Optimal Control of Deep Equilibrium Models

Anton Xue and Susmit Jha

Abstract

We study the application of convex optimization techniques to discrete-time open-loop control where the plant is parametrized by an *implicit model*. This is motivated by the observations that implicit models can effectively learn dynamical systems, and that their parameter size is often small. This then enables the use of optimal control techniques of querying a solver to generate the control signal over a receding horizon. We give a formulation of the inherently non-convex control problem, for which we present both linear and semidefinite relaxations. Moreover, we show that if the model satisfies certain *incremental quadratic constraints*, then the convex optimization problem can be augmented with sampled trajectory data.

1 Introduction

Using neural networks [1] for system identification [2, 3] is appealing for several reasons. First, one does not need to give a first-principles derivation of the internal dynamics — rather, it suffices to use sampled data to learn and imitate the underlying behavior — which is especially useful if the system is complex. Second, deep learning architectures are especially flexible, allowing engineers to rapidly iterate and improve the model design and training. And perhaps most importantly: this approach is effective. Although neural network-based system identification has existed since the 90's [4], the past decade of acceleration in computational power has made a deep learning-based data-driven model design not only feasible, but also practical [5, 6, 7, 8]. In this paper we study the natural question after system identification: given a neural network model of a dynamical system, how does one control it?

There are three critical challenges in the application of control for learned models: expressivity, stability, and scalability. *Expressivity* concerns the class of functions the learned model can represent, which is especially important if the underlying system is complex. It is well known that neural networks are universal function approximators [9] and therefore prime candidates for learning dynamics [1]. *Stability* refers to the robustness of the model against small perturbations, and is a central focus of control theory [10]. Model stability is often desirable in practice even if the underlying system may be unstable [11, 12, 13], as unstable models tend to fragile against data perturbations and may also quickly become useless in a control loop [14]. *Scalability* involves the computational feasibility and practicality of solving the control problem, which may be an issue if the model is too large, and is especially the case if one wishes to use large models in control loops or embedded systems.

Our starting point is the recently proposed Recurrent Equilibrium Network (REN) [15], an *implicit model* based on the Deep Equilibrium (DEQ) [16] architecture. These implicit models are so-called because their evaluation requires a fixpoint computation, with which they achieve remarkable performance on tasks ranging from optical flow [17] to learning dynamics [18, 19]. Importantly, DEQs are also universal approximators [16, Theorem 2] as they can represent traditional feedforward architectures, and therefore capture an *expressive* class of functions. The key extension in [15] is in *stability* by construction: RENs can be parametrized so that incremental quadratic constraints (IQCs) such as contractivity and dissipativity hold no matter how the model is trained. Moreover, implicit models often have parameter sizes that are significantly smaller than other architectures [16, 20], meaning that they are within the *scalability* of computational techniques from optimal control.

In this paper we study the optimal control of REN models. We assume a given REN model is fixed, and our goal is to formulate a convex optimization problem that is queried to a solver. Our contributions are as follows:

[1] There do not yet exist good classification results or bounds on what model sizes and architectures are needed to represent and learn different dynamical systems. Thus, model design is often heuristic-driven.

- We give a full derivation of the IQC-satisfying REN parametrizations presented in [15]. Rather than contractivity, we derive a slightly weaker condition of *incrementally nonexpansive*. These sections are intended as exposition for known results.
- We formulate the target-tracking MPC problem for RENs. This problem is fundamentally non-convex due to the neural network activation functions, and so we present two different convex relaxations: a linear relaxation and a semidefinite relaxation. We consider two cases of activations: when the activation is known to be a ReLU, and when the activation is abstracted as sector-bounds by a finite collection of linear functions.
- We show that if one knows which IQC constraints hold, then the aforementioned optimization problems can be further augmented to include sampled trajectory data.
- We make our implementation open source at `https://github.com/SRI-CSL/TrinityAI/tree/nn-sos`.

1.1 Neural Network-Based System Identification

We give a brief overview on learning and modelling dynamical systems using neural networks, and refer to [8] for an extensive survey. Consider a continuous-time dynamical system

$$\frac{\mathrm{d}x}{\mathrm{d}t}(t) = f(x(t), u(t)), \quad y(t) = g(x(t), u(t)), \tag{1}$$

where $\mathbb{R}_{\geq 0}$ is the nonnegative real numbers, $x : \mathbb{R}_{\geq 0} \to \mathbb{R}^{n_x}$ is the state, $u : \mathbb{R}_{\geq 0} \to \mathbb{R}^{n_u}$ is the control input, and $y : \mathbb{R}_{\geq 0} \to \mathbb{R}^{n_y}$ is the output. One method to solve for the value of $x(t)$ given some control input u is to apply a forward Euler discretization to yield

$$x_{k+1} = x_k + hf(x_k, u_k), \quad x_0 = x(0) \tag{2}$$

where $x_k \approx x(hk) \in \mathbb{R}^{n_x}$ approximates the state at $t = hk$ and $u_k = u(hk)$. The goal in system identification is to find functions f_θ, g_θ such that $f_\theta \approx f$ and $g_\theta \approx g$, with respect to a parameter $\theta \in \mathbb{R}^{n_\theta}$. In our context, f_θ and g_θ are neural networks — in particularly a REN. Given sampled points $\{x_k^{(i)}, u_k^{(i)}, y_k^{(i)}\}$, one may formulate the learning problem as

$$\underset{\theta}{\text{minimize}} \quad \mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\sum_{k=0}^{T} \left\| x_{k+1}^{(i)} - x_k^{(i)} - hf_\theta(x_k^{(i)}, u_k^{(i)}) \right\|_2^2 + \left\| y_k^{(i)} - g_\theta(x_k^{(i)}, u_k^{(i)}) \right\|_2^2 \tag{3}$$

2 Background

We present the formulation of the recurrent equilibrium network (REN) architecture [15], which is an instance of a deep equilibrium model (DEQ) [16]. We then discuss the notions of incremental quadratic constraints (IQCs) such as nonexpansiveness and QSR dissipativity. We also formulate the non-convex target-tracking control problem.

2.1 Recurrent Equilibrium Networks

Consider the learned dynamics model of (2),

$$x_{k+1} = x_k + hf_\theta(x_k, u_k), \quad y_k = g_\theta(x_k, u_k),$$

where f_θ, g_θ are an REN [2] that is defined by the following set of equations,

$$\partial x_k = W_x z_k + A_x x_k + B_x u_k + b_x \tag{4.1}$$
$$v_k = W_v z_k + A_v x_k + B_v u_k + b_v, \quad z_k = \sigma(v_k) \tag{4.2}$$
$$y_k = W_y z_k + A_y x_k + B_y u_k + b_y \tag{4.3}$$

[2] The primary differnce between the formulation of RENs here and that of [15] is that (4) models f_θ, while [15, Section 3] models the 1-step discrete-time dynamics. Although these two formulations are equivalent, we found it easier to learn dynamics with (4).

where in particular $f_\theta(x_k, u_k) = \partial x_k$ and $g_\theta(x_k, u_k) = y_k$. Here $x_k \in \mathbb{R}^{n_k}$ is the state, $u_k \in \mathbb{R}^{n_u}$ is the control input, $z_k \in \mathbb{R}^{n_z}$ is the hidden layer, and $y_k \in \mathbb{R}^{n_y}$ is the output, and $\sigma : \mathbb{R} \to \mathbb{R}$ is an activation function applied coordinate-wise to vectors. Let θ collectively represent the parameters $W_{(\cdot)}, A_{(\cdot)}, B_{(\cdot)}, b_{(\cdot)}$.

Note that RENs are an implicit model because (4.2) is recursive in z_k. We would like an REN to be *well-posed*: for z_k to exist and be unique for any given x_k, u_k. The sufficient conditions and computation of fixpoint solutions are therefore of central importance to implicit models, and several results presented in the literature:

- [21] shows that if the Perron-Frobenius eigenvalue of $|W_v|$, the element-wise absolute value of W_v, is < 1, then a unique solution can be found via iteration.
- [22] shows that if $I - W_v$ is *strongly monotone* (i.e. $2I - (W_v + W_v^\top) \succeq mI$ for some $m > 0$) [3], and if the activation σ acts as the proximal operator [23] of a closed convex proper function, then this is a sufficient condition. Moreover, operator splitting methods such as Peaceman-Rachford splitting may be applied.
- [24] relaxes the condition of [22] to require the existence of a strictly positive diagonal matrix Λ such that $2\Lambda - \Lambda W_v - W_v^\top \Lambda \succ 0$. This is the same condition later leveraged in [15], and that we use in this paper.

