Stability Analysis of Complementarity Systems with Neural Network Controllers

Alp Aydinoglu
University of Pennsylvania
alpayd@seas.upenn.com

Mahyar Fazlyab
Johns Hopkins University
mahyarfazlyab@jhu.edu

Manfred Morari
University of Pennsylvania
morari@seas.upenn.edu

Michael Posa
University of Pennsylvania
posa@seas.upenn.edu

ABSTRACT

Complementarity problems, a class of mathematical optimization problems with orthogonality constraints, are widely used in many robotics tasks, such as locomotion and manipulation, due to their ability to model non-smooth phenomena (e.g., contact dynamics). In this paper, we propose a method to analyze the stability of complementarity systems with neural network controllers. First, we introduce a method to represent neural networks with rectified linear unit (ReLU) activations as the solution to a linear complementarity problem. Then, we show that systems with ReLU network controllers have an equivalent linear complementarity system (LCS) description. Using the LCS representation, we turn the stability verification problem into a linear matrix inequality (LMI) feasibility problem. We demonstrate the approach on several examples, including multi-contact problems and friction models with non-unique solutions.

CCS CONCEPTS

• **Mathematics of computing → Mathematical optimization**;
• **Computer systems organization → Robotics**.

KEYWORDS

Stability verification of neural network controllers, Complementarity systems, Contact dynamics

1 INTRODUCTION

Due to recent advancements in deep learning, there has been an increasing interest in using neural networks (NNs) to stabilize dynamical systems. For instance, neural networks have been used to

approximate model predictive control policies through supervised learning [27, 29, 37, 50], or reinforcement learning [11]. Although neural network controllers can achieve satisfactory performance under mild assumptions about the model of the dynamical system or the environment it operates in, they lack guarantees. This drawback limits the application of neural networks in safety-critical systems. Therefore, it is critical to develop tools that can provide useful certificates of stability, and robustness for NN-driven systems.

Many important robotics systems are non-smooth and researchers have shown the effectiveness of NN policies on such systems without providing formal guarantees [23, 48, 51]. The goal of this paper is to introduce a method for stability analysis of non-smooth systems in feedback loops with NN controllers. Our framework is inspired by complementarity systems [26], differential equations coupled with the solution of a linear complementarity problem. Complementarity problems are a class of mathematical optimization problems with orthogonality constraints [13]. Linear complementarity problems, in particular, are widely used in computational non-smooth mechanics with unilateral contacts and friction [8], and more generally, in applications involving quadratic programming [36]. In simple terms, a linear complementarity problem can be stated as the following potentially non-convex quadratic optimization problem,

$$\text{minimize } \lambda^\top (F\lambda + q) \quad \text{subject to } F\lambda + q \geq 0, \ \lambda \geq 0.$$

With the objective function being non-negative, the solutions to the optimization problem satisfy the complementarity condition $(F\lambda + q)^\top \lambda = 0$. In the context of contact dynamics, for example, one can interpret λ as a contact force between a robot and a surface, and $F\lambda + q$ is a gap function relating the contact force and the distance from the robot to the contact surface. Because of their ability to model set-valued and non-smooth functions, complementarity problems are widely used within the robotics community for stability analysis [2, 9, 22, 28, 40], simulating contact dynamics [24, 45], and trajectory optimization [39].

1.1 Related Work

The connection between nonlinearities in neural networks and mathematical optimization has been exploited recently in various contexts. In [19, 20, 41] the authors use quadratic constraints to describe ReLU activation functions followed by a semidefinite relaxation to perform robustness analysis of ReLU networks. In [21], the authors exploit the fact that all commonly used activation functions

in deep learning are gradients of convex potentials, hence they satisfy incremental quadratic constraints that can be used to bound the global Lipschitz constant of feed-forward neural networks. The works [17, 18, 47] perform reachability analysis for closed-loop systems with neural network controllers and [43] considers safety verification for such systems. The work in [1] integrates quadratic programs as end-to-end trainable deep networks to encode constraints and more complex dependencies between the hidden states. Yin et al. [49] considers uncertain linear time-invariant systems with neural network controllers. By over approximating the input-output map of the neural network and uncertainties by quadratic and integral quadratic constraints, respectively, the authors develop an SDP whose solution yields quadratic Lyapunov functions. In [10] the authors develop a learning-based iterative sample guided strategy based on the analytic center cutting plane method to search for Lyapunov functions for piece-wise affine systems in feedback with ReLU networks where the generation of samples relies on solving mixed-integer quadratic programs. In [30], the authors use a mixed-integer programming formulation to perform output range analysis of ReLU neural networks and provide guarantees for constraint satisfaction and asymptotic stability of the closed-loop system.

1.2 Contributions

Inspired by the connection between ReLU functions and linear complementarity problems, we develop a method to analyze linear complementarity systems in feedback with ReLU network controllers. Our starting point is to show that we can represent ReLU neural networks as linear complementarity problems (Lemma 2). Next, we demonstrate that linear complementarity systems with neural network controllers have an equivalent LCS representation. Then, we leverage the theory of stability analysis for complementarity systems and derive the discrete time version of the results in [9]. We describe the sufficient conditions for stability in the form of Linear Matrix Inequalities (LMI's). Denoting by N the number of neurons in the network plus the number of complementarity variables in the LCS, the size of the LMI's scales linearly with N. Furthermore, the maximum possible number of decision variables in our LMI scales quadratically with N. To the best of our knowledge, this is the first work on analyzing the stability of LCS systems with neural network controllers.

2 BACKGROUND

2.1 Notation

We denote the set of non-negative integers by $\mathbb{N}_0$, the set of d-dimensional vectors with real components as $\mathbb{R}^d$ and the set of $n \times m$ dimensional matrices by $\mathbb{R}^{n \times m}$. For two vectors $a \in \mathbb{R}^m$ and $b \in \mathbb{R}^m$, we use the notation $0 \le a \perp b \ge 0$ to denote that $a \ge 0$, $b \ge 0$, $a^T b = 0$. For a positive integer l, $\bar{l}$ denotes the set $\{1, 2, \ldots, l\}$. Given a matrix $M \in \mathbb{R}^{k \times l}$ and two subsets $I \subseteq \bar{k}$ and $J \subseteq \bar{l}$, we define $M_{IJ} = (m_{ij})_{i \in I, j \in J}$. For the case where $J = \bar{l}$, we use the shorthand notation $M_{I\bullet}$.

2.2 Linear Complementarity Problem

The theory of linear complementarity problems (LCP) will be used throughout this work [13].

DEFINITION 1. *Given a vector $q \in \mathbb{R}^m$, and a matrix $F \in \mathbb{R}^{m \times m}$, the $LCP(q, F)$ describes the following mathematical program:*

$$\begin{aligned} &\text{find} && \lambda \in \mathbb{R}^m \\ &\text{subject to} && y = F\lambda + q, \\ &&& 0 \le \lambda \perp y \ge 0. \end{aligned}$$

The solution set of the $LCP(q, F)$ is denoted by

$$\text{SOL}(q, F) = \{\lambda : y = F\lambda + q, 0 \le \lambda \perp y \ge 0\}.$$

The $LCP(q, F)$ may have multiple solutions or none at all. The cardinality of the solution set $\text{SOL}(q, F)$ depends on the matrix F and the vector q.

DEFINITION 2. *A matrix $F \in \mathbb{R}^{m \times m}$ is a P-matrix, if the determinant of all of its principal sub-matrices are positive; that is, $det(F_{\alpha\alpha}) > 0$ for all $\alpha \subseteq \{1, \ldots, m\}$.*

The solution set $\text{SOL}(q, F)$ is a singleton for all q if F is a P-matrix [13]. If we denote the unique element of $\text{SOL}(q, F)$ as $\lambda(q)$, then $\lambda(q)$ is a piece-wise linear function of q. We can describe this function explicitly as in [9]. Consider $y = F\lambda(q) + q$, and define the index sets

$$\alpha(q) = \{i : \lambda_i(q) > 0 = y_i\},$$
$$\beta(q) = \{i : \lambda_i(q) = 0 \le y_i\},$$

Then, $\lambda(q)$ is equivalent to

$$\lambda_\alpha(q) = -(F_{\alpha\alpha})^{-1} I_{\alpha\bullet} q, \quad \lambda_\beta(q) = 0, \tag{1}$$

where $\alpha = \alpha(q)$ and $\beta = \beta(q)$. Furthermore, $\lambda(q)$ as in (1) is Lipschitz continuous since it is a continuous piece-wise linear function of q [42].

