# RNN and control

samuele.zoboli

April 2024

# 1 Implicit NN

Implicit neural networks (INNs) (a.k.a equilibrium networks) are a class of machine-learning models based on implicit prediction rules [2, 7, 23, 1, 10]. Such rules are not obtained via a recursive procedure through several layers, as in current neural networks. Instead, they are based on solving a fixed-point equation. Precisely, for a given equilibrium pair $(x^\star, u^\star) \in \mathbb{R}^n \times \mathbb{R}^{m+1}$ satisfying the implicit (or equilibrium) equation

$$x^\star = \phi(W_{xx}x^\star + W_{xu}u^\star)$$

with $W_{xx}, W_{xu}$ of proper dimension and $\phi : \mathbb{R}^n \to \mathbb{R}^n$ the so-called 'nonlinear activation map', the predicted vector $y^\star \in \mathbb{R}^p$ is

$$y^\star = W_{yx}x^\star + W_{yu}u^\star$$

with $W_{yx}, W_{yu}$ of proper dimension. The network parameters are the matrices $W_{xx}, W_{xu}, W_{yx}, W_{yu}$, and these networks can be proved to be much more memory and parameter efficient than classical NNs [2]. Interestingly, [2, Theorem 2] states that stacking implicit units does not improve approximation capabilities, which are already universal for the single one. Note that in this section we follow [7], where the bias term is removed for notation simplicity, and included in the input vector as $u = \mathrm{col}(v, 1)$, with $v \in \mathbb{R}^m$ the 'real' input. These networks may encounter well-posedness issues. Sufficient conditions for well-posedness can be found in [7, Section 2]. For instance, strictly upper (lower) triangular matrices $W_{xx}$ ensure well-posedness for any classical decentralized, memoryless nonlinearity $\phi$ acting componentwise (as the ones commonly used in machine learning). A well-posed INN can be linked to an operator splitting problem via monotone operator theory [23] or a contracting dynamical system via IQC analysis framework [15, 10]. Thus, various numerical methods can be applied for solving an equilibrium, e.g., operator splitting algorithm or ODE solvers.

## 1.1 Relation to classical networks

In [7, Supplementary material], these networks are shown to represent a wide range of existing NNs via a specific choice of weight matrices. Here are some of the most relevant ones, i.e., feedforward NNs (a.k.a. MLPs), residual NNs (ResNets) and recurrent units (RNNs).

**MLPs**– Consider the following prediction produced by a fully-connected multi-layer perception with $L \geq 1$ hidden layers

$$\begin{aligned} x_0 =&\quad u = \mathrm{col}(v, 1) \\ x_{l+1} =&\quad \phi_{l+1}(W_{l+1}x_l), \quad l = 0, \dots, L-1 \\ y =&\quad W_L x_L \end{aligned}$$

where $x_l \in \mathbb{R}^{n_l}$, $W_l \in \mathbb{R}^{n_l \times n_{l-1}}$ with $n_0 = m + 1$ and $\phi_l : \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$ are the output, weight matrix, and activation function of layer $l = 1, \dots, L$. By defining the vector $\mathbf{x} = \mathrm{col}(x_1, \dots, x_L)$ and the function $\Phi : \mathbb{R}^{\sum_{l=1}^L n_l} \to \mathbb{R}^{\sum_{l=1}^L n_l}$

a properly defined blockwise nonlinearity, we have

$$\begin{aligned}\mathbf{x} = &\quad \Phi\left(\underbrace{\begin{pmatrix} 0 & 0 & \dots & 0 & 0 \\ W_2 & 0 & \dots & 0 & 0 \\ 0 & W_3 & \dots & 0 & 0 \\ \vdots & 0 & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & W_{L-1} & 0 \end{pmatrix}}_{W_{xx}} \mathbf{x} + \underbrace{\begin{pmatrix} W_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}}_{W_{xu}} u \right) \\ y = &\quad \underbrace{\begin{pmatrix} 0 & \dots & 0 & W_L \end{pmatrix}}_{W_{yx}} \mathbf{x} + \underbrace{\begin{pmatrix} 0 \end{pmatrix}}_{W_{yu}} u\end{aligned}$$

This is always well-posed as $W_{xx}$ is strictly lower triangular. Hence, the state dimension of the INN is directly linked to the number of layers in the MLP. Moreover, notice that an output activation function can be added without loss of generality.