A key contribution of [24] is to derive a general sufficient condition for fixpoint solutions using only the slopes of the activations. We say that an (activation) function $\sigma : \mathbb{R} \to \mathbb{R}$ is $[0, 1]$-*sector bounded* if

$$0 \leq \frac{\sigma(v) - \sigma(v')}{v - v'} \leq 1, \quad \text{for all } v, v' \in \mathbb{R}.$$

This condition is satisfied by a number of popular activation functions, e.g. ReLU, leaky ReLU, tanh, sigmoid, and has been widely used in the analysis of neural networks. This inequality can be exploited to give tight Lipschitz constants [25], verify quadratic safety properties [26, 27], and importantly for RENs: a condition for when z_k is *well-posed* (exists and unique) in the fixpoint equations (4.2).

Lemma 1. *Consider* (4.2) *and suppose that* σ *is* $[0, 1]$*-sector bounded. If there exists a diagonal matrix* $\Lambda \succ 0$ *such that* $2\Lambda - \Lambda W_v - W_v^\top \Lambda \succ 0$*, then* z_k *is* well-posed *for any* x_k, u_k*.*

Proof. This follows from [24, Theorem 1]. □

The $[0, 1]$-sector bound condition is an incremental constraint because it captures a relation between two different trajectories. More generally for two z_k, v_k and z'_k, v'_k for which we can define their difference $\Delta z_k = z_k - z'_k$ and $\Delta v_k = v_k - v'_k$, for which any diagonal $\Lambda \succ 0$ will satisfy the useful inequality

$$\begin{bmatrix} \Delta z_k \\ \Delta v_k \end{bmatrix}^\top \begin{bmatrix} -2\Lambda & \Lambda \\ \Lambda & 0 \end{bmatrix} \begin{bmatrix} \Delta z_k \\ \Delta v_k \end{bmatrix} \geq 0. \tag{5}$$

Moreover, we can define the difference dynamics of (4) as

$$\Delta \partial x_k = W_x \Delta z_k + A_x \Delta x_k + B_x \Delta u_k, \quad \Delta x_{k+1} = \Delta x_k + h \Delta \partial x_k \tag{6.1}$$
$$\Delta v_k = W_v \Delta z_k + A_v \Delta x_k + B_v \Delta u_k \tag{6.2}$$
$$\Delta y_k = W_y \Delta z_k + A_y \Delta x_k + B_y \Delta u_k \tag{6.3}$$

2.1.1 Sector Bounded via Linear Functions

The notion of sector-boundedness earlier assumed that a function is zero at the origin. A more general definition can be given. We say that a function $\sigma : \mathbb{R} \to \mathbb{R}$ is sector-bounded by the linear functions $a, b : \mathbb{R} \to \mathbb{R}$ if

$$(\sigma(x) - a(x))(\sigma(x) - b(x)) \leq 0, \quad \text{for all } x \in \mathbb{R}.$$

This extension is relevant as it lets us to bound activations like tanh using sectors that may not be origin-centered.

[3] We write $A \succ 0$ (resp. $A \succeq 0$) to mean that a symmetric matrix A is positive definite (resp. positive semidefinite). Some authors such as [22] extend definiteness to nonsymmetric matrices via symmetrization, e.g. $B \succ 0$ iff $B + B^\top \succ 0$.

2.2 Incremental Quadratic Constraints

Stability is of central importance to control theory. Many different notions of stability exist [10], but usually a system is considered stable if it is robust against small perturbations: slight changes to the initial condition do not lead to drastically diverging trajectories. In this work we investigate two kinds of stability through the perspective of incremental quadratic constraints (IQCs) [28]: incremental nonexpansiveness, and incremental QSR dissipative.

2.2.1 Incrementally Nonexpansive

We say that the dynamical system (1) is *incrementally nonexpansive* if there exists a storage function $V_\Delta : \mathbb{R}^{n_x \times n_x} \to \mathbb{R}_{\geq 0}$ with $V_\Delta(x(t), x(t)) = 0$ for all t, such that for any two trajectories x, x' with the same input $\tilde{u}$,

$$V_\Delta(x(t_1), x'(t_1)) \leq V_\Delta(x(t_0), x'(t_0)) \tag{7}$$

Here $V_\Delta(x(t), x'(t))$ can be interpreted as the relative energies between trajectories x, x' at time t. Incremental nonexpansive therefore means that under the same control input, the relative energy of two trajectories do not increase. A sufficient condition is to find a V_Δ of form

$$V_\Delta(x, x') = (x - x')^\top P(x - x'), \quad P \succ 0.$$

2.2.2 Incrementally QSR Dissipative

Another form of incremental stability is known as *incremental QSR dissipativity* [29], which asserts that the relative "energy" between two trajectories cannot increase without external inputs. We say that the dynamical system (1) is *incrementally dissipative* with respect to the supply function $S_\Delta : \mathbb{R}^{n_u \times n_u \times n_y \times n_y} \to \mathbb{R}$ if there exists a storage function $V_\Delta : \mathbb{R}^{n_x \times n_x} \to \mathbb{R}_{\geq 0}$ with $V_\Delta(x(t), x(t)) = 0$ for all t, such that for any two trajectories $(x, u, y), (x', u', y')$,

$$V_\Delta(x(t_1), x'(t_1)) \leq V_\Delta(x(t_0), x'(t_0)) + \int_{t_0}^{t_1} S_\Delta(u(t), u'(t), y(t), y'(t))\, \mathrm{d}t\,, \quad \text{for all } t_0, t_1 \in \mathbb{R}_{\geq 0} \text{ with } t_0 \leq t_1. \tag{8}$$

Similar to incremental nonexpansiveness, $V_\Delta(x(t), x'(t))$ can be interpreted as the relative energy between between the trajectories x, x' at time t, while S_Δ measures the relative power in-flow to the systems. Many common cases for S_Δ are quadratic, and are therefore parametrizable by a triplet of matrices (Q, S, R) via

$$S_\Delta(u, u', y, y') = \begin{bmatrix} y - y' \\ u - u' \end{bmatrix}^\top \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \begin{bmatrix} y - y' \\ u - u' \end{bmatrix}, \tag{9}$$

where Q, R are symmetric, in which case we use the term *incrementally QSR dissipative*. Moreover, it is common to restrict the search for V_Δ to a positive-definite quadratic function, such that V_Δ takes form

$$V_\Delta(x, x') = (x - x')^\top P(x - x'), \quad P \succ 0.$$

Note that (8) extends to the discrete-time systems (2) case when the integral is replaced with a summation.

2.3 Open Loop Model Predictive Control

For an initial state x_0 and reference trajectory $x_0^\star, x_1^\star, \ldots, x_T^\star$ over horizon T, the target-tracking MPC problem is

$$\begin{aligned}
\underset{z,x,u}{\text{minimize}} \quad & J(x^\star, x, u) \coloneqq \frac{1}{2}\sum_{k=0}^{T}(x_k^\star - x_k)^\top J_x(x_k^\star - x_k) + u_k^\top J_u u_k && (10.1)\\
\text{subject to} \quad & x_{k+1} = x_k + h\partial x_k && (10.2)\\
& \partial x_k = W_x z_k + A_x x_k + B_x u_k + b_x && (10.3)\\
& v_k = W_v z_k + A_v x_k + B_v u_k + b_v && (10.4)\\
& z_k = \sigma(v_k) && (10.5)
\end{aligned}$$

where J is the objective function to be minimized, and the cost matrices $J_x, J_u \succ 0$ are given. The intent of this formulation is that a solver is to be run periodically in a loop to generate new control inputs.

3 Incrementally Nonexpansive Parametrizations of RENs

We first present a convex parametirzation of nonexpansive RENs in Section 3.1: we state a condition for well-posedness and incrementally nonexpansive as a linear matrix inequality (LMI) in the model weights θ, such that any θ which induces a feasible LMI will have the desired properties. Then in Section 3.2 we give a free (unconstrained) parametrization to a subset of the feasible LMI solutions: we show how to extract a well-posed and incrementally nonexpansive model from any semidefinite matrix obeying a specific form.

Together with Section 4, this is a more complete derivation of what is shown in [15, Section 4, Section 5]. As our formulation of RENs are different but equivalent, the derivation is likewise similar. A minor difference, however, is that [15] enforces contractiveness, which is a slightly stronger condition that nonexpansiveness.