2.3 Linear Complementarity Systems

We are now ready to introduce linear complementarity systems (LCS). In this work, we consider an LCS as a difference equation coupled with a variable that is the solution of an LCP.

DEFINITION 3. *A linear complementarity system describes the trajectories $(x_k)_{k \in \mathbb{N}_0}$ and $(\lambda_k)_{k \in \mathbb{N}_0}$ for an input sequence $(u_k)_{k \in \mathbb{N}_0}$ starting from x_0 such that*

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + D\lambda_k + z, \\ y_k &= Ex_k + F\lambda_k + Hu_k + c, \\ 0 &\le \lambda_k \perp y_k \ge 0. \end{aligned} \tag{2}$$

where $x_k \in \mathbb{R}^{n_x}$, $\lambda_k \in \mathbb{R}^{n_\lambda}$, $u_k \in \mathbb{R}^{n_u}$, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $D \in \mathbb{R}^{n_x \times n_\lambda}$, $z \in \mathbb{R}^{n_x}$, $E \in \mathbb{R}^{n_\lambda \times n_x}$, $F \in \mathbb{R}^{n_x \times n_\lambda}$, $H \in \mathbb{R}^{n_\lambda \times n_u}$ and $c \in \mathbb{R}^{n_\lambda}$.

For a given k, x_k and u_k, the corresponding complementarity variable λ_k can be found by solving $\text{LCP}(Ex_k + Hu_k + c, F)$ (see Definition 1). Similarly, x_{k+1} can be computed using the first equation in (2) when x_k, u_k and λ_k are known. In general, the trajectories (x_k) and (λ_k) are not unique since $\text{SOL}(Ex_k + Hu_k + c, F)$ can have multiple elements; hence, (2) is a difference *inclusion* [16].

In this work, we will focus on autonomous linear complementarity systems (A-LCS) because we consider the input as a function of the state and the complementarity variable, i.e., $u_k = u_k(x_k, \lambda_k)$.

An A-LCS represents the evolution of trajectories $(x_k)_{k \in \mathbb{N}_0}$ and $(\lambda_k)_{k \in \mathbb{N}_0}$ according to following dynamics,

$$
\begin{aligned}
x_{k+1} &= Ax_k + D\lambda_k + z, \\
y_k &= Ex_k + F\lambda_k + c, \\
0 &\leq \lambda_k \perp y_k \geq 0,
\end{aligned}
\tag{3}
$$

and unlike (2) there is no input. Moving forward, we will consider A-LCS models that can have non-unique trajectories.

We note that, however, the existence of a special case of (3) is continuous piecewise affine systems [25]. If F is a P-matrix, then $\lambda(x_k)$ is unique for all x_k and (3) is equivalent to

$$
x_{k+1} = Ax_k + Bu_k + D\lambda(x_k) + z,
$$

where $\lambda(x_k)$ is the unique element of $\text{SOL}(Ex_k + c, F)$ and can be explicitly described as in (1). In this setting, (2) is a piece-wise affine dynamical system and has a unique solution for any initial condition x_0.

2.4 Stability of A-LCS

We introduce the notions of stability for A-LCS that are similar to [44]. An equilibrium point x_e for (3) is defined as a point that satisfies $x_e = Ax_e + D\lambda_e + z$ where $\lambda_e = \text{SOL}(Ex_e + c, F)$ is a singleton. Without loss of generality, we assume $x_e = 0$ is an equilibrium of the system, i.e., $D \cdot \text{SOL}(c, F) = \{-z\}$.

DEFINITION 4. *The equilibrium $x_e = 0$ of A-LCS is*

(1) *stable if for any given $\epsilon > 0$, there exists a $\delta > 0$ such that*

$$||x_0|| \leq \delta \implies ||x_k|| \leq \epsilon \ \forall k \geq 0,$$

for any trajectory $\{x_k\}$ starting from x_0,

(2) *asymptotically stable if it is stable and there is a $\delta > 0$ such that*

$$||x_0|| \leq \delta \implies \lim_{k \to \infty} ||x_k|| = 0,$$

for any trajectory $\{x_k\}$ starting from x_0.

(3) *geometrically stable if there exists $\delta > 0$, $\alpha > 1$ and $0 < \rho < 1$ such that*

$$||x_0|| \leq \delta \implies ||x_k|| \leq \alpha \rho^k ||x_0|| \ \forall k \geq 0,$$

for any trajectory $\{x_k\}$ starting from x_0.

Notice that if F is a P-matrix, these are equivalent to the notions of stability for difference equations where the right side is Lipschitz continuous [31] since there is a unique trajectory $\{x_k\}$ starting from any initial condition x_0.

3 LINEAR COMPLEMENTARITY SYSTEMS WITH NEURAL NETWORK CONTROLLERS

In this section, we demonstrate that neural networks with rectified linear units (ReLU) have an equivalent LCP representation. Then, we show that an LCS combined with a neural network controller has an alternative complementarity system description.

DEFINITION 5. *A ReLU neural network $\phi \colon \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_\phi}$ with L hidden layers is the composite function*

$$\phi(x) = (h_L \circ \lambda_{ReLU} \circ h_{L-1} \circ \ldots \circ \lambda_{ReLU} \circ h_0)(x), \tag{4}$$

where $\lambda_{ReLU}(x) = \max\{0, x\}$ is the ReLU activation layer, and $h_i(x) = \theta_i x + c_i$ are the affine layers with $\theta_i \in \mathbb{R}^{n_{i+1} \times n_i}$, $c_i \in \mathbb{R}^{n_{i+1}}$.
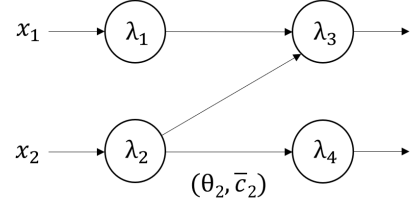

Figure 1: Two-layered neural network with ReLU activation functions (Example 1).

Here, n_i, $1 \leq i \leq L$ denotes the number of hidden neurons in the i-th layer, $n_0 = n_x$, and $n_{L+1} = n_\phi$. We denote by $n_t = \sum_{i=1}^{L} n_i$ the total number of neurons.

3.1 Representing ReLU Neural Networks as Linear Complementarity Problems

ReLU neural networks are piece-wise affine functions. Similarly, the linear complementarity problem describes a piece-wise affine function as shown in (1) as long as F is a P-matrix. In this section, we will explore the connection between two piece-wise affine representations.

It has been shown that ReLU neural networks can be represented with quadratic constraints [41], [20]. Now, our goal is to show the connection between these results and linear complementarity problems. We will describe a method to represent a multi-layered ReLU neural network as a linear complementarity problem exploring the cascade connections of complementarity problems [7].

First we consider a single ReLU unit $\lambda_{\text{ReLU}}(x) = \max(0, x)$ and its equivalent LCP representation.

LEMMA 1. *[26] Consider the following LCP for a given $x \in \mathbb{R}^d$:*

$$
\begin{aligned}
&\text{find} && \lambda^{LCP} \in \mathbb{R}^d \\
&\text{subject to} && \bar{y} = -x + \lambda^{LCP}, \\
& && 0 \leq \lambda^{LCP} \perp \bar{y} \geq 0,
\end{aligned}
$$

Then λ^{LCP} is unique and is given by $\lambda^{LCP} = \max\{0, x\}$.

PROOF. If $x_i < 0$, then $\lambda_i^{\text{LCP}} = 0$ and if $x_i \geq 0$, then $\lambda_i^{\text{LCP}} = x_i$. □

Next, we consider a two layered neural network and transform it into an LCP using Lemma 1.

