Randomized Gradient Subspace Exploration for Efficient LLM Training

Sahar Rajabi, Nayeema Nonta, Sirisha Rambhatla
University of Waterloo, Waterloo, Ontario, Canada

Overview

We investigate why random-projection methods work for efficient LLM training compared to structured approaches. We find that the gradient subspace has a nearly **flat curvature** and that a non-negligible portion of gradient energy lies outside the core low-rank subspace, meaning the exact subspace identification matters less than previously thought. Building on this insight, we propose **GrassWalk** and **GrassJump**, which update projections via random walks and jumps on the Grassmannian, while adapting optimizer states to subspace changes, achieving SOTA results, and reframing **randomization as a geometrically principled strategy rather than merely a computational shortcut.**

1. Motivation

- Training LLMs is bottlenecked by memory, with optimizer states dominating the footprint; motivating low-rank gradient methods that preserve full-parameter updates.
- Existing approaches split into two camps: **structured** methods (SVD, subspace tracking) and **randomized** methods.
- Randomized projections match or exceed structured ones, yet we have no principled understanding of the reason.
- Our analysis reveals two key geometric facts: **the core gradient subspace evolves with nearly flat curvature**, and a **non-negligible share of gradient energy lies outside it**; growing larger as training progresses and in deeper layers.
- This reframes randomization from a computational shortcut into a geometrically principled strategy, and points toward algorithms that explicitly navigate the subspace manifold while recovering the orthogonal gradient signal.

2. Prior Works

- GaLore [1] & FiRA [2] project gradients into a low-rank subspace identified via periodic SVD; accurate but computationally heavy and unstable under noisy gradients.
- SubTrack++ [3], LDAdam [4], and Online Subspace Descent [5] iteratively estimate dominant subspaces to avoid repeated SVDs; while APOLLO [6], FRUGAL [7], GaRare [8], RSO [9] are randomized cheap alternatives to SVD that beats GaLore but still lag behind SOTA optimizers.
- COAP [10] aligns subspace updates with the first momentum; FRUGAL [7] resets or projects states; LDAdam [4] and SubTrack++ [3] explicitly handles Adam's nonlinearity under subspace changes.
- Gap: randomization has been used purely for efficiency; no prior work explains why it succeeds and achieving SOTA results.

