SaFARi: State-Space Models for Frame-Agnostic Representation

Anonymous authors
Paper under double-blind review

Abstract

State-Space Models (SSMs) have re-emerged as a powerful tool for online function approximation, and as the backbone of machine learning models for long-range dependent data. However, to date, only a few polynomial bases have been explored for this purpose, and the state-of-the-art implementations were built upon the best of a few limited options. In this paper, we present a generalized method for building an SSM with any frame or basis, rather than being restricted to polynomials. This framework encompasses the approach known as HiPPO, but also permits an infinite diversity of other possible "species" within the SSM architecture. We dub this approach SaFARi: SSMs for Frame-Agnostic Representation.

1 Introduction

Modeling sequential data is a cornerstone of modern machine learning, with applications spanning natural language processing, speech recognition, video analysis, and beyond (Zubić et al., 2024; Alemohammad et al., 2021; Nguyen et al., 2022). A fundamental challenge in these domains is the efficient representation of long-range dependence in time-series data, where the goal is to capture and preserve the essential features of the input signal necessary for downstream tasks over extended time horizons while maintaining computational tractability (Hochreiter & Schmidhuber, 1997).

Machine learning approaches, such as recurrent neural networks (RNNs), struggle to learn long-range dependencies due to limited memory horizons (Elman, 1990; Hochreiter & Schmidhuber, 1997; Schuster & Paliwal, 1997; Pascanu et al., 2013). During backpropagation, gradients are repeatedly multiplied by the same weight matrix, causing them to either shrink exponentially (vanish) or grow exponentially (explode). Vanishing gradients prevent the network from updating weights effectively, while exploding gradients lead to unstable training. Although variants like LSTMs (Graves & Schmidhuber, 2005) and GRUs (Cho et al., 2014) address some of these limitations, they often require task-specific parameterization, and cannot generalize across different sequence lengths or timescales.

State-space models (SSMs) present a power alternative for online representation of sequential data. By design, SSMs enable the online computation of compressive representations, maintaining a constant-size memory footprint regardless of sequence length. The seminal work of Gu et al. introduced High-Order Polynomial Projection Operators (HiPPO), which leverages orthogonal function bases to enable theoretically grounded, real-time updates of sequence representations. This framework, and its subsequent extensions such as S4 and Mamba, have demonstrated remarkable performance in tasks involving long-range dependencies, such as language modeling and signal processing (Gu & Dao, 2023; Gu et al., 2022b; 2023; Gupta et al., 2024; Gu et al., 2022a; Smith et al., 2023; Hasani et al., 2023). By formulating sequence representation as an online function approximation problem, HiPPO provides a unified perspective on memory mechanisms, offering both theoretical guarantees and practical efficiency.

However, despite its successes, the HiPPO framework has been limited to specific families of orthogonal polynomials. While these bases are well-suited for certain applications, they are not universally optimal for all signal classes. Fourier bases, for instance, are optimal for smooth, periodic signals due to their global frequency representation. Polynomial bases, such as orthogonal polynomials (e.g., Legendre or Chebyshev), are particularly effective for approximating smooth functions over compact intervals.

The absence of a more flexible basis selection restricts the adaptability of the HiPPO framework. In this work, we address this restriction by presenting a generalized method for constructing SSMs using any frame or basis of choice, which we term SaFARi (SSMs for Frame-Agnostic Representation). Our approach extends and generalizes the HiPPO framework using a numerical (as opposed to closed-form) method, which enables us to relax the requirements on the basis (such as orthogonality of the components).

Our key contributions are as follows:

- **Generalized SSM construction:** We present SaFARi, a frame-agnostic method for deriving SSMs associated with any basis or frame, generalizing the HiPPO framework to a broader class of function representations.
- **Error Analysis:** We provide a comprehensive discussion of SaFARi's error sources and derive error bounds, offering theoretical insights into its performance and limitations.

This paper is organized as follows. In Section 2, we review the HiPPO framework and its limitations, motivating the need for a generalized approach. Section 3 provides the required mathematical preliminaries. Section 4 introduces our frame-agnostic method for SSM construction, and then Section 5 addresses the implementation considerations and strategies for SaFARi, including the approximation of its infinite-dimensional representation in finite dimensions, and provides a rigorous theoretical analysis of the associated errors. Finally, Section 7 discusses the broader implications of our work and outlines directions for future research.

2 Background

Recent advances in machine learning, computer vision, and LLMs have exploited the ability to collect and contextualize more and more data over longer time frames. Handling such sequential data presents three main challenges: 1) generating a compact and low-dimensional representation of the sequence, 2) effectively preserving information within that representation, and 3) enabling real-time updates for streaming data.

The classic linear method of obtaining the coefficients of a compressed representation of a signal is through a transform (e.g. Fourier) (Oppenheim, 1999; Abbate et al., 2012; Box et al., 2015; Proakis, 2001; Prandoni & Vetterli, 2008). However, a significant limitation of this method is its inefficiency in handling real-time updates. When new data arrives, the representation must be recalculated in its entirety, necessitating the storage of all prior data, which is sub-optimal in terms of both computation and storage requirements. This limits the horizon of the captured dependencies within the sequence.

Nonlinear models, such as recurrent neural networks (RNNs) and their variants, have been introduced more recently (Elman, 1990; Hochreiter & Schmidhuber, 1997; Cho et al., 2014; Schuster & Paliwal, 1997). Since these learned representations are task-specific, they are not easily utilized for other circumstances or applications. Furthermore, RNNs struggle to capture long-range dependencies due to issues such as vanishing and exploding gradients.

2.1 State-space models

The state-space representation itself is not new; it was introduced by Kálmán (1960) via the eponymous Kalman Filter. For an input $u(t)$, output $y(t)$, and a state representation called $x(t)$, many systems and their properties can be described and controlled with the following system of linear equations:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t). \end{aligned} \tag{1}$$

In many classic applications, we iteratively update the matrices A, B, C, and D to control or predict the output y based on previous values of u. For online function approximation, however, we instead *define* the matrices A and B such that they iteratively update a vector of coefficients c over a particular basis. For the moment, we can ignore C and D (or, equivalently, consider C to be an identity matrix and $D = 0$). For stability, A must have only negative eigenvalues, so we explicitly include a negative sign here. A and B may or may not be constant over time, so for completeness, we call these $A(t)$, $B(t)$. Eq. 1 is now

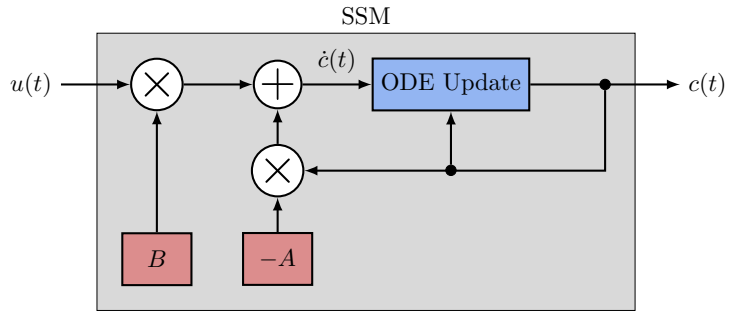

Figure 1: An SSM block-diagram, with the necessary ODE update step included.

$$\dot{c} = -A(t)c(t) + B(t)u(t). \tag{2}$$

The challenge in the problem of the approximation of online functions is to derive appropriate matrices A and B, which was answered by HiPPO (Gu et al., 2020).

2.2 HiPPO: High-order Polynomial Projection Operators

The HiPPO framework (Gu et al., 2020) enables online function approximation using pre-determined SSM parameters derived from a basis of orthogonal polynomials $\mathcal{G}$. It optimizes $\|u_T - g^{(T)}(t)\|_\mu$ for $g^{(T)}(t) \in \mathcal{G}$, to find a set of coefficients for the orthogonal polynomials at every time T, which yields the approximation of the input function over $[0, T]$.

In addition to choosing the set of polynomials, one must also select a measure: the combination of a weighting function and a windowing function. The window indicates which samples are included, and the weighting assigns a weight to each sample. HiPPO (Gu et al., 2020) considered several possible measures, two of which (illustrated in Fig. 2) are:

$$\mu_{tr}(t) = \frac{1}{\theta}\mathbb{1}_{t\in[T-\theta,T]}, \quad \mu_{sc}(t) = \frac{1}{T}\mathbb{1}_{t\in[0,T]}. \tag{3}$$

The uniform translated measure, $\mu_{tr}(t)$, gives equal weight to all samples in the most recent window with a constant length θ, and zero weight to previous samples. The uniform scaled measure, $\mu_{sc}(t)$, gives equal weight to all the times observed since $t = 0$. This can be interpreted as squeezing or stretching the basis or frame to match the current length of the signal at any given time. Thus, the representation produced by this measure becomes less informative about the finer details of the signal as more history is observed, since the stretching of the basis will gradually increase the lowest observable frequency. Gu et al. also considered another weighting function, $\mu_{ed}(t)$, which is a translated exponential decay measure that gives exponentially less weight to earlier parts of the signal history. This work considers only uniformly weighted measures. We leave alternative weighting schemes to future work.

For the space of orthogonal polynomials, Hippo ODE update (Gu et al., 2020) (see Fig. 1) follows a first-order linear differential equation:

$$\frac{d}{dT}\vec{c}(T) = -A_{(T)}\vec{c}(T) + B_{(T)}u(T) \tag{4}$$

where $u(T)$ is the value of the input signal at the current time T, and $\vec{c}(T)$ is the vector containing the representation of the $t \in [0, T]$ part of the input signal. $A_{(T)}$ and $B_{(T)}$ are also a time-varying matrix and vector that can be derived for the particular choice of the measure and orthogonal polynomial.

Solving the differential equation to update the SSM: The differential equation (Eq. 4) can be solved incrementally, and there are several methods to do so Butcher (1987). We use the *generalized bilinear transform (GBT)* (Zhang et al., 2007), relying on findings from Gu et al. (2020) that the GBT produces the best numerical error in solving first-order SSMs. The GBT update rule is

$$c(t + \Delta t) = (I + \delta t\alpha A_{(t+\delta t)})^{-1}\,(I - \delta t(1-\alpha)A_{(t)})c(t) \;+\; (I + \delta t\alpha A_{(t+\delta t)})^{-1}\,\delta t B(t)u(t), \tag{5}$$

where $0 \leq \alpha \leq 1$ is a chosen constant. (Note that for $\alpha = 0$, the update rule becomes the same as the forward Euler method.)

