# SpectraLDS: Distilling Spectral Filters into Constant-Time Recurrent Models

Hazan Lab
Princeton University

# Motivation

**Motivation**
Spectral filtering
STU
Our work
Experiments

# Architectures

RNNs

Transformers

Spectral Filtering

Recurrent Neural Networks

**Motivation**

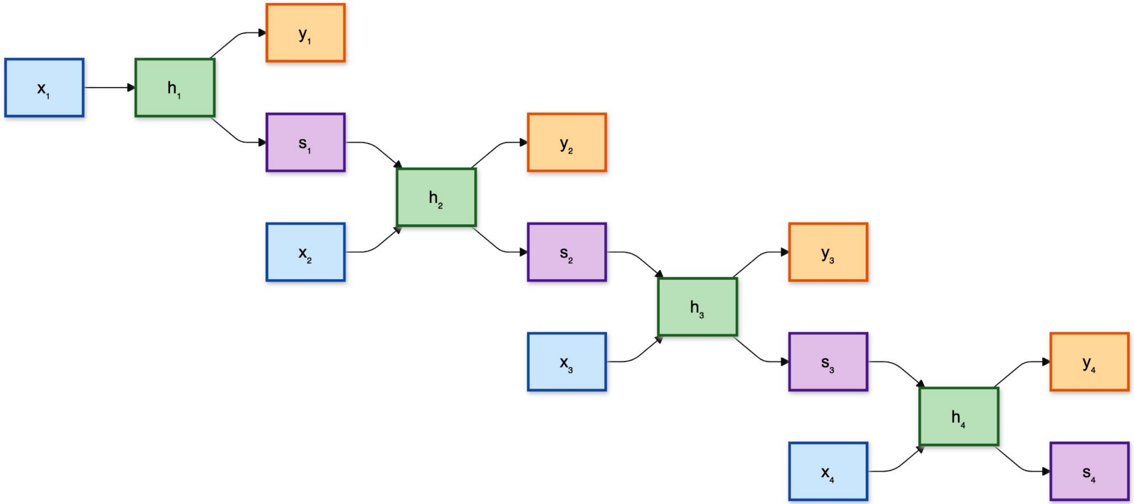
Spectral filtering

STU

Our work

Experiments

- O(L) time complexity
- But sequential
  - Need $s_{t-1}$ to compute $y_t$
- Unstable (exploding/vanishing gradients)

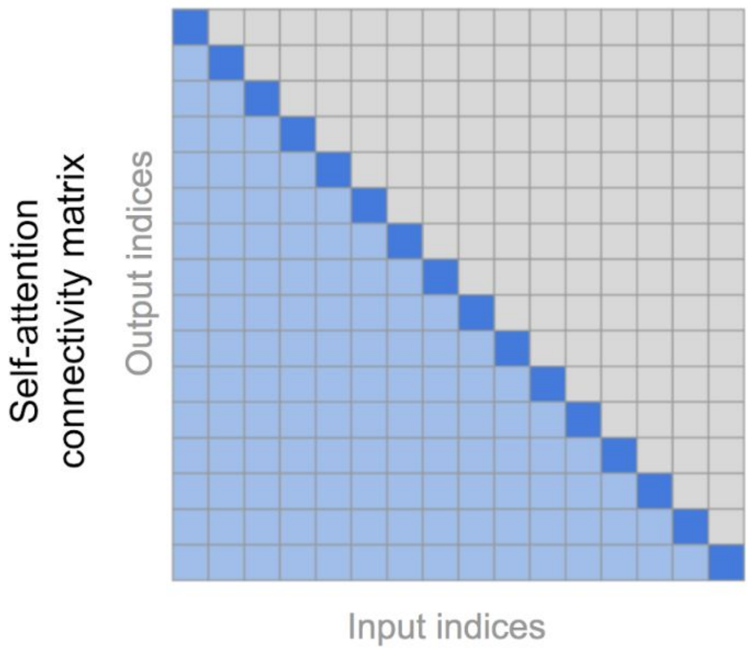


Transformers

**Motivation**
Spectral filtering
STU
Our work
Experiments

Self-attention scales as $O(L^2)$
in sequence length.

# Self-attention




**(a) Transformer**

Image source: Child et al. (2019)

Can we do better?

**Efficient** and **provable** RNNs
for long contexts?

Spectral filtering

# Linear dynamical systems

$$x_t = Ax_{t-1} + Bu_t$$
$$y_t = Cx_t + Du_t$$

Learning optimal weights for factors of *A, B, C, D* is **hard** (non-convex)

$$y_t^{\mathrm{LDS}} = (CB + D)u_t + CABu_{t-1} + CA^2Bu_{t-2} + CA^3Bu_{t-3} + \dots$$

Learning optimal weights for relaxed parameterizations is **easy** (convex)

$$\hat{y}_t = M_0 u_t + M_1 u_{t-1} + M_2 u_{t-2} + \ldots$$

# Intuition

- Consider the one-dimensional case (A = a, B,C = I, D = 0 for simplicity)

$$y_t^{\text{LDS}} = (CB + D)u_t + CABu_{t-1} + CA^2Bu_{t-2} + CA^3Bu_{t-3} + \dots$$

$$y_t^{\text{LDS}} = 1 \cdot u_t + a \cdot u_{t-1} + a^2 \cdot u_{t-2} + a^3 \cdot u_{t-3} + \dots$$

$$y_t^{\text{LDS}} = \underline{[1, a, a^2, \dots a^L]}[u_t, u_{t-1}, u_{t-2} \dots]^\top$$