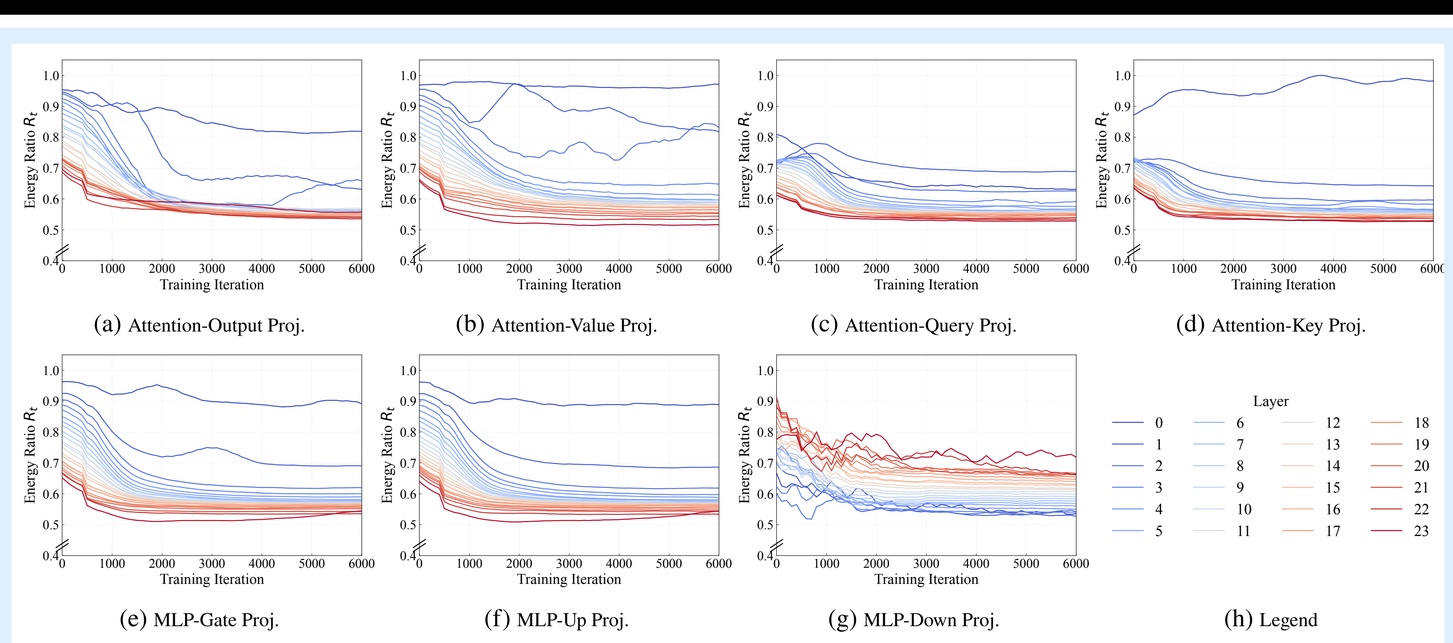
3. Beyond the Core Subspace

Gradient Energy Fraction

$$R_t = \frac{|\tilde{G}_t|_F}{|G_t|_F}$$

where $G_t \in \mathbb{R}^{m\times n}$ is the full-rank gradient and $\tilde{G}_t = S_t^\top G_t$ is its low-rank projection onto the subspace.

- **A dominant low-rank subspace exists:** across all layers, >50% of gradient energy lies in the rank-512 core. But the captured fraction decays as early in training.
- **Non-trivial orthogonal energy:** Even at rank 1024, energy stabilizes at only 70–80%, leaving 20–30% of the signal orthogonal to the core.



(a) Attention-Output Proj. (b) Attention-Value Proj. (c) Attention-Query Proj. (d) Attention-Key Proj.
(e) MLP-Gate Proj. (f) MLP-Up Proj. (g) MLP-Down Proj. (h) Legend

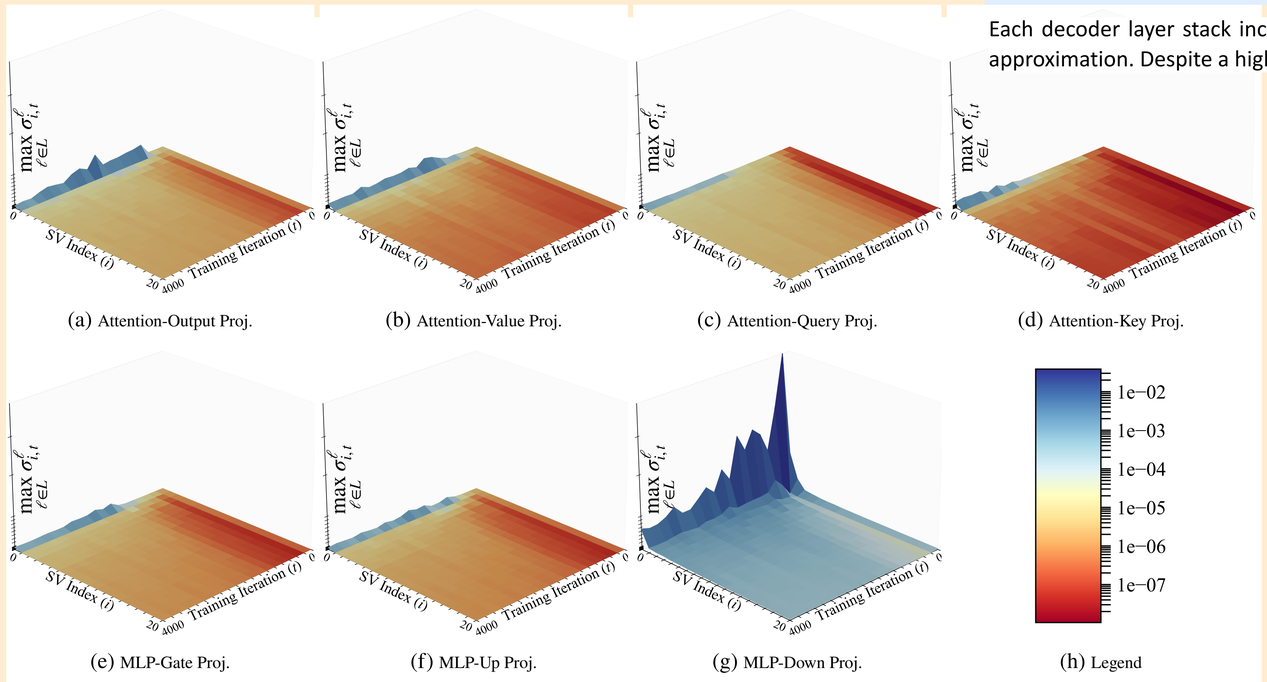
Subspace Optimization Landscape

- We take the derivative of the subspace-estimation loss w.r.t. the current projection.

$$F(S_t) = \min_A |S_t A - G_t|_F^2 \text{ (subspace estimation loss)}$$

$$\text{And } \frac{\partial F(S_t)}{\partial S_t} = -2(S_t A - G_t)A^\top$$

- Top singular values are extremely small and decay rapidly, demonstrating that the landscape is nearly flat.
- **Takeaway:** a flat landscape means random exploration is geometrically principled, not just a computational shortcut, and can help in optimization as a regularizer.