Diagonalizing the transition matrix A: The incremental update given by the GBT requires a matrix inversion and matrix products at each increment. Having a diagonal A significantly reduces the computational cost of these operations. If the measure used for the SSM is such that the eigenvectors of $A(t)$ remain constant (for example, if $A(t)$ is a constant matrix multiplied by an arbitrary function of time), then it is possible to find a change of basis that makes the SSM matrix diagonal. To do this, one finds the eigenvalue decomposition of the matrix $A(t) = V\Lambda(t)V^{-1}$ and re-write the SSM as

$$\frac{\partial}{\partial t}\widetilde{c} = -\Lambda(t)\widetilde{c} + \widetilde{B}u(t), \tag{6}$$

where $\widetilde{c} = V^{-1}c$ and $\widetilde{B} = V^{-1}B(t)$. This means that one can solve Eq. 6 instead of Eq. 4, then find the representation c from $\widetilde{c}$ with a single matrix multiplication.

2.3 Limitations of HiPPO

The original HiPPO formulation and a subsequent follow-up (Gu et al., 2023) included a handful of orthogonal polynomial bases with specific measures. Measures were chosen heuristically for each basis, so there is no method in the literature for arbitrary choice of basis and measure (see Table 1). Moreover, it was found that many of the bases explored in the early work actually perform quite poorly for function approximation tasks. Only one, the scaled-Legendre (LegS), empirically performed well, but it introduced additional challenges as its A matrix is not stably diagonalizable (Gu et al., 2022a).

The majority of the follow-up work since HiPPO has abandoned the task of function approximation by SSMs alone. Instead, most research has employed a diagonal approximation of the best-performing extant version (LegS) as an *initialization* for machine learning architectures such as S4 (Gu et al., 2022b; Smith et al., 2023) and Mamba (Gu & Dao, 2023). Still, the HiPPO framework still holds untapped potential for online function approximation and better initializations for learning tasks.

3 Mathematical Preliminaries

Prior to introducing SaFARi, we first cover some required theoretical background on the use of frames for function representation and approximation. This section will cover distinctions between approximations performed on the full signal all at once, and approximation performed in a sequential (online) fashion.

3.1 Function representation using frames

Given a function $u(t)$ of a time domain $t \in \mathcal{D}_T$, we aim to represent $u(t)$ using a collection of functions over $\mathcal{D}_T$. This representation is performed with respect to a measure that determines the relative importance of

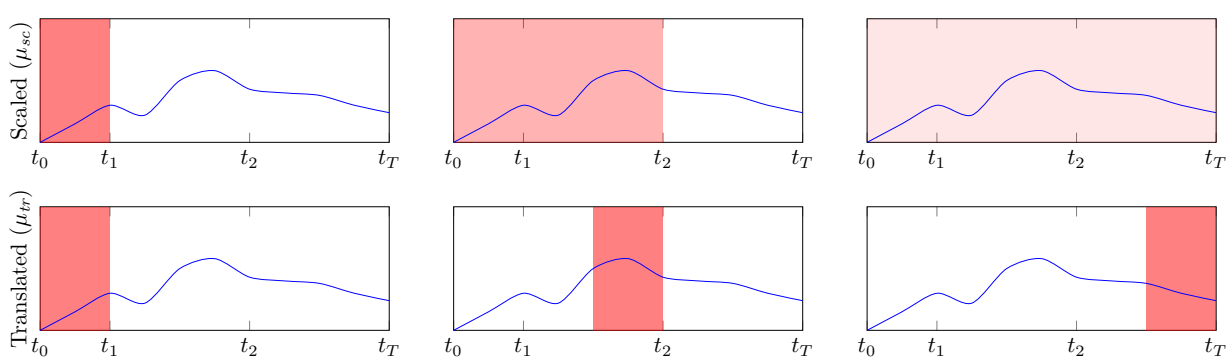

Figure 2: Two different uniform measures (red) applied to a signal (blue). The red shaded area demonstrates how the measure changes as time evolves and more samples of the input are observed.

	Measure	Scaled	Translated
Basis or Frame	Legendre	Gu et al. (2020),§4 *Gu et al. (2023)	Gu et al. (2020),§4
	Fourier	Gu et al. (2020), §4	Gu et al. (2020),§4
	Laguerre	§4	**Gu et al. (2020),§4
	Chebychev	§4	**Gu et al. (2020)§4
	Arbitrary	§4	§4

Table 1: An overview of the combinations of frames (or bases) and measures covered in the literature to date. SaFARi fills in missing combinations in this table for the scaled and translated measures in Section 4. Note that the S4 work and beyond (denoted by *), which is placed in this table under Uniform/Scaled, technically applies an exponential measure in the ODE; however, the A matrix of the SSM is generated based on a uniformly weighted scaled Legendre polynomial basis. Some implementations in the translated case (denoted by **) also apply an exponential decay measure. We consider only the case of uniform weighting for each measure for this foundational work, but note that there are infinite alternate weighting schemes for future work to explore.

the function's value at different times. We formulate the task of function representation using frames via the definitions below.

Frame: The sequence $\Phi = \{\phi_n\}_{n\in\Gamma}$ is a frame in the Hilbert space $\mathcal{H}$ if there exist two constants $A > 0$ and $B > 0$ such that for any $f \in \mathcal{H}$:

$$A_{\text{frame}}\|f\|^2 \leq \sum_{n\in\Gamma} |\langle f, \phi_n\rangle|^2 \leq B_{\text{frame}}\|f\|^2 \tag{7}$$

where $\langle\cdot,\cdot\rangle$ is the inner product of the Hilbert space $\mathcal{H}$, and Γ denotes the indices of the frame elements in Φ. If $A_{\text{frame}} = B_{\text{frame}}$, then the frame is said to be tight (Mallat, 2008; Christensen, 2003; Gröchenig, 2001).

Frame operator: If the sequence $\Phi = \{\phi_n\}_{n\in\Gamma}$ is a frame, then its associated operator $\mathbb{U}_\Phi$ is defined as:

$$\mathbb{U}_\Phi f = \vec{x}, \qquad x_n = \langle f, \phi_n\rangle, \quad \forall n \in \Gamma. \tag{8}$$

It can be shown that the frame condition (Eq. 7) is necessary and sufficient for the frame operator (Eq. 8) to be invertible on its image with a bounded inverse. This makes the frame a complete and stable (though potentially redundant) framework for signal representation.

Dual frame: The set $\widetilde{\Phi} = \{\widetilde{\phi}_n\}_{n\in\Gamma}$ is the dual of the frame Φ, $\widetilde{\Phi} = \text{Dual}\{\Phi\}$ if:

$$\widetilde{\phi}_n = (\mathbb{U}_\Phi^*\mathbb{U}_\Phi)^{-1}\phi_n \tag{9}$$

where $\mathbb{U}_\Phi^*$ is the adjoint of the frame operator: $\langle \mathbb{U}_\Phi f, x\rangle = \langle f, \mathbb{U}_\Phi^* x\rangle$.

It can be shown that the composition of the dual frame and the frame is the identity operator in the Hilbert space $\mathcal{H}$: $\mathbb{U}_{\widetilde{\Phi}}^*\mathbb{U}_\Phi f = f$ (Mallat, 2008; Christensen, 2003; Gröchenig, 2001). Thus, we can think of the dual frame as the operator that transforms the signal from frame representation back into the signal space.

Function approximation using frames: The compressive power of frame-based function approximation lies in its ability to efficiently represent functions using a relatively small number of frame elements. Different classes of functions exhibit varying effectiveness in capturing the essential features of different signal classes. This efficiency is closely tied to the decay rate of frame coefficients, which can differ significantly between frames for a given input function class. As a result, selecting an appropriate frame is critical for optimal approximations while minimizing computational resources and storage space.

In the task of online function representation, we aim to represent a function $u_T := \{u(t), t \in [0, T]\}$ for any given time T, using a frame Φ that has the dual $\widetilde{\Phi}$ and the domain $t \in \mathcal{D}$. To do so, we need an invertible

warping function so that the composition of the frame and the warping function $\Phi^{(T)}$ includes the domain $[0, T]$. Without loss of generality, we assume that the frame has the domain $\mathcal{D}_\Phi = [0, 1]$, and use the warping $\phi^{(T)}(t) = \phi\left(\frac{t}{T}\right)$. (If this is not the case, then one can easily find another warping function to warp $\mathcal{D}_\Phi \to [0, 1]$ and apply it beforehand.) To calculate the representation, the frame operator of Φ_T acts on u_T:

$$\textit{Projection:} \quad x = \Phi^{(T)} u_T, \quad x_n = \langle u_T, \phi_n^{(T)} \rangle = \int_{t=0}^{T} u(t) \overline{\phi_n^{(T)}(t)} \mu(t) dt, \tag{10}$$

where the overline in $\overline{\phi}(t)$ is the complex conjugate of $\phi(t)$. Then, the dual frame operator transforms the discrete representation back to the function u_T:

$$\textit{Reconstruction:} \quad u_T = \widetilde{\Phi}^{(T)} \Phi^{(T)} u_T, \quad u_T(t) = \sum_{n \in \Gamma} \langle u_T, \phi_n^{(T)} \rangle \widetilde{\phi}_n^{(T)}(t) \,. \tag{11}$$

Measures: Throughout this paper, we work with the Hilbert space consisting of all functions over the domain $\mathcal{D}$ with the inner product defined as:

$$\langle u, v \rangle = \int_{t \in \mathcal{D}} u(t) \overline{v(t)} \mu(t) dt. \tag{12}$$

We will consider both scaling and translating windows with uniformly weighted measures as described in Eq. 3 and Fig. 2, and derive the SSMs for each. We do not implement weighting schemes other than uniform in this work explicitly, but any other weighting function can be implemented with our method by using Eq. 12 and the appropriate SSM derivation for a scaled or translated window.

4 SaFARi

We now introduce a generalized method for online function representation using arbitrary frames. We formulate the update rule for online approximation as a first-order linear differential equation following a state-space model, and provide templates for the uniform translated and scaled measures.

4.1 Formulation

For the given frame Φ (without loss of generality, we assume Φ has the domain $\mathcal{D} = [0, 1]$), and an input function $u(T)$, the objective is to find the representation of the function using the frame Φ similar to Eq. 10:

$$\textit{Projection:} \quad c_n(T) = (\mathbb{U}_\Phi u)_n = \int_{t=t_0}^{T} u(t) \overline{\phi_n^{(T)}(t)} \mu(t) dt \,. \tag{13}$$

The representation vector $\vec{c}(T)$ is a vector with its nth component defined as above. We next find the derivative of the representation with respect to the current time T, and show that it follows a particular SSM for a scaled or translated measure.

4.2 Uniform scaled measure μ_{sc}

When we use the scaled measure ($\mu_{sc}(t)$ in Eq. 3), the representation generated by applying the frame operator to the observed history of the input $u_T(t)$ is

$$c_n(T) = \int_{t=0}^{T} u(t) \overline{\phi_n}\left(\frac{t}{T}\right) \frac{1}{T} dt \,. \tag{14}$$

Definition 1. For a given frame Φ consisting of functions on $\mathcal{D}$, we define the auxiliary derivative of the frame $\Upsilon_\Phi = \{\upsilon_n\}_{n \in \Gamma}$ as a collection having $\upsilon_n = t \frac{\partial}{\partial t} \phi_n(t)$.

