Robustifying State-space Models for Long Sequences via Approximate Diagonalization

Annan Yu[1], Arnur Nigmetov[2], Dmitriy Morozov[2], Michael W. Mahoney[2, 3, 4], and N. Benjamin Erichson[2, 4]

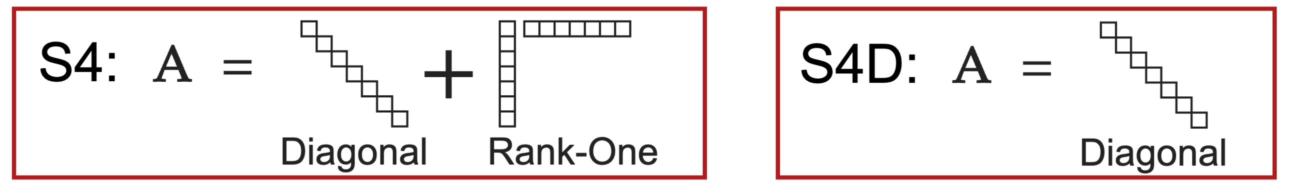
[1]Cornell University [2]Lawrence Berkeley National Laboratory [3]University of California, Berkeley [4]International Computer Science Institute

State-space Models

State-space models (SSMs) leverage linear, time-invariant (LTI) systems,

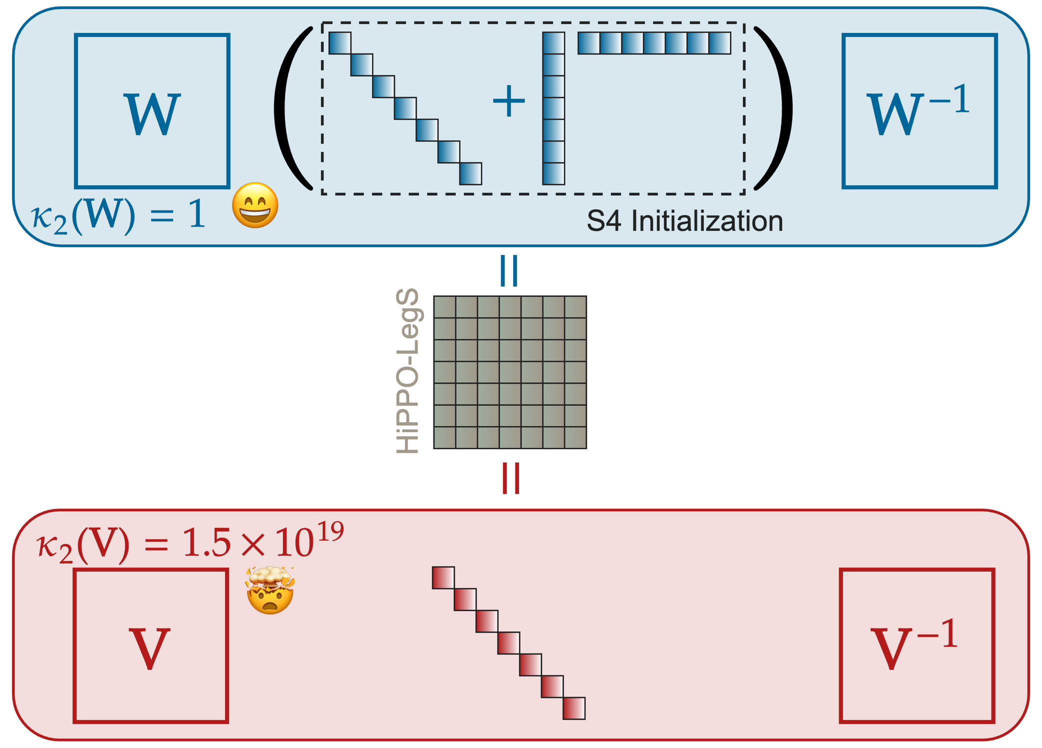
$$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t),$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t),$$

to model long sequential data. To speed up the training and inference of an SSM, one often enforces a simplified structure on $\mathbf{A}$. For example, the S4 model uses a diagonal-plus-rank-one structure while the S4D model sets $\mathbf{A}$ to be diagonal.