- We are interested in vectors of the type $\mu(a) \triangleq \underline{[1, a, a^2, \dots a^L]} \in \mathbb{R}^L \text{ for all } a \in [0, 1]$

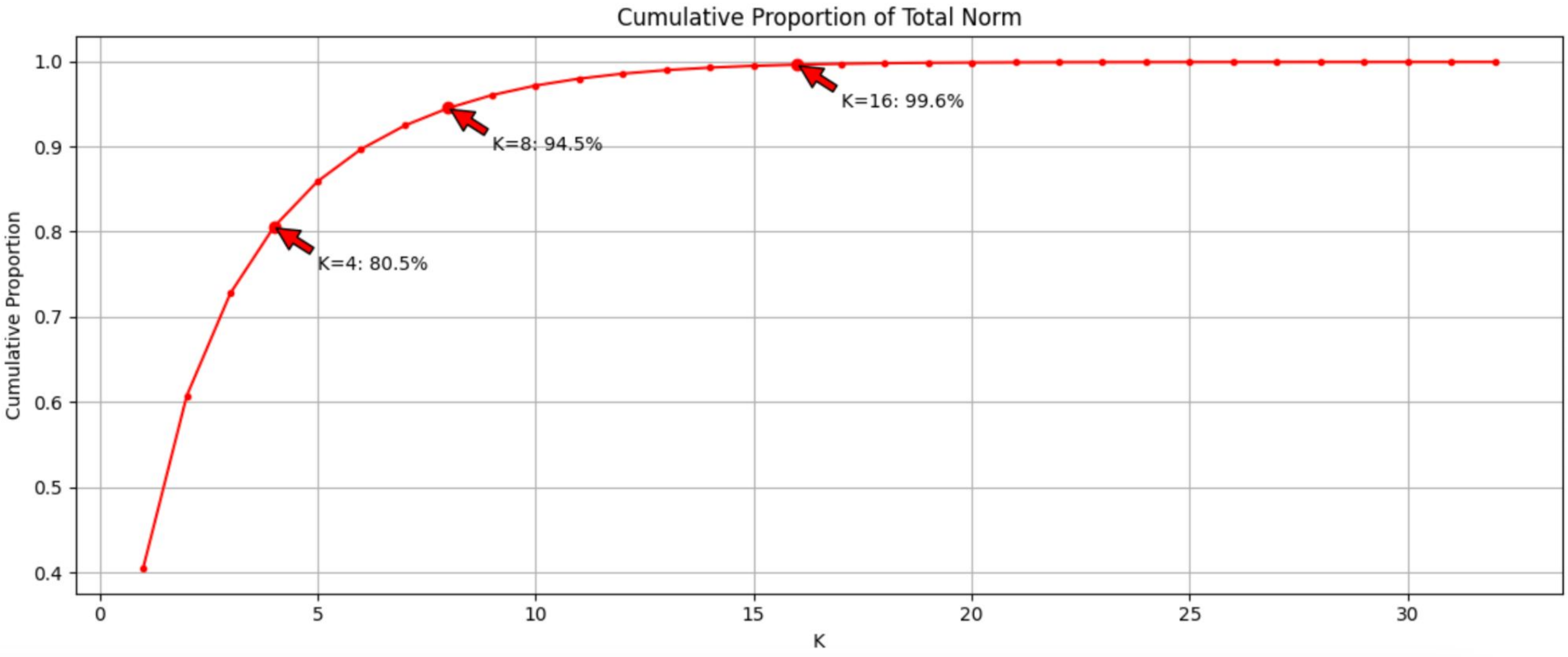
# It's a Hankel matrix!

$$\mu(a) \triangleq [1, a, a^2, \ldots a^L]$$

$$Z = \int_0^1 \mu(\alpha) \otimes \mu(\alpha)\, d\alpha$$

$$Z = \int_0^1 \begin{bmatrix} 1 & \alpha & \alpha^2 & & \\ \alpha & \alpha^2 & \alpha^3 & & \\ \alpha^2 & \alpha^3 & \alpha^4 & & \\ & & & \ddots & \\ & & & & \alpha^{2T-2} \end{bmatrix} d\alpha$$

Eigenvalues of Hankel matrices decay **exponentially.**



Cumulative Proportion of Total Norm
Cumulative Proportion
K
K=4: 80.5%
K=8: 94.5%
K=16: 99.6%
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
5
10
15
20
25
30

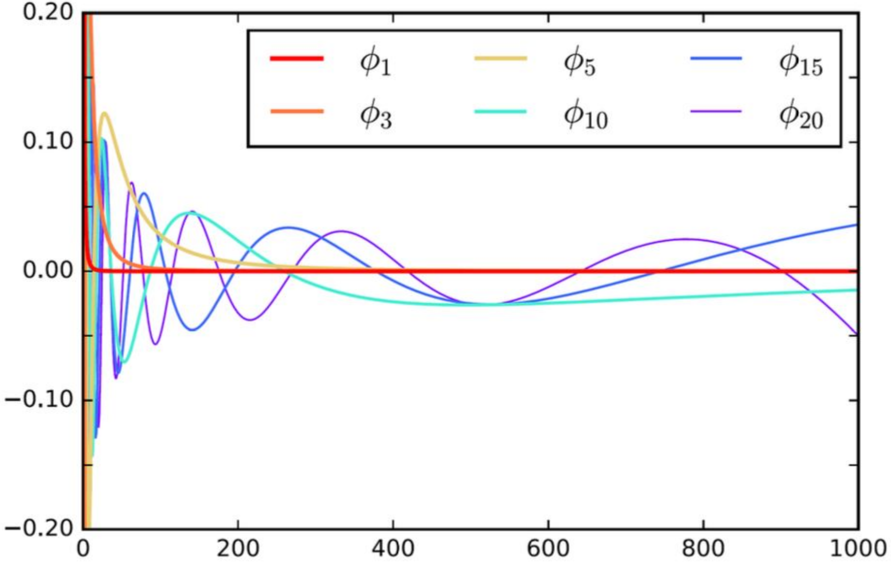



Figure 1: The filters obtained by the eigenvectors of $Z$.