The auxiliary derivative of the frame is the result of the operator $t \frac{\partial}{\partial t}$ acting on each individual frame element. Note that the auxiliary derivative of the frame is not necessarily a frame itself and the frame condition in Eq. 7 does not necessarily hold for Υ_Φ.

Theorem 1. *For the representation defined in Eq. 14, the partial derivative of $\vec{c}$ with respect to T is*

$$\frac{\partial}{\partial T} \vec{c}(T) = -\frac{1}{T} A \vec{c}(T) + \frac{1}{T} B u(T), \quad A = I + \mathbb{U}_\Upsilon \mathbb{U}_{\widetilde{\Phi}}^* \tag{15}$$

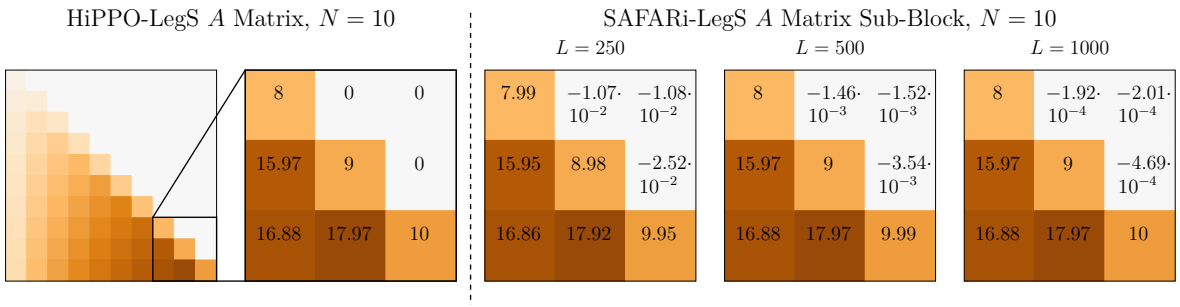

Figure 3: HiPPO provides a closed-form solution for the scaled Legendre (LegS) SSM. SaFARi provides a computed solution, where the accuracy depends on the discretization of the $N \times L$ frame. Larger L gives a finer discretization of the basis vectors and thus a better numerical result.

where B is the complex conjugate of a vector containing members of the main frame evaluated at $T = 1$ so that $B = \{\overline{\phi}_n(T = 1)\}_{n \in \Gamma}$. The A operator can also be described as a matrix

$$A_{i,j} = \delta_{i,j} + \int_0^1 \left[t \frac{\partial}{\partial t} \overline{\phi}_i(t) \right]_{t=t'} \widetilde{\phi}_j(t') dt'. \tag{16}$$

Proof is provided in the Appendix A.1. We will refer to the SSM with a scaling measure as scaled-SaFARi.

4.3 Uniform translated measure μ_{tr}

When the translated measure is used ($\mu_{tr}(t)$ in Eq. 3), the representation resulting from applying the frame operator to the window of recently observed input $u_T(t)$ is

$$c_n(T) = \int_{t=T-\theta}^{T} u(t) \overline{\phi_n} \left(\frac{t - (T - \theta)}{\theta} \right) \frac{1}{\theta} dt \,. \tag{17}$$

Definition 2. For a given frame $\Phi = \{\phi_n\}_{n \in \Gamma}$ consisting of functions of time, we define the time derivative of the frame $\dot{\Phi} = \{\dot{\phi}_n = \frac{\partial}{\partial t} \phi_n(t)\}_{n \in \Gamma}$ as a collection of time derivatives of ϕ_n components.

Definition 3. For a given frame $\Phi = \{\phi_n\}_{n \in \Gamma}$ consisting of functions of time, the zero-time frame operator is similar to the frame operator but only acts on $t = 0$ instead of the integral over the entire domain:

$$\mathbb{Q}_\Phi f = \vec{x}, \quad x_n = f(t = 0) \overline{\phi}_n(t = 0) \,. \tag{18}$$

Theorem 2. *For the representation defined in Eq. 14, the partial derivative of $\vec{c}$ with respect to T is*

$$\frac{\partial}{\partial T} \vec{c}(T) = -\frac{1}{\theta} A \vec{c}(T) + \frac{1}{\theta} B u(T), \quad A = \mathbb{U}_{\dot{\Phi}} \mathbb{U}^*_{\widetilde{\Phi}} + \mathbb{Q}_\Phi \mathbb{Q}^*_{\widetilde{\Phi}} \tag{19}$$

where B is the complex conjugate of a vector containing members of the main frame evaluated at $T = 1$ so we have $B = \{\overline{\phi}_n(T = 1)\}_{n \in \Gamma}$. The A operator can also be described using the matrix representation

$$A_{i,j} = \overline{\phi}_i(0) \widetilde{\phi}_j(0) + \int_0^1 \left[\frac{\partial}{\partial t} \overline{\phi}_i(t) \right]_{t=t'} \widetilde{\phi}_j(t') dt' \,. \tag{20}$$

Proof is provided in Appendix A.2. We will refer to the SSM with a scaling measure as translated-SaFARi.

4.4 SaFARi as generalization of HiPPO

HiPPO provides exact, closed-form solutions for A and B for a few specific basis and measure combinations. SaFARi replicates these A and B matrices to within some numerical error caused by discretization of the frame vectors into length L. Increasing L will provide matrices that converge toward the closed-form solution (see Fig. 3). When a closed-form solution exists for the desired basis and measure (e.g., HiPPO-LegS for the

scaled Legendre basis), then it is preferable to use it (see the derivations in Gu et al. (2020; 2023)).[1] SaFARi provides a method for *any* basis/measure where the closed-form solution might not exist. We also show that SaFARi preserves all of HiPPO's robustness to timescale when applied to a general frame in Appendix A.4.

5 Implementation Considerations

This section describes the computational efficiency and accuracy concerns of SaFARi, including strategies for producing the finite-dimensional approximation of the complete infinite-dimensional SaFARi in Section 5.1. We analyze the errors introduced by these approximations in Section 5.2.

5.1 Truncation of frames

Section 4 demonstrates how a particular SSM can be built from an arbitrary frame Φ. Since the input space for the SSM is the class of functions of time, no Φ with a finite number of elements can meet the frame condition (Eq. 7), since the true representation of the input signal is infinite-dimensional. In practice, the representation reduces to the truncated representation. In this section, we analyze the theoretical implications of truncated representation using SaFARi.

5.1.1 Finite-dimensional approximation of SaFARi

In the finite-dimensional case, we will use only N elements of a frame. Partial representation of size N requires that the resulting representation approximates the infinite-dimensional representation. We call the SSM with its $\vec{c}$ having N coefficients $\mathsf{SaFARi}^{(N)}$.

Definition 4. A $\text{SaFARi}^{(N)}$ sequence is a sequence of the pairs $[A^{(N)}, B^{(N)}]$ where $A^{(N)} \in \mathbb{C}^{N\times N}$ and $B^{(N)} \in \mathbb{C}^N$ such that sequence converges to $[A, B]$ of SaFARi as

$$\lim_{N\to\infty} A^{(N)} = A, \qquad \lim_{N\to\infty} B^{(N)} = B\,. \tag{21}$$

It's trivial that this results in a finite-dimensional approximation of the infinite-dimensional representation. Thus, any arbitrary precision can be achieved by selecting the appropriate truncation level.

Definition 4 is not a constructive definition; that is, it does not uniquely determine $[A^{(N)}, B^{(N)}]$. In fact, there are many such sequences that all converge to $[A, B]$ of SaFARi. Of course, this does not mean that all such sequences produce equal representation error. We present and analyze two different constructions below.

5.1.2 Truncation of dual (ToD)

The first construct of a $\text{SaFARi}^{(N)}$ sequence is through finding the infinite dimensional A and B, then truncate to size N

$$A^{(N)} = A_{[0:N,0:N]}, \quad B^{(N)} = B_{[0:N]}\,. \tag{22}$$

This construction results in a sequence that approximates the infinite-dimensional SaFARi according to Definition 4. The practical way of finding the truncated A, B involves finding $A_{i,j}$ as introduced in Eq. 16 and Eq. 20 for $i, j < N$.

Note that calculating $A_{i,j}$ requires finding $\widetilde{\Phi}$, the dual of the (infinite-dimensional) frame Φ. For certain families of functions, the dual frame can be found analytically. However, if an analytical way of finding $\widetilde{\Phi}$ is not known, then one must use a numerical approximation of the dual frame. In this case, the construction for $[A^{(N)}, B^{(N)}]$ involves forming the truncated frame for a much bigger size $N_2 \gg N$, then finding $\widetilde{\Phi}^{(N_2)} = \text{Dual}\{\Phi_{[0:\text{N}_2]}\}$ numerically as an approximation for dual frame ($\widetilde{\Phi}^{(N_2)} \approx \widetilde{\Phi}$). Next, we truncate the approximate dual and use its first N elements as an approximation for the first N elements of the dual frame in constructions in Eq. 16 and Eq. 20.

[1] Note that HiPPO used the convention of absorbing a negative sign from the ODE in Eq. 4 into the A matrix, where we do not. Additionally, authors in Gu et al. (2020; 2023) derived multiple versions for some A matrices (e.g., FouT), and we find some minor discrepancies with our derivation. See Appendix A.3 for details.

5.1.3 Dual of truncation (DoT)

The ToD construction becomes numerically intractable for cases where the dual frame is not analytically known; this motivates the need for an alternate constructor for SaFARi$^{(N)}$. To construct this sequence:

1. Truncate the frame at level N and form $\Phi^{(N)} = \{\phi_i\}_{i<N}$.
2. Numerically approximate the dual of the truncated frame $\widetilde{\Phi}^{(N)} = \text{Dual}\{\Phi^{(\text{N})}\}$ using the pseudo-inverse.
3. Use $\Phi^{(N)}$ and $\widetilde{\Phi}^{(N)}$ in Eq. 16 or Eq. 20 to compute $[A^{(N)}, B^{(N)}]$.

The constructed $[A^{(N)}, B^{(N)}]$ is a SaFARi$^{(N)}$ sequence according to Definition 4.

DoT and ToD approximate SaFARi with different rates. In the next sections, we provide a thorough error analysis of SaFARi$^{(N)}$ that enables us to compare the usage of different frames, as well as different construction methods for size N constructs. We then demonstrate that the DoT construction always has a minimum reconstruction error, and is the optimal choice for implementing SaFARi.

5.2 Error analysis

Using truncated representations for online function approximation will result in some reconstruction error, regardless of basis. We focus here on errors emanating from truncation of the representation in an SSM, rather than those caused by sampling, which have been extensively studied in the digital signal processing literature (Oppenheim, 1999).

