State Space Models (S4) and Mamba

Presented by Ivaylo Dimitrov

Apr 2025

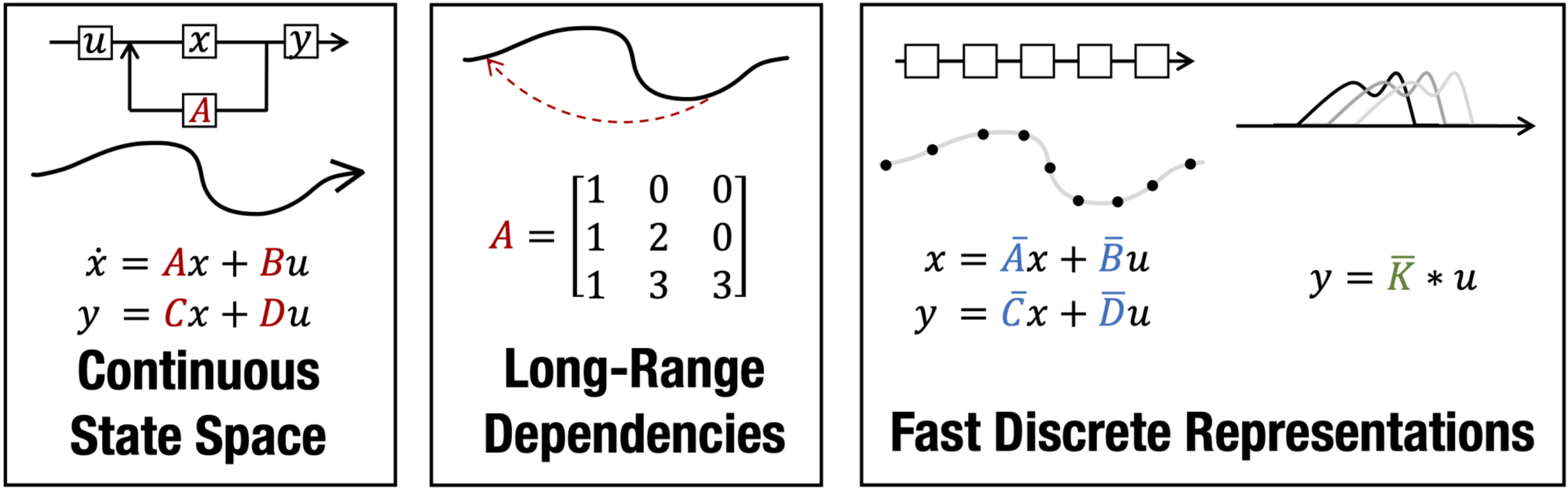
Why is it called S4?

Efficiently Modeling Long **S**equences with **S**tructured **S**tate **S**paces.

What does it consist of?

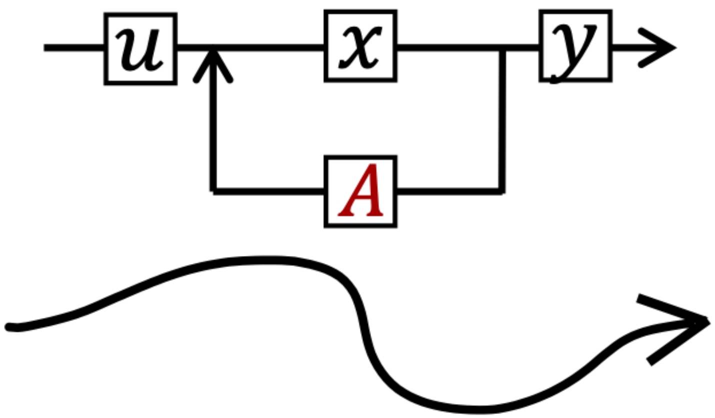
- State Space Models
- HiPPO for handling **long-range dependencies**
- Discretization for creating **recurrent** and **convolution** representations
- Why are they better than transformers? Reduction in Space and Memory complexity from $O(L^2)$ to $O(L)$, where L is Length of input

Why is it called S4?



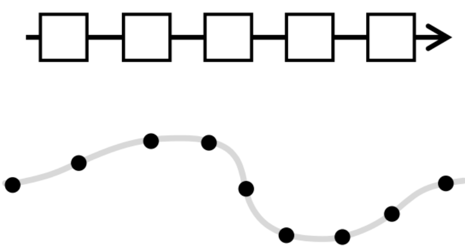
Source: https://arxiv.org/pdf/2111.00396

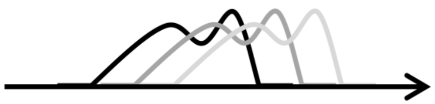
Continuous State Space Models



$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

Continuous State Space





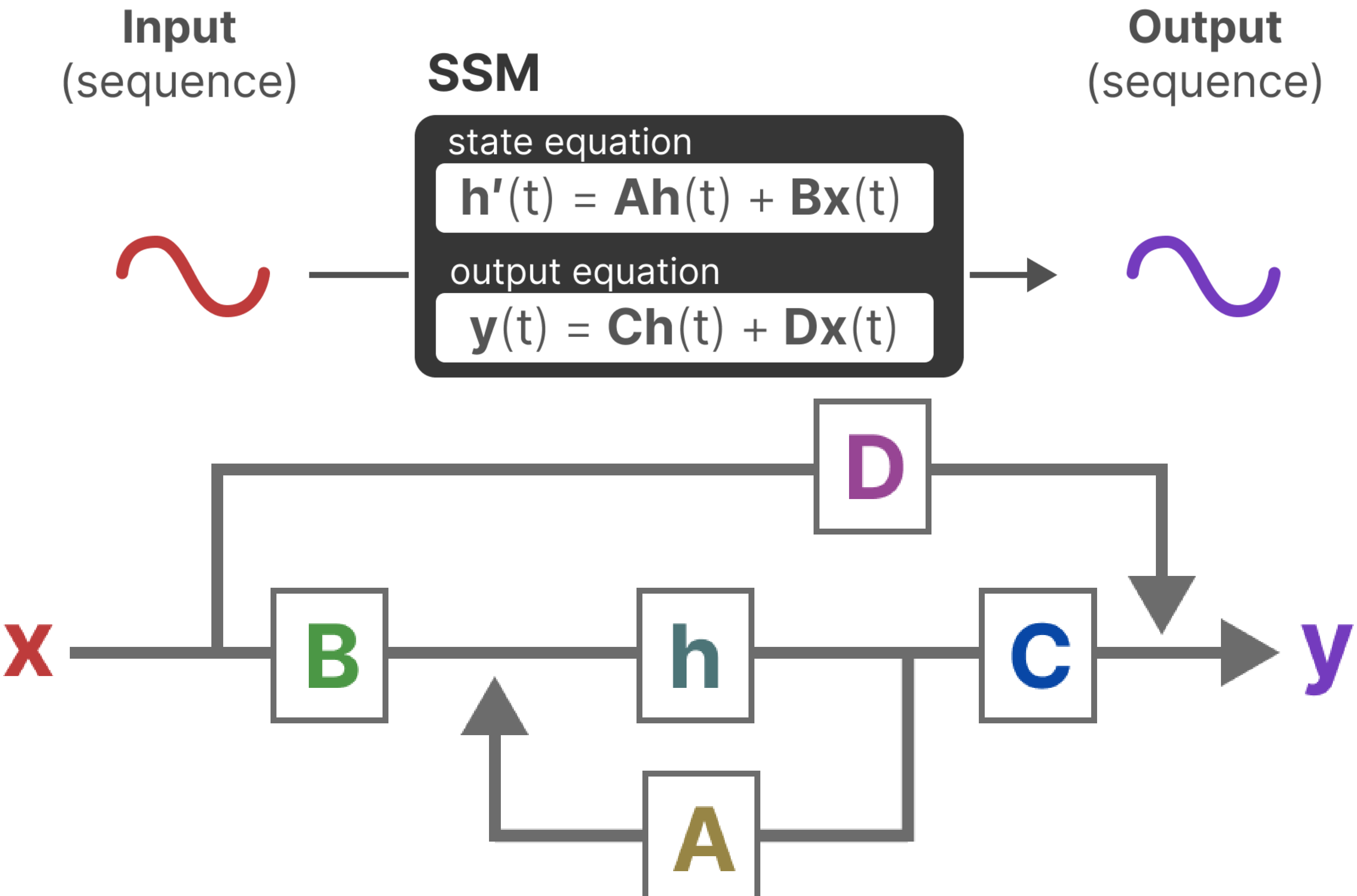
$$x = \bar{A}x + \bar{B}u$$
$$y = \bar{C}x + \bar{D}u$$

$$y = \bar{K} * u$$

Fast Discrete Representations

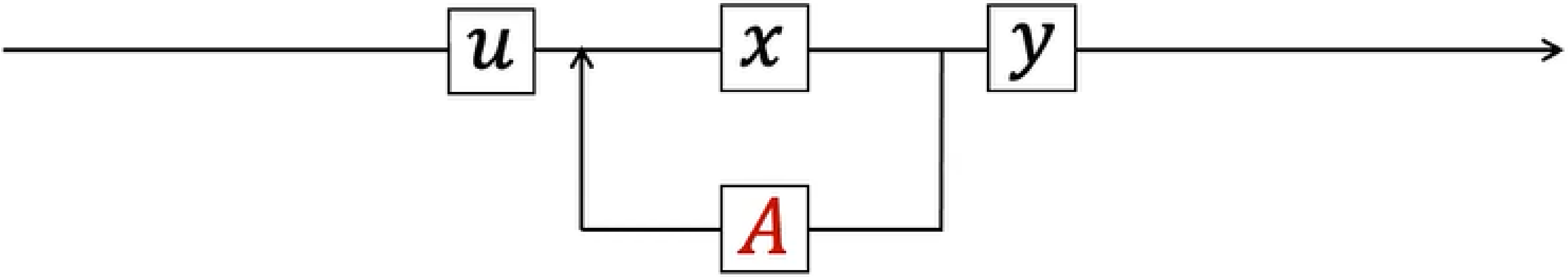
Source: https://arxiv.org/pdf/2111.00396

What is a State Space?



