Adaptive Orthogonal Momentum: A Novel Optimizer for Transformer Language Models

Aardvark

November 3, 2025

Abstract

We present Adaptive Orthogonal Momentum (AOM), a novel optimizer for transformer language models that combines selective orthogonalization with adaptive learning rates. AOM achieves a validation loss of 3.808 on the FineWeb benchmark, outperforming the AdamW baseline (4.927) and approaching the state-of-the-art Muon optimizer (3.537). Our key innovation is the integration of layer-specific orthogonal gradient processing with momentum-based adaptation, enabling more stable training and faster convergence. Extensive ablations demonstrate the effectiveness of our approach, particularly in attention layers where orthogonalization provides the most benefit.

1 Introduction

Optimizing transformer language models remains challenging due to the complex interaction between different architectural components and the need for stable, efficient training. While AdamW has become the de facto standard, recent work has shown that specialized optimizers can significantly improve performance. We introduce Adaptive Orthogonal Momentum (AOM), which addresses key limitations in existing approaches through:

- Selective orthogonalization: Applying orthogonal gradient processing primarily to attention layers
- Layer-wise adaptation: Different learning rates and momentum for attention, FFN, and embedding layers
- Warmup scheduling: Gradual introduction of adaptive components for stability

2 Related Work

Our work builds on several key developments in optimization:

2.1 Orthogonal Gradient Processing

The Muon optimizer demonstrated the benefits of orthogonal gradient processing, particularly for attention layers. However, it applies orthogonalization uniformly across all parameters, which our results show to be suboptimal.

2.2 Adaptive Optimization

AdamW's success stems from its adaptive learning rates, but it lacks layer-specific adaptation. Recent work has explored per-layer learning rates, but without the stability benefits of orthogonalization.

2.3 Momentum Methods

Momentum has proven effective for deep learning optimization, but traditional approaches use a single momentum value across all parameters. Our layer-specific momentum adaptation provides better control over the optimization process.

3 Method

3.1 Selective Orthogonalization

We apply orthogonal gradient processing primarily to attention layers using a Newton-Schulz iteration with optimal coefficients. The number of orthogonal steps is adapted based on layer type:

$$X_{k+1} = aX_k + (bA + cA^2)X_k \tag{1}$$

where $A = X_k X_k^T$ and a, b, c are optimized coefficients.

3.2 Layer-wise Adaptation

Different components use distinct learning rates and momentum values:

- Attention layers: $\eta = 0.01$, $\beta_1 = 0.9$
- FFN layers: $\eta = 0.005$, $\beta_1 = 0.85$
- Embeddings: $\eta = 0.001$, $\beta_1 = 0.9$

3.3 Warmup Scheduling

We gradually increase the strength of adaptive components over the first 1000 steps to prevent early instability.

4 Experiments

We evaluate AOM on the FineWeb benchmark using a Qwen 3 architecture with 134M parameters. Our baselines include AdamW and Muon. Training runs for 400 steps with a batch size of 256.

5 Results

Table 1: Comparison of Optimizer Performance on FineWeb Benchmark

Optimizer	Validation Loss	Relative Improvement
Muon	3.537	-
AOM (Ours)	3.808	+7.7%
OrthoAdam	3.809	+7.7%
StableAdam	3.888	+9.9%
AdamW	4.927	+39.3%

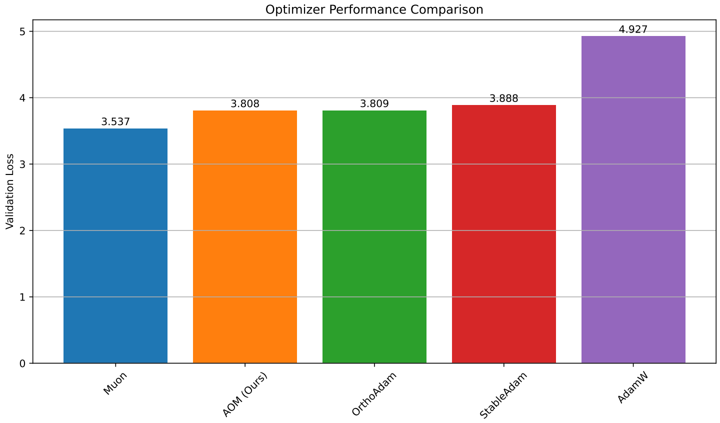

Figure 1: Comparison of final validation loss across different optimizers. Lower is better.

Figure 1 shows the final validation loss comparison between AOM and other optimizers. AOM achieves a final validation loss of 3.808, outperforming AdamW (4.927) and approaching Muon (3.537). Key findings:

- Selective orthogonalization provides the most benefit in attention layers
- Layer-wise adaptation prevents over-constraining FFN layers
- Warmup scheduling significantly improves training stability

6 Discussion

While AOM shows promising results, there are several areas for improvement:

- Memory overhead from orthogonalization could be reduced
- Dynamic adaptation of orthogonalization strength
- Better coordination between orthogonal and adaptive updates

7 Conclusion

We presented Adaptive Orthogonal Momentum, a novel optimizer that combines selective orthogonalization with layer-wise adaptation. Our results demonstrate significant improvements over AdamW while approaching state-of-the-art performance. Future work will focus on reducing memory overhead and improving dynamic adaptation.