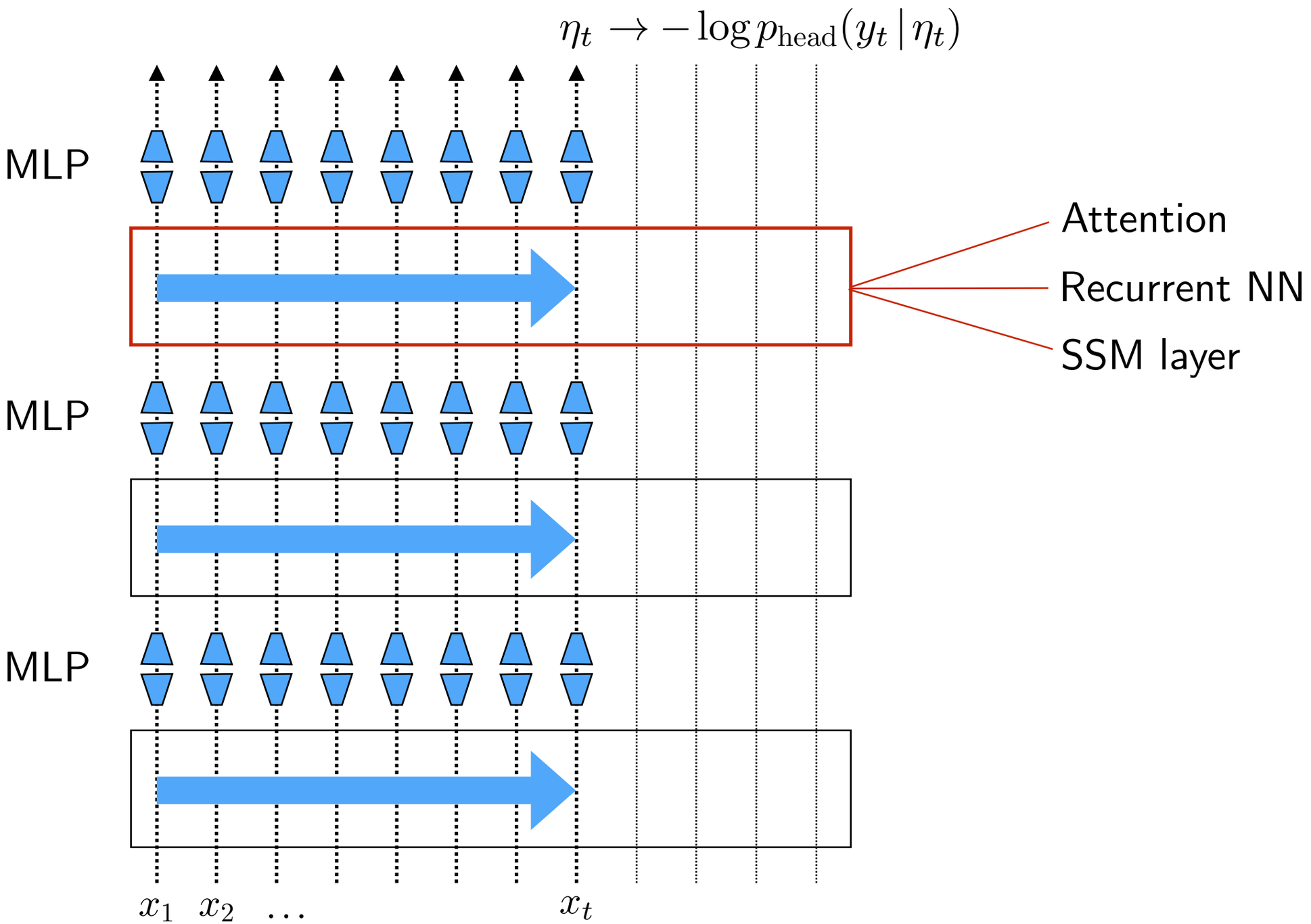
Advanced Deep Learning (BEV033DLA)
Lecture 13
State Space Models

Czech Technical University in Prague

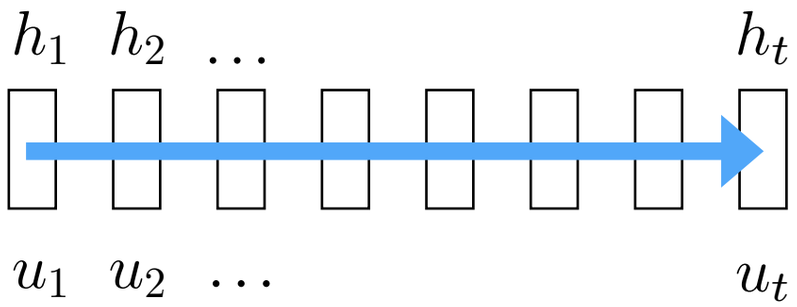
- Deep Causal Backbones
  - Limitations of transformers and RNNs
- State Space Models
  - Recurrent and convolutional forms
  - Similar filters / kernel decomposition, prefix sum form
  - Variable step size / continuous time
  - Positional encoding form
- Linear Attention Models

Deep Causal Backbones

- **Sequence / temporal modeling tasks**. Our setup:
  - Discrete time (will link to continuous time later)
  - Causal model: $p(y_t \,|\, x_1, \ldots x_t) = p_{\text{head}}(y_t; \eta(x_1, \ldots x_t))$,
  - e.g. $y_t = x_{t+1}$ – autoregressive



Motivation: Why not Transformers?