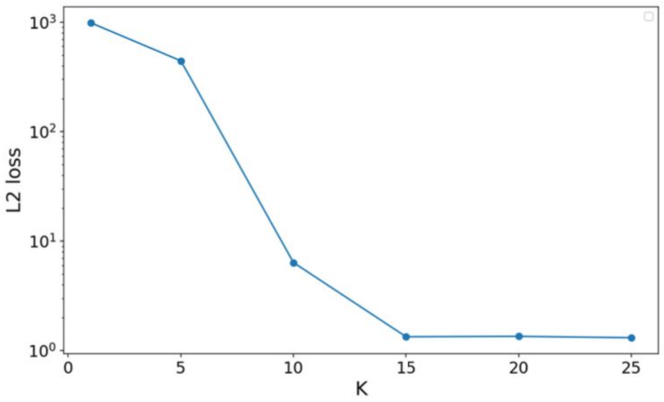


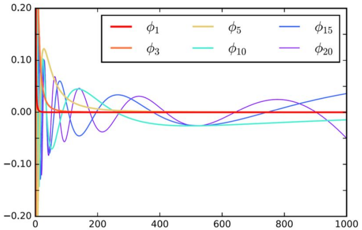

Figure 4: Error obtained by an STU layer as a function of the model parameter K. We observe an exponential drop in the reconstruction loss as predicted by the analysis.

*†Figures from the Spectral SSM paper (arXiv: 2312.06837)*

# Featurization.

- Set $\Phi_1 \ldots \Phi_K$ to be the *top-k* eigenvectors of the system matrix Z
- Featurization is the **convolution** of input sequence $u_1 \ldots u_L$ with filters $\Phi_1 \ldots \Phi_K$
- These filters are universal – they work for **any sequence prediction task!**
- Convolutions using FFT run in O(L log L) time
    - Featurizing with *k* filters runs in **O(k·L log L) time**
    - Suffices that k ~ O(log L), so we get roughly $O(L \log^2 L)$ time
- Convolutions = **GPU-friendly!**
    - NOT sequential, unlike RNNs
    - ==> Asymptotic analysis is meaningful
- Theoretical guarantees
    - **Sublinear regret** on the order of $\sqrt{L}$ (near-optimal!)

$$\hat{y}_t = M_0 u_t + M_1 u_{t-1} + M_2 u_{t-2} + \dots$$




$$y_t^{\text{LDS}} \sim \sum_{i=1}^{K} \underbrace{\langle \mu(\alpha), \Phi_i \rangle}_{\substack{\text{Learned params } M, \\ \text{scales as } e^{-i/\log(L)} \\ \text{so } k \sim \log(L)}} \cdot \underbrace{\langle \Phi_i, [u_t, u_{t-1}, u_{t-2} \dots] \rangle}_{\text{Fixed featurization}}$$

*$\mu(\alpha)$ for any $\alpha$ has all but $\epsilon$ mass concentrated on the top eigenvectors of Z.

$$\boxed{y_t^{\text{LDS}} \sim \sum_{i=1}^{K} M_i \cdot \underbrace{\langle \Phi_i, [u_t, u_{t-1}, u_{t-2} \dots] \rangle}_{\text{Fixed Featurization}}}$$

