Imposing a Weight Norm Constraint for Neuro-Adaptive Control

IEEE European Control Conference (ECC) 2025

Myeongseok Ryu[1], Jiyun Kim[2], and Kyunghwan Choi[1]

[1]Mobility Intelligence and Control Laboratory (MIC Lab)
CCS Graduate School of Mobility
Korea Advanced Institute of Science and Technology (KAIST)

[2]AI Graduate School
Gwangju Institute of Science and Technology (GIST)

KAIST

MIC Lab
Mobility Intelligence and Control Laboratory

2025-06-25

Outline

1 **Background and Contributions**
- Introduction to Neuro-Adaptive Control
- Literature Review
- Contributions

2 **Proposed Method**
- Architecture of the Proposed Method
- Problem Formulation
- Adaptation Law Derivation
- Stability Analysis

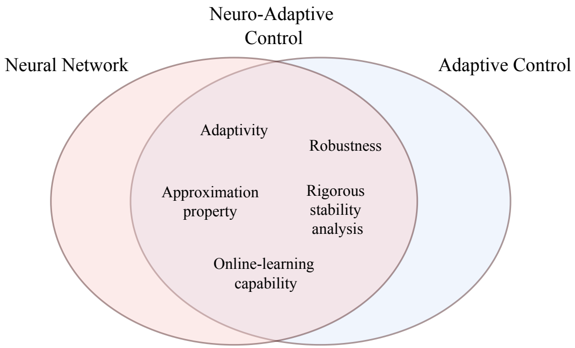
3 **Experimental Validation**
- Simulation Setup
- Simulation Results

4 **Conclusion**
- Conclusion and Future Work

1. **Background and Contributions**
   - Introduction to Neuro-Adaptive Control
   - Literature Review
   - Contributions
2. Proposed Method
3. Experimental Validation
4. Conclusion

Neuro-Adaptive Control

- **Neuro-adaptive control** (NAC) is a control strategy that combines neural networks (NNs) with adaptive control [1].
- Features of both NNs and adaptive control can be found in NAC.



Advantages of Neuro-Adaptive Control

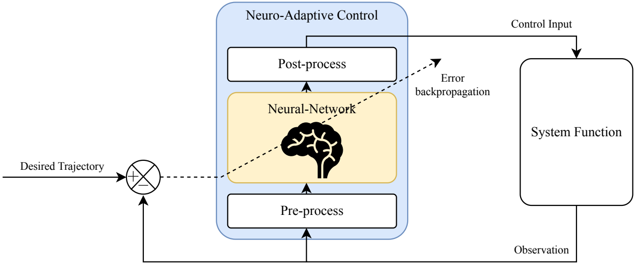


Figure: General framework of neuro-adaptive control (NAC).

Advantages of Neuro-Adaptive Control

- **Adaptability**: NAC adapts NN weights to changing environments and system dynamics.

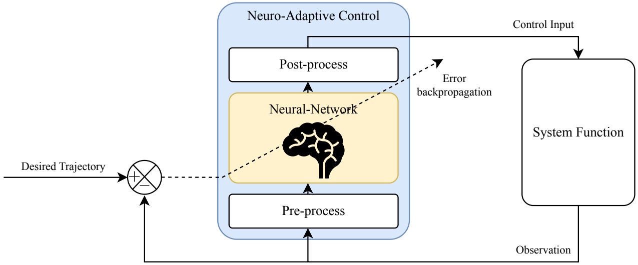


Figure: General framework of neuro-adaptive control (NAC).

Advantages of Neuro-Adaptive Control

- **Adaptability**: NAC adapts NN weights to changing environments and system dynamics.
- **Stability Guarantee**: The closed-loop stability is ensured using Lyapunov stability theory.

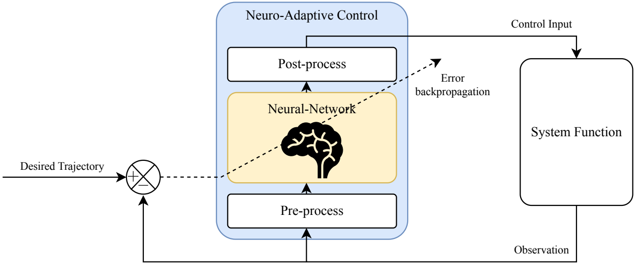


Figure: General framework of neuro-adaptive control (NAC).

Advantages of Neuro-Adaptive Control

- **Adaptability**: NAC adapts NN weights to changing environments and system dynamics.
- **Stability Guarantee**: The closed-loop stability is ensured using Lyapunov stability theory.
- **Online Learning Capability**: NAC adapts in real-time to new data with stability guarantees.

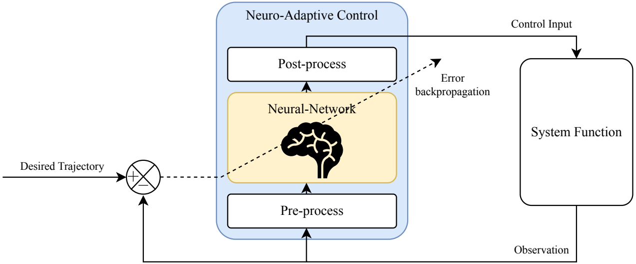


Figure: General framework of neuro-adaptive control (NAC).

Advantages of Neuro-Adaptive Control

- **Adaptability**: NAC adapts NN weights to changing environments and system dynamics.
- **Stability Guarantee**: The closed-loop stability is ensured using Lyapunov stability theory.
- **Online Learning Capability**: NAC adapts in real-time to new data with stability guarantees.
- **Robustness**: NAC handles uncertainties and disturbances effectively with adaptive control techniques.

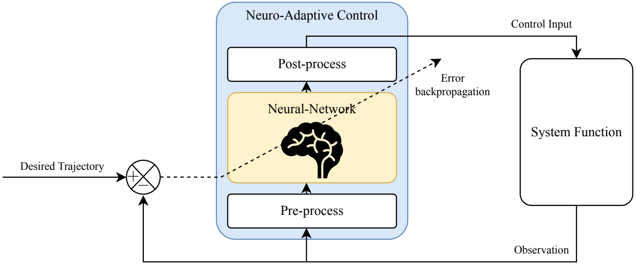


Figure: General framework of neuro-adaptive control (NAC).