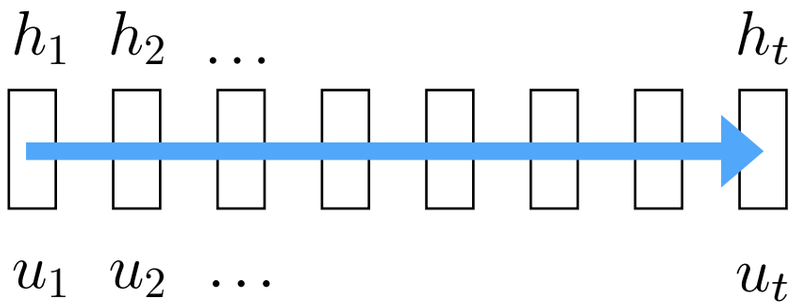


- **Attention:**
  - $k_s = W^K u_s,\ v_s = W^V u_s,\ q_t = W^Q u_t$
  - $h_t \propto \sum_{s<t} e^{\langle q_t, k_s \rangle} v_s$

- **Inference (autoregressive)**:
  - KV cache stores all past keys and values (every layer)
  - $O(Nd)$ memory and compute per step
  - N grows without bound at inference time — memory, speed and energy concerns

- **Training**:
  - $O(Nd)$ memory (with flash attention), $O(N^2 d)$ operations and IO – might be problematic for long context
  - Highly parallel – lower bound on latency is $O(\log N)$

Motivation: Why not Recurrent Neural Networks?

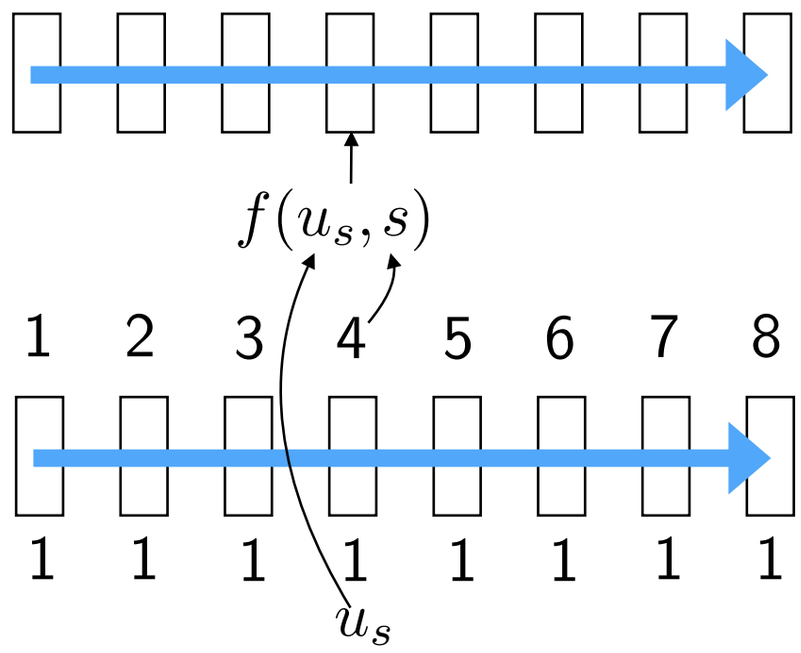


- **RNN:**
  - $h_t = f(h_{t-1}, u_t; \theta)$
  - h_t – hidden state in $\mathbb{R}^m$
  - Updated locally
- **Inference:**
  - Constant memory usage $O(m)$
  - $O(m^2)$ operations per step
  - Limited representation of the past
- **Training:**
  - $O(Nm)$, $O(Nm^2)$ operations
  - RNN is a depth N model – forward-backward stability is a concern (addressable)
  - No parallelization over N

Motivation: Expressive Power of Linear Causal Layers?

- Recall the architecture designs earlier in the course:
  - **CNN** recipe (universal approximation for translation-equivariant functions): interleave
    - linear translation-equivariant layers (convolutions)
    - point-wise non-linearities
  - **DeepSet** recipe (universal approximation for permutation-equivariant functions): interleave
    - linear permutation-invariant transforms $Au_t + B\sum_{s\neq t} u_s$
    - point-wise nonlinearities

- The simplest causal linear layer – cumulative sum:
  - $h_t = h_{t-1} + u_t$
  - $h_t = \sum_{s\leq t} u_t$
    - order blind: for position t cannot distinguish if the sequence was permuted
  - But:
    - One layer can implement position counting: $h_t = \sum_{s\leq t} 1 = t$
    - Two layers with a non-linearity in between can implement a generic position-dependent aggregation



Linear State Space Model

- **Definition: SSM Layer (Linear Recurrent Filter)**

$$h_t = A\,h_{t-1} + B\,u_t, \quad h_0 = 0,$$

$$y_t = C\,h_t.$$

  - $u_t \in \mathbb{R}^d$: *input*, $h_t \in \mathbb{R}^m$: *state*, $y_t \in \mathbb{R}^p$: *output*
  - $A \in \mathbb{R}^{m\times m}$: state transition matrix, $B \in \mathbb{R}^{m\times d}$: input injection matrix, $C \in \mathbb{R}^{p\times m}$: readout matrix

- **Universality of reservoir computers** Grigoryeva & Ortega, arXiv:1712.00754

  Setting: Input: semi-infinite sequence $\mathbf{u} = (\ldots, u_{-1}, u_0)$. **Target**: any causal, time-invariant filter F with *fading memory* (distant past influences output less and less). **Goal**: approximate F to within ε uniformly over all inputs.