Source: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state

What is a State Space?

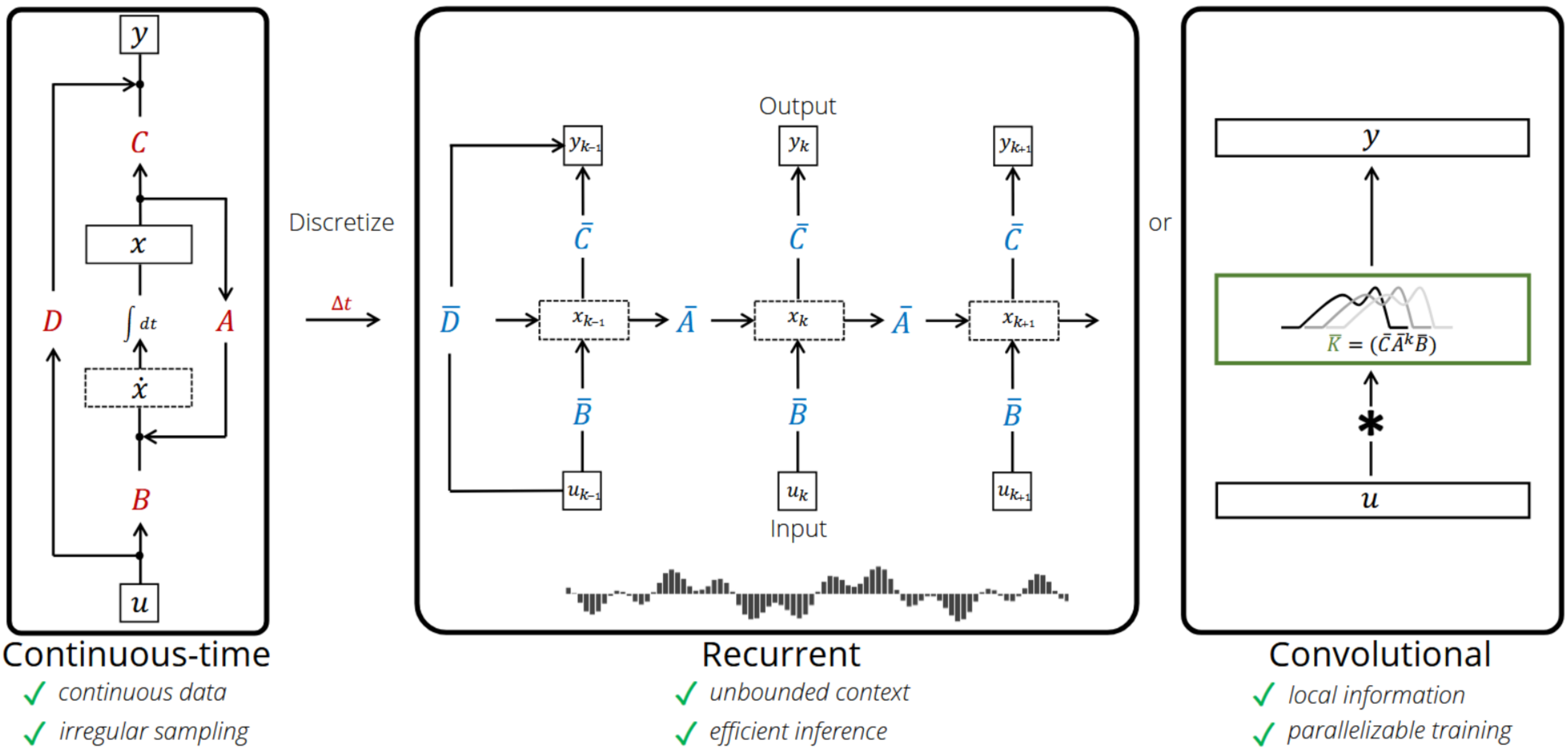


$$x' = \boldsymbol{A}x + \boldsymbol{B}u$$

$$y = \boldsymbol{C}x + \boldsymbol{D}u$$

Source: https://www.youtube.com/watch?v=OpJMn8T7Z34

Continuous State Space Models



Source: https://arxiv.org/pdf/2110.13985

Discretization

$$A = e^{\overline{A}t}$$

Therefore:

$$x(t) = Ax(0)$$

becomes:

$$x(t) = e^{\overline{A}t}x(0)$$

Approach 1: Direct Extension from Scalar Case

For a scalar differential equation $\dot{y}(t) = ay(t)$, the solution is:

$$y(t) = e^{at}y(0)$$

In the vector-matrix case, we follow the same pattern but need to use the matrix exponential instead:

$$x(t) = e^{\overline{A}t}x(0)$$

Source: https://arxiv.org/pdf/2111.00396, Claude and own calcualtions

Discretization

Taking the derivative:

$$\dot{x}(t) = \frac{d}{dt}(e^{\overline{A}t}x(0))$$

Since $x(0)$ is constant, and using the property that the derivative of a matrix exponential is:

$$\frac{d}{dt}e^{\overline{A}t} = \overline{A}e^{\overline{A}t}$$

We get:

$$\dot{x}(t) = \overline{A}e^{\overline{A}t}x(0) = \overline{A}x(t)$$

Source: https://arxiv.org/pdf/2111.00396, Claude and own calcualtions

Now we use the approximation

Discrete-time approximation [edit]

The bilinear transform is a first-order Padé approximant of the natural logarithm function that is an exact mapping of the *z*-plane to the *s*-plane. When the Laplace transform is performed on a discrete-time signal (with each element of the discrete-time sequence attached to a correspondingly delayed unit impulse), the result is precisely the Z transform of the discrete-time sequence with the substitution of

$$\begin{aligned} z &= e^{sT} \\ &= \frac{e^{sT/2}}{e^{-sT/2}} \\ &\approx \frac{1 + sT/2}{1 - sT/2} \end{aligned}$$

To discretize the continuous-time SSM, we follow prior work in using the bilinear method [43], which converts the state matrix $\boldsymbol{A}$ into an approximation $\overline{\boldsymbol{A}}$. The discrete SSM is

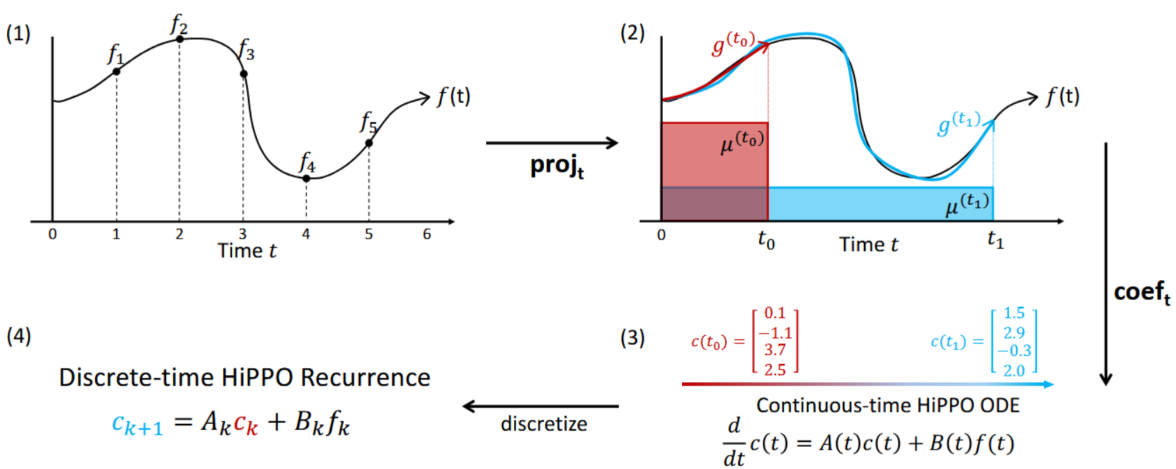
$$\begin{aligned} x_k &= \overline{\boldsymbol{A}} x_{k-1} + \overline{\boldsymbol{B}} u_k & \overline{\boldsymbol{A}} &= (\boldsymbol{I} - \Delta/2 \cdot \boldsymbol{A})^{-1} (\boldsymbol{I} + \Delta/2 \cdot \boldsymbol{A}) & & \\ y_k &= \overline{\boldsymbol{C}} x_k & \overline{\boldsymbol{B}} &= (\boldsymbol{I} - \Delta/2 \cdot \boldsymbol{A})^{-1} \Delta \boldsymbol{B} & \overline{\boldsymbol{C}} &= \boldsymbol{C}. \end{aligned} \tag{3}$$