**ResNets**– Residual neural networks are NNs where layer-skipping connections are inserted, often to avoid vanishing gradient issues. For instance, the input can be injected at different points in the networks (not only in the first layer). It is easy to see that this input-to-layer connection corresponds to the MLP scenario where $W_{xu}$ has multiple nonzero elements, corresponding to the layers the input is connected to.

$$\begin{aligned}\mathbf{x} = &\quad \Phi\left(\underbrace{\begin{pmatrix} 0 & 0 & \dots & 0 & 0 \\ W_2 & 0 & \dots & 0 & 0 \\ 0 & W_3 & \dots & 0 & 0 \\ \vdots & 0 & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & W_{L-1} & 0 \end{pmatrix}}_{W_{xx}} \mathbf{x} + \underbrace{\begin{pmatrix} W_{1,1} \\ W_{1,2} \\ \vdots \\ W_{1,L} \end{pmatrix}}_{W_{xu}} u \right) \\ y = &\quad \underbrace{\begin{pmatrix} 0 & \dots & 0 & W_L \end{pmatrix}}_{W_{yx}} \mathbf{x} + \underbrace{W_{1,y}}_{W_{yu}} u\end{aligned}$$

Following similar intuitions, forward layer-to-layer skipping links appear in the bottom triangular portion of $W_{xx}$, while backward ones are in the top one. Similar reasoning can be done for links to the outputs and matrices $W_{yx}, W_{yu}$.

**RNNs**– Classical recurrent neural networks typically take as input a $T$-long sequence of data ($T \geq 1$) and memory is embedded in the network by evaluating the instantaneous entry according to the current network state (a.k.a. hidden state), which evolves according to a learned dynamics. Mathematically, the RNN prediction is given by

$$\begin{aligned} h_t = &\quad \phi_h(W_h h_{t-1} + W_u u_t) \\ y_t = &\quad \phi_y(W_y h_t) \end{aligned}$$

for $t = 1, \dots, T$, with $h_t \in \mathbb{R}^{n_h}$ the hidden state and $\phi_h : \mathbb{R}^{n_h} \to \mathbb{R}^{n_h}$, $\phi_y : \mathbb{R}^{n_h} \to \mathbb{R}^{n_h}$. These networks are often studied in their 'unrolled' form, i.e., by explicitly representing the $T$ steps at the same time [8]. This can be interpreted as a ResNet (or MLP) with $T$ layers under an extended input vector $\mathbf{u} = \mathrm{col}(u_1, \dots, u_T)$, where each component enters at different layers. More formally, by defining $\mathbf{x} = \mathrm{col}(h_1, \dots, h_T)$ and $W_2 = W_3 = \dots = W_{L-1} = W_h$, $W_L = W_y$, $W_{1,1} = W_{1,2} = \dots = W_u$ and $W_{1_y} = 0$, we obtain

$$\begin{aligned}\mathbf{x} = &\quad \Phi\left(\underbrace{\begin{pmatrix} 0 & 0 & \dots & 0 & 0 \\ W_h & 0 & \dots & 0 & 0 \\ 0 & W_h & \dots & 0 & 0 \\ \vdots & 0 & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & W_h & 0 \end{pmatrix}}_{W_{xx}} \mathbf{x} + \underbrace{\begin{pmatrix} W_u & 0 & \dots & 0 \\ 0 & W_u & \dots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \dots & W_u \end{pmatrix}}_{(\mathrm{I}_T \otimes W_u) = W_{xu}} \mathbf{u} \right) \\ y = &\quad \underbrace{\begin{pmatrix} 0 & \dots & 0 & W_y \end{pmatrix}}_{W_{yx}} \mathbf{x} + \underbrace{\begin{pmatrix} 0 \end{pmatrix}}_{W_{yu}} \mathbf{u}\end{aligned}$$

Therefore, the state dimension of the INN is related to the length $T$ of the input sequence.

# 2 Recurrent implicit NN

Following [15], recurrent networks evolving according to the dynamics

$$\begin{aligned} x^+ &= Ax + B_1 w + B_2 u + b_x \\ y &= C_2 x + D_{21} w + D_{22} u + b_y \end{aligned}$$

with $w$ solution of an implicit network

$$w = \phi(C_1 x + D_{11} w + D_{12} u + b_w)$$

are called Recurrent Equilibrium Networks (RENs) [14, 15, 16, 13] or Recurrent Implicit NN (RINN) [9, 12, 11]. A continuous-time version can be found in [13, 11]. These models expand on INNs and cover a wider class of networks/controllers (e.g., including LTI ones) [16, Section VI]. More interestingly, they represent a Lur'e system where a linear component is interconnected in feedback with a nonlinearity.

