SubTrack++ : Gradient Subspace Tracking for Scalable LLM Training

Sahar Rajabi, Nayeema Nonta, Sirisha Rambhatla

Critical ML, Department of Management Science and Engineering, University of Waterloo

NeurIPS 2025

Problem Space & Motivation

Problem. Large Language Model (LLM) training requires massive **compute**, **memory**, & **time**.

This limits accessibility to large-scale research, and amplifies the environmental footprint of modern AI systems.

Existing solutions & gaps. Methods like GaLore [Zhao et al., 2024] reduce memory usage by projecting gradients into a low-rank subspace.

GaLore's **periodic SVD updates** introduces (1) high computational cost, (2) sensitivity to noise, and (3) instability in later training stages.

Our motivation. *Can we maintain the benefits of low-rank training without expensive decomposition or instability?*

SubTrack++: Geometry- and Projection-Aware Gradient Subspace Tracking

SubTrack++ replaces costly SVD-based gradient projections with an **efficient, geometry-aware subspace tracking** approach.

Core components.

Grassmannian Subspace tracking

Tracks gradient subspaces on a manifold via smooth geodesic updates.

Projection-aware optimization

Re-projects Adam's moments to stay aligned with evolving subspaces.

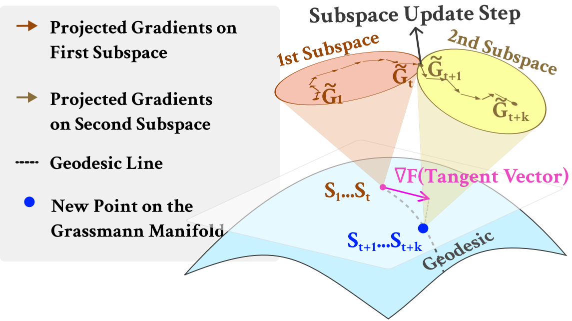
Gradient recovery scaling

Recovers & rescales gradient components lost in low-rank projection.

Our contribution.
SubTrack++ combines geometric subspace tracking, optimizer alignment, and recovery scaling for stable, time- and memory-efficient full-parameter training.

SubTrack++ Components

Grassmannian Subspace Tracking



Visualization of Grassmannian subspace tracking: Between subspace updates, gradients are projected onto a fixed subspace. The tangent vector ∇F is computed via the derivative of a loss function, measuring the subspace estimation error. The subspace is then updated by moving along the corresponding geodesic, determined by ∇F to minimize estimation error.

Initialization:

$$G_0 = USV^\top \approx \sum_{i=1}^{r} s_i u_i v_i^\top$$

Initial subspace $S_0 = [u_1, \ldots, u_r]$ spans the top-r directions.

Adjusting the subspace:

Minimizing subspace estimation error by moving along Grassmannian manifold.

Projection-Aware Optimizer

Low-rank training stores optimizer states in a reduced subspace. As this subspace shifts, standard Adam misaligns its moments. SubTrack++, inspired by LDAdam [Robert et al., 2025], corrects this via a **projection-aware optimizer**.

Standard Adam (in projected space):

$$M_t \leftarrow \beta_1 \cdot M_{t-1} + (1-\beta_1) \cdot \widetilde{G}_t, \qquad \mathcal{V}_t \leftarrow \beta_2 \cdot \mathcal{V}_{t-1} + (1-\beta_2) \cdot \widetilde{G}_t^2$$

where $\widetilde{G}_t = S_t^\top G_t$ is the projected gradient.

Projection-Aware Adam:

$$M_t \leftarrow \beta_1 \cdot (S_t^\top S_{t-1} M_{t-1}) + (1-\beta_1) \cdot \widetilde{G}_t,$$

$$\mathcal{V}_t \leftarrow \beta_2 \cdot [(1-\beta_2^{t-1})|(S_t^\top S_{t-1})^2 \cdot (\mathcal{V}_{t-1} - M_{t-1}^2) + (S_t^\top S_{t-1} \cdot M_{t-1})^2|] + (1-\beta_2) \cdot \widetilde{G}_t^2$$

Recovery Scaling

Low-rank projection removes gradient components that still carry useful signal. Inspired by Chen et al. [2025] and Zhu et al. [2025], we **recover and rescale** these components to preserve information.