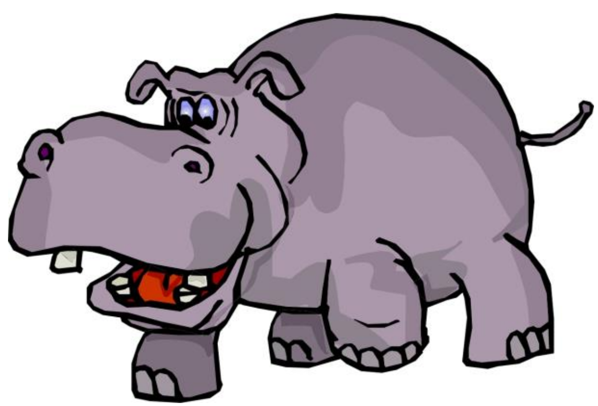
Source: https://arxiv.org/pdf/2111.00396 and https://en.wikipedia.org/wiki/Bilinear_transform

What is HiPPO?

- **Hi**gh-order **P**olynomial **P**rojection **O**perators

$$\textbf{(HiPPO Matrix)} \qquad \boldsymbol{A}_{nk} = -\begin{cases} (2n+1)^{1/2}(2k+1)^{1/2} & \text{if } n > k \\ n+1 & \text{if } n = k \,. \\ 0 & \text{if } n < k \end{cases} \qquad (2)$$

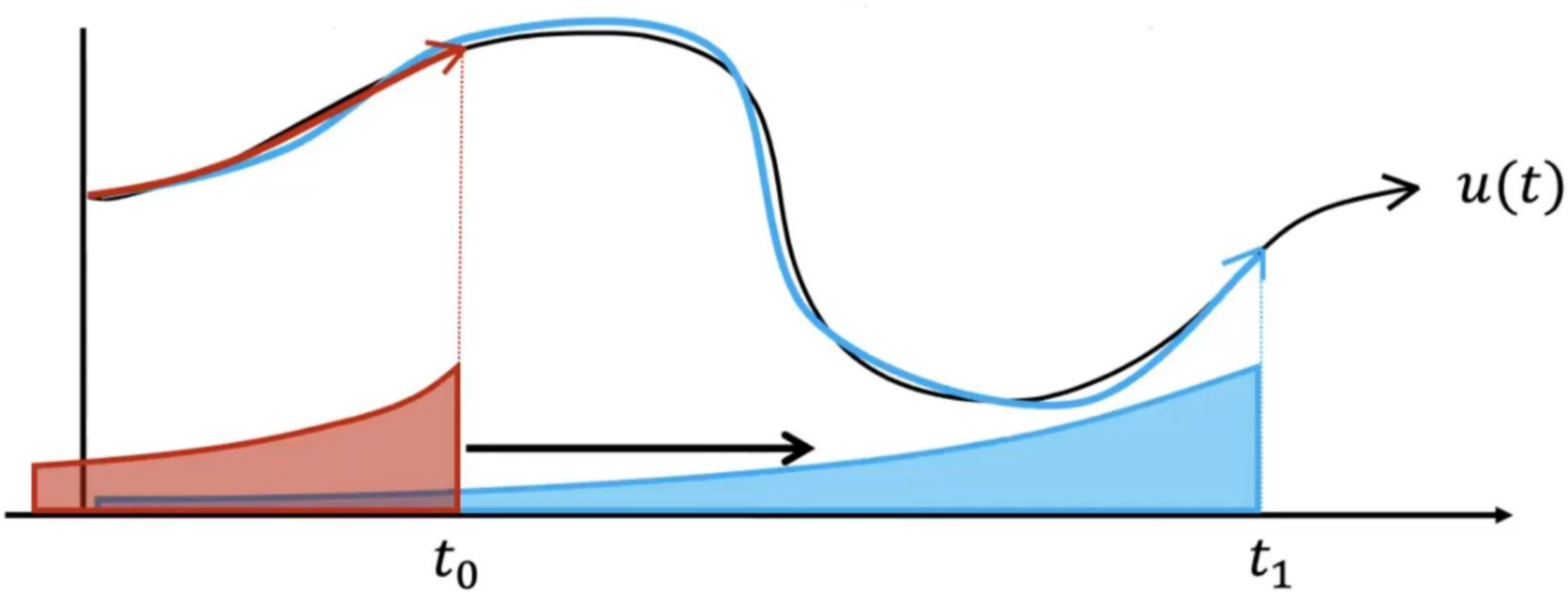




Source: https://arxiv.org/pdf/2111.00396

What is HiPPO?

- **Hi**gh-order **P**olynomial **P**rojection **O**perators

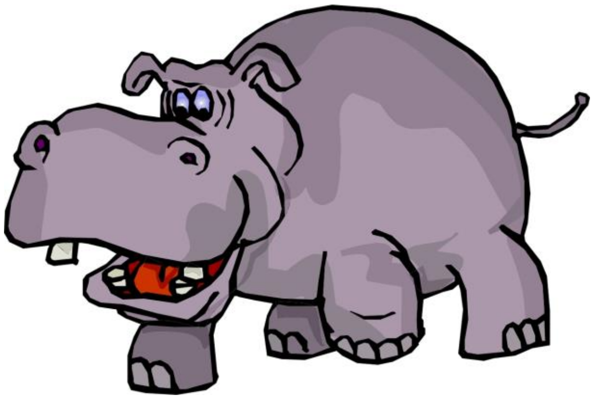


Exponential decaying measure

Source: https://www.youtube.com/watch?v=luCBXCErkCs

What is HiPPO?

- **Hi**gh-order **P**olynomial **P**rojection **O**perators
- Video:
- https://youtu.be/luCBXCErkCs?si=HXkfo9aSl4skH93m&t=1110



Source: https://www.clipartbest.com/cliparts/MTL/Mdz/MTLMdzLac.jpg

What is HiPPO? (Example)

- **Hi**gh-order **P**olynomial **P**rojection **O**perators

```
# HiPPO Matrix
N = 5
P = np.sqrt(1 + 2 * np.arange(N))
A_full = P[:, np.newaxis] * P[np.newaxis, :]
A = np.tril(A_full) - np.diag(np.arange(N))
A=-A
A
```