(a) Attention-Output Proj. (b) Attention-Value Proj. (c) Attention-Query Proj. (d) Attention-Key Proj.
(e) MLP-Gate Proj. (f) MLP-Up Proj. (g) MLP-Down Proj. (h) Legend

Evolution of the top 20 singular values of the subspace estimation error derivative across different projection layers in the LLaMA-1B architecture. Each plot shows the maximum i-th singular value within a given layer type (aggregated across all 24 decoder layers) as training progresses. The overall distribution of singular values becomes more uniform as training advances, suggesting that the gradient subspace evolves in an almost flat curvature.

Each decoder layer stack includes seven layer types in the Llama-1B model. The plots show the fraction of gradient energy explained by a rank 512 approximation. Despite a high lower bound, this fraction declines over training, and deeper layers generally exhibit smaller fractions.

Systematic Ablation

- Compares four subspace-update rules: Grassmannian subspace tracking, random walk on Grassmannian, random projections, SVD, with/without Adaptive Optimizer (AO), and Recovery Scaling (RS).
- In the baseline (no AO/RS) structured manifold updates outperform SVD.
- Adding AO helps most methods, but random projections benefit most from RS; because they may discard salient signal, recovering the orthogonal component is essential.
- **Conclusion:** pairing randomized Grassmannian exploration with AO + RS delivers the best of both worlds; flat-landscape exploration plus orthogonal-signal recovery.

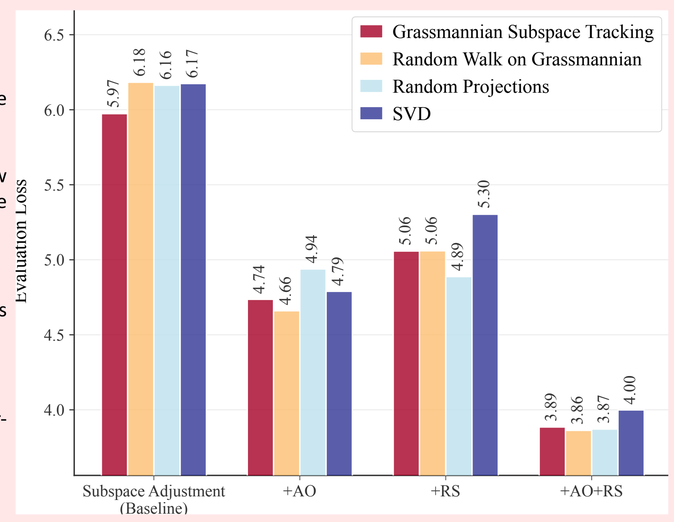
Adaptive Optimizer (AO)
When subspace changes, Adam's moments become misaligned with the new basis.

We rotate the first and second moments into the new subspace so optimizer history stays consistent with the projected gradients.

Recovery Scaling (RS)
Low-rank projection discards the residual which carries significant energy.

$$\Delta_t = G_t - S_t S_t^\top G_t$$

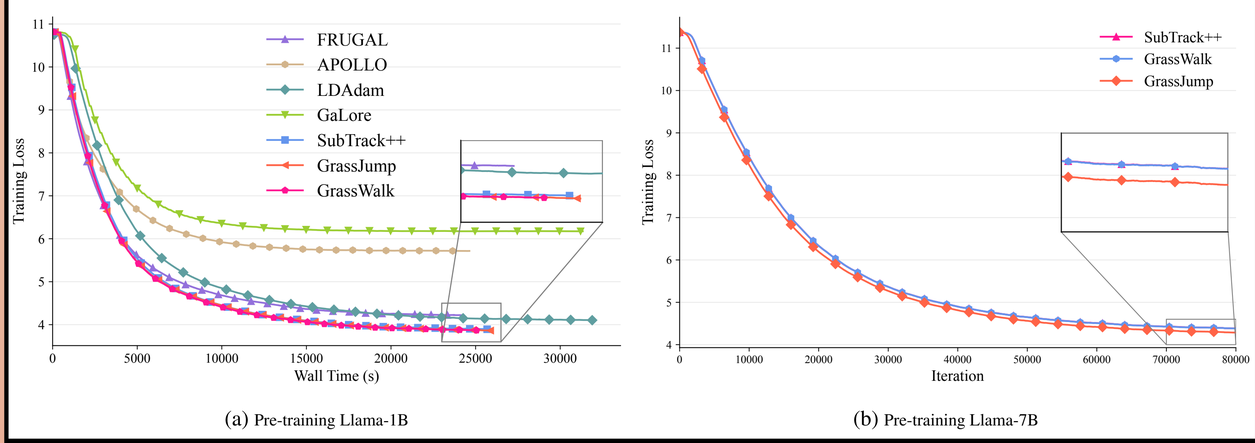
We reintroduce Δ_t via recovery scaling, scaled per-column by the optimizer's output magnitude.