State update	Readout	Universal?
$h_t = A\,h_{t-1} + B\,u_t$	linear Ch_t	No
$h_t = A\,h_{t-1} + B\,u_t$	polynomial $\phi(h_t)$	Yes
$h_t = A(u_t)\,h_{t-1} + B\,u_t$, at least affine $A(u)$	linear Ch_t	Yes

  - Universality is achieved either by a nonlinear readout (requires larger m), or by making A **input-dependent** (c.f. structure of Mamba and Griffin)
  - Caution: says nothing about how large m needs to be

Unrolled (Convolution) Form

- Unroll the recurrence $h_t = Ah_{t-1} + Bu_t$ starting from $h_0 = 0$:

$$h_1 = Ah_0 + B\,u_1 = B\,u_1,$$

$$h_2 = Ah_1 + B\,u_2 = AB\,u_1 + B\,u_2,$$

$$h_3 = Ah_2 + B\,u_3 = A^2B\,u_1 + AB\,u_2 + B\,u_3,$$

$$\vdots$$

$$h_t = \sum_{s=0}^{t-1} A^s B\,u_{t-s}.$$

  - Apply the output map: $y_t = Ch_t = \sum_{s=0}^{t-1} \underbrace{C\,A^s\,B}_{K_s}\,u_{t-s}$
  - For scalar input and output ($d = p = 1$), $K_s = C^\mathsf{T} A^s B$ is a scalar and the filter is a **convolution** with kernel K: $y = K * u$.
- **Parallel training v1**: use the convolution form — parallel across the entire sequence: $O(N \log N)$ compute and memory, $O(\log N)$ time with FFT. **But**: need to compute the kernel / its FFT. Is it work-efficient for limited m? What about $d > 1$?
- **Inference**: use the reccurence per step — sequential $\mathcal{O}(m)$ memory, $\mathcal{O}(m^2)$ compute.

