ORIGINAL RESEARCH
published: 02 March 2022
doi: 10.3389/fams.2022.818799

On Lyapunov Exponents for RNNs: Understanding Information Propagation Using Dynamical Systems Tools

Ryan Vogt[1†], *Maximilian Puelma Touzel*[2,3†], *Eli Shlizerman*[1,4*‡] *and Guillaume Lajoie*[2,3*‡]

[1] *Department of Applied Mathematics, University of Washington, Seattle, WA, United States,* [2] *Mathematics and Statistics Department, Université de Montréal, Montréal, QC, Canada,* [3] *Mila - Québec AI Institute, Montréal, QC, Canada,* [4] *Department of Electrical and Computer Engineering, University of Washington, Seattle, WA, United States*

OPEN ACCESS

Edited by:
Qingtang Jiang,
University of Missouri–St. Louis, United States

Reviewed by:
Qi Ye,
South China Normal University, China
Vishwa Goudar,
New York University, United States

***Correspondence:**
Eli Shlizerman
shlizee@uw.edu
Guillaume Lajoie
g.lajoie@umontreal.ca

†*These authors share first authorship*
‡*These authors share senior authorship*

Specialty section:
This article was submitted to Mathematics of Computation and Data Science, a section of the journal Frontiers in Applied Mathematics and Statistics

Received: *20 November 2021*
Accepted: *31 January 2022*
Published: *02 March 2022*

Citation:
Vogt R, Puelma Touzel M, Shlizerman E and Lajoie G (2022) On Lyapunov Exponents for RNNs: Understanding Information Propagation Using Dynamical Systems Tools. Front. Appl. Math. Stat. 8:818799. doi: 10.3389/fams.2022.818799

Recurrent neural networks (RNNs) have been successfully applied to a variety of problems involving sequential data, but their optimization is sensitive to parameter initialization, architecture, and optimizer hyperparameters. Considering RNNs as dynamical systems, a natural way to capture stability, i.e., the growth and decay over long iterates, are the Lyapunov Exponents (LEs), which form the Lyapunov spectrum. The LEs have a bearing on stability of RNN training dynamics since forward propagation of information is related to the backward propagation of error gradients. LEs measure the asymptotic rates of expansion and contraction of non-linear system trajectories, and generalize stability analysis to the time-varying attractors structuring the non-autonomous dynamics of data-driven RNNs. As a tool to understand and exploit stability of training dynamics, the Lyapunov spectrum fills an existing gap between prescriptive mathematical approaches of limited scope and computationally-expensive empirical approaches. To leverage this tool, we implement an efficient way to compute LEs for RNNs during training, discuss the aspects specific to standard RNN architectures driven by typical sequential datasets, and show that the Lyapunov spectrum can serve as a robust readout of training stability across hyperparameters. With this exposition-oriented contribution, we hope to draw attention to this under-studied, but theoretically grounded tool for understanding training stability in RNNs.

Keywords: hyperparameters and chaos, LSTM (long short term memory networks), Lyapunov Exponents (LEs), dynamical systems (DS), recurrent neural networks (RNN)

1. INTRODUCTION

The propagation of error gradients in deep learning leads to the study of recursive compositions and their stability [1]. Vanishing and exploding gradients arise from long products of Jacobians of the hidden state dynamics whose norm exponentially grows or decays, which potential hinders training [2]. To mitigate this sensitivity, much effort has been made to mathematically understand the link between model parameters and the eigen- and singular-value spectra of these long products [3–5]. For architectures used in practice, this approach appears to have limited a scope [6, 7] likely due to spectra having non-trivial properties reflecting complicated long-time dependencies within the trajectory. Fortunately, the theory of dynamical systems has formulated this problem for general dynamical systems, i.e., for arbitrary architectures. The Lyapunov spectrum (LS) formed by the Lyapunov exponents (LEs) is precisely the measurement that characterizes rates of expansion and contraction of non-linear system dynamics.

While there are results using LEs in machine learning [8–10], most only look at the largest exponent, the sign of which indicates the presence or absence of chaos. Additionally, to our knowledge, current machine learning uses of LEs are for the autonomous case, i.e., analyzing RNN dynamics in the absence of inputs, a regime at odds with the most common use of RNNs as input-processing systems. Thus, there is likely much to be gained from both the study of other features of the LS and from including input-driven regimes into LEs analysis. Indeed, a concurrent work coming from theoretical neuroscience takes up this approach and outlines the ongoing work using LEs to study models of neural networks in the brain [11]. However, the machinery to compute the LS and the theory surrounding its interpretation is still relatively unknown in the machine learning community.

In this exposition-style paper, we aim to fill this gap by presenting an overview of LEs in the context of RNNs, and discuss their usefulness for studying training dynamics. To compute LEs, we present a novel algorithm for various RNN architectures, taking advantage of modern deep learning environments. In addition, we present encouraging results which show a correlation between LEs and performance. Finally, we highlight key directions of research that have the potential to leverage LEs for improvement of RNNs robustness and performance, such as efficient hyperparameter tuning by allowing early detection of performance.