3.1 Convex Parametrization

To show incrementally non-expansive it suffices to find a Lyapunov-like $P \succ 0$ such that

$$\Delta x_{k+1}^\top P \Delta x_{k+1} \leq \Delta x_k^\top P \Delta x_k, \tag{11}$$

and using (5) a sufficient condition

$$\Delta x_k^\top P \Delta x_k - \Delta x_{k+1}^\top P \Delta x_{k+1} \geq \begin{bmatrix}\Delta z_k \\ \Delta v_k\end{bmatrix}^\top \begin{bmatrix}-2\Lambda & \Lambda \\ \Lambda & 0\end{bmatrix} \begin{bmatrix}\Delta z_k \\ \Delta v_k\end{bmatrix}, \tag{12}$$

where $\Delta u_k = 0$ for analyzing nonexpansiveness. The RHS expands as

$$\begin{bmatrix}\Delta z_k \\ \Delta v_k\end{bmatrix}^\top \begin{bmatrix}-2\Lambda & \Lambda \\ \Lambda & 0\end{bmatrix} \begin{bmatrix}\Delta z_k \\ \Delta v_k\end{bmatrix} = \begin{bmatrix}\Delta z_k \\ \Delta x_k\end{bmatrix}^\top \begin{bmatrix}-2\Lambda + \Lambda W_v + W_v^\top \Lambda & \Lambda A_v \\ A_v^\top \Lambda & 0\end{bmatrix} \begin{bmatrix}\Delta z_k \\ \Delta x_k\end{bmatrix}$$

A sufficient condition for (12) is therefore

$$\begin{bmatrix}0 \\ I\end{bmatrix} P \begin{bmatrix}0 \\ I\end{bmatrix}^\top - \begin{bmatrix}hW_x^\top \\ I + hA_x^\top\end{bmatrix} P \begin{bmatrix}hW_x^\top \\ I + hA_x^\top\end{bmatrix}^\top \succeq \begin{bmatrix}-2\Lambda + \Lambda W_v + W_v^\top \Lambda & \Lambda A_v \\ A_v^\top \Lambda & 0\end{bmatrix},$$

where (12) is recovered by multiplying both sides with $\begin{bmatrix}\Delta z_k^\top & \Delta x_k^\top\end{bmatrix}^\top$. Rearranging this we get

$$\begin{bmatrix}2\Lambda - \Lambda W_v - W_v^\top \Lambda & -\Lambda A_v \\ -A_v^\top \Lambda & P\end{bmatrix} - \begin{bmatrix}hW_x^\top \\ I + hA_x^\top\end{bmatrix} P \begin{bmatrix}hW_x^\top \\ I + hA_x^\top\end{bmatrix}^\top \succeq 0 \tag{13}$$

Let us introduce the variable substitutions

$$P = E^\top \mathcal{P}^{-1} E, \quad \mathcal{W}_v = \Lambda W_v, \quad \mathcal{A}_v = \Lambda A_v, \quad \mathcal{W}_x = EhW_x, \quad \mathcal{A}_x = E(I + hA_x),$$

where E is invertible. Then (13) is rearranged as

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v \\ -\mathcal{A}_v^\top & E^\top \mathcal{P}^{-1} E \end{bmatrix} - \begin{bmatrix} \mathcal{W}_x^\top \\ \mathcal{A}_x^\top \end{bmatrix} \mathcal{P}^{-1} \begin{bmatrix} \mathcal{W}_x^\top \\ \mathcal{A}_x^\top \end{bmatrix}^\top \succeq 0,$$

and using the inequality $E^\top \mathcal{P}^{-1} E \succeq E + E^\top - \mathcal{P}$ from [30, 7.2.P17], a sufficient condition is

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} \end{bmatrix} - \begin{bmatrix} \mathcal{W}_x^\top \\ \mathcal{A}_x^\top \end{bmatrix} \mathcal{P}^{-1} \begin{bmatrix} \mathcal{W}_x^\top \\ \mathcal{A}_x^\top \end{bmatrix}^\top \succeq 0,$$

and since $\mathcal{P} \succ 0$, using the Schur complement this is equivalent to

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v & \mathcal{W}_x^\top \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & \mathcal{A}_x^\top \\ \mathcal{W}_x & \mathcal{A}_x & \mathcal{P} \end{bmatrix} \succeq 0. \tag{14}$$

Call the variables of (14) by

$$\omega = (\mathcal{W}_x, \mathcal{A}_x, \mathcal{W}_v, \mathcal{A}_v, \Lambda, \mathcal{P}, E),$$

where the convex parametrization can be described by the set

$$\Omega = \big\{\omega : \mathcal{P}, \Lambda \succ 0, \ \ \Lambda \text{ diagonal}, \ \ 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top \succ 0, \ \ (14) \text{ holds}\big\}.$$

Theorem 1. *A model* (4) *with parameters from* $\omega \in \Omega$ *is well-posed and incrementally nonexpansive.*

Proof. For well-posed, by Lemma (1) the condition $2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top \succ 0$ suffices to ensure well-posedness. For incrementally nonexpansive, observe that (14) is a sufficient condition for (12). □

3.2 Free Parametrization

We use a Burer-Monteiro style parametrization [31] for (14), which has form

$$M = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12}^\top & M_{22} & M_{23} \\ M_{13}^\top & M_{23}^\top & M_{33} \end{bmatrix} = X^\top X + \begin{bmatrix} \varepsilon I & & \\ & 0 & \\ & & \varepsilon I \end{bmatrix}, \tag{15}$$

where the εI in the appropriate components ensures positive definiteness. From this we get

$$\mathcal{W}_x = M_{13}^\top, \quad \mathcal{A}_x = M_{23}^\top, \quad \mathcal{A}_v = -M_{12}, \quad \mathcal{P} = M_{33},$$

and note that E can be described as

$$E = \frac{1}{2}\big(M_{22} + \mathcal{P} + Y - Y^\top\big), \quad Y \in \mathbb{R}^{n_x \times n_x} \text{ is free.}$$

Let us parametrize $\Lambda = \operatorname{diag}(e^{\lambda_1}, \ldots, e^{\lambda_{n_z}})$ where $\lambda \in \mathbb{R}^{n_z}$, then

$$\mathcal{W}_v = \Lambda - \frac{1}{2}\big(M_{11} + Z - Z^\top\big), \quad Z \in \mathbb{R}^{n_z \times n_z} \text{ is free.}$$

The free parameters are therefore:

$$\theta = (W_y, A_y, B_y, B_x, B_v, X, Y, Z, \lambda).$$

4 Incrementally QSR Dissipative Parametrizations of RENs

We give parametrizations to enforce that the REN (4) is well-posed and incrementally QSR dissipative.

4.1 Convex Parametrization

Suppose that Q, S, R is given and consider any two trajectories (x, u, y) and (x', u', y') of the REN (4). A sufficient condition to ensure incremental QSR dissipativity is if there exists $P \succ 0$ such that

$$\Delta x_{k+1}^\top P \Delta x_{k+1} \leq \Delta x_k^\top P \Delta x_k + \begin{bmatrix} \Delta y_k \\ \Delta u_k \end{bmatrix}^\top \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \begin{bmatrix} \Delta y_k \\ \Delta u_k \end{bmatrix}, \tag{16}$$

and using (5), a sufficient condition for (16) to hold is therefore

$$\Delta x_k^\top P \Delta x_k + \begin{bmatrix} \Delta y_k \\ \Delta u_k \end{bmatrix}^\top \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \begin{bmatrix} \Delta y_k \\ \Delta u_k \end{bmatrix} - \Delta x_{k+1}^\top P \Delta x_{k+1} \geq \begin{bmatrix} \Delta z_k \\ \Delta v_k \end{bmatrix}^\top \begin{bmatrix} -2\Lambda & \Lambda \\ \Lambda & 0 \end{bmatrix} \begin{bmatrix} \Delta z_k \\ \Delta v_k \end{bmatrix}, \tag{17}$$

where in particular the RHS expands as

$$\begin{bmatrix} \Delta z_k \\ \Delta v_k \end{bmatrix}^\top \begin{bmatrix} -2\Lambda & \Lambda \\ \Lambda & 0 \end{bmatrix} \begin{bmatrix} \Delta z_k \\ \Delta v_k \end{bmatrix} = \begin{bmatrix} \Delta z_t \\ \Delta x_t \\ \Delta u_t \end{bmatrix}^\top \begin{bmatrix} -2\Lambda + \Lambda W_v + W_v^\top \Lambda & \Lambda A_v & \Lambda B_v \\ A_v^\top \Lambda & 0 & 0 \\ B_v^\top \Lambda & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta z_t \\ \Delta x_t \\ \Delta u_t \end{bmatrix},$$

and the QSR term expands as

$$\begin{bmatrix} \Delta y_k \\ \Delta u_k \end{bmatrix}^\top \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \begin{bmatrix} \Delta y_k \\ \Delta u_k \end{bmatrix} = \begin{bmatrix} \Delta z_t \\ \Delta x_t \\ \Delta u_t \end{bmatrix}^\top \left(\begin{bmatrix} 0 & 0 & W_y^\top S \\ 0 & 0 & A_y^\top S \\ S^\top W_y & S^\top A_y & B_y^\top S + S^\top B_y + R \end{bmatrix} + \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix} Q \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix}^\top \right) \begin{bmatrix} \Delta z_t \\ \Delta x_t \\ \Delta u_t \end{bmatrix}.$$

