Understanding LoRA Update Complexity Through Stable Rank

Adapting large language models for new tasks typically requires full fine-tuning, which is costly and parameter-intensive. Low-Rank Adaptation (LoRA) [1] addresses this by freezing the base weights and injecting small, trainable low-rank adapters into selected weight matrices. It has proven highly effective across a range of tasks, but recent research has noted a fundamental issue with its update complexity, meaning the effective dimensionality of the learned updates. The nominal rank specified in LoRA does not reliably correspond to the dimensionality realized during training. More specifically, LoRA updates often have a much lower stable rank (defined formally below) than the nominal value [2], while other work has demonstrated that LoRA's default scaling rule can cause higher ranks to underperform [3]. Collectively, these findings reveal that the optimization dynamics that lead to this **"rank collapse"** are poorly understood. We lack a clear explanation for why LoRA, as rank increases, even up to the full dimension, often fails to converge toward the updates learned by full fine-tuning, despite its theoretical capacity to represent them. We hypothesize that this is a result of optimization difficulties.

We investigate this by directly measuring the update complexity of **LoRA**, alongside **full fine-tuning** for reference across RoBERTa-base and RoBERTa-large on 3 **GLUE** tasks. Our probe is the **stable rank**, $\mathrm{sr}(M) = \|M\|_F^2/\|M\|_2^2$, which provides a single interpretable measure of effective dimensionality, capturing how concentrated or spread out the update directions are. Intuitively, it compares the total energy of a matrix to the energy in its largest direction, so higher stable rank means updates are distributed across more independent directions rather than dominated by one.

For LoRA, we compute stable ranks of the pretrained weights W, the adapter updates $\Delta W = BA$, and the effective weights $W + \Delta W$; for full fine-tuning we compute stable ranks of W_{pre} and $W_{\text{FT}} = W_{\text{pre}} + \Delta W$, where ΔW is not rank-constrained.

To isolate the effect of rank, we sweep adapter rank from $r = 1$ up to the layer dimension d, spanning low-through full-rank settings, while holding initialization, optimizer, batch size, and training budget fixed to standard defaults as in [1]. We also sweep learning rates logarithmically from 10^{-5} to 10^{-2}, selecting the best rate by validation performance for each configuration. This setup allows us to systematically probe how LoRA's effective complexity evolves with rank and how it diverges from or aligns with the updates learned by full fine-tuning.

We observe two consistent patterns across tasks and models. First, the stable rank of the adapter updates (ΔW) grows much more slowly than the nominal rank, remaining confined to a narrow range even as r increases. This shows that LoRA's updates occupy a much smaller subspace than the nominal setting suggests. Second, performance is strong while the effective stable rank of $(W + \Delta W)$ remains close to that of the base model. Beyond a threshold s_r^*, validation accuracy declines rather than improves. From our observations, this threshold is not universal but depends on the task and the model. With continued training this drift becomes more pronounced, indicating that once optimization leaves the low-rank regime, generalization worsens.

These observations support an optimization-dynamics account: LoRA's effectiveness arises from an implicit low-rank bias that regularizes optimization, keeping updates small and aligned with pretrained structure, rather than from approaching full-rank capacity as nominal rank increases. When this constraint relaxes and effective stable rank exceeds s_r^*, optimization follows less useful directions and performance degrades. This reframes LoRA's strength as arising from staying effectively low-rank, rather than from approximating full fine-tuning at high nominal ranks.

[1] Hu et al. LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685, 2021.

[2] Lion et al. PoLAR: Polar-Decomposed Low-Rank Adapter Representation. arXiv:2506.03133, 2025.

[3] Kalajdzievski, A. A Rank Stabilization Scaling Factor for Fine-Tuning with LoRA. arXiv:2312.03732, 2023.