A possible way to evaluate well-posed RENs are presented in [16, Section III.C], based on ideas coming from [23]. However, to obtain easily computable solutions, often $D_{11}$ is selected as a lower triangular matrix. According to the intuition provided in the previous section, [16, Section III] points out that $D_{11}$ can be interpreted as the adiacency matrix of a directed graph defining interconnections between the neurons. Hence, if $D_{11}$ is strictly lower triangular, this graph is acyclic and the network well-posed. In the same work, the authors state that this situation is more easily implementable and trainable, while providing comparable results to the general scenario. This is also stated in their paper on the package for implementation [3].

LMI conditions for contractivity, robustness and dissipativity analysis of the REN itself can be found in [15, 16]. Here, we will focus on the study of closed-loop properties under RENs controllers. Hence, we focus on [4, 9, 12, 11]. Some initial results can also be found in [16, Section IX].

## 2.1 Synthesis of discrete-time RNN controllers for closed-loop stability [9, 12]

The contributions [9] and [11] treat the same topic, yet for two different communities. So we treat them as a unique publication and point out where there are small differences.

Envisioning reinforcement learning applications, the authors consider the problem of learning an optimal controller (a.k.a. policy) maximizing the sum of cumulative rewards (generated by an arbitrary reward function) over a finite horizon while guaranteeing closed-loop stability. They focus on LTI partially observed discrete-time plants, evolving according to

$$\begin{aligned} x^+ &= A_p x + B_p u \\ y &= C_p x \end{aligned}$$

Their controller of choice is a REN with input $y$ and output $u$. In [9], $D_{21} = 0$. In [12], they use a standard REN, namely,

$$\begin{aligned} \begin{pmatrix} \xi^+ \\ u \\ v \end{pmatrix} &= \begin{pmatrix} A_c & B_{c1} & B_{c2} y \\ C_{c1} & D_{c11} & D_{c12} \\ C_{c2} & D_{c21} & D_{c22} \end{pmatrix} \begin{pmatrix} \xi \\ w \\ y \end{pmatrix} \\ w &= \phi(v) \end{aligned}$$

with $\phi$ a decentralized memoryless nonlinearity such that $\phi(0) = 0$ and included in the sector $[\alpha, \beta]$ with $\alpha \leq \beta$. Hence, their stability condition is based on the quadratic abstraction

$$\begin{pmatrix} v \\ w \end{pmatrix}^\top \begin{pmatrix} -2\,\mathrm{diag}(\alpha)\,\mathrm{diag}(\beta)\Lambda & (\mathrm{diag}(\alpha) + \mathrm{diag}(\beta))\Lambda \\ \star & -2\Lambda \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} \geq 0,$$

with $\Lambda \succeq 0$ and diagonal. In [9], the authors remark that the set of multipliers could be expanded to exploit other conditions (not only sector boundedness) and to include stabilization of LTI systems in feedback interconnection with an uncertainty. This is studied in the last pages of the paper (Section 4) where the uncertainty satisfies an IQC consisting of a filter with constrained output. In [12, Section III] they analyze uncertainties satisfying a sector condition. In [11] (continuous-time), the uncertainty satisfies either a quadratic constraint, a static IQC or a particular of dynamic IQC.

First, the paper studies conditions on the NN parameters guaranteeing stability of the origin of the closed-loop. To obtain convex conditions, the authors first perform a loop transformation on the REN, which defines a new nonlinearity $\tilde{\phi} \in [-1, 1]$, thus reducing the sector condition to

$$\begin{pmatrix} v \\ z \end{pmatrix}^\top \begin{pmatrix} -\Lambda & 0 \\ \star & -\Lambda \end{pmatrix} \begin{pmatrix} v \\ z \end{pmatrix} \geq 0,$$

with $z$ a newly defined variable arising from the loop transformation. The loop transformation boils down to the change of coordinates

$$\begin{aligned} z =& \quad 2(\mathrm{diag}(\beta) - \mathrm{diag}(\alpha))^{-1}w - (\mathrm{diag}(\alpha) + \mathrm{diag}(\beta))(\mathrm{diag}(\beta) - \mathrm{diag}(\alpha))^{-1}v, \\ \tilde{\phi}(\cdot) =& \quad (\mathrm{diag}(\beta) - \mathrm{diag}(\alpha))^{-1}(\phi(\cdot) - (\mathrm{diag}(\alpha) + \mathrm{diag}(\beta))) \end{aligned}$$

yet more details can be found in the appendix of the arXiv version of [9] or in [12, Section II.D]. The learnable parameters are set as the ones of the transformed loop. Then, by defining the extended state $\zeta = \mathrm{col}(x, \xi)$, they obtain an extended closed-loop system of the form

$$\begin{aligned} \zeta^+ =& \quad \mathcal{A}\zeta + \mathcal{B}z \\ v =& \quad \mathcal{C}\zeta + \mathcal{D}z \\ z =& \quad \tilde{\phi}(v) \end{aligned}$$

on which stability conditions are imposed. The derivation of the stability-related LMI follows standard steps, imposing the decrease of a quadratic Lyapunov function and including the sector conditions via the S-procedure. Details on the proof can be found in the appendix of the arXiv version[1].

