L. Greene

R. Sun, M. Greene, D. Le, Z. Bell, G. Chowdhary, and W. E. Dixon, "Lyapunov-Based Real-Time and Iterative Adjustment of Deep Neural Networks," IEEE Control Systems Letters, Vol. 6, pp. 193-198 (2021).

# Iterative DNN Approximation

- Multi-Timescale DNN
  - Online system ID
  - Improvement over single hidden-layer
  - Inner features updated iteratively (switched)
  - Output-layer updated in real-time
  - Feedforward term in robust nonlinear controller



# Real-Time DNN Approximation

## Dynamical System

Given a control affine nonlinear dynamical system:

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

## DNN Approximation

Approximate the drift dynamics with a DNN:

Dynamics: $$f(x) = \theta^T \phi\big(\Phi^*(x)\big) + \epsilon(x)$$

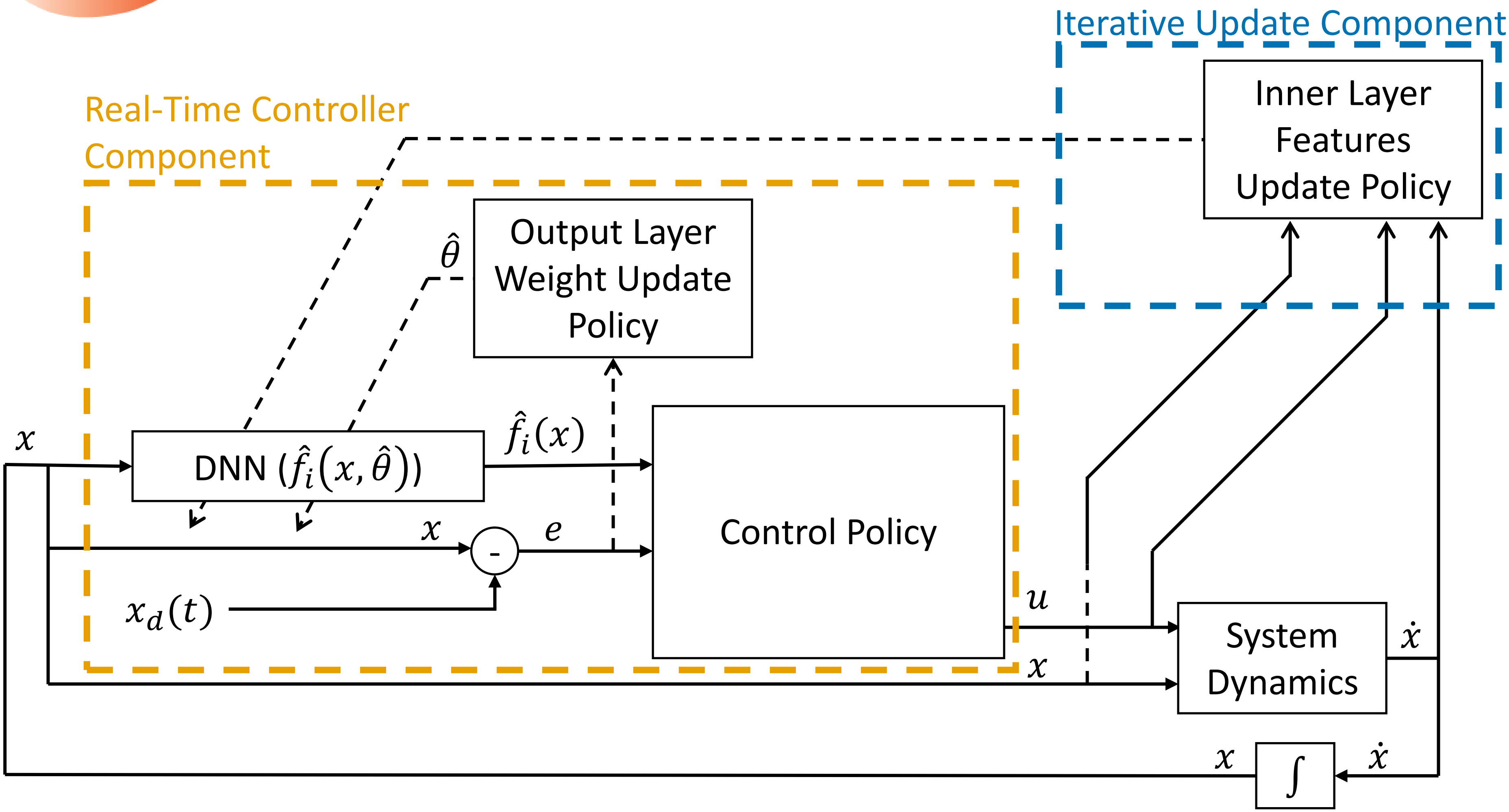
$i^{\text{th}}$ DNN Approximation: $$\hat{f}_i\big(x, \hat{\theta}\big) = \hat{\theta}^T \phi\left(\widehat{\Phi}_i(x)\right)$$