2. MOTIVATION AND DEFINITIONS

Here, we develop the problem of spectral constraints for robust gradient propagation. For transparent exposition, in this section we will consider the "vanilla" RNN defined as

$$\mathbf{o}_t = \mathbf{W}\mathbf{h}_t,\ \mathbf{h}_t = \phi(\mathbf{a}_t),\ \mathbf{a}_t = \mathbf{V}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{x}_t + \mathbf{b}\ , \tag{1}$$

where $\mathbf{V}$ is the recurrent weight matrix that couples elements of the hidden state vector $\mathbf{h}_t \in \mathbb{R}^N$, $\mathbf{U}$ projects the input, $\mathbf{x}_t$, into the network, $\mathbf{b}$ is a constant bias vector, and ϕ applies a non-linear scalar transformation element-wise. The readout weights, $\mathbf{W}$, output the activity into the output variable, $\mathbf{o}_t$. The loss over T iterates is $L = \sum_{t=1}^{T} L_t$, with $L_t = f(\mathbf{y}_t, \hat{\mathbf{y}}_t)$, with f some scalar loss function (e.g., cross-entropy loss, $f(\mathbf{y}, \hat{\mathbf{y}}) = \mathbf{y} \cdot \log \hat{\mathbf{y}}$), $\hat{\mathbf{y}}_t$ the prediction [e.g., $\hat{\mathbf{y}}_t = \mathrm{softmax}(\mathbf{o}_t)$], and $\mathbf{y}_t$ is a one-hot binary target vector. The parameters, $\Theta = (\mathbf{V}, \mathbf{U}, \mathbf{b}, \mathbf{W})$, are learned by following the gradient of the loss, *e.g.,* in the space of recurrent weights $\mathbf{V}$,

$$\nabla_{\mathbf{V}} L = \sum_{t=1}^{T}\sum_{i=1}^{N} \frac{\partial L}{\partial h_{t,i}} \nabla_{\mathbf{V}} h_{t,i} = \sum_{t=1}^{T} \mathrm{diag}(\phi'(\mathbf{a}_t)) \nabla_{\mathbf{h}_t} L\, \mathbf{h}_{t-1}^{\top}\ , \tag{2}$$

where diag($\mathbf{x}$) is the diagonal matrix formed by the vector $\mathbf{x}$ and ϕ' is the derivative of ϕ and $^\top$ denotes transpose. Here,

$$\nabla_{\mathbf{h}_t} L = \sum_{s=t}^{T} \left(\prod_{r=t+1}^{s} \mathbf{J}_r^{\top} \right) \mathbf{W}^{\top} \nabla_{\mathbf{o}_s} L\ , \tag{3}$$

where $\nabla_{\mathbf{o}_s} L$ is some simple expression (*e.g.,* $\hat{\mathbf{y}} - \mathbf{y}_t$ for cross-entropy loss) and $\mathbf{J}_t = \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}}$ is the Jacobian of the hidden state dynamics,

$$\mathbf{J}_t = \mathrm{diag}\left(\phi'(\mathbf{a}_t)\right) \mathbf{V}\ . \tag{4}$$

$\mathbf{J}_t$ varies in time with $\mathbf{x}_t$ and $\mathbf{h}_{t-1}$ *via* $\mathbf{a}_t$ and so is treated as a random matrix with ensemble properties arising from the specified input statistics and the emergent hidden state statistics.

Gradients can vanish or explode with T according to the singular value spectra of the products of Jacobians appearing in Equation (3) [1]. This fact motivated the study of hyperparameter constraints that control the average of the latter over the input distribution. Specifically, the scale parameter of Gaussian or orthogonal initializations of $\mathbf{V}$ is chosen such that the condition number (ratio of largest to smallest singular value) of $\mathbf{J}_t$ is of value around 1 for all t [2], or such that the first and second moments of the distribution of squared singular values (averaged over inputs) of the Jacobian products are chosen near 1 and 0, respectively according to dynamical isometry [4]. Parameter conditions for the latter have been derived for i.d.d. Gaussian input. These approach has been extended to RNNs under the assumption of untied weights with good empirical correspondence in vanilla and minimalRNN [3], but larger discrepancy in LSTMs [12], suggesting there are other contributions to stability in more complex, state-of-the-art architectures [6, 7].

Better understanding how robust gradient propagation emerges in these less idealized settings that include high-dimensional and temporally correlated input sequences demands a more general theory. These Jacobian products appearing in Equation (3) can each be expressed as the transpose of a forward sequence, $\prod_{r=t+1}^{s} \mathbf{J}_r^{\top} = (\mathbf{J}_s \cdots \mathbf{J}_{t+1})^{\top}$. Note that the backward time gradient dynamics shares properties with the forward time hidden state dynamics. This is an important insight that suggests that use of advanced tools from dynamical systems theory may help better understand gradient propagation.

3. BACKGROUND

Here, we describe the stochastic Lyapunov exponents from the ergodic theory of non-autonomous dynamical systems and outline their connection to the conditions that support gradient-based learning in RNNs. Jacobians are linear maps that evolve state perturbations forward along a trajectory, $\mathbf{u}_t = \mathbf{J}_t\mathbf{u}_{t-1}$, *e.g.,* for some initial perturbation $\mathbf{h}_0(\epsilon) = \mathbf{h}_0 + \epsilon\mathbf{u}_0$ with $\epsilon \ll 1$ and a given vector $\mathbf{u}_0$. Thus, perturbations can vanish or explode under the hidden state dynamics according to the singular value spectrum of $\mathbf{T}_t = \mathbf{J}_t \cdots \mathbf{J}_1$. The linear stability of the dynamics is obtained taking $\epsilon \to 0$ and then $t \to \infty$. In particular, the Lyapunov exponents, $\{\lambda_i\}_{i=1}^{N}$, are the exponential growth