Example 1. Consider a two layered network shown in Figure 1 where $\phi_{\text{2-layer}}(x) = \lambda_{\text{ReLU}} \circ h_1 \circ \lambda_{\text{ReLU}} \circ h_0(x)$ with $h_0(x) = x$ and $h_1(x) = \theta_2 x + \bar{c}_2$. An alternative representation is $\phi_{\text{2-layer}}(x) = \begin{bmatrix} \lambda_3(x)^\top & \lambda_4(x)^\top \end{bmatrix}^\top$ where

$$
\begin{aligned}
\lambda_1(x) &= \max\{0, x_1\}, & (5) \\
\lambda_2(x) &= \max\{0, x_2\}, & (6) \\
\lambda_3(x) &= \max\{0, \theta_2^{1,1}\lambda_1(x) + \theta_2^{1,2}\lambda_2(x) + \bar{c}_2^1\}, & (7) \\
\lambda_4(x) &= \max\{0, \theta_2^{2,2}\lambda_2(x) + \bar{c}_2^2\}. & (8)
\end{aligned}
$$

Here λ_i represents the output of the ith ReLU activation function, $\theta_i^{j,k}$ and $\bar{c}_i^j$ represent the coefficients of the affine function. Observe that the two-layered NN is equivalent to $\begin{bmatrix} \lambda_3 \\ \lambda_4 \end{bmatrix}$ where λ is the unique solution of the following LCP:

$$\text{find} \qquad \lambda \in \mathbb{R}^4$$
$$\text{subject to} \quad \bar{y}_1 = -x_1 + \lambda_1,$$
$$\bar{y}_2 = -x_2 + \lambda_2,$$
$$\bar{y}_3 = -\theta_2^{1,1}\lambda_1 - \theta_2^{1,2}\lambda_2 - \bar{c}_2^1 + \lambda_3,$$
$$\bar{y}_4 = -\theta_2^{2,2}\lambda_2 - \bar{c}_2^2 + \lambda_4,$$
$$0 \leq \lambda_1 \perp \bar{y}_1 \geq 0,$$
$$0 \leq \lambda_2 \perp \bar{y}_2 \geq 0,$$
$$0 \leq \lambda_3 \perp \bar{y}_3 \geq 0,$$
$$0 \leq \lambda_4 \perp \bar{y}_4 \geq 0.$$

Here, $\{\lambda_i\}_{i=1}^2$ can be represented as $\lambda_i = \max\{0, x_i\}$ and $\{\lambda_i\}_{i=3}^4$ are as in (7), (8) after direct application of Lemma 1. Then, we conclude that $\begin{bmatrix} \lambda_3 \\ \lambda_4 \end{bmatrix} = \phi_{\text{2-layer}}$.

Now, we show that all neural networks of the form (4) have an equivalent LCP representation.

LEMMA 2. *For any x, the ReLU neural network in (4) can be expressed as $\phi(x) = \bar{D}\lambda(x) + \bar{z}$, where $\lambda(x)$ is the unique solution of the following linear complementarity problem:*

$$\text{find} \qquad \lambda$$
$$\text{subject to} \quad \bar{y} = \bar{E}x + \bar{F}\lambda + \bar{c},$$
$$0 \leq \lambda \perp \bar{y} \geq 0,$$

where $\bar{c} = \begin{bmatrix} -c_0 \\ -c_1 \\ \vdots \\ -c_{L-1} \end{bmatrix}$, $\bar{E} = \begin{bmatrix} -\theta_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, *the lower triangular matrix*

$$\bar{F} = \begin{bmatrix} I & & & & & \\ -\theta_1 & I & & & & \\ 0 & -\theta_2 & I & & & \\ 0 & 0 & -\theta_3 & I & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & \cdots & -\theta_{L-1} & I \end{bmatrix},$$

and $\bar{z} = \bar{c}_L$, $\bar{D} = \begin{bmatrix} 0 & 0 & \dots & 0 & \theta_L \end{bmatrix}$ where $\bar{F}$ is a P-matrix.

PROOF. First, we write $\lambda^\top = \begin{bmatrix} \lambda_0^\top & \lambda_1^\top & \cdots & \lambda_{L-1}^\top \end{bmatrix} \in \mathbb{R}^{1 \times n_t}$ where each sub vector λ_i has the same dimension as $\bar{c}_i$. Next, we show that $\lambda_0(x) = \lambda_{\text{ReLU}} \circ h_0(x)$. Observe that λ_0 is independent of $\lambda_1, \dots, \lambda_{L-1}$ since F is lower-triangular. Hence λ_0 is the unique element of the following LCP:

$$\text{find} \qquad \lambda_0$$
$$\text{subject to} \quad \bar{y}_0 = -\theta_0 x - \bar{c}_0 + \lambda_0,$$
$$0 \leq \lambda_0 \perp \bar{y}_0 \geq 0.$$

Following Lemma 1, $\lambda_0(x) = \max\{0, h_0(x)\} = \lambda_{\text{ReLU}} \circ h_0(x)$. Similarly, notice that for $i > 0$, λ_i only depends on $\lambda_{i-1}(x)$ and is the unique element of:

$$\text{find} \qquad \lambda_i$$
$$\text{subject to} \quad \bar{y}_i = -\theta_i \lambda_{i-1}(x) - \bar{c}_i + \lambda_i,$$
$$0 \leq \lambda_i \perp \bar{y} \geq 0,$$

and is equivalent to $\lambda_i(x) = \lambda_{\text{ReLU}} \circ h_i \circ \lambda_{i-1}(x)$ as a direct application of Lemma 1. Using this equivalency recursively,

$$\bar{D}\lambda(x) + \bar{z} = h_L \circ \lambda_{\text{ReLU}} \circ h_{L-1} \circ \dots \circ \lambda_{\text{ReLU}} \circ h_0(x).$$

Notice that $\bar{F}_{\alpha\alpha}$ is lower triangular with ones on the diagonal for any α such that $\text{card}(\alpha) \geq 2$ hence $\bar{F}$ is a P-matrix. □

Each neuron in the NN is represented with a complementarity variable, therefore the dimension of the complementarity vector (λ) is equal to the number of neurons in the network. As seen in Lemma 2, transforming a ReLU neural network into an LCP only requires concatenating vectors and matrices. Furthermore, this transormation allows us to consider the ReLU neural network controller as an LCP based controller [46].

3.2 Linear Complementarity Systems with Neural Network Controllers

We will use the LCP representation of the neural network (4) and describe an LCS with a NN controller as an A-LCS. Consider a linear complementarity system with a ReLU neural network controller $u_k = \phi(x_k)$:

$$x_{k+1} = Ax_k + B\phi(x_k) + \tilde{D}\tilde{\lambda}_k + \tilde{z},$$
$$\tilde{y}_k = \tilde{E}x_k + \tilde{F}\tilde{\lambda}_k + H\phi(x_k) + \tilde{c}, \qquad (9)$$
$$0 \leq \tilde{\lambda}_k \perp \tilde{y}_k \geq 0,$$

where $x_k \in \mathbb{R}^{n_x}$ is the state, $\tilde{\lambda}_k \in \mathbb{R}^{n_{\tilde{\lambda}}}$ is the complementarity variable, $\phi(x) \in \mathbb{R}^{n_\phi}$ is a ReLU neural network as in (4) with n_t neurons. Notice that (9) is not in the A-LCS form. Using Lemma 2, we can transform (9) into an A-LCS in a higher dimensional space. To see this, observe that (9) is equivalent to

$$x_{k+1} = Ax_k + B(\bar{D}\bar{\lambda}_k + \bar{z}) + \tilde{D}\tilde{\lambda}_k + \tilde{z},$$
$$\tilde{y}_k = \tilde{E}x_k + \tilde{F}\tilde{\lambda}_k + H(\bar{D}\bar{\lambda}_k + \bar{z}) + \tilde{c},$$
$$\bar{y}_k = \bar{E}x_k + \bar{F}\bar{\lambda}_k + \bar{c},$$
$$0 \leq \tilde{\lambda}_k \perp \tilde{y}_k \geq 0,$$
$$0 \leq \bar{\lambda}_k \perp \bar{y}_k \geq 0,$$

after direct application of Lemma 2 where $\bar{\lambda} \in \mathbb{R}^{n_t}$. We can write it succinctly as

$$x_{k+1} = Ax_k + D\lambda_k + z,$$
$$y_k = Ex_k + F\lambda_k + c, \qquad (10)$$
$$0 \leq \lambda_k \perp y_k \geq 0,$$

where $\lambda_k = \begin{bmatrix} \tilde{\lambda}_k \\ \bar{\lambda}_k \end{bmatrix}$, $y_k = \begin{bmatrix} \tilde{y}_k \\ \bar{y}_k \end{bmatrix}$, $D = \begin{bmatrix} \tilde{D} & B\bar{D} \end{bmatrix}$, $E = \begin{bmatrix} \tilde{E} \\ \bar{E} \end{bmatrix}$, $F = \begin{bmatrix} \tilde{F} & H\bar{D} \\ 0 & \bar{F} \end{bmatrix}$, $c = \begin{bmatrix} \tilde{c} + H\bar{z} \\ \bar{c} \end{bmatrix}$, and $z = B\bar{z} + \tilde{z}$. Here, the size of $x_k \in \mathbb{R}^{n_x}$ does not change, but notice that now $\lambda_k \in \mathbb{R}^{n_\lambda}$ where $n_\lambda = n_t + n_{\tilde{\lambda}}$.

