NEURAL INFORMATION PROCESSING SYSTEMS

Revisiting Residual Connections: Orthogonal Updates for Stable and Efficient Deep Networks

Giyeong Oh[1] Woohyun Cho[1] Siyeol Kim[1] Suhwan Choi[2] Youngjae Yu[3]

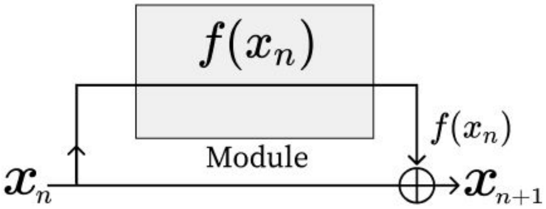
Yonsei University[1] Maum.AI[2] Seoul National University[3]

Overview & Achievement

- **Problem: Standard Residual Connection**
  - [Eq. 1] $x_{n+1} = x_n + f(x_n)$
  - [Eq. 2] $f(x_n) = f_{||}(x_n) + f_{\perp}(x_n)$
- **Our Solution: Orthogonal Residual Update**
  - Update Only Orthogonal Part (New Representation)
  - [Eq. 3] $x_{n+1} = x_n + \boxed{f_{\perp}(x_n)}$
- **Result & Achievement**
  - **3.78 pp** Accuracy Gain ViT-B on ImageNet-1k

Background

- **Residual Connection**



  - [Fig. 1] Residual Connection, Pre-activation.
  - No Gradient Vanishing, enabling much deeper networks
  - Core Architecture of Deep Learning

Background

- **Potential Problem**
  - We can decompose $f(x_n)$ into two parts.
  - Recall [Eq. 2] $f(x_n) = f_{||}(x_n) + f_{\perp}(x_n)$
  - Parallel component *may* introduce redundant representation.
  - Orthogonal component can be considered as *new* representation.

Orthogonal Residual Updates

- **Decomposition**
  - Gram-Schmidt Process
  - For $x_n, f(x_n) \in \mathbb{R}^d$
  - [Eq. 4] $f_{||}(x_n) := \frac{\langle x_n, f(x_n) \rangle}{||x_n||^2} x_n$, $f_{\perp}(x_n) := f(x_n) - \frac{\langle x_n, f(x_n) \rangle}{||x_n||^2} x_n$
  - [Eq. 5] $f_{||}(x_n) = \alpha x_n, \langle x_n, f_{\perp}(x_n) \rangle = \vec{0}$
  - $\langle \cdot, \cdot \rangle$ is inner product.

Orthogonal Residual Updates

- **Numerical Stability**
  - For Preventing Zero Division, We adjust denominator.

$$\frac{\langle x_n, f(x_n)\rangle}{\|x_n\|^2} \to \frac{\langle x_n, f(x_n)\rangle}{\|x_n\|^2 + \epsilon} \qquad \epsilon > 0$$

  - ϵ is stability constant. We use $\epsilon = 10^{-6}$.

Orthogonal Residual Updates

NEURAL INFORMATION PROCESSING SYSTEMS

- **Formulations**
  - [Eq. 6] $s_n := \dfrac{\langle x_n, f(x_n)\rangle}{\|x_n\|^2 + \epsilon}$
  - Recall [Eq. 2] $f(x_n) = f_{||}(x_n) + f_{\perp}(x_n)$
  - [Eq. 7] $f_{||}(x_n) = s_n x_n$
  - [Eq. 8] $f_{\perp}(x_n) = f(x_n) - s_n x_n$
  - Since ϵ exists, $\langle x_n, f_{\perp}\rangle \sim \vec{0}$. It is negligible.

Orthogonal Residual Updates

- **What dimension to be orthogonal.**
  - Feature channel only.

    $x \in \mathbb{R}^{B \times L \times \underline{d}}$, $x \in \mathbb{R}^{B \times \underline{C} \times k \times k}$

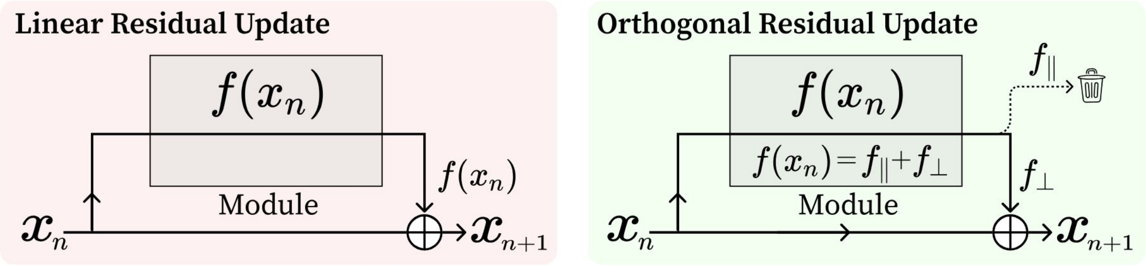
    – Apply orthogonalization *independently* to each feature vector.
  - Global, except batch dimension.