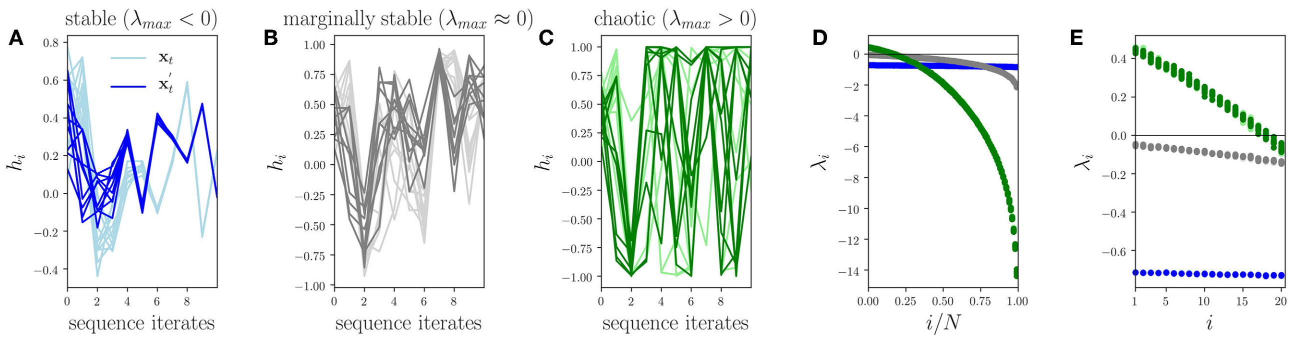

FIGURE 1 | *Stability regimes of forward dynamics using Lyapunov spectra.* **(A–C)** Example unit hidden states from simulations computed from 10 initial conditions and 2 input sequences [$\mathbf{x}_t$ (light) and $\mathbf{x}_t'$ (dark)] for weight matrix **V** with Gaussian distributed entries with mean 0 and variance $\sigma^2_{\mathbf{V}} = g/N$, with g set to 3 examples spanning the 3 qualitatively distinct stability regimes based on the sign of λ_{max}: stability, $\lambda_{max} < 0$ (a; blue, $g = 1/2$); marginal stability, $\lambda_{max} \approx 0$ (b; gray, $g = 2$); and chaos $\lambda_{max} > 0$ (c; green, $g = 10$). **(D)** Lyapunov spectra computed from **(A–C)** (same colors). **(E)** First 20 exponents from **(D)**. (vanilla RNN with $\phi = \tanh$; **V** orthogonal; $\mathbf{U} = \mathbb{I}$; 2 realizations (light/dark) of **x** Gaussian, $\Lambda_{\mathbf{x}} = \sigma^2_{\mathbf{x}}\mathbb{I}$ with $\sigma^2_{\mathbf{x}} = 0.6$; 10 Gaussian initial conditions, $\Lambda_{\mathbf{h}_0} = \sigma^2_{\mathbf{h}_0}\mathbb{I}$ with $\sigma^2_{\mathbf{h}_0} = 1$).

rates associated with the singular values of $\mathbf{T}_t^{1/t}$ for $t \to \infty$, *i.e.,* the logarithms of the eigenvalues of $\lim_{t\to\infty}\left(\mathbf{T}_t^{\top}\mathbf{T}_t\right)^{\frac{1}{2t}}$. As a property of the stationary dynamics, the Lyapunov exponents are independent of initial condition for ergodic systems, i.e., those with only one or the same type of attractor. If the maximum Lyapunov exponent $\lambda_{\max}$ is positive, the stationary dynamics is chaotic and small perturbations explode, otherwise it is stable and small perturbations vanish. The theory was initially developed for autonomous dynamical systems, where stable dynamics implies limit-cycle or fixed-point attractors. How the shape of the Lyapunov spectrum varies with network model parameters has provided unprecedented insight into autonomous (evolving in the absence of inputs) neural network models in theoretical neuroscience [11, 13, 14].

RNN dynamics are non-autonomous because the hidden state dynamics $\mathbf{h}_t$ are driven by inputs $\mathbf{x}_t$. The theory of *random dynamical systems* generalizes stability analyses to non-autonomous dynamics driven by an input sequence sampled from a stationary distribution [15]. Analytical results typically employ uncorrelated Gaussian inputs, but the framework is expected to apply to a wider range of well-behaved input statistics. This includes those with finite, low-order moments and finite correlation times like character streams from written language and sensor data from motion capture systems. The time course of the driven dynamics depends on the specific input realization, but critically, the theory guarantees that the stationary dynamics for all input realizations share the same stability properties which will, in general, depend on the input distribution (e.g., its variance). Stability here is quantified using the same definition of Lyapunov exponents in the autonomous case (now called *stochastic Lyapunov exponents*; we hereon will omit stochastic for brevity). In **Figure 1**, we show vanilla RNN dynamics in stable, marginally stable, and chaotic regimes, as measured by the sign of the largest LE. For stable driven dynamics, the stationary activity on the time-dependent attractor (called a random sink) is independent of initial condition. This holds true also in the marginally stable case, $\lambda_{max} \approx 0$, where in addition there are directions along which the sizes of perturbations (error gradients) are maintained over many iterates. For chaotic driven dynamics, the activity variance over initial conditions fluctuates in time in a complex way. Understanding stability properties in this non-autonomous setting is at the frontier of analysis of neural network dynamics in both theoretical neuroscience [11, 16, 17] and machine learning [18, 19].

The practical calculation of the Lyapunov spectra (see Section 4) is fast and offers more robust numerical behavior and generality over singular value decomposition. Together, this suggests the Lyapunov spectra as a useful diagnostic for RNN sequence learning.

There are a handful of features of the Lyapunov spectrum that dictate gradient propagation and thus influence training speed and performance. The maximum Lyapunov exponent for non-chaotic systems measures the restoration timescale after a small perturbation. The closer to zero this exponent is, the longer this time is and so it set an upper bound on the duration that information about a given input can reside in the system. The mean exponent determines the rate of contraction of full volume elements. The LE variance measures heterogeneity in stability across different directions and can reflect the conditioning of the product of many Jacobians. We use the first two in the analysis of our experiments later in this article.