## Looks Like:

Reminds us of linearly parameterizable dynamics:

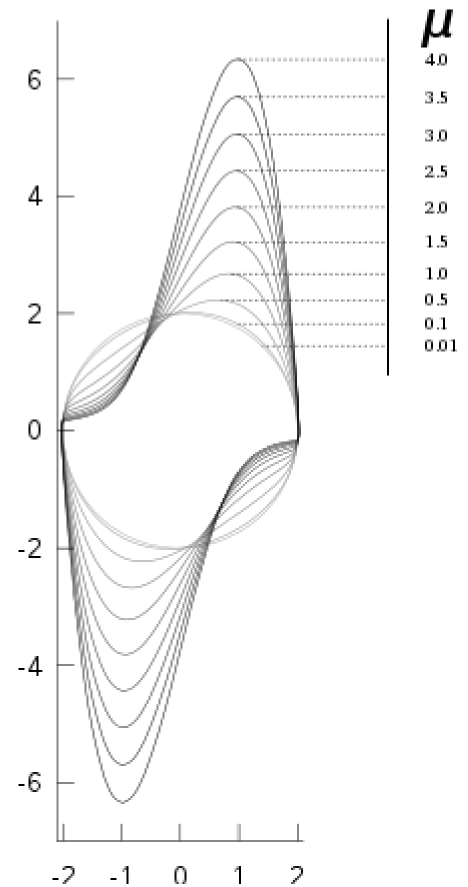
$$f(x) = \theta^T \phi\big(\Phi^*(x)\big) + \epsilon(x) \approx \theta^T Y(x) + \epsilon(x)$$