Projected optimizer output:

$$\widetilde{G}_t^O = M_t \oslash \sqrt{\mathcal{V}_t + \epsilon}, \quad \widehat{G}_t = S_t \widetilde{G}_t^O$$

Recovery-scaled update:

$$W_t \leftarrow W_{t-1} - \alpha \widehat{G}_t - \alpha\, \phi_t(G_t)(G_t - S_t \widetilde{G}_t), \quad \phi_t(G_t)_i = \frac{\|\widetilde{G}^O_{t,:,i}\|}{\|\widetilde{G}_{t,:,i}\|}$$

Stability (gradient clipping):

$$\Lambda_t = \phi_t(G_t)(G_t - S_t \widetilde{G}_t), \quad \Lambda_t \leftarrow \frac{\Lambda_t}{\|\Lambda_t\|} \zeta \|\Lambda_{t-1}\|$$

SubTrack++ Algorithm

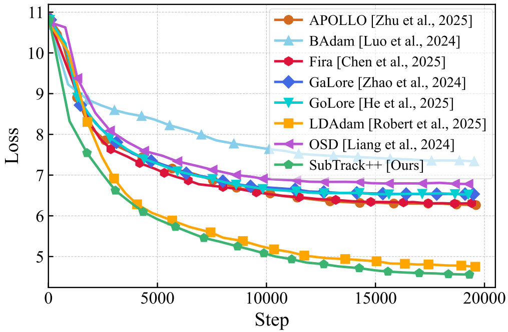
Algorithm 1 SubTrack++ (Subspace Tracking , Projection-Aware Optimizer , Recovery Scaling , Regular Adam)

Require: $W_t, G_t \in \mathbb{R}^{m \times n}$, learning rate α, β_1, β_2, step-size η, rank r, interval k, limiter ζ. $\oslash$ denotes Hadamard division.

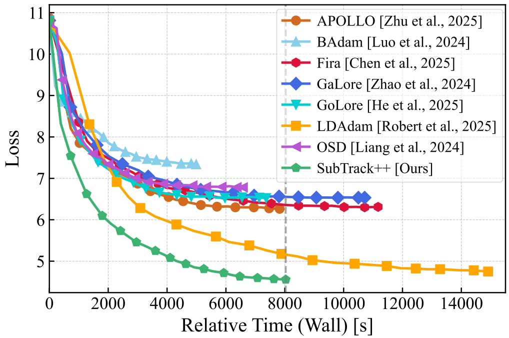
$S_0 \leftarrow U[:, :r]$, where $(U, S, V) \leftarrow \text{SVD}(G_0)$
for $t = 0, \ldots, T$ **do**
  if $t \bmod k = 0$ **then**
    $R = G_t - S_{t-1}G_{lr}$, $\nabla F = -2RG_{lr}^\top \approx \widehat{U}_F \widehat{\Sigma}_F \widehat{V}_F^\top$
    $S_t = (S_{t-1}\widehat{V}_F \ \ \widehat{U}_F)\begin{pmatrix} \cos(\widehat{\Sigma}_F\eta) \\ -\sin(\widehat{\Sigma}_F\eta) \end{pmatrix}\widehat{V}_F^\top + S_{t-1}(I - \widehat{V}_F\widehat{V}_F^\top)$
    $M_t \leftarrow \beta_1(S_t^\top S_{t-1}M_{t-1}) + (1-\beta_1)\widetilde{G}_t$
    $\mathcal{V}_t \leftarrow \beta_2(S_t^\top S_{t-1})^2\mathcal{V}_{t-1} + (1-\beta_2)\widetilde{G}_t^2$
  else
    $M_t \leftarrow \beta_1 M_{t-1} + (1-\beta_1)\widetilde{G}_t$
    $\mathcal{V}_t \leftarrow \beta_2\mathcal{V}_{t-1} + (1-\beta_2)\widetilde{G}_t^2$
  end if
  $\widetilde{G}_t^O = M_t \oslash \sqrt{\mathcal{V}_t + \epsilon}$, $\widehat{G}_t = S_t\widetilde{G}_t^O$
  $\phi_t(G_t)_i = \frac{\|\widetilde{G}^O_{t,:,i}\|}{\|\widetilde{G}_{t,:,i}\|}$, $\Lambda_t = \phi_t(G_t)(G_t - S_t\widetilde{G}_t)$
  if $\frac{\|\Lambda_t\|}{\|\Lambda_{t-1}\|} > \zeta$ then $\Lambda_t \leftarrow \frac{\Lambda_t}{\|\Lambda_t\|}\zeta\|\Lambda_{t-1}\|$
  $W_t \leftarrow W_{t-1} - \alpha\widehat{G}_t$ $-\alpha\Lambda_t$
end for

Experimental Results

Main Results: Pre-Training Llama-1B on C4 Dataset



(a) Loss vs. Training Steps.