Similar Filters / Kernel Decomposition

First and 2nd Order Filters

- **Recall 1st order EWA:**

$$h_t^{(1)} = r_1 h_{t-1}^{(1)} + u_t$$

Unrolled: $h_t^{(1)} = \sum_{s \geq 0} r_1^s u_{t-s}$

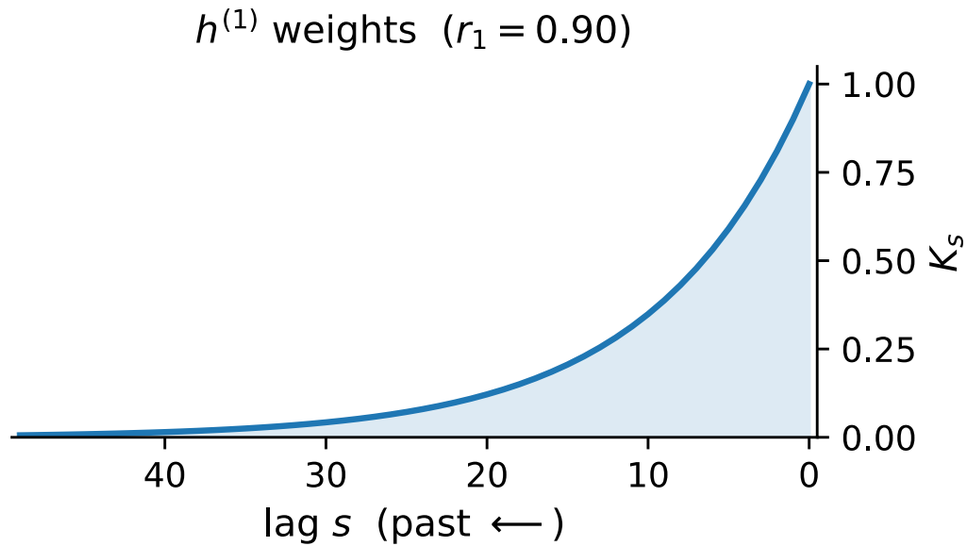
- **Apply the second EWA on top:**

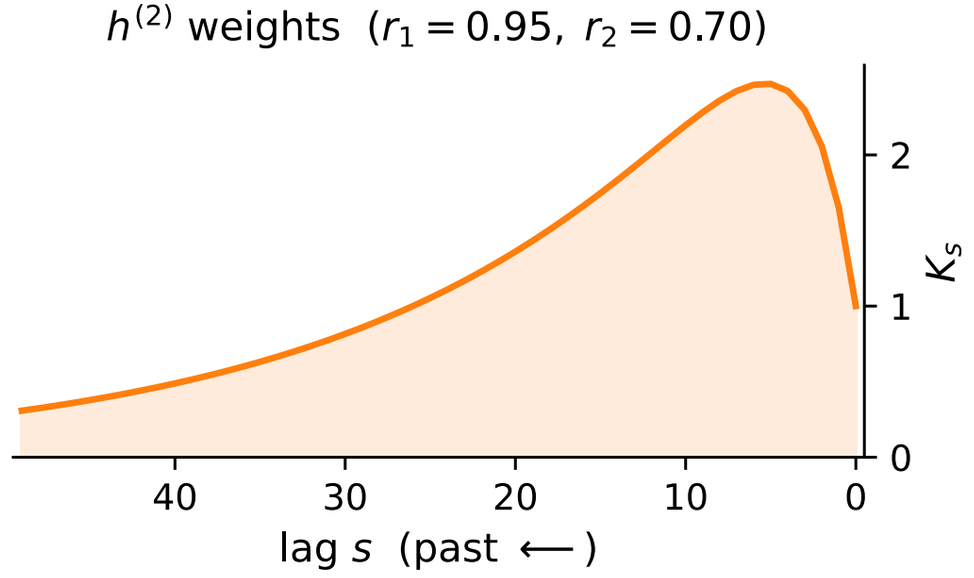
$$h_t^{(2)} = r_2 h_{t-1}^{(2)} + h_t^{(1)}$$
$$= r_2 h_{t-1}^{(2)} + r_1 h_{t-1}^{(1)} + u_t$$

- Can write in a matrix form:

$$\underbrace{\begin{pmatrix} h_t^{(1)} \\ h_t^{(2)} \end{pmatrix}}_{h_t} = \underbrace{\begin{pmatrix} r_1 & 0 \\ r_1 & r_2 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} h_{t-1}^{(1)} \\ h_{t-1}^{(2)} \end{pmatrix}}_{h_{t-1}} + \underbrace{\begin{pmatrix} 1 \\ 1 \end{pmatrix}}_{B} u_t$$

Obtained a 2nd order filter





Harmonic 2nd Order Filters

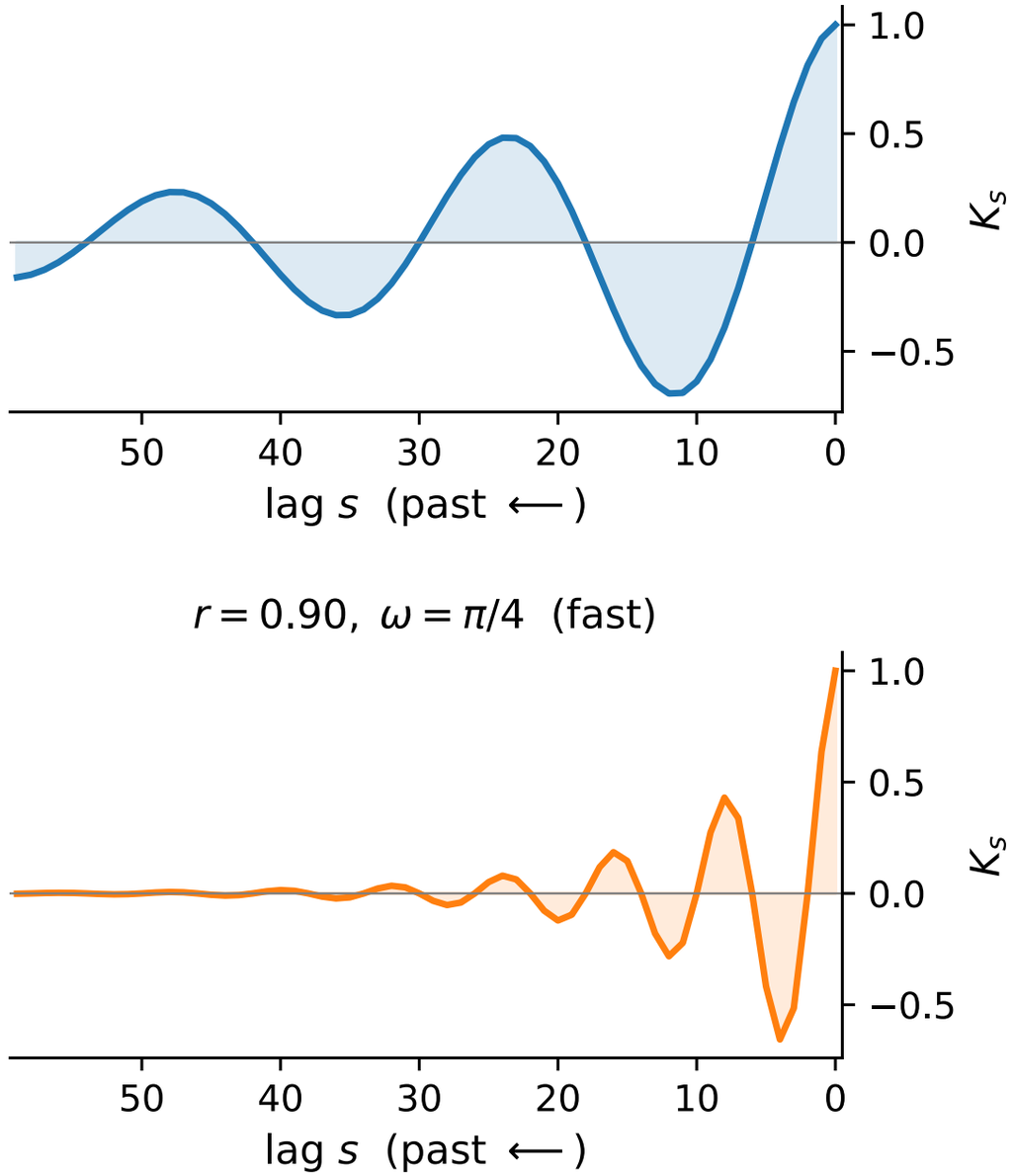
- A differnt 2nd order filter:

$$\underbrace{\begin{pmatrix} h_t^{(1)} \\ h_t^{(2)} \end{pmatrix}}_{h_t} = \underbrace{r \begin{pmatrix} \cos\omega & -\sin\omega \\ \sin\omega & \cos\omega \end{pmatrix}}_{A = r\,R(\omega)} \underbrace{\begin{pmatrix} h_{t-1}^{(1)} \\ h_{t-1}^{(2)} \end{pmatrix}}_{h_{t-1}} + \underbrace{\begin{pmatrix} 1 \\ 0 \end{pmatrix}}_{B} u_t$$

Kernel (for readout $y_t = h_t^{(1)}$):

$$K_s = \begin{pmatrix} 1 \\ 0 \end{pmatrix}^{\mathsf{T}} A^s B = r^s \cos(s\omega)$$

- $r \to 1$: sustained sinusoid, no decay
- $r < 1$: exponentially damped oscillation
- ω sets the frequency



Prony Decomposition: Filter

- **A general linear filter of m'th order**: $h_t = Ah_{t-1} + Bu_t, \quad y_t = Ch_t$
- **Equivalent representation**: write $A = V\Lambda V^{-1}$, then

$$h_t = V\Lambda V^{-1} h_{t-1} + Bu_t$$

$$\underbrace{V^{-1}h_t}_{\tilde{h}_t} = \Lambda \underbrace{(V^{-1}h_{t-1})}_{\tilde{h}_{t-1}} + \underbrace{(V^{-1}B)}_{\tilde{B}} u_t \qquad \text{readout:} \quad y_t = \underbrace{CV}_{\tilde{C}} \tilde{h}_t$$

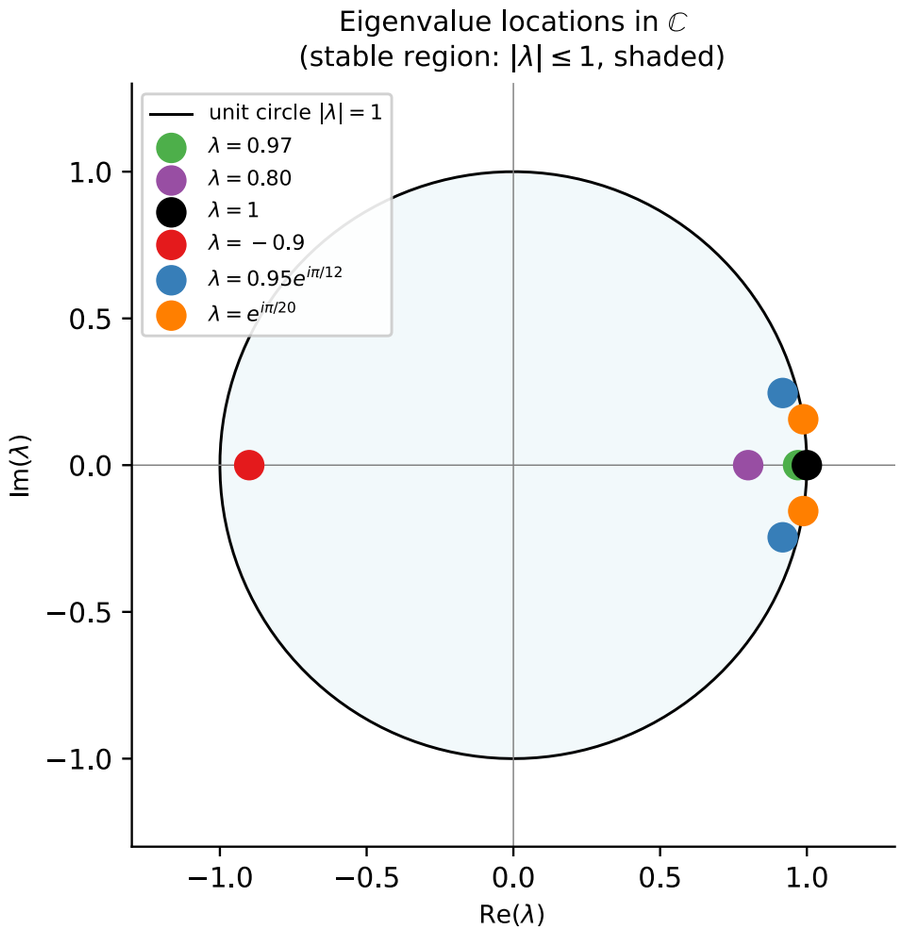
- Can reduce to $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_m)$, where λ_j are eigenvalues of A (generically true if A has distinct eigenvalues). Complex eigenvalues come in conjugate pairs $\lambda_j = r_j e^{\pm i\omega_j}$.
- This corresponds to the real decomposition