# Real-Time DNN Approximation

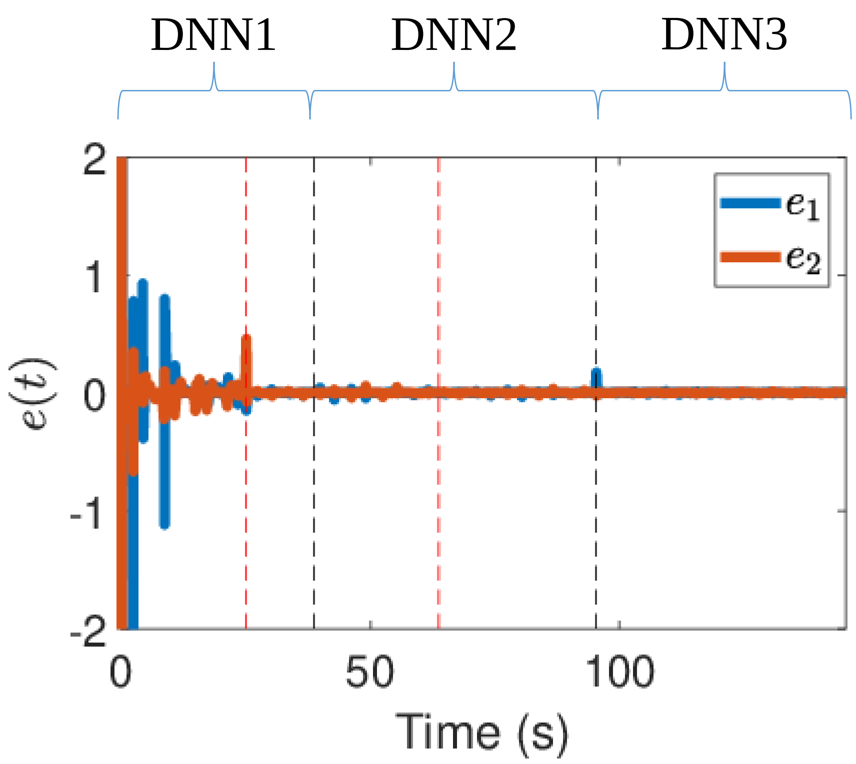


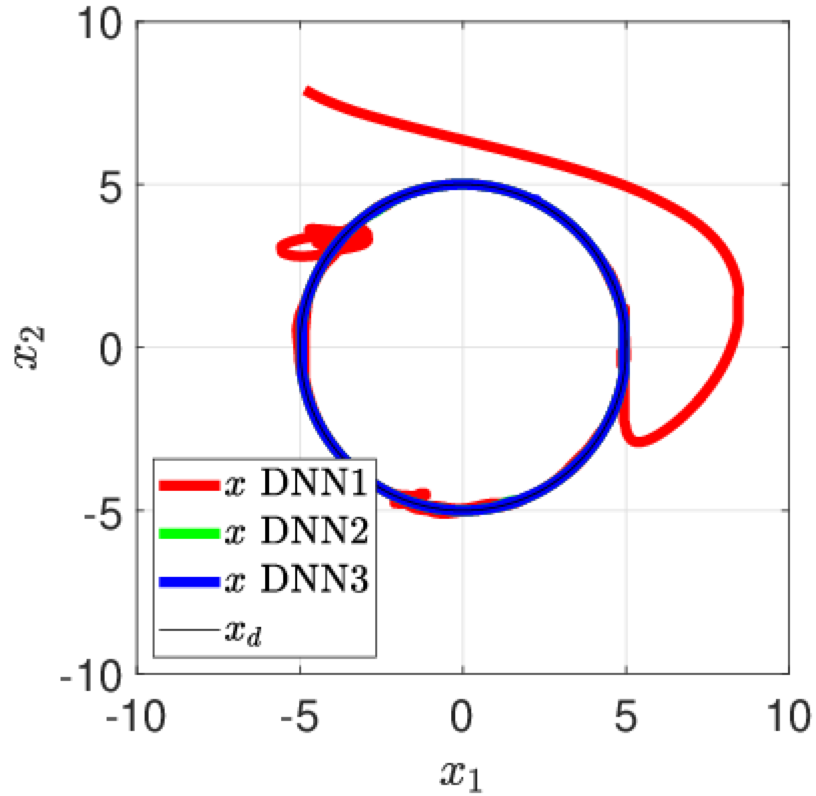
# Simulation

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



Trained on Identical Dynamics



# Simulation

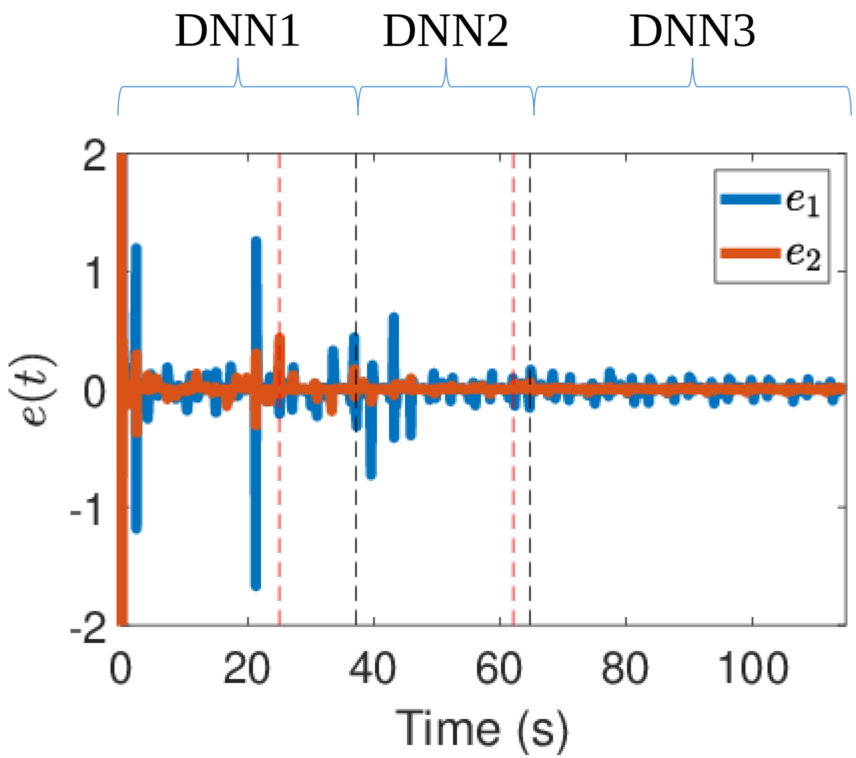
- Adversarial Attacks
  - Recall, $\hat{f}_i\left(x,\hat{\theta}\right) = \hat{\theta}^T\phi\left(\widehat{\Phi}_i(x)\right)$
  - What if an adversary alters $\widehat{\Phi}_i$?
    - Initially Poor Approximation
    - Output-layer Adjustment
    - Robustness to Adversarial Attacks
  - Adaptation to unknown/uncertain environments

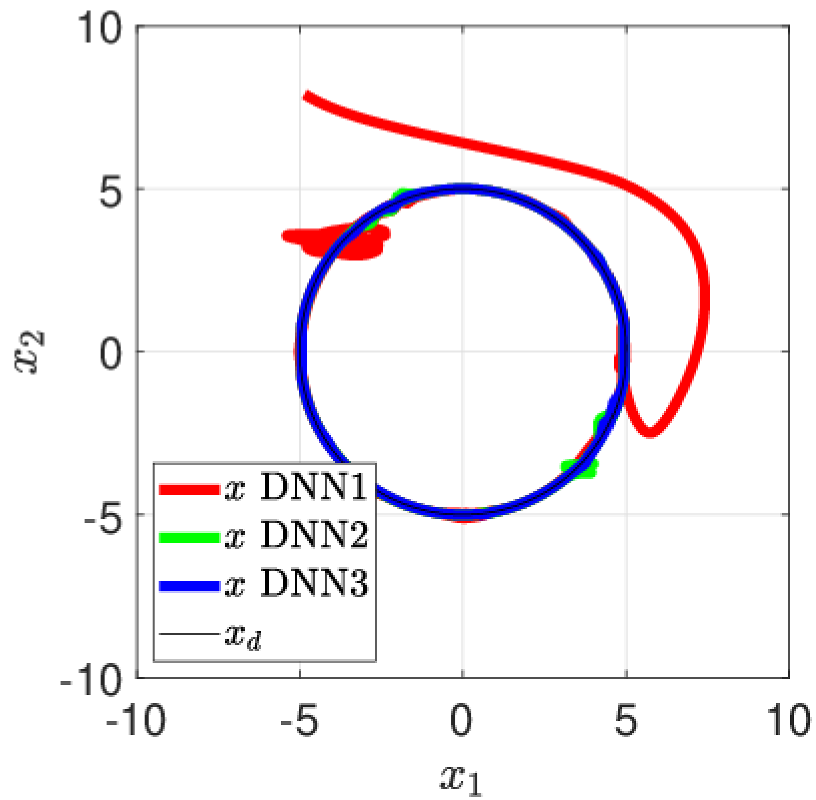
$$\hat{f}_i\left(x,\hat{\theta}\right) = \underbrace{\hat{\theta}^T}_{\text{Accommodates for poor approximation due to } \widehat{\Phi}_i(x)}\phi\left(\widehat{\Phi}_i(x)\right)$$