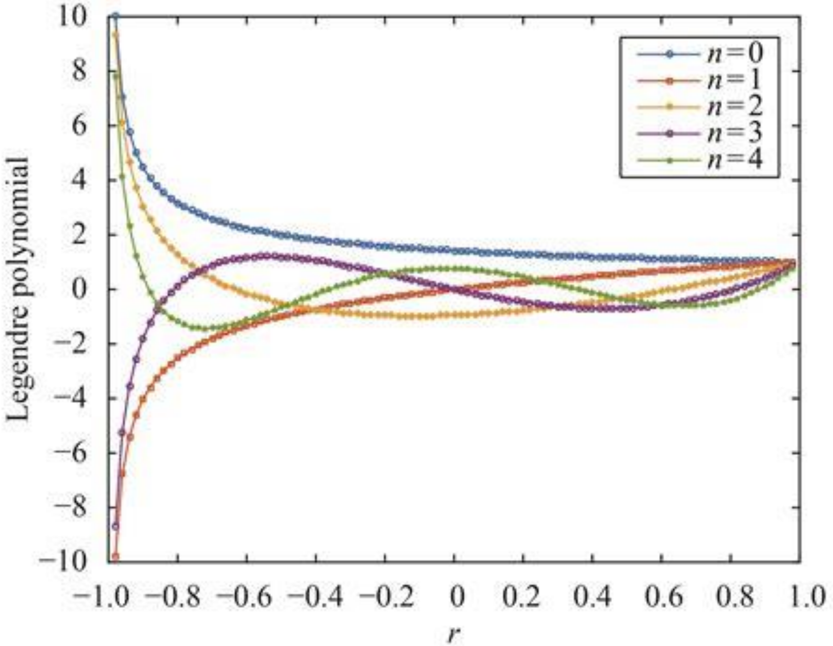
✓ 0.0s

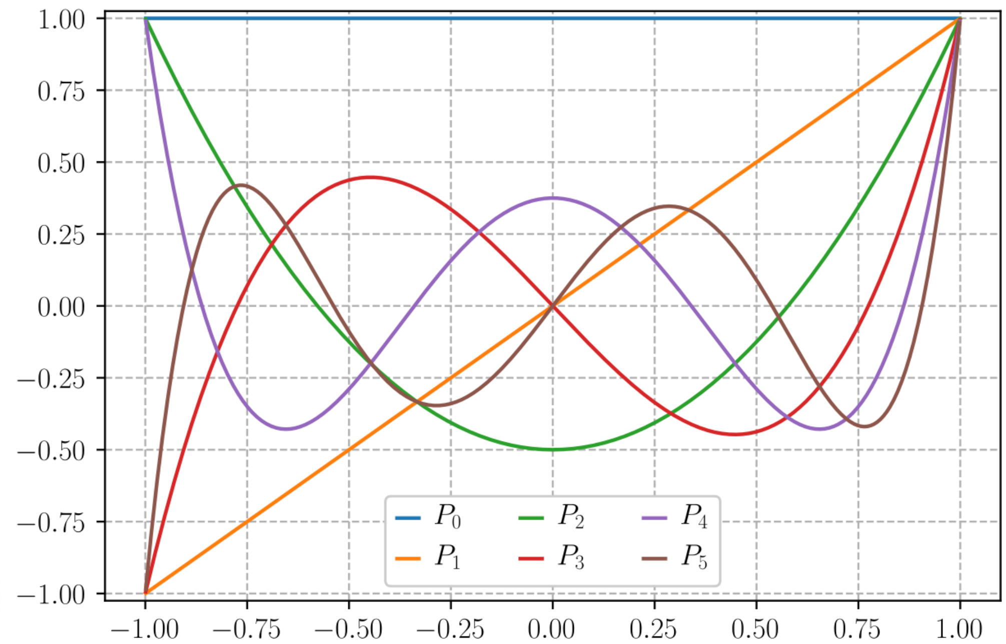
```
array([[-1.        , -0.        , -0.        , -0.        , -0.        ],
       [-1.73205081, -2.        , -0.        , -0.        , -0.        ],
       [-2.23606798, -3.87298335, -3.        , -0.        , -0.        ],
       [-2.64575131, -4.58257569, -5.91607978, -4.        , -0.        ],
       [-3.        , -5.19615242, -6.70820393, -7.93725393, -5.        ]])
```

Source: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state (and my own calculations)

Legendre polynomials

- Set of orthogonal polynomials, each with higher order. Their sum with a weight are used as function approximation.





Source1: https://en.wikipedia.org/wiki/Legendre_polynomials
Source2: https://external-content.duckduckgo.com/iu/?u=https%3A%2F%2Ftse1.mm.bing.net%2Fth%3Fid%3DOIP.io9I2Ns0heNGWmEe8as3OQHaFe%26pid%3DApi&f=1&ipt=1132e254d0dd5d37e06768872184d33e2cb9bd8373530aeca64ea83a5e1077a5

Legendre polynomials

- Set of orthogonal polynomials, each with higher order. Their sum with a weight are used as function approximation.

Main properties [edit]

Orthogonality [edit]

The standardization $P_n(1) = 1$ fixes the normalization of the Legendre polynomials (with respect to the L^2 norm on the interval $-1 \leq x \leq 1$). Since they are also orthogonal with respect to the same norm, the two statements[clarification needed] can be combined into the single equation,

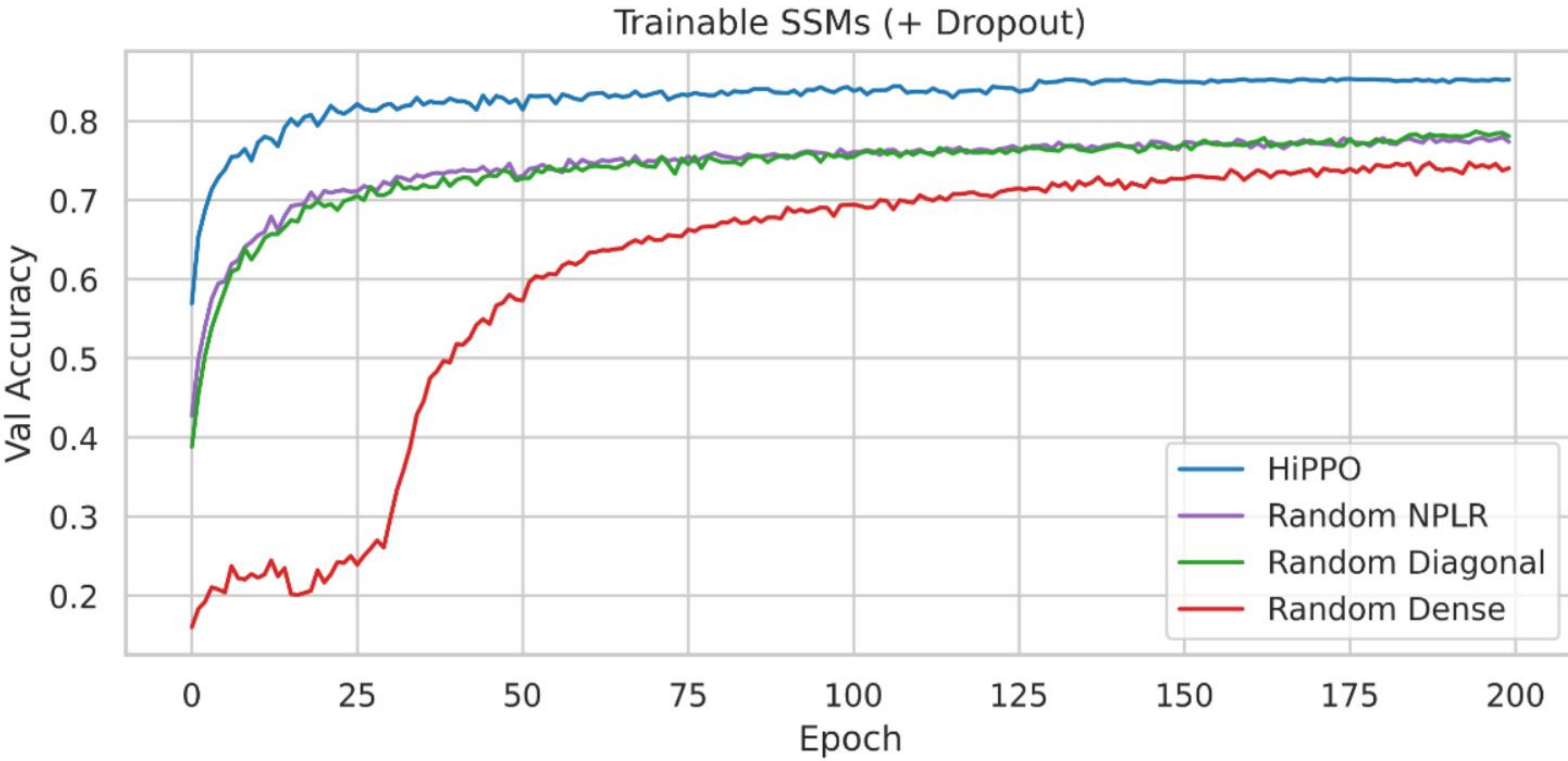
$$\int_{-1}^{1} P_m(x)P_n(x)\,dx = \frac{2}{2n+1}\delta_{mn},$$

(where δ_{mn} denotes the Kronecker delta, equal to 1 if $m = n$ and to 0 otherwise). This normalization is most readily found by employing Rodrigues' formula, given below.

Source: https://en.wikipedia.org/wiki/Legendre_polynomials

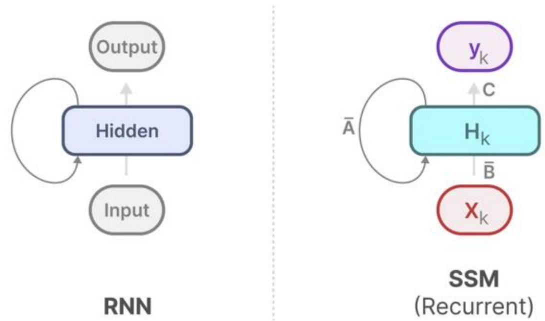
Why HiPPO Initialization?

- Ablation Study

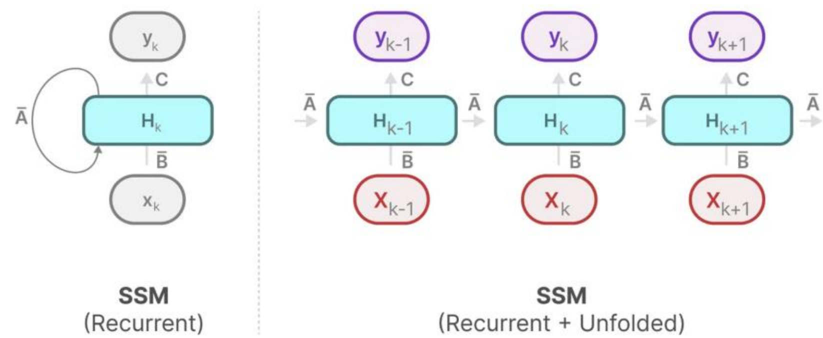


Source: https://arxiv.org/pdf/2111.00396

Recursive Representation (RNN)



Which we can unfold (or unroll) as such:



Notice how we can use this discretized version using the underlying methodology of an RNN.