Spectral Transform Unit

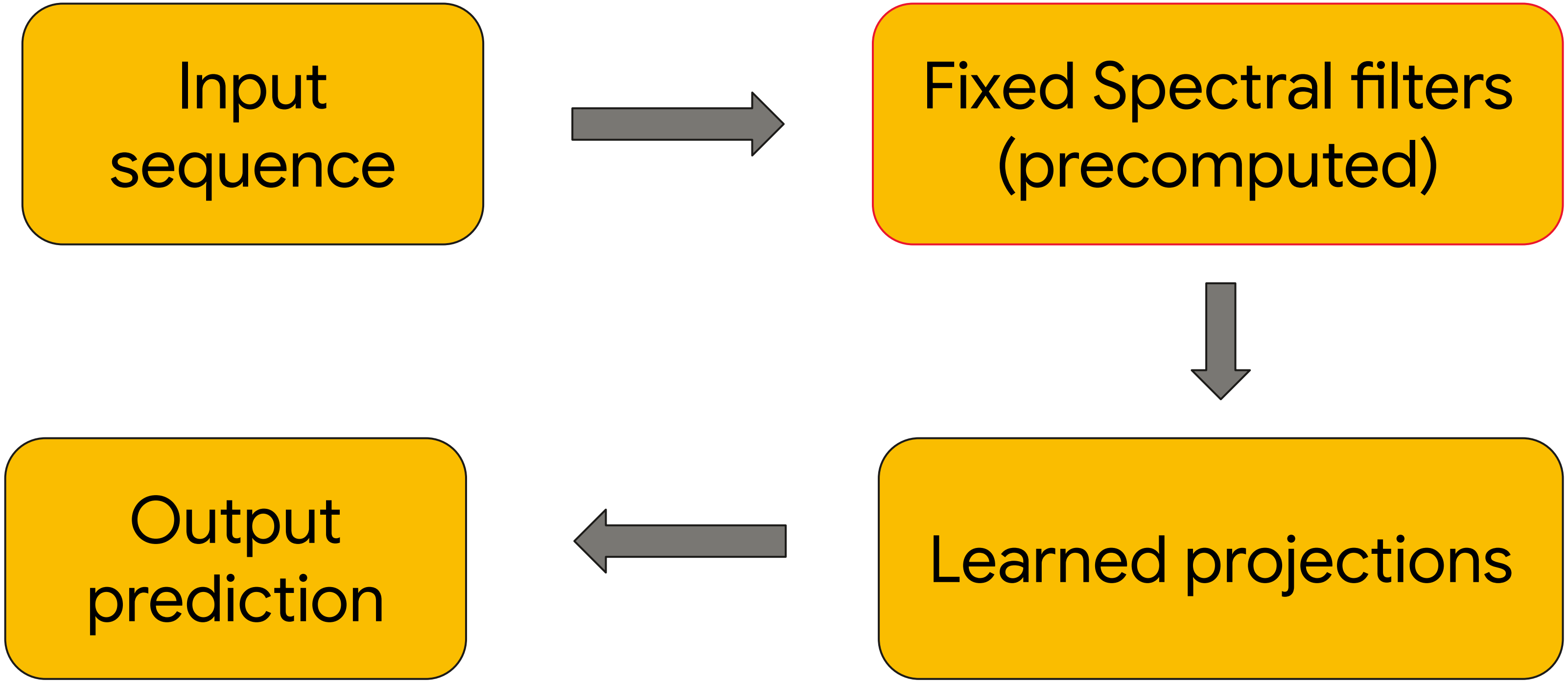
Motivation

Spectral filtering

STU

Our work

Experiments


Input sequence
Fixed Spectral filters (precomputed)
Learned projections
Output prediction

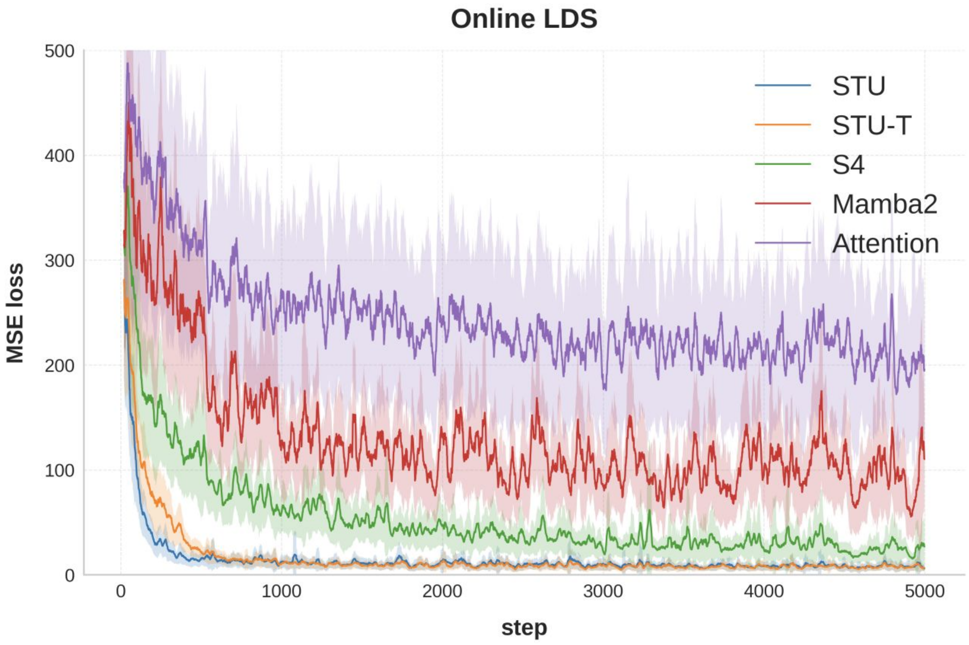



Figure 2: Mean squared error $\|\hat{y}_{t+1} - y_{t+1}\|^2$ of the different layers on a single sequence from an LDS.

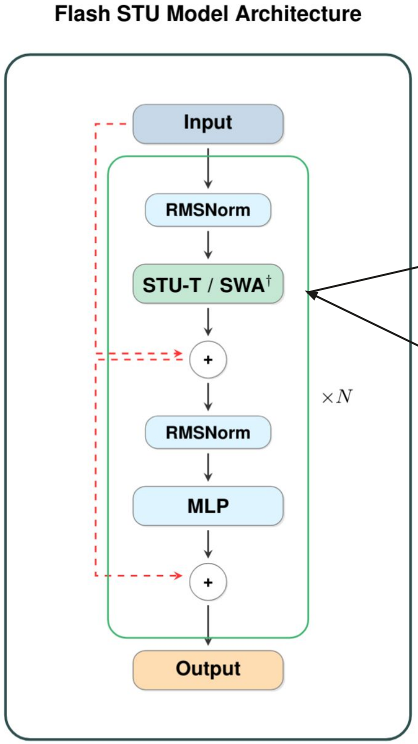
| Model | MMLU (acc ↑) | Hella. (acc_n ↑) | PIQA (acc_n ↑) | BoolQ (acc ↑) | Wino. (acc ↑) | CSQA (acc ↑) | OBQA (acc_n ↑) | ARC-e (acc ↑) | ARC-c (acc_n ↑) | Average (↑) |
|---|---|---|---|---|---|---|---|---|---|---|
| Flash STU 550M | **26.31** | **28.64** | **60.94** | **52.23** | **50.67** | 19.66 | 26.00 | **45.16** | **23.63** | **37.03** |
| Mamba-2 Hybrid 546M | 25.82 | 28.51 | 57.62 | 51.87 | 49.09 | 18.67 | **27.60** | 44.36 | 22.53 | 36.23 |
| Mamba-2 561M | 24.15 | 26.83 | 57.62 | 46.54 | 50.12 | **20.23** | 26.60 | 44.95 | 23.38 | 35.60 |
| Transformer 564M | 25.16 | 26.85 | 56.53 | 51.41 | 50.51 | 19.82 | 25.00 | 38.64 | 21.25 | 35.02 |