Existing Challenges in NAC

Existing Challenges in NAC

1. **Weight Boundedness:**
   - Generally, NN weights are adapted by gradient descent method.
     - Objective function typically consists of the control error.
   - Hence, the NN weights can grow unbounded, leading to instability (also known as *parameter drift*).
   - Unbounded weights can cause the NN to produce large control inputs, which may lead to following challenges.

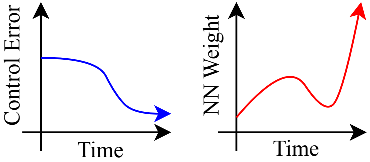


Figure: Divergence of NN weights even with convergent control error.

Existing Challenges in NAC

1. **Weight Boundedness:**
   - Generally, NN weights are adapted by gradient descent method.
     - Objective function typically consists of the control error.
   - Hence, the NN weights can grow unbounded, leading to instability (also known as *parameter drift*).
   - Unbounded weights can cause the NN to produce large control inputs, which may lead to following challenges.
2. **Control Saturation** *(unpredictable amplitude of NN outputs)*:
   - Typical issue of control problem in physical systems.
   - The NN outputs are unpredictable and not interpretable.
   - These features—unbounded NN weights and unpredictable amplitudes—can lead to input saturation.

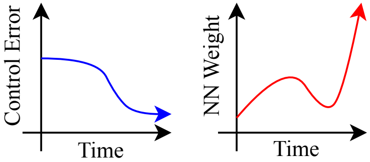


Figure: Divergence of NN weights even with convergent control error.

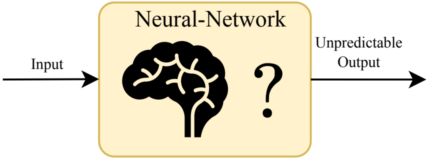


Figure: Unpredictable amplitude of NN outputs.

1. **Projection Operator** *for weight boundednss*
2. **σ-modification, and ϵ-modification** *for weight boundednss*
3. **Additional Control Inputs** *for control saturation*

1. **Projection Operator** *for weight boundednss*
   - Projects the NN weights onto a convex set.
   - Ensures that the weights remain within a predefined bound.

$$\widehat{\boldsymbol{\theta}} \leftarrow \mathrm{Proj}_{\overline{\theta}}(\widehat{\boldsymbol{\theta}}) \tag{1}$$

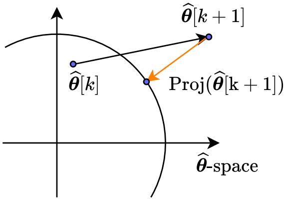


Figure: Projection of NN weights on a convex set.

2. **σ-modification, and ϵ-modification** *for weight boundednss*

3. **Additional Control Inputs** *for control saturation*

1. **Projection Operator** *for weight boundednss*
   - Projects the NN weights onto a convex set.
   - Ensures that the weights remain within a predefined bound.

$$\widehat{\boldsymbol{\theta}} \leftarrow \mathrm{Proj}_{\overline{\theta}}(\widehat{\boldsymbol{\theta}}) \tag{1}$$

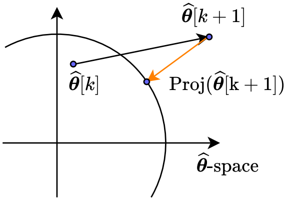


Figure: Projection of NN weights on a convex set.

2. **σ-modification, and ϵ-modification** *for weight boundednss*
   - Add a stabilizing term (e.g., $-\lambda\widehat{\boldsymbol{\theta}}$) to adaptation law.
   - Construct a invariant set of the NN weights.

$$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} \leftarrow \frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} - \lambda\widehat{\boldsymbol{\theta}} \tag{2}$$

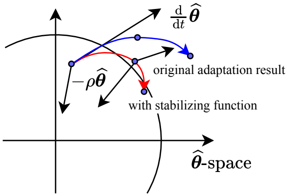


Figure: Adaptation result with stabilizing function (e.g., σ-modification).

3. **Additional Control Inputs** *for control saturation*

1. **Projection Operator** *for weight boundednss*
   - Projects the NN weights onto a convex set.
   - Ensures that the weights remain within a predefined bound.

$$\widehat{\boldsymbol{\theta}} \leftarrow \mathrm{Proj}_{\bar{\theta}}(\widehat{\boldsymbol{\theta}}) \tag{1}$$

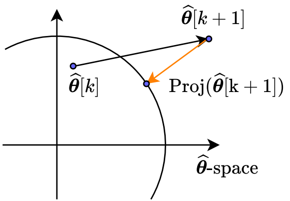


Figure: Projection of NN weights on a convex set.

2. **σ-modification, and ϵ-modification** *for weight boundednss*
   - Add a stabilizing term (e.g., $-\lambda\widehat{\boldsymbol{\theta}}$) to adaptation law.
   - Construct a invariant set of the NN weights.

$$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} \leftarrow \frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} - \lambda\widehat{\boldsymbol{\theta}} \tag{2}$$

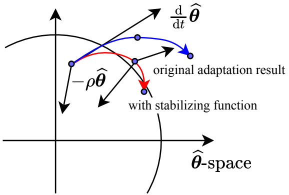


Figure: Adaptation result with stabilizing function (e.g., σ-modification).

3. **Additional Control Inputs** *for control saturation*
   - Conventional controllers are used to address control input saturation.
     - Barrier Lyapunov function or auxiliary system-based control inputs.
   - In general, nominal models are required.

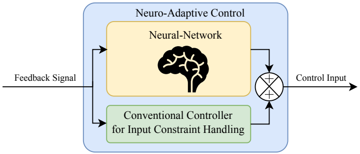


Figure: Control input saturation handling with additional control inputs.

Limitation 1: Lack of Optimality

Limitation 2: Disruption of Learning Process by Additional Control Inputs

Limitation 1: Lack of Optimality

- The existing methods do not guarantee the optimality of the control input.

Limitation 2: Disruption of Learning Process by Additional Control Inputs

Limitation 1: Lack of Optimality

- The existing methods do not guarantee the optimality of the control input.
- Projection operator:
  - The projection operator simply projects the NN weights onto a convex set, regardless of the imposed constraints (e.g., weight boundedness or input saturation).
  - Moreover, if the convex set is conservatively defined, the weights may be limited to a suboptimal region.

