SSM Adapters via Hankel Reduced-order Modeling

Injection Site Determines Task Suitability in Long-Context Fine-Tuning

Omanshu Thapliyal, PhD.
HITACHI
Sr. AI / ML Researcher,
Strategic Data Solutions Lab,
Hitachi America Ltd. Santa Clara, CA USA

4th Workshop on High-dimensional Learning Dynamics (HiLD)
ICML International Conference On Machine Learning

Hankel Reduced-order Model: Turns a frozen transformer to a temporally aware adapter

Fast as LoRA, better on long-context, stateful tasks

1 Motivation

Missing Temporal Memory in PEFT

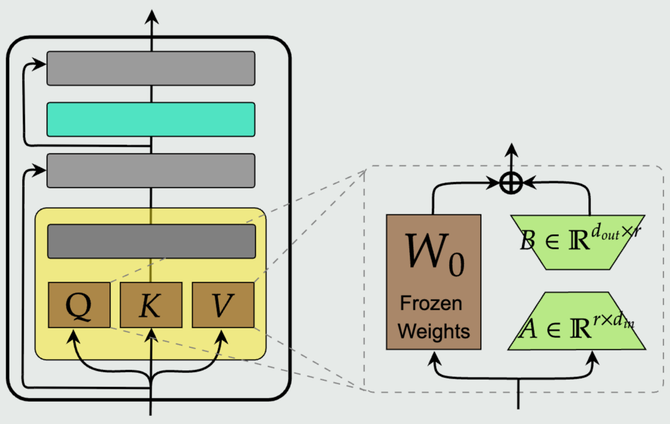
PEFT offers cheap adaptation by freezing the base model and learning few new parameters. However, standard methods like LoRA transform tokens independently, lacking an explicit cross-position temporal state.

Standard LoRA family methods parameterize updates as a static low-rank matrix product:

$$h_t = W_0 x_t + BAx_t = (W_0 + BA)x_t$$

Tasks such as DFA state tracking, long-context reasoning, meeting summarization require sequential state accumulation, not just better token-wise reweighting.

FIg. 1: LoRA modifies weight matrices; its output at position t is a static function of tokent at t



2 Background

Hankel Rank Reduction

A State space view: Model the adapter as a discrete-time linear state-space system (hidden state evolves over time under linear dynamics). This gives the adapter a mechanism for integrating information across the sequence rather than reacting to each token in isolation.

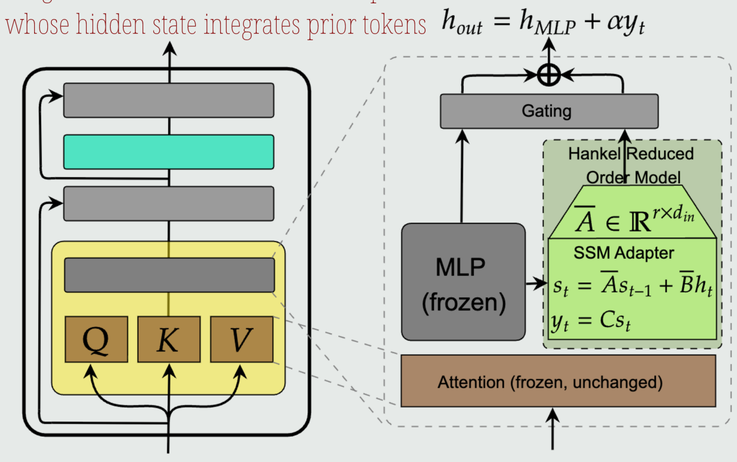
Hankel singular values identify which state dimensions are important for the input-output dynamics, making model reduction principled.

$x_t = \bar{A}x_{t-1} + \bar{B}u_t,\ y_t = Cx_t$ SSM hidden state dynamics

$W_c = \sum_{k=0}^{\infty} \bar{A}^k \bar{B}\bar{B}^T (\bar{A})^k$ Controllability Grammian: how easily a state can be reached from the input

$W_o = \sum_{k=0}^{\infty} (\bar{A}^T)^k C^T C \bar{A}^k$ Obversability Grammian: how well a state can be inferred from the output

FIg. 2: HRM inserts a recurrent adapter whose hidden state integrates prior tokens $h_{out} = h_{MLP} + \alpha y_t$



3 Problem

Why MLP Injection, and not Attention?