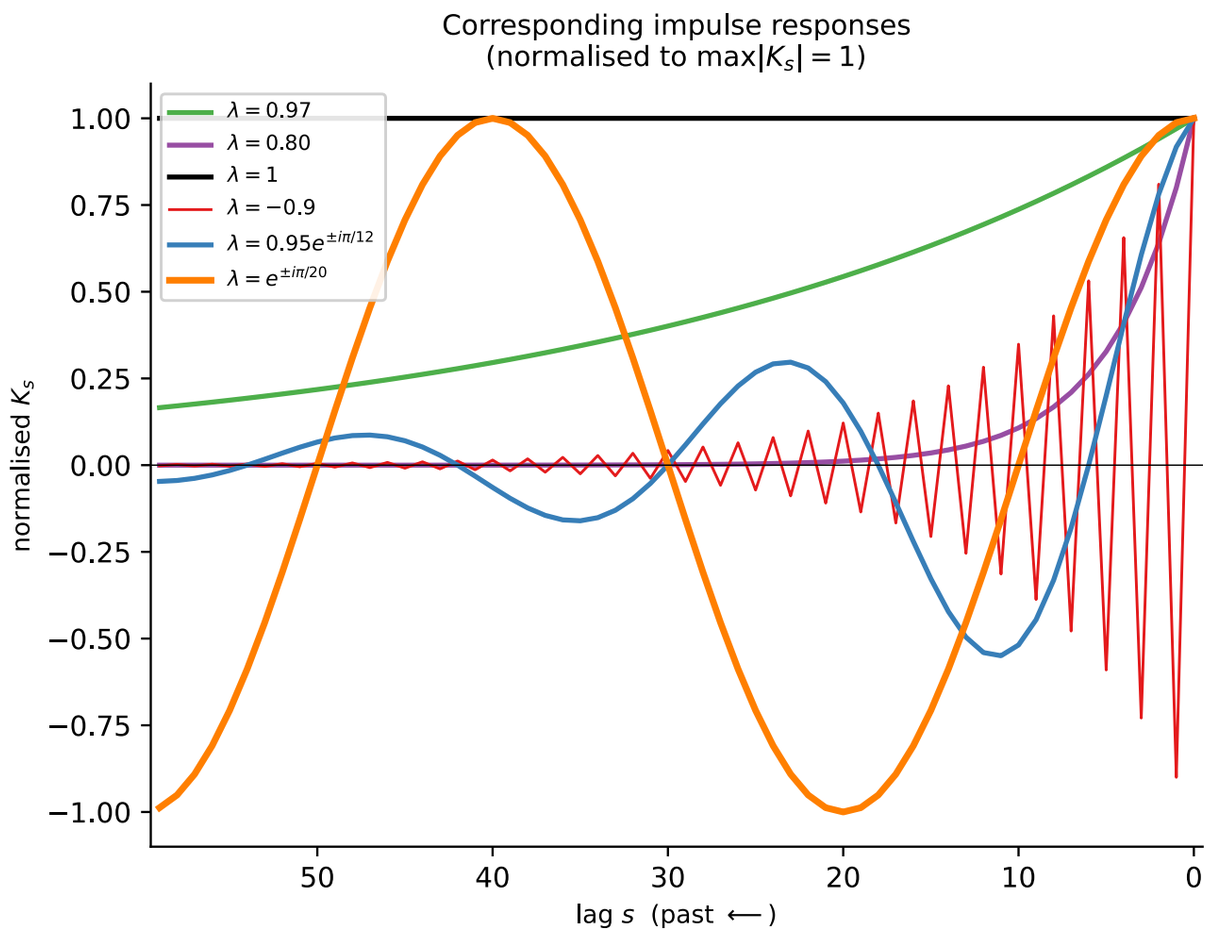
$$A = V_{\mathbb{R}} J_{\mathbb{R}} V_{\mathbb{R}}^{-1}, \quad J_{\mathbb{R}} = \begin{pmatrix} \lambda_1 & & & & & & \\ & \ddots & & & & & \\ & & \lambda_p & & & & \\ & & & r_j \begin{pmatrix} \cos\omega_j & -\sin\omega_j \\ \sin\omega_j & \cos\omega_j \end{pmatrix} & & & \\ & & & & \ddots & & \\ & & & & & r_m \begin{pmatrix} \cos\omega_m & -\sin\omega_m \\ \sin\omega_m & \cos\omega_m \end{pmatrix} \end{pmatrix}$$

Prony Decomposition: Kernel

- For any m-th order real LTI system the kernel is a linear combination of EWA filters and dumped harmonic filters (per readout dimension):

$$K_s = \underbrace{\sum_{j=1}^{p} \alpha_j \, r_j^s}_{\text{real eigenvalues}} + \underbrace{\sum_{j=1}^{q} 2\rho_j \, r_j^s \cos(\omega_j s + \phi_j)}_{\text{conjugate pairs}}$$





Real positive: EMA. $\lambda = 1$: flat average. $\lambda < 0$: alternating sign. Complex pair inside unit circle: damped sinusoid. Unit circle: sustained sinusoid. Outside of the unit circle: unstable, diverging response.