Limitation 2: Disruption of Learning Process by Additional Control Inputs

Limitation 1: Lack of Optimality

- The existing methods do not guarantee the optimality of the control input.
- Projection operator:
    - The projection operator simply projects the NN weights onto a convex set, regardless of the imposed constraints (e.g., weight boundedness or input saturation).
    - Moreover, if the convex set is conservatively defined, the weights may be limited to a suboptimal region.
- σ- and ϵ-modification:
    - The stabilizing term biases the NN weights towards the origin.
    - Therefore, the weights converge toward a suboptimal point.

Limitation 2: Disruption of Learning Process by Additional Control Inputs

Limitation 1: Lack of Optimality

- The existing methods do not guarantee the optimality of the control input.
- Projection operator:
  - The projection operator simply projects the NN weights onto a convex set, regardless of the imposed constraints (e.g., weight boundedness or input saturation).
  - Moreover, if the convex set is conservatively defined, the weights may be limited to a suboptimal region.
- σ- and ϵ-modification:
  - The stabilizing term biases the NN weights towards the origin.
  - Therefore, the weights converge toward a suboptimal point.

Limitation 2: Disruption of Learning Process by Additional Control Inputs

- Feedback tracking error for learning is disrupted by additional control inputs.
  - The feedback error does not reflect the error induced by the NN, directly.
  - The additional control inputs may exceeds the input saturation limits , already.

Contributions

Contribution 1: Unified Optimization Framework

Contribution 2: Online Learning Capability (Stability Guarantees)

Contribution 3: Weight and Control Input Constraint Handling

Contributions

Contribution 1: Unified Optimization Framework

- Trajectory tracking and constraint handling are formulated as a unified constrained optimization problem.
- The conventional controllers does not required.
  - Nominal model knowledge is not required for the conventional controllers.

Contribution 2: Online Learning Capability (Stability Guarantees)

Contribution 3: Weight and Control Input Constraint Handling

Contributions

Contribution 1: Unified Optimization Framework

- Trajectory tracking and constraint handling are formulated as a unified constrained optimization problem.
- The conventional controllers does not required.
  - Nominal model knowledge is not required for the conventional controllers.

Contribution 2: Online Learning Capability (Stability Guarantees)

- Stability are rigorously proven using Lyapunov stability theory.
- Hence, online learningwith no prior system knowledge is possible.

Contribution 3: Weight and Control Input Constraint Handling

Contribution 1: Unified Optimization Framework

- Trajectory tracking and constraint handling are formulated as a unified constrained optimization problem.
- The conventional controllers does not required.
  - Nominal model knowledge is not required for the conventional controllers.

Contribution 2: Online Learning Capability (Stability Guarantees)

- Stability are rigorously proven using Lyapunov stability theory.
- Hence, online learningwith no prior system knowledge is possible.

Contribution 3: Weight and Control Input Constraint Handling

- Weight and control input constraints are explicitly considered in the optimization problem.
- Any combination of convex input constraints can be handled.

Outline

1 Background and Contributions

2 **Proposed Method**

- Architecture of the Proposed Method
- Problem Formulation
- Adaptation Law Derivation
- Stability Analysis

3 Experimental Validation

4 Conclusion

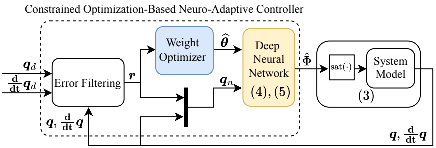


Figure: Architecture of the proposed method.

Notations: $\boldsymbol{q} \in \mathbb{R}^n$: Joint position, $\boldsymbol{M}$: Inertia matrix, $\boldsymbol{C}$: Coriolis matrix, $\boldsymbol{G}$: Gravity vector, $\boldsymbol{\tau}$: Control input, $\boldsymbol{\tau}_d$: Disturbance.

Target Two-link Robotic Manipulator System:

- Control input saturation function sat(·).
- Desired trajectory $\boldsymbol{q}_d$ is given.

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{V}_m\dot{\boldsymbol{q}} + \boldsymbol{F} + \boldsymbol{G} + \boldsymbol{\tau}_d = \text{sat}(\boldsymbol{\tau}) \qquad (3)$$

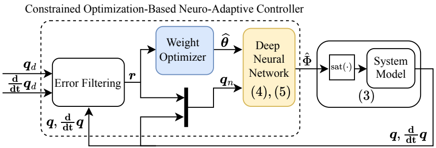


Figure: Architecture of the proposed method.

Notations: $\boldsymbol{q} \in \mathbb{R}^n$: Joint position, $\boldsymbol{M}$: Inertia matrix, $\boldsymbol{C}$: Coriolis matrix, $\boldsymbol{G}$: Gravity vector, $\boldsymbol{\tau}$: Control input, $\boldsymbol{\tau}_d$: Disturbance.

Target Two-link Robotic Manipulator System:

- Control input saturation function sat$(\cdot)$.
- Desired trajectory $\boldsymbol{q}_d$ is given.

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{V}_m\dot{\boldsymbol{q}} + \boldsymbol{F} + \boldsymbol{G} + \boldsymbol{\tau}_d = \mathrm{sat}(\boldsymbol{\tau}) \tag{3}$$

Control Input:

- NN's output $\boldsymbol{\Phi}$ is used as the control input.
- Consists of the estimated NN weights $\widehat{\boldsymbol{\theta}}$.

$$\boldsymbol{\tau} := \boldsymbol{\Phi}(\boldsymbol{q}_n; \widehat{\boldsymbol{\theta}}) \tag{4}$$

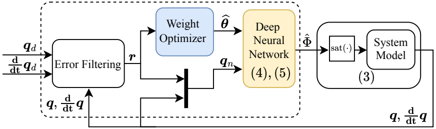


Figure: Architecture of the proposed method.

Notations: $\boldsymbol{q} \in \mathbb{R}^n$: Joint position, $\boldsymbol{M}$: Inertia matrix, $\boldsymbol{C}$: Coriolis matrix, $\boldsymbol{G}$: Gravity vector, $\boldsymbol{\tau}$: Control input, $\boldsymbol{\tau}_d$: Disturbance.

Target Two-link Robotic Manipulator System:

- Control input saturation function $\mathrm{sat}(\cdot)$.
- Desired trajectory $\boldsymbol{q}_d$ is given.