Let $c^{(\infty)}$ denote the infinite-dimensional representation obtained from an SSM without truncating its associated A, B matrices. When the associated A, B matrices for the SSM are truncated to the first N levels, it produces a truncated representation $c^{(\text{N})}$. This truncation causes two distinct types of error, which we outline below.

5.2.1 Truncation errors

Truncation errors are due to the fact that the truncated frame of size N cannot represent any part of the signal contained in indices $n > N$. This is not limited to SSMs, but is true for any basis representation of a signal. In SaFARi, truncation errors correspond to discarding the green shaded region in the $A^{(N)}$ illustrated in Fig. 4.

5.2.2 Mixing errors

Mixing errors arise from error propagation in the SSM update rule. Specifically, this update involves computing $A\,c$ (as in Eq. 15 and Eq. 19), where the matrix A introduces unwanted mixing between the omitted components ($n > N$) and the retained components ($n \leq N$) of the representation. Consequently, errors from the truncated portion of the representation propagate into the non-truncated portion. This is illustrated by the blue shaded regions in Fig. 4. For the operator A and truncation level N, the contaminating part of operator is $A_{i,j} \quad \forall (i \leq N, j > N)$.

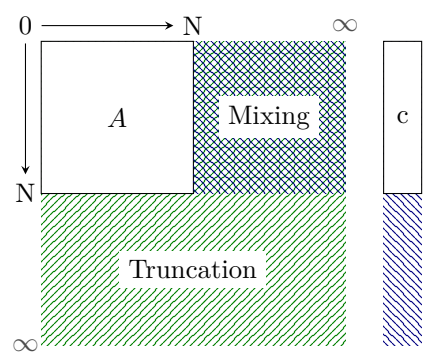

Figure 4: Illustration of error types in an SSM due to frame truncation. Truncation errors arise from signal energy in coefficients of index $n > N$, while mixing errors result from energy blending during the operation $A\,c$.

In the case of a translated measure, this mixing error is exacerbated since each update step requires estimating the initial value in that window (recall Eq. 2). A key insight from our analysis below is that mixing errors have two sources:

1. nonzero components in the upper right quadrant of A, and
2. nonzero coefficients in c at indices greater than N.

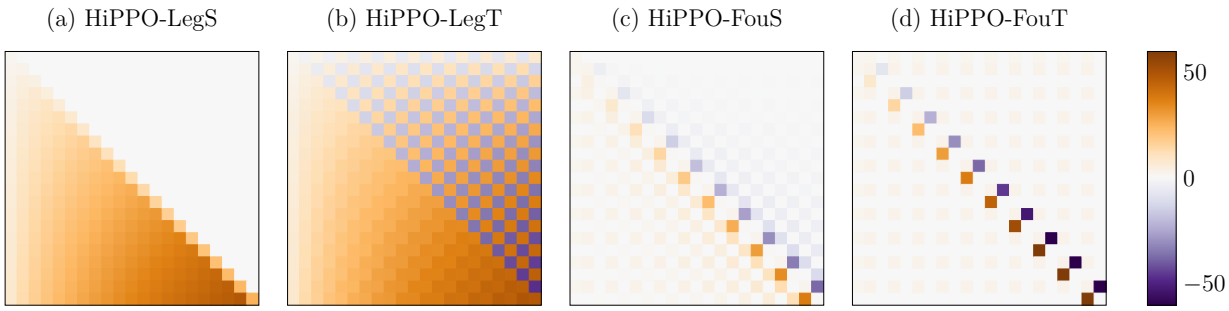

Figure 5: Examples of the A matrices of the HiPPO SSM for several basis/measure combinations: (a) Scaled Legendre, (b) Translated Legendre, (c) Scaled Fourier, (d) Translated Fourier. The dense non-zero elements in the upper right of (b) explain its poor performance compared to (a). The numerous small nonzero elements above the diagonal in (c) and (d) contribute to mixing errors over long sequences.

5.2.3 Mitigating errors

Truncation errors can never be eliminated, but may be alleviated by using a frame that exhibits a rapid decay in the energy carried by higher-order levels of representation.

To counter mixing errors, we should ensure that values in the upper right quadrant of A are as close to zero as possible, and/or that coefficents of c in the blue region of Fig. 4 are as close to zero as possible. If the matrix A is lower-triangular, then any arbitrary truncation results in the contaminating part of A being zero, which guarantees the second type of error is always zero, regardless of any coefficients in c. This is the case for the HiPPO-LegS (scaled Legendre) A matrix, as shown in Fig. 5(a). Indeed, the zero coefficents in the upper right quadrant of A were considered strictly necessary in prior work (Gu et al., 2020; 2023). This restriction explains the continued use of HiPPO-LegS in follow-up works, regardless of whether or not scaled Legendre polynomials are an optimal choice for a given application.

To summarize, there are two primary concerns when finding an appropriate frame for use in an SSM:

1. compatibility between the frame and the given class of input signals, and
2. the operator A that results from a given frame has a small contaminating part.

Truncation and mixing errors have different sources but are linked. The optimal strategy to reduce both at the same time is to choose a basis that results in a representation where the energy is concentrated in the first N coefficients. This can only be achieved, however, if we have some prior knowledge of the input signal in order to choose the right basis and truncation level. In cases where little is known about the input signal or correct truncation level, it is advisable to instead choose an SSM that is zero in the upper right quadrant, such as HiPPO-LegS, as it will inherently negate mixing errors.

5.2.4 Error bound

In order to quantify the mixing error, we show that the truncated representation follows the same differential equation with a perturbation defined by the theorem below.

Theorem 3. *(Poof in Appendix A.6) The truncated representation generated by scaled-SaFARi follows a differential equation similar to the full representation, with the addition of a perturbation factor:*

$$\frac{\partial}{\partial T}c = -\frac{1}{T}Ac + \frac{1}{T}Bu(T) - \frac{1}{T}\vec{\epsilon}(T), \tag{23}$$

where

$$\vec{\epsilon}(T) := \langle u_T, \xi \rangle, \qquad \xi = \Upsilon(\widetilde{\Phi}\Phi - I). \tag{24}$$

The mixing error term $\vec{\epsilon}(T)$ cannot be directly calculated; however, we can derive an upper bound for the mixing error and demonstrate that this bound can be made arbitrarily small with appropriate truncation.

Theorem 4. *If one finds an upper bound such that for all the times before T we have $\|\epsilon(T)\|_2 < \epsilon_m$, then the representation error is bound by*

$$\|\Delta c(T)\|_2 < \epsilon_m \sqrt{\sum \frac{1}{\lambda_i^2}} = \epsilon_m \|A^{-1}\|_F \,; \tag{25}$$

where λ_i are the eigenvalues of A, and F indicates the Frobenius norm. See proof in Appendix A.6.

5.2.5 Error analysis for different SaFARi$^{(N)}$ constructs

In Section 5.1.1 we define SaFARi$^{(N)}$ as a finite dimensional approximation for SaFARi, and provide two particular constructions, DoT and ToD. Armed with the quantification of representation error, we compare these different constructs. We provide Theorem 5 to demonstrate DoT has the optimal representation error between different choices for SaFARi$^{(N)}$ using the same frame.

Theorem 5. *Given a frame Φ, the Dual of Truncation (DoT) construct introduced in Section 5.1.3 has optimal representation error when compared to any other SaFARi$^{(N)}$ construct for the same frame Φ. See proof in Appendix A.6.*

As established by the result of Theorem 5, SaFARi$^{(N)}$ should be constructed with the DoT method.

6 Computational efficiency and runtime complexity

This section develops the computational methods for obtaining sequence representations with SaFARi, emphasizing its efficiency in both training and inference phases. We analyze the computational complexity of different update methods, and highlight the benefit of parallel computation with diagonalizable SSMs. These discussions provide a foundation for understanding the scalability of SaFARi in practical applications.

6.1 Computational complexity for sequential updates

To compute representations using scaled-SaFARi, the GBT update requires solving an $N \times N$ linear system at each step, leading to $O(N^3L)$ complexity for a sequence of length L. In contrast, translated-SaFARi reuses the same inverse across all steps, reducing the complexity to $O(N^2L)$.

If the state matrix A is diagonalizable, both scaled and translated variants can be accelerated via diagonalization. The sequence representation $\widetilde{C}$ is computed for the diagonal SSM and transformed back via $C = V\widetilde{C}$, reducing the complexity to $O(NL)$. On parallel hardware, such diagonalized systems decompose into N independent scalar SSMs, yielding $O(L)$ runtime with sufficient parallel resources.

Diagonalizability of an SSM depends on the frame or basis used in its construction. One limitation of Legendre-based SSMs such as HiPPO is that its A matrix cannot be stably diagonalized, even for representation sizes as small as $N = 20$ (Gu et al., 2022a), leading to significantly higher cost. To address this limitation, (Gu et al., 2023) proposed a fast sequential update method for HiPPO-LegS, claiming $O(N)$ computational complexity and $O(1)$ runtime complexity per update on parallel processors. However, we observe that this method becomes numerically unstable at larger N, as discussed in Appendix A.7.1. To resolve this, we suggest a simple modification: computing lower-degree representations before higher-degree ones. While the modified approach preserves the $O(N)$ overall computational cost, it increases runtime to $O(N)$ per sequential update, as it is no longer parallelizable.

Similar to HiPPO-LegS for the scaled measure, HiPPO-LegT for the translated measure also cannot be stably diagonalized. To our knowledge, LegT has no analogue for the fast LegS update method in Gu et al. (2020). Therefore, the computational complexity remains considerably higher than for diagonalizable SSMs.

6.2 Convolution kernels and diagonalization

When using an SSM for a recognition or learning task, a training phase is required in which the downstream model is trained on the generated representation. Using sequential updates for training is prohibitively taxing on computational and time resources as the whole sequence is available in the training time. Ideally, we

would perform computations of the sequential SSM in parallel. However, this is a challenge since each new update depends on the result of the previous. The authors of Gu et al. (2022b) discussed how to implement a parallel computation algorithm for SSMs produced by HiPPO. To do so, one "unrolls" the SSM as:

$$c_k = \overline{A}_k \dots \overline{A}_1 \overline{B}_0 u_0 + \dots + \overline{A}_k \overline{B}_{k-1} u_{k-1} + \overline{B}_k u_k \quad , \quad c = K * u \tag{26}$$

$$\overline{A}_i = (I + \delta t \alpha A_i)^{-1}(I - \delta t(1-\alpha)A_i), \qquad \overline{B}_i = (I + \delta t \alpha A_{i+1})^{-1} B_i. \tag{27}$$

The convolution kernel K in Eq. 26 removes the sequential dependency, enabling parallel computation on hardware such as GPUs, and significantly reducing training time.