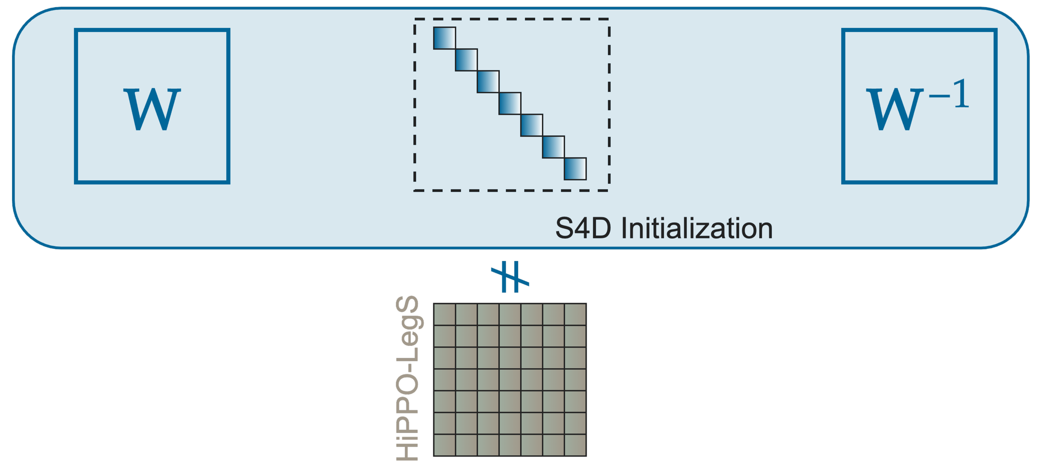


The HiPPO Initialization

To attain a good performance, an SSM often needs to be initialized by pre-designed matrices. A particularly successful one of them is called HiPPO-LegS. The matrix $\mathbf{A}$ from HiPPO-LegS can be easily written into the diagonal-plus-rank-one form by a similarity transform. On the other hand, however, it cannot be diagonalized in a numerically stable way.



To overcome this issue, the S4D model transforms $\mathbf{A}$ into the diagonal-plus-rank-one form and discards the rank-one part, but that means it deviates from the HiPPO-LegS initialization.