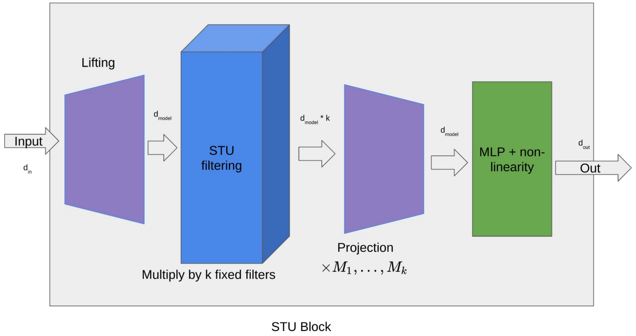
| **Transformers** | **RNNs / SSMs** | **STU** |
|---|---|---|
| $O(L^2)$ complexity | $O(L)$ complexity | $O(L \log L)$ complexity |
| Powerful but slow | Fast but unstable | Fast AND stable |
| Memory hungry | Training tricks needed | Simple to train |

# Our work

# What if we could distill the STU back to an LDS representation?

# Distilling STU layers into LDS layers




$$x_t = Ax_{t-1} + Bu_t$$
$$y_t = Cx_t + Du_t$$

Figure 12: Flash STU Model Architecture, alternating between STU-T and (sliding window) attention†.

# Fast Inference

Generating 1 token:

- **Transformers - O(prompt length + generation length)** - memory & compute
- **RNNs/SSMs - O(1) -** memory & compute
- **Flash STU - O(k log L) ≈ O($\log^2$ L)** - compute
- **Distilled STU - O(1)** - memory & compute

- **After training a Flash STU (Language) model, we distill the STU layers into LDS layers**
  - **O(1) token generation**
  - **Allows for (very) fast language models**

# How do we do distill the STU back to an LDS?

By the spectral filtering theorem, where *M* is the spectral coefficients matrix, $\Phi$ are the k spectral filters, $\Psi$ are L LDS filters of form $(1\ \alpha\ \alpha^2 \ldots \alpha^{L-1})$, and $\Gamma$ are the scalars provided by the b and c matrices, the following holds:

$$\| M\,\Phi_{1:k} \;-\; \Gamma\,\Psi_L(\alpha_1, \ldots, \alpha_k)\| \;\leq\; c\,k\,L\,e^{-\frac{k}{\log L}}$$

# How do we do distill the STU back to an LDS?

By the spectral filtering theorem, where *M* is the spectral coefficients matrix, Φ are the k spectral filters, Ψ are L LDS filters of form $(1\ \alpha\ \alpha^2 \ldots \alpha^{L-1})$, and Γ are the scalars provided by the b and c matrices, the following holds:

$$\| M \Phi_{1:k} - \Gamma \Psi_L(\alpha_1, \ldots, \alpha_k) \| \leq c k L e^{-\frac{k}{\log L}}$$

We first prove that a matrix exists that transforms the spectral filters back into an LDS:

**Theorem 1.** *For any* $\alpha_1, \ldots, \alpha_k$ *such that* $\Psi_L(\alpha_i)$ *are linearly independent, there exists* $\widetilde{M} \in \mathbb{R}^{k \times k}$ *such that*

$$\left\| \Phi_{1:k} - \widetilde{M} \Psi_L(\alpha_1, \ldots, \alpha_k) \right\| \leq c k L e^{-\frac{k}{\log L}}.$$

# How do we do distill the STU back to an LDS?

**Theorem 1.** *For any $\alpha_1, \ldots, \alpha_k$ such that $\Psi_L(\alpha_i)$ are linearly independent, there exists $\widetilde{M} \in \mathbb{R}^{k \times k}$ such that*

$$\left\| \Phi_{1:k} - \widetilde{M}\, \Psi_L(\alpha_1, \ldots, \alpha_k) \right\| \leq c\, k\, L\, e^{-\frac{k}{\log L}}.$$

# How do we do distill the STU back to an LDS?

**Theorem 1.** *For any* $\alpha_1, \ldots, \alpha_k$ *such that* $\Psi_L(\alpha_i)$ *are linearly independent, there exists* $\widetilde{M} \in \mathbb{R}^{k \times k}$ *such that*

$$\left\| \Phi_{1:k} - \widetilde{M}\, \Psi_L(\alpha_1, \ldots, \alpha_k) \right\| \leq c\, k\, L\, e^{-\frac{k}{\log L}}.$$

We then introduce our algorithm and prove that it provides such an M.