Using controllers of the form (4), we exclusively consider the linear complementarity system model (10) for notational compactness.

Similarly, one can consider the scenario where both the system dynamics and the controller are represented by ReLU neural networks as in $x_{k+1} = \phi_1(x_k) + B\phi_2(x_k)$, where ϕ_1 represents the autonomous part of the dynamics (obtained by, for example, system identification) and ϕ_2 is the controller. Using Lemma 2, this system has an equivalent A-LCS representation similar to (10), but the details are omitted for brevity.

After obtaining an A-LCS representation of the closed-loop system, one can directly use the existing tools for stability analysis of complementarity systems, such as Lyapunov functions [9] and semidefinite programming [2]. We elaborate on this in the next section.

4 STABILITY ANALYSIS OF THE CLOSED-LOOP SYSTEM

In this section, we provide sufficient conditions for stability in the sense of Lyapunov for an A-LCS. Then, we show that the stability verification problem is equivalent to finding a feasible solution to a set of linear matrix inequalities (LMI's). To begin, consider the following Lyapunov function candidate that was introduced in [9]:

$$V(x_k, \lambda_k) = \begin{bmatrix} x_k \\ \lambda_k \\ 1 \end{bmatrix}^\top \underbrace{\begin{bmatrix} P & Q & h_1 \\ Q^\top & R & h_2 \\ h_1^T & h_2^T & h_3 \end{bmatrix}}_{:=M} \begin{bmatrix} x_k \\ \lambda_k \\ 1 \end{bmatrix}, \quad (11)$$

where $P \in \mathbb{S}^{n_x}$, $Q \in \mathbb{R}^{n_x \times n_\lambda}$, $R \in \mathbb{S}^{n_\lambda}$, $h_1 \in \mathbb{R}^{n_x}$, $h_2 \in \mathbb{R}^{n_\lambda}$, and $h_3 \in \mathbb{R}$ are to be chosen. Note that if F in (10) is a P-matrix, then λ_k is a piecewise affine function of x_k, implying that the Lyapunov function (11) is quadratic in the pair (x_k, λ_k) but it is *piecewise quadratic* (PWQ) in the state x_k. If F is not a P-matrix, then V can be set valued since there can be multiple λ_k's corresponding to each x_k. In either case, V reduces to a common quadratic Lyapunov function in the special case $Q = R = 0$. Therefore, (11) is more expressive than a common quadratic Lyapunov function.

In the following theorem, we construct sufficient conditions for the stability of (10), using the Lyapunov function (11). This is the discrete time version of the results in [9].

THEOREM 2. *Consider the A-LCS in* (10) *with the equilibrium $x_e = 0$, the Lyapunov function* (11) *and a domain $X \subseteq \mathbb{R}^n$. If there exist $M \in \mathbb{S}^{n_x+n_\lambda+1}$, $\alpha_1 > 0$, and $\alpha_2 > \alpha_3 \geq 0$ such that*

$$\alpha_1 ||x_k||_2^2 \leq V(x_k, \lambda_k) \leq \alpha_2 ||x_k||_2^2, \ \forall (x_k, \lambda_k) \in \Gamma_1,$$

$$V(x_{k+1}, \lambda_{k+1}) - V(x_k, \lambda_k) \leq -\alpha_3 ||x_k||_2^2, \ \forall (x_k, \lambda_k, \lambda_{k+1}) \in \Gamma_2,$$

where $x_{k+1} = Ax_k + D\lambda_k + z$ and

$$\Gamma_1 = \{(x_k, \lambda_k) : \ 0 \leq \lambda_k \perp Ex_k + F\lambda_k + c \geq 0, \ x_k \in X\},$$

$$\Gamma_2 = \{(x_k, \lambda_k, \lambda_{k+1}) : 0 \leq \lambda_k \perp Ex_k + F\lambda_k + c \geq 0,$$
$$0 \leq \lambda_{k+1} \perp Ex_{k+1} + F\lambda_{k+1} + c \geq 0, \ x_k \in X\}.$$

Then the equilibrium is Lyapunov stable if $\alpha_3 = 0$ and geometrically stable if $\alpha_2 > \alpha_3 > 0$.

PROOF. Observe that for all (x_k, λ_k):

$$\alpha_1 ||x_k||_2^2 \leq V(x_k, \lambda_k) \leq V(x_0, \lambda_0) \leq \alpha_2 ||x_0||_2^2,$$

and Lyapunov stability follows. For geometric stability, notice that Lyapunov decrease condition is equivalent to $V(x_{k+1}, \lambda_{k+1}) - \gamma V(x_k, \lambda_k) \leq 0$, for some $\gamma \in (0, 1)$. Then

$$\alpha_1 ||x_k||_2^2 \leq V(x_k, \lambda_k) \leq \gamma^k V(x_0, \lambda_0) \leq \alpha_2 \gamma^k ||x_0||_2^2.$$

The result follows. □

Note that we do not require M in (11) to be positive definite to satisfy the requirements of Theorem 2. In light of this theorem, we must solve the following feasibility problem to verify that if the equilibrium of the closed-loop system (10) is stable on X:

$$\text{find} \quad P, Q, R, h_1, h_2, h_3, \alpha_1, \alpha_2, \alpha_3 \quad (12)$$

$$\text{s.t.} \quad \alpha_1 ||x_k||_2^2 \leq V(x_k, \lambda_k) \leq \alpha_2 ||x_k||_2^2, \text{ for } (x_k, \lambda_k) \in \Gamma_1,$$

$$\Delta V \leq -\alpha_3 ||x_k||_2^2, \text{ for } (x_k, \lambda_k, \lambda_{k+1}) \in \Gamma_2,$$

where $\Delta V = V(x_{k+1}, \lambda_{k+1}) - V(x_k, \lambda_k)$. In the following proposition, we turn (12) with $X = \mathbb{R}^n$ into an LMI feasibility problem using the S-procedure [6].

PROPOSITION 1. *The following LMI's solve* (12) *with $X = \mathbb{R}^n$:*

$$T_1 - S_1^T W_1 S_1 - \frac{1}{2}(S_{3,1} + S_{3,1}^\top) \geq 0, \quad (13a)$$

$$T_2 + S_1^\top W_2 S_1 + \frac{1}{2}(S_{3,2} + S_{3,2}^\top) \leq 0, \quad (13b)$$

$$T_3 + S_2^T W_3 S_2 + S_5^T W_4 S_5 + \frac{1}{2}[(S_4 + S_4^\top) + (S_6 + S_6^\top)] \geq 0, \quad (13c)$$

where $G_1 = D^T Pz + D^T h_1 - h_2$, $G_2 = z^T PD + h_1^T D - h_2$, $G_3 = z^T Q + h_2^T$, $S_1 = \begin{bmatrix} E & F & c \\ 0 & I & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $S_2 = \begin{bmatrix} E & F & 0 & c \\ 0 & I & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, $S_{3,i} = \begin{bmatrix} 0 & 0 & 0 \\ J_i E & J_i F & J_i c \\ 0 & 0 & 0 \end{bmatrix}$,

$S_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ J_3 E & J_3 F & 0 & J_3 c \\ 0 & 0 & 0 & 0 \end{bmatrix}$, $S_5 = \begin{bmatrix} EA & ED & Fc & Ecz + c \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$,

$S_6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ J_4 EA & J_4 ED & J_4 F & J_4 Ez + c \\ 0 & 0 & 0 & 0 \end{bmatrix}$, $T_1 = \begin{bmatrix} P - \alpha_1 I & Q & h_1 \\ Q^\top & R & h_2 \\ h_1^T & h_2^T & h_3 \end{bmatrix}$,

$$T_2 = \begin{bmatrix} P - \alpha_2 I & Q & h_1 \\ Q^\top & R & h_2 \\ h_1^T & h_2^T & h_3 \end{bmatrix},$$

$$T_3 = - \begin{bmatrix} A^T PA - P + \alpha_3 I & A^T PD - Q & A^T Q & A^T Pz - h_1 \\ D^T PA - Q^T & D^T PD - R & D^T Q & G_1 \\ Q^T A & Q^T D & R & Q^T z + h_2 \\ z^T PA - h_1^T & G_2 & G_3 & z^T Pz - h_1^T z \end{bmatrix}.$$

Here, W_i are decision variables with non-negative entries, and $J_i = \text{diag}(\tau_i)$ where $\tau_i \in \mathbb{R}^m$ are free decision variables.