We remind readers that much of the theory regarding high-dimensional dynamics is derived in the so-called thermodynamic limit $N \to \infty$. Here, the number of exponents diverges and the spectrum over i/N becomes stationary (so-called extensive dynamics), i.e., it is insensitive to N so long as N is large enough (e.g., [11, 13]). Here, self-averaging effects can take hold and enable accurate analytical results based on Gaussian assumptions justified by central limit theorem arguments. "Large enough" N is typically in the hundreds (though this should be checked for any given application) and so these results can be useful for studying RNNs.

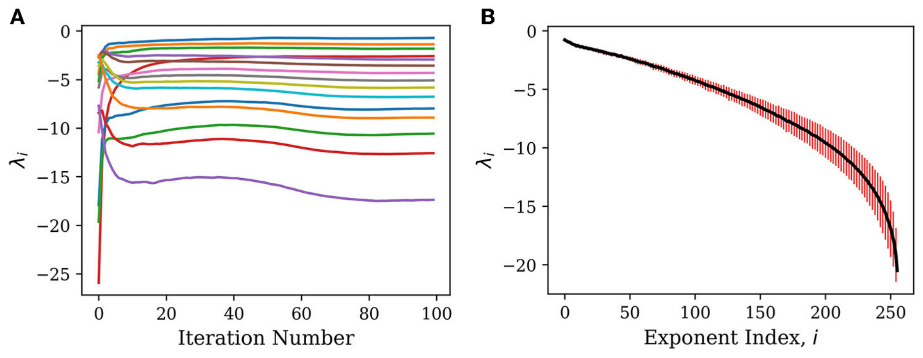

FIGURE 2 | *Convergence of Lyapunov spectrum estimator*. **(A)** Lyapunov exponent estimation as a function of time, as shown in Equation (5), where T is the number of iterations. Each line represents a different exponent in the spectrum. By 100 iterations, it is clear that each exponent is changing very little as a function of iteration number. **(B)** The mean Lyapunov spectrum over the set of input sequences. Standard deviation is by red bars.

Finally, we note that obtaining the Lyapunov exponents from their definition relates to the singular values of $\mathbf{T}_t^{1/t}$ for large t, and thus is numerically impractical. The standard approach [20] is to exploit the fact that m-dimensional volume elements grow at a rate $\lambda^{(m)} = \sum_{i=1}^{m} \lambda_i$ and so the desired rates, $\lambda_1 = \lambda^{(1)}$, $\lambda_2 = \lambda^{(2)} - \lambda_1, \ldots$ arise from volumes obtained by projecting orthogonal to subspaces of increasing dimension. These rates are a direct output of computationally efficient orthonormalization procedures such as QR-decomposition. Next, we discuss the practical aspects of computing the Lyapunov exponents in input-driven RNNs.

4. LYAPUNOV SPECTRUM ESTIMATION FOR NON-AUTONOMOUS RNNS

We extend the framework of Lyapunov exponents to recurrent neural networks by calculating the asymptotic trajectories of the hidden states of the networks when driven by the same signal.

4.1. Algorithm Description

We adopt the well-established standard algorithm for computing the Lyapunov spectrum [20, 21]. We choose the driving signal to be sampled from fixed-length sequences of the test set. For each input sequence in a batch, a matrix **Q** is initialized as the identity, and the hidden states, **h**, are initialized as zero. To track the expansion and contraction of **Q** at each step, the Jacobian **J** of the hidden state dynamics is calculated and then applied to a set of vectors of **Q**. The exact form of the Jacobian will depend on the architecture of the RNN, since additional mechanisms such as gates would effect how the hidden states evolve in a single iteration. For instance, the equations for the Jacobian of Long Short Term Memory (LSTM) RNN are given in Section B.1. The expansion of each vector is calculated and the matrix **Q** is updated using the QR decomposition at each step. If r_i^t is the expansion of the ith vector at time step t—corresponding to the ith diagonal element of **R** in the QR decomposition—then the ith Lyapunov exponent λ_i resulting from an input signal of length T is given by:

$$\lambda_i = \frac{1}{T} \sum_{t=1}^{T} \log(r_i^t) \tag{5}$$

The Lyapunov exponents resulting from each input x^j in the batch of input sequences are calculated in parallel and then averaged. For our experiments, the Lyapunov exponents were calculated over 100 time steps with 10 different input sequences. The mean of the 10 resulting Lyapunov spectra is reported as the spectrum. An example calculation of the Lyapunov spectrum is shown in **Figure 2**.

The algorithm for computing the Lyapunov exponents is described in **Algorithm** 1.

Algorithm 1: Lyapunov Exponents Calculation.

```
for $\mathbf{x}^j$ in Batch do
    initialize $\mathbf{h}$, $\mathbf{Q}$
    for $t = 1 \to T$ do
        $\mathbf{h} \leftarrow f(\mathbf{h}, \mathbf{x}_t^j)$
        $J \leftarrow \frac{d\mathbf{f}}{d\mathbf{h}}$
        $Q \leftarrow J \cdot Q$
        $Q, R \leftarrow qr(Q)$
        $\gamma_i \mathrel{+}= \log(R_{ii})$
    end
    $\lambda_i^j = \gamma_i^j / T$
end
$\lambda_i = \mathrm{mean}_j(\lambda_i^j)$
```

5. EXPERIMENTS

To illustrate the computation and application of Lyapunov exponents, we applied the approach to two tasks with distinct task constraints: character prediction from sentences, where performance depends on capturing low-dimensional and long temporal correlations, and signal prediction from motion capture

