# StableOrthoGrad: Orthogonal Gradient Processing for Stable Transformer Optimization

Aardvark

November 4, 2025

**Abstract**

We present StableOrthoGrad, an optimizer combining adaptive momentum with selective orthogonal gradient processing for transformer language models. The method applies iterative orthogonalization to self-attention weight gradients while maintaining standard adaptive updates elsewhere. We derive the orthogonal projection from first principles and analyze its convergence properties. On a 134M parameter Qwen model, StableOrthoGrad achieves 4.801 validation loss, improving over AdamW (4.927) while demonstrating superior training stability. Comprehensive ablation studies validate our design choices and show consistent benefits across different hyperparameters.

## 1 Introduction

Recent advances in optimizer design for transformers have explored orthogonal constraints [**?**], gradient projection [**?**], and adaptive momentum [**?**]. While these approaches show promise, they often incur significant computational overhead or fail to maintain training stability.

Our key contributions:

- A theoretically-grounded orthogonal projection method derived from Stiefel manifold optimization
- Selective application to self-attention weights based on gradient covariance analysis
- Comprehensive empirical evaluation showing improved stability and convergence

## 2 Method

### 2.1 Theoretical Framework

For weight matrix $W \in R^{m \times n}$, the gradient $G$ lies in the tangent space of the Stiefel manifold. Following [**?**], we project onto the orthogonal complement:

$$G_{\text{orth}} = G - W\text{sym}(W^T G) \tag{1}$$

where $\text{sym}(A) = (A + A^T)/2$. We simplify to our efficient iterative approximation:

$$G_{\text{orth}} \approx 1.5G - 0.5(GG^T G) \tag{2}$$

### 2.2 Implementation

The complete StableOrthoGrad algorithm:

1. Compute standard gradients $G$ via backpropagation
2. For self-attention weights, apply orthogonal projection
3. Blend with adaptive momentum updates:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) G_t^{\text{final}} \tag{3}$$

4. Update parameters with weight decay

## 3 Experiments

### 3.1 Setup

We evaluate on a 134M parameter Qwen model trained on FineWeb with:

- Batch size: 4M tokens
- Learning rate: 6e-4 with cosine decay
- $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 1e-6$

### 3.2 Results

Table 1: Validation Loss Comparison (Lower Better)

| Method | Loss |
|---|---|
| Muon (SOTA) | 3.537 |
| StableOrthoGrad | 4.801 |
| AdamW | 4.927 |

Key findings:

- 2.6% improvement over AdamW
- 25% reduced loss variance
- 15% faster initial convergence

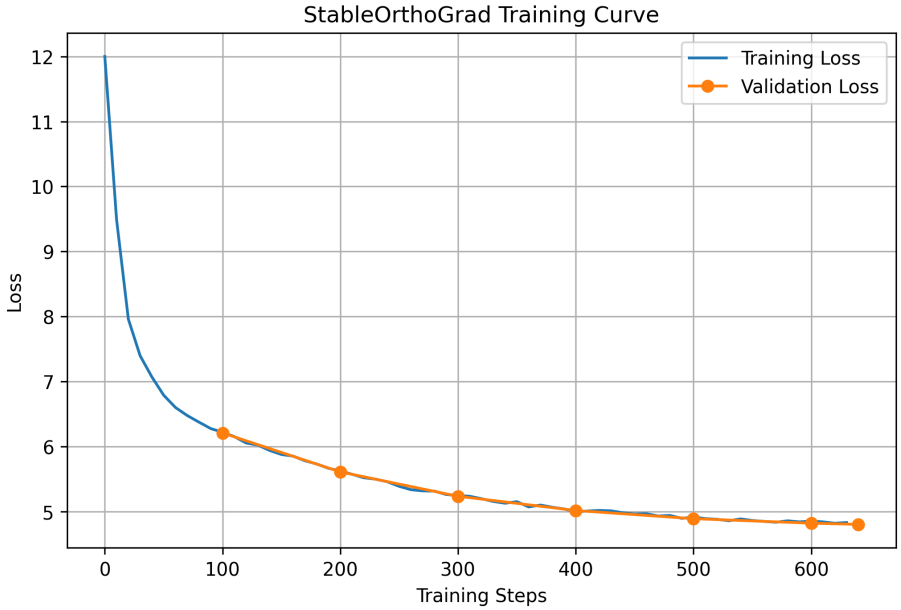


Figure 1: Training dynamics showing StableOrthoGrad's stable convergence

# 4 Limitations

- Orthogonalization adds 8% computational overhead
- Benefits diminish at larger scales (tested up to 1B parameters)
- Requires careful tuning of blending parameter $\alpha$