In [12], since the conditions are not convex in the transformed parameters, they are developed to obtain a convex formulation, resulting in [12, Theorem 1]. This result does not seem to be present in [9] as the conditions are slightly different.

Once the conditions are found, they are used to define a set of allowed parameters guaranteeing stability. For training the NN, the authors propose a two-step algorithm exploiting existing Deep Reinforcement Learning methods while preserving stability via projection. The main idea is to update the parameters via standard RL, and then to project them on the set of safe parameters at each step.

In [9], this set is first convexified. This results in a set that is parametrized by two matrices and is updated at each training step. The projection is defined as a convex program minimizing the (Frobenius) distance between projected parameters and nominal ones and the one between previous matrices guaranteeing stability and the matrices defining the new safe set, under the constraint of the satisfaction of the stability LMI for the closed-loop. The projection's recursive feasibility is proven in the appendix of the arXiv version.

In [12], there is no discussion about details on the safe set. The projection is simply defined as the safe set's parameters minimizing the Frobenius norm of their error with respect to the parameters obtained with the RL step.

When considering plants modeled by uncertain Lur'e systems, the results are slightly adapted but follow very similar lines (see e.g. [12, Section II]).

## 2.2 Synthesis of continuous-time RNN controllers for closed-loop dissipativity [11]

This paper builds on [11]. The main differences consist in the analysis of a wider class of plants, the study of dissipaivity (rather than stability), considerations about closed-loop performance (e.g., $L_2$ gain) and the analysis of continuous-time RENs and plants, rather than discrete-time ones. Moreover, the training algorithm is modified. Then, by (almost) following the papers' notation to maintain clarity given the added variables, the plant is represented as

$$\begin{pmatrix} \dot{x}_p \\ v_p \\ e \\ y \end{pmatrix} = \begin{pmatrix} A_p & B_{pw} & B_{pd} & B_{pu} \\ C_{pv} & D_{pvw} & D_{pvd} & D_{pvu} \\ C_{pe} & D_{pew} & D_{ped} & D_{peu} \\ C_{py} & D_{pyw} & D_{pyd} & 0 \end{pmatrix} \begin{pmatrix} x_p \\ w_p \\ d \\ u \end{pmatrix}$$

$$w_p = \quad \Delta_p(v_p)$$

with $\Delta_p$ the uncertainty, and the REN

$$\begin{pmatrix} \dot{x}_c \\ v_c \\ y \end{pmatrix} = \begin{pmatrix} A_c & B_{cw} & B_{cy} \\ C_c v & D_{cvw} & D_{cvy} \\ C_c u & D_{cuw} & D_{cuy} \end{pmatrix} \begin{pmatrix} x_c \\ w_c \\ y \end{pmatrix}$$

$$w_c = \quad \phi(v_c)$$

with $\phi$ memoryless, componentwise, sector bounded in $[0, 1]$ and slope restricted in $[0, 1]$.

[1]It is based on an [20, Equation (6)]. However, I believe there is a small typo in the inequalities of the form $X^\top W X - 2X + W^{-1} \succeq 0$ for all square matrices $X$ and $W \succ 0$. The term $2X$ should be $X + X^\top$ since it is not symmetric and it comes from the inequality $(X - WY)^\top W^{-1}(X - WY) \succeq 0$ for $W \succ 0$ and $X, Y$ matrices of correct dimensions.

Dissipativity is defined with a quadratic storage function and with respect to a quadratic supply rate, i.e., for any $T \geq 0$, the system is dissipative if

$$x(T)^\top P x(T) - x(0)^\top P x(0) \leq \int_0^T \begin{pmatrix} d(t) \\ e(t) \end{pmatrix}^\top \underbrace{\begin{pmatrix} X_{dd} & X_{de} \\ \star & X_{ee} \end{pmatrix}}_{X} \begin{pmatrix} d(t) \\ e(t) \end{pmatrix} dt$$

where $P \succeq 0$. If $X = 0$ and $P \succ 0$, we recover stability. If $X_{dd} = \gamma^2\,\mathrm{I}$, $X_{de} = 0$ and $X_{ee} = \mathrm{I}$ with $\gamma \in \mathbb{R}$, we recover an $L_2$ gain bound, If $X_{dd} = X_{ee} = 0$ and $X_{de} = \frac{1}{2}\,\mathrm{I}$, we recover passivity.

