Metric Entropy Limits on Recurrent Neural Network Learning of Linear Dynamical Systems

Clemens Hutter, Recep Gül, and Helmut Bölcskei

Chair for Mathematical Information Science, ETH Zurich
Sternwartstrasse 7, 8092 Zurich, Switzerland

Abstract

One of the most influential results in neural network theory is the universal approximation theorem [1, 2, 3] which states that continuous functions can be approximated to within arbitrary accuracy by single-hidden-layer feedforward neural networks. The purpose of this paper is to establish a result in this spirit for the approximation of general discrete-time linear dynamical systems—including time-varying systems—by recurrent neural networks (RNNs). For the subclass of linear time-invariant (LTI) systems, we devise a quantitative version of this statement. Specifically, measuring the complexity of the considered class of LTI systems through metric entropy according to [4], we show that RNNs can optimally learn—or identify in system-theory parlance—stable LTI systems. For LTI systems whose input-output relation is characterized through a difference equation, this means that RNNs can learn the difference equation from input-output traces in a metric-entropy optimal manner.

1. Introduction

During the past decade recurrent neural networks (RNNs) have revolutionized numerous machine learning applications, such as handwritten text recognition [5], speech recognition [6], language translation [7], and modeling of complex game dynamics [8]. Abstractly speaking, an RNN realizes a dynamical system mapping an input sequence to an output sequence through—in each time step—application of a single-hidden-layer neural network to update a hidden state vector and compute the output signal sample. It is hence natural to ask which classes of dynamical systems can be realized or approximated by RNNs. This question is inspired by the well-known universal approximation theorem for feedforward neural networks [1, 2, 3], which states that every continuous function on a compact interval can be approximated to within arbitrarily small error by a single-hidden-layer neural network, provided that the number of neurons is allowed to go to infinity as the approximation error approaches zero.

The first central result in this paper establishes that RNNs universally exactly realize the class of linear dynamical systems, including time-varying systems. This universal linear dynamical system realization theorem builds on a

strong representation theorem for general linear operators stemming from harmonic analysis [9, Theorem 14.3.5],[10, 11], which states that every "reasonable" linear operator can be written as a weighted superposition of time-frequency shift operators. Note that we conspicuously use the term "realization theorem" instead of "approximation theorem" as RNNs with real-valued weights will, indeed, be shown to exactly realize general linear dynamical systems.

The second central theme of this paper revolves around making the universal system realization theorem quantitative. Specifically, we consider classes of linear dynamical systems, quantify their complexity through metric entropy according to [12, 4], and ask how the number of bits needed to uniquely specify RNNs approximating systems in this class to within a prescribed error relates to the class's metric entropy. This part of the theory we develop is restricted to linear time-invariant (LTI) systems for conceptual reasons. The main result we obtain states that RNNs with suitably quantized weights provide optimal coverings—in the sense of metric entropy—for the class of LTI systems with exponentially decaying impulse response. In control theory parlance, this says that RNNs can be trained to identify LTI systems with exponentially decaying impulse response in a metric-entropy optimal fashion. We also show that, equivalently, this means that certain classes of linear difference equations with constant coefficients can be learned optimally by RNNs. The overall philosophy of the framework we propose is inspired by the recently established Kolmogorov-Donoho rate-distortion theory [13, 14, 15, 16] for feedforward neural networks [17, 18] which shows that deep neural networks provide optimal coverings for a wide range of function classes, such as unit balls in Besov spaces and in modulation spaces.

We hasten to add that, throughout the paper, we are exclusively concerned with the fundamental representation capabilities of RNNs and do not consider the issue of learning algorithms, a topic that has been investigated in the context of LTI system identification in [19, 20].

Previous work on the approximation of linear dynamical systems through neural networks deals with (linear and nonlinear) time-invariant systems and assumes that the system is specified in terms of a state space representation, concretely by a (time-invariant) next-state function which is approximated by a single-hidden-layer neural network, the existence of which is guaranteed by the classical universal approximation theorem [1, 2, 3]. This approach leads, however, to the accumulation of errors over time so that most results along these lines are restricted to finite time horizons [21, 22, 23]. A notable exception in this regard is [24], which avoids error build-up by imposing an "absolute summability" condition on the system's possible state trajectories. Nonetheless, all these results require that the system be characterized by a state space representation, the existence of which is not guaranteed for a given linear dynamical system [25, Theorem 2.3.3]. In the present paper, we do not impose such an existence assumption. In addition, our theory comprises time-varying systems and pertains to unbounded time horizons, but, as already mentioned, is restricted to linear systems.

As for our second central theme, namely metric-entropy-optimal RNN learn-

ing of LTI systems, to the best of our knowledge, such an approach has not been pursued before in the literature. Related previous work reported in [20] quantifies the number of real-valued RNN weights required for a desired approximation quality, but does not attempt to specify the approximating RNNs through bit-strings of finite length.

We furthermore want to highlight work on a non-recurrent neural network architecture, termed "Deep operator network" [26, 27, 28], which enables the universal approximation of nonlinear operators. Finally, RNNs have also been investigated for the approximation of algorithms, with a prominent result [29] proving that RNNs with binary input and output sequences and rational weights can simulate any Turing machine.

Outline of the paper. In the remainder of this section, we provide preparatory material on RNNs and on harmonic analysis of general linear dynamical systems. In Section 2, we develop the first central result of the paper, namely a universal realization theorem for discrete-time linear dynamical systems. In Section 3, we introduce the concept of metric entropy of classes of LTI systems, based on which, in Section 4, we state the second central result establishing that RNNs realize LTI systems of exponentially decaying impulse response in a metric-entropy-optimal fashion. Appendices A and B summarize technical results needed in the main body of the paper.

Notation. Vectors are indexed starting with $\ell = 1$. $\mathbb{1}_{\{\cdot\}}$ denotes the truth function which takes on the value 1 if the statement inside $\{\cdot\}$ is true and equals 0 otherwise. Sequences $x[t] \in \mathbb{R}$ are indexed by $t \in \mathbb{Z}$. The $N \times N$ identity matrix is $\mathbb{I}_N$ and $\mathbb{O}_N$ stands for the $N \times N$ all zeros matrix. 1_N and 0_N denote the N-dimensional column vector with all entries equal to 1 and 0, respectively. $\log(\cdot)$ refers to the natural logarithm. We write $f(\epsilon) \sim g(\epsilon)$ to mean $\lim_{\epsilon \to 0} \frac{f(\epsilon)}{g(\epsilon)} = 1$. Throughout the paper, constants are understood to be in $\mathbb{R}$ unless explicitly stated otherwise.

1.1. Recurrent Neural Networks

A recurrent neural network (RNN) is described by a hidden state vector sequence $h[t]$, the input signal $x[t]$, and the output signal $y[t]$. In each time instant t, a single-hidden-layer neural network is applied to the concatenation of the input sample $x[t]$ and the previous state vector $h[t-1]$ to produce the current output sample $y[t]$ and the new state vector $h[t]$. The formal definition is as follows.

Definition 1.1 (Recurrent neural network)**.** *For $n \in \mathbb{N}$ and hidden state dimension $m \in \mathbb{N}$, let $\Phi : \mathbb{R}^{m+1} \to \mathbb{R}^{m+1}$ be a feedforward neural network given by*

$$\Phi(x) = A_2\, \rho(A_1 x + b_1) + b_2, \quad x \in \mathbb{R}^{m+1}, \tag{1}$$

with weight matrices $A_1 \in \mathbb{R}^{n \times (m+1)}$, $A_2 \in \mathbb{R}^{(m+1) \times n}$, bias vectors $b_1 \in \mathbb{R}^n$, $b_2 \in \mathbb{R}^{m+1}$, and the ReLU activation function $\rho(x) = \max\{x, 0\}$, $x \in \mathbb{R}$, applied element-wise. The recurrent neural network associated with Φ is the operator

$\mathcal{R}_\Phi : \ell_\infty \to \ell_\infty$ *mapping input sequences* $(x[t])_{t\geq 0}$ *in* $\mathbb{R}$ *to output sequences* $(y[t])_{t\geq 0}$ *in* $\mathbb{R}$ *according to*

$$\begin{pmatrix} y[t] \\ h[t] \end{pmatrix} = \Phi\left(\begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix}\right), \ \forall t \geq 0, \tag{2}$$

where $h[t] \in \mathbb{R}^m$ *is the hidden state sequence with initial state* $h[-1] = 0_m$.

Remark 1.1. *Classical RNN definitions are often referred to as Elman networks [30], [31, p274]. We show in Appendix A that our RNN definition does not afford increased generality over Elman networks as every RNN according to Definition 1.1 can be converted into an Elman RNN. We decided, however, to work with the seemingly more general Definition 1.1 for expositional simplicity.*

We now introduce a decomposition of the weight matrix A_2 which will simplify the description of RNN constructions later in the paper. Specifically, we represent A_2 according to

$$A_2 = \begin{pmatrix} A_o A_r \\ A_h \end{pmatrix} \in \mathbb{R}^{(m+1)\times n}, \tag{3}$$

where $A_h \in \mathbb{R}^{m\times n}$ is responsible for mapping to the next hidden state and, for some $R \in \mathbb{N}$, $A_r \in \mathbb{R}^{R\times n}$ maps to an R-dimensional virtual representation which, in turn, is linearly combined through the weights $A_o \in \mathbb{R}^{1\times R}$ to deliver the output $y[\cdot]$. Consorting with this decomposition of A_2 and noting that $b_2 = 0_{m+1}$ in all our concrete RNN constructions, the hidden state sequence evolution can be written as

$$h[t] = A_h g[t], \quad \forall t \geq 0, \tag{4}$$

where

$$g[t] = \rho\left(A_1 \begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix} + b_1\right), \quad \forall t \geq 0, \tag{5}$$

and the initialization is $h[-1] = 0_m$ as before. The output sequence is accordingly obtained as

$$r[t] = A_r g[t], \tag{6}$$
$$y[t] = A_o r[t], \tag{7}$$

where $r[t] \in \mathbb{R}^R$ denotes the virtual representation sequence. We note that this virtual representation never actually manifests itself, it is introduced solely to simplify the specific RNN constructions later in the paper. Finally, we remark that, throughout, whenever we speak of "weights" of the RNN, this shall refer to nonzero entries both in the weight matrices A_1, A_2 and the bias vectors b_1, b_2.

1.2. Harmonic Analysis of Linear Dynamical Systems

We consider discrete-time causal linear systems $\mathcal{L}$ mapping input sequences $x[\cdot] \in \ell_\infty$ to output sequences $y[\cdot] \in \ell_\infty$, and we use the convention

$$x[t] = 0, \qquad \forall t < 0, \tag{8}$$

which, by causality and linearity, implies $y[t] = 0$, $\forall t < 0$.

A fundamental result from harmonic analysis [9, Theorem 14.3.5], in its incarnation for discrete-time systems, states that a wide class of linear operators, i.e., linear dynamical systems, can be represented as a weighted superposition of time-frequency shift operators according to

$$y[t] = \sum_{\tau=0}^{\infty} \int_0^1 S_{\mathcal{L}}(\tau, \nu) x[t-\tau] e^{2\pi i \nu t} d\nu, \tag{9}$$

with the weights given by the delay-Doppler spreading function $S_{\mathcal{L}}(\tau, \nu)$. Alternatively, (9) can be expressed in terms of the operator kernel, a.k.a. time-varying impulse response, $k[t, \tau]$, as

$$y[t] = \sum_{\tau=0}^{\infty} k[t, \tau] x[t-\tau], \tag{10}$$

where $k[t, \tau]$ is related to the spreading function through an inverse Fourier transform according to

$$k[t, \tau] = \int_0^1 S_{\mathcal{L}}(\tau, \nu) e^{2\pi i \nu t} d\nu. \tag{11}$$

For a mathematically accessible introduction to this theory, we refer the interested reader to [11].