PROOF. First define $e_k^\top = \begin{bmatrix} x_k^\top & \lambda_k^\top & 1 \end{bmatrix}$. By left and right multiplying both sides of (13a) by $e_k^\top$ and e_k, respectively, we obtain

$$V(x_k, \lambda_k) - \alpha_1 ||x_k||_2^2 \geq \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix}^\top W_1 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix} + 2\lambda_k^\top \text{diag}(\tau_1) y_k$$

The right hand side is non-negative due to the complementarity constraint $0 \leq \lambda_k \perp y_k \geq 0$. Similarly, by left and right multiplying

both sides of (13b) by $e_k^\top$ and e_k, respectively, we obtain

$$\alpha_2 \|x_k\|_2^2 - V(x_k, \lambda_k) \geq \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix}^\top W_2 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix} + 2\lambda_k^\top \operatorname{diag}(\tau_2) y_k$$

Again, the right hand side is non-negative due to the complementarity constraint $0 \leq \lambda_k \perp y_k \geq 0$.

Now, we define $p_k^\top = \begin{bmatrix} x_k^\top & \lambda_k^\top & \lambda_{k+1}^\top & 1 \end{bmatrix}$. Notice that if we left and right multiply both sides of (13c) by $p_k^\top$ and p_k, we obtain

$$-\Delta V - \alpha_3 \|x_k\|_2^2 \geq \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix}^\top W_3 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix} + \begin{bmatrix} y_{k+1} \\ \lambda_{k+1} \\ 1 \end{bmatrix}^\top W_4 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix}$$
$$+ 2\lambda_k^\top \operatorname{diag}(\tau_3) y_k + 2\lambda_{k+1}^\top \operatorname{diag}(\tau_4) y_{k+1}$$

Similarly, all the terms on the right hand side are non-negative since $0 \leq \lambda_k \perp y_k \geq 0$ for all k. This concludes the proof. □

Notice that (12) captures the non-smooth structure of the LCS combined with the ReLU neural network controller. In addition to that, we can assign a different quadratic function to each polyhedral partition that is created by the neural network without enumerating those partitions by exploiting the complementarity structure of the neural network. Observe that (13a), (13b) are LMI's of size $(n_x + n_\lambda + 1)$, and (13c) is an LMI of size $(n_x + 2n_\lambda + 1)$.

Note that Theorem 13 is a global result for $\mathcal{X} = \mathbb{R}^n$. We can adapt the theorem to bounded regions $\mathcal{X}$ containing the origin.

REMARK 1. *For the equilibrium $x_e = 0$, the region of attraction is defined as*
$$\mathcal{R} = \{x_0 : \lim_{k \to \infty} \|x_k\| = 0\}.$$
If one adds (to the left side) $+\eta L$ to (13c) where
$$L = \begin{bmatrix} -P & -Q & 0 & h_1 \\ -Q^T & R & 0 & -h_2 \\ 0 & 0 & 0 & 0 \\ -h_1^T & -h_2^T & 0 & \xi - h_3 \end{bmatrix},$$
and η is a non-negative scalar variable, then the closed-loop system is geometrically stable and the sub-level set
$$\mathcal{V}_\xi = \{x : V(x, \lambda) \leq \xi \; \forall (x, \lambda) \in \Gamma_1\},$$
is an approximation of the ROA, i.e., $\mathcal{V}_\xi \subseteq \mathcal{R}$. To see this, note that the resulting matrix inequality would imply
$$\alpha_1 \|x_k\|_2^2 \leq V(x_k, \lambda_k) \leq \alpha_2 \|x_k\|_2^2,$$
$$V(x_{k+1}, \lambda_{k+1}) - V(x_k, \lambda_k) + \alpha_3 \|x_k\|_2^2 + \eta(\xi - V(x_k, \lambda_k)) \leq 0.$$
From the second inequality, if $V(x_k, \lambda_k) \leq \xi$, then for some $\gamma \in (0, 1)$ we have $V(x_{k+1}, \lambda_{k+1}) \leq \gamma V(x_k, \lambda_k) \leq \xi$. By induction, if $V(x_0, \gamma_0) \leq \xi$, then $\alpha_1 \|x_k\|_2^2 \leq V(x_k, \gamma_k) \leq \gamma^k V(x_0, \gamma_0) \leq \gamma^k \alpha_2 \|x_0\|_2^2$.

REMARK 2. *In order to prove the Lyapunov conditions over the ellipsoid $\mathcal{X} = \{x : x^T N x \leq \xi\}$, one can add (to the left side) $-\beta_1 N_1$ to (13a), $+\beta_2 N_1$ to (13b) and $+\beta_3 N_2$ to (13c) where*
$$N_1 = \begin{bmatrix} -N & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \xi \end{bmatrix}, N_2 = \begin{bmatrix} -N & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \xi \end{bmatrix},$$
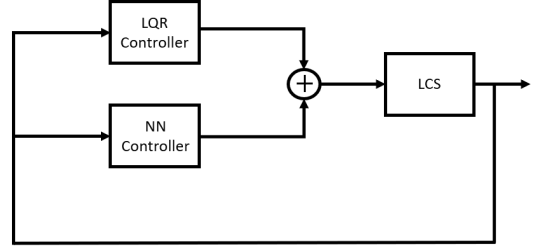*and β_i are non-negative scalar variables.*

Figure 2: Block diagram of the closed-loop system.

5 EXAMPLES

We use YALMIP [32] toolbox with MOSEK [35] to formulate and solve the linear matrix inequality feasibility problems. PATH [15] has been used to solve the linear complementarity problems when simulating the system. PyTorch is used for training neural network controllers [38]. The experiments are done on a desktop computer with the processor Intel *i7-9700* and *16GB RAM* unless stated otherwise. For all of the experiments, we consider the closed-loop system in Figure 2 and the linear-quadratic regulator controller is designed with state penalty matrix $Q^{\text{LQR}} = 10I$ and input penalty matrix $R^{\text{LQR}} = I$ unless stated otherwise. Now, we introduce the mixed-integer problem for (2) that connects the complementarity constraints into equivalent big-M mixed integer constraints:

$$\min_{x_k, \lambda_k, u_k} \quad \sum_{k=0}^{N-1} x_k^T Q^{\text{OPT}} x_k + u_k^T R^{\text{OPT}} u_k + x_k^T Q_N^{\text{OPT}} x_k$$
$$\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k + D\lambda_k + z,$$
$$M_1 s_k \geq Ex_k + F\lambda_k + Hu_k + c \geq 0, \quad (14)$$
$$M_2(\mathbf{1} - s_k) \geq \lambda_k \geq 0,$$
$$s_k \in \{0, 1\}^m, x_0 = x(0),$$

where $\mathbf{1}$ is a vector of ones, and M_1, M_2 are scalars that are used for the big M method. Moving forward, we will consider function $\pi_{\text{OPT}}(x(0))$ that returns the first element of the optimal input sequence, u_0^*, for a given $x(0)$ and learn this function using a neural network $\phi(x)$. The code for all examples is available[1].
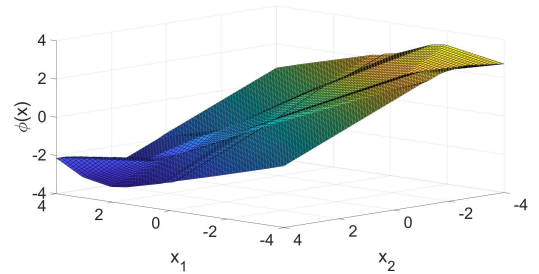
---

[1]https://github.com/AlpAydinoglu/sverification

Figure 3: Neural network (ϕ) policy for the double-integrator example.
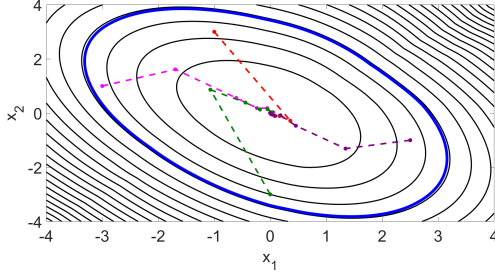
Figure 4: Sublevel sets of the piece-wise quadratic Lyapunov function $V(x_k, \lambda_k)$ with four different trajectories for the double integrator example. A sublevel set that lies in the constraint set is shown in blue.

