Enforcing robust control guarantees within neural network policies

Priya L. Donti
Carnegie Mellon University

Joint work with Melrose Roderick, Mahyar Fazlyab, Zico Kolter



Robust control

Goal: Certifiable robustness within some specified perturbation set

Unknown (nonlinear) system

Model of system uncertainty (e.g., NLDI)

$$\dot{x}(t) \in Ax(t) + Bu(t) + Gw(t) \quad \text{s.t.} \|w(t)\|_2 \leq \|Cx(t) + Du(t)\|_2$$

$$u(t) = Kx(t)$$

Simple policy

V

continuous, non-negative

Lyapunov function

s.t. $\dot{V}(x(t)) \leq -\alpha V(x(t)) \; \forall t$

"Sufficient decrease" condition

Our approach

Step 1: Obtain Lyapunov function V via robust control

Step 2: Construct policy π_θ

- Construct deep network $\hat{\pi}_\theta$
- Project output onto stabilizing action set $\mathcal{C}(x(t))$ satisfying sufficient decrease of V

$$\pi_\theta(x(t)) = \text{Proj}_{\mathcal{C}(x(t))}\left(\hat{\pi}_\theta(x(t))\right)$$

Step 3: Train end-to-end using standard deep RL techniques

Finding a stabilizing set

Given: Lyapunov function V (obtained via robust control)

Find: For given x, the set of actions decreasing V even in the worst case

$$\mathcal{C}(x) \equiv \{ u: \left(\sup_{w\,:\,\|w\|_2 \leq \|Cx+Du\|_2} \dot{V}(x) \right) \leq -\alpha V(x) \}$$

$$\Rightarrow \{u: \|k_1(x) + Du\|_2 \leq k_2(x) + k_3(x)^T u\}$$

Convex set in $u(t)$

Non-empty: $Kx \in \mathcal{C}(x)$

Note: t-dependence has been dropped for brevity

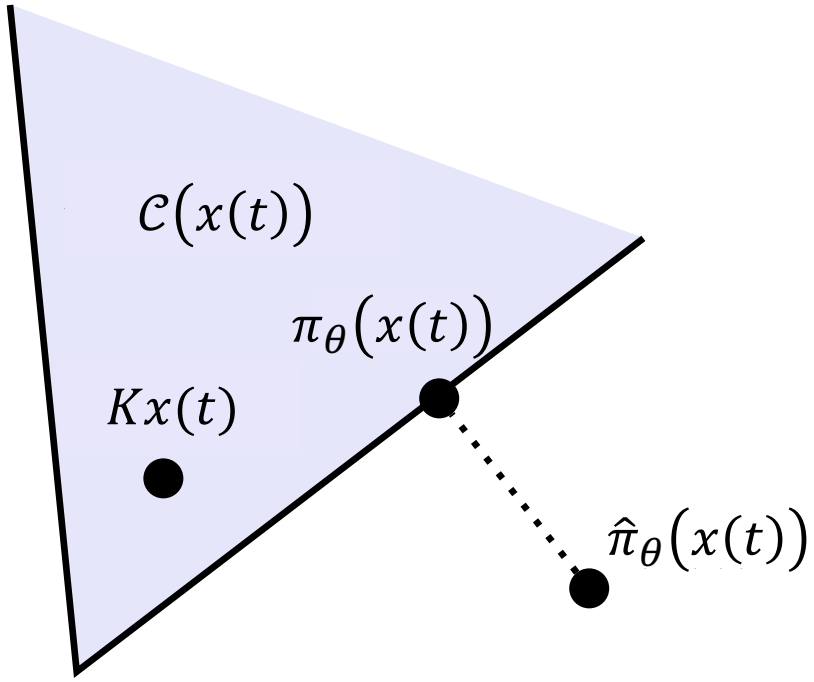
Embedding robust control constraints in deep RL

Project output of deep network onto stabilizing action set

$$\pi_\theta\big(x(t)\big) = \mathrm{Proj}_{\mathcal{C}(x(t))}\left(\hat{\pi}_\theta\big(x(t)\big)\right)$$

Differentiate through projection using recent techniques in differentiable optimization (e.g., Amos and Kolter, 2018)

- Implicit function theorem applied to KKT conditions or fixed-point equations of convex optimization problem