Where should a recurrent PEFT adapter should be inserted inside a frozen transformer: attention, or MLP?

Adding to attention: Attention retrieves context; MLP transforms tokens independently, HRM is parallel to the MLP to add temporal integration without disturbing frozen attention.

To compare HRM and LoRA fairly, we first matche trainable parameter counts iso-parametrically. **Equal parameter budget ≠ equal temporal capability**: LoRA remains a static per-token update; HRM adds recurrent memory.

LoRA r	P_{LoRA}	HRM $\hat{d}$	P_{HRM}	Δ (%)	Regime
8	16,384	16	16,156	+0.81	Low capacity
16	32,768	32	33,028	+0.79	Mid capacity
32	65,536	63	65,020	−0.79	High capacity

∵ HRM uses LTI transition matrices, the recurrence can be evaluated as a causal convolution and **accelerated with FFT-based parallel scan**, giving adapter a LoRA-like wall-clock efficiency

Context Length T	LoRA Wall clock time/epoch (s)	HRM-seq. Wall clock time/epoch	HRM-seq. / LoRA	HRM-FFT Wall clock time/epoch	HRM-FFT / LoRA
512	9	~90	10×	9	1×
1024	70	~980	14×	70	1×
2048	265	~3710	14×	265	1×

4 Solution

Balanced Truncation via Hankel Singular Values

The HRM adapter is a recurrent residual branch whose state evolves with a time-invariant transition matrix and is added back into the frozen transformer (Fig. 2). Since the dynamics are LTI, the model can be compressed via balanced truncation with explicit bounds.

$s_t^{(l)} = \bar{A}s_{t-1}^{(l)} + \bar{B}^{(l)} h_t^{MLP}$ SSM recurrence and readout; defines the recurrent hidden state at layer l

$y_t^{(l)} = C^{(l)} s_t^{(l)}$

$y_t^{(l)} = C^{(l)} \sum_{k=0}^{t} \left(\bar{A}^{(l)}\right)^{t-k} \bar{B}^{(l)} h_k^{MLP,(l)}$ adapter output as a causal sum

$\sigma_i = \sqrt{\lambda_i(W_c W_o)}$ Hankel singular values

$TW_cT^T = T^{-T}W_oT^{-1} = \Sigma = \text{diag}(\sigma_1, \cdots, \sigma_d)$ Balancing transform: simultaneously diagonalizes Grammians

$\hat{A} = \left(T_{[1:\hat{d},1:\hat{d}]} \bar{A} T^{-1}_{[1:\hat{d},:]}\right)_{[1:\hat{d},1:\hat{d}]}$, $\|G - \hat{G}\|_{\mathcal{H}_\infty} \leq 2\sum_{k=\hat{d}+1}^{d} \sigma_k$

$\hat{B} = \left(T\bar{B}\right)_{[1:\hat{d},:]}, \hat{C} = \left(CT^{-1}\right)_{[:,1:\hat{d}]}$

Reduced order System — Glover bound

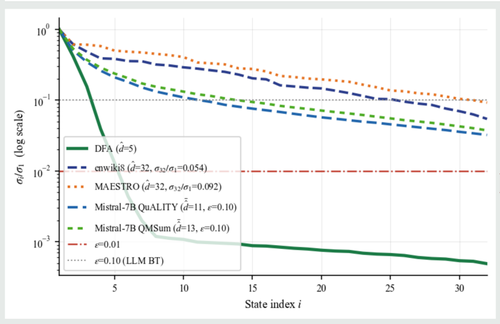


Fig. 4: HSV decay for: DFA, enwik8, MAESTRO, QuALITY, and QMSum.

5 Experiments

HRM on Stateful Long-Context Tasks

We evaluate HRM on 3 task families that all require temporal accumulation: synthetic state tracking, character-level language modeling, and long-context downstream reasoning.