Source: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state

S4 Convolution Kernel

The recurrent SSM (3) is not practical for training on modern hardware due to its sequentiality. Instead, there is a well-known connection between linear time-invariant (LTI) SSMs such as (1) and continuous convolutions. Correspondingly, (3) can actually be written as a discrete convolution.

For simplicity let the initial state be $x_{-1} = 0$. Then unrolling (3) explicitly yields

$$
\begin{aligned}
&x_0 = \overline{\boldsymbol{B}} u_0 && x_1 = \overline{\boldsymbol{A}\boldsymbol{B}} u_0 + \overline{\boldsymbol{B}} u_1 && x_2 = \overline{\boldsymbol{A}}^2 \overline{\boldsymbol{B}} u_0 + \overline{\boldsymbol{A}\boldsymbol{B}} u_1 + \overline{\boldsymbol{B}} u_2 && \dots \\
&y_0 = \overline{\boldsymbol{C}\boldsymbol{B}} u_0 && y_1 = \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}} u_0 + \overline{\boldsymbol{C}\boldsymbol{B}} u_1 && y_2 = \overline{\boldsymbol{C}\boldsymbol{A}}^2 \overline{\boldsymbol{B}} u_0 + \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}} u_1 + \overline{\boldsymbol{C}\boldsymbol{B}} u_2 && \dots
\end{aligned}
$$

This can be vectorized into a convolution (4) with an explicit formula for the convolution kernel (5).

$$
\begin{aligned}
y_k &= \overline{\boldsymbol{C}\boldsymbol{A}}^k \overline{\boldsymbol{B}} u_0 + \overline{\boldsymbol{C}\boldsymbol{A}}^{k-1} \overline{\boldsymbol{B}} u_1 + \cdots + \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}} u_{k-1} + \overline{\boldsymbol{C}\boldsymbol{B}} u_k \\
y &= \overline{\boldsymbol{K}} * u.
\end{aligned}
\tag{4}
$$

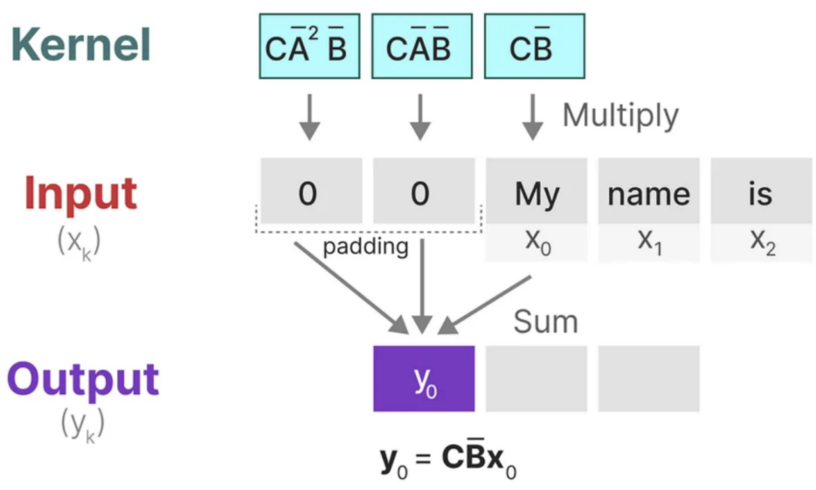
$$
\overline{\boldsymbol{K}} \in \mathbb{R}^L := \mathcal{K}_L(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \overline{\boldsymbol{C}}) := \left(\overline{\boldsymbol{C}\boldsymbol{A}}^i \overline{\boldsymbol{B}}\right)_{i \in [L]} = (\overline{\boldsymbol{C}\boldsymbol{B}}, \overline{\boldsymbol{C}\boldsymbol{A}\boldsymbol{B}}, \dots, \overline{\boldsymbol{C}\boldsymbol{A}}^{L-1} \overline{\boldsymbol{B}}). \tag{5}
$$

In other words, equation (4) is a single (non-circular) convolution and can be computed very efficiently with FFTs, *provided* that $\overline{\boldsymbol{K}}$ is known. However, computing $\overline{\boldsymbol{K}}$ in (5) is non-trivial and is the focus of our technical contributions in Section 3. We call $\overline{\boldsymbol{K}}$ the **SSM convolution kernel** or filter.

Source: https://arxiv.org/pdf/2111.00396

S4 Convolution Kernel

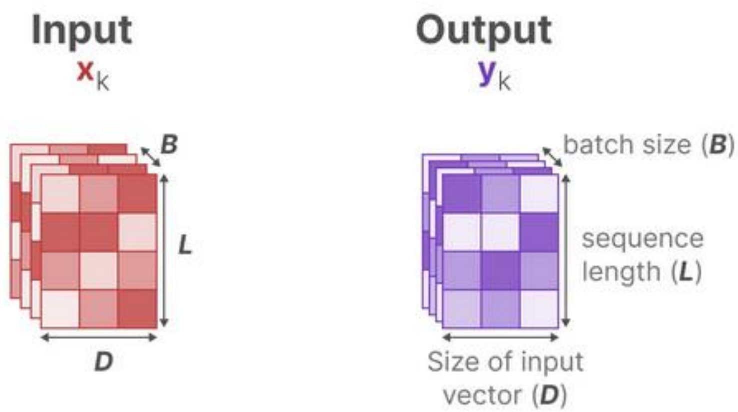
- https://youtu.be/luCBXCErkCs?si=4pu1UfoA--DGF4X7&t=2030
- Also here : **The Convolution Representation part of**



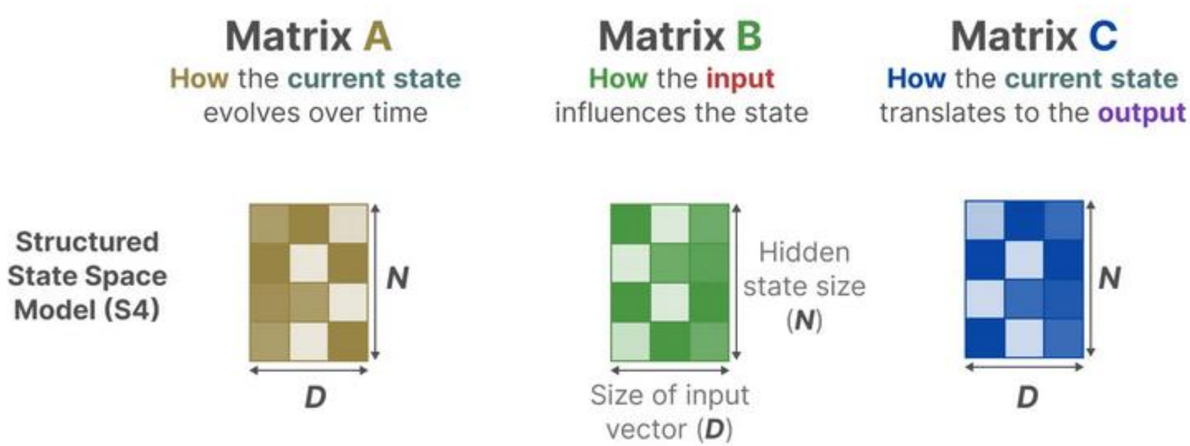
Source: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state

S4 Shape of Matrices

What's Wrong on the Picture?



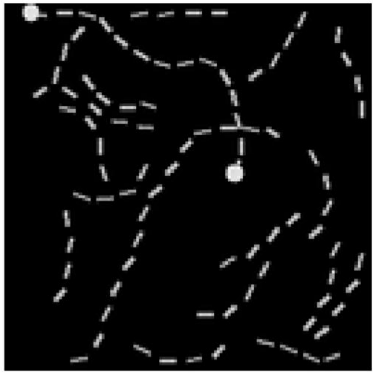
In a Structured State Space Model (S4), the matrices A, B, and C are independent of the input since their dimensions N and D are static and do not change.