5.1 Double Integrator

In this example, we consider a double integrator model:

$$x_{k+1} = Ax_k + Bu_k,$$

where $\tilde{A} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$, and $A = \tilde{A} + BK_{\text{LQR}}$, where LQR gains are $Q^{\text{LQR}} = 0.1I$ and $R^{\text{LQR}} = 1$. This simple model serves as an example where we approximate an explicit model predictive controller (explicit MPC) [4] using a neural network and verify the stability of the resulting system. We consider the state and input constraints:

$$\mathcal{X} = \{x : \begin{bmatrix} -4 \\ -4 \end{bmatrix} \leq x \leq \begin{bmatrix} 4 \\ 4 \end{bmatrix}\}, \ \mathcal{U} = \{u : -3 \leq u \leq 3\},$$

and obtain 2000 samples of the form $(x, \pi_{\text{MPC}}(x))$ with $N = 10$, $Q^{\text{MPC}} = 10I$, and $R^{\text{MPC}} = 1$ where Q^{MPC} is the penalty on the state, R^{MPC} is the penalty on the input and $\pi_{\text{MPC}}(x)$ is the first element of the optimal input sequence for a given state x [5]. Next we approximate the explicit MPC controller using a ReLU network $\phi(x)$ with two layers and 10 neurons in each layer as in Figure 3. Now, consider the closed-loop system:

$$x_{k+1} = Ax_k + B\phi(x_k). \tag{15}$$

First, we find the equivalent LCP representation of $\phi(x)$ using Lemma 2. Then, we write the equivalent LCS representation of (15)
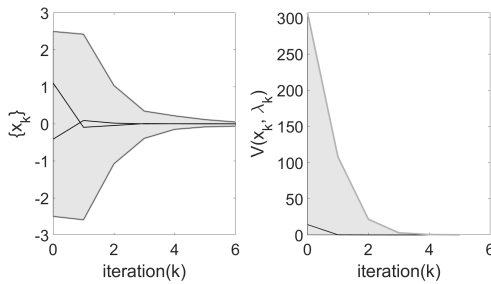

Figure 5: Envelopes for 1000 trajectories and their corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the double integrator example.
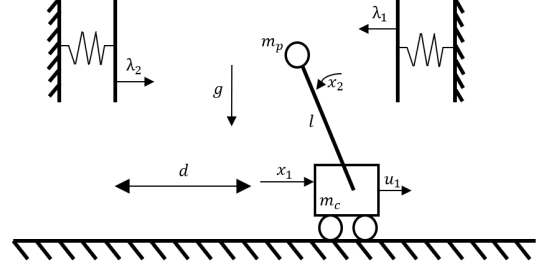

Figure 6: Regulation task of a cart-pole system exploiting contact with the soft walls.

as described in Section 3.2. We computed the piece-wise quadratic Lyapunov function of the form (11) and verified exponential stability in 0.81 seconds. The sublevel sets of the Lyapunov functions are plotted in Figure 4 and the envelopes of 1000 trajectories with their corresponding Lyapunov functions in Figure 5.

5.2 Cart-pole with Soft Walls

We consider the regulation problem of a cart-pole with soft-walls as in Figure 6. This problem has been studied in [3, 14, 33] and is a benchmark in contact-based control algorithms. In this model, x_1 represents the position of the cart, x_2 represents the angle of the pole and x_3, x_4 are their time derivatives respectively. Here, λ_1 and λ_2 represent the contact force applied by the soft walls to the pole from the right and left walls, respectively. We consider the linearized model around $x_2 = 0$:

$$\dot{x}_1 = x_3,$$
$$\dot{x}_2 = x_4,$$
$$\dot{x}_3 = g\frac{m_p}{m_c}x_2 + \frac{1}{m_c}u_1,$$
$$\dot{x}_4 = \frac{g(m_c + m_p)}{lm_c}x_2 + \frac{1}{lm_c}u_1 + \frac{1}{lm_p}\lambda_1 - \frac{1}{lm_p}\lambda_2,$$
$$0 \leq \lambda_1 \perp lx_2 - x_1 + \frac{1}{k_1}\lambda_1 + d \geq 0,$$
$$0 \leq \lambda_2 \perp x_1 - lx_2 + \frac{1}{k_2}\lambda_2 + d \geq 0,$$

where $m_c = 1$ is the mass of the cart, $m_p = 1$ is the mass of the pole, $l = 1$ is the length of the pole, $k_1 = k_2 = 1$ are the stiffness parameter of the walls, $d = 1$ is the distance between the origin and the soft walls. Then, we discretize the dynamics using the explicit Euler method with time step $T_s = 0.1$ to obtain the system matrices:

$$\tilde{A} = \begin{bmatrix} 1 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0.981 & 1 & 0 \\ 0 & 1.962 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ 0.1 \\ 0.1 \end{bmatrix}, \tilde{D} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0.1 & -0.1 \end{bmatrix}, \tilde{E} =$$

$$\begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix}, \tilde{F} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tilde{c} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, A = \tilde{A} + BK_{LQR} \text{ and,}$$

K_{LQR} is the gain of the linear-quadratic regulator that stabilizes the linear system $(\tilde{A}, B)$. However, the equilibrium $x_e = 0$ is not globally stable due to the soft walls. We solve the optimal control
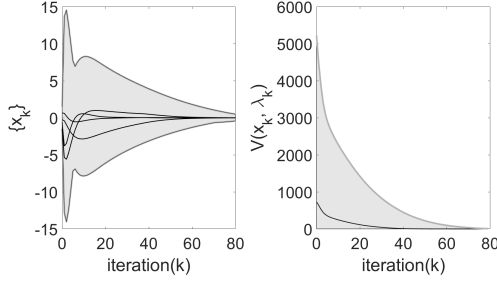
Figure 7: Envelopes for 1000 trajectories and the corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the cart-pole example.

problem (14) with $N = 10$, $Q^{\text{OPT}} = 10I$, $R^{\text{OPT}} = 1$, and Q_N^{OPT} as the solution of the discrete algebraic Riccati equation to generate samples of the form $(x, \pi_{\text{OPT}}(x))$. For this particular problem, we generate 4000 samples and we train a neural network $\phi(x)$ with two layers, each with 10 neurons, to approximate the optimal controller π_{OPT} and we used the ADAM optimizer to do the training. Then, we analyze the linear complementarity system with the neural network controller $u_k = \phi(x_k)$. Following the procedure in Section 3, we first express the neural network as a linear complementarity problem using Lemma 2 and then transform the LCS with the NN controller into the form (10). We compute a Lyapunov function of the form (11) in 1.61 seconds that verifies that the closed-loop system with the neural network controller $\phi(x)$ is globally exponentially stable. For this example, a common Lyapunov function is enough to verify stability. In Figure 7, we present the envelopes for 1000 trajectories.