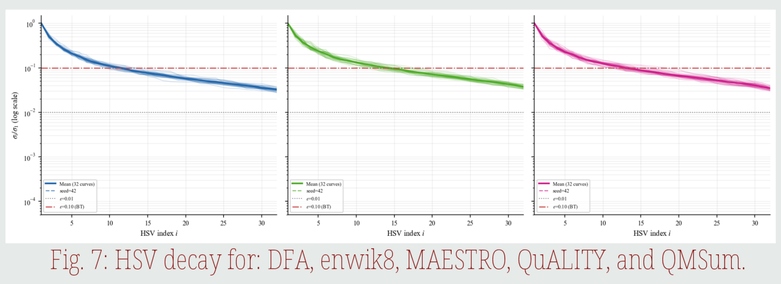


Fig. 7: HSV decay for: DFA, enwik8, MAESTRO, QuALITY, and QMSum.

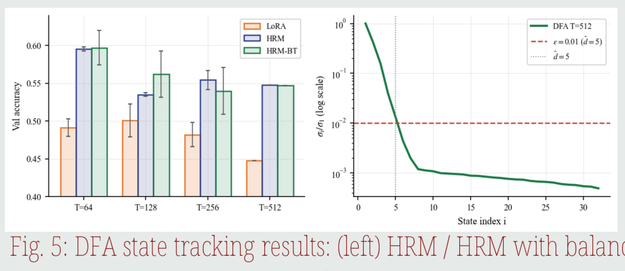


Fig. 5: DFA state tracking results: (left) HRM / HRM with balanced truncation vs. LoRA, (right) HSV decay rate for the task (cutoff=0.01)

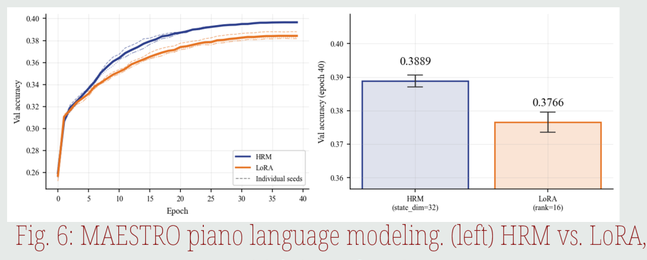


Fig. 6: MAESTRO piano language modeling. (left) HRM vs. LoRA, (right) Final BPC.

HRM achieves lower BPC than LoRA on enwik8 (stateful long-context task)

Adapter Capacity	Context length	LoRA BPC	HRM BPC	ΔBPC
$r = 18, \hat{d} = 16$	512	3.3006	3.1902	0.1104
	1024	3.0271	2.9831	0.0440
	2048	3.0039	2.9026	0.1013
$r = 16, \hat{d} = 32$	512	3.2594	3.1424	0.1170
	1024	3.0326	2.9464	0.0862
	2048	2.9911	2.8859	0.1052
$r = 32, \hat{d} = 63$	512	3.1789	3.1470	0.0319
	1024	3.0186	2.9413	0.0773
	2048	2.9906	2.8789	0.1117

6 Conclusion

Where HRM Helps and where not

HRM consistently improves over LoRA-family baselines on tasks demanding stateful integration, esp. DFA tracking, MAESTRO, enwik8, QuALITY, QMSum; Strongest long-context gains: QuALITY & QMSum; NarrativeQA is the main failure case under the truncated-context setup.

Method	QuALITY (Acc)	QMSum (R1/R2/R)	NarrativeQA (F1)
LoRA	0.3518	0.1477 / 0.0133 / 0.0925	0.1592
AdaLoRA	0.3518	0.1477 / 0.0133 / 0.0925	0.1571
DoRA	0.3478	0.1458 / 0.0133 / 0.0916	0.1569
QLoRA	0.3399	0.1641 / 0.0140 / 0.1038	0.1492
HRM	0.4743	**0.2531 / 0.0339 / 0.1180**	0.1492
HRM vs. best baseline	*+0.1225 (+34.8%)*	*+0.0890 R-1 (+54.3% vs LoRA)*	*-0.1201 (-75.4%)*

- Temporal memory matters when the downstream problem is not just retrieval
- ⇒ HRM is strongest on tasks that need long-range state accumulation & causal memory (DFA tracking, MAESTRO, QuALITY, QMSum)
- HRM is less effective when the task is mainly retrieval-based or not long-context.

CODE

PAPER

CONTACT

omanshu.thapliyal@hal.hitachi.com
https://omanshuthapliyal.github.io/