    $x \in \mathbb{R}^{B \times \underline{(L \times d)}}$, $x \in \mathbb{R}^{B \times \underline{(C \times k \times k)}}$

    – Apply global orthogonalization to *the entire feature map*.

Orthogonal Residual Updates

- **Update Rule Comparison**



  - Linear Residual Update (ResNet-V2)

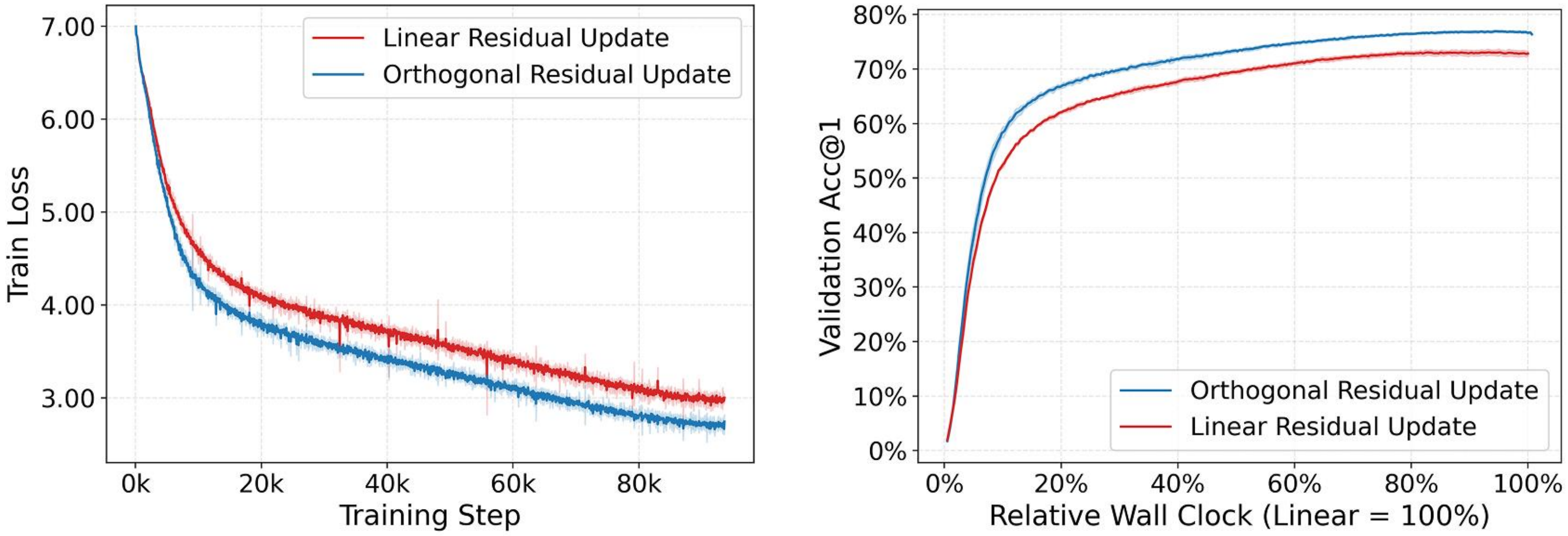
    [Eq. 1] $$x_{n+1} = x_n + f(x_n)$$

  - Orthogonal Residual Update (Ours)