The task is to construct a parametrization of (17) that is convex in its variables (REN parameters, multipliers, and possible slack variables). To begin, a sufficient condition for (17) is

$$\begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix} P \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}^\top + \begin{bmatrix} 0 & 0 & W_y^\top S \\ 0 & 0 & A_y^\top S \\ S^\top W_y & S^\top A_y & B_y^\top S + S^\top B_y + R \end{bmatrix} + \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix} Q \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix}^\top - \begin{bmatrix} hW_x^\top \\ I + hA_x^\top \\ hB_x^\top \end{bmatrix} P \begin{bmatrix} hW_x^\top \\ I + hA_x^\top \\ hB_x^\top \end{bmatrix}^\top$$
$$\succeq \begin{bmatrix} -2\Lambda + \Lambda W_v + W_v^\top \Lambda & \Lambda A_v & \Lambda B_v \\ A_v^\top \Lambda & 0 & 0 \\ B_v^\top \Lambda & 0 & 0 \end{bmatrix}$$

where (17) is recovered by multiplying both sides by $\begin{bmatrix} \Delta z_k^\top & \Delta x_k^\top & \Delta u_k^\top \end{bmatrix}^\top$. Rearranging this we get

$$\begin{bmatrix} 2\Lambda - \Lambda W_v - W_v^\top \Lambda & -\Lambda A_v & W_y^\top S - \Lambda B_v \\ -A_v^\top \Lambda & P & A_y^\top S \\ S^\top W_y - B_v^\top \Lambda & S^\top A_y & B_y^\top S + S^\top B_y + R \end{bmatrix} + \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} Q \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix}^\top - \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} P \begin{bmatrix} hW_x^\top \\ I + hA_x^\top \\ hB_x^\top \end{bmatrix}^\top \succeq 0. \tag{18}$$

However, quadratic expressions such as ΛA_v means that (18) does not describe a convex set of (θ, P, Λ), where θ is the collective REN parameters and $P, \Lambda \succ 0$ with Λ diagonal. To alleviate this we introduce the variables

$$P = E^\top \mathcal{P}^{-1} E, \quad \mathcal{W}_v = \Lambda W_v, \quad \mathcal{A}_v = \Lambda A_v, \quad \mathcal{B}_v = \Lambda B_v, \quad \mathcal{W}_x = EhW_x, \quad \mathcal{A}_x = E(I + hA_x), \quad \mathcal{B}_x = EhB_x,$$

where E is any invertible matrix. Then (18) is equivalent to

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v & W_y^\top S - \mathcal{B}_v \\ -\mathcal{A}_v^\top & E^\top \mathcal{P}^{-1} E & A_y^\top S \\ S^\top W_y - \mathcal{B}_v^\top & S^\top A_y & B_y^\top S + S^\top B_y + R \end{bmatrix} + \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} Q \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix}^\top - \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} \mathcal{P}^{-1} \begin{bmatrix} \mathcal{W}_x^\top \\ \mathcal{A}_x^\top \\ \mathcal{B}_x^\top \end{bmatrix}^\top \succeq 0$$

Using the inequality $E^\top \mathcal{P}^{-1} E \succeq E + E^\top - \mathcal{P}$ from [30, 7.2.P17], we have that a sufficient condition for (18) is

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v & W_y^\top S - \mathcal{B}_v \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & A_y^\top S \\ S^\top W_y - \mathcal{B}_v^\top & S^\top A_y & B_y^\top S + S^\top B_y + R \end{bmatrix} + \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} Q \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix}^\top + \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} \mathcal{P}^{-1} \begin{bmatrix} \mathcal{W}_x^\top \\ \mathcal{A}_x^\top \\ \mathcal{B}_x^\top \end{bmatrix}^\top \succeq 0$$

Moreover, let us decompose $Q = Q_+ - Q_-$ into its positive and negative semidefinite components such that $Q_+, Q_- \succeq 0$. Then a further sufficient condition for (18) is

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v & W_y^\top S - \mathcal{B}_v \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & A_y^\top S \\ S^\top W_y - \mathcal{B}_v^\top & S^\top A_y & B_y^\top S + S^\top B_y + R \end{bmatrix} - \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} Q_- \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \end{bmatrix}^\top + \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} \mathcal{P}^{-1} \begin{bmatrix} \mathcal{W}_x^\top \\ \mathcal{A}_x^\top \\ \mathcal{B}_x^\top \end{bmatrix}^\top \succeq 0,$$

and since $\mathcal{P} \succ 0$, by the Schur complement this is equivalent to

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v & W_y^\top S - \mathcal{B}_v & \mathcal{W}_x^\top \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & A_y^\top S & \mathcal{A}_x^\top \\ S^\top W_y - \mathcal{B}_v^\top & S^\top A_y & B_y^\top S + S^\top B_y + R & \mathcal{B}_x^\top \\ \mathcal{W}_x & \mathcal{A}_x & \mathcal{B}_x & \mathcal{P} \end{bmatrix} - \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \\ 0 \end{bmatrix} Q_- \begin{bmatrix} W_y^\top \\ A_y^\top \\ B_y^\top \\ 0 \end{bmatrix}^\top \succeq 0. \tag{19}$$

Let us denote the variables of (19) by

$$\omega = (\mathcal{W}_x, \mathcal{A}_x, \mathcal{B}_x, \mathcal{W}_v, \mathcal{A}_v, \mathcal{B}_v, W_y, A_y, B_y, \Lambda, \mathcal{P}, E),$$

where to recover the original parameters θ we have

$$hW_x = E^{-1}\mathcal{W}_x, \quad I + hA_x = E^{-1}\mathcal{A}_x, \quad hB_x = E^{-1}\mathcal{B}_x, \quad W_v = \Lambda^{-1}\mathcal{W}_v, \quad A_v = \Lambda^{-1}\mathcal{A}_v, \quad B_v = \Lambda^{-1}\mathcal{B}_v.$$

To see that an invertible E exists provided (19) holds, recall that the block-diagonals of positive semidefinite matrices are positive semidefinite. Furthermore since $Q_- \succeq 0$, we have $E + E^\top - \mathcal{P} \succeq 0$, and because $\mathcal{P} \succ 0$, any such E is therefore invertible. This allows us to form the following convex parametrization of RENs:

$$\Omega = \left\{\omega : \mathcal{P}, \Lambda \succ 0, \ \ \Lambda \text{ diagonal}, \ \ 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top \succ 0, \ \ (19) \text{ holds}\right\} \tag{20}$$

Theorem 2. *A model* (4) *with parameters from* $\omega \in \Omega$ *is well-posed and incrementally QSR dissipative.*

Proof. Any model with $\omega \in \Omega$ is incrementally QSR dissipative since (19) is a sufficient condition of (16). Moreover since $\mathcal{W}_v = \Lambda W_v$, by Lemma 1 the condition $2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top \succ 0$ implies that the model is well-posed. □

4.2 Free Parametrization

In this section we derive a *sufficient* unconstrained parametrization of (19). First, note that if we introduce $\mathcal{Q} = Q_- + \varepsilon_q I$, then a sufficient condition for (19) is achieved by substituting $\mathcal{Q}$ for Q_-. An insight in [15] is to isolate the y-parameters from the x, v-parameters via the Schur complement. We do this by swapping the third and fourth block row and columns of (19) (after $\mathcal{Q}$ substitution) to yield:

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v^\top - \mathcal{W}_v^\top & -\mathcal{A}_v & \mathcal{W}_x^\top & W_y^\top S - \mathcal{B}_v \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & \mathcal{A}_x^\top & A_y^\top S \\ \mathcal{W}_x & \mathcal{A}_x & \mathcal{P} & \mathcal{B}_x \\ S^\top W_y - \mathcal{B}_v^\top & S^\top A_y & \mathcal{B}_x^\top & B_y^\top S + S^\top B_y + R \end{bmatrix} - \begin{bmatrix} W_y^\top \\ A_y^\top \\ 0 \\ B_y^\top \end{bmatrix} \mathcal{Q} \begin{bmatrix} W_y^\top \\ A_y^\top \\ 0 \\ B_y^\top \end{bmatrix}^\top \succeq 0$$

which is equivalent to

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v^\top - \mathcal{W}_v^\top & -\mathcal{A}_v & \mathcal{W}_x^\top & W_y^\top S - \mathcal{B}_v - W_y^\top \mathcal{Q} B_y \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & \mathcal{A}_x^\top & A_y^\top S - A_y^\top \mathcal{Q} B_y \\ \mathcal{W}_x & \mathcal{A}_x & \mathcal{P} & \mathcal{B}_x \\ S^\top W_y - \mathcal{B}_v^\top - B_y^\top \mathcal{Q} W_y & S^\top A_y - B_y^\top \mathcal{Q} A_y & \mathcal{B}_x^\top & B_y^\top S + S^\top B_y + R - B_y^\top \mathcal{Q} B_y \end{bmatrix} - \begin{bmatrix} W_y^\top \\ A_y^\top \\ 0 \\ 0 \end{bmatrix} \mathcal{Q} \begin{bmatrix} W_y^\top \\ A_y^\top \\ 0 \\ 0 \end{bmatrix}^\top \succeq 0,$$

which by the Schur complement is equivalent to

$$\begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v & \mathcal{W}_x^\top \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & \mathcal{A}_x^\top \\ \mathcal{W}_x & \mathcal{A}_x & \mathcal{P} \end{bmatrix} - \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} \mathcal{R}^{-1} \begin{bmatrix} W_y^\top S - \mathcal{B}_v - W_y^\top \mathcal{Q} B_y \\ A_y^\top S - A_y^\top \mathcal{Q} B_y \\ \mathcal{B}_x \end{bmatrix}^\top - \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} \mathcal{Q} \begin{bmatrix} W_y^\top \\ A_y^\top \\ 0 \end{bmatrix}^\top \succeq 0, \quad (21)$$

where $\mathcal{R} = B_y^\top S + S^\top B_y + R - B_y^\top \mathcal{Q} B_y \succ 0$. The idea is that we will treat the $W_y, A_y, B_y, \mathcal{B}_x$ as free variables and use a Burer-Monteiro [31] style of parametrization (i.e. $X^\top X + \varepsilon I$) to stand in for the left term — from which we can derive the remaining parameters of ω. We first find a sufficient free parametrization of B_y, from which we can derive $\mathcal{R}$ because $\mathcal{Q}, R, S$ are given.

4.2.1 Solving for B_y

Consider the expression

$$\mathcal{R} = B_y^\top S + S^\top B_y + R - B_y^\top \mathcal{Q} B_y \succ 0, \quad (22)$$

where $\mathcal{Q}, S, R$ are given (i.e. fixed). Our goal is to find a free parametrization of B_y such that $\mathcal{R}$ is *obviously* positive definite. Let us conjecture that B_y has form $B_y = \mathcal{Q}^{-1} S + F$, where F is variable and to-be-parametrized, in which case (22) becomes

$$\begin{aligned} \mathcal{R} &= (\mathcal{Q}^{-1} S + F)^\top S + S^\top (\mathcal{Q}^{-1} S + F) + R - (\mathcal{Q}^{-1} S + F)^\top \mathcal{Q} (\mathcal{Q}^{-1} S + F) \\ &= S^\top \mathcal{Q}^{-1} S + F^\top S + S^\top \mathcal{Q}^{-1} S + S^\top F + R - (S^\top \mathcal{Q}^{-1} S + S^\top F + F^\top S + F^\top \mathcal{Q} F) \\ &= S^\top \mathcal{Q}^{-1} S + R - F^\top \mathcal{Q} F \end{aligned}$$

We can further simplify this by taking $F = U_Q^{-1} G$, where $\mathcal{Q} = U_Q^\top U_Q$ is a Cholesky factorization of $\mathcal{Q}$ and G is a variable to-be parametrized. Then this reduces to finding G such that

$$S^\top \mathcal{Q}^{-1} S + R - G^\top G \succ 0 \quad (23)$$

Note that these simplifications do not reduce the possible values of B_y due to the relation $B_y = \mathcal{Q}^{-1} S + U_Q^{-1} G$. Suppose that $S^\top \mathcal{Q}^{-1} S + R = U_R^\top U_R$ is a Cholesky factorization, then we may set $G = H U_R$, where H is variable, such that this becomes

$$S^\top \mathcal{Q}^{-1} S + R - G^\top G = U_R^\top (I - H^\top H) U_R \succ 0$$

It then suffices to find a free parametrization of H such that $I - H^\top H \succ 0$. One idea is to take

$$H^\top H = \left((1+\varepsilon) I + H_0^\top H_0\right)^{-1}, \quad H_0 \in \mathbb{R}^{n_u \times n_u},$$

which will ensure that $I \succ H^\top H \succ 0$ by construction. Then $B_y = \mathcal{Q}^{-1} S + U_Q^{-1} H U_R$.

4.2.2 Solving for the Other Parameters

The idea is to rewrite (21) as

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12}^\top & M_{22} & M_{23} \\ M_{13}^\top & M_{23}^\top & M_{33} \end{bmatrix} = X^\top X + \varepsilon I + \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} \mathcal{R}^{-1} \begin{bmatrix} W_y^\top S - \mathcal{B}_v - W_y^\top \mathcal{Q} B_y \\ A_y^\top S - A_y^\top \mathcal{Q} B_y \\ \mathcal{B}_x \end{bmatrix}^\top + \begin{bmatrix} \star \\ \star \\ \star \end{bmatrix} \mathcal{Q} \begin{bmatrix} W_y^\top \\ A_y^\top \\ 0 \end{bmatrix}^\top,$$

where knowing B_y we can compute the value of $\mathcal{R}^{-1}$. Since $W_y, A_y, \mathcal{B}_x, \mathcal{B}_v$ are free parameters, we can compute the value of both the $\mathcal{R}^{-1}$ and $\mathcal{Q}$ terms. The task then is to pattern match the M block matrix as follows,

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{12}^\top & M_{22} & M_{23} \\ M_{13}^\top & M_{23}^\top & M_{33} \end{bmatrix} = \begin{bmatrix} 2\Lambda - \mathcal{W}_v - \mathcal{W}_v^\top & -\mathcal{A}_v & \mathcal{W}_x^\top \\ -\mathcal{A}_v^\top & E + E^\top - \mathcal{P} & \mathcal{A}_x^\top \\ \mathcal{W}_x & \mathcal{A}_x & \mathcal{P} \end{bmatrix}$$

where we see that

$$\mathcal{W}_x = M_{13}^\top, \quad \mathcal{A}_x = M_{23}^\top, \quad \mathcal{A}_v = -M_{12}, \quad \mathcal{P} = M_{33},$$

and where one can write E as

$$E = \frac{1}{2}\big(M_{22} + \mathcal{P} + Y - Y^\top\big), \quad Y \in \mathbb{R}^{n_x \times n_x} \text{ is free.}$$

Also take

$$\mathcal{W}_v = \Lambda - \frac{1}{2}\big(M_{11} + Z - Z^\top\big), \quad Z \in \mathbb{R}^{n_z \times n_z} \text{ is free.}$$

The free parameters are then

$$\theta = (W_y, A_y, B_y, B_x, B_v, X, Y, Z, \lambda).$$

5 Formulation of Control Problems

We now assume that the REN model (4) is given, and turn our attention to formulating convex relaxations of the optimal control problem.

5.1 Linear Relaxation assuming ReLU

Suppose that $\sigma = \mathsf{ReLU}$. One way to relax (10) is to write

$$\begin{aligned}
\underset{z,x,u}{\text{minimize}} \quad & J(x^\star, x, u) && (24.1)\\
\text{subject to} \quad & x_{k+1} = x_k + h\partial x_k && (24.2)\\
& \partial x_k = W_x z_k + A_x x_k + B_x u_k + b_x && (24.3)\\
& z_k \geq 0, \quad z_k = W_v z_k + A_v x_k + B_v u_k + b_v && (24.4)
\end{aligned}$$

The relaxation is that the constraint $z_k = \sigma(v_k)$ in (10.5) is replaced by the linear constraints $z_k \geq v_k$ and $z_k \geq 0$.