Throughout the paper, in an attempt to minimize the level of technical sophistication and expositional complexity, we will work with a fully discrete and finite-dimensional version of (9) given by

$$y[t] = \sum_{\tau=0}^{D-1} \sum_{f=0}^{F-1} \widetilde{S}_{\mathcal{L}}(\tau, f) x[t-\tau] e^{2\pi i \frac{f}{F} t}, \tag{12}$$

with D, $F \in \mathbb{N}$. In the continuous-time case the size of the spreading function support area plays a critical role as there is a threshold beyond which the system becomes unidentifiable [11]. While we will not dwell on this matter, we simply note that in the setup considered here, the spread is given by $D \cdot F$, i.e., the total number of time-frequency shifts the system induces.

2. Universal Realization of Linear Dynamical Systems

In this section, we develop our first central result, a universal realization theorem for linear dynamical systems. This will be effected by building on the

spreading decomposition (12). Specifically, we first devise—in Lemma 2.1—RNNs that realize time shifts, then—in Lemma 2.2—RNNs implementing frequency shifts, and finally these building blocks are put together to obtain an RNN that realizes a weighted superposition of time-frequency shifts according to (12).

We start with RNNs that realize time shifts. For later reference, we actually construct more general RNNs that implement convolutions, i.e., weighted superpositions of time shifts.

Lemma 2.1 (RNNs can realize time shifts and convolutions)**.** *Let $L \in \mathbb{N}$ and $k \in \mathbb{R}^L$. There exists an RNN with input-output relation*

$$y[t] = \sum_{\ell=1}^{L} k_\ell \, x[t-(\ell-1)], \quad \forall t \geq 0, \tag{13}$$

hidden state dimension $L-1$, and hidden state sequence satisfying

$$h_\ell[t] = x[t-(\ell-1)], \quad \forall t \geq 0, \ell \in \{1,\dots,L-1\}. \tag{14}$$

Proof. The proof is constructive in the sense of specifying the RNN as a function of the impulse response vector $k \in \mathbb{R}^L$. We start by choosing weight matrices and bias vectors such that (14) holds. The basic idea is to design the network such that the past values of $x[\cdot]$ in the hidden state vector h are shifted downward by one position in each time step t, dropping the oldest value at the bottom of the vector and inserting the current value $x[t]$ at the top. To move the values through the non-linear activation function without modifying them, we employ the identity

$$x = \rho(x) - \rho(-x). \tag{15}$$

We set

$$A_1 = \begin{pmatrix} \mathbb{I}_L \\ -\mathbb{I}_L \end{pmatrix} \in \mathbb{R}^{2L \times L}, \tag{16}$$

$$A_h = \begin{pmatrix} \mathbb{I}_{L-1} & 0_{L-1} & -\mathbb{I}_{L-1} & 0_{L-1} \end{pmatrix} \in \mathbb{R}^{(L-1)\times 2L}, \tag{17}$$

$b_1 = 0_{2L}$, and $b_2 = 0_L$.

With these choices, the proof of (14) is now effected by induction over t. First, we note that for $h[t]$ in (14) to constitute a valid hidden state sequence according to Definition 1.1, the initial state needs to satisfy $h[-1] = 0_{L-1}$. This follows directly from $x[t] = 0$, $\forall t < 0$, which is by assumption (8), and also constitutes the base case of the induction argument. To establish the induction step, we assume that (14) holds for $t-1$ for some $t \geq 0$, i.e.,

$$h_\ell[t-1] = x[(t-1)-(\ell-1)] = x[t-\ell],$$

and show that—thanks to the choices for A_1, A_h, b_1, and b_2 made above—this

implies validity of (14) for t. Using (16) and $b_1 = 0_{2L}$ in (5), one obtains

$$g[t] = \rho\left(A_1 \begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix}\right) = \rho\left(A_1 \begin{pmatrix} x[t] \\ x[t-1] \\ \vdots \\ x[t-(L-1)] \end{pmatrix}\right) \tag{18}$$

$$= \begin{pmatrix} \rho(x[t]) & \dots & \rho(x[t-(L-1)]) & \rho(-x[t]) & \dots & \rho(-x[t-(L-1)]) \end{pmatrix}^T .$$

Then, we evaluate (4) with A_h from (17) and use (15) to get

$$h[t] = A_h g[t] = \begin{pmatrix} \rho(x[t]) - \rho(-x[t]) \\ \vdots \\ \rho(x[t-(L-2)]) - \rho(-x[t-(L-2)]) \end{pmatrix} = \begin{pmatrix} x[t] \\ \vdots \\ x[t-(L-2)] \end{pmatrix},$$

or equivalently $h_\ell[t] = x[t-(\ell-1)], \forall \ell \in \{1, \dots, L-1\}$, which establishes the induction step.

It remains to realize the input-output relation (13). To this end, we set

$$A_r = \begin{pmatrix} \mathbb{I}_L & -\mathbb{I}_L \end{pmatrix} \in \mathbb{R}^{L \times 2L}, \qquad A_o = k^T \in \mathbb{R}^{1 \times L}, \tag{19}$$

and use (18), (6), (7), and (15) to conclude that

$$r[t] = A_r g[t] = \begin{pmatrix} \rho(x[t]) - \rho(-x[t]) \\ \vdots \\ \rho(x[t-(L-1)]) - \rho(-x[t-(L-1)]) \end{pmatrix} = \begin{pmatrix} x[t] \\ \vdots \\ x[t-(L-1)] \end{pmatrix},$$

$$y[t] = k^T r[t] = \sum_{\ell=1}^{L} k_\ell \, x[t-(\ell-1)],$$

which, in turn, completes the proof. □

Remark 2.1. *An RNN realizing a time shift by m instants, as needed in (12), is now obtained from Lemma 2.1 by choosing the impulse response vector $k \in \mathbb{R}^L$ such that it has a 1 in the $(m+1)$-th entry and zeros elsewhere.*

The next step in our program is to construct an RNN that realizes frequency shifts by integer multiples of $1/F$, again as needed in (12). As this operation corresponds to multiplication of the input signal by a complex exponential, it produces complex outputs $y[\cdot]$. In slight abuse of Definition 1.1, where all weight matrices and bias vectors are real-valued, for ease of exposition, we will here allow complex weights in the output layer, specifically for the quantities A_o and A_r in (3). As A_o and A_r do not appear in the state evolution equations (4) and (5), it is guaranteed that the activation function ρ continues to be applied to real-valued quantities only.

Lemma 2.2 (RNNs can realize frequency shifts)**.** *Let $F \in \mathbb{N}$ and $f \in \{0, \dots, F-1\}$. There exists an RNN that realizes the input-output mapping*

$$y[t] = x[t]\, e^{2\pi i \frac{f}{F} t}, \quad \forall t \geq 0. \tag{20}$$

Proof. Again the proof is constructive in the sense of specifying the RNN. Throughout the proof, unless explicitly stated otherwise, relations involving t apply for all $t \geq 0$. We start by noting that the function $e^{2\pi i \frac{f}{F} t}$ is F-periodic in $t \in \mathbb{N}$. This F-periodicity motivates the choice of an $(F-1)$-dimensional hidden state sequence $h[t] \in \{0,1\}^{(F-1)}$ encoding the current position within the fundamental period. Specifically, our construction will be seen to ensure

$$h_\ell[t] = \mathbb{1}_{\{((t+1) \bmod F) = \ell\}}, \quad \forall \ell \in \{1, \dots, F-1\}. \tag{21}$$

The hidden state vector at time t hence contains a one at position $(t+1)$ mod F or equals the all-zeros vector at the end of each period, i.e., when $(t+1)$ mod $F = 0$. We will realize (21) by appropriate choice of the RNN weight matrices A_1, A_2 and bias vectors b_1, b_2 and the proof will proceed by induction. First, we note that for $h[t]$ in (21) to constitute a valid hidden state sequence according to Definition 1.1, the initial state needs to satisfy $h[-1] = 0_{F-1}$, which at the same time would constitute the base case $t = -1$ of the induction argument. The relation $h[-1] = 0_{F-1}$ now follows independently of the choices for A_1, A_2, b_1, b_2 and is simply by virtue of the index 0 not being contained in the set $\{1, \dots, F-1\}$ so that the truth function on the RHS of (21) yields the all-zeros vector. For the induction step, we assume that (21) holds for $t-1$ for some $t \geq 0$, i.e., $h_\ell[t-1] = \mathbb{1}_{\{(t \bmod F) = \ell\}}$, $\forall \ell \in \{1, \dots, F-1\}$. Next, we set

$$A_e := \begin{pmatrix} -1 & -1 & \dots & -1 \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix} \in \mathbb{R}^{F \times (F-1)}, \quad b_e := \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^F, \tag{22}$$

and define the sequence $e[t] \in \{0,1\}^F$ according to

$$e[t] := A_e h[t-1] + b_e. \tag{23}$$

Direct calculation now yields

$$e_\ell[t] = \mathbb{1}_{\{(t \bmod F)+1 = \ell\}}, \quad \forall \ell \in \{1, \dots, F\}. \tag{24}$$

That is, $e[t]$ indicates its argument t, modulo F to account for F-periodicity of $e^{2\pi i \frac{f}{F} t}$, through a one at the corresponding position in the period and, unlike the state vector, never equals the all-zeros vector. We now use $e[t]$ to construct an indicator function applied to the input signal. To this end, we first recall that

RNNs according to Definition 1.1 accept input signals in ℓ_∞, and set $C = \|x\|_{\ell_\infty}$. Next, for all $t \geq 0$, consider the sequence

$$\begin{aligned} \widetilde{x}[t] &= \rho\left(\begin{pmatrix} 1_F & 2CA_e \end{pmatrix}\begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix} + 2Cb_e - C1_F\right) \\ &= \rho\left(1_F\, x[t] + 2Ce[t] - C1_F\right), \end{aligned} \tag{25}$$

where we used (23). Equivalently, we can express (25) as

$$\widetilde{x}_\ell[t] = \rho(x[t] + 2C\mathbb{1}_{\{(t \bmod F)+1=\ell\}} - C) = (x[t] + C)\mathbb{1}_{\{(t \bmod F)+1=\ell\}}, \tag{26}$$

for $\ell \in \{1, \ldots, F\}$, where we made use of $|x[t]| \leq C$. We proceed to set

$$A_1 = \begin{pmatrix} 1_F & 2CA_e \\ 0_F & A_e \end{pmatrix} \in \mathbb{R}^{(2F)\times F}, \qquad b_1 = \begin{pmatrix} 2Cb_e - C1_F \\ b_e \end{pmatrix} \in \mathbb{R}^{2F}, \tag{27}$$

and $b_2 = 0_F$. Inserting into (5) yields

$$g[t] = \rho\left(A_1\begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix} + b_1\right) = \begin{pmatrix} \widetilde{x}[t] \\ e[t] \end{pmatrix},$$

where we employed (25) and (23), and used the fact that $\rho(e[t]) = e[t]$ as the entries of $e[t]$ equal either 0 or 1. Next, we let

$$\begin{aligned} A_h &= \begin{pmatrix} \mathbb{O}_{F-1} & 0_{F-1} & \mathbb{I}_{F-1} & 0_{F-1} \end{pmatrix}, \\ A_r &= \begin{pmatrix} \mathbb{I}_F & -C\,\mathbb{I}_F \end{pmatrix}, \\ A_o &= \begin{pmatrix} e^{2\pi i \frac{0}{F} f} & e^{2\pi i \frac{1}{F} f} & \ldots & e^{2\pi i \frac{F-1}{F} f} \end{pmatrix}. \end{aligned} \tag{28}$$

We are now ready to finalize the induction step. From $h[t] = A_h g[t]$ we get

$$h_\ell[t] = e_\ell[t] = \mathbb{1}_{\{(t \bmod F)+1=\ell\}} = \mathbb{1}_{\{((t+1) \bmod F)=\ell\}}, \; \forall \ell \in \{1, \ldots, F-1\}, \tag{29}$$

where we used that

$$(t \bmod F) + 1 = ((t+1) \bmod F), \tag{30}$$

for all t with $(t \bmod F) \neq F-1$. For t such that $(t \bmod F) = F-1$, the LHS of (30) equals F while the RHS is equal to 0; as the indices 0 and F do not occur in the set $\{1, \ldots, F-1\}$, we trivially have equality between the last two expressions in (29). This establishes (21) and thereby completes the induction step.