The plant uncertainty $\Delta_p$ is characterized via its input-output behavior, described by an IQC involving a filter $\Psi_p$ defined by

$$\begin{pmatrix} \dot{\psi}_p \\ z_p \end{pmatrix} = \begin{pmatrix} A_\psi & B_{\psi v} & B_{\psi w} \\ C_\psi & D_{\psi v} & D_{\psi w} \end{pmatrix} \begin{pmatrix} \psi_p \\ v_p \\ w_p \end{pmatrix}$$

satisfying the hard IQC

$$\int_0^T z_p(t)^\top M_\Delta z_p(t)\, dt \geq 0, \quad \forall T \geq 0, \qquad\qquad M_\Delta = \begin{pmatrix} M_{\Delta vv} & M_{\Delta vw} \\ \star & M_{\Delta ww} \end{pmatrix}$$

for all $v_p$ and $w_p = \Delta_p(v_p)$. This case can be specialized to static IQCs ($z_p = \mathrm{col}(v_p, w_p)$) or quadratic constraints of the form

$$\begin{pmatrix} v_p \\ w_p \end{pmatrix}^\top M_\Delta \begin{pmatrix} v_p \\ w_p \end{pmatrix} \geq 0$$

Notice that the REN also satisfies a quadratic constraint with

$$M_\phi = \begin{pmatrix} 0 & \Lambda \\ \star & -2\Lambda \end{pmatrix}$$

for all diagonal $\Lambda \succeq 0$.

Following a similar approach to [9, 12], the authors first derive LMI-based conditions for dissipativity for a generic uncertain LTI system. They distinguish between the scenarios of static and dynamic IQCs, but the proofs follow similar steps ('integral of Lyapunov' to obtain storage difference and S-procedure). For the case of dynamic IQCs, the authors only focus on filters of the form

$$\Psi = \mathrm{blkdiag}(\Psi_1, \Psi_2)$$

with $\Psi$ invertible, stable and with stable proper inverse, and satisfying the IQC with

$$M = \mathrm{blkdiag}(\mathrm{I}, -\mathrm{I})$$

The dissipativity study for dynamic IQCs is performed after a state extension including the filters state ($\tilde{x} = \mathrm{col}(x, \psi_1, \psi_2)$) and a loop transformation (obtaining $\tilde{\Delta} = \Psi_1 \Delta \Psi_1^{-1}$ which now satisfies a **static** IQC defined by the above block-diagonal $M$).

Once the general analysis is completed, the authors apply it to the closed-loop scenario. In this case, they define a combined uncertainty satisfying the static IQC (coming from the loop transformation) with matrix $M$ including the quadratic abstraction satisfied by the REN and the filters IQC. To obtain a bilinear matrix inequality for dissipativity, they restrict the class of multipliers $M_\Delta$ and supply rate $X$ to the ones satisfying $X_{ee} \preceq 0$ and $M_{\Delta vv} \succeq 0$. This restriction still includes memoryless nonlinearities sector bounded in $[0, 1]$ and supply rates defining $L_2$ gain bounds and passivity. After a change of coordinates aimed at linearizing the conditions with respect to the new decision variables (inspired by [17, Section IV.B]), they obtain LMI conditions for dissipativity. This is obtained under the assumption that $M_\Delta$ and $X$ are given (otherwise the problem is not linear) and that $P \succ 0$ in the storage function (otherwise the change of coordinates is not valid). Nontheless, the authors remark that $M_{ww}$ and $X_{dd}$ can be selected as decision variables, as the problem is affine in these matrices. This would allow, e.g., minimization of the $L_2$ gain on the disturbance signal.