(b) Loss vs. Relative Wall-Time.

Comparison of baselines in pre-training Llama-1B architecture. (a) shows training loss (↓) versus training steps. (b) shows the same runs against wall-time. SubTrack++ outperforms all baselines; substantially reducing wall-time, especially compared to LDAdam, the top-performing baseline.

Main Results: Pre-Training Llama-1B on C4 Dataset

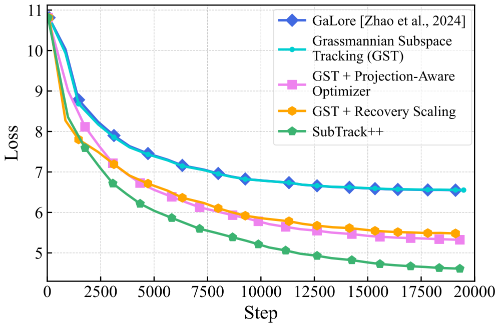
Baselines. GaLore [Zhao et al., 2024], Fira [Chen et al., 2025], LDAdam [Robert et al., 2025], BAdam [Luo et al., 2024], APOLLO [Zhu et al., 2025], OSD [Liang et al., 2024].

Key findings:

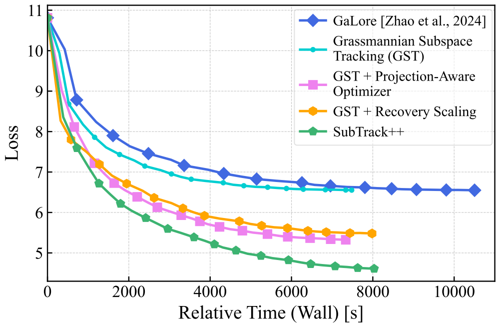
- SubTrack++ achieves **fastest convergence** in both step-wise and wall-time metrics.
- **43% faster** than LDAdam on 1B model; **67% faster** on 3B model.
- Matches or exceeds full-rank training performance in some settings, due to implicit regularization.
- Gradient and projected-gradient norms follow nearly identical decay ($0.46 \to 0.08$ vs. $0.45 \to 0.05$), confirming stable convergence.

Key takeaway. Geometry-aware subspace tracking and projection-aligned optimization yield superior efficiency without compromising performance.

Ablation Study: SubTrack++ Component Contributions



(a) Loss vs. Training Steps.



(b) Loss vs. Relative Wall-Time.

Ablation study comparing pure Grassmannian subspace tracking with incremental additions of the projection-aware optimizer and recovery scaling, leading to SubTrack++ . While Grassmannian tracking alone almost matches GaLore's step-wise convergence (a), it significantly reduces wall-time (b).

Discussion and Conclusion

- **SubTrack++ :** a time- and memory-efficient method projecting gradients into a low-rank subspace.
- **Grassmannian subspace tracking:** preserves the learned subspace while incorporating gradient components from the orthogonal complement.
- **Projection-aware optimizers:** adapt Adam's internal statistics to reflect subspace changes.
- **Recovered gradient information:** leverages components lost during low-rank projection to enhance performance.
- **Results:** achieves state-of-the-art convergence and accuracy across baselines.
- **Compatibility:** functions as a plug-and-play module with standard optimizers.

References

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection, 2024. URL `https://arxiv.org/abs/2403.03507`.

Thomas Robert, Mher Safaryan, Ionut-Vlad Modoranu, and Dan Alistarh. LDAdam: Adaptive optimization from low-dimensional gradient statistics. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=Zkp1GuHerF`.

Xi Chen, Kaituo Feng, Changsheng Li, Xunhao Lai, Xiangyu Yue, Ye Yuan, and Guoren Wang. Fira: Can we achieve full-rank training of LLMs under low-rank constraint?, 2025. URL `https://openreview.net/forum?id=lR7rqLtsXZ`.

Hanqing Zhu, Zhenyu Zhang, Wenyan Cong, Xi Liu, Sem Park, Vikas Chandra, Bo Long, David Z. Pan, Zhangyang Wang, and Jinwon Lee. Apollo: Sgd-like memory, adamw-level performance, 2025. URL `https://arxiv.org/abs/2412.05270`.

Qijun Luo, Hengxu Yu, and Xiao Li. Badam: A memory efficient full parameter optimization method for large language models, 2024. URL `https://arxiv.org/abs/2404.02827`.

Kaizhao Liang, Bo Liu, Lizhang Chen, and qiang liu. Memory-efficient LLM training with online subspace descent. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=P8rTCT6g45`.

Thank You!

Email us at: srajabi@uwaterloo.ca