It remains to prove that the input-output relation of the RNN specified along the way is, indeed, given by (20). Using (28), (26), and (24) in $r[t] = A_r g[t]$, it follows that

$$\begin{aligned} r_\ell[t] &= \widetilde{x}_\ell[t] - Ce_\ell[t] \\ &= (x[t] + C)\mathbb{1}_{\{(t \bmod F)+1=\ell\}} - C\mathbb{1}_{\{(t \bmod F)+1=\ell\}} \\ &= x[t]\mathbb{1}_{\{(t \bmod F)+1=\ell\}}, \quad \forall \ell \in \{1, \ldots, F\}. \end{aligned} \tag{31}$$

The output signal is hence given by

$$\begin{aligned} y[t] &= A_o r[t] \\ &= \sum_{\ell=1}^{F} e^{2\pi i \frac{\ell-1}{F} f} x[t] \, \mathbb{1}_{\{(t \bmod F)+1=\ell\}} \\ &= x[t] \, e^{2\pi i \frac{(t \bmod F)}{F} f} \\ &= x[t] \, e^{2\pi i \frac{t}{F} f}, \end{aligned} \tag{32}$$

where, in the last step, we made use of the F-periodicity of $e^{2\pi i \frac{t}{F} f}$. This completes the proof. □

Having established the RNN realizations of the basic building blocks of the spreading representation (12), namely RNNs that realize time shifts (or, more generally, convolutions) and frequency shifts, we proceed to devise RNNs that implement weighted linear combinations of time-frequency shift operators. This entails showing that linear combinations of compositions of time shift RNNs and frequency shift RNNs are again RNNs. As opposed to feedforward networks where compositions and linear combinations trivially preserve the feedforward structure [18], this is not obvious in the RNN case. The basic idea underlying the construction provided next is hidden-state sharing across component networks, which not only preserves the RNN structure, but also leads to an economical—in terms of the number of nonzero weights—RNN realization.

Lemma 2.3 (RNNs can realize LTV systems)**.** *Let $D, F \in \mathbb{N}$ and consider the spreading function $\widetilde{S}_{\mathcal{L}}(\tau, f) \in \mathbb{C}$, $\tau \in \{0, \dots, D-1\}$, $f \in \{0, \dots, F-1\}$. There exists an RNN that realizes the input-output relation*

$$y[t] = \sum_{\tau=0}^{D-1} \sum_{f=0}^{F-1} \widetilde{S}_{\mathcal{L}}(\tau, f) x[t-\tau] e^{2\pi i \frac{f}{F} t}, \quad \forall t \geq 0. \tag{33}$$

Proof. There are two main components in the construction of the RNN realizing the desired input-output relation, namely the composition of time shift and frequency shift operators and weighted linear combinations thereof. The latter is easily realized through proper choice of the output layer weight matrix A_2, whereas the former requires more effort. Specifically, we will design the RNN such that its hidden state vector concatenates the hidden state vectors of the time shift and the frequency shift RNNs in Lemmas 2.1 and 2.2, respectively, and that this concatenated hidden state vector follows the hidden state evolution equations of the constituent time shift and frequency shift networks. Concretely, our goal will be to design the RNN such that its hidden state vector[1] is given by

$$h[t] = \begin{pmatrix} \dot{h}[t] \\ \ddot{h}[t] \end{pmatrix}, \tag{34}$$

[1]Note that the symbols $\dot{h}$ and $\ddot{h}$ do not refer to derivatives of h in any form.

where $\dot{h} \in \mathbb{R}^{(D-1)}$ corresponds to the hidden state of the convolution RNN from Lemma 2.1 particularized for pure time shifts, and $\ddot{h} \in \{0,1\}^{(F-1)}$ represents the hidden state of the frequency shift RNN in Lemma 2.2. The component vector sequences $\dot{h}[t]$ and $\ddot{h}[t]$ now need to follow the state evolution laws in (14) and (21), respectively, i.e.,

$$\dot{h}_\ell[t] = x[t-(\ell-1)], \quad \forall \ell \in \{1,\dots,D-1\} \tag{35}$$

$$\ddot{h}_\ell[t] = \mathbb{1}_{\{((t+1) \bmod F)=\ell\}}, \quad \forall \ell \in \{1,\dots,F-1\}, \tag{36}$$

both for all $t \geq 0$. The approach we follow will be as in the proofs of Lemmas 2.1 and 2.2, namely, we proceed by induction and in the process specify the network weight matrices and bias vectors to make the induction work out. The proof will be finalized by showing how the state vector $h[t]$ following (35) and (36) leads to the desired overall input-output relation by proper choice of A_2.

The base case $t = -1$ of the induction, i.e., $h[-1] = 0_{D+F-2}$, follows as the base case in Lemma 2.1 is by virtue of $x[t] = 0, \forall t < 0$, and that in Lemma 2.2 holds as a consequence of the definition of the state vector. Notably, for both components, $\dot{h}_\ell[t]$ and $\ddot{h}_\ell[t]$, the base case follows independently of the choices of the weight matrices and bias vectors. We remark that the base case also establishes that the initial state $h[-1]$ of the hidden state sequence in (34)—by virtue of being equal to the all zeros vector—conforms with Definition 1.1.

To establish the induction step, we will have to choose A_1, A_2, b_1, and b_2 appropriately. Concretely, we start by assuming that (35) and (36) hold for $t-1$ for some $t \geq 0$, and set

$$A_1 = \begin{array}{c} \\ \\ DF \\ \\ \\ 2D \\ F \end{array} \overbrace{\phantom{\begin{pmatrix} 1_F & 0_F & \dots & 0_F \end{pmatrix}}}^{D} \overbrace{\phantom{\begin{pmatrix} 2CA_e \end{pmatrix}}}^{F-1} \begin{pmatrix} 1_F & 0_F & \dots & 0_F & 2CA_e \\ 0_F & 1_F & \dots & 0_F & 2CA_e \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_F & 0_F & \dots & 1_F & 2CA_e \\ \hline \multicolumn{4}{c|}{\mathbb{I}_D} & \mathbb{0} \\ \multicolumn{4}{c|}{-\mathbb{I}_D} & \\ \hline \multicolumn{4}{c|}{\mathbb{0}} & A_e \end{pmatrix}, \quad b_1 = \begin{pmatrix} 2Cb_e - C1_F \\ 2Cb_e - C1_F \\ \vdots \\ 2Cb_e - C1_F \\ \hline \mathbb{0} \\ \hline b_e \end{pmatrix}, \tag{37}$$

where A_e, b_e are as defined in (22), $C = \|x\|_{\ell_\infty}$, and the unsubscripted $\mathbb{0}$ symbols stand for all zeros matrices of appropriate dimensions. The bias vector b_2 is chosen as $b_2 = 0_{D+F-1}$. The first D columns of A_1 operate on

$$\begin{pmatrix} x[t] \\ \dot{h}[t-1] \end{pmatrix} = \begin{pmatrix} x[t] \\ \vdots \\ x[t-(D-1)] \end{pmatrix} \tag{38}$$

and the last $F-1$ columns multiply $\ddot{h}[t-1]$. Further, A_1 is divided vertically into three parts. The first DF rows produce, for each time shift (including the shift by 0 time instants), a representation akin to (25), the middle $2D$ rows

correspond to (16) in the time shift RNN construction, and the last F rows pertain to the frequency shift RNN, specifically to (23). It is hence natural to think of $g[t]$ from (5) in three parts according to

$$g[t] = \rho\left(A_1 \begin{pmatrix} x[t] \\ \dot{h}[t-1] \\ \ddot{h}[t-1] \end{pmatrix} + b_1\right) = \begin{array}{c} \\ DF \updownarrow \\ \\ \\ 2D \updownarrow \\ F \updownarrow \end{array} \left(\begin{array}{c} \widetilde{x}[t,0] \\ \widetilde{x}[t,1] \\ \vdots \\ \widetilde{x}[t,D-1] \\ \hline \dot{g}[t] \\ \hline e[t] \end{array}\right), \tag{39}$$

where, following the steps leading to (26) in Lemma 2.2, the vectors $\widetilde{x}[t,\tau]$, $\tau \in \{0,\dots,D-1\}$, are obtained as

$$\widetilde{x}_\ell[t,\tau] = (x[t-\tau]+C)\mathbb{1}_{\{(t \bmod F)+1=\ell\}}, \tag{40}$$

$\dot{g}[t]$ equals $g[t]$ in (18) with $L=D$, and $e[t]$ is as in (24).

We proceed to specify A_h, the submatrix of A_2, which maps to the next hidden state according to (4), as

$$A_h = \begin{array}{c} \\ D-1 \updownarrow \\ F-1 \updownarrow \end{array} \begin{array}{c} \begin{array}{ccccc} \overset{DF}{\longleftrightarrow} & & \overset{2D}{\longleftrightarrow} & & \overset{F}{\longleftrightarrow} \end{array} \\ \left(\begin{array}{c|cccc|cc} \mathbb{O} & \mathbb{I}_{D-1} & 0_{D-1} & -\mathbb{I}_{D-1} & 0_{D-1} & \mathbb{O} & \\ \hline \mathbb{O} & \multicolumn{4}{c|}{\mathbb{O}} & \mathbb{I}_{F-1} & 0_{F-1} \end{array}\right) \end{array}, \tag{41}$$

where again the unsubscripted $\mathbb{O}$ symbols refer to all-zeros matrices of appropriate dimensions. Carrying out the state transition for the concatenated hidden state vector (34) according to (4) with A_h in (41) and $g[t]$ in (39) yields (35) directly and (36) upon using the last identity in (29).