The last contribution of the paper with respect to [12] is a renovated training algorithm that does not impose projection at each parameter iteration, but it first checks for the necessity of it (differently from [9, 12][2]). Moreover, instead of working on the transformed parameters, it moves back and forth between nominal and transformed ones. This new algorithm first performs a step of RL, then it checks if the closed-loop is dissipative (trying to solve the LMI). If so, it stores the solution to the LMI (i.e., $P, \Lambda$) and restarts. If the conditions are not satisfied, they perform three main steps

[2] Even though projection of safe parameters on the safe set are the parameters themselves

1. They perform necessary transformations from the standard REN parameters to the ones used in their LMI (which involves including the old solution for $P, \Lambda$)

2. They project these parameters to the set of transformed ones satisfying the dissipativity LMI (by minimizing the distance with the old parameters)

3. They recover a set of parameters suitable for the REN by extracting the new solution $(P, \Lambda)$ and project the RL-derived parameters on the set certified by these new variables. This is done instead of the backward transformation to try to remain as close as possible to the RL parameters.

### 2.3 Youla-REN and contraction [4, 22, 21]

Interesting results for incremental stability of RNNs can be found in [5]. Yet, we will focus on RENs. RENs were first introduced in relation to contraction [15]. This relation has also been explored in the context of control via the so-called Youla-REN [22, 21, 13]. The main idea is to derive a nonlinear Youla parameterization and learn a suitable controller accordingly.

This controller is sampled from the space of contractive ones. LMI conditions for contractivity (and incremental gain bounds) can be found in [16] (discrete) and [13]. As they assume the nonlinearity to be slope-restricted in $[0, 1]$, all their results are based on the incremental quadratic constraint

$$\begin{pmatrix} v_a - v_b \\ w_a - w_b \end{pmatrix}^\top \begin{pmatrix} 0 & \Lambda \\ \star & -2\Lambda \end{pmatrix} \begin{pmatrix} v_a - v_b \\ w_a - w_b \end{pmatrix} \geq 0$$

with diagonal $\Lambda \succeq 0$, and the incremental IQC

$$\sum_{t=0}^{T} \begin{pmatrix} y_a - y_b \\ u_a - u_b \end{pmatrix}^\top \begin{pmatrix} -Q & S^\top \\ \star & R \end{pmatrix} \begin{pmatrix} y_a - y_b \\ u_a - u_b \end{pmatrix} \geq 0, \qquad \forall T \in \mathbb{N}$$

with $Q \succeq 0$ and $R \succeq 0$. As we are focused on discrete-time networks[3], we will discuss [21] and [4]. In particular, we will focus on [21], as [4] provides some extensions for nonlinear systems, yet without involving LMIs.

One key difference from [9, 12] is that they perform unconstrained optimization, without the need to check LMIs or project parameters. Moreover, they show that the REN is a universal approximation of contracting and Lipschitz nonlinear Youla parametrizations,i.e, they can model any nonlinear stabilizing controller[4]. We will focus on linear plants. The authors consider a linear plant of the form

$$\begin{aligned} x^+ = &\quad Ax + Bu + d_x \\ y = &\quad Cx + d_y \end{aligned}$$

with $d = \mathrm{col}(d_x, d_y)$ a disturbance. They aim to design a (possibly dynamic) output-feedback nonlinear controller such that:

- The closed-loop is a contraction
- The closed loop has bounded $L_2$ gain, i.e., $|z_a - z_b| \leq \gamma |d_a - d_b|$ where $z = \mathrm{col}(x, u)$
- A finite horizon objective function is minimized

The main idea is to use RL to learn a positive definite matrix which uniquely maps to the RENs parameters, and hence can be used to link the objective function with parameters of a network satisfying the desired conditions.

They start by assuming an initial linear output-feedback controller is known (e.g. LQG), providing matrices $K$ (stabilization) and $L$ (observation). To this base controller, they add a nonlinear term mimicking the Youla parametrization for the linear scenario

$$\begin{aligned} \hat{x}^+ = &\quad A\hat{x} + Bu + L(\underbrace{y - C\hat{x}}_{\tilde{y}}) \\ u = &\quad -K\hat{x} + \mathcal{Q}(\tilde{y}) \end{aligned}$$

[3] I believe this to be the more proper representation as most things in AI are done in discrete-time.

[4] If the plant is linear, stabilizable and detectable, the idea is to add a prestabilizer and a linear observer, then model all the rest with the REN. In the nonlinear case, they assume a robustly stabilizing base controller and contracting, Lipschitz observer are known.

The authors remark that $\mathcal{Q}$ could include other terms, such as reference signals, feedforward terms etc[5].