Accommodates for poor approximation due to $\widehat{\Phi}_i(x)$

# Simulation

Inner-Layer Features Altered By Adversary




UF UNIVERSITY of FLORIDA

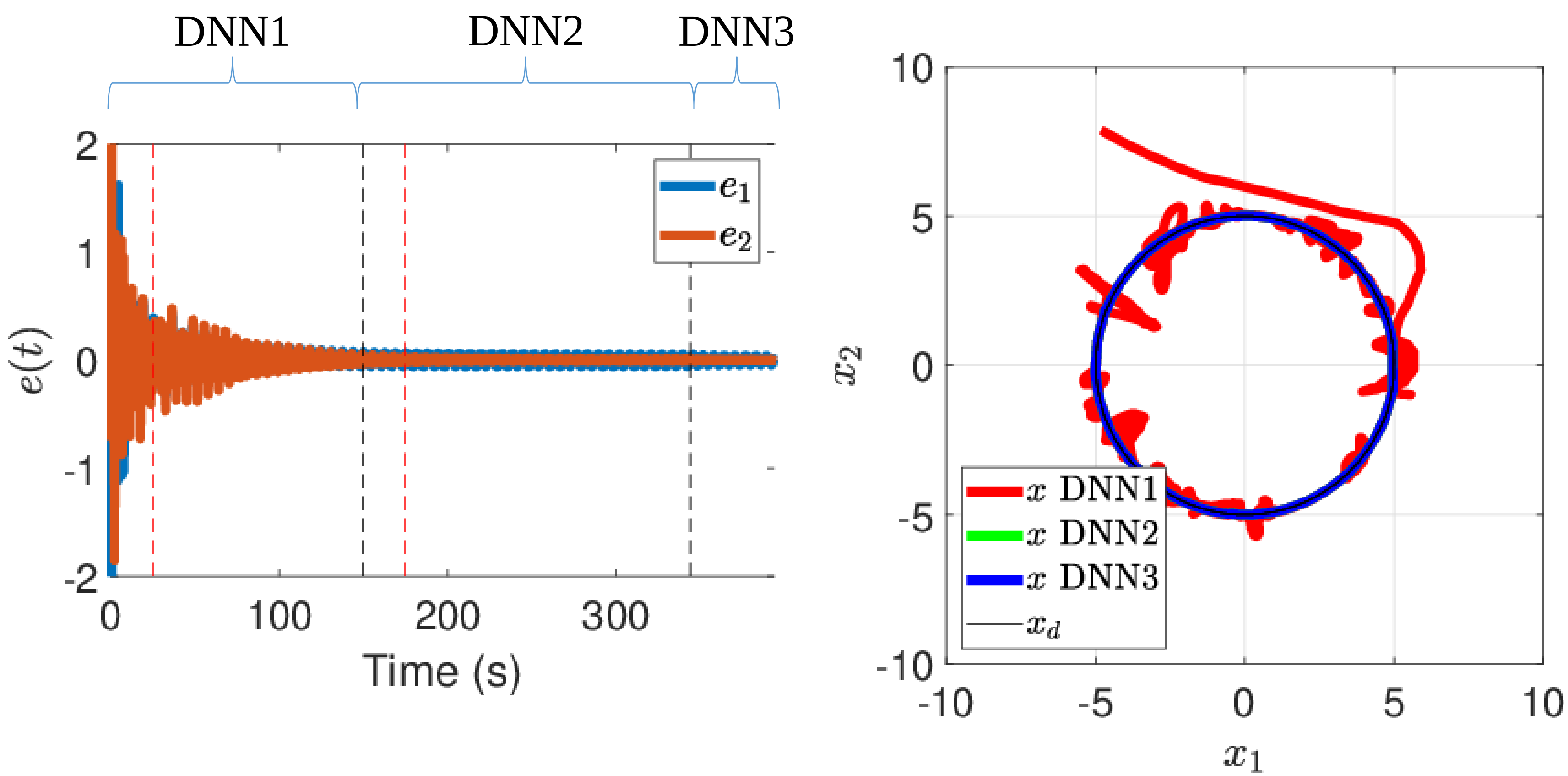
Duke UNIVERSITY

TEXAS The University of Texas at Austin

UC SANTA CRUZ

# Simulation

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



# Conclusion

- Raised Questions
  - Training the output-layer is good
  - Can we update every layer online?
    - Class of update policies to update every layer?
    - Can every layer be updated simultaneously?