4. GrassWalk and GrassJump

Algorithm 1
GrassWalk and **GrassJump**

Require: $W_t, G_t \in \mathbb{R}^{m\times n}$ with $m \le n$ (w.l.o.g.), learning rate α, decay rates β_1 and β_2, subspace update interval T, recovery scaling limiter factor ζ.
GrassWalk: $S_0 \leftarrow U[:, :r]$, where $U, S, V \leftarrow \text{SVD}(G_0)$ {initialing the subspace}
GrassJump: $S_0 \leftarrow$ random gaussian matrix
while Convergence **do**
  if $step$ mod $T == 0$ **then**
    GrassWalk: $S_t(\tau) = S_{t-1}\widehat{V}_X \cos(\widehat{\Sigma}_X\tau)\widehat{V}_X^\top + \widehat{U}_X \sin(\widehat{\Sigma}_X\tau)\widehat{V}_X^\top + S_{t-1}(I - \widehat{V}_X\widehat{V}_X^\top)$, {updating the subspace}
    where $\widehat{U}_X, \widehat{\Sigma}_X, \widehat{V}_X^\top \leftarrow$ SVD(random gaussian matrix)
    GrassJump: $S_t \leftarrow$ random gaussian matrix
    $\widetilde{G}_t = S_t^\top G_t$ {calculate low-rank gradient}
    $M_t \leftarrow \beta_1(S_t^\top S_{t-1}M_{t-1}) + (1-\beta_1)\widetilde{G}_t$ {toggle adaptive optimizer}
    $V_t \leftarrow \beta_2\left[(1-\beta_2^{t-1})|(S_t^\top S_{t-1})^2 \cdot (V_{t-1} - M_{t-1}^2) + (S_t^\top S_{t-1} \cdot M_{t-1})^2|\right] + (1-\beta_2)\widetilde{G}_t^2$.
  else
    $S_t = S_{t-1}$
    $M_t \leftarrow \beta_1 \cdot M_{t-1} + (1-\beta_1)\cdot \widetilde{G}_t$ {regular optimizer}
    $\mathcal{V}_t \leftarrow \beta_2 \cdot \mathcal{V}_{t-1} + (1-\beta_2)\cdot \widetilde{G}_t^2$
  end if
  $\widetilde{G}_t^O \leftarrow$ optimizer's output, $\widehat{G}_t = S_t\widetilde{G}_t^O$
  $\phi_t(G_t)_i = \frac{\|\widetilde{G}^O_{t,:,i}\|}{\|\widetilde{G}_{t,:,i}\|}$, $\Lambda_t = \phi_t(G_t)\,\Delta_t$, {recovery scaling}
  $\Lambda_t \leftarrow \Lambda_t \cdot \frac{\zeta\|\Lambda_{t-1}\|}{\|\Lambda_t\|}$
  $W_t \leftarrow W_{t-1} - \alpha\cdot\widehat{G}_t - \alpha\cdot\Lambda_t$ {weight update rule}
end while



(a) Pre-training Llama-1B (b) Pre-training Llama-7B

5. Results

Table 1. Comparison of low-rank methods on pretraining LLaMA-1B and Qwen 1.5B models. We report evaluation loss (↓), peak memory (GB), and wall-time (m). Best results are in **bold**, and second best are underlined.