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{V}_m\dot{\boldsymbol{q}} + \boldsymbol{F} + \boldsymbol{G} + \boldsymbol{\tau}_d = \mathrm{sat}(\boldsymbol{\tau}) \quad (3)$$

Control Input:

- NN's output $\boldsymbol{\Phi}$ is used as the control input.
- Consists of the estimated NN weights $\widehat{\boldsymbol{\theta}}$.

$$\boldsymbol{\tau} := \boldsymbol{\Phi}(\boldsymbol{q}_n; \widehat{\boldsymbol{\theta}}) \quad (4)$$

Deep Neural Network (DNN):

- k layers with weights $\widehat{\boldsymbol{\theta}}_i := \mathrm{vec}(\widehat{\boldsymbol{W}}_i)$.
- Activation function: $\phi(\cdot) := tanh(\cdot)$.

$$\boldsymbol{\Phi}(\boldsymbol{q}_n; \widehat{\boldsymbol{\theta}}) := \begin{cases} \widehat{\boldsymbol{W}}_i^\top \phi_i(\widehat{\boldsymbol{\Phi}}_{i-1}), & i \in \{1, \ldots, k\}, \\ \widehat{\boldsymbol{W}}_0^\top \boldsymbol{q}_n, & i = 0, \end{cases} \quad (5)$$

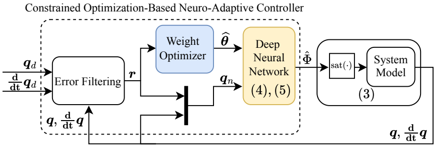


Figure: Architecture of the proposed method.

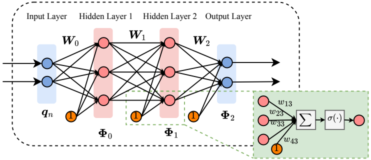


Figure: Architecture of the DNN.

Notations: $\boldsymbol{q} \in \mathbb{R}^n$: Joint position, $\boldsymbol{M}$: Inertia matrix, $\boldsymbol{C}$: Coriolis matrix, $\boldsymbol{G}$: Gravity vector, $\boldsymbol{\tau}$: Control input, $\boldsymbol{\tau}_d$: Disturbance.

Optimization Problem Statement:

- Find NN weights $\widehat{\boldsymbol{\theta}}$,
- That minimize objective function $J(\cdot)$,

$$J(\boldsymbol{r};\widehat{\boldsymbol{\theta}}) := \frac{1}{2}\boldsymbol{r}^\top\boldsymbol{r}. \tag{6}$$

  - where $\boldsymbol{r} := \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{e} + \Lambda\boldsymbol{e}$ is filtered tracking error,
- while satisfying the following constraints:
  - Boundedness of the NN weights $\widehat{\boldsymbol{\theta}}$.
  - Saturation of the control input $\boldsymbol{\tau}$.

Notations: $\Lambda \in \mathbb{R}^{n\times n}_{>\mathbf{0}}$: filtering matrix

MIC Lab Mobility Intelligence and Control Laboratory

Optimization Problem Statement:

- Find NN weights $\widehat{\boldsymbol{\theta}}$,
- That minimize objective function $J(\cdot)$,

$$J(\boldsymbol{r}; \widehat{\boldsymbol{\theta}}) := \frac{1}{2} \boldsymbol{r}^\top \boldsymbol{r}. \tag{6}$$

  - where $\boldsymbol{r} := \frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{e} + \Lambda \boldsymbol{e}$ is filtered tracking error,
- while satisfying the following constraints:
  - Boundedness of the NN weights $\widehat{\boldsymbol{\theta}}$.
  - Saturation of the control input $\boldsymbol{\tau}$.

Considered Constraints

- **Weight Boundedness for Each Layer**:

$$c_{\theta_i}(\widehat{\boldsymbol{\theta}}) := \|\widehat{\boldsymbol{\theta}}_i\|^2 - \overline{\theta_i}^2 \leq 0, \forall i \in \{0, \dots, k\} \tag{7}$$

- **Convex control Input Saturation**:
  - Input bound constraint *for each control input*:

$$c_{\overline{\tau}_i}(\widehat{\boldsymbol{\theta}}) := \tau_i - \overline{\tau_i} \leq 0, \quad c_{\underline{\tau}_i}(\widehat{\boldsymbol{\theta}}) := \underline{\tau_i} - \tau_i \leq 0 \tag{8}$$

  - Input norm constraint:

$$c_{\boldsymbol{\tau}}(\widehat{\boldsymbol{\theta}}) := \|\boldsymbol{\tau}\|^2 - \overline{\tau}^2 \leq 0 \tag{9}$$

Notations: $\Lambda \in \mathbb{R}^{n \times n}_{>\mathbf{0}}$: filtering matrix

Original Optimization Problem

- Constrained optimization problem to minimize the tracking error.
- Inequality constraints $c_j(\widehat{\boldsymbol{\theta}}) \leq 0$ for $j \in \mathcal{I}$.

$$
\begin{aligned}
&\min_{\widehat{\boldsymbol{\theta}}} J(\boldsymbol{r}; \widehat{\boldsymbol{\theta}}) \\
&\text{s.t. } c_j(\widehat{\boldsymbol{\theta}}) \leq 0, \forall j \in \mathcal{I}
\end{aligned} \tag{10}
$$

Original Optimization Problem

- Constrained optimization problem to minimize the tracking error.
- Inequality constraints $c_j(\widehat{\boldsymbol{\theta}}) \leq 0$ for $j \in \mathcal{I}$.

$$\begin{aligned} &\min_{\widehat{\boldsymbol{\theta}}} J(\boldsymbol{r}; \widehat{\boldsymbol{\theta}}) \\ &\text{s.t. } c_j(\widehat{\boldsymbol{\theta}}) \leq 0, \forall j \in \mathcal{I} \end{aligned} \tag{10}$$

Define Lagrangian Function

$$L(\boldsymbol{r}, \widehat{\boldsymbol{\theta}}, [\lambda_j]_{j\in\mathcal{I}}) := J(\boldsymbol{r}; \widehat{\boldsymbol{\theta}}) + \sum_{j\in\mathcal{I}} \lambda_j c_j(\widehat{\boldsymbol{\theta}}) \tag{11}$$