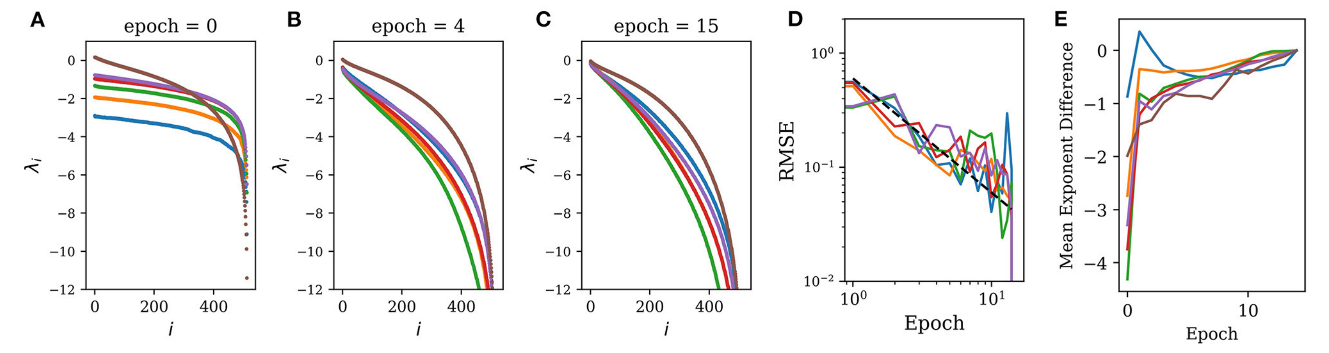

FIGURE 3 | *Evolution of Lyapunov Spectra over training*. For values of initialization parameter given in legend in **(A)** [$p = 0.04$ (Blue), $p = 0.08$ (Orange), $p = 0.12$ (Green), $p = 0.16$ (Red), $p = 0.2$ (Purple), $p = 0.5$ (Brown)]. The spectra before training **(A)** and after four **(B)**, and fifteen **(C)** epochs. The reduced mean-squared error between successive epochs **(D)** shows that the each spectrum converges quickly during training, with t^{-1} plotted as a dashed black line for reference. Similarly, the difference between the mean of the Lyapunov exponents at the fifteenth epoch and the mean at previous epochs **(E)** rapidly approaches zero throughout training.

data, where performance relies also on signal correlations across the many dimensions of the input.

5.1. Task Details

For character prediction, we use Leo Tolstoy's *War and Peace* as the data and follow the character-level language modeling task outlined by Karpathy et al. [22]. For signal prediction, we use the CMU motion-capture dataset and pre-process it using the procedure outlined in Li et al. [23]. Both tasks require predicting the next step in a sequence given the preceding input sequence. For the character prediction task (CharRNN), we use input sequences of 100 characters. For the motion capture task (CMU Mocap), we use input sequences of 25 time steps.

For each task, we use a single-layer LSTM architecture with weight parameters initialized uniformly on $[-p, p]$, where p is referred to as the initialization parameter. For further details of the implementation of both tasks, (see **Appendix**) and the Github repository[1] for available code and ongoing work.

5.2. Algorithm Convergence Properties

We find that the general shape of the spectrum is reached early in training. In **Figure 3**, we use the CharRNN task as an example to show that the spectrum rapidly changes in the first few epochs of training, quickly converging after a small number of training epochs to a spectrum near that of the trained network. The reduced mean-squared difference shows a $\mathcal{O}(t^{-1})$ convergence with learning epoch t, while the mean difference shows this convergence is from below. This was true across the range of initialization parameters tested.

5.3. Performance Efficiency Relative to Training

The calculation of the Lyapunov spectrum is very efficient relative to the computation required for training, as long as the user has

[1]https://github.com/shlizee/lyapunov-hyperopt

TABLE 1 | LE calculation rate relative to training.

Example	Training units needed	LE calculation time (training units)	Fraction of total training time
CharRNN	26 ± 10 epochs	1.04 ± 0.04 epochs	0.04 ± 0.02
CMU Mocap	$2.4 \pm 0.5 \times 10^3$ iters	104 ± 3 iters	0.04 ± 0.01

an analytical expression for the Jacobian of the recurrent layer (as we describe the expressions for GRU and LSTM in B). The average time required for calculating the spectrum for a network by evolving 10 input sequences for 100 time steps relative to the average time required to train those networks on the same device is shown in **Table 1** for the two tasks we consider (c.f. Section 5.1). The mean number of training units (epochs or iterations) needed to reach the minimum validation loss is calculated for each task, and the total training time is compared to the Lyapunov spectra calculation time. For both tasks, the time required to calculate the spectrum was equivalent to about 4% of the total training time.

Further improvements to the computation time of the exponents can be made by making slight modifications to **Algorithm 1**. First, it is possible to "warm-up" the hidden states **h** and **Q** onto the attractor of the dynamical system without the computational cost of the QR decomposition, allowing a faster convergence of the spectrum. A further improvement can be made by not taking the QR decomposition every time step during the calculation of the exponents, but instead every T_{ON} steps. Since the QR step is the most expensive computation in the algorithm, the increase in speed over orthonormalizing every step is approximately T_{ON}. However, since increasing this interval leads to greater expansion and contraction of the vectors of **Q** before orthonormalization steps, this can lead to a spurious plateau at higher indices due to the accumulation of rounding errors. However, if one cares only about the first few exponents, this effect is negligible for reasonable selections of T_{ON} (see C for details about warmup and effect of increasing T_{ON}).