6.3 Runtime complexity of scaled-SaFARi

For scaled-SaFARi, if the SSM is not diagonalizable, then the kernel can still be computed in parallel by framing the problem as a scan over the associative prefix product. Since matrix multiplication is associative, all such prefix products can be computed efficiently using parallel scan algorithms (Blelloch, 1990; Hillis & Jr., 1986). When implemented on parallel hardware such as GPUs, this strategy achieves a time complexity of $O(N^3 logL)$ if enough parallel processors are available.

However, if the SSM is diagonalizable, all the matrix products $\bar{A}_k$ matrices in the matrix products appearing in the kernel expression become diagonal. As a result, the convolution kernel can be calculated with the time complexity of $O(NlogL)$. Furthermore, the below theorem suggests how to find the kernel in closed form.

Theorem 6. *For scaled-SaFARi, if A is diagonalizable, then the convolution kernel K that computes the representations can be found in closed form. See the appendix for the closed form formula. See Appendix A.7.*

As noted above, HiPPO-LegS is not diagonalizable, complicating kernel computation. A heuristic method proposed by Gu et al. (2023) enables approximate kernel evaluation with time complexity $O(N \log L)$, but it introduces additional computation and lacks a closed-form solution.

6.4 Runtime complexity of translated-SaFARi

Similar to the scaled case, all the matrix products $\bar{A}_k \dots \bar{A}_{k-m}$ can be computed efficiently using parallel scan algorithms (Blelloch, 1990; Hillis & Jr., 1986). As a result, the convolution kernel K can be computed with an overall time complexity of $O(N^2 log(L))$ since $\bar{A}_k$ remains the same for all values of k. Similarly to the scaled case, if the A matrix is diagonalizable, then the $\bar{A}$ matrices become diagonal. With access to parallel processors, the runtime complexity can be reduced to $O(N^2 log(L))$.

Furthermore, for diagonalizable SSMs, the convolution kernel for the translated measure has a closed form.

Theorem 7. *For translated-SaFARi, if A is diagonalizable, then the convolution kernel K that computes the representations can be found in closed form. See Appendix A.7.*

7 Conclusions

In this work, we demonstrate how function approximation with SSMs, originally introduced by Gu et al. (2020), can be extended to general frames through SaFARi. Our method generalizes the HiPPO framework to accommodate bases or frames where closed-form solutions are unavailable, paving the way for broader applicability in sequential modeling tasks. We provide guidelines for frame selection and SSM construction in Sec. 5, highlighting the importance of both the representational power of the chosen frame and the structural properties of the A matrix in the SSM.

The versatility of SaFARi unlocks new opportunities for exploring novel SSM-based architectures in machine learning and signal processing. Future directions include the evaluation of wavelet and other structured frames within SaFARi, aiming to enhance localized and sparse representations beyond the capabilities of polynomial or Fourier methods. Moreover, integrating SaFARi into advanced SSM architectures like S4 and Mamba offers promising avenues for improving long-range sequence modeling. By extending SSM construction beyond specific bases, SaFARi provides a flexible foundation for efficient state-space representations, linking theoretical advancements to practical applications in sequential data processing.

References

Agostino Abbate, Casimer DeCusatis, and Pankaj K Das. *Wavelets and Subbands: Fundamentals and Applications*. Springer, 2012.

Sina Alemohammad, Hossein Babaei, Randall Balestriero, Matt Y. Cheung, Ahmed Imtiaz Humayun, Daniel LeJeune, Naiming Liu, Lorenzo Luzi, Jasper Tan, Zichao Wang, and Richard G. Baraniuk. Wearing a mask: Compressed representations of variable-length sequences using recurrent neural tangent kernels. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2950–2954, 2021.

Guy E. Blelloch. Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University, 1990.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 2015.

John Charles Butcher. *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods*. Wiley-Interscience, 1987.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

O Christensen. *An Introduction to Frames and Riesz Bases*. Birkhauser, 2003.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005.

Karlheinz Gröchenig. *Foundations of Time-Frequency Analysis*. Springer, 2001.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. HiPPO: Recurrent memory with optimal polynomial projections. In *Advances in Neural Information Processing Systems*, 2020.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. In *Advances in Neural Information Processing Systems*, 2022a.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022b.

Albert Gu, Isys Johnson, Aman Timalsina, Atri Rudra, and Christopher Re. How to train your HiPPO: State space models with generalized orthogonal basis projections. In *International Conference on Learning Representations*, 2023.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. In *Advances in Neural Information Processing Systems*, 2024.

Ramin Hasani, Mathias Lechner, Tsun-Hsuan Wang, Makram Chahine, Alexander Amini, and Daniela Rus. Liquid structural state-space models. In *International Conference on Learning Representations*, 2023.

W. Daniel Hillis and Guy L. Steele Jr. Data parallel algorithms. *Communications of the ACM*, 29(12): 1170–1183, 1986.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Rudolf E. Kálmán. A new approach to linear filtering and prediction problems. *J. Basic Eng.*, pp. 35–46, March 1960.

Stéphane Mallat. *A Wavelet Tour of Signal Processing: The Sparse Way.* Academic Press, 3rd edition, 2008.

Eric Nguyen, Karan Goel, Albert Gu, Gordon Downs, Preey Shah, Tri Dao, Stephen Baccus, and Christopher Ré. S4ND: Modeling images and videos as multidimensional signals with state spaces. In *Advances in Neural Information Processing Systems*, 2022.

Alan V Oppenheim. *Discrete-Time Signal Processing.* Pearson, 1999.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pp. 1310–1318, 17–19 Jun 2013.

Paolo Prandoni and Martin Vetterli. *Signal Processing for Communications.* EPFL Press, 2008.

John G Proakis. *Digital Signal Processing: Principles, Algorithms, and Applications.* Pearson, 2001.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *International Conference on Learning Representations*, 2023.

Guofeng Zhang, Tongwen Chen, and Xiang Chen. Performance recovery in digital implementation of analogue systems. *SIAM Journal on Control and Optimization*, 45(6):2207–2223, 2007.

Nikola Zubić, Mathias Gehrig, and Davide Scaramuzza. State space models for event cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

A Appendix

A.1 Derivation of SaFARi with the scaled measure

Theorem (1)**.** *Assuming elements in the frame Φ and the input signal u are right-continuous, for the representation defined in Eq. 14, the partial derivative is:*

$$\frac{\partial}{\partial T}\vec{c}(T) = -\frac{1}{T}A\vec{c}(T) + \frac{1}{T}Bu(T) \tag{A.1}$$

where A is a linear operator defined as:

$$A = I + \mathbb{U}_{\Upsilon}\mathbb{U}^{*}_{\widetilde{\Phi}} \tag{A.2}$$

and B is the complex conjugate of a vector containing members of the main frame evaluated at $T = 1$ so we have $B = \{\overline{\phi}_n(T = 1)\}_{n\in\Gamma}$. One can show that the A operator can also be described using the matrix multiplication:

$$A_{i,j} = \delta_{i,j} + \int_0^1 \overline{v}_i(t)\widetilde{\phi}_j(t)dt \tag{A.3}$$

Proof: Taking partial derivative with respect to T, we have:

$$\frac{\partial}{\partial T}c_n(T) = \frac{\partial}{\partial T}\int_{t=0}^{T}\frac{1}{T}u(t)\overline{\phi_n}\left(\frac{t}{T}\right)dt \tag{A.4}$$

Now applying the Leibniz integral rule we have:

$$\begin{aligned}\frac{\partial}{\partial T}c_n(T) &= \int_{t=0}^{T}\frac{-1}{T^2}u(t)\overline{\phi_n}\left(\frac{t}{T}\right)dt + \int_{t=0}^{T}\frac{-1}{T^2}u(t)\frac{t}{T}\overline{\phi'_n}(\frac{t}{T})dt + \frac{\overline{\phi}_n(1)}{T}u(T)\\ &= \frac{-1}{T}c_n + \frac{-1}{T}\int_{t=0}^{T}u(t)\frac{1}{T}\overline{\upsilon}_n\left(\frac{t}{T}\right)dt + \frac{\overline{\phi}_n(1)}{T}u(T)\\ &= \frac{-1}{T}c_n + \frac{-1}{T}\langle u, \upsilon_n\rangle + \frac{\overline{\phi}_n(1)}{T}u(T) \end{aligned} \tag{A.5}$$

This is still not an SSM since the second term is not explicitly a linear form of $\vec{c}(T)$. To convert this to a linear form of $\vec{c}(T)$, we use the equality given in Eq. 11 to represent υ_n using the frame $\widetilde{\Phi}$

$$\upsilon_n(t) = \sum_{j\in\Gamma}\langle \upsilon_n, \phi_j\rangle\widetilde{\phi}_j = \sum_{j\in\Gamma}\langle \upsilon_n, \widetilde{\phi}_j\rangle\phi_j \tag{A.6}$$

$$\rightarrow \langle u, \upsilon_n\rangle = \langle u, \sum_{j\in\Gamma}\langle \upsilon_n, \widetilde{\phi}_j\rangle\phi_j\rangle = \sum_{j\in\Gamma}\overline{\langle \upsilon_n, \widetilde{\phi}_j\rangle}\langle u, \phi_j\rangle = \sum_{j\in\Gamma}\overline{\langle \upsilon_n, \widetilde{\phi}_j\rangle}\, c_j(T) \tag{A.7}$$

Putting Eq. A.7 in Eq. A.5 results in:

$$\frac{\partial}{\partial T}\vec{c}(T) = -\frac{1}{T}(I + \mathbb{U}_{\Upsilon}\mathbb{U}^*_{\widetilde{\Phi}})\vec{c}(T) + \frac{\overline{\phi}_n(1)}{T}u(T) \tag{A.8}$$

This proves the theorem. □

A.2 Derivation of SaFARi with the translated measure

Theorem (2)**.** *Assuming elements in the frame Φ and the input signal u are right-continuous, for the representation defined in Eq. 14, the partial derivative is:*

$$\frac{\partial}{\partial T}\vec{c}(T) = -\frac{1}{\theta}A\vec{c}(T) + \frac{1}{\theta}Bu(T) \tag{A.9}$$

where A is a linear operator defined as:

$$\overline{A} = \mathbb{U}_{\dot{\Phi}}\mathbb{U}^*_{\widetilde{\Phi}} + \mathbb{Q}_{\Phi}\mathbb{Q}^*_{\widetilde{\Phi}} \tag{A.10}$$

and B is the complex conjugate of a vector containing members of the main frame evaluated at $T = 1$ so we have $B = \{\overline{\phi}_n(T = 1)\}_{n\in\Gamma}$. One can show that the A operator can also be described using the matrix multiplication:

$$A_{i,j} = \overline{\phi}_i(0)\widetilde{\phi}_j(0) + \int_0^1\left[\frac{\partial}{\partial t}\overline{\phi}_i(t)\right]_{t=t'}\widetilde{\phi}_j(t')dt' \tag{A.11}$$

Proof: We can write the partial derivative as:

$$c_n(T) = \int_{T-\theta}^{T}u(t)\frac{1}{\theta}\overline{\phi}_n\left(\frac{t-(T-\theta)}{\theta}\right)dt \tag{A.12}$$