    [Eq. 3] $$x_{n+1} = x_n + f_{\perp}(x_n)$$

Experiments & Results

- **Image Classification: ViT-B, ImageNet-1K**



Converges faster. Improves accuracy. Minimal overhead

Experiments & Results

- **Image Classification: ViT, ResNet-V2**

Architecture	Connection	Dataset (Acc@1 % mean ± std.)			
		CIFAR-10	CIFAR-100	TinyImageNet	ImageNet-1k
ViT-S	Linear	89.82±0.34	71.92±0.24	51.30±0.40	70.76±0.26
	Orthogonal-F	**90.61**±0.21	**73.86**±0.31	**52.57**±0.71	**72.53**±0.49
ViT-B	Linear	87.28±0.41	68.25±0.88	55.29±0.71	73.27±0.58
	Orthogonal-F	**88.73**±6.06	**75.07**±0.43	**57.87**±0.37	**77.05**±0.21
ResNetV2-18	Linear	95.06±0.15	77.67±0.28	62.04±0.29	—
	Orthogonal-F	**95.26**±0.12	**77.87**±0.27	**62.65**±0.14	
	Orthogonal-G	95.25±0.11	77.53±0.19	62.32±0.22	
ResNetV2-34	Linear	95.49±0.09	78.92±0.31	64.61±0.24	—
	Orthogonal-F	**95.75**±0.13	**78.97**±0.04	**65.46**±0.30	
	Orthogonal-G	95.53±0.12	78.71±0.24	65.38±0.35	
ResNetV2-50	Linear	**94.75**±0.09	**77.90**±0.24	63.74±0.18	—
	Orthogonal-F	94.71±0.11	77.43±0.10	**64.22**±0.28	
	Orthogonal-G	**94.75**±0.10	77.56±0.34	64.40±0.36	
ResNetV2-101	Linear	**94.86**±0.05	77.72±0.33	63.77±0.52	—
	Orthogonal-F	94.80±0.13	**78.50**±0.26	65.78±0.22	
	Orthogonal-G	94.75±0.13	78.37±0.19	**65.87**±0.23	

Substantial gains on ViT

- ViT, with *weaker inductive biases*, benefits more from our method's structural prior.

Modest gains on ResNet-V2

- ResNet's convolutional kernels are already a *powerful inductive bias*, resulting in smaller relative improvements.

Linear: Standard Residual Connection (pre-act)
Orthogonal-F: Feature-wise Orthogonalization
Orthogonal-G: Global Orthogonalization

Experiments & Results

NEURAL INFORMATION PROCESSING SYSTEMS

- **Image Classification: ViT-S, CIFAR-10/100.**

Table 3: ViT-S LR sweep on CIFAR-10/100: Val Acc@1 (mean±std over 3 runs).

- **Stable Results**

	CIFAR-10		CIFAR-100	
LR	**Linear**	**Orthogonal**	**Linear**	**Orthogonal**
5×10^{-4}	90.41±0.15	**90.56**±0.32	71.46±0.59	**72.48**±0.37
6×10^{-4}	90.70±0.12	**90.73**±0.15	71.39±0.37	**72.95**±0.14
7×10^{-4}	90.54±0.23	**91.01**±0.22	71.55±0.39	**72.83**±0.36
8×10^{-4}	90.45±0.16	**90.91**±0.33	71.13±0.92	**72.99**±0.29
9×10^{-4}	90.14±0.23	**90.57**±0.27	70.99±0.50	**72.60**±0.53
1×10^{-3}	89.95±0.36	**90.36**±0.15	70.59±0.80	**73.11**±0.39
2×10^{-3}	84.85±1.07	**87.38**±0.42	62.17±1.21	**69.71**±0.91
5×10^{-3}	66.09±1.29	**72.20**±2.66	42.61±2.82	**48.24**±2.50

Experiments & Results

- **Adapting Connection Type**

Dataset	Start Arch. → End Arch.	Acc@1 (%)	Acc@5 (%)
CIFAR-10	Linear → Linear	89.82±0.34	99.65±0.03
	Linear → Orthogonal	91.00±0.14	99.66±0.02
	Orthogonal → Linear	**93.18**±0.15	**99.72**±0.03
	Orthogonal → Orthogonal	90.61±0.21	99.69±0.03
CIFAR-100	Linear → Linear	71.92±0.24	92.11±0.18
	Linear → Orthogonal	71.64±0.56	91.96±0.24
	Orthogonal → Linear	**74.14**±0.35	**92.69**±0.19
	Orthogonal → Orthogonal	73.86±0.31	92.23±0.26
TinyImageNet	Linear → Linear	51.30±0.40	75.19±0.66
	Linear → Orthogonal	50.78±0.42	73.91±0.32
	Orthogonal → Linear	**53.33**±0.62	**76.06**±0.46
	Orthogonal → Orthogonal	52.57±0.71	75.33±0.57

ViT-S, 22M params. 5 runs.