Next, with the $F \times F$ (unnormalized) DFT matrix $[A_F]_{f,n} = e^{2\pi i \frac{f}{F} n}$, $f \in \{0,\dots,F-1\}$, $n \in \{0,\dots,F-1\}$, we define the vectors

$$\widehat{x}[t,\tau] := \begin{pmatrix} A_F & -CA_F \end{pmatrix} \begin{pmatrix} \widetilde{x}[t,\tau] \\ e[t] \end{pmatrix}, \quad \tau \in \{0,\dots,D-1\}, \tag{42}$$

and note, by direct calculation, that

$$\widehat{x}_f[t,\tau] = x[t-\tau]e^{2\pi i \frac{f-1}{F} t}, \quad f \in \{1,\dots,F\}. \tag{43}$$

Now we stack the frequency-shifted versions of $x[t-\tau]$ in (43) in the virtual representation sequence $r[t]$ by setting

$$A_r = \begin{array}{c} \\ DF \updownarrow \\ \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccccc} & & \overset{DF}{\longleftrightarrow} & & & \overset{2D}{\longleftrightarrow} & \overset{F}{\longleftrightarrow} \end{array} \\ \left(\begin{array}{cccc|c|c} A_F & \mathbb{O} & \dots & \mathbb{O} & \mathbb{O} & -CA_F \\ \mathbb{O} & A_F & \dots & \mathbb{O} & \mathbb{O} & -CA_F \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \mathbb{O} & \mathbb{O} & \dots & A_F & \mathbb{O} & -CA_F \end{array}\right) \end{array}, \tag{44}$$

which yields

$$r[t] = A_r g[t] = \begin{pmatrix} \widehat{x}[t,0] \\ \widehat{x}[t,1] \\ \vdots \\ \widehat{x}[t,D-1] \end{pmatrix}. \tag{45}$$

This finalizes the first part of the construction, namely the composition of time shifts and frequency shifts according to (43). We are left with the weighted superposition of the time-frequency-shifted versions of $x[t]$ implementing the input-output relation (33). To this end, we set, for $\tau \in \{0, \dots, D-1\}$,

$$\widetilde{S}_{\mathcal{L}}(\tau, \cdot) := \begin{pmatrix} \widetilde{S}_{\mathcal{L}}(\tau, 0) \\ \widetilde{S}_{\mathcal{L}}(\tau, 1) \\ \vdots \\ \widetilde{S}_{\mathcal{L}}(\tau, F-1) \end{pmatrix} \in \mathbb{C}^F \tag{46}$$

and take

$$A_o = \begin{pmatrix} \widetilde{S}_{\mathcal{L}}(0,\cdot)^T & \widetilde{S}_{\mathcal{L}}(1,\cdot)^T & \dots & \widetilde{S}_{\mathcal{L}}(D-1,\cdot)^T \end{pmatrix}, \tag{47}$$

which results in

$$y[t] = A_o r[t] = \sum_{\tau=0}^{D-1} \widetilde{S}_{\mathcal{L}}(\tau, \cdot)^T \widehat{x}[t, \tau] \tag{48}$$

$$= \sum_{\tau=0}^{D-1} \sum_{f=0}^{F-1} \widetilde{S}_{\mathcal{L}}(\tau, f) x[t-\tau] e^{2\pi i \frac{f}{F} t}, \tag{49}$$

as desired. □

We note that the RNN constructed in Lemma 2.3 has $\mathcal{O}(DF)$ non-zero weights, i.e., the "size" of the network is proportional to the spread of the system it is to realize. This insight is based on the fact that the virtual representation sequence $r[t]$ is never actually manifested. Hence, by (3) only the product $A_o A_r$ has to be stored instead of the (bigger) individual matrices A_o and A_r. Finally, we remark that the magnitudes of the RNN weights in the proof of Lemma 2.3 depend on the ℓ_∞-norm of the inputs the RNN accepts.

3. Metric Entropy of LTI Systems

Having established that RNNs can universally realize linear dynamical systems with network size proportional to the spread of the system, we proceed to develop a deepened and more quantitative theory along those lines. Specifically, we shall be interested in the approximation of classes of linear dynamical systems to within a prescribed worst-case (within the class) error ϵ through RNNs

that can be specified by bitstrings of finite length. Of particular interest will be the scaling behavior of the required length of the bitstring as a function of ϵ and, in particular, whether RNNs can achieve the fundamental limit—over all possible system approximation methods—on this scaling behavior. Answering this question requires the concept of metric entropy of linear systems, a topic originating from control theory [12, 4]. The aim of the present section is to introduce this concept, with the presentation geared towards our purposes. We restrict ourselves to LTI systems for conceptual reasons.

A linear dynamical system is time-invariant if the operator kernel $k[t,\tau]$ in (10) is a function of τ only, i.e., the input-output relation of the system is given by the convolution of the input signal $x[\cdot]$ with the impulse response $k[\cdot]$ according to

$$(\mathcal{L}x)[t] = \sum_{\tau=0}^{\infty} k[\tau]x[t-\tau] =: (k * x)[t], \tag{50}$$

where, as before, we assume that $x[t] = 0$, $k[t] = 0$, for $t < 0$, that is we consider one-sided input signals and causal systems. We shall frequently make use of the one-sided $\mathcal{Z}$-transform for ℓ^2-signals defined as[2]

$$(\mathcal{Z}\{x[\cdot]\})(z) = \sum_{t=0}^{\infty} x[t]z^t, \quad |z| < 1. \tag{51}$$

Whenever there is no source of ambiguity, we shall use capital letters to denote the $\mathcal{Z}$-transform according to $X(z) = (\mathcal{Z}\{x[\cdot]\})(z)$. Next, we note the well-known relation

$$(\mathcal{Z}\{(\mathcal{L}x)[\cdot]\})(z) = (\mathcal{Z}\{(k * x)[\cdot]\})(z) = K(z) \cdot X(z), \tag{52}$$

where $K(z) := (\mathcal{Z}\{k[\cdot]\})(z)$ is commonly referred to as the system's transfer function.

We proceed to establish the concept of metric entropy of classes of LTI systems largely following [12, 4]. On a conceptual level, this complexity notion allows to formulate answers to the following question: *Given a class of LTI systems, how many bits of information do we need to identify a specific system in the class to within a prescribed error?* To formalize matters, we start by defining the metric entropy of general sets.

Definition 3.1 ([32]). *Let $(\mathcal{X}, \rho)$ be a metric space. An ϵ-covering of a compact set $\mathcal{C} \subseteq \mathcal{X}$ with respect to the metric ρ is a set of points $\{x_1, \ldots, x_N\} \subset \mathcal{C}$ such that for each $x \in \mathcal{C}$, there exists an $i \in [1, N]$ so that $\rho(x, x_i) \leq \epsilon$. The ϵ-covering number $N(\epsilon; \mathcal{C}, \rho)$ is the cardinality of a smallest ϵ-covering of $\mathcal{C}$ and $\mathcal{E}(\epsilon; \mathcal{C}, \rho) := \log_2(N(\epsilon; \mathcal{C}, \rho))$ is the metric entropy of $\mathcal{C}$.*

[2]Note the positive exponents of z in the definition. This convention is chosen to maintain consistency with Definition 3.2 below adopted from [4].

As LTI systems are uniquely determined by their impulse response, we shall incarnate the concept of "classes of LTI systems" by considering compact sets of impulse responses. More specifically, motivated by [4], we consider systems with exponentially decaying impulse response, that is, the set of LTI systems characterized by

$$\mathcal{C}(a,b) := \{\mathcal{L} \mid |k_{\mathcal{L}}[t]| \leq ae^{-bt},\, \forall t \geq 0\}, \quad a,b>0, \tag{53}$$

where $k_{\mathcal{L}}[\cdot]$ denotes the impulse response of the system $\mathcal{L}$. The constants b and a quantify the decay behavior of the system memory. Note that the set $\mathcal{C}(a,b)$ encompasses exponentially decaying impulse responses of arbitrary decay rate according to

$$\mathcal{C}(a,\log(1/\beta)) = \{\mathcal{L} \mid |k_{\mathcal{L}}[t]| \leq a\beta^{t},\, \forall t \geq 0\}, \quad a>0,\, \beta\in(0,1). \tag{54}$$

Next, we equip the ambient space $\mathcal{X}$ with a suitable metric which quantifies the distance between LTI systems, or equivalently their impulse responses. To this end, we first define Hardy spaces and norms of transfer functions as follows.

Definition 3.2 ([33, Chapter 17])**.** *For the transfer function $K(z)$, we define the Hardy norms*

$$\|K\|_{\mathcal{H}^2} := \sqrt{\sup_{r<1}\frac{1}{2\pi}\int_0^{2\pi}|K(re^{i\theta})|^2 d\theta}, \tag{55}$$

$$\|K\|_{\mathcal{H}^\infty} := \sup_{|z|<1}|K(z)|. \tag{56}$$

The corresponding Hardy spaces are given by $\mathcal{H}^2 = \{K(\cdot)\,|\,\|K\|_{\mathcal{H}^2}<\infty\}$ and $\mathcal{H}^\infty = \{K(\cdot)\,|\,\|K\|_{\mathcal{H}^\infty}<\infty\}$.

The distance between the LTI systems $\mathcal{L}$ and $\mathcal{L}'$ with transfer functions $K(z)$ and $K'(z)$, respectively, both in $\mathcal{H}^\infty$, is now defined as

$$\rho(\mathcal{L},\mathcal{L}') := \|K-K'\|_{\mathcal{H}^\infty}. \tag{57}$$

The following result relates $\rho(\mathcal{L},\mathcal{L}')$ to distance—in terms of squared error—in the system output space.

Theorem 3.1. *Let $\mathcal{L}$ and $\mathcal{L}'$ be LTI systems with corresponding transfer functions $K(z)$ and $K'(z)$, both in $\mathcal{H}^\infty$. It holds that*

$$\rho(\mathcal{L},\mathcal{L}') = \sup_{\|x\|_{\ell^2}=1}\|\mathcal{L}x-\mathcal{L}'x\|_{\ell^2}.$$

Proof. The proof is established through the following chain of arguments

$$\rho(\mathcal{L}, \mathcal{L}') = \|K - K'\|_{\mathcal{H}^\infty} \tag{58}$$

$$= \sup_{X \in \mathcal{H}^2} \frac{\|(K - K')X\|_{\mathcal{H}^2}}{\|X\|_{\mathcal{H}^2}} \tag{59}$$

$$= \sup_{\|X\|_{\mathcal{H}^2}=1} \|(K - K')X\|_{\mathcal{H}^2} \tag{60}$$

$$= \sup_{\|x\|_{\ell^2}=1} \|k * x - k' * x\|_{\ell^2} \tag{61}$$

$$= \sup_{\|x\|_{\ell^2}=1} \|\mathcal{L}x - \mathcal{L}'x\|_{\ell^2}, \tag{62}$$

where (59) follows from Theorem B.2 upon noting that $K - K' \in \mathcal{H}^\infty$ by application of the triangle inequality and (61) is by Theorem B.1 together with (52), where k and k' denote the impulse responses of the systems $\mathcal{L}$ and $\mathcal{L}'$, respectively. □

Theorem 3.1 shows that identifying a reference system $\mathcal{L}$ to within error $\rho(\mathcal{L}, \mathcal{L}') = \epsilon$ guarantees that the estimated system $\mathcal{L}'$ results in output signals that deviate no more than ϵ—in ℓ_2-norm—from the output that would be produced by the reference system $\mathcal{L}$.

We are now ready to recall a result due to Zames and Owen [4] which quantifies the metric entropy of $\mathcal{C}(a, b)$ with respect to the distance measure $\rho(\mathcal{L}, \mathcal{L}')$.