Original Optimization Problem

- Constrained optimization problem to minimize the tracking error.
- Inequality constraints $c_j(\widehat{\boldsymbol{\theta}}) \leq 0$ for $j \in \mathcal{I}$.

$$\begin{aligned} &\min_{\widehat{\boldsymbol{\theta}}} J(\boldsymbol{r}; \widehat{\boldsymbol{\theta}}) \\ &\text{s.t. } c_j(\widehat{\boldsymbol{\theta}}) \leq 0, \forall j \in \mathcal{I} \end{aligned} \tag{10}$$

Define Lagrangian Function

$$L(\boldsymbol{r}, \widehat{\boldsymbol{\theta}}, [\lambda_j]_{j\in\mathcal{I}}) := J(\boldsymbol{r}; \widehat{\boldsymbol{\theta}}) + \sum_{j\in\mathcal{I}} \lambda_j c_j(\widehat{\boldsymbol{\theta}}) \tag{11}$$

Dual Problem

- The dual problem is to minimize the Lagrangian function with respect to the NN weights $\widehat{\boldsymbol{\theta}}$, while maximizing with respect to the Lagrange multipliers λ_j.
- The Lagrange multipliers λ_j are non-negative, i.e., $\lambda_j \geq 0$.

$$\min_{\widehat{\boldsymbol{\theta}}} \max_{[\lambda_j]_{j\in\mathcal{I}}} L(\boldsymbol{r}, \widehat{\boldsymbol{\theta}}, [\lambda_j]_{j\in\mathcal{I}}) \tag{12}$$

To solve the dual problem,

$$\min_{\widehat{\boldsymbol{\theta}}} \max_{[\lambda_j]_{j\in\mathcal{I}}} L(\boldsymbol{r}, \widehat{\boldsymbol{\theta}}, [\lambda_j]_{j\in\mathcal{I}}), \tag{13}$$

the first-order gradient descent/ascent method is used to derive the adaptation law.

Adaptation Law

α: adaptation gain (learning rate), β_j: update rate of the Lagrange multipliers.

To solve the dual problem,

$$\min_{\widehat{\boldsymbol{\theta}}} \max_{[\lambda_j]_{j\in\mathcal{I}}} L(\boldsymbol{r}, \widehat{\boldsymbol{\theta}}, [\lambda_j]_{j\in\mathcal{I}}), \tag{13}$$

the first-order gradient descent/ascent method is used to derive the adaptation law.

Adaptation Law

Gradient Descent Method for weights $\widehat{\boldsymbol{\theta}}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha \frac{\partial L}{\partial \widehat{\boldsymbol{\theta}}} = -\alpha \left(\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}} + \sum_{j\in\mathcal{I}} \lambda_j \frac{\partial c_j}{\partial \widehat{\boldsymbol{\theta}}} \right), \tag{14}$$

α: adaptation gain (learning rate), β_j: update rate of the Lagrange multipliers.

To solve the dual problem,

$$\min_{\widehat{\boldsymbol{\theta}}} \max_{[\lambda_j]_{j\in\mathcal{I}}} L(\boldsymbol{r}, \widehat{\boldsymbol{\theta}}, [\lambda_j]_{j\in\mathcal{I}}), \tag{13}$$

the first-order gradient descent/ascent method is used to derive the adaptation law.

Adaptation Law

Gradient Descent Method for weights $\widehat{\boldsymbol{\theta}}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha \frac{\partial L}{\partial \widehat{\boldsymbol{\theta}}} = -\alpha\left(\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}} + \sum_{j\in\mathcal{I}} \lambda_j \frac{\partial c_j}{\partial \widehat{\boldsymbol{\theta}}}\right), \tag{14}$$

Gradient Ascent Method for Lagrange multipliers $\lambda_j, \forall j \in \mathcal{I}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\lambda_j = \beta_j \frac{\partial L}{\partial \lambda_j} = \beta_j c_j, \tag{15}$$

α: adaptation gain (learning rate), β_j: update rate of the Lagrange multipliers.

MIC Lab Mobility Intelligence and Control Laboratory

To solve the dual problem,

$$\min_{\widehat{\boldsymbol{\theta}}} \max_{[\lambda_j]_{j\in\mathcal{I}}} L(\boldsymbol{r}, \widehat{\boldsymbol{\theta}}, [\lambda_j]_{j\in\mathcal{I}}), \tag{13}$$

the first-order gradient descent/ascent method is used to derive the adaptation law.

Adaptation Law

Gradient Descent Method for weights $\widehat{\boldsymbol{\theta}}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha \frac{\partial L}{\partial \widehat{\boldsymbol{\theta}}} = -\alpha \left(\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}} + \sum_{j\in\mathcal{I}} \lambda_j \frac{\partial c_j}{\partial \widehat{\boldsymbol{\theta}}} \right), \tag{14}$$

Gradient Ascent Method for Lagrange multipliers $\lambda_j, \forall j \in \mathcal{I}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\lambda_j = \beta_j \frac{\partial L}{\partial \lambda_j} = \beta_j c_j, \tag{15}$$

For non-negativity of the Lagrange multipliers,

$$\lambda_j \leftarrow \max(\lambda_j, 0). \tag{16}$$

α: adaptation gain (learning rate), β_j: update rate of the Lagrange multipliers.

Theorem 1 [2]

For the dynamical system described in (3), the neuro-adaptive controller in (4) with the weight adaptation laws in (14), (15) and (16) ensure the boundedness of the filtered error $\boldsymbol{r}$ and the weight estimate $\widehat{\boldsymbol{\theta}}$, under the control input constraints satisfying Assumption 1 and 2. This holds under the weight norm constraint (7).

Assumption 1 (Convex Input Constraint)

The constraint functions $c_j(\widehat{\boldsymbol{\theta}}), \forall j \in \mathcal{I}$, are convex in the τ-space and satisfy $c_j(\widehat{\boldsymbol{\theta}}) \leq 0$ and $c_j(\boldsymbol{\theta}^*) \leq 0$.

Assumption 2, Linear Independence Constraint Qualification (LICQ)

The selected constraints satisfy the Linear Independence Constraint Qualification (LICQ) [3, Chap. 12 Def. 12.1].

Proof of Theorem 1 is **omitted** due to space limitations. The detailed proof can be found in [2].