5.2 Semidefinite Relaxations assuming ReLU

Suppose that $\sigma = \mathsf{ReLU}$. Another way to relax the activation is to note the relation

$$z_k = \mathsf{ReLU}(v_k) \quad \iff \quad \begin{cases} z_k \geq 0 \\ z_k \geq v_k \\ z_k \odot (z_k - v_k) = 0 \end{cases}, \tag{25}$$

and note that the element-wise product of two vectors satisfies

$$(Avu^\top)_i = \sum_{k=1}^{1} (Av)_{ik}(u^\top)_{ki} = (Av)_i u_i = (u \odot Av)_i \quad \implies \quad u \odot Av = \mathsf{diag}(Avu^\top).$$

Moreover, since $v_k = W_v z_k + A_v x_k + B_v u_k + b_v$, we need a way to induce the quadratic $z_k z_k^\top$, $z_k x_k^\top$, $z_k u_k^\top$ terms in a convex manner. The idea is to use a semidefinite relaxation of form

$$M_k = \begin{bmatrix} (M_k)_{zz} & (M_k)_{zx} & (M_k)_{zu} & z_k \\ (M_k)_{xz} & (M_k)_{zz} & (M_k)_{xu} & x_k \\ (M_k)_{uz} & (M_k)_{ux} & (M_k)_{uu} & u_k \\ z_k^\top & x_k^\top & u_k^\top & 1 \end{bmatrix} \succeq 0, \tag{26}$$

which will give the relaxations

$$(M_k)_{zz} \approx z_k z_k^\top,\ \ (M_k)_{zx} \approx z_k x_k^\top,\ \ (M_k)_{zu} \approx z_k u_k^\top,\ \ (M_k)_{xx} \approx x_k x_k^\top,\ \ (M_k)_{xu} \approx x_k u_k^\top,\ \ (M_k)_{uu} \approx u_k u_k^\top.$$

Let us now consider how to relax the $z_k \odot v_k$ term, which expands as

$$\begin{aligned} z_k \odot v_k &= z_k \odot (W_v z_k + A_v x_k + B_v u_k + b_v) \\ &= \mathsf{diag}(W_v z_k z_k^\top) + \mathsf{diag}(W_v x_k z_k^\top) + \mathsf{diag}(B_v u_k z_k^\top) + z_k \odot b_v \end{aligned}$$

whose relaxation we then define in terms of M_k as

$$\mathcal{R}[z_k \odot v_k] = \mathsf{diag}(W_v (M_k)_{zz}) + \mathsf{diag}(A_v (M_k)_{xz}) + \mathsf{diag}(B_v (M_k)_{uz}) + z_k \odot b_k.$$

Crudely, we intend $\mathcal{R}$ to be an "operator" that *relaxes* the expression inside the brackets $[\cdot]$, and we use this with the same intuition as $\approx$. Using this notation, the ReLU conditions (25) can then be relaxed as

$$z_k = \mathsf{ReLU}(v_k) \quad \approx \quad \begin{cases} z_k \geq 0, \\ z_k \geq v_k = W_v z_k + A_v x_k + B_v u_k + b_v, \\ \mathsf{diag}((M_k)_{zz}) = \mathcal{R}[z_k \odot v_k] \end{cases}$$

The SDP relaxation of the control problem with ReLU activations is then

$$\begin{aligned} \underset{z,x,u,M_k}{\text{minimize}} \quad & J(x^\star, x, u) && (27.1) \\ \text{subject to} \quad & M_k = \begin{bmatrix} (M_k)_{zz} & (M_k)_{zx} & (M_k)_{zu} & z_k \\ (M_k)_{xz} & (M_k)_{zz} & (M_k)_{xu} & x_k \\ (M_k)_{uz} & (M_k)_{ux} & (M_k)_{uu} & u_k \\ z_k^\top & x_k^\top & u_k^\top & 1 \end{bmatrix} \succeq 0 && (27.2) \\ & x_{k+1} = x_k + h\partial x_k && (27.3) \\ & \partial x_k = W_x z_k + A_x x_k + B_x u_k + b_x && (27.4) \\ & z_k \geq 0, \quad z_k \geq W_v z_k + A_v x_k + B_v u_k + b_v, \quad \mathsf{diag}((M_k)_{zz}) = \mathcal{R}[z_k \odot v_k] && (27.5) \end{aligned}$$

5.3 Semidefinite Relaxations assuming Sector Bounds by Linear Functions

Let $(a^{(1)}, b^{(1)}), (a^{(2)}, b^{(2)}), \ldots : \mathbb{R} \to \mathbb{R}$ be a finite collection of linear functions that sector-bound the activation via

$$\Big(z - a^{(i)}(v)\Big)\Big(z - b^{(i)}(v)\Big) \leq 0, \quad \text{for all } v \in \mathbb{R} \text{ and } z = \sigma(v).$$

This framework admits monotone nondecreasing functions like tanh and sigmoid [4]. Our goal is to construct a semidefinite relaxation similar in nature to M_k in (26). Consider any a, b in the sequence, then as a vector-valued inequality this becomes,

$$z \odot z \leq z \odot (a(v) + b(v)) - a(v) \odot b(v). \tag{28}$$

Noting that $a(v) = a_0 + a_1 v$ and $b(v) = b_0 + b_1 v$, where each $a_0, a_1, b_0, b_1 \in \mathbb{R}$, we may then write (28) as

$$z \odot z \leq z \odot \big((a_1 + b_1)v + (a_0 + b_0)\mathbf{1}\big) - \big(a_1 b_1 v \odot v + (a_0 b_1 + a_1 b_0)v + a_0 b_0 \mathbf{1}\big). \tag{29}$$

Let us define M_k analogously as before in (26). We have previously analyzed how the $z_k \odot v_k$ terms expand, and so it remains to analyze the $v_k \odot v_k$ terms and their respective relaxations. In particular,

$$\begin{aligned} v_k \odot v_k &= (W_v z_k + A_v x_k + B_v u_k + b_v) \odot (W_v z_k + A_v x_k + B_v u_k + b_v) \\ &= W_v z_k \odot W_v z_k + A_v x_k \odot A_v x_k + B_v u_k \odot B_v u_k + b_v \odot b_v \\ &\quad + 2W_v z_k \odot A_v x_k + 2W_v z_k \odot B_v u_k + 2W_v z_k \odot b_v \\ &\quad + 2A_v x_k \odot B_v u_k + 2A_v x_k \odot b_v + 2B_v u_k \odot b_v, \end{aligned}$$

[4] This also admits ReLU, in which case the relaxation in Section 5.2 is recommended.

which suggests the relaxation

$$\begin{aligned}\mathcal{R}[v_k \odot v_k] = \mathsf{diag}(W_v(M_k)_{zz}W_v^\top) + \mathsf{diag}(A_v(M_k)_{xx}A_v^\top) + \mathsf{diag}(B_v(M_k)_{uu}B_v^\top) + b_v \odot b_v \\ + 2\mathsf{diag}(W_v(M_k)_{zx}A_v^\top) + 2\mathsf{diag}(W_v(M_k)_{zu}B_v^\top) + 2W_v z_k \odot b_v \\ + 2\mathsf{diag}(A_v(M_k)_{xu}B_v^\top) + 2A_v x_k \odot b_v + 2B_v u_k \odot b_v.\end{aligned}$$

For the sector-bounded condition of $a^{(i)}, b^{(i)}$ on z_k, v_k, we present the following relaxation for the RHS of (29), which we call $S_k^{(i)}$,

$$\mathcal{R}\Big[S_k^{(i)}\Big] = \Big(a_1^{(i)} + b_1^{(i)}\Big)\mathcal{R}[z_k \odot v_k] + \Big(a_0^{(i)} + b_0^{(i)}\Big)z_k - a_1^{(i)}b_1^{(i)}\mathcal{R}[v_k \odot v_k] - \Big(a_0^{(i)}b_1^{(i)} + a_1^{(i)}b_0^{(i)}\Big)v_k - a_0^{(i)}b_0^{(i)}\mathbf{1} \quad (30)$$

The relaxed control problem is then

$$\begin{aligned}
\underset{z,x,u,M_k}{\text{minimize}} \quad & J(x^\star, x, u) && (31.1)\\
\text{subject to} \quad & M_k = \begin{bmatrix} (M_k)_{zz} & (M_k)_{zx} & (M_k)_{zu} & z_k \\ (M_k)_{xz} & (M_k)_{zz} & (M_k)_{xu} & x_k \\ (M_k)_{uz} & (M_k)_{ux} & (M_k)_{uu} & u_k \\ z_k^\top & x_k^\top & u_k^\top & 1 \end{bmatrix} \succeq 0 && (31.2)\\
& x_{k+1} = x_k + h\partial x_k && (31.3)\\
& \partial x_k = W_x z_k + A_x x_k + B_x u_k + b_x && (31.4)\\
& \mathsf{diag}((M_k)_{zz}) \le \mathcal{R}\Big[S_k^{(i)}\Big], \quad i = 1, 2, \dots && (31.5)
\end{aligned}$$