**Theorem 2.** *For any* $\alpha_1, \ldots, \alpha_k$ *such that* $\Psi_L(\alpha_i)$ *are linearly independent, Algorithm 2 provides* $\widetilde{M} \in \mathbb{R}^{k \times k}$ *such that*

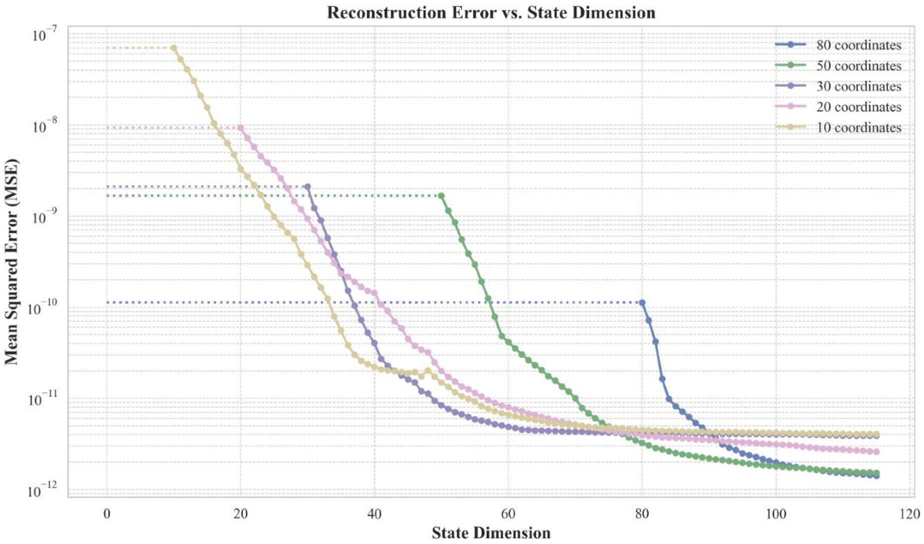
$$\left\| \Phi_{1:k} - \widetilde{M}\, \Psi_L(\alpha_1, \ldots, \alpha_k) \right\| \leq c\, k\, L\, e^{-\frac{k}{\log L}}.$$

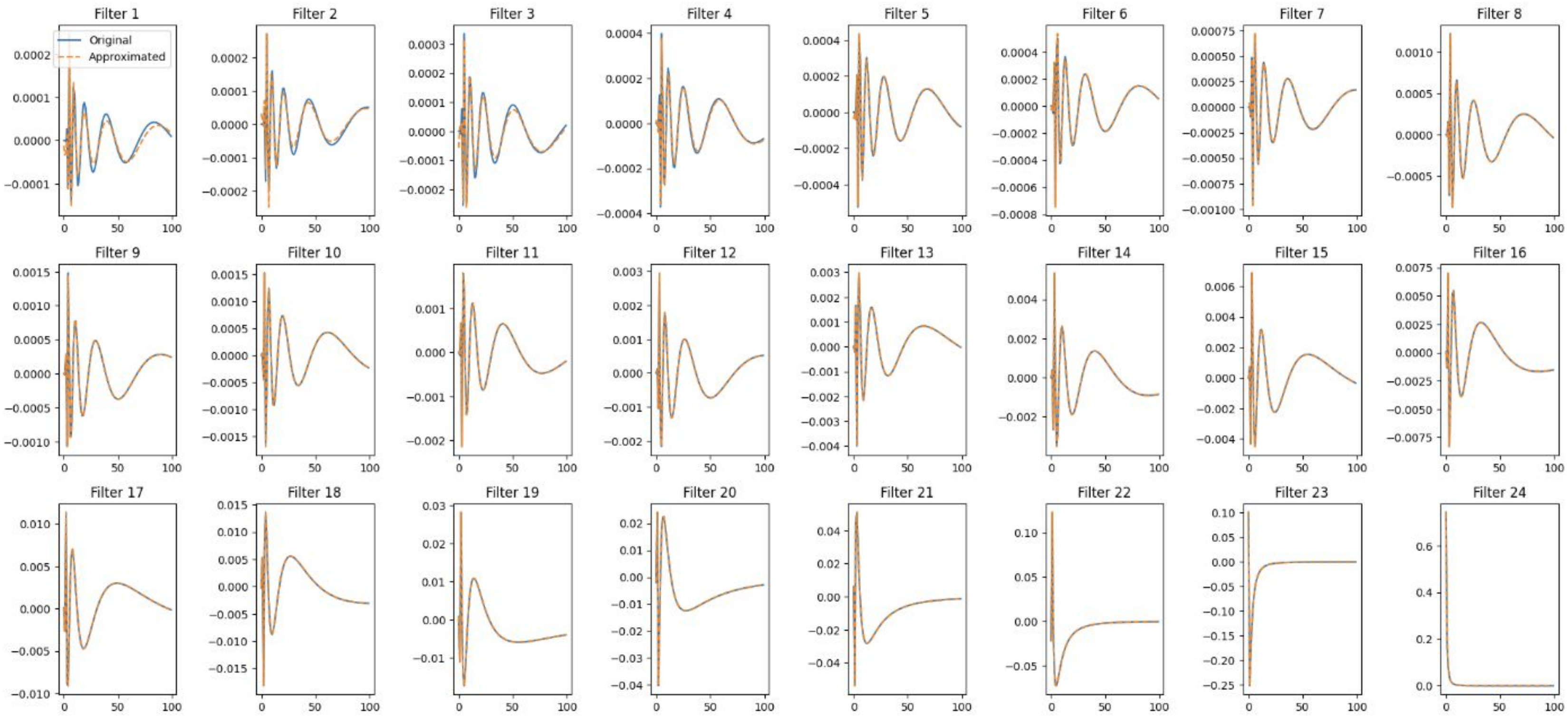
# How do we do distill the STU back to an LDS?