1 Background and Contributions

2 Proposed Method

3 Experimental Validation
- Simulation Setup
- Simulation Results

4 Conclusion

Target System:

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{V}_m\dot{\boldsymbol{q}} + \boldsymbol{F} + \boldsymbol{G} + \boldsymbol{\tau}_d = \boldsymbol{\tau}$$

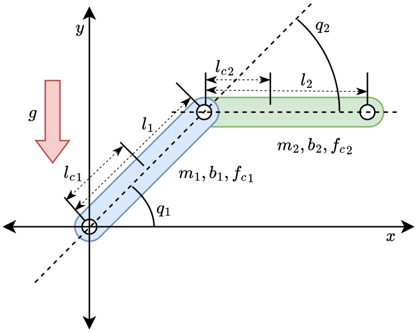


Figure: Two-link robotic manipulator model.

Desired Trajectory:

$$\boldsymbol{q}_d = \begin{pmatrix} q_{d1} \\ q_{d2} \end{pmatrix} = \begin{pmatrix} +\cos(\frac{\pi}{2}t) + 1 \\ -\cos(\frac{\pi}{2}t) - 1 \end{pmatrix}. \quad (17)$$

System Model Parameters:

Table: System model parameters.

Symbol	Description	Link 1	Link 2
m_p	Mass	23.902 kg	3.88 kg
l_p	Length	0.45 m	0.45 m
l_{cp}	COM	0.091 m	0.048 m
b_p	Viscous coef.	2.288 Nms	0.172 Nms
f_{cp}	Friction coef.	7.17 Nm	1.734 Nm

- Only weight norm constraint is considered in this presentation (for **ECC**).
- Input saturation constraints are not considered.
- For simplicity, single hidden layer NNis used.

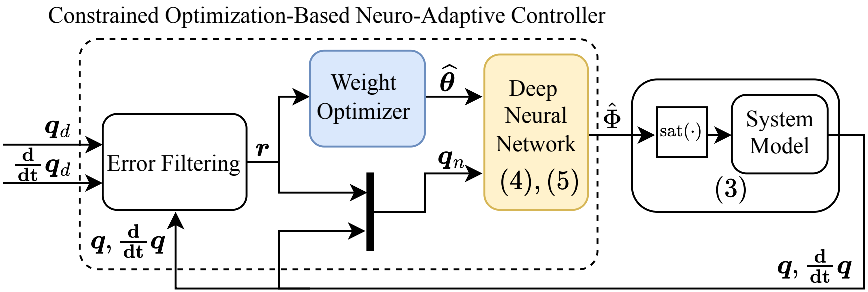


Figure: Architecture of the controllers.

Simulation Setup — Controllers for Comparative Study

- NAC-CO denotes the proposed controller based on constrained optimization .
- For NAC-L2 and NAC-eMod, the stabilizing terms $-\lambda\widehat{\boldsymbol{\theta}}$ and $\rho\|\boldsymbol{r}\|\widehat{\boldsymbol{\theta}}$ ensures the weights boundedness, respectively.

Name	Description	Adaptation Law
NAC-L2	NAC with L_2-regularization (equal to σ-modification) (λ stabilizes $\widehat{\boldsymbol{\theta}}$ towards origin)	$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha\left(\frac{\partial J}{\partial\widehat{\boldsymbol{\theta}}} + \lambda\widehat{\boldsymbol{\theta}}\right)$
NAC-eMod	NAC with ϵ-modification (ρ stabilizes proportionally to filtered error $\boldsymbol{r}$)	$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha\left(\frac{\partial J}{\partial\widehat{\boldsymbol{\theta}}} + \rho\|\tilde{\boldsymbol{r}}\|\widehat{\boldsymbol{\theta}}\right)$
NAC-CO (proposed)	Constrained Optimization-based NAC (β_j determines λ_j adaptation speed)	$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha\left(\frac{\partial J}{\partial\widehat{\boldsymbol{\theta}}} + \sum_{j\in\mathcal{I}}\lambda_j\frac{\partial c_j}{\partial\widehat{\boldsymbol{\theta}}}\right)$ $\frac{\mathrm{d}}{\mathrm{d}t}\lambda_j = \beta_j c_j$, and $\lambda_j \leftarrow \max(\lambda_j, 0)$

Simulation Setup Controllers for Comparative Study

- NAC-CO denotes the proposed controller based on constrained optimization .
- For NAC-L2 and NAC-eMod, the stabilizing terms $-\lambda\widehat{\boldsymbol{\theta}}$ and $\rho\|\boldsymbol{r}\|\widehat{\boldsymbol{\theta}}$ ensures the weights boundedness, respectively.

Name	Description	Adaptation Law
NAC-L2	NAC with L_2-regularization **(equal to σ-modification)** (λ stabilizes $\widehat{\boldsymbol{\theta}}$ towards origin)	$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha\left(\frac{\partial J}{\partial\widehat{\boldsymbol{\theta}}} + \lambda\widehat{\boldsymbol{\theta}}\right)$
NAC-eMod	NAC with ϵ-modification (ρ stabilizes proportionally to filtered error $\boldsymbol{r}$)	$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha\left(\frac{\partial J}{\partial\widehat{\boldsymbol{\theta}}} + \rho\|\tilde{\boldsymbol{r}}\|\widehat{\boldsymbol{\theta}}\right)$
NAC-CO (proposed)	Constrained Optimization-based NAC (β_j determines λ_j adaptation speed)	$\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha\left(\frac{\partial J}{\partial\widehat{\boldsymbol{\theta}}} + \sum_{j\in\mathcal{I}}\lambda_j\frac{\partial c_j}{\partial\widehat{\boldsymbol{\theta}}}\right)$ $\frac{\mathrm{d}}{\mathrm{d}t}\lambda_j = \beta_j c_j$, and $\lambda_j \leftarrow \max(\lambda_j, 0)$

Simulation Objective

By varying the parameters , i.e., β_j, λ, and ρ, the parameter dependencies will be investigated.

Parameter Dependencies Investigation:

- The parameters ranged from 0.001 to 1 across 10 samples.

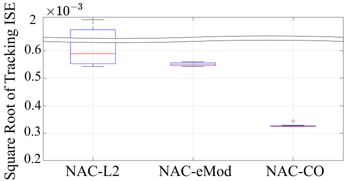


Figure: Box-and-whisker plots of the tracking error ISE.