Taking the partial derivative with respect to T, we have

$$\frac{\partial c_n(T)}{\partial T} = \frac{-1}{\theta^2}\int_{T-\theta}^{T}u(t)\overline{\phi}'_n\left(\frac{t-(T-\theta)}{\theta}\right)dt + \frac{1}{\theta}\overline{\phi}_n(1)u(T) - \frac{1}{\theta}\overline{\phi}_n(0)u(T-\theta) \tag{A.13}$$

Similar to our approach for the previous theorem, we write $\phi'_n(z)$ as

$$\phi'_n(z) = \sum_{i\in\Gamma}\langle \phi'_n, \widetilde{\phi}_i\rangle\phi_i(z) = \sum_{i\in\Gamma}Q_{n,i}\,\phi_i(z) \tag{A.14}$$

$$Q_{n,i} = \int_{z=0}^{1} \phi_n'(z)\overline{\widetilde{\phi}}_i(z)dz \tag{A.15}$$

Now, if we use this expansion, and put it in Eq. A.13 we have

$$\frac{\partial c_n(T)}{\partial T} = \frac{-1}{\theta^2}\int_{T-\theta}^{T} u(t)\left[\sum_{i\in\Gamma}\overline{Q}_{n,i}\,\overline{\phi}_i\left(\frac{t-(T-\theta)}{\theta}\right)\right]dt + \frac{1}{\theta}\overline{\phi}_n(1)u(T) - \frac{1}{\theta}\overline{\phi}_n(0)u(T-\theta) \tag{A.16}$$

$$\frac{\partial c_n(T)}{\partial T} = \frac{-1}{\theta}\sum_{i\in\Gamma}\overline{Q}_{n,i}\left[\frac{-1}{\theta}\int_{T-\theta}^{T} u(t)\overline{\phi}_i\left(\frac{t-(T-\theta)}{\theta}\right)dt\right] + \frac{1}{\theta}\overline{\phi}_n(1)u(T) - \frac{1}{\theta}\overline{\phi}_n(0)u(T-\theta) \tag{A.17}$$

We also write $u(T-\theta)$ as a reconstruction using the current representation $u(T-\theta) = \sum_i c_i\widetilde{\phi}_i(0)$

$$\frac{\partial c_n(T)}{\partial T} = \frac{-1}{\theta}\sum_i \overline{Q}_{n,i}\,c_i + \frac{1}{\theta}\overline{\phi}_n(1)u(T) - \frac{1}{\theta}\overline{\phi}_n(0)\sum_i c_i\widetilde{\phi}_i(0) \tag{A.18}$$

$$\frac{\partial c_n(T)}{\partial T} = \frac{-1}{\theta}\sum_i (\overline{Q}_{n,i} + \widetilde{\phi}_i(0)\overline{\phi}_n(0))\,c_i + \frac{1}{\theta}\overline{\phi}_n(1)u(T) \tag{A.19}$$

If we put $A_{i,j} = \overline{Q}_{i,j} + \overline{\phi}_i(0)\widetilde{\phi}_j(0)$, it proves the theorem. □

A.3 Comparison of SaFARi and HiPPO A matrices

Our results with SaFARi converge to HiPPO for LegS, and for the version of LegT given in Gu et al. (2023). We noted that there were some discrepancies with published versions of FouT. In Gu et al. (2023), a complete closed-form solution is given as follows (flipping negative signs to be consistent with our notation):

$$A_{n,k} = \begin{cases} 2 & n=k=0 \\ 2\sqrt{2} & n=0, k \text{ odd} \\ 2\sqrt{2} & k=0, n \text{ odd} \\ 4 & n,k \text{ odd} \\ -2\pi k & n-k=1, k \text{ odd} \\ 2\pi n & k-n=1, n \text{ odd} \\ 0 & \text{otherwise} \end{cases} \tag{A.20}$$

Our derivation for a closed-form solution is different by a factor of 2, and has different values on the off-diagonals:

$$A_{n,k} = \begin{cases} 1 & n=k=0 \\ \sqrt{2} & n=0, k \text{ odd} \\ \sqrt{2} & k=0, n \text{ odd} \\ 2 & n,k \text{ odd} \\ \pi(k+1) & n-k=1, k \text{ odd} \\ -\pi(n+1) & k-n=1, n \text{ odd} \\ 0 & \text{otherwise} \end{cases} \tag{A.21}$$

The discrepancy may be due to Gu et al. using a Fourier basis other than the DFT over the range $[0,1]$ normalized to unit energy.

A.4 Mathematical properties of SaFARi

Proposition. *For any scalar $\beta > 0$, if $h(t) = u(\beta t)$ then for the scaled measure we have* $\mathrm{SaFARi(h)(T)} = \mathrm{SaFARi(u)}(\beta\mathrm{T})$

Proof: We start by writing the representation generated by the scaled SaFARi for $h(t)$.

$$\mathrm{SaFARi(h)(T)} = \int_{t=0}^{T} h(t)\overline{\phi_n}\left(\frac{t}{T}\right)\frac{1}{T}dt \tag{A.22}$$

$$= \int_{t=0}^{T} u(\beta t)\overline{\phi_n}\left(\frac{t}{T}\right)\frac{1}{T}dt \tag{A.23}$$

$$\int_{t=0}^{T} u(t')\overline{\phi_n}\left(\frac{t'}{\beta T}\right)\frac{1}{T}\frac{dt'}{\beta} = \mathrm{SaFARi(u)}(\beta\mathrm{T}) \tag{A.24}$$

Proposition. *For any scalar $\beta > 0$, if $h(t) = u(\beta t)$ then for the translated measure with parameter θ we have* $\mathrm{SaFARi}_\theta\mathrm{(h)(T)} = \mathrm{SaFARi}_{\beta\theta}\mathrm{(u)}(\beta\mathrm{T})$

Similar to the previous proposition, we start by writing the representation of the scaled SaFARi:

$$\mathrm{SaFARi}_\theta\mathrm{(h)(T)} = \int_{t=T-\theta}^{T} h(t)\overline{\phi_n}\left(\frac{t-(T-\theta)}{\theta}\right)\frac{1}{\theta}dt \tag{A.25}$$

$$= \int_{t=T-\theta}^{T} u(\beta t)\overline{\phi_n}\left(\frac{t-(T-\theta)}{\theta}\right)\frac{1}{\theta}dt \tag{A.26}$$

$$= \int_{t'=\beta T-\beta\theta}^{\beta T} u(t')\overline{\phi_n}\left(\frac{t'-(\beta T-\beta\theta)}{\beta\theta}\right)\frac{1}{\theta}\frac{dt'}{\beta} = \mathrm{SaFARi}_{\beta\theta}\mathrm{(u)}(\beta\mathrm{T}) \tag{A.27}$$

A.5 The closed-form solution for SaFARi differential equations

Lemma 1. *The closed form solution for the differential equation introduced in Eq. 15 is:*

$$c = \int_{t=0}^{T} \frac{1}{t}\exp\left(A\ln\frac{t}{T}\right)Bu(t)dt. \tag{A.28}$$

Proof: We begin by re-writing the differential equation for any time t

$$\frac{\partial}{\partial t}\vec{c}(t) + \frac{1}{t}A\vec{c}(t) = \frac{1}{t}Bu(t) \tag{A.29}$$

Now we multiply both sides by $\exp(A\ln(t))$

$$\exp(A\ln(t))\frac{\partial}{\partial t}\vec{c}(t) + \frac{1}{t}A\exp\left(A\ln(t)\right)\vec{c}(t) = \frac{1}{t}\exp(A\ln(t))Bu(t) \tag{A.30}$$

The left side of the equality is now a complete differential

$$\frac{\partial}{\partial t}\left(\exp(A\ln(t))\vec{c}(t)\right) = \frac{1}{t}\exp\left(A\ln(t)\right)Bu(t) \tag{A.31}$$

$$\exp\left(A\ln(T)\right)\vec{c}(T) = \int_{t=0}^{T}\frac{1}{t}\exp\left(A\ln(t)\right)Bu(t) \tag{A.32}$$

$$\vec{c}(T) = \exp\left(-A\ln(T)\right)\int_{t=0}^{T}\frac{1}{t}\exp\left(A\ln(t)\right)Bu(t) \tag{A.33}$$

This proves Lemma 1. □

Lemma 2. *The closed form solution for the differential equation introduced in Eq. 19 is:*

$$c = \int_{t=T-\theta}^{T} \frac{1}{\theta} \exp\left(A\frac{t-T}{\theta}\right) Bu(t). \tag{A.34}$$

Proof: We begin by re-writing the differential equation for any time t as

$$\frac{\partial}{\partial t}\vec{c}(t) + \frac{1}{\theta}A\vec{c}(t) = \frac{1}{\theta}Bu(t). \tag{A.35}$$

Now we multiply both sides by $\exp(A\frac{t}{\theta})$

$$\exp\left(A\frac{t}{\theta}\right)\frac{\partial}{\partial t}\vec{c}(t) + \frac{1}{\theta}A\exp\left(A\frac{t}{\theta}\right)\vec{c}(t) = \frac{1}{\theta}\exp\left(A\frac{t}{\theta}\right)Bu(t) \tag{A.36}$$

The left side of the equality is now a complete differential

$$\frac{\partial}{\partial t}\left(\exp\left(A\frac{t}{\theta}\right)\vec{c}(t)\right) = \frac{1}{\theta}\exp\left(A\frac{t}{\theta}\right)Bu(t) \tag{A.37}$$

$$\vec{c}(T) = \exp\left(-A\frac{T}{\theta}\right)\int_{t=T-\theta}^{T}\frac{1}{\theta}\exp\left(A\frac{t}{\theta}\right)Bu(t) \tag{A.38}$$

This proves Lemma 2. □

A.6 Error analysis

Theorem (3)**.** *The truncated representation generated by the scaled-SaFARi follows a differential equation similar to the full representation, with the addition of a perturbation factor.*

$$\frac{\partial}{\partial T}c = -\frac{1}{T}Ac + \frac{1}{T}Bu(T) - \frac{1}{T}\vec{\epsilon}(T). \tag{A.39}$$

where $\vec{\epsilon}$ is defined as

$$\vec{\epsilon}(T) = \langle u_T, \xi\rangle \tag{A.40}$$

$$\xi = \Upsilon(\widetilde{\Phi}\Phi - I) \tag{A.41}$$

Proof: Repeat the steps taken in the proof of Theorem 1 until Eq. A.6. Truncating the frame results in an error in this step which can be written as

$$\upsilon_n(t) = \sum_{j\in\Gamma}\langle \upsilon_n, \widetilde{\phi}_j\rangle\phi_j + \xi_n(t) \tag{A.42}$$