Arch.	Method	Eval. Loss	Peak Mem. (GB)	Wall Time (m)
LLaMA-1B	AdamW [Full-Rank]	4.10	35.2	417.0
	GaLore (Zhao et al., 2024a)	6.17	31.1	522.2
	APOLLO (Zhu et al., 2025)	5.71	35.5	410.5
	LDAdam (Robert et al., 2025)	4.10	34.9	532.8
	FRUGAL (Zmushko et al., 2025)	4.22	39.3	405.1
	SubTrack++ (Rajabi et al., 2025)	3.89	32.6	429.2
	GrassWalk [Ours]	**3.86**	32.0	418.6
	GrassJump [Ours]	<u>3.87</u>	32.1	415.2
Qwen-1.5B	AdamW [Full-Rank]	4.84	37.7	421.0
	SubTrack++ (Rajabi et al., 2025)	4.70	33.1	436.4
	GrassWalk [Ours]	<u>4.68</u>	33.6	436.6
	GrassJump [Ours]	**4.67**	33.1	432.2

Table 2. Comparison of low-rank gradient methods for pretraining LLaMA-7B. We report evaluation loss (↓), peak memory usage (GB), and wall-clock time (hours) after 10k and 100k of training iterations. Other baselines are omitted as their performance differs substantially from these three methods. Best results are in **bold**.

Method	Eval. Loss - 10k (Wall Time)	Eval. Loss - 100k (Wall Time)	Peak Mem. (GB)
SubTrack++ (Rajabi et al., 2025)	4.94 (9.57 hours)	3.36 (93.2 hours)	50.3
GrassWalk [Ours]	4.94 (9.55 hours)	3.37 (93.2 hours)	50.3
GrassJump [Ours]	**4.86** (9.24 hours)	**3.34** (91.6 hours)	48.7

- **The gradient subspace optimization landscape has nearly flat curvature with non-negligible orthogonal energy**; making randomization a geometrically principled strategy, not just a shortcut.
- **GrassWalk & GrassJump** exploit this via random walks/jumps on the Grassmannian, paired with subspace-aware optimizer updates and recovery of the lost signal. They achieve state-of-the-art on LLaMA-1B, LLaMA-7B, and Qwen-1.5B pretraining with random projections.

Acknowledgement

Sirisha Rambhatla would like to acknowledge support of the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant, RGPIN-2022-03512.

References

[1] Zhao, J., Zhang, Z., Chen, B., Wang, Z., Anandkumar, A., & Tian, Y. (2024). GaLore: Memory-Efficient LLM Training by Gradient Low-Rank Projection. Forty-First International Conference on Machine Learning (ICML 2024).
[2] Chen, X., Fent, K., Li, C., Lai, X., Yue, X., Yuan, Y., & Wang, G. (2025). Fira: Can We Achieve Full-rank Training of LLMs under Low-rank Constraint? Thirty-Ninth Conference on Neural Information Processing Systems (NeurIPS 2025).
[3] Rajabi, S., Nonta, N., & Rambhatla, S. (2025). SubTrack++: Gradient Subspace Tracking for Scalable LLM Training. Thirty-Ninth Conference on Neural Information Processing Systems (NeurIPS 2025).
[4] Robert, T., Safaryan, M., Modoranu, & I., Alistarh, D. (2025). LDAdam: Adaptive Optimization from Low- Dimensional Gradient Statistics. Thirty-Ninth International Conference on Learning Representations (ICLR 2025).
[5] Liang, K., Liu, B., Chen, L., & Liu, Q. (2024). Memory-Efficient LLM Training with Online Subspace Descent. Thirty-Eighth Conference on Neural Information Processing Systems (NeurIPS 2024).
[6] Zhu, H., Zhang, Z., Cong, W., Liu, X., Park, S., Chandra, V., Long, B., Pan, D., Wang, Z., & Lee, J. (2025). APOLLO: SGD-Like Memory, AdamW-Level Performance. Eighth MLSys Conference.
[7] Zmushko, P., Beznosikov, A., Takáč, M., & Horváth, S. (2025). FRUGAL: Memory-Efficient Optimization by Reducing State Overhead for Scalable Training. Forty-second International Conference on Machine Learning (ICML 2025).
[8] Liu, X.-H., Du, Y., Wang, J., & Yu, Y. (2025). On the Optimization Landscape of Low Rank Adaptation Methods for Large Language Models. The Thirteenth International Conference on Learning Representations (ICLR 2025).
[9] Chen, Y., Zhang, Y., Liu, Y., Yuan, K., & Wen, Z. (2025). A Memory Efficient Randomized Subspace Optimization Method for Training Large Language Models. Forty-second International Conference on Machine Learning (ICML 2025).
[10] Xiao, J., Sang, S., Zhi, T., Liu, J., Yan, Q., Zhang, Y., Luo, L., & Yuan, B. (2025). COAP: Memory-Efficient Training with Correlation-Aware Gradient Projection.