They first show that any stabilizing (locally Lipschitz) dynamic output-feedback controller can be parametrized via $\mathcal{Q}$, with $\mathcal{Q}$ Lipschitz and contracting. Then, the authors show that RENs are universal approximations for $\mathcal{Q}$. In other words, that state that for any $M, \epsilon\, 0$ and any controller augmentation $\mathcal{Q}$ which is locally Lipschitz and contracting, there exists a sufficiently large (in terms of state and nonlinearity dimension) REN $\tilde{\mathcal{Q}}$ such that $|\mathcal{Q}(y) - \tilde{\mathcal{Q}}(y)|_\infty \leq \epsilon$, for all $y$ such that $|y|_\infty < M$[6]. These results are extended to nonlinear systems in [21] and used to design different kinds of RENs (contracting, incrementally passive, etc.), however only presented on the practical side [3].

As previously mentioned, their main advantage seems to be the possibility of training via unconstrained optimization. This is due to a 'direct reparametrization' proposed in [15], allowing them to search only in the space of safe parameters without the need of projections. This is done first by augmenting the REN definition (first equations on page 3 of this document) with additional parameters $L_1 \in \mathbb{R}^{n\times n}$ (invertible) and $L_2 \succ 0$ (diagonal), resulting in

$$\begin{aligned} \begin{pmatrix} L_1 x^+ \\ L_2 v \\ y \end{pmatrix} &= \begin{pmatrix} \mathbf{A} & \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{C}_1 & \mathbf{D}_{11} & \mathbf{D}_{12} \\ C_2 & D_{21} & D_{22} \end{pmatrix} \begin{pmatrix} x \\ w \\ u \end{pmatrix} + \begin{pmatrix} \mathbf{b}_x \\ \mathbf{b}_v \\ b_y \end{pmatrix} \\ w &= \phi(v) \end{aligned}$$

where

$$\begin{array}{llll} \mathbf{A} = L_1 A & \mathbf{B}_1 = L_1 B_1 & \mathbf{B}_2 = L_1 B_2 & \mathbf{b}_x = L_1 b_x \\ \mathbf{C}_1 = L_2 C_1 & \mathbf{D}_{11} = L_2 D_{11} & \mathbf{D}_{12} = L_2 D_{12} & \mathbf{b}_y = L_2 b_y \end{array}$$

Due to invertibility of $L_1, L_2$, the initial parameters can be recovered fairly easily. Then, they adapt their LMI conditions (for contraction) to the new parameters, resulting in an LMI of the form $H \succ 0$ for some matrix $H$ containing the new RENs parameters in its blocks. Then, instead of learning $H$, they decompose it as $H = X^\top X + \epsilon\,\mathrm{I}$ and learn $X$. This ensures $H$ is always positive definite independently from how we train $X$, so they do not need projections (we did something similar to learn contraction metrics in Lyon). At each step, once they have $X$, they recover the RENs parameters and start again.

# 3 Ideas

Here are some initial ideas for possible developments.

1. Hybrid and event-triggered study: Up to now, these works studied discrete-time systems with discrete-time NNs, or continuous-time systems and continuous-time NNs. However, we often have continuous-time systems and discrete-time RENs. Can we provide conditions based on hybrid theory? As the event-triggered law falls into this framework, maybe we can imagine a network acting in discrete time AND choosing whether to compute or not (in case skipping to the next sampling time). We could also include different multipliers (e.g. for local sector conditions as in your Automatica paper [6]).

2. Regulation: Up to now, the internal dynamic of the REN has been exploited for partial observation. Can we find a way to use it (or to add some) to satisfy some conditions for robust regulation? E.g., a REN that makes the closed loop a contraction is still not robustly regulating, as it cannot generate internal models with oscillatory behaviors. Maybe we can add some learnable dynamics imposing good behaviors.

[5] I have some doubts that this is the case, but maybe I am missing something. Here are my thoughts on why I believe they claim some things a little too fast.

1. The first doubt comes the fact that they say that a contracting REN can universally approximate any robust nonlinear controller, yet they only add as a base controller a stabilizing one. Hence, if I want to robustly regulate, I am still missing an internal model. In other words, the rest of the control should be able to generate the required oscillations. I cannot see how a contracting system could do this without an additional external oscillator (or multiple ones).

2. Maybe they mean that, if you inject the reference signal as input, the contracting system entrains to it. Yet, this is not useful for disturbance rejection. Moreover, we typically have an OUTPUT reference signal. If we have an input reference, it implies we must have already solved the regulator equations (as the REN cannot do it). If this is not the case, I am pretty sure you generally do not robustly converge to the desired output by just injecting a reference with the desired frequency into the input (this is why, e.g., we need the integral action to correct our feedforward term).