5.3.1 Computationally Efficient Implementations

Part of the formulation for (31) requires the computation of matrices of form $\mathsf{diag}(AMB^\top)$, where M are the optimization variables and A, B are given. This is a computational bottleneck for Julia's mathematical programming library `JuMP.jl`, which must handle symbolic matrix manipulations. Crucially, an observation is that we care only about the diagonal elements of $AMB^\top$, so it is possible to simplify the two matrix-matrix multiplications with symbolic optimization variables. In particular, note that

$$(AMB^\top)_{ii} = \sum_k A_{ik}(MB^\top)_{ki} = \sum_k A_{ik}\sum_l M_{kl}B_{li}^\top = \sum_{k,l} A_{ik}M_{kl}B_{li}^\top = \sum_{k,l} M_{kl}A_{ik}B_{il}$$

In our setting the matrices A, B are the REN parameters and so will remain the same in many computations, while it is M (which stands in for $(M_k)_{zz}, (M_k)_{zx}, \dots$) that will vary for each step of the horizon. The idea is to first compute and cache the terms involving A, B to avoid repeated computations. Let us define a 3-tensor $A \wedge B$ where each element is $(A \wedge B)_{ikl} = A_{ik}B_{il}$, such that

$$(AMB^\top)_{ii} = \sum_{k,l} M_{kl}A_{ik}B_{il} = \sum_{k,l} M_{kl}(A \wedge B)_{ikl}$$

The computation of $A \wedge B$ is relatively fast since we need only operate on known scalar values, and not the symbolic optimization variables in M. For each $i = 1, \dots, n_z$ the slice $(A \wedge B)_i$ is a matrix the same dimension of M, and so computation of the sum can be expressed as an element-wise product followed by summation of all terms, which is relatively fast for `JuMP.jl` to execute. Programmatically in Julia, to create the $(A_v(M_k)_{xu}B_v^\top)_{ii}$ term we would do

```
AB = zeros(nz, nx, nx) # The 3-tensor we are caching
for i in 1:nz; for k in 1:nx; for l in 1:nu; AB[i,k,l] = Av[i,k]*Bv[i,l] end end end
AMkB = zeros(nz) # The result of diag(AMkB)
for i in 1:nz; AMkB[i] = sum(Mkxu.*AB[i,:,:])  end
```

6 Data-Augmented Optimization

IQCs relate different trajectories to each other, which opens the possibility that sampled trajectories of the REN (4) may be used to tighten the relaxations of the previously presented convex problems. For instance, one may first quickly solve and simulate the trajectory of the linear relaxation (24) in order to gain data samples for the more computationally expensive semidefinite relaxations (27) and (31). We study only the case of incremental nonexpansiveness for now, as it is a simpler IQC than incremental QSR dissipativity.

6.1 Assuming Incremental Nonexpansiveness

Suppose that we know the REN (4) satisfies incremental nonexpansiveness, i.e.

$$\Delta x_{k+1}^\top P \Delta x_{k+1} \leq \Delta x_k^\top P \Delta x_k, \quad \text{for any two trajectories } x, u \text{ and } x', u',$$

where the difference dynamics are given in (6) and $P \succ 0$ is known [5]. Also suppose that we have a collection of trajectories $\{\tilde{x}^{(j)}, \tilde{u}^{(j)}\}_{j=1}^N$ sampled from the model (4). The idea is to further constrain the matrix variables M_k in problems (27) and (31) using these sampled data. Let us define

$$\Delta z_k^{(j)} = z_k - \tilde{z}_k^{(j)}, \quad \Delta x_k^{(j)} = x_k - \tilde{x}_k^{(j)}, \quad \Delta u_k^{(j)} = u_k - \tilde{u}_k^{(j)},$$

Sometimes omitting the $(\cdot)^{(j)}$ superscript for conciseness and using the cyclic permutation of traces, the incremental nonexpansive condition is then equivalent to

$$\begin{aligned} \Delta x_{k+1}^\top P \Delta x_{k+1} &\leq \Delta x_k^\top P \Delta x_k \\ \Delta x_k^\top P \Delta x_k + 2h \Delta x_k^\top P \Delta \partial x_k + h^2 \Delta \partial x_k^\top P \Delta \partial x_k &\leq \Delta x_k^\top P \Delta x_k \\ 2\mathsf{tr}\big(P \Delta \partial x_k \Delta x_k^\top\big) + h\mathsf{tr}\big(P \Delta \partial x_k \Delta \partial x_k^\top\big) &\leq 0 \end{aligned} \tag{32}$$

To embed this information into a semidefinite program, we need to find the appropriate semidefinite relaxations of the vector-vector outerproducts for $\Delta \partial x_k \Delta x_k^\top$ and $\Delta \partial x_k \Delta \partial x_k^\top$. In particular, note that

$$\begin{aligned} \Delta z_k^{(j)} \Delta z_k^{(j)\top} &= \Big(z_k - \tilde{z}_k^{(j)}\Big)\Big(z_k - \tilde{z}_k^{(j)}\Big)^\top = z_k z_k^\top - z_k z_k^{(j)\top} - z_k^{(j)} z_k^\top + z_k^{(j)} z_k^{(j)\top}, \\ \Delta z_k^{(j)} \Delta x_k^{(j)\top} &= \Big(z_k - \tilde{z}_k^{(j)}\Big)\Big(x_k - \tilde{x}_k^{(j)}\Big)^\top = z_k x_k^\top - z_k \tilde{x}_k^{(j)\top} - \tilde{z}_k^{(j)} x_k^\top + \tilde{z}_k^{(j)} \tilde{x}_k^{(j)\top}, \end{aligned}$$

and analogously for $\Delta z_k \Delta u_k^\top$, $\Delta x_k \Delta x_k^\top$, $\Delta x_k \Delta u_k^\top$, $\Delta u_k \Delta u_k^\top$. This motivates us to define the relaxations

$$\begin{aligned} (\Delta M_k)_{zz}^{(j)} &= (M_k)_{zz} - z_k \tilde{z}_k^{(j)\top} - \tilde{z}_k^{(j)} z_k^\top + \tilde{z}_k^{(j)} \tilde{z}_k^{(j)\top}, \\ (\Delta M_k)_{zx}^{(j)} &= (M_k)_{zx} - z_k \tilde{x}_k^{(j)\top} - \tilde{z}_k^{(j)} x_k^\top + \tilde{z}_k^{(j)} \tilde{x}_k^{(j)\top}, \end{aligned}$$

and analogously for the other blocks of $\Delta M_k^{(j)}$. Thus, we expand the quadratic terms in (32),

$$\begin{aligned} \Delta \partial x_k^{(j)} \Delta x_k^{(j)\top} &= W_x \Delta z_k \Delta x_k^\top + A_x \Delta x_k \Delta x_k^\top + B_x \Delta u_k \Delta x_k^\top \\ \mathcal{R}\Big[\Delta \partial x_k^{(j)} \Delta x_k^{(j)\top}\Big] &= W_x (\Delta M_k)_{zx}^{(j)} + A_x (\Delta M_k)_{xx}^{(j)} + B_x (\Delta M_k)_{ux}^{(j)} \\ \Delta \partial x_k^{(j)} \Delta \partial x_k^{(j)\top} &= \begin{bmatrix} W_x^\top \\ A_x^\top \\ B_x^\top \end{bmatrix}^\top \begin{bmatrix} \Delta z_k^{(j)} \\ \Delta x_k^{(j)} \\ \Delta u_k^{(j)} \end{bmatrix} \begin{bmatrix} \Delta z_k^{(j)} \\ \Delta x_k^{(j)} \\ \Delta u_k^{(j)} \end{bmatrix}^\top \begin{bmatrix} W_x^\top \\ A_x^\top \\ B_x^\top \end{bmatrix} \\ \mathcal{R}\Big[\Delta \partial x_k^{(j)} \Delta \partial x_k^{(j)\top}\Big] &= \begin{bmatrix} W_x^\top \\ A_x^\top \\ B_x^\top \end{bmatrix}^\top \begin{bmatrix} (\Delta M_k)_{zz}^{(j)} & (\Delta M_k)_{zx}^{(j)} & (\Delta M_k)_{zu}^{(j)} \\ (\Delta M_k)_{xz}^{(j)} & (\Delta M_k)_{xx}^{(j)} & (\Delta M_k)_{xu}^{(j)} \\ (\Delta M_k)_{uz}^{(j)} & (\Delta M_k)_{ux}^{(j)} & (\Delta M_k)_{uu}^{(j)} \end{bmatrix} \begin{bmatrix} W_x^\top \\ A_x^\top \\ B_x^\top \end{bmatrix} \end{aligned}$$

[5] A solution to (14) allows us to recover P.