Questions?

maxgreene12@ufl.edu

# Real-time Modular Deep Neural Network Control

D. Le, M. Greene, W. Makumi, and W. Dixon, “Real-time Modular Deep Neural Network-Based Adaptive Control of Nonlinear Systems,” IEEE Control Sys. Lett. 2022

AFOSR Center of Excellence Review via Zoom

November 9th , 2021

# Dynamics & Objective

- Nonlinear dynamic system

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

- DNN approximation
  - Using DNN to approximate the uncertain drift dynamics
  - Real-time learning of DNN weights with Lyapunov-based methods

$$f(x) = W^{*T}\sigma\left(\Phi^{*}(x)\right) + \varepsilon(x)$$

$$\Phi^{*}(x) = \left(V_k^{*T}\phi_k \circ \ldots \circ V_1^{*T}\phi_1\right)\left(V_0^{*T}x\right)$$

- Control Objective

Track a user-defined time-varying trajectory $x_d$

$$e = x - x_d$$

# Control Development

- Control input

$$u = g^{+}(x)\left(\dot{x}_d - k_1 e - k_s \mathrm{sgn}(e) - \hat{f}(x)\right)$$

- Output-layer weight adaptation

$$\dot{\hat{W}} = \Gamma_W \sigma\left(\hat{\Phi}(x)\right) e^T$$

- Inner-layer weight adaptation

$$\dot{\hat{V}}_j = p_j(t)\,\nu_j(e,t)\,\mathbf{1}_{\left\{\underline{\hat{V}_j} \leq \left\|\hat{V}_j\right\|_F \leq \overline{\hat{V}_j}\right\}}$$

- Modular adaptive constraints

$$\left\|\nu_j\right\|_F \leq \rho(\|e\|)\,\|e\|$$

# Stability Analysis

- Closed-loop error system

$$\dot{e} = W^{*T}\sigma\left(\Phi^*\left(x\right)\right) + \varepsilon\left(x\right) - k_1 e - k_s \text{sgn}\left(e\right) - \hat{W}^T\sigma\left(\hat{\Phi}\left(x\right)\right)$$

- **Theorem**. The designed control input, output-layer weight adaptation law, and the family of potential inner-layer weight adaptation laws that satisfy the modular adaptive constraints ensure the closed-loop error system yields semi-global asymptotic tracking, provided the following sufficient gain condition is satisfied

$$k_s > 2\overline{\sigma}\overline{W^*} + \overline{\varepsilon} + 2\overline{V^*}\left(k+1\right)\rho\left(\sqrt{\frac{\overline{\alpha}}{\underline{\alpha}}}\left\|z\left(0\right)\right\|\right),$$

where $\underline{\alpha}, \overline{\alpha} \in \mathbb{R}_{\geq 0}$ are known constants.

# Simulations

- System Dynamics

$$f(x) = \left[-x_1 + x_2, -\frac{1}{2}x_1 + \frac{1}{2}x_2\left(1 - \left(\cos\left(2x_1 + 2\right)^2\right)\right)\right]^T$$

$$g(x) = \operatorname{diag}[5, 3]$$

- DNN structure
  - 6 layers: 12, 10, 15, 15, 20 neurons
  - tanh activation function

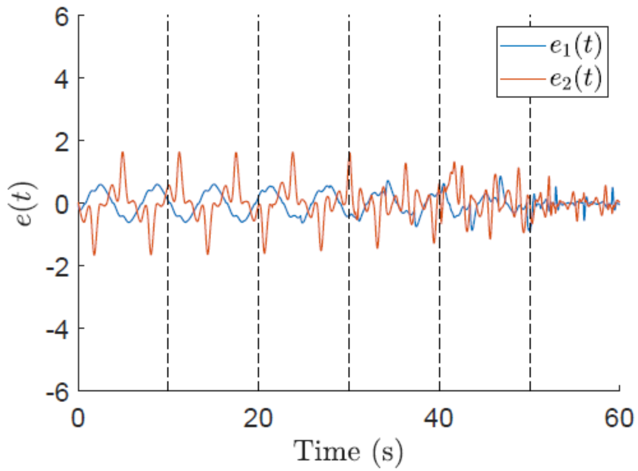
- Inner-layer weight adaptation
  - Switching signal $p_j(t) = \begin{cases} 1, & t \in [10j, 10(j+1)], \\ 0, & \text{else}, \end{cases}$
  - Update law

$$\nu_j = \Gamma_{V_j} e \cdot \operatorname{re}\left(\left(\tanh \circ \hat{\phi}_j^{-1} \circ \hat{V}_j^{+T} \hat{\phi}_{j+1}^{-1} \circ \hat{V}_5^{+T} \hat{\sigma}^{-1}\right)\left(\hat{W}^{+T} \dot{\hat{x}}\right)\right) \hat{V}_j$$