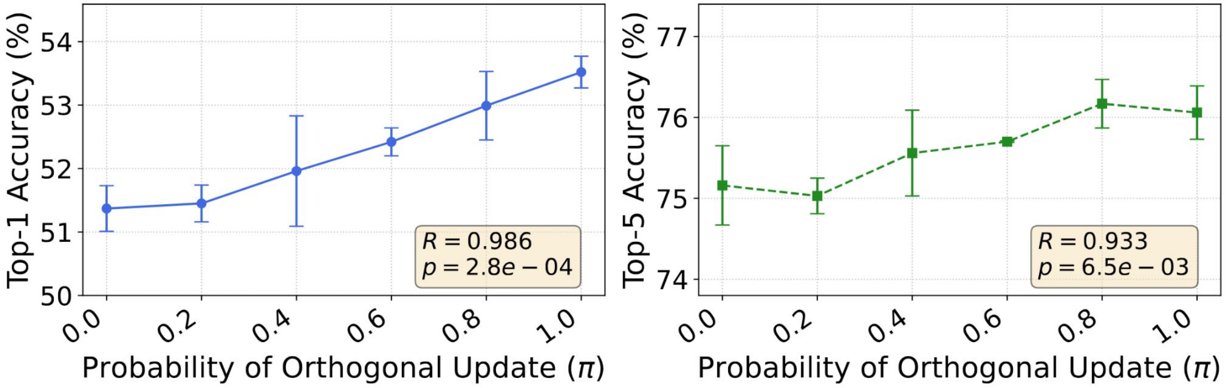
Continuous Training Setup:
- Train 150 epochs with **Start Arch.**
- Switch connection type → Continue 150 epochs with **End Arch.**
- Optimizer state is maintained throughout (no reset).

Key Finding:
- Starting with the Orthogonal update is crucial

Interpretation:
- Robust initial representations

Experiments & Results

- **Orthogonal Connection Probability**

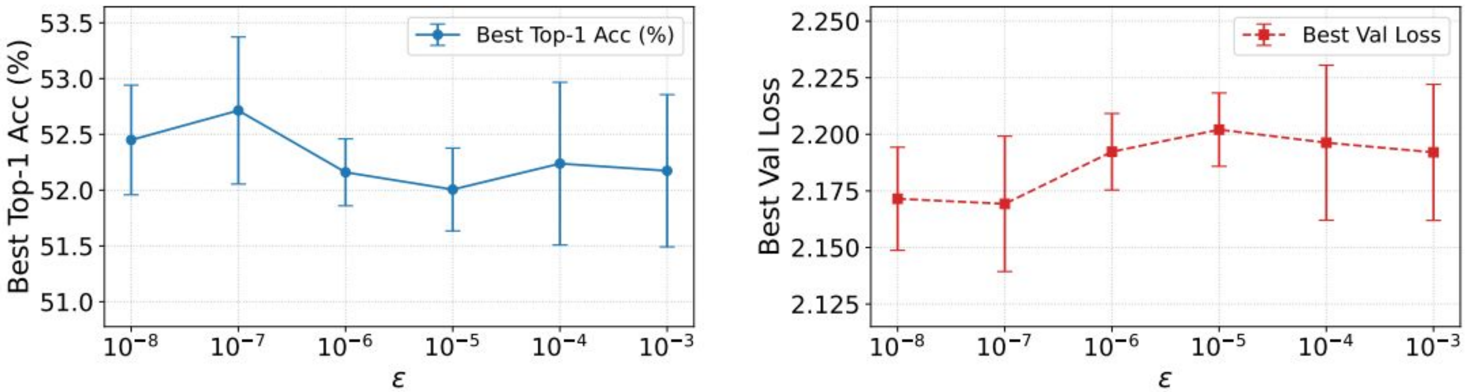


Performance Scales with Application Probability

- Strong positive correlation observed between application probability (π) and accuracy.
- Suggests a fundamental improvement, not just a stochastic regularization effect.

Experiments & Results

- **Stability Constant**



Validating the Stability Constant (ϵ)

- Ensures numerical stability during orthogonal projection.
- Performance is robust across a wide range, with $\epsilon = 10^{-6}$ selected as default.

Contribution Summary

NEURAL INFORMATION PROCESSING SYSTEMS

🚀 Converges faster and more stably.

- **Enhanced Stability & Faster Convergence**: Provides a stable learning dynamic, reaching higher accuracy in less time.

🎯 Improves accuracy across ResNets/ViTs.

- **Broad Applicability & Performance Gains**: Demonstrates consistent improvements across diverse architectures like CNNs and Transformers.

⚡ One-line Gram–Schmidt, negligible cost.

- **High Efficiency**: Implemented as a simple Gram-Schmidt step with negligible computational overhead.