5.3　Box with Friction

In this example, we consider the regulation task of a box on a surface as in Figure 8. This simple model serves as an example where the contact forces λ_k are not unique due to Coulomb friction between the surface and the box. Here, x_1 is the position of the cart, x_2 is the velocity of the cart, u is the input applied to the cart, $g = 9.81$ is the gravitational acceleration, $m = 1$ is the mass of the cart, and $\mu = 0.1$ is the coefficient of friction between the cart and the surface. The system can be modeled by:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + \tilde{D}\tilde{\lambda}_k, \\ 0 &\le \tilde{\lambda}_k \perp \tilde{E}x_k + \tilde{F}\tilde{\lambda}_k + Hu_k + \tilde{c} \ge 0, \end{aligned} \tag{16}$$
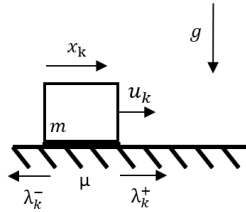

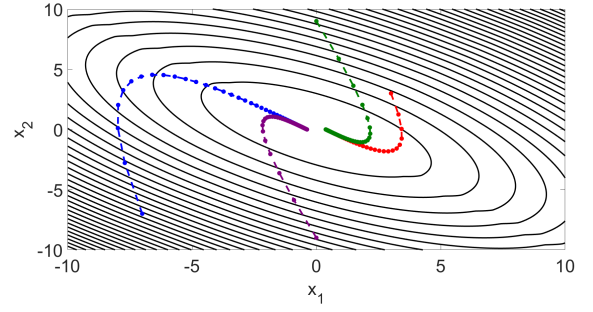**Figure 8: Regulation task of a box on a surface with friction.**

Figure 9: Sublevel sets of the piece-wise quadratic Lyapunov function $V(x_k, \lambda_k)$ with four different trajectories for the box with friction example.

where $\tilde{A} = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix}$, $\tilde{D} = \begin{bmatrix} 0 & 0 & 0 \\ 0.1 & -0.1 & 0 \end{bmatrix}$, $\bar{E} = \begin{bmatrix} 0 & 1 \\ 0 & -1 \\ 0 & 0 \end{bmatrix}$, $\tilde{F} = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & 1 \\ -1 & -1 & 0 \end{bmatrix}$, $\tilde{c} = \begin{bmatrix} 0 \\ 0 \\ 0.981 \end{bmatrix}$, $H = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$, $\tilde{E} = \bar{E} + HK_{LQR}$, $A = \tilde{A} + BK_{LQR}$ and, K_{LQR} is the gain that (the linear-quadratic regulator controller) stabilizes the linear system $(\tilde{A}, B)$. Observe that the matrix $\tilde{F}$ is not a P-matrix, hence for a given x_k, the contact forces λ_k are not unique. Similar to the previous example, we generate 2000 samples $(x, \pi_{\text{OPT}}(x))$ for the LCS in (16) with $N = 5$, $Q^{\text{OPT}} = Q_N^{\text{OPT}} = 0.1$, $R^{\text{OPT}} = 1$ and train a neural network $\phi(x)$ that approximates the optimal controller. Then we convert the system in (16) with the neural network controller $\phi(x)$ into the form (10). Next, we compute the piece-wise quadratic Lyapunov function (with sublevel sets shown in Figure 9) of the form (11) in 1.05 seconds such that the exponential stability condition is verified outside a ball around the origin, $\mathcal{D} = \{x : ||x||_2^2 > 0.6\}$. More precisely, we prove convergence to a set (smallest sublevel set of V that contains $\mathcal{D}$) which contains the equilibrium. This is expected because the trajectories do not reach the origin due to stiction. We demonstrate the envelopes for 1000 trajectories and their respective Lyapunov functions in Figure 10.
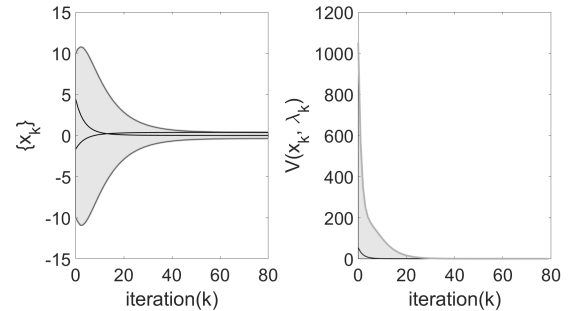

Figure 10: Envelopes for 1000 trajectories and the corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the box with friction example.
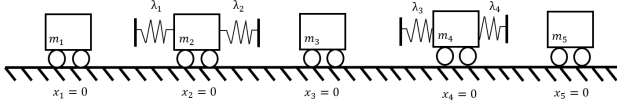
Figure 11: Regulation task of five carts to their respective origins.

5.4 Five Carts

We consider the regulation task of five carts as in Figure 11. Here x_i describes the state of the i-th cart, the interaction between the carts is modeled by soft springs represented by λ_i, and all carts can be controlled via the applied force u_i. We approximate Newtons's second law with a force balance equation and obtain the following quasi-static model:

$$x_{k+1}^{(1)} = x_k^{(1)} + u_k^{(1)} - \lambda_k^{(1)},$$
$$x_{k+1}^{(i)} = x_k^{(i)} + u_k^{(i)} - \lambda_k^{(i-1)} - \lambda_k^{(i)}, \text{ for } i = 2, 3, 4,$$
$$x_{k+1}^{(5)} = x_k^{(5)} + u_k^{(5)} + \lambda_k^{(4)},$$
$$0 \le \lambda_k^{(i)} \perp x_k^{(i+1)} - x_k^{(i)} + \lambda_k^{(i)} \ge 0.$$

We designed an LQR controller with with state penalty matrix $Q^{\text{LQR}} = I$ and input penalty matrix $R^{\text{LQR}} = I$. Then, we solve the optimal control problem (14) with $N = 10$, $Q^{\text{OPT}} = Q_N^{\text{OPT}} = 10I$, and $R^{\text{OPT}} = 1$ to generate 2000 samples of the form $(x, \pi_{\text{OPT}}(x))$. Using these samples, we train $\phi(x)$ with two layers of size 10 and express the neural network as a linear complementarity problem using Lemma 2.

We compute a piece-wise quadratic Lyapunov function of the form (11) in 1.63 seconds (sub-level sets as in Figure 12) that verifies that the closed-loop system with the neural network controller $\phi(x)$ is globally exponentially stable outside a ball $\mathcal{D} = \{x : ||x||_2^2 > 0.1\}$. We also verified that there is not a common Lyapunov function that satisfies the LMI's in (13). We note that a common Lyapunov function that satisfies Theorem 2 might exist, but no such function satisfies our relaxation in (13). On the other hand, this demonstrates the importance of searching over a wider class of functions. In Figure 13, we present the envelopes for 1000 trajectories and the
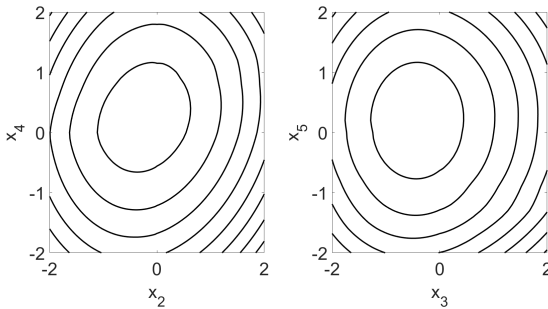

Figure 12: Sublevel sets of the piece-wise quadratic Lyapunov function for the five carts example on the planes $\mathcal{P}_1 = \{x : x_1 = x_3 = x_5 = 0\}$ and $\mathcal{P}_2 = \{x : x_1 = x_2 = x_4 = 0\}$ respectively.
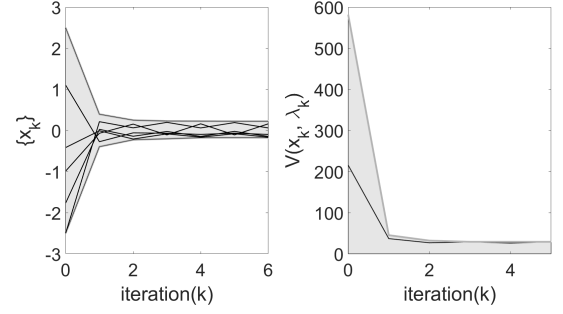

Figure 13: Envelopes for 1000 trajectories and the corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the five carts example.

corresponding Lyapunov functions. We note that *memory* is the limiting factor in terms of scalability of our method and present scalability tests in Table 1.

6 CONCLUSION AND FUTURE WORK

In this work, we have shown that neural networks with ReLU activation functions have an equivalent linear complementarity problem representation. Furthermore, we have shown that a linear complementarity system with a ReLU neural network controller can be transformed into an LCS with a higher dimensional complementarity variable than the original system. This allows one to use the existing literature on linear complementarity systems when analyzing an LCS with NN controller with ReLU activations.

Towards this direction, we have derived the discrete-time version of the stability results in [9] and shown that searching for a Lyapunov function for an LCS with ReLU NN controller is equivalent to finding a feasible solution to a set of linear matrix inequalities. The proposed method exploits the complementarity structure of both the system and the NN controller and avoids enumerating the exponential number of potential modes. We have also demonstrated the effectiveness of our method on numerical examples, including a difference inclusion model.