We provide a practical algorithm that provides such a matrix M.

**Algorithm 2** LDS Filters to Matrix Representation

**Input:** Learned STU filters $\Phi_{1:k} \in \mathbb{R}^{k \times L}$
**Output:** System matrices $\{A, B, C\}$ of a discrete-time LDS.
Initialize $\boldsymbol{\Psi} \in \mathbb{R}^{N \times L}$ and $\Theta \in \mathbb{R}^{N \times k}$ as empty matrices.
**for** $i = 1, \ldots, N$ **do**
  Sample $(a_i, b_i, c_i)$ with $|a_i| \leq 1$
  Train an STU of filter length $L$ on the 1D-LDS defined by $(a_i, b_i, c_i)$
  Let $\boldsymbol{\psi}_i \in \mathbb{R}^{1 \times L}$ be the LDS impulse response; update the $i^{\text{th}}$ row of $\boldsymbol{\Psi}$ by $\boldsymbol{\Psi}[i,:] \leftarrow \boldsymbol{\psi}_i$.
  Let $\boldsymbol{\theta}_i \in \mathbb{R}^{1 \times k}$ be the corresponding learned STU weights; update the $i^{\text{th}}$ row of $\Theta$ by $\Theta[i,$
**end for**
Pick a row-subset $\boldsymbol{\Psi}_{\text{sub}}, \Theta_{\text{sub}}$, and compute $\widetilde{M} = \Theta_{\text{sub}}^{\dagger}$ such that $\Phi_{1:k} \approx \widetilde{M}\, \boldsymbol{\Psi}_{\text{sub}}$
**repeat**
  Let $i^* \in \{1, \ldots, N\} \setminus I_{\text{sub}}$ be the index minimizing $\|\Theta[i,:]^{\dagger}\, \Phi_{1:k} - \boldsymbol{\Psi}[i,:]\|_F^2$; update $I_{\text{sub}} \leftarrow$ $\boldsymbol{\Psi}_{\text{sub}} \leftarrow \big[\boldsymbol{\Psi}_{\text{sub}};\, \boldsymbol{\Psi}[i^*,:]\big]$, and $\Theta_{\text{sub}} \leftarrow \big[\Theta_{\text{sub}};\, \Theta[i^*,:]\big]$.
  Recompute $\widetilde{M} = \Theta_{\text{sub}}^{\dagger}$ and evaluate $\|\Phi_{1:k} - \widetilde{M}\, \boldsymbol{\Psi}_{\text{sub}}\|_F$.
**until** convergence
Define loss $E(\widetilde{M}) = \|\Phi_{1:k} - \widetilde{M}\, \boldsymbol{\Psi}_{\text{sub}}\|_F^2$
**repeat**
  Compute $\nabla_{\widetilde{M}} E(\widetilde{M})$
  Update $\widetilde{M} \leftarrow \widetilde{M} - \eta\, \nabla_{\widetilde{M}} E(\widetilde{M})$
**until** convergence
Using $\widetilde{M}$, select the index set $I^*$ that minimizes $\|\Phi_{1:k} - \widetilde{M}\, \boldsymbol{\Psi}_{\text{sub}}(i)\|_F^2$ and collect $\{(a_i, b_i, c_i)$
Set $\mathbf{A} \leftarrow \operatorname{diag}(\{a_i\}_{i \in I^*})$, $\quad \mathbf{B} \leftarrow (b_i)_{i \in I^*}$, $\quad \mathbf{C} \leftarrow \widetilde{M} \operatorname{diag}(\{c_i\}_{i \in I^*})$
**return** $\{A, B, C\}$






The spectral filters successfully approximated with our LDS representation.

# From fitting the filters to fitting any STU instantly

$$\left\|\Phi_{1:k} \; - \; \widetilde{M}\,\Psi_L(\alpha_1, \ldots, \alpha_k)\right\| \; \leq \; \epsilon$$

$$\implies \; M_i \cdot \langle \Phi_i, [u_t, u_{t-1}, u_{t-2} \ldots] \rangle \approx M_i \cdot \langle \sum_{j=1}^{K} \widetilde{M}_j \Psi_j, [u_t, u_{t-1}, u_{t-2} \ldots] \rangle$$

$$\implies \; \sum_{i=1}^{K} M_i \cdot \langle \Phi_i, [u_t, u_{t-1}, u_{t-2} \ldots] \rangle \approx \sum_{i=1}^{K} M_i \cdot \langle \sum_{j=1}^{K} \widetilde{M}_j \Psi_j, [u_t, u_{t-1}, u_{t-2} \ldots] \rangle$$

$$\implies \; y_t^{STU} \approx M\widetilde{M} \left\langle \begin{bmatrix} \vec{1}, \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \end{bmatrix}, \begin{bmatrix} \alpha_1^2 \\ \alpha_2^2 \\ \vdots \end{bmatrix}, \ldots \end{bmatrix}, [u_t, u_{t-1}, u_{t-2} \ldots] \right\rangle$$