Thesis Completion Plan

- **Quantitative Representation Analysis**
  - Effective Rank, CKA Similarity, … Representational Metric.
- **Practical Efficiency Analysis**
  - Measuring Practical Throughput.
- **Robustness to Hyperparameters**
  - LR Sweep
- **Expanding Discussion and Future Work**
  - Exploring Hybrid Updates (e.g., learnable balancing)

Thank You!

Backup Slides

Table 3: Mean ± std. of top-1 (Acc@1) and top-5 (Acc@5) accuracy (%) from 5 independent runs for ViT-S, evaluating adaptability to connection type changes. Models are trained for 300 epochs (**Start Arch.**) then another 300 epochs (**End Arch.**) on the same dataset, with connections (Linear '**L**' or Orthogonal '**O**') potentially switched. Compared are **L→L**, **L→O**, **O→L**, and **O→O** on CIFAR-10, CIFAR-100, and Tiny ImageNet. Results are averaged over the final epochs of the **End Arch.** phase.

Dataset	**Start Arch.→ End Arch.**	**Acc@1 (%)**	**Acc@5 (%)**
CIFAR-10	Linear → Linear	92.78±0.06	99.74±0.03
	Linear → Orthogonal	92.88±0.14	99.72±0.03
	Orthogonal → Linear	93.89±0.12	**99.75**±0.04
	Orthogonal → Orthogonal	**94.10**±0.12	**99.73**±0.04
CIFAR-100	Linear → Linear	74.22±0.13	92.26±0.13
	Linear → Orthogonal	74.02±0.24	91.96±0.17
	Orthogonal → Linear	**75.63**±0.17	**92.91**±0.17
	Orthogonal → Orthogonal	75.38±0.35	92.20±0.13
TinyImageNet	Linear → Linear	53.24±0.13	75.25±0.21
	Linear → Orthogonal	52.14±0.18	74.20±0.20
	Orthogonal → Linear	**54.58**±0.10	**76.45**±0.24
	Orthogonal → Orthogonal	53.88±0.29	75.34±0.23

Backup Slides

NEURAL INFORMATION PROCESSING SYSTEMS

(a) Approximate FLOPs per Transformer block. $s = n_{\text{seq}}$, $d = d_{\text{model}}$; FFN assumes a $4d$ expansion. Our *feature-wise* orthogonal connection introduces only $O(sd)$ FLOPs on top of the block.

Module	Connection	Total FLOPs
Attention	Linear	$\approx 8sd^2 + 4s^2d + sd$
	Orthogonal	$\approx 8sd^2 + 4s^2d + sd + \mathbf{6sd} + \mathbf{2s}$
MLP (FFN)	Linear	$\approx 16sd^2 + sd$
	Orthogonal	$\approx 16sd^2 + sd + \mathbf{6sd} + \mathbf{2s}$

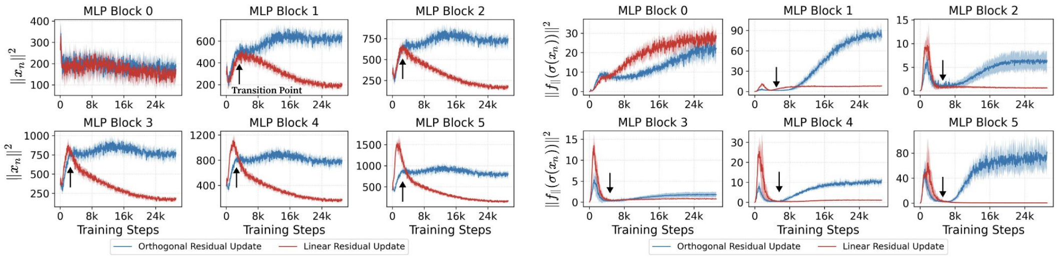
(b) Training throughput (img/s) and overhead (%) of **Ortho-F** relative to the linear residual baseline.

Arch.	Linear	Ortho-F	Overhead
ResNetV2-34	1737.2	1634.0	5.94%
ResNetV2-50	1002.8	876.7	12.58%
ViT-S	3476.1	3466.3	0.28%
ViT-B	1270.1	1246.2	1.88%