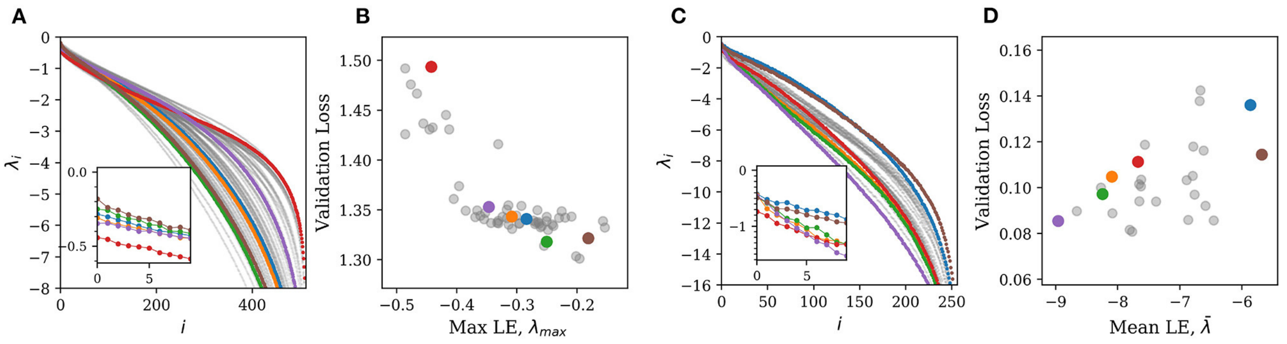

FIGURE 4 | *Correlation between Lyapunov spectra and performance*. Shown are spectra **(A,C)** and loss dependence on spectra **(B,D)** for networks trained on CharRNN **(A,B)**, and CMU MoCap **(C,D)**. Those in **(A,B)** are for 60 random combinations of learning rate and log-dropout rate, while those in **(C,D)** are for 30 values of the initialization parameter, *p*. The colors in **(A,B)** and **(C,D)** are 6 samples selected by eye to roughly span the range of gross spectra shapes seen in **(A)** and **(C)**, respectively. All other spectra are shown in gray.

5.4. Lyapunov Spectrum as a Robust Readout of Training Stability

In general, the dependence of the Lyapunov spectrum on hyperparameters is tangled. To direct our exploration of this dependence and how it relates to performance, we used the task constraints to guide our interpretation of spectra behavior of trained networks from randomly sampled hyperparameters.

For CharRNN, we hypothesized from existing work that to satisfy the constraint of propagating a scalar signal over many iterates, it would be sufficient to have a single exponent approaching 0 with the other exponents more negative [24]. Focusing here on hyperparameters, we uniformly sampled a fixed range of log-dropout and learning rate while keeping the initialization parameter fixed. In **Figures 4A,B**, we observe the spectra and indeed find a correlation between maximum Lyapunov exponent and validation loss. Here, we opt to report performance with loss value, as opposed to task accuracy. This is because loss is the exact quantity underlying gradients, and as such highlights the direct link between RNN dynamics and learning. Nevertheless, we verify that the phenomena reported here persist when measuring task accuracy instead, and refer the reader to **Supplementary Figure 4** in the Appendix for further details. Networks with a maximum LE closer to zero indeed performed better. Of these 60 spectra, we selected a smaller subset of 6 that spanned the range of the gross shapes. Interestingly, they also spanned the range of losses, suggesting there is a consistent signal about the loss encoded in the complex, yet systematic variation of the maximum Lyapunov exponent with hyperparameters. The gross shape of these spectra correlated less with the loss, supporting our hypothesis. A straightforward interpretation of the signal we have observed is that the slower the contraction of the effect of an input on the activity in the activity mode associated with this maximum LE, the longer gradient information can propagate backward in time. This would give the dynamics more predictive power and thus better performance, but it remains to be verified.

Next, we focused on the motion capture prediction task. Here, the higher dimensional data means that the changes that training induces in the shape of the Lyapunov spectra will likely be more complicated. We hypothesized that nevertheless there would be more information in the gross shape of the spectra of the dynamics of trained networks and less in the maximum LE. Here, we vary the initialization parameter from 0.05 to 0.5 while keeping the dropout and learning rates fixed. Indeed, we find (**Figures 4C,D**) consistent variation of the gross shape with loss. We have quantified the variation of loss with gross shape using the mean Lyapunov exponent, though the spectra vary with higher order features as well.

The plots of **Figure 4** illustrate that distinct properties of the LE spectrum correlate with validation performance, depending upon task structure. For the CharRNN task, the maximum exponent being larger (closer to 0), corresponds to better performance. For the Mocap learning task, the larger mean Lyapunov exponent correspond to better performance. Please see D for other spectrum statistics of all networks tested.

The evolution of these statistics over the course of training of the CharRNN task is demonstrated in **Figure 5**. In particular, we observe that both the maximum and mean Lyapunov exponents change systematically with respect to validation loss early in training. While further training leads to further changes in these spectrum statistics, increasing the maximum LE beyond the threshold of -0.5 or decreasing the mean LE beyond the threshold of -4 has little impact on performance as these changes occur later in training, after the loss has converged.

Given our observation that the spectrum changes most rapidly early in training, we believe the most strongly-correlated metrics for a task could allow an alternative way to assess performance of a network early in training.

6. CONCLUSION AND DISCUSSION

In this article, we presented an exposition and example application of Lyapunov exponents for understanding training stability in RNNs. We motivated them as a natural quantity related to stability of dynamics and useful as a complementary approach to existing mathematical approaches for understanding