	NAC-L2	**NAC-eMod**	**NAC-CO** (proposed)
Maximum	11.1753×10^{-3}	0.5603×10^{-3}	0.3439×10^{-3}
Median	0.5898×10^{-3}	0.5519×10^{-3}	0.3240×10^{-3}
Minimum	0.5434×10^{-3}	0.5434×10^{-3}	0.3235×10^{-3}

Squared root of the tracking error ISE (Integral of Squared Error), i.e., $\sqrt{\int_0^T \|\boldsymbol{r}\|^2 \, \mathrm{d}t}$, where T denotes a simulation time.

Parameter Dependencies Investigation:

- The parameters ranged from 0.001 to 1 across 10 samples.
- NAC-L2 shows the worst performance with high variance.

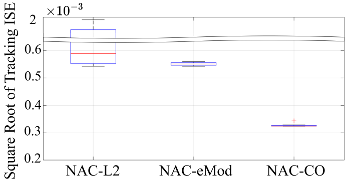


Figure: Box-and-whisker plots of the tracking error ISE.

	NAC-L2	**NAC-eMod**	**NAC-CO** (proposed)
Maximum	11.1753×10^{-3}	0.5603×10^{-3}	0.3439×10^{-3}
Median	0.5898×10^{-3}	0.5519×10^{-3}	0.3240×10^{-3}
Minimum	0.5434×10^{-3}	0.5434×10^{-3}	0.3235×10^{-3}

Squared root of the tracking error ISE (Integral of Squared Error), i.e., $\sqrt{\int_0^T \|r\|^2 \, dt}$, where T denotes a simulation time.

Parameter Dependencies Investigation:

- The parameters ranged from 0.001 to 1 across 10 samples.
- NAC-L2 shows the worst performance with high variance.
- NAC-CO (proposed) shows the best performance and lowest variance.

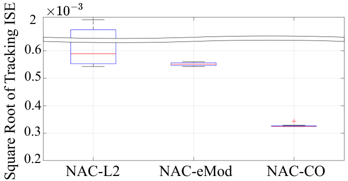


Figure: Box-and-whisker plots of the tracking error ISE.

	NAC-L2	**NAC-eMod**	**NAC-CO** (proposed)
Maximum	11.1753×10^{-3}	0.5603×10^{-3}	0.3439×10^{-3}
Median	0.5898×10^{-3}	0.5519×10^{-3}	0.3240×10^{-3}
Minimum	0.5434×10^{-3}	0.5434×10^{-3}	0.3235×10^{-3}

Squared root of the tracking error ISE (Integral of Squared Error), i.e., $\sqrt{\int_0^T \|\boldsymbol{r}\|^2 \, \mathrm{d}t}$, where T denotes a simulation time.

Parameter Dependencies Investigation:

- The parameters ranged from 0.001 to 1 across 10 samples.
- NAC-L2 shows the worst performance with high variance.
- NAC-CO (proposed) shows the best performance and lowest variance.
- This result is because,

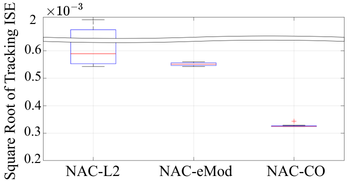


Figure: Box-and-whisker plots of the tracking error ISE.

	NAC-L2	**NAC-eMod**	**NAC-CO** (proposed)
Maximum	11.1753×10^{-3}	0.5603×10^{-3}	0.3439×10^{-3}
Median	0.5898×10^{-3}	0.5519×10^{-3}	0.3240×10^{-3}
Minimum	0.5434×10^{-3}	0.5434×10^{-3}	0.3235×10^{-3}

Squared root of the tracking error ISE (Integral of Squared Error), i.e., $\sqrt{\int_0^T \|r\|^2 \, \mathrm{d}t}$, where T denotes a simulation time.

MIC Lab
Mobility Intelligence and Control Laboratory

Parameter Dependencies Investigation:

- The parameters ranged from 0.001 to 1 across 10 samples.
- NAC-L2 shows the worst performance with high variance.
- NAC-CO (proposed) shows the best performance and lowest variance.
- This result is because,
  - NAC-L2 and NAC-eMod are biased towards the origin. $\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha(\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}} + \lambda\widehat{\boldsymbol{\theta}})$ (NAC-L2) or $+\rho\|\boldsymbol{r}\|\widehat{\boldsymbol{\theta}}$ (NAC-eMod), proportionally to λ and ρ, respectively.

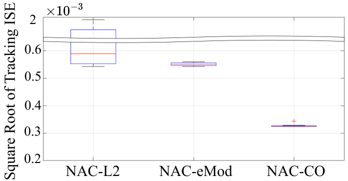


Figure: Box-and-whisker plots of the tracking error ISE.

	NAC-L2	NAC-eMod	NAC-CO (proposed)
Maximum	11.1753×10^{-3}	0.5603×10^{-3}	0.3439×10^{-3}
Median	0.5898×10^{-3}	0.5519×10^{-3}	0.3240×10^{-3}
Minimum	0.5434×10^{-3}	0.5434×10^{-3}	0.3235×10^{-3}

Squared root of the tracking error ISE (Integral of Squared Error), i.e., $\sqrt{\int_0^T \|\boldsymbol{r}\|^2 \,\mathrm{d}t}$, where T denotes a simulation time.

Parameter Dependencies Investigation:

- The parameters ranged from 0.001 to 1 across 10 samples.
- NAC-L2 shows the worst performance with high variance.
- NAC-CO (proposed) shows the best performance and lowest variance.
- This result is because,
  - NAC-L2 and NAC-eMod are biased towards the origin. $\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha(\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}} + \lambda\widehat{\boldsymbol{\theta}})$ (NAC-L2) or $+\rho\|\boldsymbol{r}\|\widehat{\boldsymbol{\theta}}$ (NAC-eMod), proportionally to λ and ρ, respectively.
  - $-\lambda_j \frac{\partial c_j}{\partial \widehat{\boldsymbol{\theta}}}$ in NAC-CO (proposed) (i.e., $\frac{\mathrm{d}}{\mathrm{d}t}\widehat{\boldsymbol{\theta}} = -\alpha(\frac{\partial J}{\partial \widehat{\boldsymbol{\theta}}} + \lambda_j \frac{\partial c_j}{\partial \widehat{\boldsymbol{\theta}}})$) disappears when constraints are inactive (i.e., $c_j < 0$, and $\lambda = \beta_j c_j$ and $\lambda_j \leftarrow \max(\lambda_j, 0)$).