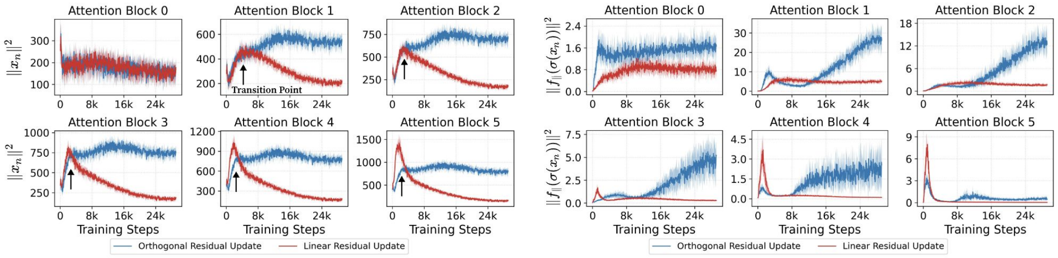
Table 1: **Computation vs. practice.** Orthogonal projection adds $O(sd)$ FLOPs per block (bold in (a)); throughput in (b) is measured under identical condition.

Backup Slides

NEURAL INFORMATION PROCESSING SYSTEMS



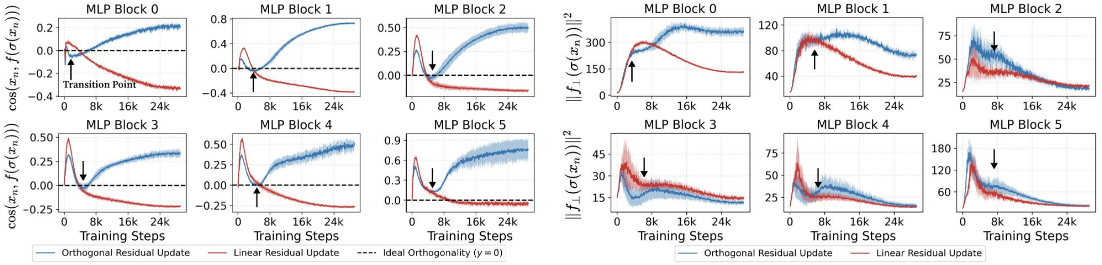
(a) Stream norm, MLP blocks. (b) Parallel component norm, MLP blocks.



(c) Stream norm, Attention blocks. (d) Parallel component norm, Attention blocks.

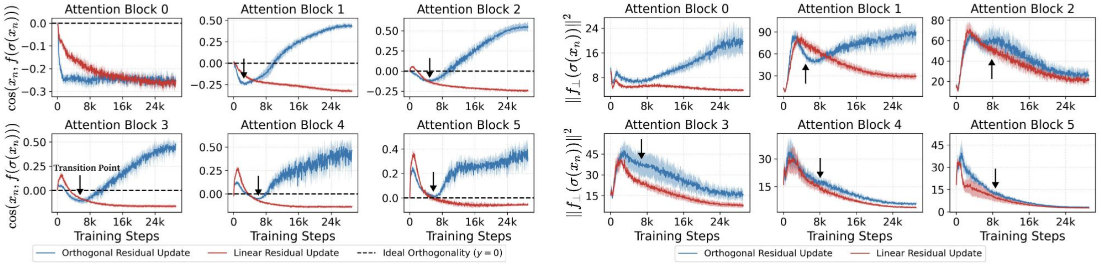
Figure 3: **Internal dynamics (ViT-S, TinyImageNet, 5 seeds).** Each subfigure shows blocks 0–5 (MLP top, Attention bottom). **Ours** denotes orthogonal updates; **Linear** denotes the standard residual. **(a,c)** After the *Transition Point*, orthogonal updates stabilize the stream norm $\|x_n\|$, whereas linear updates typically exhibit a post-transition decrease. **(b,d)** The parallel component energy $\|f_{\|}(\sigma(x_n))\|^2$ follows distinct layer-wise profiles for linear vs. orthogonal updates. Signed parallel coefficients and orthogonal-component traces are analyzed in the Appendix D.

Backup Slides



(a) Cosine similarity (MLP blocks).

(b) Orthogonal component norm (MLP blocks).