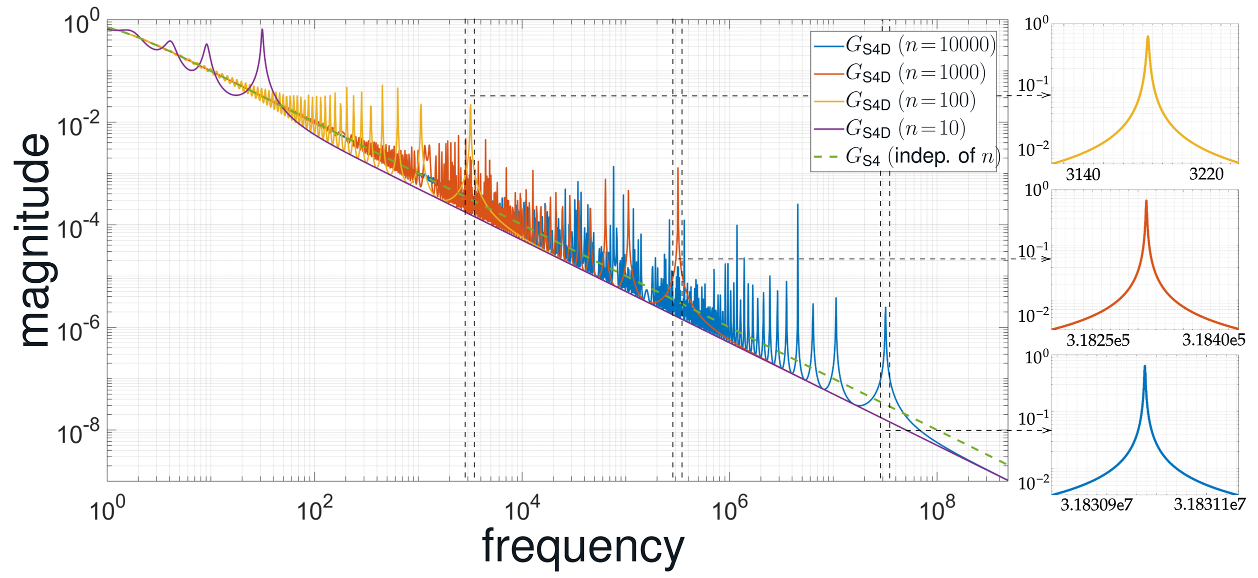


Comparing the S4 and S4D Initializations

Why can we remove the rank-one component from HiPPO-LegS to initialize an S4D model? To answer this question, we study the transfer function G of an LTI system. The transfer function maps the inputs to the outputs in the frequency domain by multiplication:

$$\hat{\mathbf{y}}(s) = G(is)\hat{\mathbf{u}}(s).$$

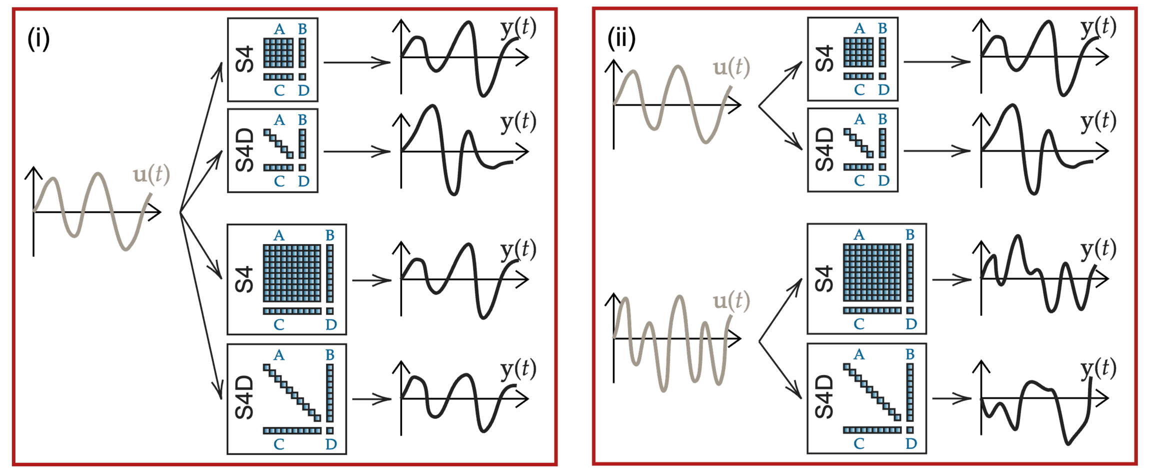
We compare the transfer functions of the S4 initialization and those of the S4D initialization.



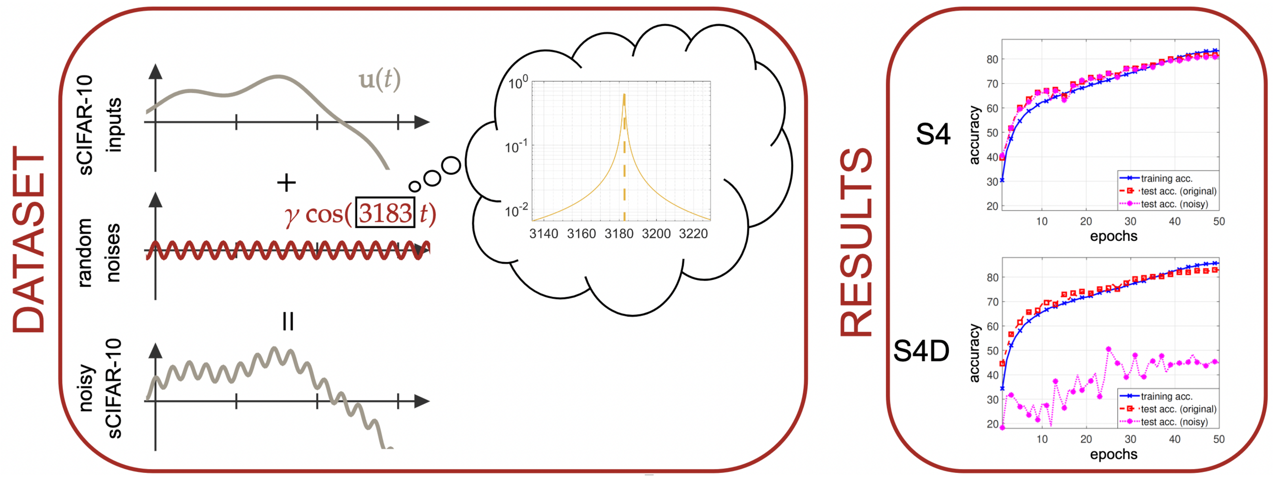
We show that as n, the number of internal states in $\mathbf{x}$, goes to infinity, two things happen:

(i) G_{S4D} converges to G_{S4} pointwise. Hence, fixing a smooth input $\mathbf{u}$, the output $\mathbf{y}$ of S4D converges to the output $\mathbf{y}$ of S4 in L^2.

(ii) G_{S4D} does not converge to G_{S4} uniformly. Hence, for any n, there exists a smooth input $\mathbf{u}$ so that $\mathbf{y}$ of S4D is very different from $\mathbf{y}$ of S4.

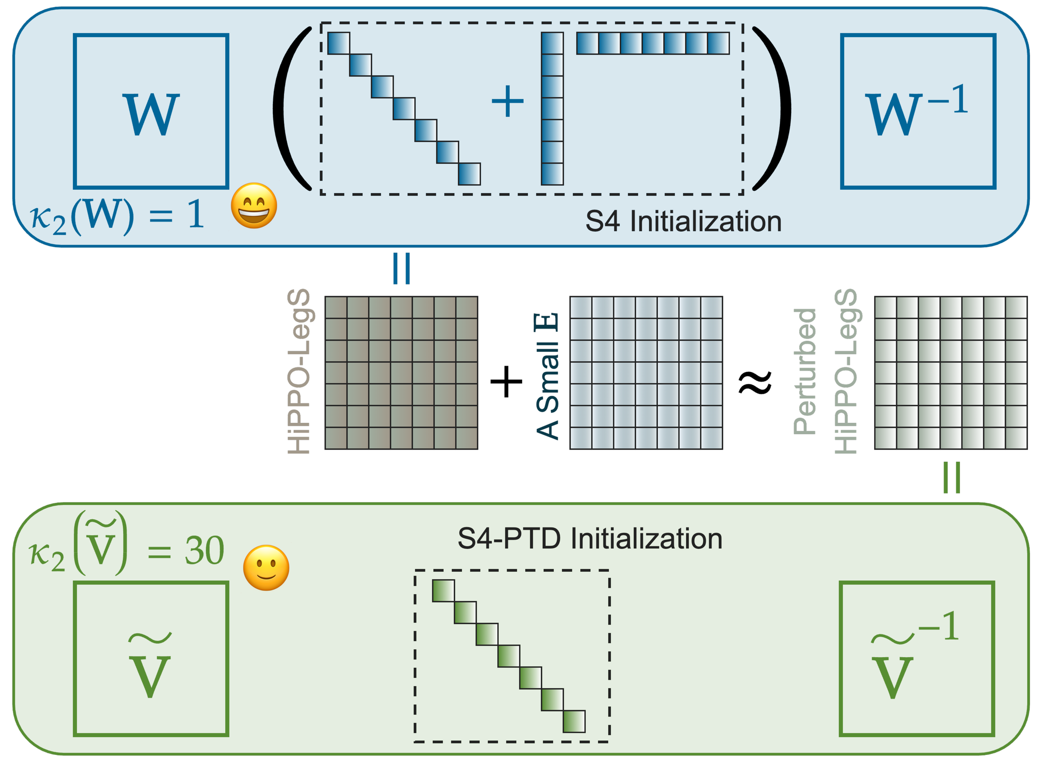


Moreover, since G_{S4D} is not smooth, a small input perturbation could cause a large change in its output, making the S4D initialization not robust.