# Simulations

- Parameters

$$x_d = [3\cos(t), 5\sin(t)]^T$$
$$k_s = 0.5,\ k = 7$$
$$\Gamma_W = 10 \cdot \mathbf{1}_{L\times L},\ \Gamma_{V_j} = 1000 \cdot \mathbf{1}_{L_j \times 2}$$



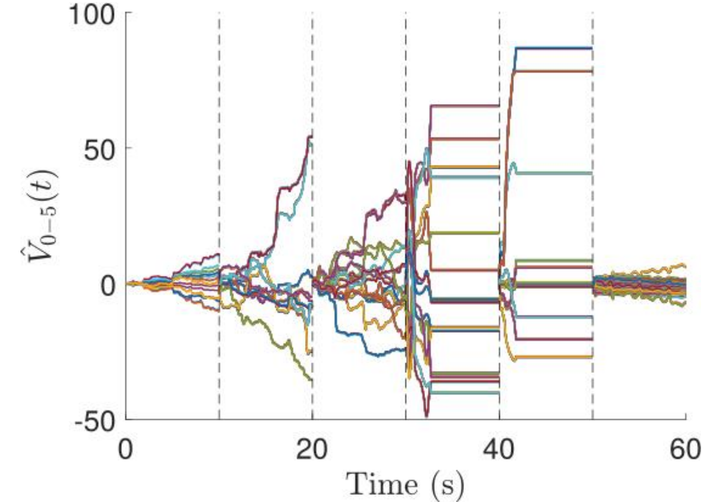




Table 1: RMS and SD Error

| Active $\hat{V}_j$ | RMS Error | SD Error |
|---|---|---|
| $\hat{V}_0$ | 0.735 | 0.145 |
| $\hat{V}_1$ | 0.727 | 0.142 |
| $\hat{V}_2$ | 0.710 | 0.136 |
| $\hat{V}_3$ | 0.640 | 0.116 |
| $\hat{V}_4$ | 0.658 | 0.087 |
| $\hat{V}_5$ | 0.289 | 0.039 |

# Lyapunov-Derived Control and Adaptive Update Laws for Inner and Outer Layer Weights of a Deep Neural Network

O. Patil, D. Le, M. Greene, and W. Dixon, “Lyapunov-Derived Control and Adaptive Update Laws for Inner and Outer Layer Weights of a Deep Neural Network,” submitted to IEEE Control Sys. Lett.

AFOSR Center of Excellence Review @ Zoom

November 9th , 2021

# Control Objective

- Dynamic Model

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

- Tracking Error $e \triangleq x - x_d$

where $x_d(t) \in \Omega$, given a compact set $\Omega$.

- Control Objective : $e(t) \to 0$ as $t \to \infty$

# DNN Architecture

- Fully-connected feedforward DNN

$$\Phi(x_d, V_0, V_1, \ldots, V_k) \triangleq \left(V_k^T \phi_k \circ \ldots \circ V_1^T \phi_1\right)\left(V_0^T x_{da}\right)$$

- Recursive representation

$$\Phi_j \triangleq \begin{cases} V_j^T \phi_j\left(\Phi_{j-1}\right), & j \in \{1, \ldots, k\}, \\ V_0^T x_{da}, & j = 0, \end{cases}$$

- Universal Approximation Property

$$\sup_{x_d \in \Omega} \left\| f(x_d) - \Phi(x_d, V_0^*, V_1^*, \ldots, V_k^*) \right\| \leq \overline{\varepsilon}$$

- Then

$$f(x_d) = \Phi(x_d, V_0^*, V_1^*, \ldots, V_k^*) + \varepsilon(x_d)$$

where $\sup_{x_d \in \Omega} \|\varepsilon(x_d)\| \leq \overline{\varepsilon}$ .

# Control Law Development

- Adaptive Feedforward DNN term

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

where $\hat{V}_0, \ldots, \hat{V}_k$ denote weight estimates.