As future work, we are planning to explore tools from algebraic geometry that use samples instead of the S-procedure terms which result in a stronger relaxation [12]. Also, we consider using passivity results [34] in order to develop algorithms that can verify the stability for larger neural networks. At last, it is of interest to learn stabilizing neural network controllers utilizing the complementarity viewpoint.

RAM	Number of neurons	Solve time
8GB RAM	20	2.1 seconds
8GB RAM	60	194.72 seconds
8GB RAM	100	OOM
16GB RAM	100	1364.78 seconds
16GB RAM	140	OOM

Table 1: Scalability tests.

ACKNOWLEDGMENTS

REFERENCES

[1] Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. *arXiv preprint arXiv:1703.00443*, 2017.
[2] Alp Aydinoglu, Victor M Preciado, and Michael Posa. Contact-aware controller design for complementarity systems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1525–1531. IEEE, 2020.
[3] Alp Aydinoglu, Victor M Preciado, and Michael Posa. Stabilization of complementarity systems via contact-aware controllers. *arXiv preprint arXiv:2008.02104*, 2020.
[4] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.
[5] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.
[6] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, 1994.
[7] Bernard Brogliato. Some perspectives on the analysis and control of complementarity systems. *IEEE Transactions on Automatic Control*, 48(6):918–935, 2003.
[8] Bernard Brogliato and B Brogliato. *Nonsmooth mechanics*. Springer, 1999.
[9] M Kanat Camlibel, Jong-Shi Pang, and Jinglai Shen. Lyapunov stability of complementarity and extended systems. *SIAM Journal on Optimization*, 17(4):1056–1101, 2007.
[10] Shaoru Chen, Mahyar Fazlyab, Manfred Morari, George J Pappas, and Victor M Preciado. Learning lyapunov functions for piecewise affine systems with neural network controllers. *arXiv preprint arXiv:2008.06546*, 2020.
[11] Steven Chen, Kelsey Saulnier, Nikolay Atanasov, Daniel D Lee, Vijay Kumar, George J Pappas, and Manfred Morari. Approximating explicit model predictive control using constrained neural networks. In *2018 Annual American control conference (ACC)*, pages 1520–1527. IEEE, 2018.
[12] Diego Cifuentes and Pablo A Parrilo. Sampling algebraic varieties for sum of squares programs. *SIAM Journal on Optimization*, 27(4):2381–2404, 2017.
[13] Richard W Cottle, Jong-Shi Pang, and Richard E Stone. *The linear complementarity problem*. SIAM, 2009.
[14] Robin Deits, Twan Koolen, and Russ Tedrake. Lvis: Learning from value function intervals for contact-aware robot controllers. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7762–7768. IEEE, 2019.
[15] Steven P Dirkse and Michael C Ferris. The path solver: a nommonotone stabilization scheme for mixed complementarity problems. *Optimization methods and software*, 5(2):123–156, 1995.
[16] Asen Dontchev and Frank Lempio. Difference methods for differential inclusions: a survey. *SIAM review*, 34(2):263–294, 1992.
[17] Souradeep Dutta, Xin Chen, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. Sherlock-a tool for verification of neural network feedback systems: demo abstract. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, pages 262–263, 2019.
[18] Jiameng Fan, Chao Huang, Xin Chen, Wenchao Li, and Qi Zhu. Reachnn*: A tool for reachability analysis of neural-network controlled systems. In *International Symposium on Automated Technology for Verification and Analysis*, pages 537–542. Springer, 2020.
[19] Mahyar Fazlyab, Manfred Morari, and George J Pappas. Probabilistic verification and reachability analysis of neural networks via semidefinite programming. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2726–2731. IEEE, 2019.
[20] Mahyar Fazlyab, Manfred Morari, and George J Pappas. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *arXiv preprint arXiv:1903.01287*, 2019.
[21] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 11427–11438, 2019.
[22] Daniel Goeleven and Bernard Brogliato. Stability and instability matrices for linear evolution variational inequalities. *IEEE Transactions on Automatic Control*, 49(4):521–534, 2004.
[23] Tuomas Haarnoja, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine. Learning to walk via deep reinforcement learning. *arXiv preprint arXiv:1812.11103*, 2018.
[24] Mathew Halm and Michael Posa. A quasi-static model and simulation approach for pushing, grasping, and jamming. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 491–507. Springer, 2018.

[25] Wilhemus PMH Heemels, Bart De Schutter, and Alberto Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.
[26] WPMH Heemels, Johannes M Schumacher, and S Weiland. Linear complementarity systems. *SIAM journal on applied mathematics*, 60(4):1234–1269, 2000.
[27] Michael Hertneck, Johannes Köhler, Sebastian Trimpe, and Frank Allgöwer. Learning an approximate model predictive controller with guarantees. *IEEE Control Systems Letters*, 2(3):543–548, 2018.
[28] Luigi Iannelli, Raffaele Iervolino, and Francesco Vasca. Linear complementarity systems and cone-copositive lyapunov stability. *IEEE Control Systems Letters*, 3(4):799–804, 2019.
[29] Benjamin Karg and Sergio Lucia. Efficient representation and approximation of model predictive control laws via deep learning. *IEEE Transactions on Cybernetics*, 50(9):3866–3878, 2020.
[30] Benjamin Karg and Sergio Lucia. Stability and feasibility of neural network-based controllers via output range analysis. *arXiv preprint arXiv:2004.00521*, 2020.
[31] Hassan K Khalil and Jessy W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.
[32] Johan Lofberg. Yalmip: A toolbox for modeling and optimization in matlab. In *2004 IEEE international conference on robotics and automation (IEEE Cat. No. 04CH37508)*, pages 284–289. IEEE, 2004.
[33] Tobia Marcucci and Russ Tedrake. Warm start of mixed-integer programs for model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 2020.
[34] Felix A Miranda-Villatoro, Fulvio Forni, and Rodolphe Sepulchre. Dominance analysis of linear complementarity systems. *arXiv preprint arXiv:1802.00284*, 2018.
[35] APS Mosek. The mosek optimization software. *Online at http://www. mosek. com*, 54(2-1):5, 2010.
[36] Brendan O'Donoghue. Operator splitting for a homogeneous embedding of the monotone linear complementarity problem. *arXiv preprint arXiv:2004.02177*, 2020.
[37] Thomas Parisini and Riccardo Zoppoli. A receding-horizon regulator for nonlinear systems and a neural approximation. *Automatica*, 31(10):1443–1451, 1995.
[38] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.
[39] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014.
[40] Michael Posa, Mark Tobenkin, and Russ Tedrake. Stability analysis and control of rigid-body systems with impacts and friction. *IEEE Transactions on Automatic Control*, 61(6):1423–1437, 2015.
[41] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.
[42] Stefan Scholtes. *Introduction to piecewise differentiable equations*. Springer Science & Business Media, 2012.
[43] Chelsea Sidrane and Mykel J Kochenderfer. Overt: Verification of nonlinear dynamical systems with neural network controllers via overapproximation. In *Safe Machine Learning workshop at ICLR*, 2019.
[44] Georgi V Smirnov. *Introduction to the theory of differential inclusions*, volume 41. American Mathematical Soc., 2002.
[45] David E Stewart and Jeffrey C Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering*, 39(15):2673–2691, 1996.
[46] Aneel Tanwani, Bernard Brogliato, and Christophe Prieur. Well-posedness and output regulation for implicit time-varying evolution variational inequalities. *SIAM Journal on Control and Optimization*, 56(2):751–781, 2018.
[47] Hoang-Dung Tran, Xiaodong Yang, Diego Manzanas Lopez, Patrick Musau, Luan Viet Nguyen, Weiming Xiang, Stanley Bak, and Taylor T Johnson. Nnv: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *International Conference on Computer Aided Verification*, pages 3–17. Springer, 2020.
[48] Zhaoming Xie, Glen Berseth, Patrick Clary, Jonathan Hurst, and Michiel van de Panne. Feedback control for cassie with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1241–1246. IEEE, 2018.
[49] He Yin, Peter Seiler, and Murat Arcak. Stability analysis using quadratic constraints for systems with neural network controllers. *arXiv preprint arXiv:2006.07579*, 2020.
[50] Xiaojing Zhang, Monimoy Bujarbaruah, and Francesco Borrelli. Safe and near-optimal policy learning for model predictive control using primal-dual neural networks. In *2019 American Control Conference (ACC)*, pages 354–359. IEEE, 2019.
[51] Henry Zhu, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar. Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3651–3657. IEEE, 2019.