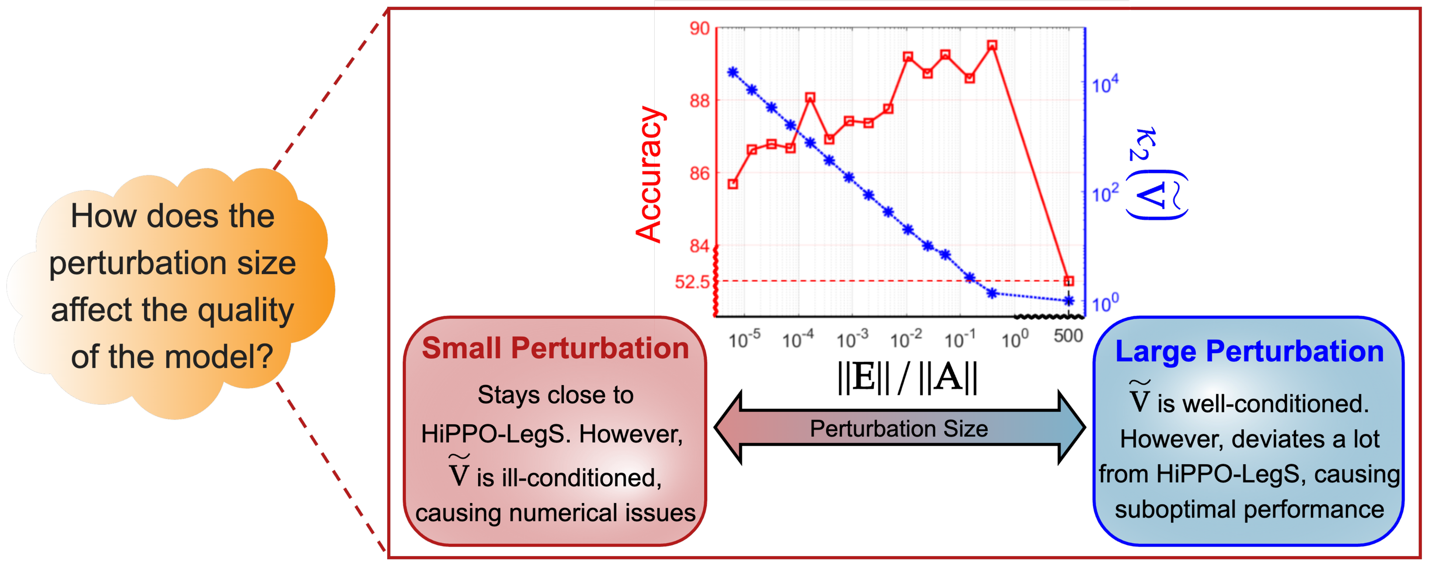


Approximate Diagonalization

To make the initialization robust, we propose to approximately diagonalize the HiPPO-LegS matrix $\mathbf{A}$. Our strategy is called **perturb-then-diagonalize (PTD)**. That is, we perturb the matrix $\mathbf{A}$ to $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{E}$ and use $\tilde{\mathbf{A}}$ to initialize the LTI systems. The eigenvector matrix of $\tilde{\mathbf{A}}$ is well-conditioned.



The size of the perturbation $\mathbf{E}$ is a hyperparameter. One can use it to balance the performance of the model and its numerical stability.



PTD models in the Long-Range Arena

Our PTD models demonstrate good performances in the Long-Range Arena.

Model	ListOps	Text	Retrieval	Image	Pathfinder	Path-X	Average
S4	59.60	86.82	90.90	88.65	94.20	96.35	86.09
Liquid-S4	**62.75**	89.02	91.20	**89.50**	94.80	96.66	87.32
S4D	60.47	86.18	89.46	88.19	93.06	91.95	84.89
S4-PTD (ours)	60.65	88.32	91.07	88.27	94.79	96.39	86.58
S5	62.15	89.31	91.40	88.00	95.33	**98.58**	87.46
S5-PTD (ours)	**62.75**	**89.41**	**91.51**	87.92	**95.54**	98.52	**87.61**