Note: We build a custom differentiable solver

Experiments

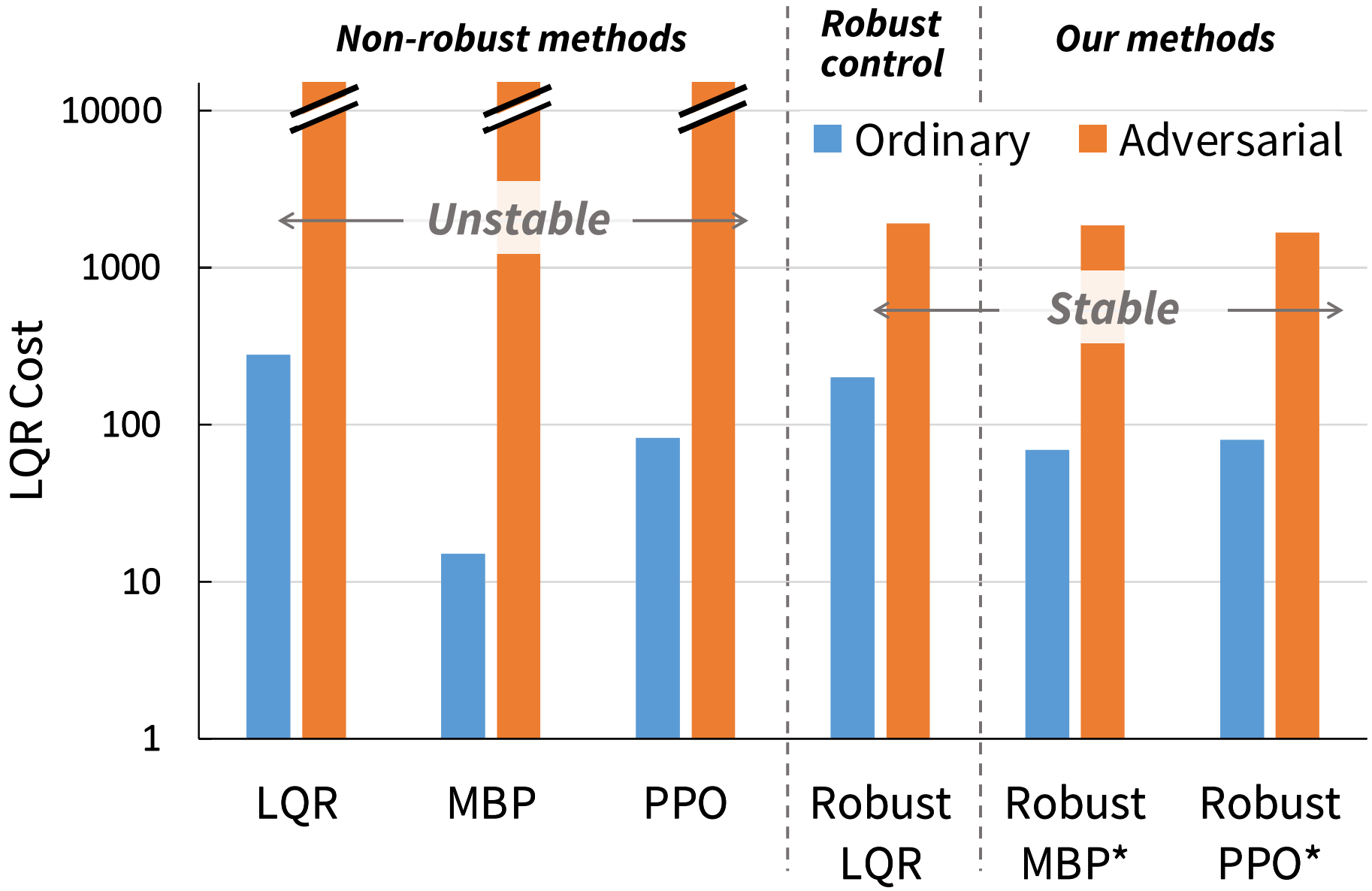
(Simulated) experiments on four different domains

1. Random norm-bounded linear differential inclusions (NLDIs)
2. Cartpole stabilization
3. Quadrotor control
4. Microgrid simulator
5. PLDI and H_∞ settings

In all cases, compare performance under

- “Ordinary” operation
- “Adversarial” disturbances

Illustrative results: NLDI



Summary

Method for learning provably robust nonlinear policies using deep RL

Key insight: *Project* output of neural network onto stabilizing set of actions