Lyapunov Stability in AI-Based Control

Motivation

Neural networks (NNs) are powerful function approximators, but:

- They do not inherently guarantee stability.
- They are hard to interpret or certify.
- In control systems, we require provable stability (often via Lyapunov functions).

Question: Can we train a neural network (e.g., controller or model) and still prove that the system is stable?

Core Idea

We combine Lyapunov theory with AI by either:

1. **Lyapunov-Constrained Learning**: Enforce that a Lyapunov function decreases along trajectories.
2. **Learning Lyapunov Functions**: Use a neural network to learn a valid Lyapunov function from data.

1. Lyapunov-Constrained Learning

We require that a candidate Lyapunov function $V(x)$ satisfies:

$$\dot{V}(x) \leq -\alpha \|x\|^2$$

This can be enforced via:

- LMIs (Linear Matrix Inequalities)
- Differentiable constraints or penalty terms
- Safe reinforcement learning formulations

For a system of the form:

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

and a neural network controller $u(x;\theta)$, we define:

$$\dot{V}(x) = \nabla V(x)^\top (f(x) + g(x)u(x))$$

Then, we penalize violation of the Lyapunov condition during training:

$$L = L_{\text{performance}} + \lambda \cdot \text{ReLU}\left(\dot{V}(x) + \alpha \|x\|^2\right)$$

2. Learning Lyapunov Functions

Alternatively, we train a neural network to represent $V(x)$, ensuring it satisfies:

$$\begin{aligned} V(x) &> 0, \quad \forall x \neq 0 \\ \dot{V}(x) &< 0, \quad \forall x \neq 0 \end{aligned}$$

This can be done either as part of training a controller or post hoc to verify stability of a learned policy.

Tools and Frameworks

- **CVXPYLayer** (PyTorch): Differentiable convex optimization layers.
- **Lyapunov neural networks**: NN models trained to act as valid Lyapunov functions.
- **Control Lyapunov Function (CLF)**: Traditional Lyapunov theory embedded in learning.
- **Safe RL / Constrained Policy Optimization**: Incorporate stability conditions as constraints.

Example Workflow

Given a system:

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

Train a neural network controller $u(x;\theta)$ with the constraint:

$$\dot{V}(x) = \nabla V(x)^\top (f(x) + g(x)u(x)) \leq -\alpha \|x\|^2$$

This ensures asymptotic stability, provided $V(x)$ is positive definite.

Summary Table

Goal	Method
Stabilize NN controller	Lyapunov-constrained training
Learn a Lyapunov function	Use neural network to approximate $V(x)$
Certify a learned policy	Train/verify $V(x)$ post hoc
Reinforce stability during RL	Penalize $\dot{V}(x) > 0$ violations

Suggested Readings

- *Safe Control with Learned Control Barrier Functions*
- *Lyapunov Networks: Dynamically Stable Neural Network Models*
- *Constrained Policy Optimization (Achiam et al.)*
- *Stable Reinforcement Learning via Policy Gradient with Lyapunov Constraints*