In fact, this is how ξ is defined. Adding ξ as a correction term here changes the SSM derivation:

$$\rightarrow \langle u, \upsilon_n\rangle = \langle u, \xi_n + \sum_{j\in\Gamma}\langle \upsilon_n, \widetilde{\phi}_j\rangle\phi_j\rangle = \sum_{j\in\Gamma}\overline{\langle \upsilon_n, \widetilde{\phi}_j\rangle}\langle u, \phi_j\rangle + \langle u, \xi_n\rangle = \sum_{j\in\Gamma}\overline{\langle \upsilon_n, \widetilde{\phi}_j\rangle}\, c_j(T) + \langle u, \xi_n\rangle \tag{A.43}$$

Putting Eq. A.43 in Eq. A.5 results in:

$$\frac{\partial}{\partial T}c = -\frac{1}{T}Ac + \frac{1}{T}Bu(T) - \frac{1}{T}\vec{\epsilon}(T). \qquad \square \tag{A.44}$$

Theorem (4)**.** *If one finds an upper bound such that $\forall t < T$ we have $\|\epsilon(t)\|_2 < \epsilon_m$, then the representation error can be bound by*

$$\|\Delta c(T)\|_2 < \epsilon_m\sqrt{\sum\frac{1}{\lambda_i^2}} = \epsilon_m\|A^{-1}\|_F \tag{A.45}$$

Proof: Using the result of Theorem 3:

$$\frac{\partial}{\partial T}c = -\frac{1}{T}Ac + \frac{1}{T}\left(Bu(T) - \vec{\epsilon}(T)\right) \tag{A.46}$$

We can use Lemma 1 to find the closed form solution of the perturbed SSM above

$$c = \int_{t=0}^{T} \frac{1}{t}\exp\left(A\ln\frac{t}{T}\right)\left(Bu(t) - \vec{\epsilon}(t)\right)dt \tag{A.47}$$

$$c = \int_{t=0}^{T} \frac{1}{t}\exp\left(A\ln\frac{t}{T}\right)Bu(t)dt - \int_{t=0}^{T} \frac{1}{t}\exp\left(A\ln\frac{t}{T}\right)\vec{\epsilon}(t)dt \tag{A.48}$$

$$\text{Size N representation} = \text{True representation} - \text{Error} \tag{A.49}$$

The last term is indeed the second type error that we have discussed in the error analysis section of the paper. Using eigenvalue decomposition of $A = V\Lambda V^{-1}$ we re-write the above error term as

$$\text{Error} = \mathrm{V}\int_{\mathrm{t}=0}^{\mathrm{T}} \frac{1}{\mathrm{t}}\exp\left(\Lambda\ln\frac{\mathrm{t}}{\mathrm{T}}\right)\mathrm{V}^{-1}\tilde{\epsilon}(\mathrm{t})\mathrm{dt} \tag{A.50}$$

with a change of variable $s = \ln\frac{t}{T}$

$$V^{-1}\text{Error} = \int_{\mathrm{t}=0}^{\mathrm{T}} \exp(\Lambda\mathrm{s})\mathrm{V}^{-1}\tilde{\epsilon}(\mathrm{s})\mathrm{ds} \tag{A.51}$$

According to the assumption of this theorem $\|V^{-1}\epsilon(t)\|_2 = \|\epsilon(t)\|_2 \leq \epsilon_m$

$$\rightarrow [V^{-1}\text{Error}]_{\mathrm{j}} \leq \int_{\mathrm{t}=0}^{\mathrm{T}} \exp(\mathrm{s}\lambda_{\mathrm{j}})\epsilon_{\mathrm{m}}\mathrm{ds} = \frac{\epsilon_{\mathrm{m}}}{\lambda_{\mathrm{j}}} \tag{A.52}$$

$$\|Error\|^2 = \|V^{-1}Error\|^2 \leq \epsilon_m\sqrt{\sum\frac{1}{\lambda_i^2}} = \epsilon_m\|A^{-1}\|_F \tag{A.53}$$

Theorem (5)**.** *Suppose a frame Φ is given. The Dual of Truncation (DoT) construct introduced in Section 5.1.3 has optimal representation error when compared to any other SaFARi$^{(N)}$ construct for the same frame Φ.*

Proof: The proof for this theorem involves two steps.

1. First, we show that the optimal representation error in the theorem can be reduced to optimal error of the second type (mixing).
2. Then, we demonstrate that for a given frame Φ, and given truncation level N, the construct with the optimal second type error control is DoT.

As discussed in Sec. 5.2, the first type of error is due to truncating the frame, and is independent of the SSM. In the scope of this theorem, all the SaFARi$^{(N)}$ constructs use the same frame and the same truncation. Therefore, comparing the representation error between SaFARi$^{(N)}$ in the theorem reduces to comparing the mixing error.

The mixing error is shown in Theorem 3 to be proportional to

$$\vec{\epsilon}(T) = \langle u_T, \xi\rangle \tag{A.54}$$

To minimize $\|\vec{\epsilon}(T)\|$ irrespective of the input signal, one has to minimize $\|\xi\|_F^2$

$$\xi = \Upsilon_{[0:N]}(\widetilde{\Phi}^{(N)}\Phi_{[0:N]} - I) \tag{A.55}$$

Where $\Upsilon_{[0:N]}$ and $\Phi_{[0:N]}$ are the first N elements of Υ and Φ. $\widetilde{\Phi}^{(N)}$ is the approximation of the dual frame that determines the SaFARi$^{(N)}$ construction. For the ease of notation, we rewrite Eq. A.55 as:

$$\xi = \Upsilon(\widetilde{\Phi}\Phi - I) \tag{A.56}$$

For a fixed Υ, and Φ, we aim to minimize $\|\xi\|_F^2$ with respect to $\widetilde{\Phi}$:

$$Argmin_{\widetilde{\Phi}} \|\Upsilon(\widetilde{\Phi}\Phi - I)\|_F^2 \tag{A.57}$$

For the optimal $\widetilde{\Phi}$, the partial derivative is zero.

$$\frac{\partial}{\partial\widetilde{\Phi}}\|\xi\|_F^2 = \frac{\partial}{\partial\widetilde{\Phi}}\|\Upsilon(\widetilde{\Phi}\Phi - I)\|_F^2 = 2\Upsilon\Upsilon^T(\widetilde{\Phi}\Phi - I)\Phi^T = 0 \tag{A.58}$$

$$\widetilde{\Phi} = (\Phi\Phi^T)^{-1}\Phi^T \tag{A.59}$$

One should note that the described $\widetilde{\Phi}$ is indeed the pseudo-inverse dual for the truncated frame $\Phi_{[0:N]}$. Therefore, among the possible SaFARi$^{(N)}$ constructs for the same frame, the Dual of Truncation (DoT) has optimal representation error. □

A.7 Parallelization using the convolution kernel

Theorem (6)**.** *For SaFARi using the scaling measure, if A is diagonalizable, computing the scaled representation on a sequence with L samples can be done using a kernel multiplication.*
a) For the discretization using General Biliniear Transform (GBT) with parameter α, the kernel can be computed using:

$$K_L[i,j] = \frac{\prod_{k=j+1}^{L}\left(1 - \frac{1-\alpha}{k+1}\lambda_i\right)}{\prod_{k=j}^{L}\left(1 + \frac{\alpha}{k+1}\lambda_i\right)} \in \mathbb{R}^{N\times L} \tag{A.60}$$

b) For long sequences, the kernel K can be approximated using

$$K_L(i,j) = \frac{1}{j}\left(\frac{j}{L}\right)^{\lambda_i} \in \mathbb{R}^{N\times L} \tag{A.61}$$

For either case a or b, the representation is computed as:

$$c = MK_L\vec{u}, \quad M = V\mathrm{diag}(\mathrm{V}^{-1}\mathrm{B}) \tag{A.62}$$

where V and λ_i are the eigenvector matrix and eigenvalues of A.

Proof: a) rewriting the GBT update rule for the diagnoalzied SSM we have:

$$\begin{aligned}\widetilde{C}[n+1] &= \left(I + \frac{\alpha}{n+1}\Lambda\right)^{-1}\left(I - \frac{1-\alpha}{n+1}\Lambda\right)\widetilde{C}[n] + \left(I + \frac{\alpha}{n+1}\Lambda\right)^{-1}\widetilde{B}u[n] \\ &= \bar{A}_n\widetilde{C}[n] + \bar{B}_n u[n]\end{aligned} \tag{A.63}$$

$$\rightarrow \widetilde{C}[L+1] = \bar{B}_L u[L] + \bar{A}_L\bar{B}_{L-1}u[L-1] + ... + \bar{A}_L\ldots\bar{A}_1\bar{B}_0 u[0] \tag{A.64}$$

for the ease of computation and notation, we define K_L such that:

$$K_L[i,j] = \bar{A}_L\ldots\bar{A}_{j+1}\left(1 + \frac{\alpha}{j+1}\lambda_i\right)^{-1} = \frac{\prod_{k=j+1}^{L}\left(1 - \frac{1-\alpha}{k+1}\lambda_i\right)}{\prod_{k=j}^{L}\left(1 + \frac{\alpha}{k+1}\lambda_i\right)} \tag{A.65}$$

$$\widetilde{C}_i = \widetilde{B}_i\sum_j K_L[i,j]u[j] = \widetilde{B}_i[K_L\vec{u}]_i \tag{A.66}$$

$$\widetilde{c} = \widetilde{B} \odot (K_L \vec{u}) = \mathrm{diag}(\widetilde{\mathrm{B}})\mathrm{K_L}\tilde{\mathrm{u}} \tag{A.67}$$

$$c = V\widetilde{c} = V\mathrm{diag}(\widetilde{\mathrm{B}})\mathrm{K_L}\tilde{\mathrm{u}} = \mathrm{MK_L}\tilde{\mathrm{u}} \tag{A.68}$$

Proof: b) Using Lemma 1 for the diagonalized version of Scaled-SaFARi the closed form solution is

$$\widetilde{c} = \int_{t=0}^{T} \frac{1}{t} \exp\left(\Lambda \ln \frac{t}{T}\right) \widetilde{B} u(t) dt. \tag{A.69}$$

$$\widetilde{c}_i = \int_{t=0}^{T} \frac{1}{t} \exp\left(\lambda_i \ln \frac{t}{T}\right) \widetilde{B}_i u(t) dt = \widetilde{B}_i \int_{t=0}^{T} \frac{1}{t} \left(\frac{t}{T}\right)^{\lambda_i} u(t) dt = \widetilde{B}_i [K_L \vec{u}]_i \tag{A.70}$$

$$\widetilde{c} = \widetilde{B} \odot (K_L \vec{u}) = \mathrm{diag}(\widetilde{\mathrm{B}})\mathrm{K_L}\tilde{\mathrm{u}} \tag{A.71}$$

$$c = V\widetilde{c} = V\mathrm{diag}(\widetilde{\mathrm{B}})\mathrm{K_L}\tilde{\mathrm{u}} = \mathrm{MK_L}\tilde{\mathrm{u}} \tag{A.72}$$

Theorem (7)**.** *For SaFARi using translated measure with θ_L samples long sliding window,if A is diagonalizable, computing the translated representation on a sequence with L samples can be done using a kernel multiplication.*

a) for the discretization using General Bilinear Transform(GBT) with parameter α, the kernel can be computed using:

$$K_L[i,j] = \frac{1}{1 + \frac{\alpha}{\theta}\lambda_i} \left(\frac{1 - \frac{1-\alpha}{\theta}\lambda_i}{1 + \frac{\alpha}{\theta}\lambda_i}\right)^{L-j} \tag{A.73}$$

b) For long sequences, the kernel K can be approximated using

$$K_L(i,j) = \frac{1}{\theta_L} \exp\left(-\lambda_i \frac{L-j}{\theta_L}\right) \in \mathbb{R}^{N \times L} \tag{A.74}$$

For either of case a or b, the representation is computed by

$$c = MK_L\vec{u}, \quad M = V\mathrm{diag}(\mathrm{V}^{-1}\mathrm{B})\,. \tag{A.75}$$

where V and λ_i are the eigenvectors matrix, and eigenvalues of A.