Theorem 3.2 ([4])**.** *Let $a, b > 0$ and consider the set*

$$\mathcal{C}(a, b) = \{\mathcal{L} \mid |k_{\mathcal{L}}[t]| \leq ae^{-bt}, \forall t \geq 0\}.$$

The metric entropy of $\mathcal{C}(a, b)$ with respect to

$$\rho(\mathcal{L}, \mathcal{L}') = \|K - K'\|_{\mathcal{H}^\infty} \tag{63}$$

satisfies

$$\mathcal{E}(\epsilon; \mathcal{C}(a, b), \rho) \sim \frac{1}{b}\left(\log\left(\frac{a}{\epsilon}\right)\right)^2. \tag{64}$$

Note that for all systems $\mathcal{L} \in \mathcal{C}(a, b)$, the transfer function is in $\mathcal{H}^\infty$, which by application of the triangle inequality, shows that (63) is well-defined. The proof of Theorem 3.2 proceeds by establishing lower and upper bounds on metric entropy according to

$$\frac{1}{b}\left(\log\left(\frac{a}{\epsilon}\right)\right)^2 - o\left(\left(\log\left(\frac{a}{\epsilon}\right)\right)^2\right) \leq \mathcal{E}(\epsilon; \mathcal{C}(a, b), \rho) \tag{65}$$

and

$$\mathcal{E}(\epsilon; \mathcal{C}(a, b), \rho) \leq \frac{1}{b}\left(\log\left(\frac{a}{\epsilon}\right)\right)^2 + o\left(\left(\log\left(\frac{a}{\epsilon}\right)\right)^2\right), \tag{66}$$

respectively, where $g(\epsilon) = o(f(\epsilon))$ stands for $\lim_{\epsilon\to 0}\left|\frac{g(\epsilon)}{f(\epsilon)}\right| = 0$. The lower bound (65) is derived through an embedding argument and the upper bound is obtained by constructing an explicit ϵ-covering [4].

We note that the metric entropy in (64) scaling according to $(\log(1/\epsilon))^2$ shows that the set $\mathcal{C}(a,b)$ is not overly massive. Richer function classes such as the set of all Lipschitz functions from $[0,1]^d$ to $\mathbb{R}$ with a given Lipschitz constant have metric entropy scaling according to ϵ^{-d} [32]. Moreover, it follows from (64) that impulse responses of slower (exponential) decay, i.e., with smaller b, are more complex to describe.

4. Optimal Covering Through Quantized RNNs

We are now in a position to state the second central result of this paper. Specifically, we show that RNNs with suitably quantized weights provide an optimal—in the sense of Theorem 3.2—ϵ-covering of $\mathcal{C}(a,b)$ with respect to the metric $\rho(\mathcal{L},\mathcal{L}')$. Operationally, this means that RNNs can optimally—in the sense of metric entropy—learn (or identify) the class of LTI systems with exponentially decaying impulse response. This result quantifies what is possible in principle and thereby provides a benchmark against which practical learning algorithms can be assessed.

The results presented so far apply to RNNs with real-valued weights. Constructing an optimal covering through RNNs requires, however, encoding of the approximating RNNs into bitstrings of length scaling in the approximation error ϵ according to (64). Now, there are two components that go into such an encoding of RNNs, namely the values of the nonzero weights in the matrices A_1, A_2 and the vectors b_1, b_2 and the locations of these weights, i.e., the topology of the network. The former requires quantization of the weights at a resolution that scales adequately in ϵ. We shall see below that encoding the topology is a non-issue. The main technical problem hence resides in ensuring that weight quantization in the approximating RNN can be effected at a resolution that allows metric entropy optimality—in terms of the covering realized—and at the same time guarantees that the resulting error incurred at the system output consorts with the desired approximation accuracy.

We start by defining an RNN weight quantization scheme.

Definition 4.1 (Quantized weights)**.** *For $\delta > 0$, define the set*

$$\mathbb{S}_\delta := \{\delta k \mid k \in \mathbb{Z}\}. \tag{67}$$

We say that an RNN has δ-quantized weights if all its weights are in $\mathbb{S}_\delta \cup \{-1, 1\}$. Further, define the quantization function

$$S_\delta(w) := sign(w)\left\lfloor \frac{|w|}{\delta} \right\rfloor \delta, \quad w \in \mathbb{R}. \tag{68}$$

Clearly, we have $|S_\delta(w) - w| \leq \delta$ and $|S_\delta(w)| \leq |w|$.

The main idea underlying the proof of the optimal RNN covering result builds on the approximation of the exponentially decaying impulse responses in $\mathcal{C}(a,b)$ through finite impulse response (FIR) filters of suitable length and with suitably quantized impulse response coefficients. In order to quantify the approximation error—in terms of $\rho(\mathcal{L},\mathcal{L}') = \|K - K'\|_{\mathcal{H}^\infty}$—resulting from this truncation and coefficient quantization, we will need the following simple technical result.

Lemma 4.1. *Consider the LTI systems with impulse responses $k[\cdot]$ and $\widetilde{k}[\cdot]$ and corresponding transfer functions $K(z)$ and $\widetilde{K}(z)$, both in $\mathcal{H}^\infty$. We have*

$$\|K(\cdot) - \widetilde{K}(\cdot)\|_{\mathcal{H}^\infty} \leq \sum_{t=0}^{\infty} |k[t] - \widetilde{k}[t]|.$$

Proof. The proof is by the following chain of relations

$$\begin{aligned}
\|K(\cdot) - \widetilde{K}(\cdot)\|_{\mathcal{H}^\infty} &= \sup_{|z|<1} \left| \sum_{t=0}^{\infty} k[t] z^t - \sum_{t=0}^{\infty} \widetilde{k}[t] z^t \right| \\
&= \sup_{|z|<1} \left| \sum_{t=0}^{\infty} (k[t] - \widetilde{k}[t]) z^t \right| \\
&\leq \sup_{|z|<1} \sum_{t=0}^{\infty} |k[t] - \widetilde{k}[t]| |z|^t \\
&= \sum_{t=0}^{\infty} |k[t] - \widetilde{k}[t]|.
\end{aligned}$$

□

We are now ready to state the main result.

Theorem 4.1 (RNNs are metric-entropy-optimal)**.** *Consider an LTI system $\mathcal{L}$ with impulse response satisfying $|k[t]| \leq a\, e^{-bt}$, $\forall t \geq 0$, for some $a, b > 0$, and corresponding transfer function $K(z)$. For every $\epsilon > 0$, with*

$$M := \left\lceil \frac{1}{b} \log\left(\frac{a}{\epsilon}\right) + \frac{1}{b} \log\left(\frac{2}{1 - e^{-b}}\right) \right\rceil,$$

$\mathcal{L}$ can be approximated by a $\delta := \frac{\epsilon}{2M}$-quantized RNN $\widetilde{\mathcal{R}}$—of hidden state size $M-1$—realizing an FIR filter with transfer function $\widetilde{K}(z)$ such that

$$\|K(\cdot) - \widetilde{K}(\cdot)\|_{\mathcal{H}^\infty} \leq \epsilon.$$

Moreover, $\widetilde{\mathcal{R}}$ can be encoded in a uniquely decodable fashion, provided that both encoder and decoder know a and b, using no more than

$$\frac{1}{b}\left(\log\left(\frac{a}{\epsilon}\right)\right)^2 + o\left(\left(\log\left(\frac{1}{\epsilon}\right)\right)^2\right)$$

bits.

Proof. The idea of the proof is to δ-quantize the suitably truncated impulse response $k[t]$ corresponding to $\mathcal{L}$, which is then realized (exactly) by an RNN, denoted as $\widetilde{\mathcal{R}}$, following the construction in Lemma 2.1. Concretely, we choose the truncated quantized impulse response according to $\widetilde{k}[t] := S_\delta(k[t])\mathbb{1}_{\{t\leq(M-1)\}}$, denote the corresponding transfer function by $\widetilde{K}(z)$, and then use Lemma 4.1 to bound

$$\begin{aligned}
\|K(\cdot)-\widetilde{K}(\cdot)\|_{\mathcal{H}^\infty} &\leq \sum_{t=0}^{\infty}|k[t]-\widetilde{k}[t]| && (69)\\
&= \sum_{t=0}^{M-1}|k[t]-\widetilde{k}[t]| + \sum_{t=M}^{\infty}|k[t]| && (70)\\
&\leq M\delta + \sum_{t=M}^{\infty} ae^{-bt} && (71)\\
&= M\delta + a\,\frac{e^{-bM}}{1-e^{-b}} && (72)\\
&\leq M\delta + a\,\frac{e^{-\log\left(\frac{2a}{\epsilon(1-e^{-b})}\right)}}{1-e^{-b}} && (73)\\
&= M\delta + a\,\frac{\frac{\epsilon(1-e^{-b})}{2a}}{1-e^{-b}} && (74)\\
&= M\frac{\epsilon}{2M}+\frac{\epsilon}{2}=\epsilon, && (75)
\end{aligned}$$

where in (72) we used $\sum_{n=M}^{\infty} r^n = \frac{r^M}{1-r}$, for $|r|<1$.

It remains to establish that the RNN realizing the FIR system with impulse response $\widetilde{k}[t]$ can be encoded in a uniquely decodable fashion into a bitstring of length consorting with covering optimality according to (64). As mentioned earlier, encoding an RNN in a bitstring requires specifying its topology and quantized weights, both in binary form. We first convince ourselves that the topology of the RNN realizing $\widetilde{k}[t]$ is fixed and hence does not need to be encoded. This follows by recognizing that in the RNN construction in the proof of Lemma 2.1 the quantities A_1, A_h, A_r, b_1, and b_2 are all independent of the impulse response of the FIR system to be realized and only A_o depends on the impulse response according to $A_o = \widetilde{k}^T$. The locations of the nonzero entries in the weight matrices and bias vectors of the approximating RNN hence need not be encoded. This leaves us with having to represent the M quantized impulse response coefficients $\widetilde{k}[t]$ through a bitstring of length scaling in ϵ such that covering optimality is attained. To this end, we first note that from Definition 4.1, we get

$$|\widetilde{k}[t]| = |S_\delta(k[t])| \leq |k[t]| \leq ae^{-bt}, \qquad \forall t\in\{0,\ldots,M-1\}.$$

The quantized impulse response coefficients hence satisfy

$$\widetilde{k}[t] \in \mathbb{S}_\delta \cap [-ae^{-bt}, ae^{-bt}], \qquad \forall t\in\{0,\ldots,M-1\},$$

and can therefore be stored using at most $\left\lceil \log_2\left(\frac{ae^{-bt}}{\delta}\right)\right\rceil + 1$ bits. As a and b are known to the encoder and the decoder by assumption, we can encode the quantized impulse response coefficients into a uniquely decodable bitstring simply by allocating $\left\lceil \log_2\left(\frac{ae^{-bt}}{\delta}\right)\right\rceil + 1$ bits to each coefficient, concatenating the corresponding binary labels (filled up with zeros if they are of smaller than the alloted length) and have the decoder read out the labels sequentially to deliver the corresponding points in $\mathbb{S}_\delta$.