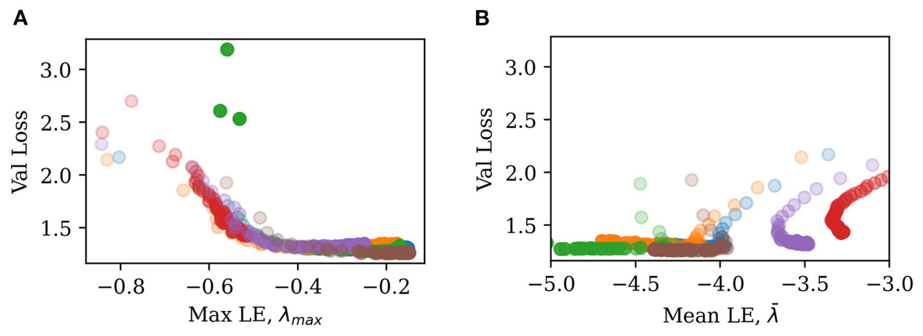

FIGURE 5 | *Evolution of relation between performance and* **(A)** Max LE and **(B)***Mean LE for CharRNN task over the course of training*. The evolution of the spectra highlighted in **Figures 4A,B** are shown in the corresponding color. For each color, the opacity of the points increases with epoch, so the most transparent points are from early in training. For this example, we see that, in general, **(A)** the maximum Lyapunov exponent increases over the course of training, corresponding to a decrease in the validation loss, but the decrease in loss is limited once the maximum increases beyond about -0.5, whereas **(B)** the mean LE tends to decrease over training, but has limited impact when decreasing beyond -4.

training stability focused on the singular value spectrum. We adapted the standard algorithm for LE spectrum computation to RNNs, and showed how it could be made more efficient in standard machine learning development environments. We demonstrated that even the basic implementation runs almost 2 orders of magnitude faster than typical training time [see [11] for further precision and efficiency considerations] and that it converges relatively quickly with learning time. The latter implies it could be useful as a readout of performance early in training. To test this application, we studied it in two tasks with distinct constraints. In both tasks, we find interpretable variation with performance reflecting these distinct constraints.

There are a few points to be made about LEs on RNNs. First, one should consider carefully the degrees of freedom in the architecture under consideration. We have considered the hidden state dynamics of an LSTM such that there are as many Lyapunov exponents as there are units. LSTMs have gating variables that can also carry their own dynamics and could be considered as degrees of freedom, adding another N exponents to the spectrum. While we were able to obtain evidence from the spectra of hidden states alone, recent work [11, 25] suggests that studying the stability in the subspace of gating variables can also be informative. A second point is that one should also consider the role of the input weights $\mathbf{U}$ [c.f. Equation (1)]. The strength of inputs (e.g., input signal variance) that drive high-dimensional dynamics has long been known to have a stabilizing effect [16, 26, 27]. Thus, the input statistics of the task's data can play a role that is modulated by $\mathbf{U}$, making the latter a target for gradient-based algorithms aiming to decrease loss. Finally, we raise perhaps the most glaring adaptation: that of approximating this asymptotic quantity to settings of finite-length sequences. We proposed to use averaging over inputs, in addition to averaging over initial conditions, as a way to take advantage of batch tensor mechanics to achieve faster LE estimates. This is in contrast to long-time averages used in the LE definition, although our method is justified under assumptions of ergodicity and short transient dynamics. Analyzing the influence of input batch size on this precision is a topic of future research.

The task analysis we have performed suggests a use case for the Lyapunov spectrum as having features that serve as an early readout of performance useful to speed up hyperparameter search. We examined the maximum and mean Lyapunov exponents as two features highlighted by hypotheses we made based on our knowledge of the task constraints. Presumably, there are others relevant here and in other tasks. The search for generic features that do not require knowledge of task constraints (e.g., dynamical isometry) is important, as is demonstrating how useful these features are in practice with complex sequence data. One limitation is how quickly they converge with learning. For example, as a composite quantity, the mean LE converges rather quickly, while the maximum LE is significantly slower to converge. In [11], it is shown that loss function/learning rule combinations can also sculpt the spectra and alter its statistics, demonstrating that these are more sources of variation to understand. Interestingly, Full Force [28], an alternative to Back-propagation-through-time, has a qualitatively similar spectra (same max, min, and mean LE) but with much lower variance, presumably a desirable feature for generalization. This method also demands more information about performance than just a gradient, suggesting the interesting hypothesis that the precision with which the features of the Lyapunov spectra can be sculpted can scale with the amount of information provided in training. Finally, recent work has given theoretical grounding to why LSTMs avoid vanishing gradients [25]. Of course this was the reason for the design of LSTM so it only recapitulates the original design intuition. Looking forward, however, as more complex architectures are developed with less interpretable design, this approach based on LEs can still provide novel insight into their inner workings.

We close with a short discussion of open problems as this is a prominent area in understanding network dynamics at the intersection of machine learning and computational neuroscience. Having supporting analytical results is essential for robust control of complex problems. To this end, understanding how and where the assumption of untied weights breaks down and how forward propagation differs from backward propagation will be important in extending analytical results on spectral constraints. The Lyapunov spectrum can guide this work. Last, insightful parallels into theoretical neuroscience work could be made. For example, Lyapunov spectrum have been derived for additional degrees of freedom in the unit dynamics, different connectivity ensembles, and more. Also, a discrepancy between the loss of linear stability and the onset of chaotic dynamics in driven systems must be understood [27]. Making these connections explicit will serve both fields.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. This data and code for computing the LE spectrum for RNN can be found at: https://github.com/lyapunov-hyperopt/lyapunov_hyperopt.

AUTHOR CONTRIBUTIONS

ES and GL conceptualized the study and obtained funding. RV developed the computational tools in the Github repository and generated the reported experiments. MP developed the theoretical background and motivation. ES conceptualized the experimental design and supervised RV. GL conceptualized the theoretical motivation, experimental design, and supervised MP. All authors contributed to the article, its revision, and approved the final version.