- Control Law

Robust terms

DNN

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

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

- $$\hat{\Phi}_j \triangleq \Phi_j(x_d, \hat{V}_0, \ldots, \hat{V}_j) \qquad \Phi_j^* \triangleq \Phi_j(x_d, V_0^*, \ldots, V_j^*)$$

$$\hat{\phi}_j \triangleq \phi_j\left(\hat{\Phi}_{j-1}\right) \qquad \phi_j^* \triangleq \phi_j\left(\Phi_{j-1}^*\right)$$

$$\hat{\phi}_j' \triangleq \phi_j'\left(\hat{\Phi}_{j-1}\right) \qquad \widetilde{\Phi}_j \triangleq \Phi_j^* - \hat{\Phi}_j$$

# Adaptation Laws

- Adaptation law for the $j^{th}$ layer

$$\text{vec}(\dot{\hat{V}}_j) \triangleq \text{proj}(\Gamma_j \Lambda_j^T e)$$

$$\Lambda_0 \triangleq \Xi_0 (I_{L_1} \otimes x_{da}^T) \qquad \Lambda_j \triangleq \Xi_j (I_{L_{j+1}} \otimes \hat{\phi}_j^T)$$

$$\Xi_0 \triangleq \overset{\curvearrowleft}{\prod}_{l=1}^{k} \hat{V}_l^T \hat{\phi}_l' \qquad \Xi_j \triangleq \overset{\curvearrowleft}{\prod}_{l=1}^{k} \hat{V}_l^T \hat{\phi}_l'$$

- Projection operator guarantees weight estimates stay in the set

$$\mathcal{B}_j \triangleq \{\theta \in \mathbb{R}^{L_j L_{j+1}} : \|\theta\|_F \leq \overline{V}\}$$

# Stability Analysis

- First-order Taylor series approximation

$$\phi_j^* = \hat{\phi}_j + \hat{\phi}_j' \widetilde{\Phi}_{j-1} + \mathcal{O}^2\left(\widetilde{\Phi}_{j-1}\right)$$

- Output mismatch at the $j^{th}$ layer

Linearly-parameterized | Recursion | Higher-Order Terms

$$\widetilde{\Phi}_j = \left(I_{L_{j+1}} \otimes \hat{\phi}_j^T\right) \text{vec}\left(\widetilde{V}_j\right) + \hat{V}_j^T \hat{\phi}_j' \widetilde{\Phi}_{j-1} + \Delta_j$$

where $\Delta_j \triangleq \widetilde{V}_j^T \hat{\phi}_j' \widetilde{\Phi}_{j-1} + V_j^{*T} \mathcal{O}^2\left(\widetilde{\Phi}_{j-1}\right)$

- DNNs with nonsmooth units are modeled via a switching mechanism involving smooth DNNs.

$$f(x_d) = \Phi_{k,\sigma}(x_d, V_0^*, V_1^*, \ldots, V_k^*) + \varepsilon_\sigma(x_d)$$

# Stability Analysis

- Using recursion

$$\widetilde{\Phi} = \widetilde{\Phi}_{k,\sigma} = \sum_{j=0}^{k} \Lambda_{j,\sigma} \mathrm{vec}(\widetilde{V}_j) + \sum_{j=1}^{k} \Xi_{j,\sigma} \Delta_{j,\sigma}$$

- Tracking Error Dynamics

$$\dot{e} = f(x) - f(x_d) + \sum_{j=0}^{k} \Lambda_{j,\sigma} \mathrm{vec}(\widetilde{V}_j) + \sum_{j=1}^{k} \Xi_{j,\sigma} \Delta_{j,\sigma} + \varepsilon_\sigma(x_d) - \rho(\|e\|)e - k_1 e - k_s \mathrm{sgn}(e)$$

- Weight Estimation Error Dynamics

$$\mathrm{vec}(\dot{\widetilde{V}}_j) = -\mathrm{proj}(\Gamma_j \Lambda_{j,\sigma}^T e)$$

- Theorem 1. The designed controller and adaptation laws ensure global asymptotic tracking error convergence in the sense that

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

provided the following gain condition is satisfied

$$k_s > \bar{\varepsilon} + c$$

# Stability Analysis

- Candidate Common Lyapunov Function

$$\mathcal{V}_L(z) \triangleq \frac{1}{2}e^T e + \frac{1}{2}\sum_{j=0}^{k} \operatorname{vec}(\widetilde{V}_j)^T \Gamma_j^{-1} \operatorname{vec}(\widetilde{V}_j),$$

- Generalized time-derivative

$$\dot{\overline{\mathcal{V}}}_\sigma(z,t) \triangleq \max_{p\in\partial\mathcal{V}_L(z)} \max_{q\in F_\sigma(z,t)} p^T q$$

$$\dot{\overline{\mathcal{V}}}_\sigma(z,t) \overset{a.e.}{\leq} e^T\left(f_e + \sum_{j=1}^{k} \Xi_{j,\sigma}\Delta_{j,\sigma} + \varepsilon_\sigma(x_d)\right) - k_s\left\|e\right\|_1$$
$$+\max\sum_{j=0}^{k}\{e^T\Lambda_{j,\sigma}\operatorname{vec}(\widetilde{V}_j) - \operatorname{vec}(\widetilde{V}_j)^T\Lambda_{j,\sigma}^T e\}$$
$$-\rho(\left\|e\right\|)\left\|e\right\|^2 - k_1\left\|e\right\|^2$$