Parallel Training v2

- **SSM step as affine map:**
  - $b_t = \tilde{B} u_t$
  - $h_t = A_t h_{t-1} + b_t \quad \Leftrightarrow \quad \begin{pmatrix} h_t \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} A_t & b_t \\ 0 & 1 \end{pmatrix}}_{\Phi_t} \begin{pmatrix} h_{t-1} \\ 1 \end{pmatrix} \quad \Leftrightarrow \quad h_t = \Phi_t \Phi_{t-1} \dots \Phi_2 \Phi_1 \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
- Matrix product is associative: $\Phi_1(\Phi_2\Phi_3) = (\Phi_1\Phi_2)\Phi_3$ (could also say that composition of affine maps is associative — more direct but less clear)
  - In the Prony basis each A block is 1×1 (scalar multiply) or 2×2 (damped rotation) — cheap matrix products
- **Prefix Scan:** for an *associative* operator $\oplus$ compute all prefixes

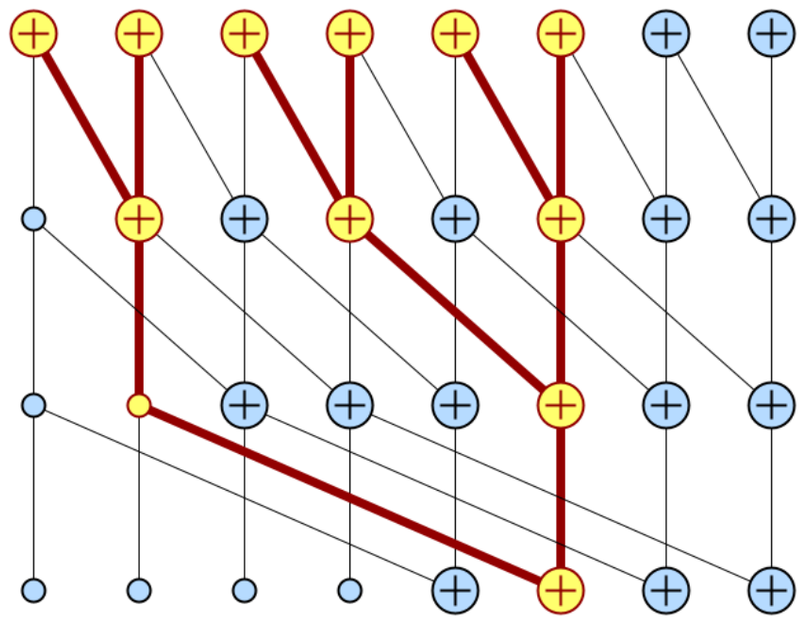
  $\phi_1,$

  $\phi_1 \oplus \phi_2,$

  $\phi_1 \oplus \phi_2 \oplus \phi_3,$

  $\dots$

  - Binary reduction tree: depth $\mathcal{O}(\log N)$, work $\mathcal{O}(N)$ — works for any associative $\oplus$ (addition, max, matrix product, . . .)



Parallel Training v2

Cost summary

Model	Training work	Training depth	Infer work/step	Infer memory
Transformer	$\mathcal{O}(N^2 d)$	$\mathcal{O}(\log N)$	$\mathcal{O}(N d + d^2)$	$\mathcal{O}(N d)$
Deep RNN	$\mathcal{O}(N m^2)$	$\mathcal{O}(N)$	$\mathcal{O}(m^2)$	$\mathcal{O}(m)$
SSM (scan)	$\mathcal{O}(N m)$	$\mathcal{O}(\log N)$	$\mathcal{O}(m^2)$	$\mathcal{O}(m)$

N: sequence length; d: model dimension; m: SSM state size.

- SSM-based models offer *parallel training* (faster than Transformers – work IO bounded) combined with *constant-memory recurrent inference* (like RNNs)

Variable Step Size

Zero-order hold: the bridge to physical time

- Input held constant for $\Delta \in \mathbb{N}$ steps: $u_k = u$, $k = 1, \ldots, \Delta$. Unrolling the recurrence gives a *cumulative update*:

$$\boxed{h_\Delta = \bar{A}(\Delta)\, h_0 + \bar{B}(\Delta)\, u,} \qquad \bar{A}(\Delta) = A^\Delta, \quad \bar{B}(\Delta) = (I - A)^{-1}(I - A^\Delta)\, B$$

- Tokens at irregular gaps Δ_k: replace (A, B) with $(\bar{A}(\Delta_k), \bar{B}(\Delta_k))$ at each step

In the Prony basis no matrix power / inverse needed:

- **Scalar EWA block** r**:** $\bar{A}(\Delta) = r^\Delta$; $\quad \bar{B}(\Delta) = \dfrac{1 - r^\Delta}{1 - r}\, b$
- **Rotation block** (r, ω)**:** $\bar{A}(\Delta) = r^\Delta R(\Delta\omega)$; $\quad \bar{B}(\Delta) = (I - rR(\omega))^{-1}(I - r^\Delta R(\Delta\omega))\, B$

One model handles: discrete text (Δ=1), downsampled audio (small Δ), irregular sensor logs (variable Δ_k).

Straightforward to generalise to $\Delta \in \mathbb{R}$ (we avoided continuous integration and matrix exponentiation).