It remains to establish that the length of the bitstring just constructed conforms with (64). To this end, we first upper-bound the length of the bitstring according to

$$\sum_{t=0}^{M-1}\left(\left\lceil \log_2\left(\frac{ae^{-bt}}{\delta}\right)\right\rceil + 1\right) \tag{76}$$

$$\leq \sum_{t=0}^{M-1}\left(\log_2\left(\frac{a}{\delta}\right) + \log_2(e^{-bt}) + 2\right) \tag{77}$$

$$= 2M + M\gamma\log\left(\frac{a}{\delta}\right) + \sum_{t=0}^{M-1}\gamma\log\left(e^{-bt}\right) \tag{78}$$

$$= 2M + M\gamma\log\left(\frac{a}{\delta}\right) - b\gamma\sum_{t=0}^{M-1} t \tag{79}$$

$$= 2M + M\gamma\log\left(\frac{2aM}{\epsilon}\right) - b\gamma\frac{M(M-1)}{2} \tag{80}$$

$$= M\left(\gamma\log\left(\frac{2aM}{\epsilon}\right) - b\gamma\frac{(M-1)}{2} + 2\right) \tag{81}$$

$$= M\left(\gamma\log\left(\frac{a}{\epsilon}\right) - M\frac{b\gamma}{2} + \gamma\log(M) + \frac{b\gamma}{2} + 3\right), \tag{82}$$

where we used $\log_2(x) = \gamma\log(x)$ with $\gamma := \log_2(e)$ and in (80) we employed $\delta = \frac{\epsilon}{2M}$. Next, we note from the definition of M that

$$\frac{1}{b}\log\left(\frac{a}{\epsilon}\right) + K_1(b) \leq M \leq \frac{1}{b}\log\left(\frac{a}{\epsilon}\right) + K_1(b) + 1, \tag{83}$$

with $K_1(b) := \frac{1}{b}\log\left(\frac{2}{1-e^{-b}}\right)$. Using (83) in (82) allows us to further upper-

bound (82) as follows:

$$M\left(\gamma\log\left(\frac{a}{\epsilon}\right)-M\frac{b\gamma}{2}+\gamma\log(M)+\frac{b\gamma}{2}+3\right) \tag{84}$$

$$\leq M\left(\gamma\log\left(\frac{a}{\epsilon}\right)-\frac{1}{b}\log\left(\frac{a}{\epsilon}\right)\frac{b\gamma}{2}-K_1(b)\frac{b\gamma}{2}+\gamma\log(M)+\frac{b\gamma}{2}+3\right) \tag{85}$$

$$= M\left(\frac{\gamma}{2}\log\left(\frac{a}{\epsilon}\right)+\gamma\log(M)+K_2(b)\right) \tag{86}$$

$$\leq \left(\frac{1}{b}\log\left(\frac{a}{\epsilon}\right)+K_1(b)+1\right)\left(\frac{\gamma}{2}\log\left(\frac{a}{\epsilon}\right)+\gamma\log(M)+K_2(b)\right) \tag{87}$$

$$= \frac{\gamma}{2b}\left(\log\left(\frac{a}{\epsilon}\right)\right)^2+\frac{\gamma}{b}\log\left(\frac{a}{\epsilon}\right)\log(M)+\frac{1}{b}\log\left(\frac{a}{\epsilon}\right)K_2(b) \tag{88}$$

$$+(K_1(b)+1)\frac{\gamma}{2}\log\left(\frac{a}{\epsilon}\right)+(K_1(b)+1)\gamma\log(M)+(K_1(b)+1)K_2(b) \tag{89}$$

$$= \frac{\gamma}{2b}\left(\log\left(\frac{a}{\epsilon}\right)\right)^2+\frac{\gamma}{b}\log(M)\log\left(\frac{1}{\epsilon}\right)+K_3(b)\log\left(\frac{1}{\epsilon}\right) \tag{90}$$

$$+K_4(a,b)\log(M)+K_5(a,b) \tag{91}$$

$$\leq \frac{1}{b}\left(\log\left(\frac{a}{\epsilon}\right)\right)^2+o\left(\left(\log\left(\frac{1}{\epsilon}\right)\right)^2\right), \tag{92}$$

where $K_2(b) := -K_1(b)\frac{b\gamma}{2}+\frac{b\gamma}{2}+3$, $K_3(b) := \frac{K_2(b)}{b}+\frac{(K_1(b)+1)\gamma}{2}$, $K_4(a,b) := \gamma(K_1(b)+1)+\log(a)\frac{\gamma}{b}$, and $K_5(a,b) := (K_1(b)+1)K_2(b)+K_3(b)\log(a)$. The last inequality follows from $\gamma \leq 2$ and $\log(M) = o(\log(\epsilon^{-1}))$. □

We conclude by noting that the dependence of the hidden state size and the weight quantization resolution of the approximating RNN in Theorem 4.1 on the parameters a, b, ϵ reflects that more complex sets $\mathcal{C}(a,b)$ and smaller target approximation error require larger hidden state size and higher quantization resolution.

5. Metric-Entropy-Optimal Learning of Linear Difference Equations

Over the last few years a significant body of literature on deep neural network learning of the solutions of parametric PDEs was developed [34, 35, 36]. More specifically, this line of work is concerned with learning the map taking the right-hand side of the PDE and its parameters to the solution. We next suggest an alternative viewpoint in its simplest possible mathematical incarnation, namely that of learning differential, in fact difference, equations themselves. From a practical perspective this amounts to identifying the dynamics of physical, biological, mechanical, or chemical processes from observed input-output traces [37].

We consider linear difference equations with constant coefficients given by

$$\sum_{j=0}^{P} b_j y[t-j] = \sum_{i=0}^{Q} a_i x[t-i], \tag{93}$$

where $P, Q \in \mathbb{N}, b_j, a_i \in \mathbb{R}$, and $x[t]$ and $y[t]$ designate the input and the output, respectively, of the dynamical system characterized by the difference equation. Difference equations of the form (93) correspond to LTI systems with rational transfer functions. Concretely application of Lemma B.1 yields $Y(z) = K(z)X(z)$ with

$$K(z) = \frac{\sum_{i=0}^{Q} a_i z^i}{\sum_{j=0}^{P} b_j z^j}. \tag{94}$$

Learning of the difference equation (93) from input-output traces, i.e., determining the coefficients a_i and b_j in (93) based on the outputs $y[\cdot]$ corresponding to given inputs $x[\cdot]$ hence amounts to identifying the LTI system with transfer function (94). We first convince ourselves that RNNs can, in principle, realize systems with rational transfer functions, thereby extending Lemma 2.1 where this was shown for polynomial transfer functions.

Theorem 5.1 (RNNs can realize all rational transfer functions)**.** *Let $\mathcal{L}$ be an LTI system with transfer function*

$$K(z) = \frac{\sum_{i=0}^{Q} a_i z^i}{\sum_{j=0}^{P} b_j z^j}, \tag{95}$$

where $Q, P \in \mathbb{N}$ and $a_i, b_j \in \mathbb{R}$ with $b_0 \neq 0$. Then, there exists an RNN that realizes $\mathcal{L}$ exactly.

Proof. The proof will be effected by constructing the RNN realizing $\mathcal{L}$. We start by noting that application of the inverse $\mathcal{Z}$-transform and the time shift property Lemma B.1 to

$$Y(z) = X(z)K(z)$$

yields the difference equation

$$y[t] = \sum_{i=0}^{Q} c_i x[t-i] + \sum_{j=1}^{P} d_j y[t-j], \tag{96}$$

with $c_i = \frac{a_i}{b_0}$ and $d_j = -\frac{b_j}{b_0}$. In contrast to the construction in Lemma 2.1 which realizes a forward part only, here given by $\sum_{i=0}^{Q} c_i x[t-i]$, we will need to account for both the forward part and the backward part $\sum_{j=1}^{P} d_j y[t-j]$. This will be accomplished by choosing the RNN weight matrices A_1, A_2 and bias vectors b_1, b_2 such that the hidden state vector $h[t]$ contains the last Q values of the input signal $x[\cdot]$ and the last P values of the output signal $y[\cdot]$ according to

$$h[t] = \begin{pmatrix} x[t] \\ \vdots \\ x[t-(Q-1)] \\ \hline y[t] \\ \vdots \\ y[t-(P-1)] \end{pmatrix} \begin{matrix} \updownarrow Q \\ \\ \\ \\ \updownarrow P \\ \\ \end{matrix}. \tag{97}$$

Then, based on (97), we establish that these choices also yield the output signal as desired. We commence by specifying the RNN weights and proving (97) by induction. By slight abuse of notation, we let the vector c have a zero-th entry and define

$$\begin{aligned} c &:= \begin{pmatrix} c_0 & c_1 & \dots & c_Q \end{pmatrix}^T \in \mathbb{R}^{Q+1}, \\ d &:= \begin{pmatrix} d_1 & \dots & d_P \end{pmatrix}^T \in \mathbb{R}^P, \end{aligned} \tag{98}$$

and the matrix

$$W := \begin{array}{c} \\ 1 \\ Q \\ 1 \\ P-1 \end{array} \begin{array}{c} \overset{Q+1}{\longleftrightarrow} \\ \left(\begin{array}{c|c} c^T & d^T \\ \mathbb{I}_Q \quad 0_Q & \mathbb{O} \\ c^T & d^T \\ \mathbb{O} & \mathbb{I}_{P-1} \quad 0_{P-1} \end{array}\right) \end{array} \overset{P}{\longleftrightarrow}, \tag{99}$$

where the unsubscripted symbols $\mathbb{O}$ stand for all-zeros matrices of appropriate dimensions. The network weights are now chosen according to

$$A_1 = \begin{pmatrix} \mathbb{I}_{P+Q+1} \\ -\mathbb{I}_{P+Q+1} \end{pmatrix}, \qquad A_2 = W \begin{pmatrix} \mathbb{I}_{P+Q+1} & -\mathbb{I}_{P+Q+1} \end{pmatrix}, \tag{100}$$

and $b_1 = 0_{2P+2Q+2}$, $b_2 = 0_{P+Q+1}$. With (1) and thanks to (15), this yields

$$\begin{pmatrix} y[t] \\ h[t] \end{pmatrix} = W \begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix}, \quad \forall t \geq 0. \tag{101}$$

We are now ready to establish (97) by induction. First, we note that for $h[t]$ in (97) to constitute a valid hidden state sequence according to Definition 1.1, the initial state needs to satisfy $h[-1] = 0_{Q+P}$. This, indeed, follows from the assumption $x[t] = y[t] = 0$, $\forall t < 0$, and, in turn, also yields the base case $t = -1$ of the induction argument. To establish the induction step, we assume that (97) holds for $t-1$ for some $t \geq 0$. Next, let $h_{1:Q}[t] \in \mathbb{R}^Q$ denote the subvector of $h[t]$ containing the entries 1 through Q. It now follows from (101) and (99) that

$$h_{1:Q}[t] = \begin{pmatrix} \mathbb{I}_Q & 0_Q \end{pmatrix} \begin{pmatrix} x[t] \\ x[t-1] \\ \vdots \\ x[t-Q] \end{pmatrix} = \begin{pmatrix} x[t] \\ \vdots \\ x[t-(Q-1)] \end{pmatrix}$$

and

$$\begin{aligned} h_{Q+1}[t] &= \begin{pmatrix} c^T & d^T \end{pmatrix} \begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix} \\ &= \begin{pmatrix} c^T & d^T \end{pmatrix} \begin{pmatrix} x[t] \\ \vdots \\ x[t-Q] \\ y[t-1] \\ \vdots \\ y[t-P] \end{pmatrix} \\ &= \sum_{i=0}^{Q} c_i x[t-i] + \sum_{j=1}^{P} d_j y[t-j] = y[t], \end{aligned} \tag{102}$$

where we used (96). The proof of the induction step is now completed upon noting that

$$\begin{aligned} h_{(Q+2):(Q+P)}[t] &= W \begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix} \\ &= \begin{pmatrix} \mathbb{I}_{P-1} & 0_{P-1} \end{pmatrix} \begin{pmatrix} y[t-1] \\ \vdots \\ y[t-P] \end{pmatrix} \\ &= \begin{pmatrix} y[t-1] \\ \vdots \\ y[t-(P-1)] \end{pmatrix}. \end{aligned}$$

Finally, it follows by combining (101), (99), and (102) that the weights we chose yield the desired output signal. □

We have hence established that RNNs with real-valued weights can realize LTI systems with rational transfer functions exactly. In fact, as inspection of the weight matrices A_1, A_2 and the bias vectors b_1, b_2 in the proof of Theorem 5.1 reveals, the size of the RNN is $\mathcal{O}(P+Q)$ and hence proportional to the number of parameters in the system transfer function.