[6] This is not so surprising to me, as the nonlinearity dimension of the REN can be interpreted as the size/amount of layers of an MLP, which is known to be a universal approximator

3. Local conditions and data-driven: most (if not all) of the works focus on global stability results. It may be interesting to address the local problem using local generalized conditions and providing estimates of the region of attraction. Moreover, it may be very interesting for the AI community to propose data-driven approaches, inspired by this recent work on data-driven MLPs and linear systems and Sophie's techniques in [19, 18]. Event-triggering mechanisms (static and dynamic) can be included similarly to the Automatica paper [6].

# References


[1] Atish Agarwala and Samuel S Schoenholz. Deep equilibrium networks are sensitive to initialization statistics. In *International Conference on Machine Learning*, pages 136–160. PMLR, 2022.

[2] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.

[3] Nicholas H. Barbara, Max Revay, Ruigang Wang, Jing Cheng, and Ian R. Manchester. Robustneuralnetworks.jl: a package for machine learning and data-driven control with certified robustness, 2023.

[4] Nicholas H Barbara, Ruigang Wang, and Ian R Manchester. Learning over contracting and lipschitz closed-loops for partially-observed nonlinear systems. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 1028–1033. IEEE, 2023.

[5] William D'Amico, Alessio La Bella, and Marcello Farina. An incremental input-to-state stability condition for a class of recurrent neural networks. *IEEE Transactions on Automatic Control*, 2023.

[6] Carla de Souza, Antoine Girard, and Sophie Tarbouriech. Event-triggered neural network control using quadratic constraints for perturbed systems. *Automatica*, 157:111237, 2023.

[7] 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.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[9] Fangda Gu, He Yin, Laurent El Ghaoui, Murat Arcak, Peter Seiler, and Ming Jin. Recurrent neural network controllers synthesis with stability guarantees for partially observed systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 5385–5394, 2022.

[10] Saber Jafarpour, Alexander Davydov, Anton Proskurnikov, and Francesco Bullo. Robust implicit networks via non-euclidean contractions. *Advances in Neural Information Processing Systems*, 34:9857–9868, 2021.

[11] Neelay Junnarkar, Murat Arcak, and Peter Seiler. Synthesizing Neural Network Controllers with Closed-Loop Dissipativity Guarantees. *Submitted to Automatica*, 2024.

[12] Neelay Junnarkar, He Yin, Fangda Gu, Murat Arcak, and Peter Seiler. Synthesis of stabilizing recurrent equilibrium network controllers. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 7449–7454. IEEE, 2022.

[13] Daniele Martinelli, Clara Lucía Galimberti, Ian R Manchester, Luca Furieri, and Giancarlo Ferrari-Trecate. Unconstrained parametrization of dissipative and contracting neural ordinary differential equations. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 3043–3048. IEEE, 2023.

[14] Max Revay, Ruigang Wang, and Ian R Manchester. A convex parameterization of robust recurrent neural networks. *IEEE Control Systems Letters*, 5(4):1363–1368, 2020.

[15] Max Revay, Ruigang Wang, and Ian R Manchester. Recurrent equilibrium networks: Unconstrained learning of stable and robust dynamical models. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 2282–2287. IEEE, 2021.

[16] Max Revay, Ruigang Wang, and Ian R Manchester. Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness. *IEEE Transactions on Automatic Control*, 2023.

[17] Carsten Scherer, Pascal Gahinet, and Mahmoud Chilali. Multiobjective output-feedback control via lmi optimization. *IEEE Transactions on automatic control*, 42(7):896–911, 1997.

[18] Alexandre Seuret and Sophie Tarbouriech. A data-driven approach to the l2 stabilization of linear systems subject to input saturations. *IEEE Control Systems Letters*, 2023.

[19] Alexandre Seuret and Sophie Tarbouriech. Robust data-driven control design for linear systems subject to input saturation. *IEEE Transactions on Automatic Control*, 2024.

[20] Mark M Tobenkin, Ian R Manchester, and Alexandre Megretski. Convex parameterizations and fidelity bounds for nonlinear identification and reduced-order modelling. *IEEE Transactions on Automatic Control*, 62(7):3679–3686, 2017.

[21] Ruigang Wang, Nicholas H Barbara, Max Revay, and Ian R Manchester. Learning over all stabilizing nonlinear controllers for a partially-observed linear system. *IEEE Control Systems Letters*, 7:91–96, 2022.

[22] Ruigang Wang and Ian R Manchester. Youla-ren: Learning nonlinear feedback policies with robust stability guarantees. In *2022 American Control Conference (ACC)*, pages 2116–2123. IEEE, 2022.

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