# Stability Analysis

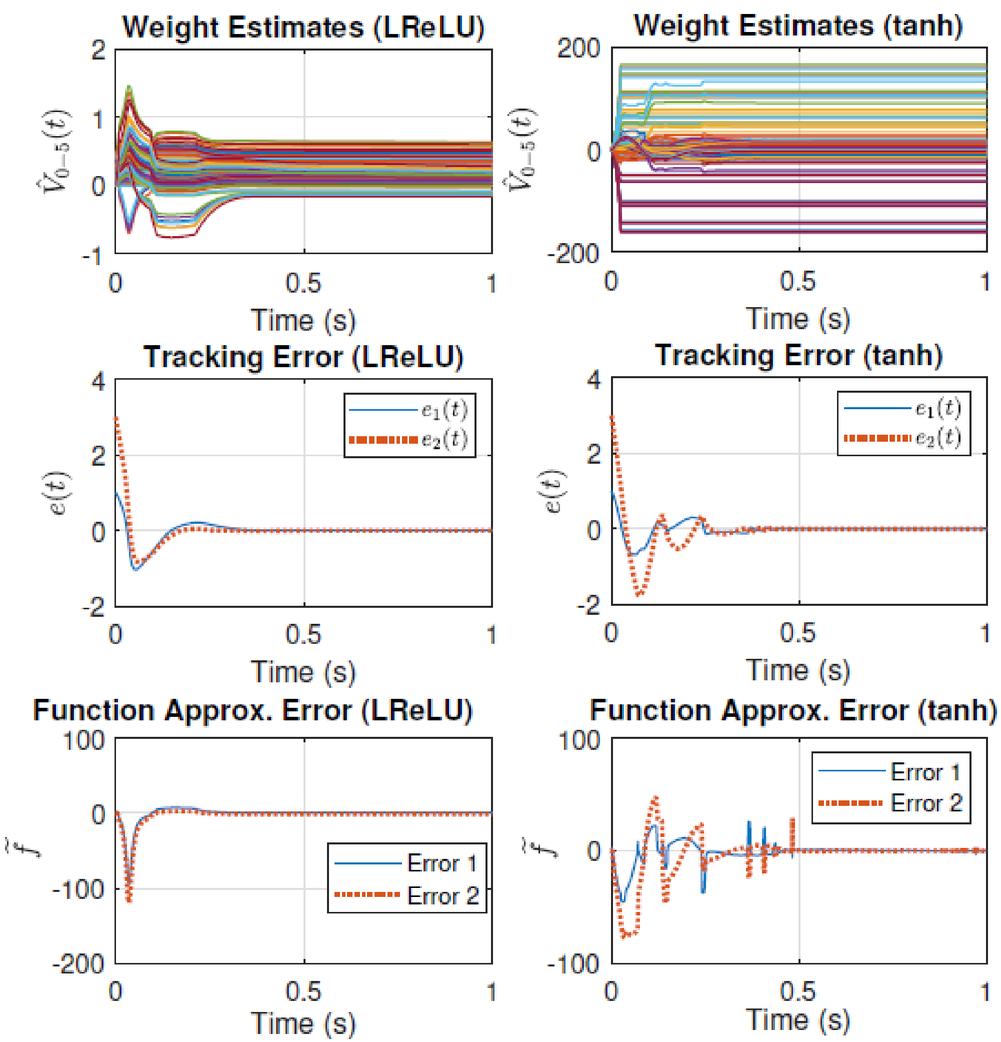
- Further upper-bounding yields

$$\dot{\overline{\mathcal{V}}}_\sigma(z,t) \overset{a.e.}{\leq} -k_1 \left\| e \right\|^2$$

- Using the LaSalle-Yoshizawa theorem for switched systems yields

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

# Simulation Results

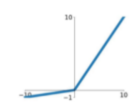



- System

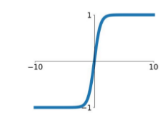
$$f(x) = \begin{bmatrix} x_1 x_2 \tanh(x_2) + \text{sech}^2(x_1) \\ \text{sech}^2(x_1 + x_2) - \text{sech}^2(x_2) \end{bmatrix}$$

- Each DNN has 6 hidden layers with 7 neurons

**Leaky ReLU**
$\max(0.1x, x)$



**tanh**
$\tanh(x)$



| Activation Function | $\|e_{\text{RMS}}\|$ | $\|\tilde{f}_{\text{RMS}}\|$ |
|---|---|---|
| LReLU | 0.3045 | 5.4326 |
| Hyperbolic Tangent | 2.0898 | 42.1838 |

# Conclusions and Future Work

- The first result on Lyapunov-derived adaptation laws for all layers
- Leaky ReLU-based DNN provided seven-fold improvement in tracking and function estimation performance
- Future work will involve extension for residual networks

**Contact**
Daniel Le (ledan50@ufl.edu)
Max Greene (maxgreene12@ufl.edu)
Runhan Sun (runhansun@ufl.edu)
Omkar Patil (patilomkarsudhir@ufl.edu)
Warren Dixon (wdixon@ufl.edu)

**NCR Links**
Website: https://ncr.mae.ufl.edu/
YouTube: https://www.youtube.com/user/NCRatUF/videos

# Thank you!