We now proceed to argue that the results established in Section 4 provide a fundamental limit on how well difference equations of the form (93) can be learned in principle and that RNNs can achieve this fundamental limit. But first, we state an important restriction, namely to LTI systems (of rational transfer function) that have corresponding impulse responses in ℓ_1. In system theory parlance such systems are often referred to as stable [38, Section 2.6]. If the coefficients of $K(z)$ in (93) are such that the system is, indeed, stable, the impulse response is necessarily a linear combination of terms of the form $p(t)\cos(\theta t + \omega)\beta^t$, where $p(t)$ is a polynomial in t, $\beta \in (0,1)$, and $\theta, \omega \in \mathbb{R}$

[39]. Denoting the largest β occurring in this linear combination by $\widetilde{\beta}$, this class of impulse responses is contained in the set $\mathcal{C}(a', \log(1/\beta'))$ with $\beta' > \widetilde{\beta}$ and a' chosen suitably, where such a $\beta' \in (0,1)$ always exists thanks to the set $(0,1)$ being open and a' exists as $a > 0$ in (54). Application of [33, Theorem 13.6] or inspection of the embedding argument used in [4] to establish the lower bound (65) shows that the covering number of the set of stable rational transfer functions equals that of $\mathcal{C}(a', \log(1/\beta'))$. Hence, Theorem 4.1 allows us to conclude that RNNs can, in principle, learn difference equations of the form (93) with coefficients a_i, b_j such that the corresponding LTI system is stable in a metric-entropy-optimal manner.

6. Conclusion

The setting in this paper was deliberately chosen so as to allow the minimum level of mathematical sophistication needed to bring out the main conceptual findings. Numerous extensions abound, such as the continuous-time case and the approximation of nonlinear systems. It would furthermore be interesting to understand how metric entropy results can be obtained for linear time-varying systems. This would possibly allow to establish RNN covering optimality for general linear dynamical systems. From a control theory perspective our findings state that RNNs can be trained to optimally—in the sense of metric entropy—identify LTI systems. Here, it would be interesting to understand whether the presence of feedback, which is known to reduce identification complexity, could be incorporated into our theory and whether the corresponding fundamental limits can again be shown to be achievable through identification by RNNs. Furthermore, we consider it worthwhile to investigate how concepts such as controllability, reachability, and observability for linear dynamical systems transfer to the state-space representation of RNNs realizing these systems. An issue we have not touched upon at all is that of algorithms for learning the weights of approximating RNNs and whether such algorithms are likely to find the RNN constructions we exhibit. Another important aspect we did not discuss is that of minimality of linear dynamical system realizations [38] and how it relates to corresponding RNN realizations [40]. A question cast in the same mould is that of uniqueness of neural network realizations, a large field of research, both in feedforward as well as recurrent neural network theory [41, 42, 43, 44, 45, 46]. Finally, we find that extensions of the ideas in Section 5 to linear, nonlinear, partial, and stochastic differential equations constitute a worthwhile endeavor. In this regard, we mention that the universal realization result Lemma 2.3, in its continuous-time incarnation, suggests that pseudo-differential operators [9, Chapter 14] can be represented exactly by (continuous-time) RNNs.

A. Alternative Definitions of RNNs

Definition A.1 (Elman RNN)**.** *[30], [31, p. 274] For $\mathring{m} \in \mathbb{N}$, weights $\mathring{U} \in \mathbb{R}^{\mathring{m}\times 1}, \mathring{W}_1 \in \mathbb{R}^{\mathring{m}\times\mathring{m}}$, $\mathring{W}_2 \in \mathbb{R}^{1\times\mathring{m}}$, and biases $\mathring{b}_1 \in \mathbb{R}^{\mathring{m}}$, $\mathring{b}_2 \in \mathbb{R}$, the Elman RNN*

with hidden state sequence $\mathring{h}[t] \in \mathbb{R}^{\mathring{m}}$ *of initial state* $\mathring{h}[-1] = 0_{\mathring{m}}$ *and output* $y[t] \in \mathbb{R}$*, for all* $t \geq 0$*, is defined by*

$$\mathring{h}[t] = \rho(\mathring{U}x[t] + \mathring{W}_1\mathring{h}[t-1] + \mathring{b}_1) \tag{A.1}$$

$$y[t] = \mathring{W}_2\mathring{h}[t] + \mathring{b}_2. \tag{A.2}$$

Lemma A.1. *The input-output relation of every RNN according to Definition 1.1 can equivalently be realized by an Elman RNN.*

Proof. Given an RNN according to Definition 1.1 with weight matrices A_1, A_2 and bias vectors b_1, b_2, we construct an Elman RNN that realizes the same input-output map. First, set

$$A_1 = \begin{pmatrix} A_x & A_g \end{pmatrix}, \qquad A_2 = \begin{pmatrix} A_y \\ A_h \end{pmatrix}, \qquad b_1 = b_g, \qquad b_2 = \begin{pmatrix} b_y \\ b_h \end{pmatrix},$$

with $A_x \in \mathbb{R}^{n\times 1}$, $A_g \in \mathbb{R}^{n\times m}$, $A_y \in \mathbb{R}^{1\times n}$, $A_h \in \mathbb{R}^{m\times n}$, $b_g \in \mathbb{R}^n$, $b_y \in \mathbb{R}$, and $b_h \in \mathbb{R}^m$. With these definitions, (1) and (2) can be written as

$$\begin{pmatrix} y[t] \\ h[t] \end{pmatrix} = \begin{pmatrix} A_y \\ A_h \end{pmatrix} g[t] + \begin{pmatrix} b_y \\ b_h \end{pmatrix}, \tag{A.3}$$

where

$$g[t] = \rho\left(\begin{pmatrix} A_x & A_g \end{pmatrix} \begin{pmatrix} x[t] \\ h[t-1] \end{pmatrix} + b_g\right).$$

The equivalent—in the sense of input-output relation—Elman RNN is now obtained by setting $\mathring{m} = n$ and

$$\begin{gathered} \mathring{W}_1 = A_g A_h, \qquad \mathring{U} = A_x, \qquad \mathring{b}_1 = A_g b_h + b_g, \\ \mathring{W}_2 = A_y, \qquad \mathring{b}_2 = b_y. \end{gathered} \tag{A.4}$$

We first establish, by induction, that these choices lead to the hidden state sequences of the original RNN and the equivalent Elman RNN to be related according to $h[t] = A_h\mathring{h}[t] + b_h$, for all $t \geq 0$. The base case follows by choosing the initial hidden state $\mathring{h}[-1]$ of the Elman RNN such that $h[-1] = 0_m = A_h\mathring{h}[-1] + b_h$. If $b_h = 0$, which is the case for all RNN constructions in this paper, one can, indeed, simply set $\mathring{h}[-1] = 0$. Next, we assume that $h[t-1] = A_h\mathring{h}[t-1] + b_h$, for some $t \geq 0$, and insert (A.4) into (A.1) to obtain

$$\begin{aligned} \mathring{h}[t] &= \rho(A_x x[t] + A_g A_h \mathring{h}[t-1] + A_g b_h + b_g) \\ &= \rho(A_x x[t] + A_g(A_h\mathring{h}[t-1] + b_h) + b_g) \\ &= \rho(A_x x[t] + A_g h[t-1] + b_g) \\ &= g[t]. \end{aligned}$$

Using $\mathring{h}[t] = g[t]$ in (A.3) then yields $h[t] = A_h\mathring{h}[t] + b_h$ as desired. This completes the proof of the induction step. The input-output relation of the Elman RNN is seen to equal that of the original RNN—given by (A.3) as $y[t] = A_y g[t] + b_y$—upon inserting $\mathring{h}[t] = g[t]$, $\mathring{W}_2 = A_y$, and $\mathring{b}_2 = b_y$ in (A.2). □

B. Properties of the $\mathcal{Z}$-transform and of Hardy Norms

Lemma B.1. *Let $x[t]$ be a one-sided sequence, i.e., $x[t] = 0$, for $t < 0$. Then, for $k \in \mathbb{N}$, it holds that*

$$(\mathcal{Z}\{x[\cdot - k]\})(z) = z^k(\mathcal{Z}\{x[\cdot]\})(z).$$

Proof. We have

$$\begin{aligned}(\mathcal{Z}\{x[\cdot - k]\})(z) &= \sum_{t=0}^{\infty} x[t-k]z^t = \sum_{t=-k}^{\infty} x[t]z^{t+k} \\ &= z^k \sum_{t=0}^{\infty} x[t]z^t = z^k(\mathcal{Z}\{x[\cdot]\})(z),\end{aligned}$$

where we used that $x[t]$ is one-sided. □

Theorem B.1. *Let $x \in \ell^2$ be a one-sided sequence, i.e., $x[t] = 0$, for $t < 0$. Then, we have*

$$\|X\|_{\mathcal{H}^2} = \|x\|_{\ell^2}.$$

Proof.

$$\begin{aligned}\|X\|_{\mathcal{H}^2}^2 &= \sup_{r<1} \frac{1}{2\pi} \int_0^{2\pi} \left|X(re^{i\theta})\right|^2 d\theta \\ &= \sup_{r<1} \frac{1}{2\pi} \int_0^{2\pi} \left|\sum_{t=0}^{\infty} x[t](re^{i\theta})^t\right|^2 d\theta \\ &= \sup_{r<1} \sum_{t=0}^{\infty}\sum_{t'=0}^{\infty} x[t]\overline{x[t']}\, r^{t+t'} \frac{1}{2\pi} \int_0^{2\pi} e^{i\theta(t-t')} d\theta \\ &= \sup_{r<1} \sum_{t=0}^{\infty}\sum_{t'=0}^{\infty} x[t]\overline{x[t']}\, r^{t+t'} \mathbb{1}_{\{t=t'\}} \\ &= \sup_{r<1} \sum_{t=0}^{\infty} |x[t]|^2\, r^{2t} \\ &= \sum_{t=0}^{\infty} |x[t]|^2 = \|x\|_{\ell^2}^2.\end{aligned}$$

□

Theorem B.2. *For $K(\cdot)$ such that $\|K\|_{\mathcal{H}^\infty} < \infty$, it holds that*

$$\|K\|_{\mathcal{H}^\infty} = \sup_{X \in \mathcal{H}^2} \frac{\|KX\|_{\mathcal{H}^2}}{\|X\|_{\mathcal{H}^2}}. \tag{B.1}$$

Proof. The proof essentially follows [47] with minor refinements and details filled in. We start by noting that the RHS of (B.1) is the operator norm $\|\!|K|\!\|_2 :=$

$\sup_{X\in\mathcal{H}^2} \frac{\|KX\|_{\mathcal{H}^2}}{\|X\|_{\mathcal{H}^2}}$ of the multiplication operator $X(z) \to K(z)X(z)$ and first establish that $|||K|||_2 \leq \|K\|_{\mathcal{H}^\infty}$. For every $X \in \mathcal{H}^2$, we have

$$\begin{aligned}
\|K\,X\|_{\mathcal{H}^2} &= \sqrt{\sup_{r<1} \frac{1}{2\pi} \int_0^{2\pi} |K(re^{i\theta})X(re^{i\theta})|^2 d\theta} \\
&\leq \sqrt{\sup_{r<1} \frac{1}{2\pi} \int_0^{2\pi} |X(re^{i\theta})|^2 \left(\sup_{|z|<1} |K(z)|\right)^2 d\theta} \\
&= \|K\|_{\mathcal{H}^\infty} \sqrt{\sup_{r<1} \frac{1}{2\pi} \int_0^{2\pi} |X(re^{i\theta})|^2 d\theta} \\
&= \|K\|_{\mathcal{H}^\infty} \|X\|_{\mathcal{H}^2},
\end{aligned}$$

which, upon division by $\|X\|_{\mathcal{H}^2}$ establishes the desired upper bound.