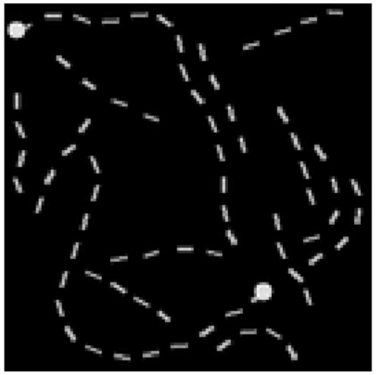
Source: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state

S4 for Sequence Modelling

- See results in the paper and ablations

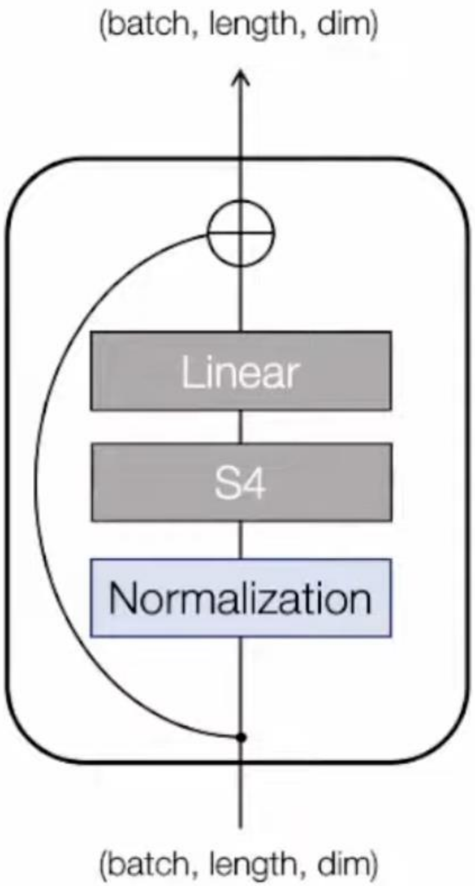


(a) A positive example.



(b) A negative example.

Figure 1: Samples of the Pathfinder task.



Source: https://www.youtube.com/watch?v=EvQ3ncuriCM, Albert Gu and https://arxiv.org/pdf/2111.00396

S4 Complexity vs Transformers

Table 1: Complexity of various sequence models in terms of sequence length ($\boldsymbol{L}$), batch size ($\boldsymbol{B}$), and hidden dimension ($\boldsymbol{H}$); tildes denote log factors. Metrics are parameter count, training computation, training space requirement, training parallelizability, and inference computation (for 1 sample and time-step). For simplicity, the state size N of S4 is tied to H. Bold denotes model is theoretically best for that metric. Convolutions are efficient for training while recurrence is efficient for inference, while SSMs combine the strengths of both.

	Convolution[3]	Recurrence	Attention	S4
Parameters	LH	$\boldsymbol{H^2}$	$\boldsymbol{H^2}$	$\boldsymbol{H^2}$
Training	$\boldsymbol{\tilde{L}H(B+H)}$	BLH^2	$B(L^2H+LH^2)$	$\boldsymbol{BH(\tilde{H}+\tilde{L})+B\tilde{L}H}$
Space	$\boldsymbol{BLH}$	$\boldsymbol{BLH}$	$B(L^2+HL)$	$\boldsymbol{BLH}$
Parallel	**Yes**	No	**Yes**	**Yes**
Inference	LH^2	$\boldsymbol{H^2}$	L^2H+H^2L	$\boldsymbol{H^2}$

Source: https://arxiv.org/pdf/2111.00396

S4 for Sequence Modelling

- Hyperparameters

Table 11: The values of the best hyperparameters found for classification datasets; LRA (Top) and images/speech (Bottom). LR is learning rate and WD is weight decay. BN and LN refer to Batch Normalization and Layer Normalization.

	Depth	**Features** H	**Norm**	**Pre-norm**	**Dropout**	**LR**	**Batch Size**	**Epochs**	**WD**	**Patience**
ListOps	6	128	BN	False	0	0.01	100	50	0.01	5
Text	4	64	BN	True	0	0.001	50	20	0	5
Retrieval	6	256	BN	True	0	0.002	64	20	0	20
Image	6	512	LN	False	0.2	0.004	50	200	0.01	20
Pathfinder	6	256	BN	True	0.1	0.004	100	200	0	10
Path-X	6	256	BN	True	0.0	0.0005	32	100	0	20
CIFAR-10	6	1024	LN	False	0.25	0.01	50	200	0.01	20
Speech Commands (MFCC)	4	256	LN	False	0.2	0.01	100	50	0	5
Speech Commands (Raw)	6	128	BN	True	0.1	0.01	20	150	0	10

S4 Math Details, not covered

- Shifted Legendre Polynomials to HiPPO
- Cauchy Kernels
- Diagonal Case
- Diagonal Plus Low Rank (DLPR)
- Normal Plus Low Rank (NLPR)
- HiPPO to DLPR

Source: https://www.youtube.com/watch?v=EvQ3ncuriCM, Albert Gu

What is S6?

- **S**elective State Space Models
- Memory allocation in GPU
- No HiPPO Needed
- Selective Scan needed, because A is dependent on the output.

Source: https://arxiv.org/pdf/2312.00752

Memory Allocation on GPU

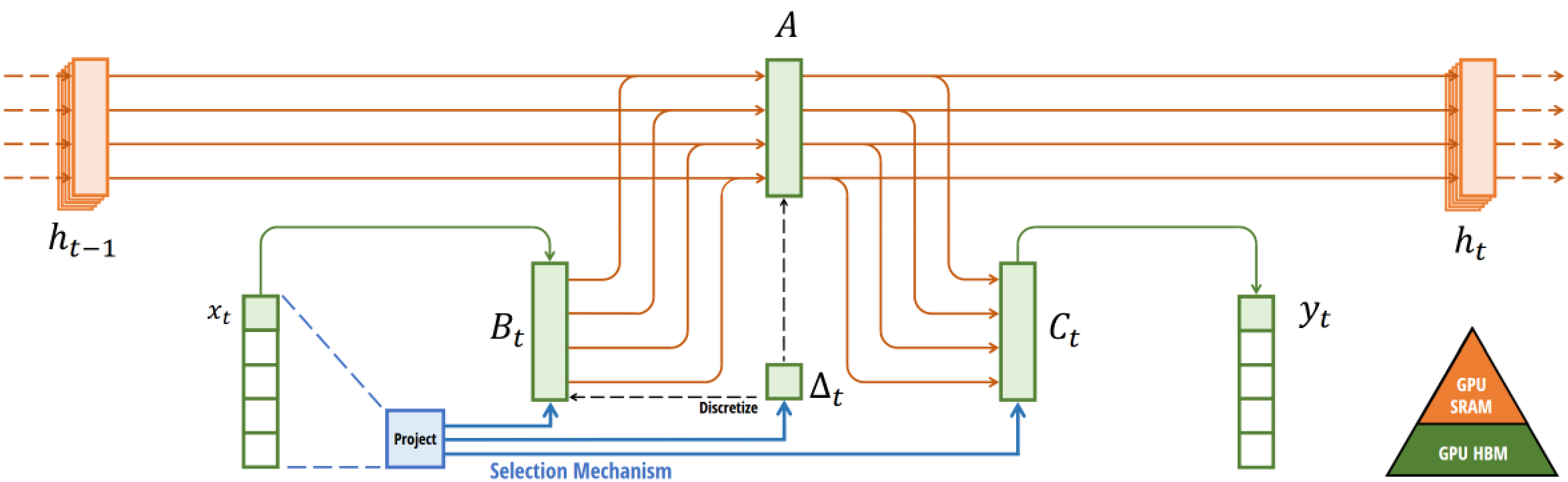


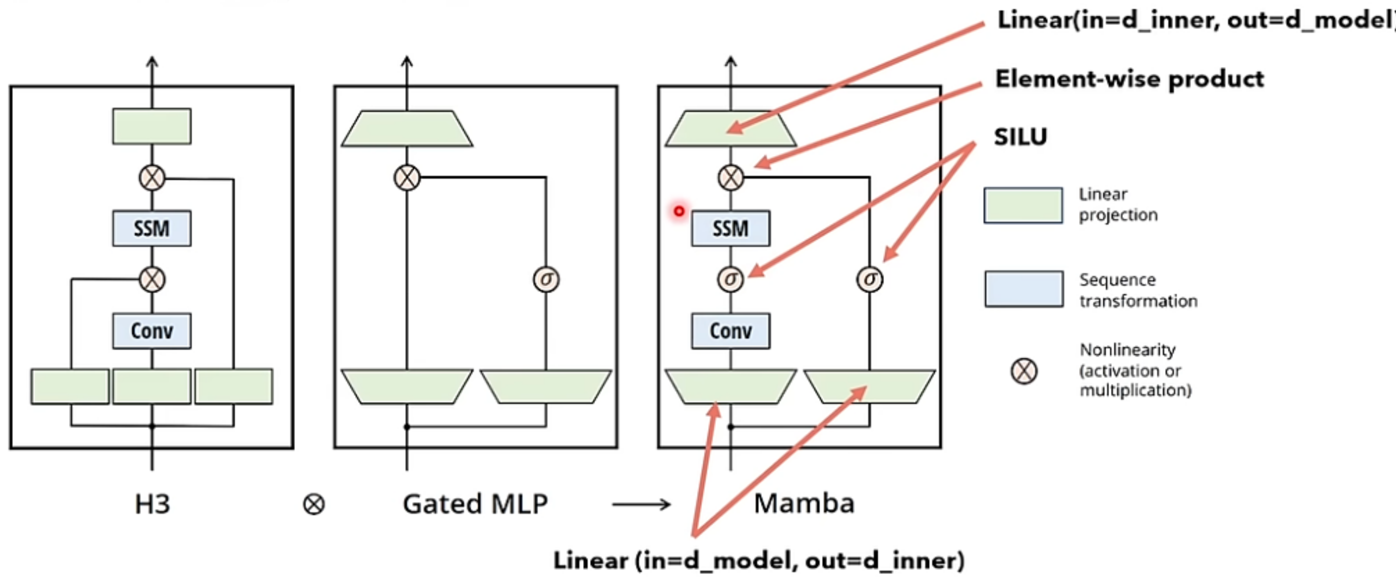
Figure 1: (**Overview**.) Structured SSMs independently map each channel (e.g. $D = 5$) of an input x to output y through a higher dimensional latent state h (e.g. $N = 4$). Prior SSMs avoid materializing this large effective state (DN, times batch size B and sequence length L) through clever alternate computation paths requiring time-invariance: the $(\Delta, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ parameters are constant across time. Our selection mechanism adds back input-dependent dynamics, which also requires a careful hardware-aware algorithm to only materialize the expanded states in more efficient levels of the GPU memory hierarchy.