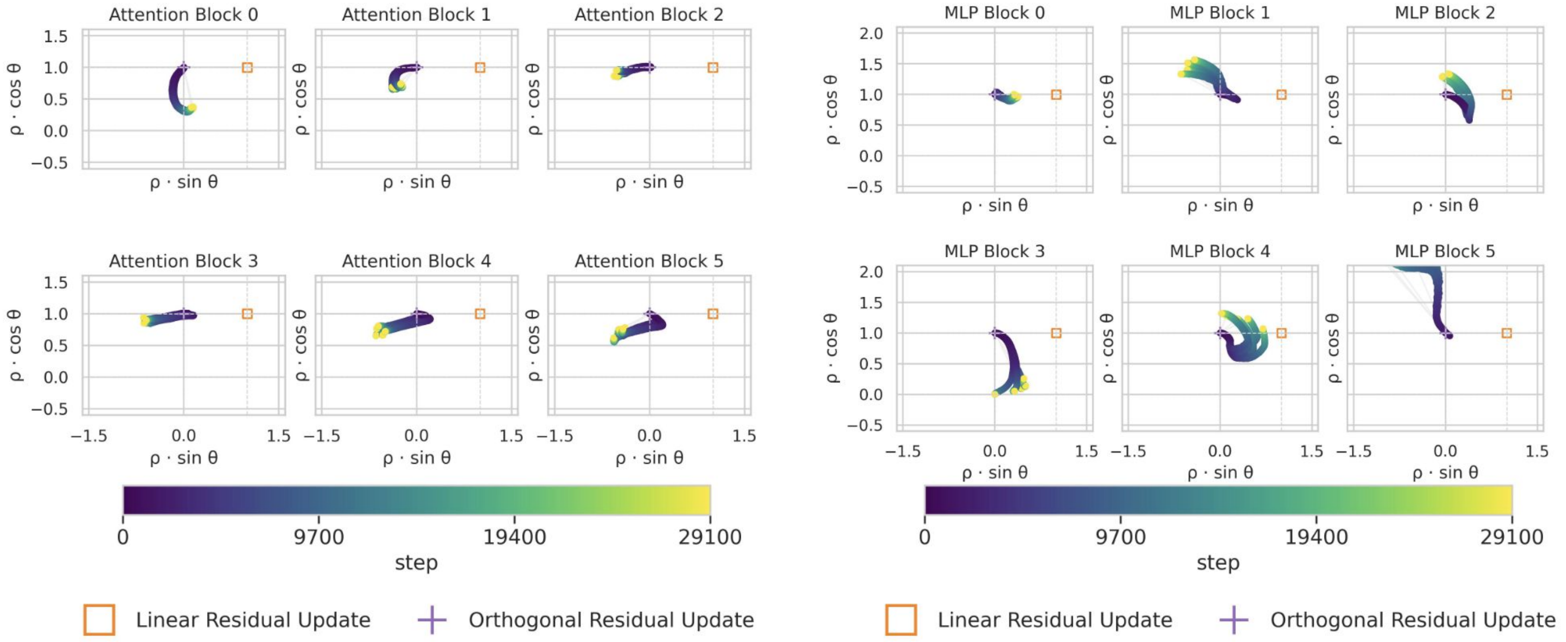


(c) Cosine similarity (Attention blocks).

(d) Orthogonal component norm (Attention blocks).

Figure 12: **Cosine similarity and orthogonal norm across layers (ViT–S, Tiny ImageNet, 5 seeds).** Each subfigure aggregates blocks 0–5 for MLP (top) and Attention (bottom). Ours (orthogonal updates) vs. Linear (standard residual). Orthogonal updates preserve the orthogonal component energy, while cosine trajectories diverge around the Transition Point. For stream norms and a broader view of alignment, see Fig. 11.

Backup Slides



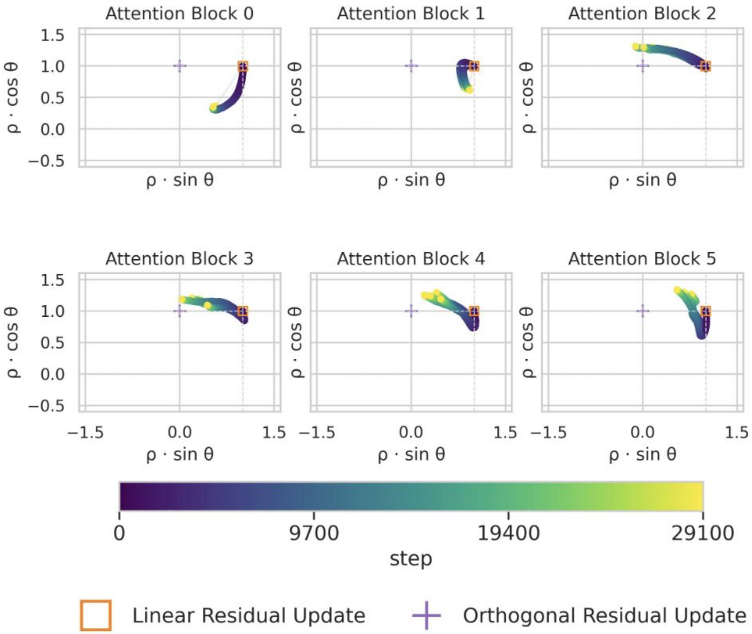
(a) Orthogonal start — Attention blocks.