To complete the proof, we show that $|||K|||_2 \geq \|K\|_{\mathcal{H}^\infty}$. Applying

$$\|KX\|_{\mathcal{H}^2} \leq |||K|||_2 \|X\|_{\mathcal{H}^2}$$

repeatedly, we get, for every $n \in \mathbb{N}$,

$$\|K^n X\|_{\mathcal{H}^2} \leq |||K|||_2^n \|X\|_{\mathcal{H}^2}. \tag{B.2}$$

Without loss of generality, we can restrict ourselves to $|||K|||_2 = 1$ as otherwise we can simply consider $K' := K/|||K|||_2$. Next, towards a contradiction, assume that $|||K|||_2 < \|K\|_{\mathcal{H}^\infty}$, which, thanks to $|||K|||_2 = 1$, results in $1 < \|K\|_{\mathcal{H}^\infty} = \sup_{r<1,\, 0\leq\theta<2\pi} |K(re^{i\theta})|$. As $\|K\|_{\mathcal{H}^\infty} < \infty$ by assumption, it follows that $K(z)$ is analytic and thus continuous inside the unit disk. Hence, there exist $0 < r' < 1, \epsilon > 0$ and an interval $[\underline{\theta}, \overline{\theta}) \in [0, 2\pi)$ with $\overline{\theta} - \underline{\theta} = \delta > 0$ such that

$$|K(r'e^{i\theta'})| > 1 + \epsilon, \quad \forall \theta' \in [\underline{\theta}, \overline{\theta}). \tag{B.3}$$

Now we take $X(z) = 1$ which clearly satisfies $\|X\|_{\mathcal{H}^2} = 1$. Inserting this into (B.2), we obtain

$$\|K^n X\|_{\mathcal{H}^2}^2 \leq |||K|||_2^{2n} \|X\|_{\mathcal{H}^2}^2 = 1.$$

This, however, finalizes the proof by leading to the following contradiction

$$\begin{aligned}
1 &\geq \|K^n X\|_{\mathcal{H}^2}^2 \\
&= \sup_{0<r<1} \frac{1}{2\pi} \int_0^{2\pi} |K(re^{i\theta})|^{2n} d\theta \\
&\geq \frac{1}{2\pi} \int_0^{2\pi} |K(r'e^{i\theta})|^{2n} d\theta \\
&\geq \frac{1}{2\pi} \int_0^{2\pi} ((1+\epsilon)\mathbb{1}_{\{\theta\in[\underline{\theta},\overline{\theta})\}})^{2n} d\theta \qquad \text{(B.4)} \\
&= \frac{\delta}{2\pi}(1+\epsilon)^{2n} \xrightarrow[n\to\infty]{} \infty,
\end{aligned}$$

where in (B.4) we used (B.3) and the fact that $|K(r'e^{i\theta})| \geq 0$, for $\theta \notin [\underline{\theta}, \overline{\theta})$. □

References

[1] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (1989) 359–366. doi:10.1016/0893-6080(89)90020-8.

[2] K.-I. Funahashi, On the approximate realization of continuous mappings by neural networks, Neural Networks 2 (1989) 183–192. doi:10.1016/0893-6080(89)90003-8.

[3] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control, Signals and Systems 2 (1989) 303–314. doi:10.1007/BF02551274.

[4] G. Zames, J. G. Owen, A note on metric dimension and feedback in discrete time, IEEE Transactions on Automatic Control 38 (1993) 664–667. doi:10.1109/9.250545.

[5] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2009) 855–868. doi:10.1109/TPAMI.2008.137.

[6] A. Graves, S. Fernández, F. Gomez, J. Schmidhuber, Connectionist temporal classification: Labeling unsegmented sequence data with recurrent neural networks, in: ICML '06: Proceedings of the 23rd International Conference on Machine Learning, 2006, pp. 369–376. doi:10.1145/1143844.1143891.

[7] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Advances in Neural Information Processing Systems, volume 27, 2014. URL: https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf.

[8] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, D. Silver, Mastering Atari, Go, chess and shogi by planning with a learned model, Nature 588 (2020) 604–609. doi:10.1038/s41586-020-03051-4.

[9] K. Gröchenig, Foundations of Time-Frequency Analysis, Birkhäuser Boston, 2001. doi:10.1007/978-1-4612-0003-1.

[10] C. L. Fefferman, The uncertainty principle, Bulletin (New Series) of the American Mathematical Society 9 (1983) 129–206. doi:10.1090/S0273-0979-1983-15154-6.

[11] G. Matz, H. Bölcskei, F. Hlawatsch, Time-frequency foundations of communications: Concepts and tools, IEEE Signal Processing Magazine 30 (2013) 87–96. doi:10.1109/MSP.2013.2269702.

[12] G. Zames, On the metric complexity of causal linear systems: ϵ-entropy and ϵ-dimension for continuous time, IEEE Transactions on Automatic Control 24 (1979) 222–230. doi:10.1109/TAC.1979.1101976.

[13] A. Kolmogorov, V. Tikhomirov, ϵ-entropy and ϵ-capacity of sets in functional spaces, in: A. N. Shiryayev (Ed.), Selected Works of A. N. Kolmogorov — Volume III: Information Theory and the Theory of Algorithms, Springer Netherlands, Dordrecht, 1993, pp. 86–170. doi:10.1007/978-94-017-2973-4_7.

[14] D. Donoho, Sparse components of images and optimal atomic decompositions, Constructive Approximation 17 (2001) 353–382. doi:10.1007/s003650010032.

[15] D. Donoho, Unconditional bases and bit-level compression, Applied and Computational Harmonic Analysis 3 (1996) 388–392. doi:10.1006/acha.1996.0032.

[16] D. Donoho, M. Vetterli, R. DeVore, I. Daubechies, Data compression and harmonic analysis, IEEE Transactions on Information Theory 44 (1998) 2435–2476. doi:10.1109/18.720544.

[17] H. Bölcskei, P. Grohs, G. Kutyniok, P. Petersen, Optimal approximation with sparsely connected deep neural networks, SIAM Journal on Mathematics of Data Science 1 (2019) 8–45. doi:10.1137/18m118709x.

[18] D. Elbrächter, D. Perekrestenko, P. Grohs, H. Bölcskei, Deep neural network approximation theory, IEEE Transactions on Information Theory 67 (2021) 2581–2623. doi:10.1109/TIT.2021.3062161.

[19] M. Hardt, T. Ma, B. Recht, Gradient descent learns linear dynamical systems, Journal of Machine Learning Research 19 (2018) 1–44. URL: http://jmlr.org/papers/v19/16-465.html.

[20] Z. Li, J. Han, W. E, Q. Li, On the curse of memory in recurrent neural networks: Approximation and optimization analysis, in: International Conference on Learning Representations, 2021. URL: https://openreview.net/forum?id=8Sqhl-nF50.

[21] A. M. Schäfer, H. G. Zimmermann, Recurrent neural networks are universal approximators, International Journal of Neural Systems 17 (2007) 253–263. doi:10.1142/S0129065707001111.

[22] E. D. Sontag, Neural nets as systems models and controllers (1992). URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.55.8164.

[23] K.-I. Funahashi, Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks, Neural Networks 6 (1993) 801–806. doi:10.1016/S0893-6080(05)80125-X.

[24] M. B. Matthews, Approximating nonlinear fading-memory operators using neural network models, Circuits, Systems and Signal Processing 12 (1993) 279–307. doi:10.1007/BF01189878.

[25] C. Heij, A. C. M. Ran, F. van Schagen, Introduction to Mathematical Systems Theory, Springer International Publishing, 2021. doi:10.1007/978-3-030-59654-5.

[26] T. Chen, H. Chen, Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems, IEEE Transactions on Neural Networks 6 (1995) 911–917. doi:10.1109/72.392253.

[27] L. Lu, P. Jin, G. Pang, Z. Zhang, G. E. Karniadakis, Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, Nature Machine Intelligence 3 (2021) 218–229. doi:10.1038/s42256-021-00302-5.

[28] S. Lanthaler, S. Mishra, G. E. Karniadakis, Error estimates for DeepOnets: A deep learning framework in infinite dimensions, 2021. arXiv:2102.09618.

[29] H. T. Siegelmann, E. D. Sontag, On the computational power of neural nets, Journal of Computer and System Sciences 50 (1995) 132–150. doi:10.1006/jcss.1995.1013.

[30] J. L. Elman, Finding structure in time, Cognitive Science 14 (1990) 179–211. doi:10.1207/s15516709cog1402_1.

[31] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016. URL: www.deeplearningbook.org.

[32] M. J. Wainwright, High-Dimensional Statistics, Cambridge University Press, 2019. doi:10.1017/9781108627771.

[33] W. Rudin, Real and Complex Analysis, 3rd ed., McGraw-Hill, 1987.

[34] P. Grohs, F. Hornung, A. Jentzen, P. von Wurstemberger, A proof that artificial neural networks overcome the curse of dimensionality in the numerical approximation of Black-Scholes partial differential equations, Mem. Amer. Math. Soc. (2018) to appear. arXiv:1809.02362.

[35] J. Berner, P. Grohs, A. Jentzen, Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of Black–Scholes partial differential equations, SIAM Journal on Mathematics of Data Science 2 (2020) 631–657. doi:10.1137/19M125649X.

[36] M. Raslan, Solving parametric PDEs with neural networks: Unfavorable structure vs. expressive power, Ph.D. thesis, TU Berlin, 2021.

[37] Z. Y. Wan, P. Vlachas, P. Koumoutsakos, T. Sapsis, Data-assisted reduced-order modeling of extreme events in complex dynamical systems, PLOS ONE 13 (2018) e0197704. doi:10.1371/journal.pone.0197704.

[38] T. Kailath, Linear Systems, Prentice-Hall, Englewood Cliffs, N.J, 1980.

[39] A. V. Oppenheim, R. W. Schafer, J. R. Buck, Discrete-Time Signal Processing, 2nd ed., Prentice-Hall, Inc., USA, 1999.

[40] E. Sontag, Recurrent neural networks: Some systems-theoretic aspects, in: M. Kárný, K. Warwick, V. Kůrková (Eds.), Dealing with Complexity, Springer London, 1998. doi:10.1007/978-1-4471-1523-6.

[41] C. Fefferman, Reconstructing a neural net from its output, Revista Matemática Iberoamericana 10 (1994) 507–555. doi:10.4171/RMI/160.

[42] V. Vlačić, H. Bölcskei, Neural network identifiability for a family of sigmoidal nonlinearities, Constructive Approximation (2021). doi:10.1007/s00365-021-09544-3.

[43] V. Vlačić, H. Bölcskei, Affine symmetries and neural network identifiability, Advances in Mathematics 376 (2021). doi:10.1016/j.aim.2020.107485.

[44] F. Albertini, E. D. Sontag, For neural networks, function determines form, Neural Networks 6 (1993) 975–990. doi:10.1016/S0893-6080(09)80007-5.

[45] F. Albertini, E. D. Sontag, V. Maillot, Uniqueness of weights for neural networks, in: Artificial Neural Networks with Applications in Speech and Vision, Chapman and Hall, 1993, pp. 115–125.

[46] F. Albertini, E. Sontag, State observability in recurrent neural networks, in: Proceedings of 32nd IEEE Conference on Decision and Control, volume 4, 1993, pp. 3706–3707. doi:10.1109/CDC.1993.325908.

[47] J. E. McCarthy, Pick's Theorem-What's the big deal?, The American Mathematical Monthly 110 (2003) 36–45. doi:10.2307/3072342.