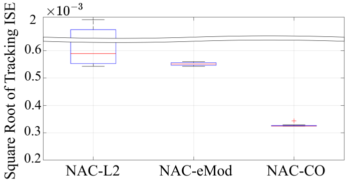


Figure: Box-and-whisker plots of the tracking error ISE.

	NAC-L2	NAC-eMod	NAC-CO (proposed)
Maximum	11.1753×10^{-3}	0.5603×10^{-3}	0.3439×10^{-3}
Median	0.5898×10^{-3}	0.5519×10^{-3}	0.3240×10^{-3}
Minimum	0.5434×10^{-3}	0.5434×10^{-3}	0.3235×10^{-3}

Squared root of the tracking error ISE (Integral of Squared Error), i.e., $\sqrt{\int_0^T \|\boldsymbol{r}\|^2 \,\mathrm{d}t}$, where T denotes a simulation time.

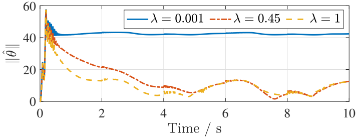


Figure: Weight norms of NAC-L2

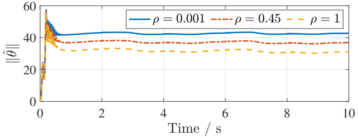


Figure: Weight norms of NAC-eMod

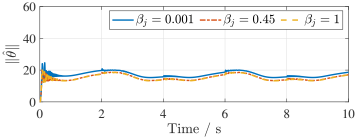


Figure: Weight norms of NAC-CO (proposed)

- NAC-CO (proposed) showed the weight norms are bounded under pre-defined constraint $\overline{\theta} = 20$.

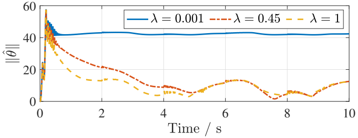


Figure: Weight norms of NAC-L2

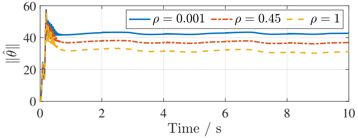


Figure: Weight norms of NAC-eMod

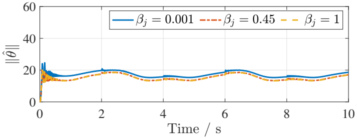


Figure: Weight norms of NAC-CO (proposed)

- NAC-CO (proposed) showed the weight norms are bounded under pre-defined constraint $\overline{\theta} = 20$.
- NAC-L2 and NAC-eMod showed the bounded weight norms, but they depended on the parameters λ and ρ, respectively.

Simulation Results Weight Norms

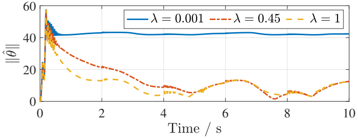


Figure: Weight norms of NAC-L2

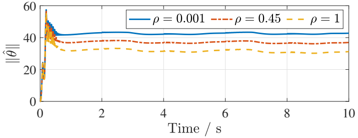


Figure: Weight norms of NAC-eMod

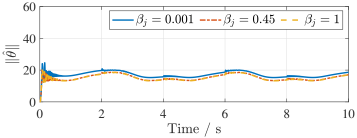


Figure: Weight norms of NAC-CO (proposed)

- NAC-CO (proposed) showed the weight norms are bounded under pre-defined constraint $\overline{\theta} = 20$.
- NAC-L2 and NAC-eMod showed the bounded weight norms, but they depended on the parameters λ and ρ, respectively.
- In other words, NAC-CO tracked the desired trajectory with a smaller weight norm than NAC-L2 and NAC-eMod.

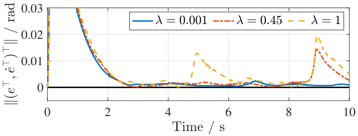


Figure: Tracking error of NAC-L2

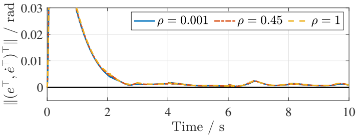


Figure: Tracking error of NAC-eMod

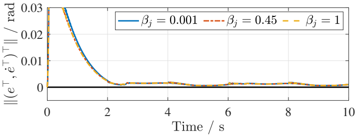


Figure: Tracking error of NAC-CO (proposed)

- NAC-CO (proposed) outperformed NAC-L2 and NAC-eMod in terms of tracking performance.
- As the weights are biased towards the origin proportionally to the parameters λ and ρ in NAC-L2 and NAC-eMod, respectively, the tracking performance of NAC-L2 and NAC-eMod deteriorated, as approaching toward suboptimal points.

Real-time Implementation Result

- This video demonstrates:
  - Applicability of the proposed method to real-time control (under 4 ms sampling time).
  - Convex input constraints handling.

Outline

1 Background and Contributions

2 Proposed Method

3 Experimental Validation

4 Conclusion

- Conclusion and Future Work

Summary of Contributions

- Proposed a novel constrained optimization-based neuro-adaptive control (CONAC) method.
- Adaptation laws are derived using constrained optimization method.
- The proposed method guarantees the stability of the system and the boundedness of the NN weights.
- Feasibility of the proposed method is validated through numerical simulations.

Future Work

- Extend the proposed method to state constraints.
- Enhance the robustness and flexibility of the proposed method for various systems.

Thank you for your attention!

[1] J. A. Farrell and M. M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches (Adaptive and Learning Systems for Signal Processing, Communications and Control Series)*. USA: Wiley-Interscience, 2006. [Online]. Available: https://doi.org/10.1002/0471781819

[2] M. Ryu, N. Monzen, P. Seitter, K. Choi, and C. M. Hackl, "Constrained optimization-based neuro-adaptive control (conac) for synchronous machine drives under voltage constraints," TechRxiv, Preprint, Apr. 2025. [Online]. Available: http://dx.doi.org/10.36227/techrxiv.174585949.94234666/v1

[3] J. Nocedal and S. Wright, *Numerical optimization*, 2nd ed., ser. Springer series in operations research and financial engineering. New York, NY: Springer, 2006. [Online]. Available: https://doi.org/10.1007/978-0-387-40065-5