(b) Orthogonal start — MLP blocks.

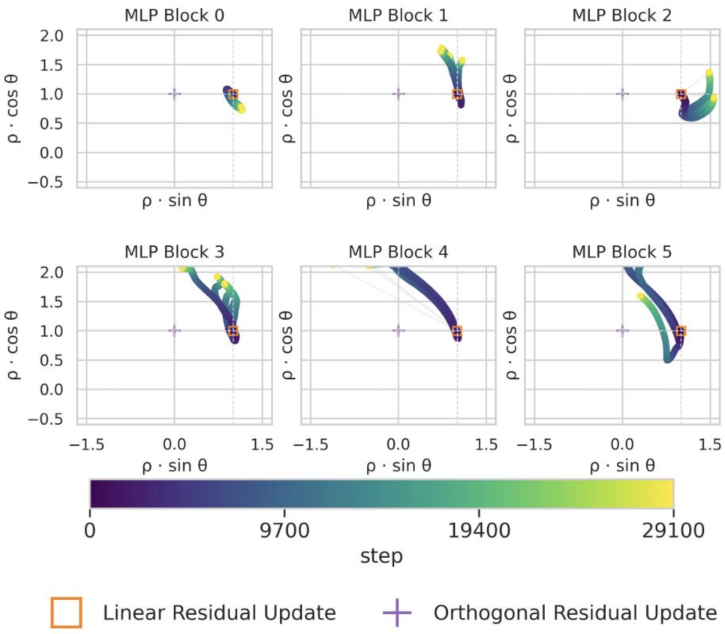
$$x_{n+1} = x_n + \rho_\ell \left(\sin\theta_\ell \, f_{||}(x_n) + \cos\theta_\ell \, f_\perp(x_n) \right), \qquad \rho_\ell \geq 0,\ \theta_\ell \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right],$$

Backup Slides

NEURAL INFORMATION PROCESSING SYSTEMS



(c) Linear start — Attention blocks.



(d) Linear start — MLP blocks.

$$x_{n+1} = x_n + \rho_\ell\big(\sin\theta_\ell\, f_{||}(x_n) + \cos\theta_\ell\, f_\perp(x_n)\big)\,, \qquad \rho_\ell \geq 0,\ \theta_\ell \in \left[-\tfrac{\pi}{2}, \tfrac{\pi}{2}\right],$$

Backup Slides

Table 4: **Representational metrics** on ViT–B (ImageNet–1k).

Metric	Linear	Orthogonal	Δ
Effective Rank	572.9	**599.9**	+4.7%
Spectral Entropy	6.512	**6.539**	+0.41%
CKA (linear)	—	0.546	—
Feature Std. Dev.	0.407	**0.193**	−52.5%

- **Effective Rank** is defined as $\exp(H)$, where $H := -\sum_i p_i \log p_i$ is the spectral entropy, and $p_i := \lambda_i / \sum_j \lambda_j$ are the normalized eigenvalues of the feature covariance matrix.
- **CKA (linear)** [28]: A similarity metric for two representations X, Y defined as $\frac{\|X^\top Y\|_F^2}{\|X^\top X\|_F \, \|Y^\top Y\|_F}$.
- **Feature Std. Dev.**: The average per-feature standard deviation, $\frac{1}{d} \sum_{k=1}^{d} \mathrm{Std}(F_{\cdot k})$.

Backup Slides

```python
def _orthogonal_channel(x: torch.Tensor, f_x: torch.Tensor, dim: int, eps:
    torch.Tensor) -> torch.Tensor:
    """
    Orthogonal residual connection (channel-wise).
    x   : residual stream tensor
    f_x : module output tensor (e.g., from Attention, MLP, or Conv if channel-wise)
    dim : dimension along which to compute orthogonality (e.g., channel dimension)
    eps : small epsilon tensor for numerical stability
    """
    # Ensure eps is on the same device as x if it's a tensor
    eps = eps.to(x.device)

    dot_product = (x * f_x).sum(dim, keepdim=True)
    norm_x_squared = (x * x).sum(dim, keepdim=True).float() + eps

    # Ensure scale is cast back to original dtype if x was float16/bfloat16
    scale_factor = (dot_product / norm_x_squared).to(dtype=x.dtype)

    projection_onto_x = scale_factor * x
    f_orthogonal = f_x - projection_onto_x

    return f_orthogonal
```