Proof: a) rewriting the GBT update rule for the diagonalized SSM we have:

$$\widetilde{C}[n+1] = \left(I + \frac{\alpha}{\theta}\Lambda\right)^{-1} \left(I - \frac{1-\alpha}{\theta}\Lambda\right) \widetilde{C}[n] + \left(I + \frac{\alpha}{\theta}\Lambda\right)^{-1} \widetilde{B} u[n] = \bar{A}\widetilde{C}[n] + \bar{B}u[n] \tag{A.76}$$

we take a similar approach to the previous theorem. The only difference is that $\bar{A}$ and $\bar{B}$ remain the same for all the time indices.

$$\widetilde{C}[L+1] = \bar{B}u[L] + \bar{A}\bar{B}u[L-1] + ... + \bar{A}^L\bar{B}u[0] \tag{A.77}$$

for the ease of computation and notation, we define K_L such that:

$$K_L[i,j] = \frac{1}{1 + \frac{\alpha}{\theta}\lambda_i} \left(\frac{1 - \frac{1-\alpha}{\theta}\lambda_i}{1 + \frac{\alpha}{\theta}\lambda_i}\right)^{L-j} \tag{A.78}$$

$$\widetilde{C}_i = \widetilde{B}_i \sum_j K_L[i,j]u[j] = \widetilde{B}_i [K_L\vec{u}]_i \tag{A.79}$$

$$\widetilde{c} = \widetilde{B} \odot (K_L \vec{u}) = \mathrm{diag}(\widetilde{\mathrm{B}})\mathrm{K_L}\tilde{\mathrm{u}} \tag{A.80}$$

$$c = V\widetilde{c} = V\mathrm{diag}(\widetilde{\mathrm{B}})\mathrm{K_L}\tilde{\mathrm{u}} = \mathrm{MK_L}\tilde{\mathrm{u}} \tag{A.81}$$

Proof: b) Using Lemma 2 for the diagonalized version of Translated-SaFARi, the closed-form solution is:

$$\widetilde{c} = \int_{t=T-\theta}^{T} \frac{1}{\theta} \exp\left(\Lambda \frac{t-T}{\theta}\right) \widetilde{B} u(t). \tag{A.82}$$

$$\widetilde{c}_i = \int_{t=T-\theta}^{T} \frac{1}{\theta} \exp\left(\lambda_i \frac{t-T}{\theta}\right) \widetilde{B}_i u(t) = \widetilde{B}_i [K_L \vec{u}]_i \tag{A.83}$$

$$\widetilde{c} = \widetilde{B} \odot (K_L \vec{u}) = \mathrm{diag}(\widetilde{\mathrm{B}}) \mathrm{K_L} \tilde{\mathrm{u}} \tag{A.84}$$

$$c = V\widetilde{c} = V \mathrm{diag}(\widetilde{\mathrm{B}}) \mathrm{K_L} \tilde{\mathrm{u}} = \mathrm{M} \mathrm{K_L} \tilde{\mathrm{u}} \tag{A.85}$$

A.7.1 Numerical instability of the fast sequential legS solver

As part of our experimental findings, we realized that the proposed method for sequential updates for LegS SSM(in Gu et al. (2020),Appendix E) suffers from numerical instability when working with larger SSMs.

$$x = \frac{\mathrm{cumsum}(\beta \mathrm{cumprod} \frac{1}{\alpha})}{\mathrm{cumprod} \frac{1}{\alpha}} \tag{A.86}$$

where the introduced $\alpha_k = \frac{d_k}{1+d_k}$ is a decreasing function. One can confirm that in the tth iteration, and for the kth degree Legendre polynomial

$$\alpha_k = \frac{d_k}{1+d_k} = \frac{2(t+1)-k}{2(t+1)+k+1} \tag{A.87}$$

Then the proposed solution requires finding cumulative product of $\frac{1}{\alpha_k}$ for $k \in [1, N]$ in each step.

$$\log \left| \mathrm{cumprod}_{k'} \left(\frac{1}{\alpha_{k'}} \right)(k) \right| = \sum_{k'=1}^{K} \log \left| \frac{1}{\alpha_{k'}} \right| = \sum_{k'=1}^{K} \log \left| -1 + \frac{4(t+1)+1}{2(t+1)-k'} \right| \tag{A.88}$$

$$= \sum_{k'=1}^{2(t+1)} \log \left(-1 + \frac{4(t+1)+1}{2(t+1)-k'} \right) + \sum_{k'=2(t+1)+1}^{K} \log \left(1 + \frac{4(t+1)+1}{k'-2(t+1)} \right) \tag{A.89}$$

$$= \sum_{k'=1}^{2(t+1)} \log \left(-1 + \frac{4(t+1)+1}{2(t+1)-k'} \right) + \sum_{k'=1}^{K-2(t+1)} \log \left(1 + \frac{4(t+1)+1}{k'} \right) \tag{A.90}$$

For any specific iteration (fixed t), as K grows(higher representation index), the second summation above grows to infinity. Thus, for large enough N, cumprod($\frac{1}{\alpha_k}$) diverges beyond machine precision. As a result, the proposed fast sequential legS solver proposed in (Gu et al. (2020), Appendix E) fails. Figure. 6 Shows an example where for $N = 500$, fast LegS numerically diverges for any sequence longer than 80 samples. It is crucial to note that this numerical instability is fundamental to legS, and does not depend on the input signal at all.

We also investigate the longest sequence length that the given fast legS sequential solution can handle without numerical diversion. Figure 7 shows that as N grows, then length of sequence that fast legS sequential solver can handle before becoming numerically unstable decrease to a limited length.

The stable version of solving legS would be to take the similar approach as the fast legS, but in the last step, instead of introducing the proposed α and β, we find x_1, then recursively find x_i after finding all the pervious x_is. This way, the overall computation complexity remains the same, while the run-time complexity becomes longer as one has to compute x_i after $x_0, \ldots, x_{i-1}$ are all computed.

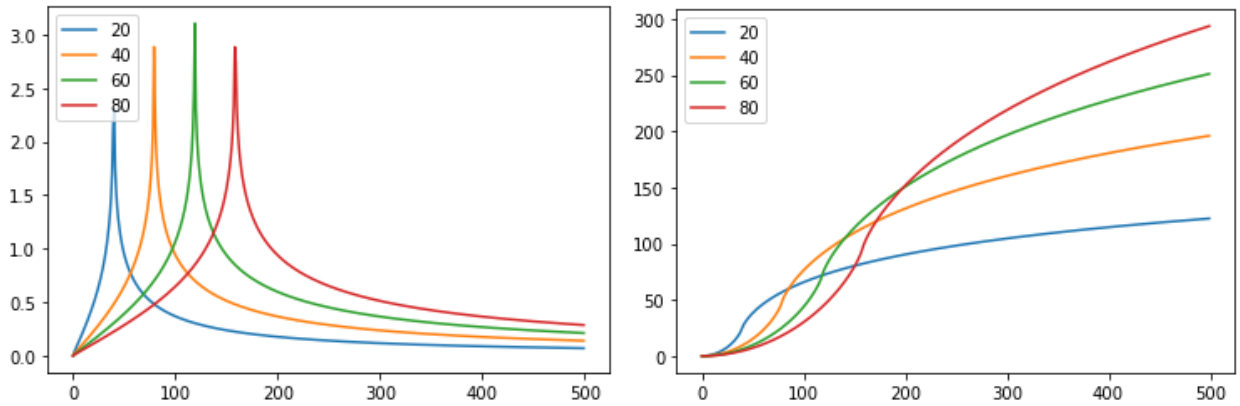

Figure 6: Fast Legs numerically diverges. **left:** For a system with $N = 500$, $\log|\frac{1}{\alpha_k}|$ for different values of $k \in [1, N]$ is plotted at different iterations $t = 20, 40, 60, 80$. **right:** log of the cumulative product which is equal to the cumulative sum of the left plot is plotted for different iterations. In the right plot, it is notable that for $t = 80$, the cumulative product reaches to 10^{300} for a $k < 500$ which is the largest value that a float-64 variable can handle. The studied Fast-LegS method for an SSM having more then 500 coefficients diverges after only 80 sequential updates.

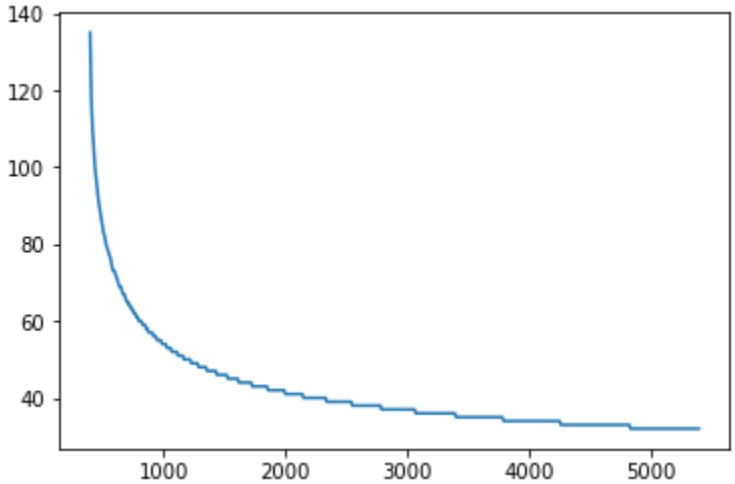

Figure 7: As the given LegS size N grows, the longest sequence length before observing numerical diversion is given in the above plot. For $N < 400$ we did not observe the numerical diversion. for $N > 400$, the fast legS sequential update method cannot handle sequences longer than a limited length before it becomes numerically unstable.