SSM Reduction to A=I (xPos / RoPE trick)

EWA channel $h_s = r\,h_{s-1} + b\,u_s,\;\; y_t = c\,h_t,\;\; r \in (0,1)$. Define $z_s = r^{-s} h_s$ (i.e. $h_s = r^s z_s$).
Plug into the recurrence:

$$\underbrace{r^t z_s}_{h_s} = r\,\underbrace{r^{s-1} z_{s-1}}_{h_{s-1}} + b\,u_s \quad \xrightarrow{\times\, r^{-s}} \quad \boxed{z_s = z_{s-1} + \underbrace{r^{-s} b}_{\text{input enc.}}\; u_s,} \qquad y_t = \underbrace{c\,r^t}_{\text{output enc.}}\; z_t$$

Harmonic block $h_s = r\,R(\omega)\,h_{s-1} + b\,u_s$, $R(\omega)$ is $2{\times}2$ rotation.

Same substitution $h_s = r^s R(s\omega)\, z_s$ gives:

$$\boxed{z_s = z_{s-1} + \underbrace{r^{-s} R(-s\omega)\, b}_{\text{input enc.}}\, u_s,} \qquad y_t = r^t\, z_t^\top \underbrace{R(-t\omega)\, c}_{\text{output enc.}}$$

- Fully reduced to the basic case $A = I$ (with a position-dependent input / readout)
- **Parallel training v3**: prefix sums only, no matrix products
- Effectively **derived** RoPE (channels split in pairs and rotated)
- Irregular time steps and Mamba are reducible to this form as well (need to compute cumulative time positions first)

Linear Attention Models

Retentive Networks (RetNet)

Sun et al., *Retentive Network: A Successor to Transformer for Large Language Models*, 2023 [arXiv:2307.08621]

Key idea: make B and C *input-dependent* via linear projections with positional rotation (cf. $A{=}I$ reduction):

$$k_t = R(t\omega)\, W_K u_t \in \mathbb{R}^m, \qquad q_t = R(t\omega)\, W_Q u_t \in \mathbb{R}^m, \qquad v_t = W_V u_t \in \mathbb{R}^d$$

With $B_t = k_t$, $C_t = q_t^\top$, and $A = \gamma I$, the SSM state $H_t \in \mathbb{R}^{m\times d}$:

$$H_t = \gamma\, H_{t-1} + k_t\, v_t^\top, \qquad y_t = H_t^\top q_t$$

unrolls to a scaled-dot-product-like form:

$$y_t = \sum_{s=1}^{t} \gamma^{t-s} \left(q_t^\top k_s\right) v_s \qquad \Leftrightarrow \qquad Y = (Q^\top K \odot D)\, V, \quad D_{ts} = \begin{cases} \gamma^{t-s} & s \le t \\ 0 & s > t \end{cases}$$

- **Parallel (training):** $O(N^2(m+d))$ – matrix multiplications.
- **Recurrent (inference):** $H_t = \gamma H_{t-1} + k_t v_t^\top$, $y_t = H_t^\top q_t$, $O(md)$ mem and work per

Linear Attention

Katharopoulos et al., *Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention*, ICML 2020 [arXiv:2006.16236]

Standard causal (softmax) attention:

$$y_t = \frac{\sum_{s \le t} \exp(q_t^\mathsf{T} k_s)\, v_s}{\sum_{s \le t} \exp(q_t^\mathsf{T} k_s)}$$

Key idea: $\exp(q^\mathsf{T} k)$ is a PD kernel, use $\phi(q)^\mathsf{T}\phi(k)$ instead (e.g. $\phi = \mathrm{elu}(x)+1$)

$$y_t = \frac{\phi(q_t)^\mathsf{T} \sum_{s \le t} \phi(k_s)\, v_s^\mathsf{T}}{\phi(q_t)^\mathsf{T} \sum_{s \le t} \phi(k_s)}$$

- The sums can be accumulated left-to-right:

$$H_t = H_{t-1} + \phi(k_t)\, v_t^\mathsf{T}, \qquad y_t \propto \phi(q_t)^\mathsf{T} H_t$$

— up to normalization, a linear SSM layer with $A = I$, matrix-valued state $H_t \in \mathbb{R}^{d \times d}$; no positional encoding used

Performer

Choromanski et al., *Rethinking Attention with Performers*, ICLR 2021 [arXiv:2009.14794]

Goal: find ϕ such that $\phi(q)^\mathsf{T}\phi(k) \approx \exp(q^\mathsf{T}k)$

Key tool — Gaussian moment generating function: for $\omega \sim \mathcal{N}(0, I)$,

$$\mathbb{E}_\omega\left[e^{\omega^\mathsf{T}x}\right] = e^{\|x\|^2/2}$$

so, using linearity of ω in the exponent,

$$\mathbb{E}_\omega\left[e^{\omega^\mathsf{T}q}\, e^{\omega^\mathsf{T}k}\right] = \mathbb{E}_\omega\left[e^{\omega^\mathsf{T}(q+k)}\right] = e^{\|q+k\|^2/2} = e^{\|q\|^2/2}\, e^{q^\mathsf{T}k}\, e^{\|k\|^2/2}$$

Feature map (draw $\omega_1, \ldots, \omega_r \overset{\text{iid}}{\sim} \mathcal{N}(0, I)$):

$$\phi(x) = \frac{e^{-\|x\|^2/2}}{\sqrt{r}}\left(e^{\omega_1^\mathsf{T}x}, \ldots, e^{\omega_r^\mathsf{T}x}\right)$$