Source: https://arxiv.org/pdf/2312.00752

Mamba

Mamba: the Mamba Block(1)

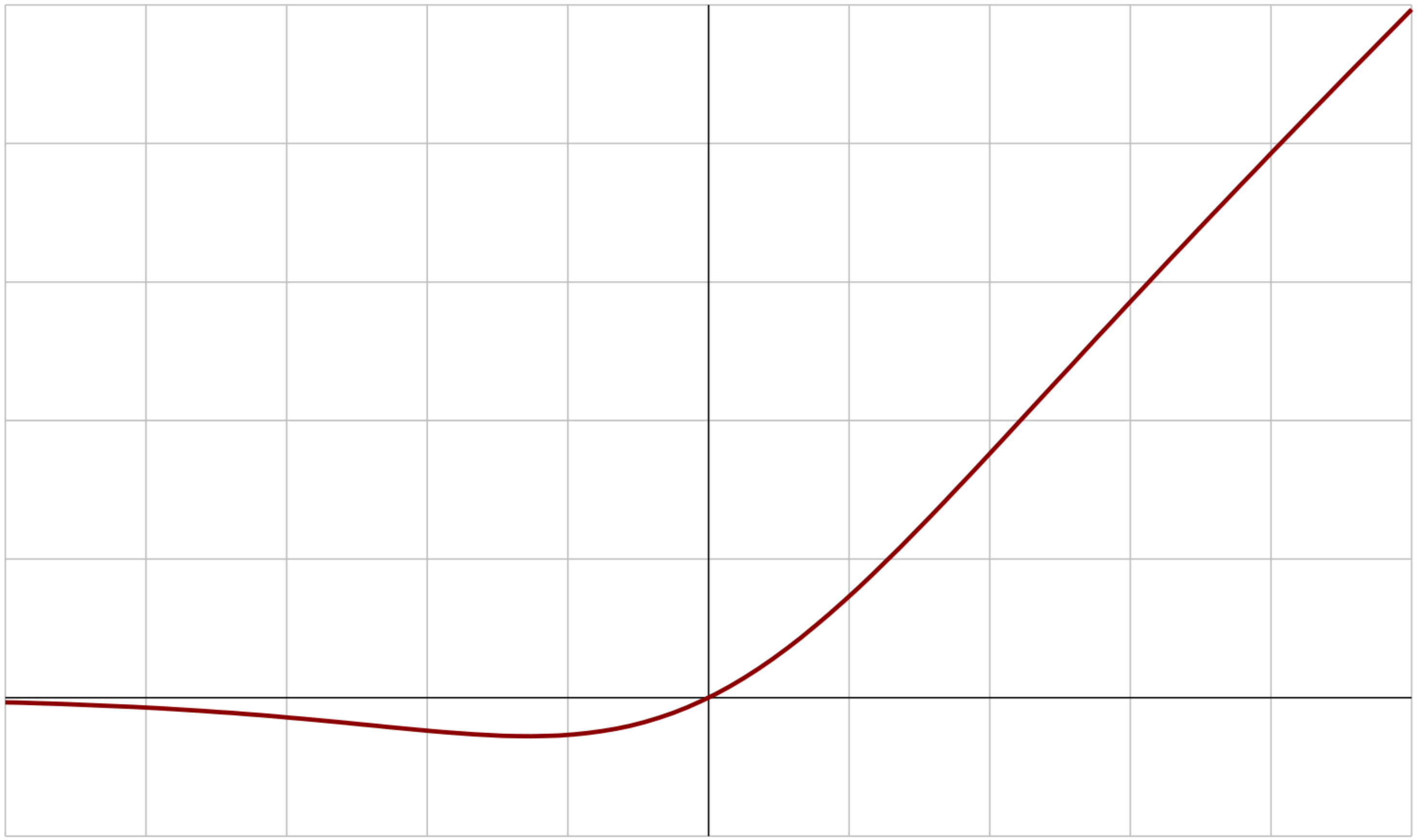
Mamba is built by stacking multiple layers of the Mamba block, shown below. This is very similar to the stacked layers of the Transformer model. The Mamba architecture derives from the *Hungry Hungry Hippo* (H3) architecture.



Source: https://github.com/hkproj/mamba-notes or https://arxiv.org/pdf/2312.00752

Mamba (Swish / SiLU)

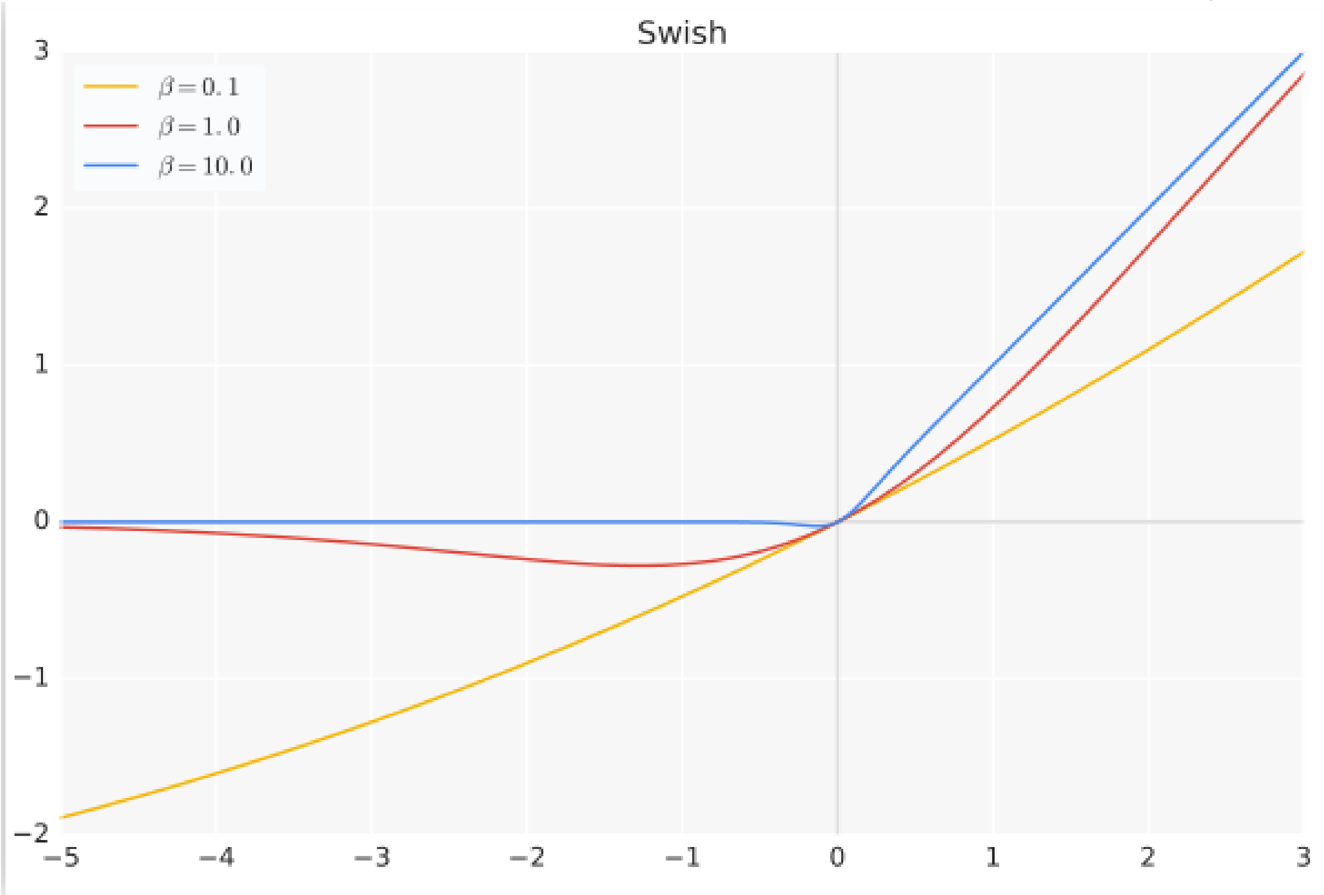
$$\text{swish}_\beta(x) = x\,\text{sigmoid}(\beta x) = \frac{x}{1 + e^{-\beta x}}$$



