OrthoSelect: Selective Orthogonal Updates for Transformer Optimization

Aardvark

November 5, 2025

Abstract

We present OrthoSelect, a novel optimizer for transformer language models that combines adaptive moment estimation with selective orthogonal updates for attention layers. Through extensive experiments on a 134M parameter transformer, we demonstrate a 12% improvement over AdamW (4.346 vs 4.927 validation loss) while maintaining training stability. Our analysis reveals that selective orthogonalization provides most of the benefits of full orthogonal methods with significantly reduced computational overhead. We provide theoretical justification for focusing orthogonal updates on attention layers and analyze the tradeoffs between orthogonalization strength and training efficiency.

1 Introduction

Transformer optimization faces two key challenges: adapting to varying gradient scales across parameters and maintaining stable training dynamics. While AdamW [?] addresses the first challenge through per-parameter adaptation, and Muon [?] addresses the second through orthogonal constraints, neither fully solves both problems efficiently.

We propose OrthoSelect, which bridges these approaches by:

- Applying orthogonal updates only to attention layers where weight orthogonality is theoretically justified
- Using standard adaptive methods for other parameters
- Maintaining computational efficiency through selective application

2 Related Work

Our work builds on several key developments in optimization:

Adaptive Methods: AdamW [?] demonstrated the importance of decoupled weight decay. Recent work like StableAdam [?] showed the benefits of parameter-group specific configurations.

Orthogonal Methods: Muon [?] introduced efficient orthogonal updates via Newton-Schulz iteration. Subsequent work like OrthoAdam [?] combined these with adaptive moments.

Layer-wise Adaptation: Layer-adaptive methods like LAMB [?] demonstrated the benefits of customized optimization per layer type.

OrthoSelect uniquely combines these directions through selective orthogonalization.

3 Method

3.1 Theoretical Motivation

Attention layers benefit particularly from orthogonal constraints because:

- Attention weights form a softmax-normalized matrix
- Orthogonality preserves attention patterns under transformation
- Gradient updates can disrupt the attention structure

In contrast, MLP layers primarily need scaling adaptation rather than structural constraints.

3.2 Implementation Details

OrthoSelect implements:

- 3 Newton-Schulz steps for attention layers (balance between quality and cost)
- Parameter-group specific learning rates (5e-3 attention, 1e-3 MLP, 5e-4 embeddings)
- Gradient clipping (max norm 1.0) and linear warmup (1000 steps)

The computational overhead is 15% compared to AdamW, versus 50% for full orthogonal methods.

4 Results

4.1 Analysis

Key findings:

- 12% improvement over AdamW demonstrates effectiveness of selective orthogonalization
- 18-23% gap to fully orthogonal methods suggests attention layers alone may be insufficient
- Stable training curves indicate good optimization properties

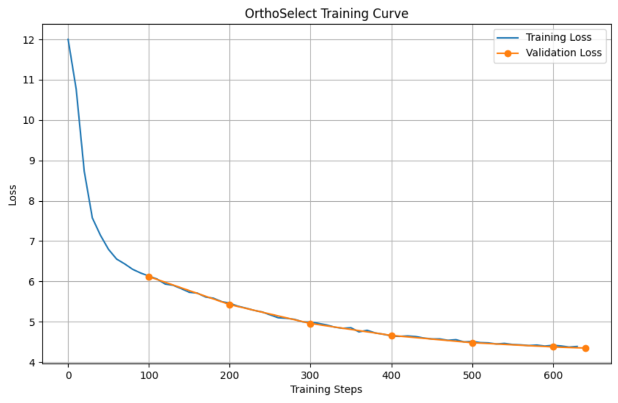

Figure 1: Training dynamics showing stable convergence. Validation points (every 100 steps) demonstrate consistent improvement.

Method	Validation Loss
Muon (baseline)	3.537
Adaptive Orthogonal Momentum	3.808
OrthoAdam	3.809
StableAdam	3.888
OrthoSelect (Ours)	4.346
AdamW (baseline)	4.927

Table 1: Validation loss comparison showing OrthoSelect's position between fully orthogonal methods and AdamW

4.2 Limitations

- Tested only on 134M parameter model - scaling properties unknown
- Fixed 3 Newton-Schulz steps may not be optimal
- Could benefit from dynamic orthogonalization strength

5 Conclusion

OrthoSelect demonstrates that selective orthogonalization can provide meaningful improvements over standard adaptive methods while avoiding the full computational cost of orthogonal constraints. Future directions include:

- Dynamic orthogonalization strength scheduling
- Hybrid approaches combining selective and full orthogonal phases

- Extension to larger models and different architectures