The derivation in (32) is then relaxed as

$$2\mathsf{tr}\Big(P\mathcal{R}\Big[\Delta\partial x_k^{(j)}\Delta x_k^{(j)\top}\Big]\Big)+h\mathsf{tr}\Big(P\mathcal{R}\Big[\Delta\partial x_k^{(j)}\Delta\partial x_k^{(j)\top}\Big]\Big)\leq 0 \tag{33}$$

That is, (32) is a semidefinite relaxation of the the incrementally nonexpansive constraint in relation to the sampled data $\{\tilde{x}^{(j)},\tilde{u}^{(j)}\}_{j=1}^N$. This can be augmented to both the SDP relaxed ReLU problem (27) and the sector-bounded problem (31). We use the sector-bounded problem to illustrate this formulation:

$$\begin{aligned}
\underset{z,x,u,M_k}{\text{minimize}} \quad & J(x^\star,x,u) && (34.1)\\
\text{subject to} \quad & (31.2),(31.3),(31.4),(31.5), && (34.2)\\
& 2\mathsf{tr}\Big(P\mathcal{R}\Big[\Delta\partial x_k^{(j)}\Delta x_k^{(j)\top}\Big]\Big)+h\mathsf{tr}\Big(P\mathcal{R}\Big[\Delta\partial x_k^{(j)}\Delta\partial x_k^{(j)\top}\Big]\Big)\leq 0,\quad j=1,\dots,N && (34.3)
\end{aligned}$$

7 Experiments

For convention, let us call the dynamics of (2) the *true model* and that of (4) the *learned model* or *REN*. When the REN is subject to additional IQC constraints we will make this explicit, and for instance say: *REN satisfying incremental nonexpansiveness*. There are a number of experiments that are required to supplement the theory presented in this paper.

Solver vs True Loss Take a control signal generated from the solver calls on problems (24), (27), (31), or (34). We wish to compare the *solver loss* derived from the convex problem's optimal value, with the *true loss* from running u on the true system (2).

Additional Sector Bounds and Loss The formulation of (31) permits the use of multiple different linear functions to sector bound the activation. We plan to focus on the tanh function, and examine the solver loss vs the true loss as we more tightly constrain the tanh by linear sectors. Moreover, we would like to investigate the solver runtime as more sectors are added.

Data Augmented Optimization We would like to see whether using trajectories will improve the solver loss vs the true loss. Moreover, we would like to investigate the solver runtime as more trajectories are added. We plan to try this assuming incremental nonexpansiveness first, as this is a simpler IQC.

Hidden Dimension vs Loss It would be interesting to study how large n_z needs to be for the learned model to well-approximate the true model.

7.1 Current Challenges

At the moment we are focused on using the free parametrization for incremental nonexpansiveness based on (14) and (15). The biggest challenge at the moment is in training RENs, though a few other concerns are also present. Below we document some of the challenges encountered so far while trying to learn simple dynamical systems like the pendulum or cartpole.

Training Loss Goes to Infinity Using ReLU as the activation will sometimes make the training loss blow up to infinity. This can be fixed by using tanh instead. We suspect this is because tanh might be easier for the fixpoint solver to handle, but some reference implementation of RENs like [15] seem to have successfully used ReLU. The work of [13] uses tanh. A workaround is to try to reimplement a fixpoint solver, but this may be more work than it is worth if only to fix ReLU.

Training Seems to Stall Training with tanh seems to sometimes stall: a trajectory sampled from the learned model may be quite off from the true model. Beyond using a larger hidden dimension n_z, playing with the values of ε, and experimenting with different learning rates and initial parameter conditions, we are short on ideas. At the moment, a simple feedforward model with a few hundred neurons can get significantly much better loss than our REN model with $n_z \approx 100$. We also do not wish to make n_z too big, as this will affect the size of the semidefinite constraints to be given to the solver.

Large Affine Expressions for JuMP The semidefinite relaxations, if encoded naively, will induce large affine expressions for JuMP, Julia's mathematical programming interface to solvers. These may slow down solver times, but can usually be resolved with appropriate algebraic rewriting of constraints.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[2] Lennart Ljung. System identification. In *Signal analysis and prediction*, pages 163–173. Springer, 1998.

[3] Oliver Nelles. Nonlinear dynamic system identification. In *Nonlinear System Identification*, pages 547–577. Springer, 2001.

[4] S Narendra Kumpati, Parthasarathy Kannan, et al. Identification and control of dynamical systems using neural networks. *IEEE Transactions on neural networks*, 1(1):4–27, 1990.

[5] Olalekan Ogunmolu, Xuejun Gu, Steve Jiang, and Nicholas Gans. Nonlinear systems identification using deep dynamic neural networks. *arXiv preprint arXiv:1610.01439*, 2016.

[6] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Multistep neural networks for data-driven discovery of nonlinear dynamical systems. *arXiv preprint arXiv:1801.01236*, 2018.

[7] Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual review of fluid mechanics*, 52:477–508, 2020.

[8] Christian Møldrup Legaard, Thomas Schranz, Gerald Schweiger, Ján Drgoňa, Basak Falay, Cláudio Gomes, Alexandros Iosifidis, Mahdi Abkar, and Peter Gorm Larsen. Constructing neural network-based models for simulating dynamical systems. *arXiv preprint arXiv:2111.01495*, 2021.

[9] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[10] Hassan K Khalil. *Nonlinear control*, volume 406. Pearson New York, 2015.

[11] J Zico Kolter and Gaurav Manek. Learning stable deep dynamics models. *Advances in neural information processing systems*, 32, 2019.

[12] Minghao Han, Yuan Tian, Lixian Zhang, Jun Wang, and Wei Pan. Reinforcement learning control of constrained dynamic systems with uniformly ultimate boundedness stability guarantee. *Automatica*, 129:109689, 2021.

[13] Luca Furieri, Clara Lucía Galimberti, and Giancarlo Ferrari-Trecate. Neural system level synthesis: Learning over all stabilizing policies for nonlinear systems. *arXiv preprint arXiv:2203.11812*, 2022.

[14] Giorgos Mamakoukas, Ian Abraham, and Todd D Murphey. Learning data-driven stable koopman operators. *arXiv preprint arXiv:2005.04291*, 2020.

[15] Max Revay, Ruigang Wang, and Ian R Manchester. Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness. *arXiv preprint arXiv:2104.05942*, 2021.

[16] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32, 2019.

[17] Shaojie Bai, Zhengyang Geng, Yash Savani, and J Zico Kolter. Deep equilibrium optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 620–630, 2022.

[18] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[19] Andreas Schlaginhaufen, Philippe Wenk, Andreas Krause, and Florian Dorfler. Learning stable deep dynamics models for partially observed or delayed dynamical systems. *Advances in Neural Information Processing Systems*, 34:11870–11882, 2021.

[20] Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Multiscale deep equilibrium models. *Advances in Neural Information Processing Systems*, 33:5238–5250, 2020.

[21] Laurent El Ghaoui, Fangda Gu, Bertrand Travacca, Armin Askari, and Alicia Tsai. Implicit deep learning. *SIAM Journal on Mathematics of Data Science*, 3(3):930–958, 2021.

[22] Ezra Winston and J Zico Kolter. Monotone operator equilibrium networks. *Advances in neural information processing systems*, 33:10718–10728, 2020.

[23] Ernest K Ryu and Stephen Boyd. Primer on monotone operator methods. *Appl. Comput. Math*, 15(1):3–43, 2016.

[24] Max Revay, Ruigang Wang, and Ian R Manchester. Lipschitz bounded equilibrium networks. *arXiv preprint arXiv:2010.01732*, 2020.

[25] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[26] Mahyar Fazlyab, Manfred Morari, and George J Pappas. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*, 2020.

[27] Matthew Newton and Antonis Papachristodoulou. Neural network verification using polynomial optimisation. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 5092–5097. IEEE, 2021.

[28] David Angeli. A lyapunov approach to incremental stability properties. *IEEE Transactions on Automatic Control*, 47(3):410–421, 2002.

[29] Chris Verhoek, Patrick JW Koelewijn, Roland Tóth, and Sofie Haesaert. Convex incremental dissipativity analysis of nonlinear systems. *arXiv preprint arXiv:2006.14201*, 2020.

[30] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[31] Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.