$$\implies \; y_t^{STU} \approx \sum_{i=0}^{t} C A^i B u_{t-i} = y_t^{LDS}$$

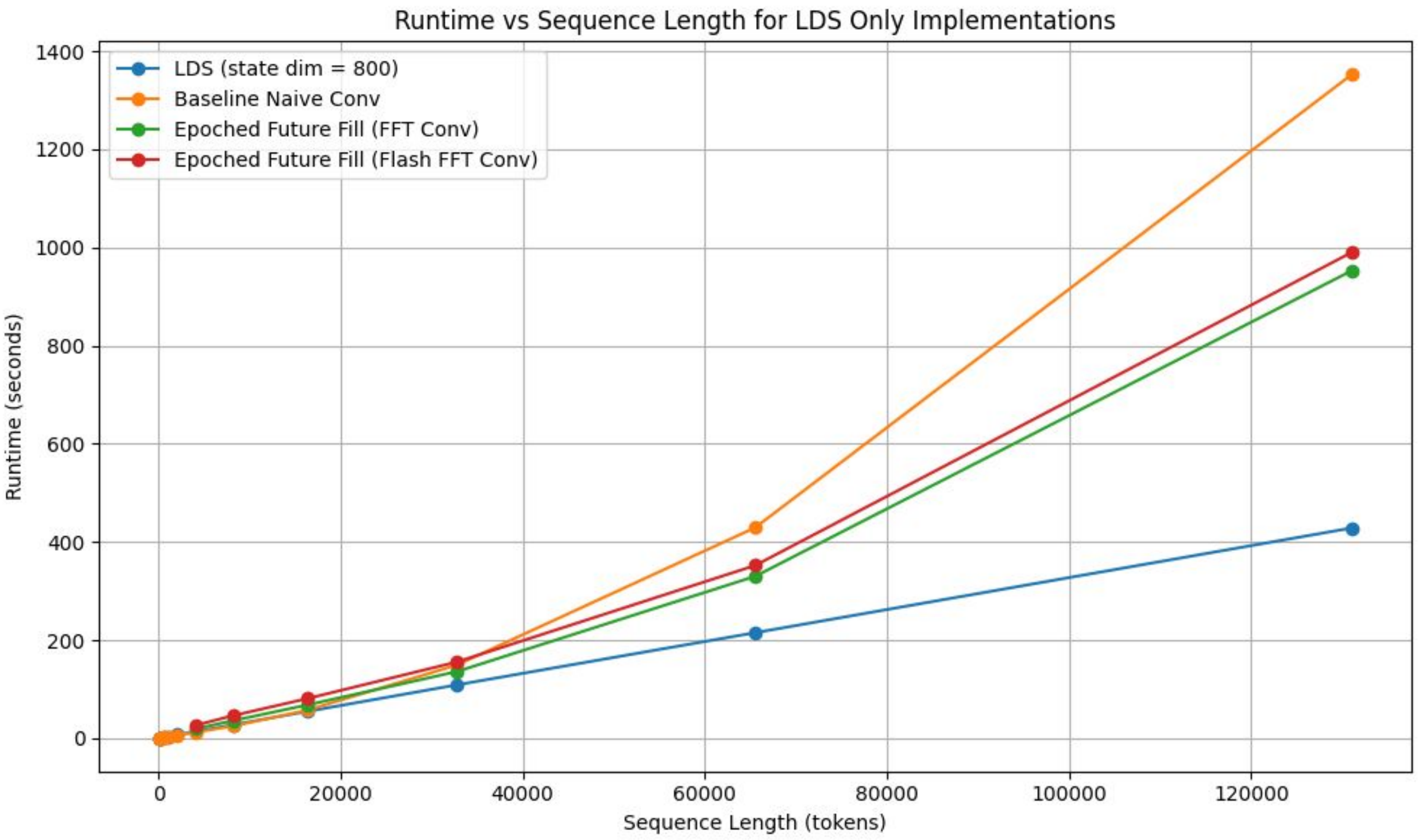
Where $A = \begin{bmatrix} \alpha_1 & 0 & \ldots \\ 0 & \alpha_2 & \\ \vdots & & \end{bmatrix}$, $B = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \end{bmatrix}$, $C = M\widetilde{M}$, and $\Psi = \begin{bmatrix} \vec{1}, \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \end{bmatrix}, \begin{bmatrix} \alpha_1^2 \\ \alpha_2^2 \\ \vdots \end{bmatrix}, \ldots \end{bmatrix}$

# Experiments

Motivation
Spectral filtering
STU
Our work
Experiments

| Model | MMLU | Hella. | PIQA | BoolQ | Wino. | CSQA | OBQA | ARC-e | ARC-c | Average |
|---|---|---|---|---|---|---|---|---|---|---|
| Flash STU 550M | 22.99 | 26.51 | 58.00 | 39.30 | 52.01 | 19.49 | 29.80 | 39.10 | 23.12 | 34.48 |
| LDS | 22.98 | 26.55 | 58.22 | 39.33 | 52.01 | 19.49 | 29.40 | 39.02 | 23.04 | 34.34 |
| Flash STU Std. Err. | 0.35 | 0.44 | 1.15 | 0.85 | 1.40 | 1.13 | 2.05 | 1.00 | 1.23 | – |
| LDS Std. Err. | 0.35 | 0.44 | 1.15 | 0.85 | 1.40 | 1.13 | 2.04 | 1.00 | 1.23 | – |

No degradation in performance on language benchmarks using LDS representation.




LDS implementation grows linearly with sequence length generated, faster than the convolutional implementations.

# Recap

- STU, a neural network architecture with subquadratic time complexity and leading performance
- Can provably learn symmetric marginally stable LDS
  - Can learn more difficult settings in practice
- STU to LDS Distillation - O(1) Language Models
- The first method for system identification of an LDS of arbitrarily high effective memory with performance guarantees on the loss

Thanks!

Motivation
Spectral filtering
STU
Our work
Experiments