ACKNOWLEDGMENTS

ES and RV acknowledge the support of the National Science Foundation under (Grant No. DMS-1361145) and partial support by the Departments of Applied Mathematics, Electrical, and Computer Engineering, the Center of Computational Neuroscience (CNC), and the eScience Center at the University of Washington. ES acknowledges the support of National Science Foundation HDR Institute Accelerated AI Algorithms for Data-Driven Discovery (Grant No. OAC-2117997) and Washington Research Foundation Fund. MP acknowledges support from IVADO. GL acknowledges support from a Canada CIFAR AI Chair, NSERC Discovery grant (RGPIN-2018-04821), and FRQNT Young Investigator Startup Program (2019-NC-253251). This manuscript has been released as a pre-print: https://arxiv.org/abs/2006.14123.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fams.2022.818799/full#supplementary-material

REFERENCES

1. Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans Neural Netw.* (1994) 5:157–66.
2. Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks. In: *International Conference on Machine Learning.* Atlanta, GA (2013). p. 1310–8.
3. Chen M, Pennington J, Samuel. Gating enables signal propagation in recurrent neural networks. In: *ICML.* Stockholm (2018).
4. Pennington J, Schoenholz S, Ganguli S. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In: *Advances in Neural Information Processing Systems.* Long Beach, CA: Advances in Neural Information Processing Systems (2017). p. 4785–95.
5. Poole B, Lahiri S, Raghu M, Sohl-Dickstein J, Ganguli S. Exponential expressivity in deep neural networks through transient chaos. In: *Advances in Neural Information Processing Systems.* Barcelona (2016). p. 3360–8.
6. Yang G. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint* arXiv:190204760. (2019).
7. Zheng Y, Shlizerman E. R-FORCE: robust learning for random recurrent neural networks. *arXiv preprint* arXiv:200311660. (2020).
8. Legenstein R, Maass W. Edge of chaos and prediction of computational performance for neural circuit models. *Neural Netw.* (2007) 20:323–34. doi: 10.1016/j.neunet.2007.04.017
9. Pennington J, Schoenholz SS, Ganguli S. The emergence of spectral universality in deep networks. *arXiv preprint* arXiv:180209979. (2018).
10. Laurent T, von Brecht J. A recurrent neural network without chaos. *arXiv preprint* arXiv:161206212. (2016).
11. Engelken R, Wolf F, Abbott L. Lyapunov spectra of chaotic recurrent neural networks. *arXiv preprint* arXiv:200602427. (2020).
12. Gilboa D, Chang B, Chen M, Yang G, Schoenholz SS, Chi EH, et al. Dynamical isometry and a mean field theory of LSTMs and GRUs. *arXiv preprint* arXiv:190108987. (2019).
13. Monteforte M, Wolf F. Dynamical entropy production in spiking neuron networks in the balanced state. *Phys Rev Lett.* (2010) 105:1–4. doi: 10.1103/PhysRevLett.105.268104
14. Puelma Touzel M. *Cellular Dynamics and Stable Chaos in Balanced Networks.* Gottingen: Georg-August University (2015).
15. Arnold L. *Random Dynamical Systems.* Berlin: Springer (1991). Available online at: http://arxiv.org/abs/math/0608162
16. Lajoie G, Lin KK, Shea-Brown E. Chaos and reliability in balanced spiking networks with temporal drive. *Phys Rev E.* (2013) 87:1–5. doi: 10.1103/PhysRevE.87.052901
17. Hennequin G, Vogels T, Gerstner W. Non-normal amplification in random balanced neuronal networks. *Phys Rev E.* (2012) 86:1–12. doi: 10.1103/PhysRevE.86.011909
18. Kerg G, Goyette K, Puelma Touzel M, Gidel G, Vorontsov E, Bengio Y, et al. Non-normal Recurrent Neural Network (nnRNN): learning long time dependencies while improving expressivity with transient dynamics. In: Wallach H, Larochelle H, Beygelzimer A, d' Alché-Buc F, Fox E, Garnett R, editors. *Advances in Neural Information Processing Systems 32.* Vancouver, BC: Curran Associates, Inc. (2019). p. 13613–23.
19. Liu GH, Theodorou EA. Deep learning theory review: an optimal control and dynamical systems perspective. *arXiv preprint* arXiv:190810920. (2019).
20. Benettin G, Galgani L, Giorgilli A, Strelcyn JM. Lyapunov characteristic exponents for smooth dynamical systems and for Hamiltonian systems; a method for computing all of them. Part 1: Theory. *Meccanica.* (1980) 15:9–20.
21. Dieci L, Van Vleck ES. Computation of a few Lyapunov exponents for continuous and discrete dynamical systems. *Appl Numer Math.* (1995) 17:275–91.
22. Karpathy A, Johnson J, Fei-Fei L. Visualizing and understanding recurrent networks. *arXiv preprint* arXiv:150602078. (2015).

23. Li C, Zhang Z, Sun Lee W, Hee Lee G. Convolutional sequence to sequence model for human dynamics. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* Salt Lake City, UT (2018).
24. Henaff M, Szlam A, LeCun Y. Recurrent orthogonal networks and long-memory tasks. *arXiv preprint* arXiv:160206662. (2016).
25. Can T, Krishnamurthy K, Schwab DJ. Gating creates slow modes and controls phase-space complexity in GRUs and LSTMs. *arXiv preprint* arXiv:200200025. (2020).
26. Molgedey L, Schuchhardt J, Schuster HG. Suppressing chaos in neural networks by noise. *Phys Rev Lett.* (1992) 69:3717–9. doi: 10.1103/PhysRevLett.69.3717
27. Schuecker J, Goedeke S, Helias M. Optimal sequence memory in driven random networks. *Phys Rev X.* (2018) 8:41029. doi: 10.1103/PhysRevX.8.041029
28. DePasquale B, Cueva CJ, Rajan K, Escola GS, Abbott L. full-FORCE: a target-based method for training recurrent networks. *PLoS ONE.* (2018) 13:e0191527. doi: 10.1371/journal.pone.0191527