Source: https://en.wikipedia.org/wiki/File:Swish.svg

Mamba (Swish / SiLU)

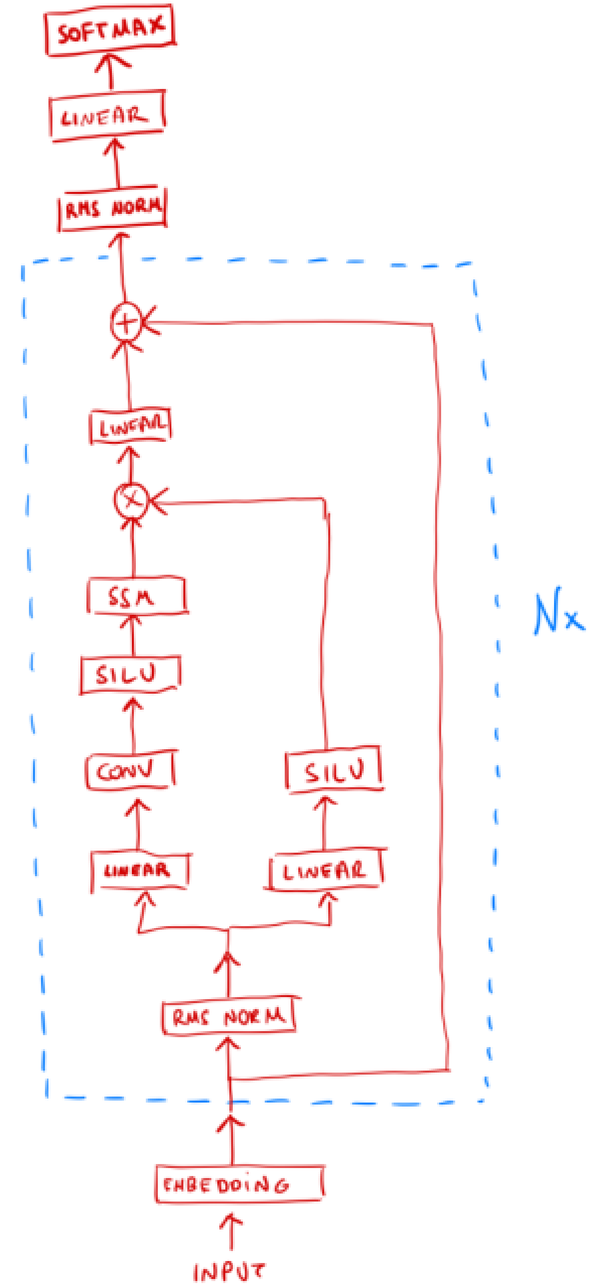
$$\mathrm{swish}_\beta(x) = x\,\mathrm{sigmoid}(\beta x) = \frac{x}{1 + e^{-\beta x}}$$



Source: https://dsworld.org/content/images/2022/06/swish.png

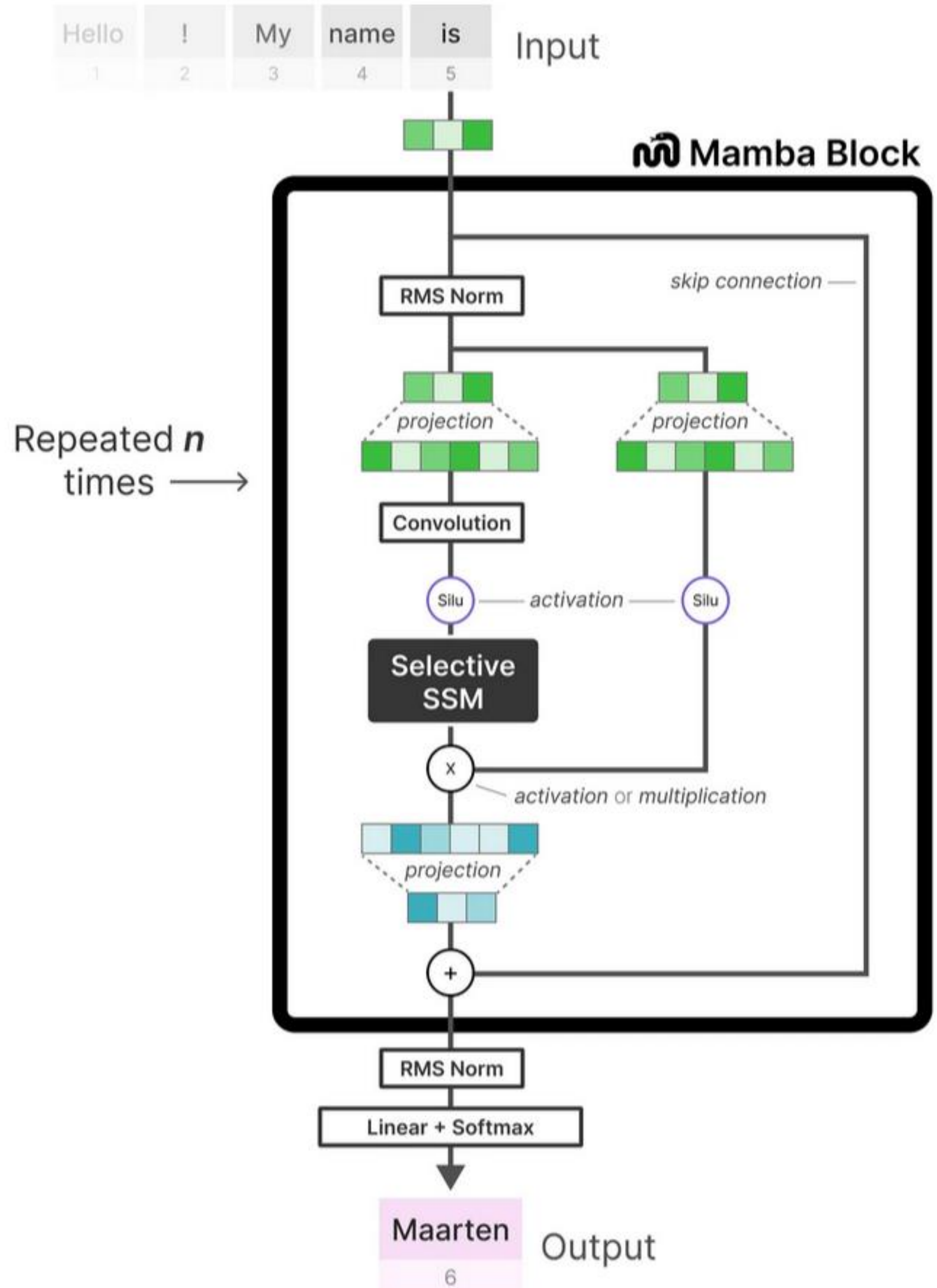
Mamba

This is where we run the SSM



Source: https://github.com/hkproj/mamba-notes based on https://arxiv.org/pdf/2312.00752

Mamba



Source: https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state

SSMs Applications

- Really long sequences:
- Audio (many samples/sec)
- DNA modelling
- Long Language Sequences
- Image example from Albert Gu MedAI Stanford

Other Mambas

- RWKV
- Mamba-MoE
- Jamba (Mamba + Transformers)
- Mamba-2

Sources

- https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state
- https://www.youtube.com/watch?v=OpJMn8T7Z34
- https://www.youtube.com/watch?v=H0KrEZC9eyA
- https://www.youtube.com/watch?v=ouF-H35atOY
- https://www.youtube.com/watch?v=QJHA-PY8zDc&list=TLGGcAWMs_ycx7syNTA0MjAyNQ
- https://www.youtube.com/watch?v=LefUVgPbnNg
- https://www.youtube.com/watch?v=8Q_tqwpTpVU
- https://newsletter.maartengrootendorst.com/p/a-visual-guide-to-mamba-and-state
- https://arxiv.org/abs/2401.04081
- https://arxiv.org/abs/2401.09417
- https://ar5iv.labs.arxiv.org/html/2202.09729
- https://ar5iv.labs.arxiv.org/html/2111.00396
- https://arxiv.org/pdf/2208.04933
- https://arxiv.org/pdf/2403.17844
- https://arxiv.org/html/2312.00752v2
- https://arxiv.org/pdf/2405.21060
- https://github.com/PeaBrane/mamba-tiny
- https://arxiv.org/pdf/2305.13048
- https://www.rwkv.com/
- https://tridao.me/blog/
- https://arxiv.org/pdf/2406.06484

Tranks for listening!
Feedback form: