Rethinking Attention in Depth for Operator Learning

Jaehyeon Lee[1,2] **Kiwon Um**[3] **Min-Koo Kang**[1*] **JungHyun Han**[2*]
[1]Korea Institute of Science and Technology, [2]Korea University, [3]LTCI, Télécom Paris, IP Paris
{jaehyeon,minkoo}@kist.re.kr, kiwon.um@telecom-paris.fr, jhan@korea.ac.kr

Abstract

Transformers have rapidly emerged as powerful surrogate models for learning solution operators for partial differential equations (PDEs). Standard attention tends to exhibit depth-wise representational bottlenecks across layers, with the role of depth remaining underexplored in modeling complex PDEs. In this paper, we rethink the role of depth by promoting it to an explicit computational dimension of attention kernel construction, rather than a passive architectural parameter. With this strategy, each attention block's native representation is enriched via a gated mixture of inter-layer differences, enabling information to be coupled and propagated across layers. The depth-wise hierarchy admits a theoretically motivated multi-kernel interpretation with weakly correlated kernels and improved generalization bounds. Beyond dense feature aggregation, our approach yields a more expressive form of depth-wise attention, mitigating spectral collapse, preserving interaction-scale diversity, and improving intra- and inter-layer route utilization. Our kernel-level differential mixing design achieves more significant gains over the standard Transformer than inter-layer information mixing methods. Extensive empirical evaluations demonstrate its broad compatibility across state-of-the-art solvers, presenting consistent performance gains on eight PDE benchmarks.

1 Introduction

Partial Differential Equations (PDEs) are foundational tools for modeling the dynamics of multiphysics and multiscale systems in various fields, ranging from geophysics to biophysics [15]. These PDEs are typically solved using numerical methods such as the finite element method. However, numerical solvers remain computationally expensive and struggle with nonlinear dynamics, inverse problems, and noisy or incomplete data [19]. These issues are more pronounced in real-world systems with heterogeneous scales. Recently, deep learning models have emerged as powerful tools for solving PDEs by learning mappings between input functions and PDE solutions [28, 36]. Benefiting from the expressivity of neural networks, these models could be trained on a dataset to generalize to unseen cases and produce solutions with significant speedups over numerical solvers [12].

Transformers [51] have succeeded across diverse domains [23, 4, 33] and are increasingly adopted in neural operator architectures [26]. Relying on the standard Transformer structure, previous studies have sought to improve PDE modeling through various strategies such as domain partitioning [56], hierarchical attention with multigrid strategies [32], and hybrid architectures with convolutional layers [30]. These approaches introduce valuable inductive biases but primarily operate at the input or intra-layer level.

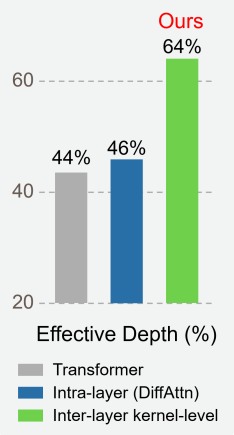


Figure 1: Transformer and intra-layer strategy often under-utilize model depth.

*Co-corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

For better attention expressivity, Transformer layers can be stacked in depth, where depth serves as an intrinsic parameter. Such a stacked design, however, often leads to structural limitations, including weak inter-layer coupling and ineffective depth-wise refinement [10, 46]. Prior studies have reported depth-related drawbacks, including feature oversmoothing [48, 63, 59], local feature attenuation in deeper layers [44, 63], and extraneous attention noise [61]. To improve depth-wise information flow, inter-layer mixing has been explored, largely focusing on cross-layer aggregation [41, 57]; however, how such information is processed remains underexplored. Figure 1 shows the normalized effective number of layers whose attention score patterns contribute to output prediction, indicating a bottleneck in expressivity of Transformer (see the metric definition in Appendix F.1). The standard Transformer shows limited effective depth utilization, with many layers contributing marginally to output prediction. Although recent Differential Transformer (DiffAttn [61]), an intra-layer denoising strategy, improves attention score utilization, it has little effect on this bottleneck. Our inter-layer kernel-level strategy improves effective depth utilization by nearly $1.5\times$. This motivates a fully Transformerized design that effectively leverages information flow across depth. Such a structure is essential for capturing complex physical dynamics [22, 2, 26], while remaining compatible with existing Transformer-based operator learning pipelines.

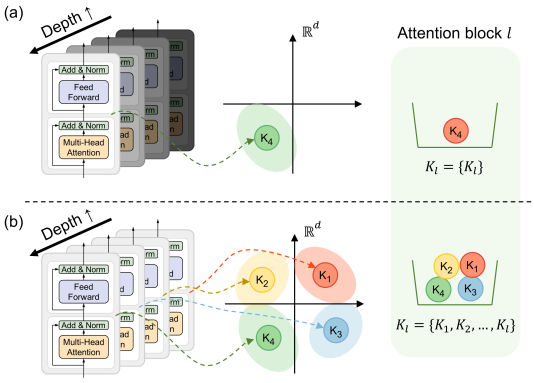


Figure 2: Illustration of kernel usage in attention across depth: (a) The standard attention using a single kernel generated at the current attention block and (b) our MoDD attention using a set of uncorrelated kernel experts accumulated from all previous depth layers in the representation space $\mathbb{R}^d$.

In this paper, we present a cross-layer attention framework, named Mixture of Depth Differential (MoDD). As illustrated in Figure 2, MoDD induces each attention block to operate as depthwise mixture of uncorrelated kernel experts. The kernel experts are accumulated from previous depth layers in the representation space, in contrast to standard Transformer attention, which relies solely on the kernel generated at the current depth. Crucially, MoDD is a *plug-and-play* module that operates within standard attention blocks, ensuring compatibility with Transformer-based operator solvers.

The key contributions of MoDD are summarized as follows:

- A depth-wise attention structure that treats depth as an explicit representational dimension, forming a structured hierarchy of attention representations, rather than as a passive, intrinsic architectural hyper-parameter.
- A theoretical characterization of depth-wise attention as a hierarchy of weakly correlated kernels, exploring the proposed depth-wise construction from a multi-kernel perspective.
- Extensive experiments on operator learning for PDEs, demonstrating consistent performance gains of MoDD, as a plug-and-play architecture, over state-of-the-art models across eight PDE benchmarks and favorable tradeoffs between accuracy and efficiency.

2 Related work

Neural operators learn data-driven function-to-function mappings for PDEs without relying on explicit PDE formulations. DeepONet [35] and FNO [28] learn operators via trunk–branch architectures and Fourier kernels, respectively. Subsequent variants improve multiscale/multi-input modeling [54, 18, 62], efficiency [24], and irregular-domain handling [29, 31].

Transformers [51], a foundational architecture in deep learning, have emerged as neural operators for solving PDEs. Recent work on Transformer-based neural operators explores linear attention [5], self-/cross-attention for continuous functions, varying grids, multiple inputs, and irregular meshes [26, 14], hierarchical or multigrid attention [32], and scalable factorized, sliced-token, or latent-space attention mechanisms [27, 56, 53]. These advancements have primarily focused on attention over spatial discretizations, often motivated by continuum or integral-operator views of neural operators. In contrast, MoDD extends to an orthogonal direction by modifying how attention representations are composed across network depth, making it complementary to Transformer-based neural operators.

Inter-layer information mixing in Transformers improves information flow and depth utilization beyond standard residual accumulation by enabling selective access, aggregation, or routing of representations across layers. Cross-layer designs retrieve and update representations from earlier layers [11, 52], aggregate preceding block outputs through static and input-dependent dense connections [41, 57], and generalize residual pathways through learned inter-layer mixing and depth-wise selective aggregation [64, 20]. These approaches differ in how information is routed across depth, but primarily operate on layer representations or residual states themselves. In contrast, MoDD adaptively mixes inter-layer differences of kernel-forming representations, shifting the target of cross-layer mixing from layer representations to their differential representations at the attention-kernel level.

3 Mixture of Depth Differential

Unlike the sequentially stacked multi-layer structure of Transformers with local inter-layer dependency, MoDD introduces a depth-mixture architecture, as illustrated in Figure 3(a). A cross-layer stack aggregates query, key, and value features from all preceding layers to serve as input to MoDD. Following MoDD, the resulting features are used in multi-head attention. The core mechanism of MoDD (Figure 3(b)) consists of two components: a differential operator and a gating network.

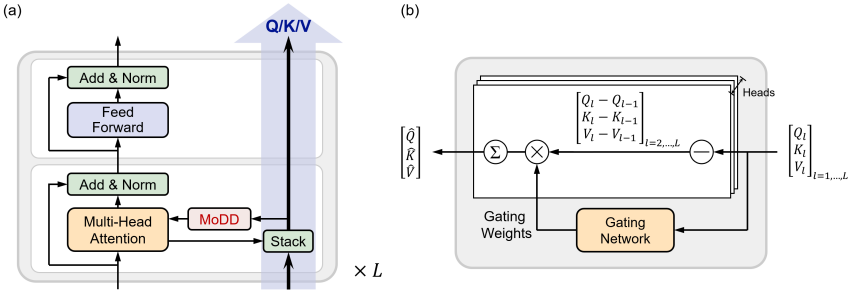


Figure 3: Architecture and working mechanism of MoDD. (a) A Transformer layer with MoDD leverages a cross-layer path of query, key, and value. (b) MoDD computes inter-layer differences of Q, K, and V, and aggregates them using a gating strategy so that each head at layer L takes a weighted combination as input.

Given an input $X^{(l)}$ at layer l, we project it into query, key, and value using the following head-specific projection matrices for each head $i \in [1, h]$: $Q_i^{(l)} = X^{(l)} W_i^Q$, $K_i^{(l)} = X^{(l)} W_i^K$, and $V_i^{(l)} = X^{(l)} W_i^V$.

Step 1. Layer-wise differentiation Residual connections propagate information across layers but do not explicitly regulate correlations between depth-wise attention kernels [16]. MoDD, on the other hand, computes inter-layer differences to address redundant attention representations across depth:

$$\Delta Z_i^{(l)} = Z_i^{(l)} - Z_i^{(l-1)} \quad \text{for } Z \in \{Q, K, V\}. \tag{1}$$

Theorem 3.1 provides a kernel-level view showing that layer-wise differentiation can reduce an upper bound on cross-layer kernel alignment of induced exponential dot-product kernels. Compared to kernels constructed from raw layer representations, the reduction in cross-layer alignment arises through an additional attenuation factor α_{lm} on the $\mathcal{B}_{lm}$ term, which decreases as layer-wise differencing suppresses cross-layer correlations. This reduction in cross-layer alignment is subsequently leveraged in Theorem 3.2 to yield a tighter upper bound on the variance component of the excess-risk bound. We observe the decorrelation behavior derived from Theorem 3.1, while empirically examining the associated assumptions in trained networks (see Appendices F.2–F.5). We discuss this in Sec. 4.5.

Definition 3.1. Let $l \geq 1$ and let $q^{(l)}, k^{(l)} \in \mathbb{R}^d$ denote the query and key representations at layer l. For the scaled dot-product $S_l(q, k) := \frac{1}{\sqrt{d}} (q^{(l)})^\top k^{(l)}$, the associated exponential dot-product kernel [8] is $K^{(l)}(q, k) := \exp\big(S_l(q, k)\big)$.

Theorem 3.1 (Layer-wise differentiation tightens an upper bound on cross-layer kernel alignment)**.** *Fix $l \neq m$ with $l, m \geq 2$. For each layer $t \geq 2$, let $\widetilde{K}^{(t)}$ denote the kernel from Definition 3.1 evaluated at $(\Delta q^{(t)}, \Delta k^{(t)})$. Under the assumptions in Appendix G.7, the population centered alignment $\rho(\cdot, \cdot)$ satisfies*

$$\left|\rho(\widetilde{K}^{(l)}, \widetilde{K}^{(m)})\right| \;\le\; \alpha_{lm}\,\widetilde{\mathcal{B}}_{lm} \;+\; \widetilde{\mathcal{R}}_{lm},$$

where α_{lm} is an explicit contraction factor depending only on baseline-relative cross-layer delta-feature covariance ratios, $\widetilde{\mathcal{B}}_{lm}$ is a normalized baseline cross-layer second-order covariance coupling term, and $\widetilde{\mathcal{R}}_{lm}$ is an explicit remainder term collecting centering and higher-order effects.

Proof. See Appendix G.7.

Step 2. Gated mixture To improve expressive representations, MoDD linearly combines inter-layer differentials using a learnable soft gating mechanism:

$$\hat{Z}_i^{(l)} \;=\; \sum_{k=1}^{L-1} \lambda_i^{(k)}\, \Delta Z_i^{(l-k)} \;\text{ for } Z \in \{Q, K, V\}, \tag{2}$$

where the gating weights $\lambda_i^{(k)} \in \mathbb{R}$. They are computed via softmax over the linearly projected stacked K across depth: $\boldsymbol{\lambda}_i = \mathrm{softmax}\Big([K_i^{(0)}; K_i^{(1)}; \cdots; K_i^{(L-1)}]W_g\Big)$, with $\sum_{l=0}^{L-1} \lambda_i^{(l)} = 1$. Here, each key vector $K_i^{(l)} \in \mathbb{R}^D$ is projected to a scalar gating logit via a shared weight matrix $W_g \in \mathbb{R}^{D\times 1}$. Gating logits are derived from key representations, motivated by the role of keys in defining attention operator structure, and supported by ablation results in Appendix B.1. The gating parameterization is orthogonal to MoDD, thus compatible with other gating strategies. Appendix B.2 presents various results, ranging from shared gating to per-stream gating.

While nonlinear transformations can enrich attention representations, MoDD instead employs a linear combination of inter-layer differences to avoid additional inductive bias [60]. This is particularly valuable in operator learning, where models already incorporate their own inductive priors such as structured token inputs or positional encodings. By avoiding additional biases, MoDD remains fully compatible with these models. This gated aggregation yields an explicit multi-kernel product representation, as formalized in Lemma 3.1. The lemma characterizes the MoDD attention kernel induced by the gated query–key construction, rather than the full Transformer block. This product structure is subsequently used in Theorem 3.2 to derive a tighter upper bound on the RKHS-norm bias term, motivating the gated mixture structure.

Lemma 3.1 (Gated mixture induces a multi-kernel product)**.** *Let $\{\lambda^{(i)}\}_{i=2}^{l}$ be nonnegative weights shared by the gated mixtures of both queries and keys, and let K_{MoDD} denote the exponential dot-product kernel (Definition 3.1) applied to the gated mixtures defined in Equation* (2)*. Then K_{MoDD} is positive semidefinite on the representation space and admits the explicit product-kernel representation*

$$K_{\mathrm{MoDD}}(q,k) = \prod_{i=2}^{l}\prod_{j=2}^{l} K^{(i,j)}\big(\Delta q^{(i)},\, \Delta k^{(j)}\big),$$

where the factor kernel is defined for each layer index (i,j) by $K^{(i,j)}(u,v) = \exp\Big(\frac{\lambda^{(i)}\lambda^{(j)}}{\sqrt{d}}\, u^\top v\Big)$.

Proof. See Appendix G.8.

Step 3. Multi-head attention The outputs of all attention heads are computed, concatenated, and projected:

$$\mathrm{head}_i^{(l)} = \mathrm{softmax}\Big(\hat{Q}_i^{(l)} \hat{K}_i^{(l)\top}/\sqrt{d}\Big)\, \hat{V}_i^{(l)}, \quad \mathrm{MultiHead}(X^{(l)}) = \mathrm{Concat}(\mathrm{head}_1^{(l)}, \ldots, \mathrm{head}_h^{(l)})W^O, \tag{3}$$

where W^O is a learnable projection matrix.

The depth-wise product kernel structure motivates our design, and a kernel ridge regression (KRR) analysis facilitates the investigation of its generalization behavior. Theorem 3.2 provides an excess-risk upper bound $\mathbb{E}[R(\hat{f}_{\lambda,n}) - R(f^*)]$ that admits a standard bias–variance decomposition. The bound is derived for the KRR estimator associated with the MoDD attention kernel, rather than for the full Transformer architecture. Specifically, the bound consists of (i) a bias term controlled by the RKHS norm $\|f^*\|_{\mathcal{H}_{\mathrm{MoDD}}}$ and regularization, (ii) a noise-independent finite-sample estimation term governed by the effective dimension $d_{\mathrm{MoDD}}(\lambda)$, and (iii) a noise-driven variance term that depends on an alignment-sensitive second-moment quantity $\mathcal{V}_{\mathrm{MoDD}}$. Crucially, the reduced cross-layer alignment in Theorem 3.1 tightens $\mathcal{V}_{\mathrm{MoDD}}$, reducing the variance bound and linking depth-wise kernel decorrelation to a tighter generalization bound.

Table 1: MoDD outperforms existing inter-layer information mixing methods. Relative ℓ^2 errors ($\times 10^2$) are reported on both benchmarks. Parameter count, GPU memory, and training time per epoch are measured on Elasticity under the same Transformer backbone and training protocol.

Layer-mixing strategy	Param. #	Memory	Time	Relative ℓ^2 error [$\times 10^2$]	
	[M]	[GB]	[s/epoch]	Elasticity	Airfoil
None	0.83	0.87	6.7	0.62	0.75
DenseFormer [41]	0.83	0.90	7.2	0.53	0.70
MUDDFormer [57]	0.91	0.95	13.1	0.52	0.69
Full AttnRes [20]	0.83	1.15	13.6	0.49	0.69
MoDD (ours)	0.83	0.95	9.2	**0.40**	**0.62**

Theorem 3.2 (Excess-risk bound for KRR with MoDD kernels)**.** *Let $\hat{f}_{\lambda,n}$ be the kernel ridge regression estimator associated with K_{MoDD} and regularization parameter $\lambda > 0$. Under the assumptions stated in Appendix G.9, there exists a constant $C_{\mathrm{ed}} > 0$ such that*

$$\mathbb{E}\big[R(\hat{f}_{\lambda,n}) - R(f^*)\big] \le 2\lambda \|f^*\|^2_{\mathcal{H}_{\mathrm{MoDD}}} + \frac{2C_{\mathrm{ed}}}{n}\, d_{\mathrm{MoDD}}(\lambda) + 2\tau^2 \min\left\{ \frac{C_{\mathrm{ed}}}{n\lambda}\, d_{\mathrm{MoDD}}(\lambda),\ \frac{1}{n\lambda^2}\, \mathcal{V}_{\mathrm{MoDD}} \right\}$$

where $d_{\mathrm{MoDD}}(\lambda)$ denotes the effective dimension associated with K_{MoDD}, and $\mathcal{V}_{\mathrm{MoDD}}$ is an alignment-sensitive upper bound on the second moment $\mathbb{E}[K_{\mathrm{MoDD}}(X, X')^2]$.

Proof. See Appendix G.9.

4 Experiments

We evaluate our model, MoDD, for a wide variety of different physical systems involving various PDE problems. To this end, we conduct experiments using well-established benchmarks for fluid dynamics and solid mechanics problems. The benchmarks cover diverse geometries such as point clouds, structured meshes, and regular grids. Specifically, we adopt eight datasets [28, 29, 42, 6], including a large-scale 4D (i.e., spatiotemporal) Shape-Net Car benchmark. We adopt the experimental setups and training protocol used in the SOTA solver, i.e., Transolver [56] and AMG [25] , to first ensure a fair and direct comparison with the best baseline model. The MoDD-integrated SOTA architecture and implementation details are provided in Appendix A.

4.1 Problem setup

We consider PDEs defined on a bounded open domain $D \subset \mathbb{R}^d$. Let $\mathcal{F}$ and $\mathcal{U}$ denote the input and solution function spaces over D, respectively. Then, the input function is $f : D \to \mathbb{R}^{d_f} \in \mathcal{F}$ and the PDE solution is $u : D \to \mathbb{R}^{d_u} \in \mathcal{U}$. The goal of operator learning is to approximate the ground-truth solution operator $G : \mathcal{F} \to \mathcal{U}$ using a neural operator G_θ parameterized by θ. The model is trained on a dataset of function pairs $\{(f_i, u_i)\}_{i=1}^N$. Since neural networks cannot directly handle infinite-dimensional functions of input or output, f_i and u_i are discretized on mesh $\mathbf{X} := \{x_j \in D\}_{1 \le j \le M}$, resulting in $f_i := \{(x_j, f_i(x_j))\}_{1 \le j \le M}$ and $u_i := \{(x_j, u_i(x_j))\}_{1 \le j \le M}$, with M mesh points. The model is trained to minimize the relative ℓ^2 error: $\min_\theta \frac{1}{N} \sum_{i=1}^N \frac{\|G_\theta(f_i) - u_i\|_2}{\|u_i\|_2}$, where mean-squared error (MSE) is normalized by $\|u_i\|_2$ to account for variations in absolute scale across benchmarks.

4.2 Inter-layer information mixing evaluation

We first compare MoDD with representative inter-layer information mixing methods [41, 57, 20], using the same backbone and training protocol. In Table 1, MoDD consistently provides the largest improvement over the standard Transformer, achieving 1.7–2.6$\times$ greater error reductions than DenseFormer, MUDDFormer, and Full AttnRes across the two benchmarks. MoDD also requires only 68–70% of the training time of MUDDFormer and Full AttnRes, while using fewer parameters than MUDDFormer and less memory than Full AttnRes. DenseFormer is computationally lighter, but yields higher errors than MoDD. Collectively, MoDD's gains arise not merely from cross-layer aggregation, but from mixing inter-layer differential representations at the attention-kernel level.

Table 2: Benchmark evaluation. The means and standard deviations of relative ℓ^2 errors ($\times 10^2$) over three trials with different random seeds are reported; lower mean indicates better accuracy. Transolver* is the result reproduced by our implementation and training; the others are the performances reported in the original papers.

Model	**Point cloud**	**Structured mesh**			**Regular grid**	
	Elasticity	Plasticity	Airfoil	Pipe	Navier-Stokes	Darcy
FNO [28]	/	/	/	/	15.56	1.08
OFormer [26]	1.83	0.17	1.83	1.68	17.05	1.24
ONO [58]	1.18	0.48	0.61	0.52	11.95	0.76
Transolver [56]	0.64	0.12	0.53	0.33	9.00	0.57
Transolver*	$0.63_{\pm 0.038}$	$0.13_{\pm 0.006}$	$0.53_{\pm 0.022}$	$0.48_{\pm 0.028}$	$9.49_{\pm 0.283}$	$0.59_{\pm 0.012}$
Transolver* + **MoDD (ours)**	$0.56_{\pm 0.017}$	$0.11_{\pm 0.007}$	$0.43_{\pm 0.010}$	$0.40_{\pm 0.018}$	$9.07_{\pm 0.302}$	$0.54_{\pm 0.010}$
Promotion w.r.t. Transolver*	11.1%	15.4%	18.9%	16.7%	4.4%	8.5%

4.3 Performance and depth scaling

Table 2 summarizes the evaluation results of representative neural operator methods for the six benchmarks using the same metric. (Table 11 in Appendix A provides a comprehensive comparison with broader models.) MoDD, integrated into the SOTA solver, consistently improves prediction accuracy across all the tasks. Efficiency comparisons and prediction visualizations for each benchmark are summarized in Appendix D and Appendix E, respectively. These gains show the effectiveness of MoDD for operator learning. It is also worth highlighting its compatibility with the standard Transformer architecture, including token embeddings, positional encodings, and attention operations.

To assess the scaling of MoDD's expressivity in depth, Figure 4 compares relative prediction errors across model depths with corresponding FLOPs and parameter counts. On Elasticity, MoDD (red curve) with $L = 8$ achieves accuracy comparable to Transformer (blue curve) with $L = 32$, while using 3.9× fewer parameters and FLOPs. Crucially, this depth-efficiency benefit extends to a more challenging industrial-scale setting. We evaluate MoDD with Transolver++, a neural solver for large-scale geometries, on AIRCRAFT, a 3D benchmark with approximately 0.3M mesh points per sample [37]. MoDD (orange curve) with $L = 4$ achieves accuracy comparable to Transolver++ (teal curve) with $L = 16$, while using 3.8× fewer parameters and FLOPs. This highlights the parameter and FLOP efficiency of MoDD, making it a preferable alternative to simply increasing model depth. MoDD also shows favorable memory and training-time scaling over depth, as well as negligible GPU memory overhead under increasing token resolution (see Appendices D.2 and D.3).

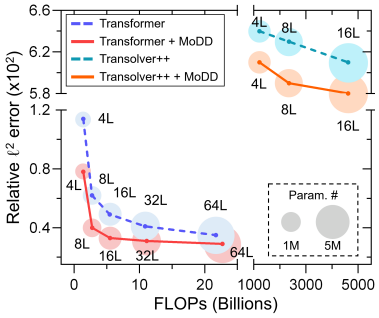


Figure 4: Model size vs. prediction error. L denotes the number of Transformer layers. Marker size indicates parameter count.

Beyond standard Transformer-based solvers, we further evaluate MoDD integrated into AMG, a recent SOTA graph-based Transformer solver. As shown in Table 3, MoDD consistently improves AMG on both 2D and 3D benchmarks, with notable gains of up to 20.5%, indicating seamless integration with transformer architectures using both standard and graph-based attention blocks. Further evaluations in Appendices B.3 and C.1 show that MoDD maintains consistent gains across data regimes, including limited-data settings, and across model depths, supporting its robustness beyond a specific data scale or architectural depth.

Table 3: Evaluation of different models (Transolver [56] and AMG [25]). Lower mean relative ℓ^2 errors ($\times 10^2$) over three random seeds indicate better accuracy.

Model	2D		3D		
	Darcy	NS	DP	Shape-Net Car	
				pres.	vel.
Transolver	0.57	9.00	/	/	/
Transolver*	$0.59_{\pm 0.012}$	$9.49_{\pm 0.283}$	$2.83_{\pm 0.021}$	$8.18_{\pm 0.089}$	$8.21_{\pm 0.069}$
Transolver* + **MoDD**	$0.54_{\pm 0.010}$	$9.07_{\pm 0.302}$	$2.71_{\pm 0.022}$	$8.10_{\pm 0.097}$	$8.05_{\pm 0.067}$
Gain over Transolver*	8.5%	4.4%	4.2%	1.1%	2.0%
AMG	/	4.76	2.57	8.78	9.19
AMG*	$0.50_{\pm 0.032}$	$4.58_{\pm 0.754}$	$2.79_{\pm 0.063}$	$8.75_{\pm 0.181}$	$9.28_{\pm 0.530}$
AMG* + **MoDD**	$0.40_{\pm 0.004}$	$3.64_{\pm 0.157}$	$2.62_{\pm 0.044}$	$8.33_{\pm 0.216}$	$8.86_{\pm 0.116}$
Gain over AMG*	20.0%	20.5%	6.1%	4.8%	4.5%

4.4 Ablation study

We perform an ablation study to investigate the contribution of each component of MoDD. Two base models are considered: the vanilla Transformer and Transolver. We compare five attention strategies: None (i.e., standard attention block of each base model), the recently proposed Differential Transformer (DiffAttn [61]), Inter-layer Differentiation (our ablation model), Mixture of Depths (MoD, our ablation model), and our MoDD. The study conducts the evaluation on the Elasticity and Airfoil datasets. We note that the batch size is reduced to three for all ablation cases with the Airfoil dataset when using vanilla Transformer, due to the GPU memory limits.

Table 4: Ablation study. Five attention strategies, i.e., none, DiffAttn [61], inter-layer differentiation, MoD, and MoDD (ours), are applied to two base models: vanilla Transformer and Transolver.

Base model	Attention strategy	Relative ℓ^2-error [$\times 10^{-2}$]	
		Elasticity	Airfoil
Vanilla Transformer	None	0.62	0.75
	DiffAttn [61]	0.46	0.70
	Inter-layer Diff.	0.57	0.75
	MoD	0.49	0.68
	MoDD (ours)	**0.40**	**0.62**
Promotion w.r.t. base model		**+35.5%**	**+17.3%**
Transolver (SOTA)	None	0.63	0.53
	DiffAttn [61]	0.58	0.46
	Inter-layer Diff.	0.63	0.52
	MoD	0.59	0.49
	MoDD (ours)	**0.56**	**0.43**
Promotion w.r.t. base model		**+11.1%**	**+18.9%**

As shown in Table 4, MoDD achieves the best accuracy across both base models and datasets, improving the vanilla Transformer by 35.5% on Elasticity and 17.3% on Airfoil over the standard attention baseline (denoted as "None"). Detailed efficiency comparisons, including parameter count, GPU memory usage, and training time, are provided in Appendix B.5. To better understand the advantage of MoDD structure, we discuss the effectiveness of each variant. Differential Transformer (DiffAttn [61]) performs intra-layer attention-score differencing, but this intra-layer mechanism underperforms MoDD on the vanilla Transformer while requiring 1.53$\times$ more GPU memory and 2.59$\times$ longer training time, supporting the advantage of MoDD's inter-layer kernel-level design. Additional cross-head differencing controls also underperform MoDD under the same Transformer backbone (Appendix B.6). Inter-layer Differentiation applies layer-wise differencing without depth-wise mixture and yields limited gains, suggesting that differencing alone is insufficient and becomes effective within depth-wise multi-kernel construction. MoD aggregates preceding-layer attention features without differential operations. Its improvement over the baseline indicates the benefit of multi-kernel construction, while MoDD improves it by decorrelating cross-layer kernels through layer-wise differentials.

Although the same MoDD operation can be applied at different stages of the attention pipeline, its kernel-level mechanism critically depends on the application target as shown in Table 5. When applied to the input representation X, MoDD reduces to a pullback of a single kernel (see kernel proof in Appendix F.6) due to the use of a single projection matrix at the current layer, and consequently yields no performance improvement. This confirms that a simple aggregation from previous layers, analogous to skip connections, is insufficient to explain MoDD's gains. In contrast, applying MoDD to Q/K/V realizes a multi-kernel mechanism across depth (see proof in Appendix G.8), leading to consistent accuracy gains. These results indicate that MoDD operates via a multi-kernel mechanism rather than mere depth-wise feature reuse; accordingly, we adopt Q/K/V as the default MoDD targets.

Table 5: Ablation study on the application target (input representation X and Q/K/V representations) of MoDD operation in Transolver. Lower values indicate better performance.

Base model	MoDD target	Relative ℓ^2-error [$\times 10^{-2}$]	
		Elasticity	Airfoil
Transolver (SOTA)	None	0.63	0.53
	Input X	0.63	0.53
	Q/K/V	0.56	0.43

4.5 Model analysis

In this section, we examine the internal representations and representation flow in MoDD, investigating the attention layers to understand how MoDD achieves the superior performance for our given tasks. First, we empirically examine Theorem 3.1 by showing layer-wise differentiation induces weakly correlated attention kernels across depth. Second, we examine the 2D routing architecture formed by combining MoDD with spatial-routing-based solvers to assess its route utilization. Third, we perform the spectral analysis on attention matrices, which will assess the expressiveness of the representations in depth. Fourth, the spatial analysis is conducted, which will assess the level of both

local and global attentions. Finally, we investigate the physics awareness to assess whether the model is promoted by learning the physically grounded attention behavior.

Base kernel alignment To empirically support Theorem 3.1, we analyze the depth-wise alignment of base attention kernels prior to gated aggregation using centered kernel alignment (CKA). On Elasticity, Figure 5 shows that kernels constructed without layer-wise differentiation exhibit high off-diagonal alignment, whereas layer-wise differentiation concentrates alignment values near zero, indicating suppressed cross-layer redundancy and weakly correlated kernels. On Navier–Stokes, adding MoDD to Transolver reduces the mean off-diagonal alignment of base attention kernels from 0.194 to 0.072.

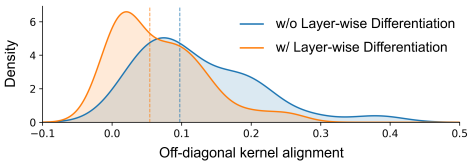


Figure 5: Kernel density estimates of off-diagonal alignment values between base attention kernels constructed with and without layer-wise differentiation, measured prior to gated aggregation. Dashed lines indicate mean off-diagonal alignment per model.

These observations provide direct empirical support for the reduced cross-layer kernel alignment predicted by Theorem 3.1, and justify modeling the depth as a hierarchy of weakly correlated kernels. Additional analyses, including layer-wise shared subspace structure, feature-level correlation, and base-kernel alignment, are provided in Appendices F.2–F.5. Using the mixture of these base kernels, MoDD supports long-range depth-wise connectivity and stable inter-layer representation propagation, as evidenced by gating behavior and CKA analysis in Appendix F.7.

2D spatial–depth routing From the routing perspective, MoDD adds a depth-routing axis to spatial routing-based solvers. In Transolver, token-to-slice spatial routing assigns input tokens to a fixed number of learned slice tokens. Combined with MoDD, this forms a 2D router over spatial slices and network depth. To evaluate whether this 2D router achieves balanced and non-collapsed route utilization, we analyze routing statistics across both axes.

In Table 6, Transolver with 16 slices shows a spatial bottleneck: low assignment entropy and many dead slices indicate collapsed routing, by which only a limited subset of slices is effectively used. MoDD alleviates this bottleneck by increasing assignment entropy, reducing dead slices, and lowering the prediction error. The high depth-gate entropy further shows that the depth routing does not collapse to a single layer. Moreover, compared with layer-wise dense feature aggregation, MoDD distributes depth-wise gates more evenly across layers, indicating improved cross-layer route utilization (Appendix F.7). These results support a 2D routing view in which MoDD improves utilization along both axes: Transolver-style intra-layer spatial routing and inter-layer depth routing. We further observe similarly non-collapsed depth routing on Navier–Stokes, suggesting that this behavior extends beyond the Airfoil benchmark. See Appendix F.8 for details.

Table 6: 2D router behavior combining Transolver's spatial routing (slice entropy, dead-slice ratio) and MoDD's depth routing (gate entropy).

Model	Slices	Rel. ℓ_2 ($\times 10^{-2}$)	Spatial		Depth
			Ent. ↑	Dead ↓	Gate ↑
Transolver	16	0.57	0.23	0.45	–
	32	0.54	0.43	0.40	–
	64	0.53	0.40	0.43	–
Transolver + MoDD	16	0.45	0.40	0.32	0.81
	32	0.46	0.50	0.29	0.87
	64	0.43	0.49	0.30	0.82

Spectral property and rank behavior The strong CKA similarity of MoDD's inter-layer representations (Appendix F.7) indicates a stable propagation of preceding representations and effective feature reuse; however, it is conceivable that the feature space is saturated or learns redundancy, as discussed in previous work [39]. To assess this, we perform spectral analysis on the attention matrices to examine whether MoDD maintains sufficient representational diversity across layers.

Figure 6 shows the eigenvalues of the attention score matrix for every head of two attention blocks, i.e., Blocks 1 and 2, where we can notice meaningful characteristics between the two trained models. (The entire plot is shown in Appendix F.9.) In vanilla Transformer, we observe that dominant outlier eigenvalues for each head are placed at one, while the others are compressed near the origin. This presents a large spectral gap, which is defined as the difference between the largest and second-largest eigenvalues [45]. Such a large gap potentially indicates that the attention operator behaves approximately as a rank-one projection. Consequently, the diverse token representations collapse into a narrow subspace, which can cause unstable gradient flow problem [40].

In contrast, MoDD exhibits a markedly flatter spectrum in both blocks, where the spectral gap is effectively eliminated. This spectral behavior is consistently observed across multiple test samples. Importantly, we observe similar spectral degeneracy even in Transolver across different benchmarks, reaffirming the general prevalence of this issue (see Appendix F.9). When MoDD is integrated into this solver, it consistently mitigates the spectral collapse, highlighting its robustness and generalizability in preserving representational diversity for our tasks.

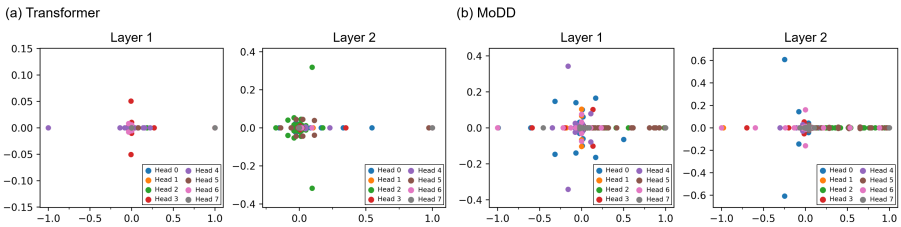


Figure 6: Spectral analysis of attention score matrices for (a) Transformer and (b) Transformer with MoDD (ours), evaluated on the same sample from the Elasticity dataset. The eigenvalues of attention score matrices for all attention heads of two selected layers are plotted in the complex plane, where the x-axis represents the real part and the y-axis represents the imaginary part.

Attention distance To understand the spatial behavior of our attention strategy, we perform a spatial analysis of attention blocks and heads using the mean attention distance, as proposed in prior work [44]. In short, the smaller value indicates local attention, while the greater value means global attention. Further details of the distance evaluation are shown in Appendix F.10.

Figure 7 shows the results for each block and head of three models. Note that the attention heads in the x-axis are sorted in ascending order. The vanilla Transformer drastically loses the spatial coverage in downstream layers. Particularly, the multi-heads of Block 0 cover a wide spatial range, i.e., approximately $[0.27, 0.96]$; then, in Blocks 1 and 2, the coverage significantly narrows down. This behavior corresponds to the reduced spectral diversity of the two blocks observed in the aforementioned spectral analysis, reflecting again a pronounced collapse in attention diversity. On the other hand, MoD preserves local attention at depth but lacks global coverage, with half the heads having low mean attention distances (i.e., $d < 0.3$) and none of them exceeds 0.8 across all blocks.

MoDD uniquely exhibits diverse attention distances across heads in all blocks, preserving both local ($d < 0.2$) and global ($d > 0.6$) coverages. This head-wise diversity reflects MoDD's ability to maintain functional heterogeneity and spatial awareness across attention blocks. Furthermore, MoDD exhibits a consistent depthwise progression in coverage, presenting smooth and gradual evolution of head-wise attention distances. In contrast, other models show collapsed or irregular evolution.

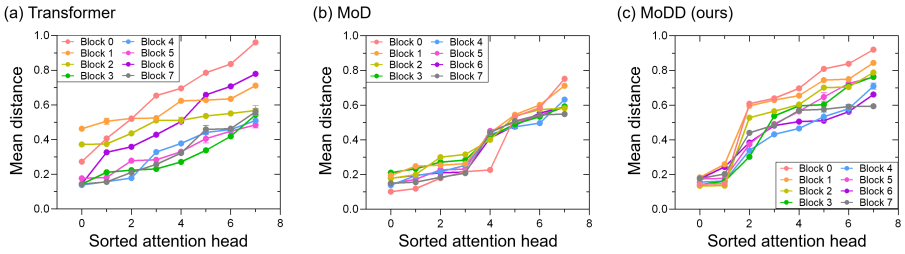


Figure 7: Mean attention distances of multi-heads for three different models: (a) Transformer, (b) Transformer with MoD (an ablation variant), and (c) Transformer with MoDD (ours). For each attention head, the attention distances between query and key positions are evaluated, weighted by the attention scores, and averaged over 194,400 data points.

Physics awareness We further analyze the physics awareness of our model on Elasticity. This analysis investigates which input regions of tokens contribute most to the task, i.e., stress predictions, examining regions near the void boundary, where high stress concentration is observed. To this end, we compute gradient-based Effective Receptive Fields (ERFs). See Appendix F for details.

Figure 8 shows the error maps of two models and their selected ERF maps for representative prediction, where the queried location is indicated by a red star. We observe that MoDD progressively sharpens

receptive fields across downstream layers and aligns them with physically relevant regions. The void boundary regions are highlighted accordingly. We also highlight red circles near the void boundary where stress concentration occurs. Notably, the top-left boundary near the void (A) exhibits the highest stress, and a corresponding high-stress region emerges at the bottom-right (A′) naturally due to geometric symmetry. MoDD accurately captures both regions, with gradients strongly localized around these stress-critical zones. These behaviors indicate that MoDD learns the characteristic physical phenomena [43] of the given physical problem. In contrast, Transformer exhibits broader and less structured receptive fields, with gradients dispersing toward unrelated regions, potentially indicating feature redundancy and weak physics-aware learning. We further confirm this with other void shape cases (See Appendix F.11). Error maps also show that MoDD achieves better prediction, especially with lower errors around the void boundary. This alignment between the localized ERF and physically meaningful regions highlights that MoDD not only enhances prediction accuracy but also facilitates physics-aware learning throughout attention layers.

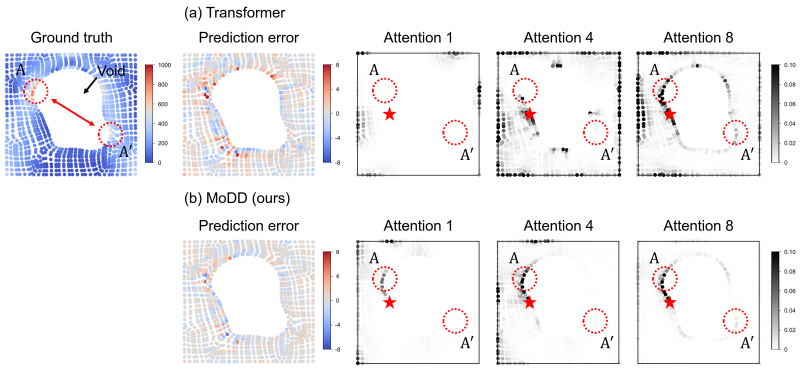


Figure 8: Ground truth stress fields and prediction error maps of two trained models: (a) Transformer and (b) Transformer with MoDD (ours), for an Elasticity benchmark task. When predicting a solution at the specific input coordinates marked with a red star, the effective receptive fields of the selected attention layers are visualized. The maximum stress region A and its symmetric counterpart A′ are marked with red dashed circles. Each ERF is computed as the gradient magnitude of the output after attention with the residual connection.

5 Conclusion

This paper proposes MoDD, a cross-layer attention framework that explicitly treats Transformer depth as a computational dimension rather than a passive stacking parameter. By aggregating inter-layer differentials, MoDD constructs a depth-wise multi-kernel attention representation that enables effective information coupling across layers. Unlike inter-layer information mixing methods that aggregate layer-wise representations themselves, MoDD performs kernel-level differential mixing and achieves superior predictive accuracy. Our experiments demonstrate MoDD's consistent performance improvements when integrated into Transformers and several state-of-the-art neural PDE solvers. While MoDD provides empirical gains, further understanding when it most effectively reduces cross-layer redundancy remains an important future direction. Our future work includes a detailed analysis of potential failure modes, such as overly diffuse or sharp gates that weaken depth-wise mixture behavior as well as large-magnitude differentials that may affect optimization stability. Extending MoDD to other domains, such as natural language processing and computer vision, is another promising direction.

Acknowledgments and Disclosure of Funding

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2026-25521426, Development of XR-Based Remote Control Technology for Facilities Operating in Extreme Environments), the Korea Institute of Science and Technology (KIST) Institutional Program (Project No. 26E0212), and grants from the Institute of Information & Communications Technology Planning & Evaluation (IITP), funded by the Korea government (MSIT), under the SW StarLab Program (No. RS-2020-II200861), the ICT Creative Consilience Program (No. IITP-2026-RS-2020-II201819), and the Information Technology Research Center (ITRC) Program (No. IITP-2026-RS-2020-II201460), and the European Union's HORIZON MSCA Doctoral Networks programme under Grant Agreement No. 101072344, project AQTIVATE (Advanced computing, QuanTum algorIthms and data-driVen Approaches for science, Technology and Engineering). The authors declare no competing interests.

References

[1] Sergio A Alvarez. Gaussian rbf centered kernel alignment (cka) in the large-bandwidth limit. *IEEE transactions on pattern analysis and machine intelligence*, 45(5):6587–6593, 2022.

[2] Mohamad Bayat, Wen Dong, Jesper Thorborg, Albert C To, and Jesper H Hattel. A review of multi-scale and multi-physics simulations of metal additive manufacturing processes with focus on modeling strategies. *Additive Manufacturing*, 47:102278, 2021.

[3] Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. *Advances in Neural Information Processing Systems*, 35:23463–23478, 2022.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Shuhao Cao. Choose a transformer: Fourier or galerkin. *Advances in neural information processing systems*, 34:24924–24940, 2021.

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[7] Xiang Cheng, Yuxin Chen, and Suvrit Sra. Transformers implement functional gradient descent to learn non-linear functions in context. *arXiv preprint arXiv:2312.06528*, 2023.

[8] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

[9] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13(1):795–828, 2012.

[10] Angela Fan, Thibaut Lavril, Edouard Grave, Armand Joulin, and Sainbayar Sukhbaatar. Addressing some limitations of transformers with feedback memory. *arXiv preprint arXiv:2002.09402*, 2020.

[11] Yanwen Fang, Yuxi Cai, Jintai Chen, Jingyu Zhao, Guangjian Tian, and Guodong Li. Cross-layer retrospective retrieving via layer attention. *arXiv preprint arXiv:2302.03985*, 2023.

[12] Somdatta Goswami, Katiana Kontolati, Michael D Shields, and George Em Karniadakis. Deep transfer operator learning for partial differential equations under conditional shift. *Nature Machine Intelligence*, 4(12):1155–1164, 2022.

[13] Arthur Gretton, Kenji Fukumizu, Choon Teo, Le Song, Bernhard Schölkopf, and Alex Smola. A kernel statistical test of independence. *Advances in neural information processing systems*, 20, 2007.

[14] Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. Gnot: A general neural operator transformer for operator learning. In *International Conference on Machine Learning*, pages 12556–12569. PMLR, 2023.

[15] Jane K Hart and Kirk Martinez. Environmental sensor networks: A revolution in the earth system science? *Earth-Science Reviews*, 78(3-4):177–191, 2006.

[16] Ruining He, Anirudh Ravula, Bhargav Kanagal, and Joshua Ainslie. Realformer: Transformer likes residual attention. *arXiv preprint arXiv:2012.11747*, 2020.

[17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[18] Zhongyi Jiang, Min Zhu, and Lu Lu. Fourier-mionet: Fourier-enhanced multiple-input neural operators for multiphase modeling of geological carbon sequestration. *Reliability Engineering & System Safety*, 251:110392, 2024.

[19] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[20] Kimi Team, Guangyu Chen, Yu Zhang, Jianlin Su, Weixin Xu, Siyuan Pan, Yaoyu Wang, Yucheng Wang, Guanduo Chen, Bohong Yin, et al. Attention residuals. *arXiv preprint arXiv:2603.15031*, 2026.

[21] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

[22] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[23] JDMCK Lee and K Toutanova. Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 3(8):4171–4186, 2018.

[24] Fanny Lehmann, Filippo Gatti, Michaël Bertin, and Didier Clouteau. 3d elastic wave propagation with a factorized fourier neural operator (f-fno). *Computer Methods in Applied Mechanics and Engineering*, 420:116718, 2024.

[25] Zhihao Li, Haoze Song, Di Xiao, Zhilu Lai, and Wei Wang. Harnessing scale and physics: A multi-graph neural operator framework for pdes on arbitrary geometries. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 729–740, 2025.

[26] Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *arXiv preprint arXiv:2205.13671*, 2022.

[27] Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for pde surrogate modeling. *Advances in Neural Information Processing Systems*, 36:28010–28039, 2023.

[28] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

[29] Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

[30] Qibang Liu, Vincient Zhong, Hadi Meidani, Diab Abueidda, Seid Koric, and Philippe Geubelle. Geometry-informed neural operator transformer. *arXiv preprint arXiv:2504.19452*, 2025.

[31] Songming Liu, Zhongkai Hao, Chengyang Ying, Hang Su, Ze Cheng, and Jun Zhu. Nuno: A general framework for learning parametric pdes with non-uniform data. In *International Conference on Machine Learning*, pages 21658–21671. PMLR, 2023.

[32] Xinliang Liu, Bo Xu, and Lei Zhang. Ht-net: Hierarchical transformer based operator learning model for multiscale pdes. 2022.

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[35] Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

[36] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

[37] Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for pdes on million-scale geometries. *arXiv preprint arXiv:2502.02414*, 2025.

[38] Hans Z Munthe-Kaas, Olivier Verdier, and Gilles Vilmart. A short proof of isserlis' theorem. *arXiv preprint arXiv:2503.01588*, 2025.

[39] Thao Nguyen, Maithra Raghu, and Simon Kornblith. Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth. *arXiv preprint arXiv:2010.15327*, 2020.

[40] Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. *Advances in Neural Information Processing Systems*, 35:27198–27211, 2022.

[41] Matteo Pagliardini, Amirkeivan Mohtashami, Francois Fleuret, and Martin Jaggi. Denseformer: Enhancing information flow in transformers via depth weighted averaging. *Advances in neural information processing systems*, 37:136479–136508, 2024.

[42] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.

[43] W.D. Pilkey, D.F. Pilkey, and Z. Bi. *HOLES*, chapter 4, pages 209–437. John Wiley & Sons, Ltd, 2020. ISBN 9781119532552. doi: 10.1002/9781119532552.ch4. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119532552.ch4`.

[44] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in neural information processing systems*, 34:12116–12128, 2021.

[45] Thiziri Nait Saada, Alireza Naderi, and Jared Tanner. Mind the gap: a spectral analysis of rank collapse and signal propagation in attention layers. *arXiv preprint arXiv:2410.07799*, 2024.

[46] Clayton Sanford, Daniel J Hsu, and Matus Telgarsky. Representational strengths and limitations of transformers. *Advances in Neural Information Processing Systems*, 36:36677–36707, 2023.

[47] Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. Feature selection via dependence maximization. *The Journal of Machine Learning Research*, 13(1):1393–1434, 2012.

[48] Yehui Tang, Kai Han, Chang Xu, An Xiao, Yiping Deng, Chao Xu, and Yunhe Wang. Augmented shortcuts for vision transformers. *Advances in Neural Information Processing Systems*, 34:15316–15327, 2021.

[49] Rachel SY Teo and Tan Nguyen. Unveiling the hidden structure of self-attention via kernel principal component analysis. *Advances in Neural Information Processing Systems*, 37:101393–101427, 2024.

[50] Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. *arXiv preprint arXiv:2111.13802*, 2021.

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[52] Kaishen Wang, Xun Xia, Jian Liu, Zhang Yi, and Tao He. Strengthening layer interaction via dynamic layer attention. *arXiv preprint arXiv:2406.13392*, 2024.

[53] Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems. *Advances in Neural Information Processing Systems*, 37:33085–33107, 2024.

[54] Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M Benson. U-fno—an enhanced fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022.

[55] Haixu Wu, Tengge Hu, Huakun Luo, Jianmin Wang, and Mingsheng Long. Solving high-dimensional pdes with latent spectral models. *arXiv preprint arXiv:2301.12664*, 2023.

[56] Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*, 2024.

[57] Da Xiao, Qingye Meng, Shengping Li, and Xingyuan Yuan. Muddformer: Breaking residual bottlenecks in transformers via multiway dynamic dense connections. *arXiv preprint arXiv:2502.12170*, 2025.

[58] Zipeng Xiao, Zhongkai Hao, Bokai Lin, Zhijie Deng, and Hang Su. Improved operator learning by orthogonal attention. *arXiv preprint arXiv:2310.12487*, 2023.

[59] Hanqi Yan, Lin Gui, Wenjie Li, and Yulan He. Addressing token uniformity in transformers via singular value transformation. In *Uncertainty in artificial intelligence*, pages 2181–2191. PMLR, 2022.

[60] Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W Cohen. Breaking the softmax bottleneck: A high-rank rnn language model. *arXiv preprint arXiv:1711.03953*, 2017.

[61] Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=OvoCm1gGhN.

[62] Huaiqian You, Quinn Zhang, Colton J Ross, Chung-Hao Lee, and Yue Yu. Learning deep implicit fourier neural operators (ifnos) with applications to heterogeneous material modeling. *Computer Methods in Applied Mechanics and Engineering*, 398:115296, 2022.

[63] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021.

[64] Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. Hyper-connections. In *International Conference on Learning Representations*, 2025.

A Implementation Details

In this section, we provide the details of our experiments, including benchmarks and implementations.

A.1 Benchmarks

We evaluate our model on eight standard benchmark datasets [28, 29, 6, 42] in operator learning, summarized in Table 7. These benchmarks cover diverse physical domains (solid mechanics, fluid dynamics, and porous media flow) and geometry types (point clouds, structured meshes, and regular grids).

Table 7: Summary of benchmark datasets. #Mesh denotes the number of discrete mesh points. #Dataset represents the number of samples in the training and testing sets.

Geometry	Benchmarks	#Dim	#Mesh	#Input	#Output	#Dataset
Point Cloud	Elasticity	2D	972	Structure	Inner Stress	(1000, 200)
	Shape-Net Car	3D+Time	32,186	Past Fluid Quantity	Future Fluid Quantity	(690, 100)
Structured Mesh	Plasticity	2D+Time	3,131	External Force	Mesh Displacement	(900, 80)
	Airfoil	2D	11,271	Structure	Mach Number	(1000, 200)
	Pipe	2D	16,641	Structure	Fluid Velocity	(1000, 200)
	Deforming Plate	3D+Time	–	Initial Stress	Stress Field	(24000, 2000)
Regular Grid	Navier–Stokes	2D+Time	4,096	Past Velocity	Future Velocity	(1000, 200)
	Darcy	2D	7,225	Porous Medium	Fluid Pressure	(1000, 200)

Elasticity This benchmark aims to predict the internal stress distribution within a hyperelastic solid, given the 2D spatial coordinates of mesh points. Each sample corresponds to a unit square domain that contains a circular cavity whose size and position vary, introducing internal structural variation. This variation induces mechanical stress patterns across the domain. The material is clamped along the bottom boundary (zero displacement), and a uniform vertical force is applied at the top. The input is a tensor of shape 972×2, representing the 2D coordinates of 972 mesh points, and the output is a tensor of shape 972×1, denoting the corresponding stress values. The task is to learn the mapping from the geometric configuration to the resulting stress field. The dataset contains 1000 training samples and 200 test samples.

Plasticity This benchmark focuses on predicting the time-dependent elasto-plastic deformation of a solid block subjected to compression by a rigid die. The task is to forecast a sequence of deformation fields over 20 future time steps, based solely on the geometry of the die. The input is a structured mesh of size 101×31 encoding the die shape, and the output is a tensor of shape $20 \times 101 \times 31 \times 4$, where the four channels represent displacements and other derived physical quantities. In this setup, the die presses vertically downward with constant velocity, while the block is fixed at its base. The resulting deformation behavior combines both elastic and plastic responses. The dataset consists of 900 training samples and 80 test samples.

Airfoil This benchmark aims to predict the Mach number distribution over a 2D airfoil surface, given only the geometric configuration of the airfoil. The input is a structured O-type mesh of size 221×51 representing the deformed geometry based on a base NACA-0012 profile, and the output is a tensor of shape $221 \times 51 \times 1$ indicating the Mach number at each mesh point. The underlying physics is governed by the Euler equations for transonic, inviscid flow. Freestream inflow conditions are imposed at the inlet, and slip boundary conditions are applied along the airfoil surface to ensure tangential flow. This setup captures critical flow phenomena such as shock waves and compressibility effects near the surface. The dataset contains 1000 training samples and 200 test samples.

Pipe This benchmark focuses on predicting the steady-state horizontal velocity field of viscous flow in a 2D pipe-like geometry, given the spatial mesh configuration. The input is a structured mesh of size 129×129, where each point contains its 2D spatial coordinates, forming a tensor of shape $129 \times 129 \times 2$. The output is a tensor of shape $129 \times 129 \times 1$, representing the steady-state horizontal

velocity at each point. The flow is governed by the incompressible Navier–Stokes equations. Each sample features a different pipe geometry created by perturbing the domain's centerline, which affects the internal flow behavior. A parabolic inflow profile is applied at the inlet (left boundary), no-slip conditions are enforced on the upper and lower walls, and a zero-pressure condition is set at the outlet (right boundary). The dataset includes 1000 training samples and 200 test samples.

Navier-Stokes This benchmark aims to forecast future vorticity fields in a 2D incompressible, viscous fluid by using partial temporal observations of past vorticity. The input is a tensor of shape $64 \times 64 \times 10$, representing the vorticity field over the previous 10 time steps, and the output is a tensor of the same shape for the next 10 time steps. The flow evolves on a unit torus domain governed by the 2D incompressible Navier–Stokes equations, with constant fluid density and viscosity set to $\nu = 10^{-5}$, producing turbulent dynamics. The domain is discretized into a 64×64 regular grid with periodic boundary conditions in both spatial directions. Each sample represents a spatiotemporal trajectory of fluid motion generated from unique initial conditions. The dataset includes 1000 training and 200 test samples.

Darcy This benchmark predicts steady-state pressure fields resulting from fluid flow through porous media, given the spatial structure of the medium. The input is a tensor of shape $85 \times 85 \times 1$ representing the spatially varying diffusion coefficient (permeability), and the output is a tensor of the same shape denoting the corresponding pressure field. The underlying physics is governed by the Darcy equation, a second-order elliptic partial differential equation. Each sample is defined on a unit square domain initially discretized to a 421×421 grid and downsampled to 85×85 for learning. The porous medium is generated as a binary Gaussian random field, while a fixed forcing term drives the flow and Dirichlet boundary conditions (zero pressure) are applied on all edges. The dataset includes 1000 training and 200 test samples, each corresponding to a distinct porous configuration.

Deforming Plate This benchmark considers a 3D dynamic simulation of a deformable plate with hyperelastic material properties. The task is to predict the future stress evolution of the plate based on its initial stress condition. Each sample consists of a temporal sequence spanning 50 time steps. The input corresponds to the initial stress field of the plate, while the output is the predicted stress field at subsequent time steps. The dataset contains 24,000 training samples, along with 2,000 samples each for validation and testing, all generated from diverse initial stress configurations to ensure a wide range of deformation behaviors.

Shape-Net Car This benchmark models the temporal evolution of fluid dynamics around cars simulated at a constant speed of 72 km/h. The dataset consists of 889 samples with distinct car geometries, split into 690 training, 99 validation, and 100 test samples. The airflow domain is discretized using an unstructured mesh, where fluid quantities, including velocity and pressure, are recorded. The learning task is formulated as a one-step prediction problem, in which the model predicts fluid quantities at the next timestep given those at the current timestep. The predicted velocity and pressure fields are subsequently used to compute the aerodynamic drag coefficient for evaluation.

A.2 Implementations

This section outlines the implementation details of both the baseline Transformer solvers and the attention variants evaluated in this work.

Standard Transformer solvers We use three Transformer-based models as baselines: the vanilla Transformer and the state-of-the-art Transolver [56] and AMG [25]. The vanilla Transformer represents the most generic form of Transformer architecture [51], while Transolver and AMG incorporate recent advances tailored for PDE modeling and achieves state-of-the-art performance. These three baselines provide a means to assess the applicability of our MoDD in standard settings and its complementary effect when integrated into representative SOTA solvers spanning both Transformer-based and graph-based architectures.

The vanilla Transformer adopts the standard encoder block of the original Transformer, consisting of multi-head self-attention with scaled dot-product attention and softmax activation, followed by position-wise feed-forward layers. In contrast, Transolver introduces physics-aware components, including slice tokens to capture structured mesh information and a hierarchical attention mechanism

for improved efficiency. AMG adopts a graph-based formulation that combines local, global, and physics-aware graph attention modules to model multi-scale interactions on irregular geometries. We follow the original implementation of Transolver and AMG, as summarized in Table 8. These base models serve as the foundation for integrating and evaluating all proposed attention variants.

Transformer variants To evaluate the effectiveness of our proposed MoDD, we conduct ablation studies by applying three variants, i.e., DiffAttn [61], MoD (our ablation variant), and MoDD (ours), to two baseline architectures: vanilla Transformer and Transolver, as described in the **Standard Transformer solvers** section above. To this end, each variant is integrated into both baseline models by replacing their original attention modules. In addition to the ablation studies, we apply MoDD to representative state-of-the-art solvers, including Transolver and AMG, and evaluate them on PDE benchmarks. For MoDD, the layer-difference and gating-weight tensors have shapes of $[B, L-1, H, T, D]$ and $[B, L-1, H, T]$, respectively, where B, H, T, and D denote the batch, head, token, and per-head feature dimensions. Each gating weight is broadcast over the feature dimension D. The integration strategy of each variant is described in detail below.

Integration strategies. Firstly, the integration of our MoDD into Transolver, featuring physics-aware slice tokens instead of standard tokens, is illustrated in Figure 9. The original Physics-Attention module in Transolver [56] performs three stages: slicing the input into slice tokens, computing attention among these tokens, and deslicing the output back to the original spatial domain using learned slice weights. The first layer operates as in the original Transolver, without applying MoDD. Starting from the second layer, we compute layer-wise differences (ΔQ, ΔK, ΔV) from the query, key, value representations of the slice tokens at each layer. To guide the aggregation of these inter-layer differences, we stack slice tokens from all previous layers and use them to compute soft gate weights through a learnable projection. These gate weights are then used to compute a weighted sum of ΔQ, ΔK, and ΔV. Finally, the output of MoDD is passed through multi-head attention and de-sliced back to the original domain using the slice weights of the current layer.

Secondly, the integration of our MoD into Transolver follows the same overall structure as MoDD but omits the computation of layer-wise differences. Instead, we directly stack the query, key, and value representations of the slice tokens from all previous layers. Soft gate weights are derived from the stacked slice tokens via a learnable projection and used to compute a weighted sum of the stacked query, key, and value features. The result is then processed by multi-head attention and de-sliced back to the original domain.

Thirdly, the integration of DiffAttn [61] into Transolver applies intra-layer differentiation to the attention score matrix within the slice token attention module. We adopt the official implementation of multi-head differential attention without modification. Briefly, two sets of query–key projections are generated to compute two separate attention score matrices. One score matrix is scaled by a learnable coefficient and subtracted from the other, resulting in a contrastive attention map within each Transolver layer.

Fourthly, we integrate MoDD into the AMG architecture, which consists of three graph attention modules operating on local, global, and physics graphs. In AMG, the local and global GraphFormer modules are implemented using the GATv2Conv operator from PyTorch Geometric, which performs edge-based attention without explicit query, key, and value representations. Consequently, MoDD is applied only to the Physics GraphFormer, which employs Transformer-style self-attention over virtual physics nodes with explicit query, key, and value parameterization. The first multi-graph processing layer follows the original AMG design. From the second layer onward, we compute layer-wise differences $(\Delta Q, \Delta K, \Delta V)$ from the physics-node representations. These inter-layer differences are aggregated by stacking query, key, and value representations from previous layers and computing soft gate weights via a learnable linear projection. The resulting weighted differences are processed by multi-head self-attention among physics nodes and propagated back to spatial nodes through the original physics-to-node message passing, while the local and global modules remain unchanged.

These three variants are identically applied to the Vanilla Transformer baseline by replacing the original attention module. As slice tokens are not used, the stacking and gating operations are applied to standard query, key, and value representations. In particular, the integration of MoDD into the Vanilla Transformer is described in detail in Section 3 of the manuscript.

Model configuration. As shown in Table 8, both Transolver and Transolver with MoDD (ours) are trained and tested using the same training strategy and model architecture as the original Transolver

for performance evaluation on eight benchmark datasets. Notably, for structured mesh benchmarks (i.e., Navier-Stokes and Darcy), Transolver employs a unique positional encoding by computing the Euclidean distances from each 2D spatial coordinate to a predefined reference grid. Each 2-dimensional point is thereby embedded into a 64-dimensional positional vector, which is concatenated with the input physical quantity and subsequently processed by the network. Its integration prior to the physics-attention blocks may implicitly introduce an inductive bias beneficial to learning complex physical states. For Transolver with our MoDD, we adopt the same encoding scheme but reduce the positional dimension to 36, as the weighted aggregation mechanism in MoDD inherently enhances rank expressiveness, allowing us to avoid over-parameterization. To ensure a fair comparison, a matched positional-encoding comparison in Appendix B.4 confirms that the regular-grid gains are not explained by this positional-encoding difference.

As summarized in Table 9, we further apply the proposed MoDD operator to the state-of-the-art AMG solver [25]. Both AMG and AMG with MoDD (ours) are trained and tested using the same training strategy and model architecture as the original AMG for performance evaluation on four benchmark datasets.

For ablation study, Table 10 summarizes the configurations used to evaluate the proposed variants. All variants follow the same training and model configurations as their respective base models to ensure a fair comparison. The only differences are that slice tokens are used exclusively in Transolver-based models, and that the batch size is adjusted based on GPU memory limits.

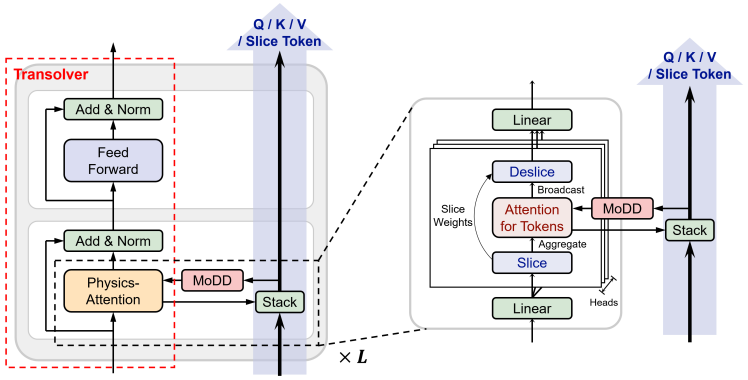


Figure 9: Architecture of the SOTA solver, Transolver [56] with MoDD, leveraging a cross-layer path of query, key, value and slice token.

Table 8: Training and model configurations of a SOTA solver Transolver [56] and Transolver with MoDD (ours). Training settings, including the AdamW optimizer [34], follow previous works [3, 14] without additional tuning. Model configurations are directly adopted from the official Transolver paper [56], except for Deforming Plate and Shape-Net Car benchmarks. The Darcy benchmark uses the relative ℓ^2 loss with an additional gradient regularization term $\mathcal{L}_g$, following [58].

Benchmarks	Training Configuration					Model Configuration			
	Loss	Epochs	Initial LR	Optimizer	Batch Size	Layers	Heads	Channels C	Slices M
Elasticity	relative ℓ^2	500	10^{-3}	AdamW	1	8	8	128	64
Plasticity					8			128	64
Airfoil					4			128	64
Pipe					4			128	64
Navier-Stokes					2			256	32
Deforming Plate					16			128	32
Shape-Net Car					1			256	32
Darcy	$\mathcal{L}_{\text{relative } \ell^2} + 0.1\mathcal{L}_g$				4			128	64

Table 9: Training and model configurations of a SOTA solver AMG [25] and AMG with MoDD (ours). Training settings and model configurations are directly adopted from the official AMG implementation, except that the batch size is reduced due to GPU memory limits.

Benchmarks	Training Configuration					Model Configuration		
	Loss	Epochs	Initial LR	Optimizer	Batch Size	Layers	Heads	Channels C
Darcy	Relative L2	500	10^{-3}	AdamW	8	3	8	128
Navier–Stokes					20	3	8	
Deforming Plate					64	2	8	
Shape-Net Car					4	3	4	

Table 10: Training and model configurations used in the ablation study, including two base models (Vanilla Transformer and Transolver) and their variants with DiffAttn [61], MoD (our ablation), and MoDD (ours). Training settings, including the AdamW optimizer [34], follow previous works [3, 14]. All variants built on the same base model use identical training and model configurations to ensure fair comparison. The batch size for training the Vanilla Transformer on the Airfoil dataset is reduced to 3 due to GPU memory limitations.

Benchmark	Base Model	Training Configuration					Model Configuration			
		Loss	Epochs	Initial LR	Optimizer	Batch Size	Layers	Heads	Channels	Slices
Elasticity	Vanilla Transformer	Relative L2	500	10^{-3}	AdamW	1	8	8	128	–
	Transolver									64
Airfoil	Vanilla Transformer	Relative L2	500	10^{-3}	AdamW	3	8	8	128	–
	Transolver					4				64

Table 11: Evaluation of different models. The means of relative ℓ^2 errors are reported; lower value indicates better accuracy. "/" denotes the inapplicable case. **The best** and second-best results are shown in bold and underlined, respectively. Transolver* is the result reproduced by our implementation and training; the others are the performances reported in the original papers.

Model	**Point cloud**	**Structured mesh**			**Regular grid**	
	Elasticity	Plasticity	Airfoil	Pipe	Navier-Stokes	Darcy
FNO [28]	/	/	/	/	0.1556	0.0108
U-FNO [54]	0.0239	0.0039	0.0269	0.0056	0.2231	0.0183
Geo-FNO [29]	0.0229	0.0074	0.0138	0.0067	0.1556	0.0108
F-FNO [50]	0.0263	0.0047	0.0078	0.0070	0.2322	0.0077
LSM [55]	0.0218	0.0025	0.0059	0.0050	0.1535	0.0065
Galerkin [5]	0.0240	0.0120	0.0118	0.0098	0.1401	0.0084
OFormer [26]	0.0183	0.0017	0.0183	0.0168	0.1705	0.0124
GNOT [14]	0.0086	0.0336	0.0076	0.0047	0.1380	0.0105
FactFormer [27]	/	0.0312	0.0071	0.0060	0.1214	0.0109
ONO [58]	0.0118	0.0048	0.0061	0.0052	0.1195	0.0076
Transolver [56]	0.0064	0.0012	0.0053	**0.0033**	**0.0900**	0.0057
Transolver*	0.0063	0.0013	0.0053	0.0048	0.0949	0.0059
Transolver* + **MoDD (ours)**	**0.0056**	**0.0011**	**0.0043**	0.0040	0.0907	**0.0054**
Promotion w.r.t. Transolver*	11.1%	15.4%	18.9%	16.7%	4.4%	8.5%

B Ablation

B.1 Gating Logit Source

In MoDD, the gating network produces mixture weights over depthwise differentials by projecting a stack of representations to scalar logits. An important design choice is the source of this projection, namely whether the gating logits are derived from queries (Q), keys (K), or values (V).

From a spectral perspective on attention, the key representations play a central role in shaping the structure of the attention operator, as they define the similarity relationships that govern how information is aggregated. In contrast, values are propagated along these relationships, while queries act as selectors over key-defined directions [49]. Although this interpretation does not fully characterize the behavior of deep attention models, it suggests that operating in the key space may provide a more direct handle on the evolution of attention structure across layers. Motivated by this intuition, MoDD computes gating logits from stacked key representations across depth. This choice is intended to better retain diversity in the attention representations and to reduce overly dominant directions, without enforcing strict spectral constraints.

We empirically evaluate this design by comparing three variants in which the gating logits are computed from Q, K, or V, respectively. Table 12 reports the relative ℓ_2 errors on the Elasticity benchmark. Among the three variants, key-based gating achieves the lowest error, indicating that using key representations as the gating source leads to more effective depthwise aggregation in practice.

Table 12: Effect of gating logit source on prediction accuracy.

Gating logit source	Q	K	V
Relative ℓ_2 error [$\times 10^{-2}$]	0.42	0.40	0.45

B.2 Gating Parameterization

The primary design focus of MoDD is its application target, which determines the proposed multi-kernel interpretation, as examined in Table 5. The gating parameterization is orthogonal to this mechanism and can be instantiated with different inter-layer gating strategies. We adopt a shared gate across the Q, K, and V mixtures as the default formulation because it provides coordinated $Q/K/V$ depth selection, incurs lower computational overhead, and matches the shared Q/K gating formulation analyzed in Lemma 3.1.

To examine whether MoDD can accommodate a more flexible gating parameterization, we additionally compare the default shared gating with independent per-stream gating, which is closer to the per-stream weighting used in MUDDFormer [57]. We evaluate Transformer+MoDD on the Elasticity and Airfoil benchmarks using relative ℓ_2 error. We also report parameter count, GPU memory usage, and training time per epoch, all measured on Elasticity under the same Transformer backbone and training protocol.

As shown in Table 13, both gating variants outperform the standard Transformer and the prior inter-layer information mixing methods, including DenseFormer, MUDDFormer, and Full AttnRes, reported in Table 1. This suggests that the main benefit of MoDD is retained under different gating parameterizations. Their relative performance is task-dependent: per-stream gating performs better on Airfoil, whereas shared gating performs better on Elasticity and requires fewer parameters, less GPU memory, and less training time. We therefore retain shared gating as the default formulation for its simplicity, efficiency, and consistency with the current analysis, while viewing per-stream gating as a promising extension.

Table 13: Comparison of the vanilla Transformer baseline and Transformer+MoDD with per-stream and shared gating. Relative ℓ_2 errors ($\times 10^{-2}$) are measured on Elasticity and Airfoil. Parameter count, GPU memory, and training time per epoch are measured on Elasticity under the same backbone and training protocol.

Gating strategy	Param. # [M]	Memory [GB]	Time [s/epoch]	Elasticity	Airfoil
None	0.83	0.87	6.7	0.62	0.75
MoDD w/ per-stream gating	0.84	0.99	10.2	0.43	**0.61**
MoDD w/ shared gating	0.83	0.95	9.2	**0.40**	0.62

B.3 Robustness to Reduced Training Data

We evaluate the sensitivity of MoDD to the amount of training data on the Elasticity benchmark. Starting from the original benchmark setting with 1000 training samples, we construct reduced-data settings by subsampling the training set to 800, 600, 400, and 200 samples. This experiment examines whether the proposed depth-differential attention mechanism remains effective as data availability decreases. Table 14 reports the relative l_2 errors of Transformer and Transformer+MoDD, along with the relative gain and absolute error reduction of MoDD over the Transformer baseline.

Across all training-set sizes, MoDD consistently improves the Transformer baseline. The relative gains remain of a similar order across data regimes, indicating that MoDD provides stable improvements rather than being effective only at a particular training-set size. We therefore do not claim that MoDD yields larger relative gains specifically in low-data settings. Nevertheless, the absolute error reduction is largest in the smallest-data setting with 200 training samples, where MoDD reduces the relative l_2 error from 2.50 to 1.71. These results suggest that the proposed depth-differential mechanism remains effective under reduced data availability.

Table 14: Performance comparison under different training-set sizes on the Elasticity benchmark. Relative l_2 errors are reported in $\times 10^{-2}$.

Model	200	400	600	800	1000 (default)
Transformer	2.50	1.34	1.10	0.71	0.62
Transformer + **MoDD(ours)**	1.71	0.92	0.73	0.53	0.40
Rel. gain	**31.6%**	**31.3%**	**33.6%**	**25.4%**	**35.5%**
Abs. reduction	**0.79**	**0.42**	**0.37**	**0.18**	**0.22**

B.4 Controlled Positional Encoding Comparison

We conduct a controlled comparison to ensure that the improvement of MoDD on the 2D regular-grid benchmarks is not explained by differences in the Transolver-specific positional encoding setting. In the main experiments, Transolver uses its positional encoding scheme for the 2D regular-grid benchmarks, Navier–Stokes and Darcy, and the positional encoding dimension can affect the model capacity. Therefore, we compare Transolver and Transolver+MoDD under the same positional encoding dimension of 36 on these two benchmarks. Table 15 reports the relative l_2 errors averaged over three trials.

The results show that MoDD continues to improve Transolver under the matched positional encoding setting. On Navier–Stokes, Transolver+MoDD reduces the error from 10.35 to 9.07, corresponding to a 12.4% relative gain over Transolver. On Darcy, it reduces the error from 0.60 to 0.54, corresponding to a 10.0% relative gain. These results indicate that the performance gains on the 2D regular-grid benchmarks are not explained by differences in the positional encoding dimension, but remain present under a matched Transolver-specific positional encoding configuration.

Table 15: Comparison under matched positional encoding dimension on 2D regular-grid benchmarks. Relative l_2 errors are reported in $\times 10^{-2}$ and averaged over three trials.

Model	Navier-Stokes	Darcy
Transolver	10.35 ± 0.125	0.60 ± 0.026
Transolver + **MoDD(ours)**	9.07 ± 0.302	0.54 ± 0.010
Gain	**12.4%**	**10.0%**

B.5 Attention Strategies

We perform an ablation study to investigate the contribution of each component of MoDD. Two base models are considered: the vanilla Transformer and Transolver. We compare five different attention strategies: None (i.e., the standard attention block of each base model), Differential Transformer (DiffAttn; ICLR 2025 [61]), Inter-layer Differentiation (our ablation model), Mixture of Depths (MoD, our ablation model), and our MoDD. The study conducts the evaluation on the Elasticity and Airfoil datasets. We note that the batch size is reduced to three for all ablation cases with the Airfoil dataset when using the vanilla Transformer, due to the GPU memory limits.

As shown in Table 16, our MoDD strategy consistently outperforms all the other attention mechanisms across both base models and datasets. With the vanilla Transformer, MoDD yields relative accuracy improvements of 35.5% on Elasticity and 17.3% on Airfoil compared to the base model using standard multi-layer attention blocks (denoted as "None"), without any notable additional parameters. Although MoDD introduces memory and runtime overheads at the same depth, it remains more efficient than recent strategies such as DiffAttn, requiring less memory while achieving better accuracy.

To better understand the advantage of the MoDD architecture, we discuss the effectiveness of each variant. DiffAttn introduces intra-layer attention-score differencing to suppress common attention noise within each attention block, but it does not explicitly exploit depth-wise kernel construction and incurs substantial memory and runtime overhead. Inter-layer Differentiation applies layer-wise differencing without depth-wise mixture and yields limited gains, suggesting that differencing alone is insufficient and becomes effective within depth-wise multi-kernel construction. MoD adopts a depth-wise attention strategy, where attention features are aggregated from preceding layers without differential operations. Thus, MoD is a simplified ablation variant of MoDD. MoD achieves performance comparable to DiffAttn with lower GPU memory usage, underscoring the benefit of reusing depth-wise features. The full MoDD architecture combines depth-wise feature reuse with inter-layer differential operations at the attention-kernel level. As shown in Table 16, this combination leads to the best accuracy across both backbones and datasets.

Table 16: Ablation study. Five attention strategies, i.e., none, Differential Transformer (DiffAttn; ICLR 2025 [61]), inter-layer differentiation (our ablation model), MoD (our ablation model), and MoDD (ours), are applied to two base models: vanilla Transformer and Transolver. Both accuracy and efficiency in parameter count, GPU memory usage, and training time are shown.

Base model	Attention strategy	Param. #	Memory	Time	Relative ℓ^2-error [$\times 10^{-2}$]	
		[M]	[GB]	[s/epoch]	Elasticity	Airfoil
Vanilla Transformer	None	0.83	0.87	6.7	0.62	0.75
	DiffAttn [61]	0.83	1.45	23.8	0.46	0.70
	Inter-layer Diff.	0.83	0.87	7.1	0.57	0.75
	MoD	0.83	0.95	8.5	0.49	0.68
	MoDD (ours)	0.83	0.95	9.2	**0.40**	**0.62**
	Promotion of MoDD w.r.t. base model				**+35.5%**	**+17.3%**
Transolver (SOTA)	None	0.71	0.57	7.7	0.63	0.53
	DiffAttn [61]	0.71	0.58	10.1	0.58	0.46
	Inter-layer Diff.	0.71	0.57	8.4	0.63	0.52
	MoD	0.71	0.58	10.0	0.59	0.49
	MoDD (ours)	0.71	0.58	10.9	**0.56**	**0.43**
	Promotion of MoDD w.r.t. base model				**+11.1%**	**+18.9%**

B.6 Cross-head versus Depth-wise Differencing

We further evaluate whether the benefit of MoDD arises merely from applying a differencing operation or from the depth axis along which differencing is performed. Under the same vanilla Transformer backbone, we compare direct cross-head differencing and a gated mixture of cross-head differences with MoDD.

As shown in Table 4, the direct cross-head differencing substantially degrades performance, suggesting that simple differencing is highly sensitive and may require an adaptive mechanism to control the subtraction, such as the learnable scalar of DiffAttn. When we introduce a gated mixture of cross-head differences, on the other hand, performance improves to a level comparable to DiffAttn [61] and MoD (our ablation). However, it remains less accurate than MoDD, indicating that MoDD's inter-layer differencing is more effective than intra-layer cross-head differencing in our experiments.

Table 17: Comparison of cross-head and depth-wise differencing under the same vanilla Transformer backbone. Relative ℓ_2 errors are reported in $\times 10^{-2}$; computational costs are measured on Elasticity.

Strategy	Param. # [M]	Memory [GB]	Time [s/epoch]	Elasticity	Airfoil
None	0.83	0.87	6.7	0.62	0.75
Cross-head difference	0.83	0.89	9.1	1.11	0.82
Mixture of cross-head differences	0.83	0.89	9.2	0.46	0.67
MoDD (ours)	0.83	0.95	9.2	**0.40**	**0.62**

C Depth Scaling

C.1 Performance Scaling with Model Depth

We evaluate how the benefit of MoDD changes as the Transformer depth increases. Since MoDD explicitly leverages depth-wise differential information, this experiment examines whether it remains effective across both shallow and deep Transformer stacks. Table 18 reports the relative l_2 errors on the Elasticity benchmark for Transformer and Transformer+MoDD across different depths.

As the model depth increases, MoDD consistently improves the Transformer baseline across all tested depths, indicating that the proposed depth-wise aggregation does not degrade either shallow or deep stacks. The relative gain is substantial for shallow and moderately deep models, and becomes smaller at larger Transformer depths, possibly reflecting a task-level performance ceiling. Notably, the highlighted entries in Table 18 show that a 32-layer Transformer and an 8-layer Transformer+MoDD achieve nearly identical relative l_2 errors, 0.41 and 0.40, respectively. This suggests that MoDD can achieve comparable accuracy with a much shallower and smaller model by more effectively utilizing depth-wise attention representations. The corresponding computational costs are analyzed in Appendix D.2.

Table 18: Performance scaling with model depth on the Elasticity benchmark. Relative l_2 errors are reported in $\times 10^{-2}$.

Model	4	8 (default)	16	32	64
Transformer	1.14	0.62	0.49	**0.41**	0.35
Transformer + **MoDD(ours)**	0.78	**0.40**	0.33	0.31	0.29
Gain over Transformer	**31.6%**	**35.5%**	**32.7%**	**24.4%**	**17.1%**

D Efficiency

D.1 Benchmark Efficiency

We evaluate the computational efficiency of integrating the proposed MoDD module into the Transolver architecture across six different PDE benchmarks, as summarized in Table 19. Importantly, the number of model parameters remains virtually unchanged, confirming that MoDD introduces no significant parameter overhead. In terms of GPU memory usage, the differences are minimal across all datasets, with some cases showing slight increases and others even slight reductions. For example, memory usage on the Airfoil and NS2d datasets increases marginally. The training time per epoch increases moderately due to the additional differential computations introduced by MoDD. For instance, training time for the Elasticity dataset increases from 7.7s to 10.9s. The performance gains achieved by MoDD come without added parameter count or memory burden, suggesting that its benefits arise from architectural efficiency rather than over-parameterization or resource-intensive computation.

Table 19: Comparison of model efficiency before and after integrating MoDD into the state-of-the-art solver (Transolver).

Metric	Model	Elasticity	Plasticity	Airfoil	Pipe	NS2d	Darcy
Param. Count(M)	Transolver	0.71	2.84	2.81	3.07	11.23	2.83
	Transolver + **MoDD(ours)**	0.71	2.84	2.81	3.07	11.22	2.82
GPU Memory(GB)	Transolver	0.57	2.85	4.17	6.62	9.91	3.08
	Transolver + **MoDD(ours)**	0.58	2.87	4.24	6.56	9.99	3.02
Running Time(s/epoch)	Transolver	7.7	89.4	19.7	31.6	136.8	12.5
	Transolver + **MoDD(ours)**	10.9	93.7	20.1	31.8	141.9	12.8

D.2 Depth-wise Efficiency Scaling

Complementing the depth-wise performance analysis in Appendix C.1, we further evaluate how the computational cost of MoDD scales with model depth. Table 20 reports the number of parameters, GPU memory usage, and training time per epoch for Transformer and Transformer+MoDD on the Elasticity benchmark across different depths.

MoDD introduces almost no additional parameters, since it reuses the standard attention projections and modifies only the depth-wise aggregation of attention representations. However, its memory usage and training time increase more rapidly with depth because MoDD stores and aggregates depth-differential states from previous layers.

Nevertheless, MoDD provides a favorable accuracy–efficiency tradeoff when models are compared at a similar accuracy level rather than at the same depth. To make this comparison explicit, we highlight the computational costs of two models that reach comparable accuracy. As shown in Appendix C.1, a 32-layer Transformer achieves a relative l_2 error of 0.41, while an 8-layer Transformer+MoDD achieves a comparable error of 0.40. The 8-layer Transformer+MoDD uses substantially fewer parameters, lower GPU memory, and less training time than the 32-layer Transformer, as highlighted in Table 20. Therefore, despite the additional per-layer cost, MoDD can reach comparable accuracy with a much shallower model, yielding a better overall efficiency-performance tradeoff.

Table 20: Depth-wise computational efficiency on the Elasticity benchmark. Highlighted cells compare a 32-layer Transformer and an 8-layer Transformer+MoDD, which achieve comparable relative l_2 errors of 0.41 and 0.40, respectively, while the MoDD model is substantially more efficient. See corresponding relative l_2 errors in Appendix C.1.

Model	Metric	Depth				
		4	8 (default)	16	32	64
Transformer	Param. # [M]	0.43	0.83	1.62	**3.21**	6.38
	Memory [GB]	0.72	0.87	1.16	**1.74**	2.90
	Time [s/epoch]	3.8	6.7	13.7	**26.6**	53.4
Transformer + **MoDD(ours)**	Param. # [M]	0.43	**0.83**	1.62	3.21	6.38
	Memory [GB]	0.75	**0.95**	1.46	2.93	8.29
	Time [s/epoch]	4.7	**9.2**	21.6	48.3	134.2

D.3 Memory Scaling with Token Resolution

Complementing the depth-wise efficiency analysis in Appendix D.2, we further evaluate how GPU memory usage scales with token resolution. We vary the number of input tokens from 1024 to 8192 under the same training setup and compare Transformer and Transformer+MoDD. Table 21 reports the measured GPU memory usage.

As the token resolution increases, GPU memory usage increases sharply for both Transformer and Transformer+MoDD, reflecting the dominant token-dependent activation cost. In contrast, the additional memory introduced by MoDD remains negligible across all tested resolutions (≤ 0.03 GB). This indicates that, in this setting, MoDD has nearly identical GPU-memory scaling to the baseline with respect to token resolution. Together with the depth-wise scaling results in Appendix D.2, this suggests that the main computational overhead of MoDD is associated with depth-wise state aggregation rather than token-resolution scaling.

Table 21: GPU memory scaling with token resolution. Memory usage is measured under the same training setup for Transformer and Transformer+MoDD. MoDD introduces negligible additional memory overhead across token resolutions.

Model	Token Resolution			
	1024	2048	4096	8192
Transformer	0.59	0.89	2.31	7.67
Transformer + **MoDD (ours)**	0.60	0.90	2.33	7.70

E Visualization

As shown in Figure 10, we present visual comparisons of prediction results and corresponding error maps across six PDE benchmarks: Elasticity, Plasticity, Airfoil, Pipe, Navier–Stokes, and Darcy flow. Each benchmark includes the ground truth solution, predictions from the SOTA solver Transolver [56], and predictions from Transolver with MoDD. The error maps highlight the pointwise differences between the predicted and ground truth fields. Across all benchmarks, the MoDD-augmented model consistently exhibits lower prediction error with error maps revealing meaningful reductions.

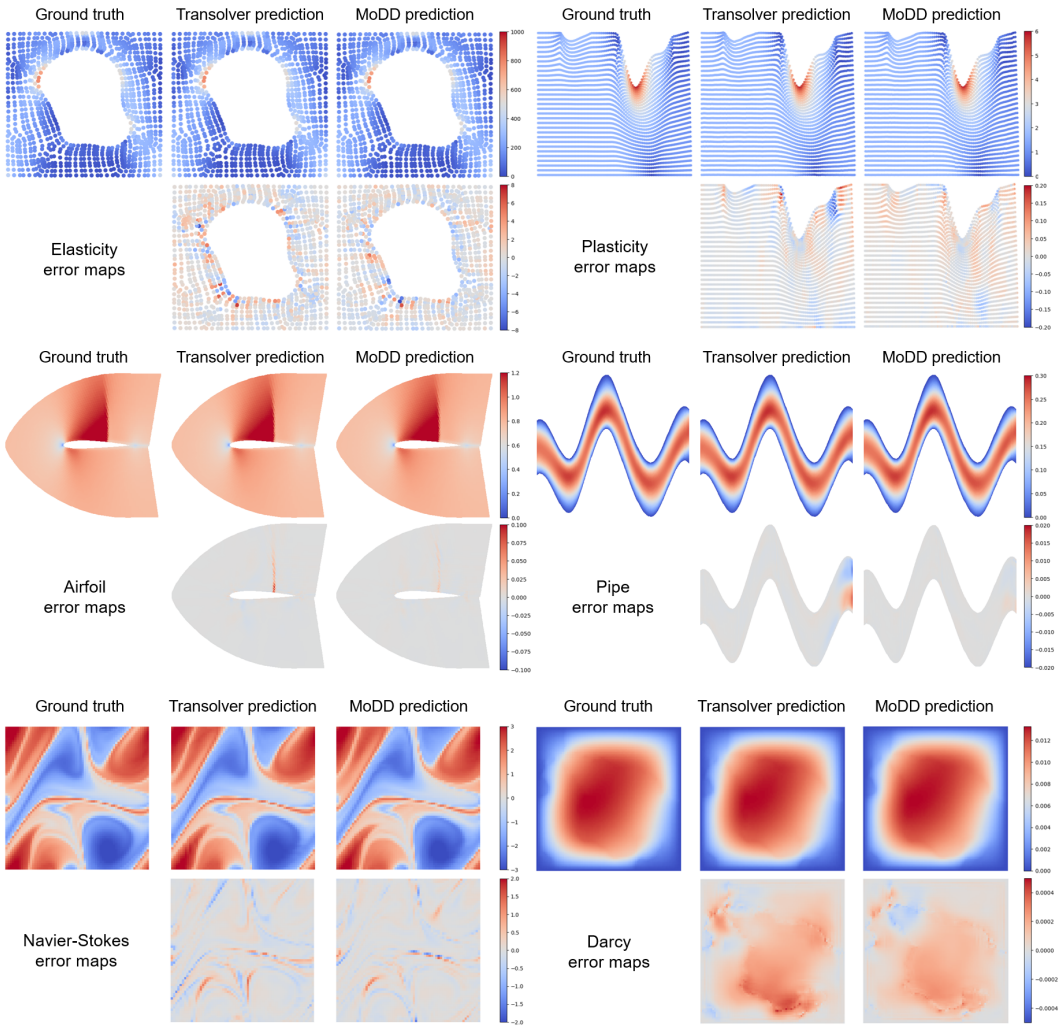


Figure 10: Visualization of prediction results and associated error maps across six PDE benchmarks, comparing the baseline SOTA solver (Transolver) and Transolver with MoDD (ours).

F Analysis

This section provides detailed methodologies and supplementary results for the analyses presented in the main manuscript.

F.1 Effective Depth Analysis

We quantify effective depth by measuring how broadly prediction-relevant contributions are distributed across model depth. For each attention head h in layer ℓ, we perform a single-head ablation at test time by setting its post-attention output to zero. The contribution of the corresponding head is defined as

$$c_{\ell,h} = \max\Big(E^{(0)}_{\ell,h} - E_{\text{base}},\, 0\Big)\,, \tag{4}$$

where $E^{(0)}_{\ell,h}$ denotes the test relative ℓ_2 error after setting the output of head h in layer ℓ to zero, and E_{base} denotes the error of the unmodified model.

The normalized effective depth is then computed using the participation ratio:

$$D_{\text{eff}} = \frac{\Big(\sum_{\ell,h} c_{\ell,h}\Big)^2}{LH \sum_{\ell,h} c^2_{\ell,h}}, \tag{5}$$

where L and H denote the numbers of layers and attention heads, respectively. A larger D_{eff} indicates that prediction-relevant contributions are distributed across a broader set of head-wise components along the model depth, whereas a smaller value indicates that they are concentrated in only a few components.

F.2 Layer-wise Shared Subspace Analysis as Diagnostics for Theorem 3.1

F.2.1 Purpose

This analysis provides empirical diagnostics for the sufficient conditions underlying Theorem 3.1. Specifically, we test (i) whether attention representations contain a non-trivial layer-shared low-rank component (Assumption G.2), and (ii) whether inter-layer differentiation suppresses this component.

F.2.2 Methodology

We compare a baseline model without differencing (MoD) and a differential model (MoDD) on the Elasticity dataset by analyzing layer-wise $Q/K/V$ representations. For each layer ℓ and $X \in \{Q, K, V\}$, we extract attention features $X^{(\ell)} \in \mathbb{R}^{B\times H\times N\times D}$. MoD uses $\tilde{X}^{(\ell)} = X^{(\ell)}$ for all layers, while MoDD uses $\tilde{X}^{(1)} = X^{(1)}$ and $\tilde{X}^{(\ell)} = X^{(\ell)} - X^{(\ell-1)}$ for $\ell \geq 2$. For each X and condition, we estimate a global rank-k shared subspace in $\mathbb{R}^D$ by aggregating all vectors $\tilde{z}^{(\ell)}_{b,h,n}$ across layers, batches, heads, and tokens and extracting the top-k principal directions. Given the basis $F_{X,k} \in \mathbb{R}^{k\times D}$, we define the common-mode ratio

$$\mathrm{CMR}_{X,k}(\ell, h) = \frac{\mathbb{E}_{b,n}\left[\|F_{X,k}\tilde{z}^{(\ell)}_{b,h,n}\|_2^2\right]}{\mathbb{E}_{b,n}\left[\|\tilde{z}^{(\ell)}_{b,h,n}\|_2^2\right]}, \qquad \mathrm{CMR}_{X,k}(\ell) = \frac{1}{H}\sum_{h=1}^{H}\mathrm{CMR}_{X,k}(\ell, h),$$

which measures the fraction of representation energy aligned with the shared subspace. Results are reported for $k \in \{1, 4\}$ together with global variance spectra.

F.2.3 Result

Figure 11 shows that MoD exhibits large common-mode ratios across layers for all $X \in \{Q, K, V\}$. For $k = 1$, a dominant shared direction explains a substantial fraction of the energy; for $k = 4$, a low-rank shared subspace remains prominent. These observations support Assumption G.2. MoDD yields systematically lower common-mode ratios than MoD, with the strongest and most consistent reductions for Value representations. For $k = 1$, the average ratio is reduced to $0.79\times$ for Query ($p = 0.017$) and $0.49\times$ for Value ($p = 0.020$), while no significant reduction is observed for Key. For $k = 4$, the reduction remains statistically significant for Value ($0.81\times$, $p = 0.009$), with smaller, non-significant changes for Query and Key. This indicates that inter-layer differencing primarily suppresses the dominant shared components.

Figure 12 shows similar global variance spectra for MoD and MoDD. Although leading components in MoDD are slightly less dominant, the overall spectral structure is preserved, indicating that reduced common-mode ratios are not due to variance collapse.

F.2.4 Connection to Theorem 3.1

These results provide empirical support for the sufficient conditions of Theorem 3.1:

(1) **Layer-shared component.** In MoD, a small number of global principal directions explain a substantial fraction of $Q/K/V$ energy across layers, consistent with Assumption G.2.

(2) **Effect of differencing.** MoDD exhibits reduced common-mode ratios for both $k = 1$ and $k = 4$, indicating suppression of the shared component by inter-layer differencing.

(3) **Structural interpretation.** The similarity of global variance spectra rules out variance collapse, supporting a structural reduction in alignment.

(4) **Low-rank regime.** The dominance of the leading principal component suggests a strongly low-rank shared structure, in which first-order differencing primarily affects the dominant common mode.

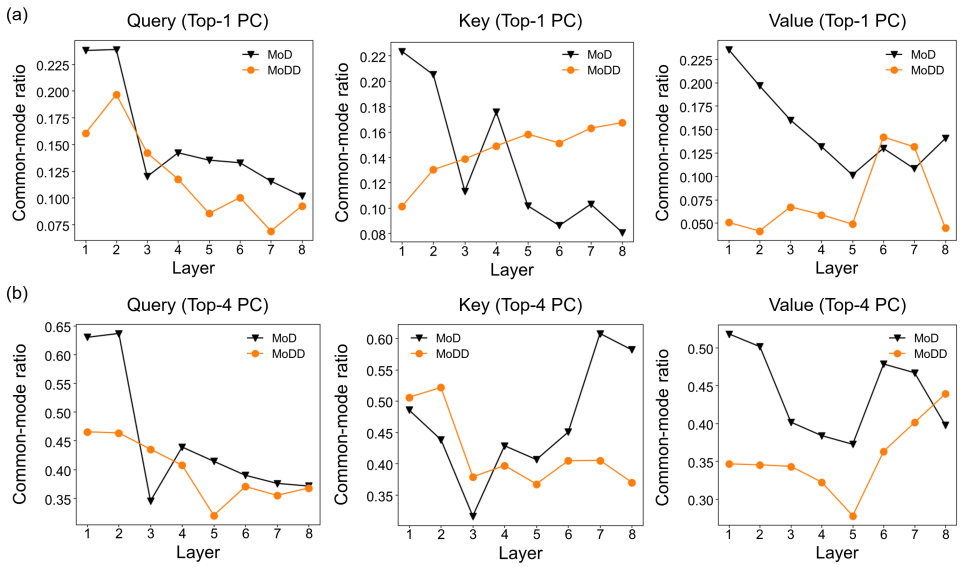


Figure 11: Layer-wise shared principal subspace ratios of attention representations. (a) Energy fraction captured by the top-1 shared principal component (PC). (b) Energy fraction captured by the top-4 shared principal subspace. The baseline model (MoD) exhibits a substantial layer-shared low-rank component, while the differential model (MoDD) consistently shows reduced alignment with the global shared subspace, indicating suppression of common-mode structure under inter-layer differencing.

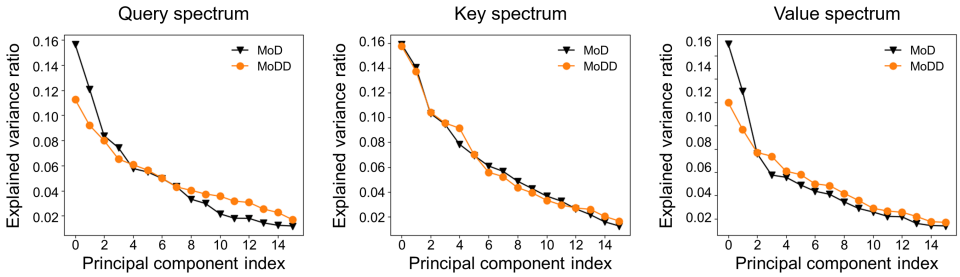


Figure 12: Global variance spectra of attention representations projected onto the common subspace. The normalized explained variance is shown as a function of principal component index for Query, Key, and Value representations. The baseline (MoD) and differential (MoDD) models exhibit similar spectral decay profiles, indicating that the overall variance structure is preserved. The slightly reduced dominance of leading components in MoDD, particularly for Query and Value, is consistent with reduced alignment with globally shared directions rather than variance collapse.

F.3 Layer-wise Feature Correlation Analysis as Diagnostics for Theorem 3.1

Purpose This analysis provides empirical diagnostics for the representation-level decorrelation effect used in the proof of Theorem 3.1. Specifically, we examine whether inter-layer differencing empirically reduces cross-layer dependence of Query and Key features, as predicted by the baseline-relative covariance reduction in Part (I) of the proof of Theorem 3.1.

F.3.1 Methodology

On the Elasticity dataset, we extract layer-wise Query and Key features $\{Q^{(l)}\}_{l=1}^{L}$ and $\{K^{(l)}\}_{l=1}^{L}$ from trained models and flatten across heads and tokens to obtain one feature vector per layer. For the baseline model (MoD), we analyze the original layer-wise features. For the differential model (MoDD), the first layer uses the original features, while for $l \geq 2$ we use inter-layer differences $\Delta Q^{(l)} := Q^{(l)} - Q^{(l-1)}$ and $\Delta K^{(l)} := K^{(l)} - K^{(l-1)}$. We compute layer-by-layer cosine similarity matrices $C^Q, C^K \in \mathbb{R}^{L\times L}$, whose off-diagonal entries summarize cross-layer feature correlation.

F.3.2 Result

Figure 13 shows that MoD exhibits substantial off-diagonal similarity, indicating non-negligible cross-layer correlation in both Query and Key features. In contrast, MoDD shows markedly attenuated off-diagonal similarity: the mean off-diagonal cosine similarity is close to zero or mildly negative for both Query and Key (Query: -0.13, Key: -0.12), while diagonal entries remain high. This indicates that inter-layer differencing suppresses cross-layer feature alignment while preserving within-layer structure.

F.3.3 Connection to Theorem 3.1

(1) **Delta-feature decorrelation.** The reduction in off-diagonal cosine similarity after inter-layer differencing is qualitatively consistent with the feature-level decorrelation mechanism in Part (I) of the proof of Theorem 3.1.

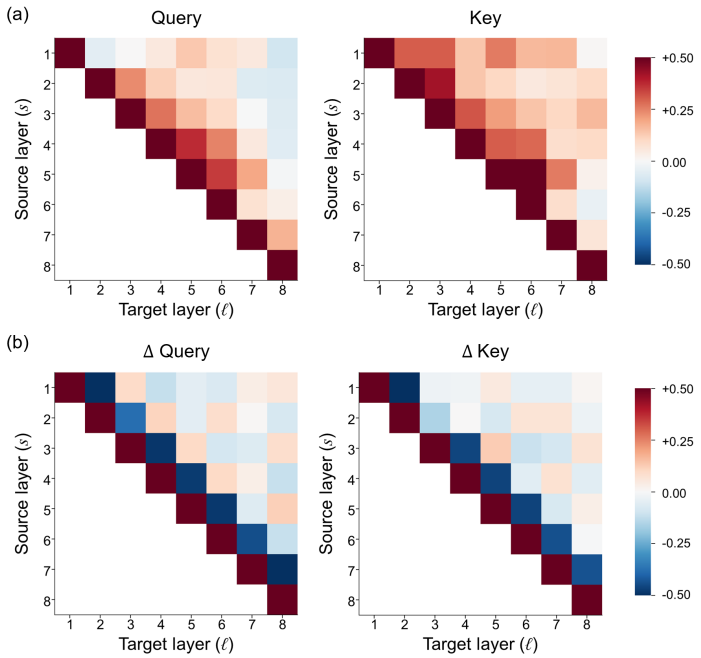


Figure 13: Cosine similarity matrices of (a) layer-wise Query and Key features and (b) inter-layer differential features, where each entry measures the cosine similarity between features from source layer s and target layer ℓ. Diagonal entries correspond to within-layer similarity, while off-diagonal entries reflect cross-layer feature correlation.

F.4 Empirical Assumption Diagnostics for Theorem 3.1

Purpose Theorem 3.1 establishes a deterministic sufficient condition under which layer-wise differentiation reduces cross-layer kernel alignment. The role of this appendix is not to re-validate the theorem itself, but to examine how the sufficient-regime quantities appearing in the theorem are manifested inside trained Transformer representations. Accordingly, all diagnostics below are interpreted at the level of trained-network head/layer-pair components.

Decay of differential correlation We first measure normalized correlations between inter-layer differential features as a function of the layer gap. The differential correlations decrease with increasing layer separation: Q–Q correlations decay from 1.00 to 0.90, 0.84, and 0.81, with similar trends observed for K–K and Q–K pairs. This supports the weak-dependence and depth-wise decorrelation mechanism formalized in Theorem 3.1.

Shared principal components We next apply pooled PCA to raw Q/K representations across layers to quantify layer-shared structure. The top-4 shared components explain 0.68–0.80 of the energy for Q and 0.65–0.74 for K, while the mean rank-1 fit is $R^2 = 0.47$ for Q and $R^2 = 0.39$ for K. These results show that trained representations contain substantial shared cross-layer structure, which is the structural component modeled in Theorem 3.1.

Gain drift To examine the gain-variation quantities used in the theorem, we estimate the relative drift of shared gains across layers. The median drift is 0.45 for Q and 0.74 for K, indicating that the gain variation remains controlled in trained representations. This provides empirical support for the gain-variation regime used in Theorem 3.1.

Mixed third moments We also estimate proxies for the mixed third-moment terms across distinct layers. The median value is small, approximately 0.058, indicating that the typical mixed third-order contribution is weak. This is consistent with the moment-control regime considered in Theorem 3.1.

Covariance dominance We examine the covariance-dominance condition at the level of individual head/layer-pair components in trained networks. Theorem 3.1 is a component-level sufficient-regime result: whenever the covariance-dominance condition is satisfied for a given component, the corresponding alignment-reduction conclusion follows deterministically for that component.

Specifically, we measure the joint covariance-dominance condition requiring Q–Q < 1, K–K < 1, and Q–K < 1 simultaneously. This joint sufficient condition is satisfied by approximately 21% of the evaluated head/layer-pair components in trained networks. This shows that the sufficient regime used in Theorem 3.1 is not merely hypothetical, but is concretely instantiated in trained Transformer representations.

Since the covariance-dominance condition is sufficient rather than necessary, this diagnostic is used to verify the presence of theorem-certified components, not to exhaustively characterize all components where decorrelation occurs. The aggregate trained-model effect is evaluated separately through direct cross-layer kernel-alignment measurements, which show that layer-wise differentiation reduces off-diagonal kernel alignment across trained layers.

Summary Overall, these diagnostics show that the structural quantities used in Theorem 3.1 are empirically reflected in trained Transformer representations. Shared cross-layer structure, differential-correlation decay, controlled gain variation, weak mixed third-moment contributions, and components satisfying the covariance-dominance sufficient condition are all observed in the trained model.

Together with the direct reduction of cross-layer kernel alignment after layer-wise differentiation, these results support the central conclusion that MoDD's layer-wise differential operation induces depth-wise kernel decorrelation at the trained-model level.

F.5 Layer-wise Base-Kernel Alignment as Diagnostics for Theorem 3.1

F.5.1 Purpose

This analysis serves as an empirical diagnostic for the kernel-level decorrelation mechanism underlying Theorem 3.1. Specifically, we examine whether inter-layer differencing reduces cross-layer alignment of base attention kernels constructed independently at each layer, as implied by the score- and kernel-level arguments in the proof.

F.5.2 Method

We analyze layer-wise alignment of *base attention kernels* in the standard attention model (MoD) and the Δ-attention model (MoDD) on the Elasticity dataset. In MoD, the base attention kernel at layer ℓ is defined from the layer-wise Query and Key representations $(Q^{(\ell)}, K^{(\ell)})$ produced at that layer. In MoDD, base kernels are constructed from inter-layer difference representations: at the first layer, the original representations $(Q^{(1)}, K^{(1)})$ are used, while for layers $\ell \geq 2$ the kernel inputs are $(\Delta Q^{(\ell)}, \Delta K^{(\ell)})$, with $\Delta Q^{(\ell)} = Q^{(\ell)} - Q^{(\ell-1)}$ and $\Delta K^{(\ell)} = K^{(\ell)} - K^{(\ell-1)}$. In both models, attention is governed by an exponential kernel of scaled Query–Key dot products, and we analyze its normalized realization given by the softmax attention weights. To quantify functional similarity between base attention kernels from layers s and ℓ, we compute centered kernel alignment (CKA), producing a layer-wise alignment matrix whose off-diagonal entries measure cross-layer kernel similarity.

F.5.3 Results

Figure 14 shows that in the standard Transformer, base attention kernels exhibit substantial off-diagonal alignment, indicating strong functional similarity across layers. In contrast, the Δ-attention variant shows markedly reduced off-diagonal CKA values, with alignments concentrated near zero away from the diagonal. This indicates that inter-layer differencing suppresses cross-layer redundancy at the level of base attention kernels.

F.5.4 Connection to Theorem 3.1

This analysis supports the kernel-level components of the proof in the following sense:

(1) **Reduced kernel alignment from delta features.** In the theory, base attention kernels are induced by exponential functions of scaled Query–Key inner products. Parts (II)–(III) in the proof of Theorem 3.1 show that reduced cross-layer correlations of the underlying delta features $(\Delta q^{(\ell)}, \Delta k^{(\ell)})$ lead to weaker cross-layer dependence of the corresponding kernels. The observed reduction in base-kernel CKA is consistent with this mechanism.

(2) **Kernel-level manifestation of score decorrelation.** The proof relates kernel alignment to score cross-moments via centered exponential kernels. Lower alignment among base attention kernels empirically reflects the attenuation of cross-layer score dependence predicted by the theory, without invoking assumptions beyond those already used in Theorem 3.1.

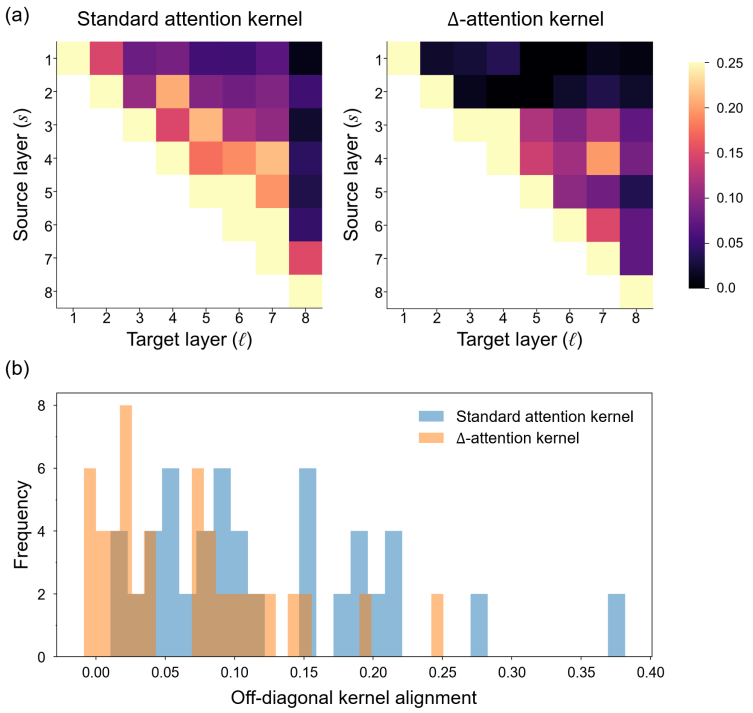


Figure 14: Layer-wise alignment of *base attention kernels* in the standard Transformer and the proposed Δ-attention variant. (a) Centered kernel alignment (CKA) matrices computed from softmax-normalized *base* attention kernels constructed independently at each layer, where each entry measures the functional similarity between kernels from source layer s and target layer ℓ. The standard Transformer exhibits strong off-diagonal alignment, indicating substantial redundancy among base attention kernels across layers. In contrast, the Δ-attention variant shows markedly reduced cross-layer alignment. (b) Distribution of off-diagonal CKA values for base attention kernels. The Δ-attention model yields a distribution sharply concentrated near zero, reflecting enhanced inter-layer decorrelation and increased functional diversity at the level of base attention kernels.

F.6 Input-level MoDD as a Control: Reduction to a Single Current-layer Attention-kernel Form

This subsection provides a control construction in which MoDD-style delta differencing and gated mixing are applied at the *input* representation level. We show that when queries and keys are formed using a *single* projection pair at the current layer, the resulting attention kernel reduces to the pullback of a *single* exponential bilinear form. Consequently, this construction does not realize the depth-wise multi-kernel mechanism established for MoDD in Lemma 3.1.

Definition F.1 (Input-level MoDD mixing (control))**.** Fix $l \geq 2$ and nonnegative weights $\{\lambda^{(j)}\}_{j=2}^{l}$ satisfying $\sum_{j=2}^{l} \lambda^{(j)} = 1$. Given layer-wise representations $\{X^{(j)}\}_{j=2}^{l}$, define the layer-wise deltas

$$\Delta X^{(j)} := X^{(j)} - X^{(j-1)}, \qquad j = 2, \ldots, l,$$

and the mixed input representation

$$\hat{X} := \sum_{j=2}^{l} \lambda^{(j)} \, \Delta X^{(j)}.$$

At layer l, define queries and keys using the *current-layer* projection matrices $W_Q^{(l)}$ and $W_K^{(l)}$ by

$$\hat{Q} := \hat{X} W_Q^{(l)}, \qquad \hat{K} := \hat{X} W_K^{(l)}.$$

The resulting scaled dot-product score is

$$S_{\mathrm{in}}(x, y) := \frac{1}{\sqrt{d}} \, \hat{Q}(x) \, \hat{K}(y)^{\top} = \frac{1}{\sqrt{d}} \big(\hat{X}(x) W_Q^{(l)}\big) \big(\hat{X}(y) W_K^{(l)}\big)^{\top}.$$

Proposition F.1 (Input-level MoDD retains a single current-layer attention-kernel construction)**.** *Let S_{in} be defined as in Definition F.1 and define the associated kernel*

$$K_{\mathrm{in}}(x, y) := \exp\big(S_{\mathrm{in}}(x, y)\big).$$

Then K_{in} is the pullback of a single *exponential bilinear kernel: there exist a fixed bilinear form $A^{(l)}$ and a representation map ϕ such that*

$$K_{\mathrm{in}}(x, y) = \exp\left(\frac{1}{\sqrt{d}} \, \phi(x)^{\top} A^{(l)} \phi(y)\right), \qquad \phi(x) := \sum_{j=2}^{l} \lambda^{(j)} \, \Delta X^{(j)}(x), \quad A^{(l)} := W_Q^{(l)} W_K^{(l)\top}.$$

In particular, although K_{in} admits a formal expansion of the exponent, it does not *realize a genuine multi-kernel product structure: all contributions are absorbed into a single bilinear form $A^{(l)}$ acting on a single representation space. With arbitrary distinct $W_Q^{(l)}$ and $W_K^{(l)}$, this identity does not imply a symmetric PSD kernel on the input space.*

Proof. We follow the structure of the proof of Lemma 3.1 and emphasize the key difference.

By Definition F.1, the mixed representation $\hat{X}(x)$ aggregates all layer-wise delta information into a single vector in $\mathbb{R}^d$:

$$\hat{X}(x) = \sum_{j=2}^{l} \lambda^{(j)} \, \Delta X^{(j)}(x).$$

Unlike the Q/K/V-level construction, no stacking or separation of layer-wise coordinates is introduced at the representation level.

Using the current-layer projections, the induced score can be written as

$$\begin{aligned} S_{\mathrm{in}}(x, y) &= \frac{1}{\sqrt{d}} \big(\hat{X}(x) W_Q^{(l)}\big) \big(\hat{X}(y) W_K^{(l)}\big)^{\top} \\ &= \frac{1}{\sqrt{d}} \, \hat{X}(x) \big(W_Q^{(l)} W_K^{(l)\top}\big) \, \hat{X}(y)^{\top}. \end{aligned}$$

Let $A^{(l)} := W_Q^{(l)} W_K^{(l)\top}$ and define $\phi(x) := \hat{X}(x)$. Then

$$S_{\text{in}}(x, y) = \frac{1}{\sqrt{d}} \phi(x)^\top A^{(l)} \phi(y),$$

and hence

$$K_{\text{in}}(x, y) = \exp\left(\frac{1}{\sqrt{d}} \phi(x)^\top A^{(l)} \phi(y) \right).$$

This expression shows that K_{in} is the pullback of the single exponential bilinear kernel $k_{A^{(l)}}(u, v) = \exp(\frac{1}{\sqrt{d}} u^\top A^{(l)} v)$ under the feature map ϕ. Because all layer-wise contributions are combined prior to projection, there is no decomposition of S_{in} into bilinear terms indexed by layer pairs (i, j) acting on independent coordinates. Accordingly, no product of distinct factor kernels arises. □

Discussion (control). The input-level construction applies the same delta-and-mix operation as MoDD, but *before* Q/K/V formation. Because queries and keys are generated using a single projection pair at the current layer, the resulting attention kernel necessarily collapses to the pullback of a single exponential bilinear kernel. This contrasts with the Q/K/V-level application of MoDD, where distinct projections across layers induce a product of weakly correlated kernels. Empirically, this input-level control yields negligible performance improvement, supporting that MoDD's gains arise from its depth-wise multi-kernel mechanism rather than from mere feature aggregation.

F.7 CKA Analysis

F.7.1 Methodology

To characterize the internal representation structure of MoDD, we analyze the hidden layer representations across attention blocks. However, the representations generated at each layer of a neural network are typically distributed across many neurons, making it difficult to analyze and compare internal representations across neural networks. To address this challenge, we adopt Centered Kernel Alignment (CKA) [9, 21], a method that enables quantitative comparison of representations both within a single model and across different models. Let $X \in \mathbb{R}^{m \times p_1}$ and $Y \in \mathbb{R}^{m \times p_2}$ be the representations of two layers, with p_1 and p_2 neurons respectively, computed over the same set of m examples. The corresponding Gram matrices $K = XX^\top$ and $L = YY^\top$ capture the pairwise similarities between examples based on their respective representations. The Gram matrices are centered using the centering matrix $H = I_m - \frac{1}{m}\mathbf{1}\mathbf{1}^\top$, yielding $K' = HKH$ and $L' = HLH$. The Hilbert-Schmidt Independence Criterion (HSIC) [13] quantifies the similarity between these centered Gram matrices by computing the dot product between their vectorized forms: $\text{HSIC}(K, L) = \text{vec}(K') \cdot \text{vec}(L')/(m-1)^2$. To obtain a similarity measure that is also invariant to isotropic scaling, the Centered Kernel Alignment (CKA) is defined as the normalized HSIC:

$$\text{CKA}(K, L) = \frac{\text{HSIC}(K, L)}{\sqrt{\text{HSIC}(K, K)\,\text{HSIC}(L, L)}} \tag{6}$$

To scale this computation to our models and datasets, we employ a minibatch-based approximation of the unbiased HSIC estimator [47], as proposed in prior work [39].

F.7.2 Results: Internal Representation

As the core of our model is the cross-layer attention strategy, we first investigate the gating behavior of inter-layer representations of our model. Figure 15a shows heatmaps for layer-to-layer gating weights of the vanilla Transformer with MoD and MoDD, trained on the Elasticity dataset. The learned gating weights imply the dependency of each attention layer on its preceding layers. The higher value at a pixel (ℓ, s) in the plot indicates that the target layer ℓ makes strong use of preceding s source layers.

MoD enables each attention block to access stacked representations from all preceding layers via soft gating by its design. This results in a dense connectivity pattern, similar to DenseNet [17], which promotes direct and efficient reuse of features across the model depth. While encouraging feature reuse across the model depth as MoD, our MoDD incorporates the layer-wise differential operation, which yields a more dispersed gating pattern. This leads to long-range connectivity among layers and thus facilitates multi-level attention representations by connecting different levels of layers. We observe that, at each block, the gating weights for preceding layers are smooth in MoDD, whereas the gating weight of each target layer is high itself in MoD. The final regression layer of MoDD exhibits more uniform gating patterns compared to MoD's, as shown in Figure 15a. This indicates that MoDD learns with similar significance of multi-layer attention features so that they can more evenly contribute to accurate regression.

We now investigate the transformation of internal representations of our MoDD using a similarity metric among outputs of attention blocks. To this end, the centered kernel alignment (CKA) is adopted [9, 21]. Figure 15b shows CKA maps among different outputs of each attention block for two models, i.e., vanilla Transformer and MoDD. We observe that, compared to the vanilla Transformer, MoDD exhibits a more uniform similarity map and strong similarities even across distant layers. This indicates that the preceding representations propagate through subsequent layers more faithfully, such that it leads to an enhanced alignment of internal representations. Furthermore, the smooth transition of similarity across depth in MoDD reflects a more stable and gradual transformation of internal representations throughout the architecture.

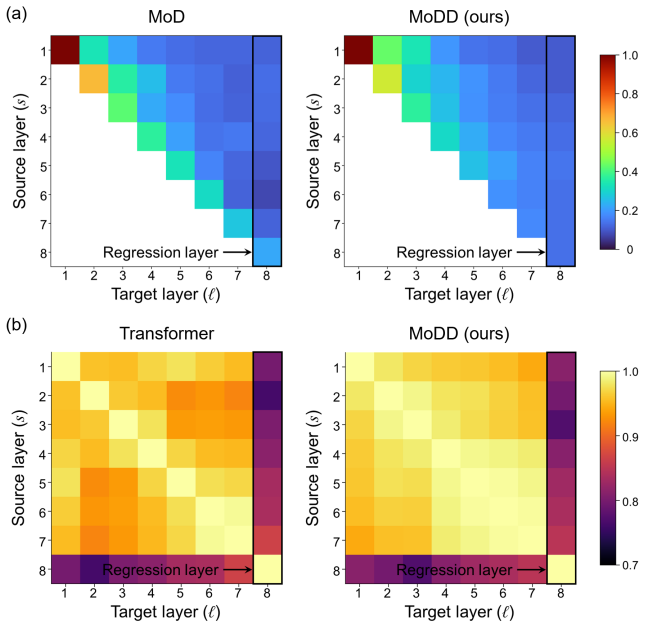


Figure 15: (a) Gating weights across attention layers for Transformer with MoD (left) and MoDD (right). Each pixel at position (ℓ, s) indicates the average soft gating weight of source layer s for target layer ℓ. Black boxes highlight the regression layer. The first row encodes weights for the input attention layer. (b) CKA similarity maps of attention block outputs for Transformer (left) and Transformer with MoDD (right). Each pixel shows the similarity between two layers s and ℓ; higher values reflect more similar internal representations. The regression layer is marked with black boxes.

F.8 2D Spatial–Depth Routing Interaction Induced by MoDD

We examine whether MoDD induces a 2D spatial–depth routing interaction when applied to a spatial-routing-based operator solver. In Transolver, token-to-slice assignment provides the spatial routing axis, but a limited number of slices can create a bottleneck that leads to repeated low-rank mixing across layers. MoDD adds a complementary depth axis by mixing inter-layer attention differentials, encouraging decorrelated representations across depth. Together, these mechanisms can be viewed as forming a two-dimensional routing space over spatial slices and network depth. To test whether this composition yields a task-level interaction rather than a slice-independent additive improvement, we sweep the number of slices and measure both relative l_2 error and routing statistics on the Airfoil benchmark.

For **spatial routing** in Transolver, we measure (1) *slice assignment entropy*, the normalized entropy of the weight distribution averaged over tokens; (2) *effective slices*, computed from the slice utilization distribution u, obtained by averaging routing weights over tokens, as $\exp\left(-\sum u \log u\right)/G$; and (3) *dead-slice ratio*, the fraction of slices whose utilization falls below 10% of the baseline under uniform slice use.

For **depth routing** by MoDD, we measure (4) *gate entropy*, the normalized entropy of the depth-gating distribution averaged over slices. The table below reports these statistics on Airfoil, averaged over batches, attention heads, and layers.

Table 22 shows that MoDD improves routing robustness and mitigates collapse, whereas Transolver is more sensitive to the number of slices. In particular, Transolver with 16 slices exhibits a clear slice bottleneck: the slice assignment entropy drops to 0.23, the effective-slice ratio decreases to 0.33, and the dead-slice ratio increases to 0.45. These statistics indicate that only a limited subset of slices is effectively utilized, leading to collapsed spatial routing under the strong slice bottleneck. Under the same 16-slice setting, Transolver with MoDD presents better spatial routing behavior compared to Transolver, with higher slice assignment entropy and effective slices, and a lower dead-slice ratio. At the same time, the gate entropy remains high across all slice counts, indicating that the depth routing does not collapse to a trivial single-depth selection.

In addition, the relative error reduction induced by MoDD varies across slice counts: the gain is largest under the strongest slice bottleneck with 16 slices (21.1%), while it becomes smaller when the number of slices is increased to 128 (12.7%). This slice-dependent gain indicates that MoDD does not merely provide a slice-independent additive improvement, but interacts with the spatial routing regime at the task-performance level.

We additionally evaluate Transolver and Transolver+MoDD on Navier–Stokes using 32 slices. Unlike Airfoil, spatial routing is already broad in the baseline and remains nearly unchanged after adding MoDD. The normalized assignment entropy is 0.55 for both models, while the dead-slice ratio is 0.20 for both. Meanwhile, MoDD maintains broad depth routing with a normalized gate entropy of 0.98.

Overall, these results support the view that MoDD introduces a complementary depth-routing axis to spatial-routing-based solvers, yielding a 2D spatial–depth routing interaction that can alleviate routing collapse under slice bottlenecks.

Table 22: Routing diagnostics across different numbers of slices on the Airfoil benchmark. Relative l_2 errors are reported in $\times 10^{-2}$.

Model	# Slices	Performance Metrics		Spatial Routing Metrics			Depth Routing Metric
		Rel. $l_2 \downarrow$	Rel. Gain over Transolver $\uparrow$	Assign. Ent. $\uparrow$	Eff. Slices $\uparrow$	Dead Slice $\downarrow$	Gate Ent. $\uparrow$
Transolver	16	0.57	–	0.23	0.33	0.45	–
	32	0.54	–	0.43	0.43	0.40	–
	64 (default)	0.53	–	0.40	0.43	0.43	–
	128	0.55	–	0.40	0.31	0.40	–
Transolver + **MoDD(ours)**	16	0.45	21.1%	0.40	0.47	0.32	0.81
	32	0.46	14.8%	0.50	0.48	0.29	0.87
	64 (default)	0.43	18.9%	0.49	0.44	0.30	0.82
	128	0.48	12.7%	0.51	0.45	0.32	0.86

F.9 Spectral Analysis of Attention Score Matrices

F.9.1 Methodology

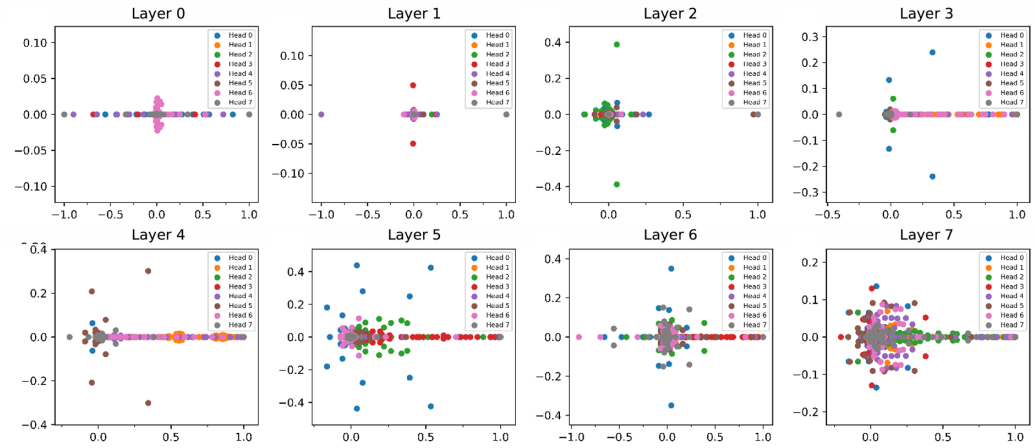
To analyze the spectral properties of the MoDD's attention mechanism, we examined the attention score matrices produced at each layer and head. These matrices are computed by applying the softmax function to the scaled dot-product of the query and key matrices. Specifically, for each layer l and attention head h, the attention score matrix is defined as $A^{(l,h)} = \mathrm{softmax}(QK^\top/\sqrt{d_k}) \in \mathbb{R}^{N\times N}$, where $Q, K \in \mathbb{R}^{N\times d_k}$ and N is the number of spatial tokens. For all $A^{(l,h)}$ matrices across layers and heads, we performed eigenvalue decomposition and visualized the resulting eigenvalues in the complex plane to characterize the spectral structure of attention.

F.9.2 Supplementary Results

To examine whether MoDD's ability to mitigate spectral collapse persists across all layers, Figure 16 shows the full-layer spectral analysis shown in Figure 6 of the main manuscript to all 8 layers. It confirms that the mitigation of rank collapse by MoDD, previously observed in layers 1 and 2, consistently holds across all 8 layers.

To further assess the generalizability of this property beyond the base Transformer and dataset, we conduct additional spectral analysis on a state-of-the-art solver, i.e., Transolver, using the Airfoil dataset. Figure 17 compares Transolver and Transolver with MoDD, evaluated on the same sample. In Transolver, distinct spectral gaps are observed in layers 0, 1, 2, 5, and 6, indicating low effective rank attention behavior at both shallow and deep layers. In contrast, Transolver with MoDD demonstrates a mitigation of spectral gaps in layers 2, 5, and 6. On Navier-Stokes, the mean spectral gap also decreases modestly from 0.593 to 0.565, while the mean magnitude of the second-largest eigenvalue increases from 0.407 to 0.435. This reflects the cumulative effect of MoDD, whose mechanism becomes increasingly effective in deeper layers, thereby preserving the rank diversity of attention maps and stabilizing deep feature propagation.

(a) Transformer



(b) MoDD

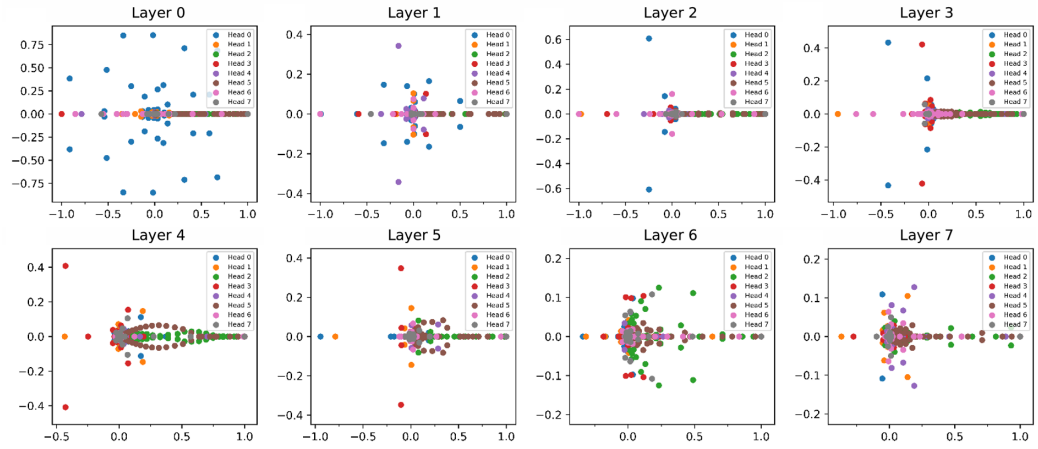
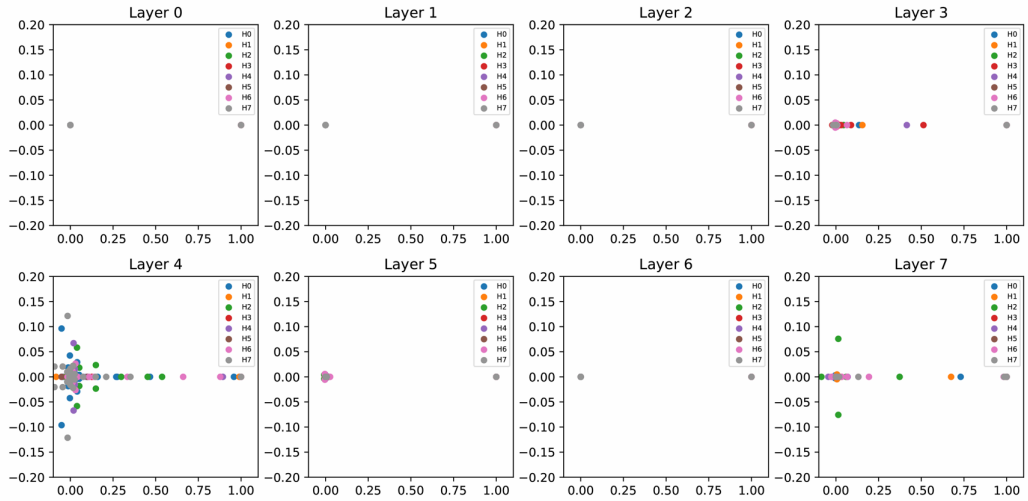


Figure 16: Spectral analysis of attention score matrices for (a) Transformer and (b) Transformer with MoDD (ours), evaluated on the same sample from the Elasticity dataset. The eigenvalues of attention score matrices for all attention heads of two selected layers are plotted in the complex plane, where the x-axis represents the real part and the y-axis represents the imaginary part.

(a) Transolver



(b) MoDD

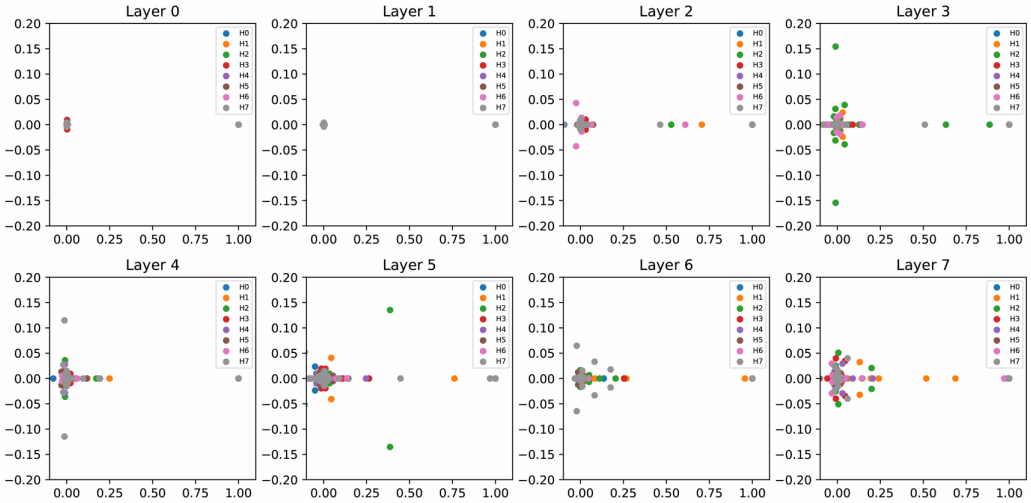


Figure 17: Spectral analysis of attention score matrices for (a) SOTA solver Transolver [56] and (b) Transolver with MoDD (ours), evaluated on the same sample from the Airfoil dataset. The eigenvalues of attention score matrices for all attention heads of two selected layers are plotted in the complex plane, where the x-axis represents the real part and the y-axis represents the imaginary part.

F.10 Attention Mean Distance Analysis

F.10.1 Methodology

To analyze how local or global information is captured across layers in Transformer with MoDD, we measured the mean attention distance for each attention head. following the methodology proposed in prior work [44]. Specifically, we weighted the Euclidean distances between query and key positions by the attention scores for each head, and averaged the results over 200 evaluation samples, each containing 972 spatial tokens (totaling 194,400 datapoints). For each layer and head, the mean and standard deviation of attention distances were computed across all tokens and batches, and the results were visualized as plots with error bars.

F.11 Effective Receptive Field Analysis

F.11.1 Methodology

To analyze the spatial behavior of MoDD, we perform a gradient-based Effective Receptive Field (ERF) analysis. This method quantifies how the output representation at a specific token index i^* is influenced by each input spatial location, enabling a layer-wise interpretation of the model's receptive field. Given the input coordinates $\mathbf{x} \in \mathbb{R}^{N\times 2}$ and the output features from the ℓ-th Transformer block $\mathbf{f}^{(\ell)} \in \mathbb{R}^{N\times d}$, we first compute a scalar response at token i^* by averaging its feature dimensions:

$$s^{(\ell)} = \frac{1}{d}\sum_{k=1}^{d} f^{(\ell)}[i^*, k] \tag{7}$$

We then compute the gradient of this scalar with respect to the input coordinates and define the ERF at input location $\mathbf{x}_i$ as:

$$\mathrm{ERF}^{(\ell)}(i) = \left\| \nabla_{\mathbf{x}_i} s^{(\ell)} \right\|_2 \tag{8}$$

This quantifies the spatial sensitivity of the output at i^*, providing a layer-wise view of the model's receptive field. The resulting ERF maps are normalized and visualized over the input mesh.

F.11.2 Supplementary Results

To perform an ablation-based extension of the ERF analysis presented in the main manuscript (Figure 8), Figure 18 visualizes the effective receptive fields (ERFs) for the vanilla Transformer, MoD, and MoDD on the same elasticity sample. Compared to the vanilla Transformer, MoD exhibits stronger attention around the void boundary, indicating that its mixture-based mechanism allows for faster adaptation to the dominant physical characteristic—specifically, stress concentration near the void. Nonetheless, MoD still produces relatively broad and diffuse attention patterns, with residual responses extending beyond the target token (red star) and into non-critical regions.

Building upon this, MoDD further sharpens and localizes the ERF by introducing layer-wise differential mechanisms. It concentrates attention more precisely along the void boundary and symmetric stress lines (A and A'), while effectively suppressing responses in irrelevant areas. This demonstrates that MoDD not only inherits the benefits of MoD's mixture formulation but also significantly enhances spatial specificity through its depth-aware refinement, enabling the model to focus locally on physically meaningful regions.

To further assess the generalizability of this behavior across different void geometries, Figure 19 presents ERF maps for an elasticity sample with an alternative void shape. Consistent with earlier observations, MoDD in Attention Layer 1 already delineates the void boundary. With increasing depth, the model increasingly attends to the void boundary and symmetric stress lines (A and A'), reaffirming its ability to capture physical priors and adapt to varying geometries.

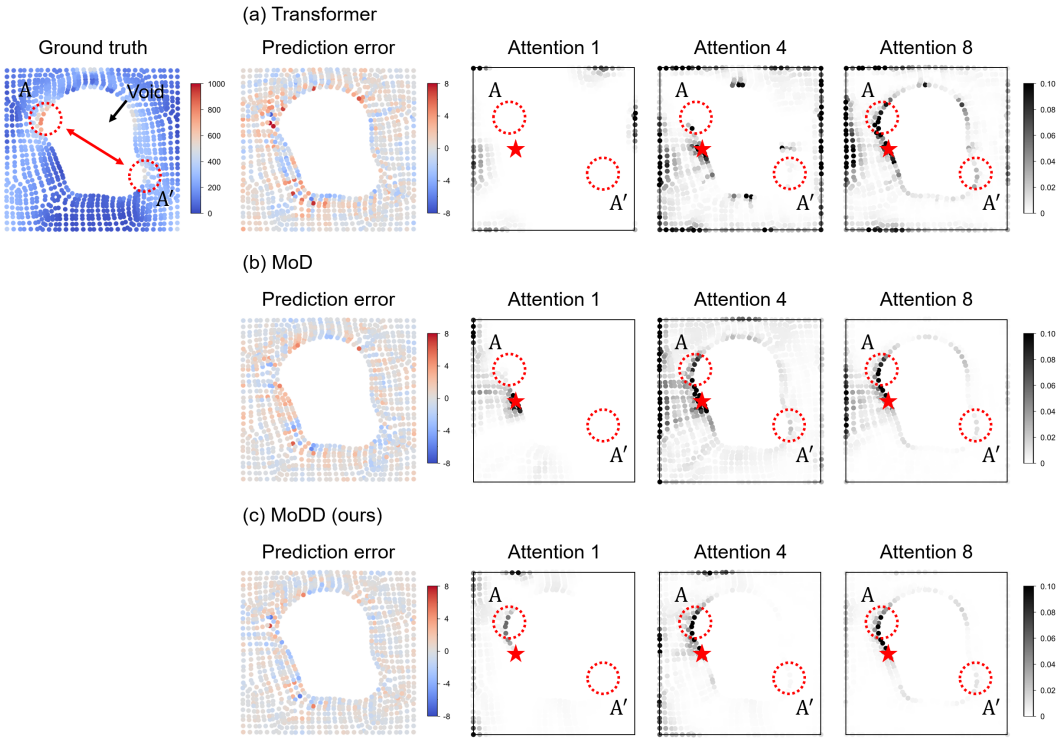


Figure 18: Ground truth stress field and prediction error maps of two trained models, i.e., (a) Transformer and (b) Transformer with MoD, (c) Transformer with MoDD (ours), for an Elasticity benchmark task. When predicting a solution at the specific input coordinates marked with a red star, the effective receptive fields of the selected attention layers are visualized. The maximum stress region A and its symmetric counterpart A′ are marked with red dashed circles. Each ERF is computed as the gradient magnitude of the output after applying attention with the residual connection.

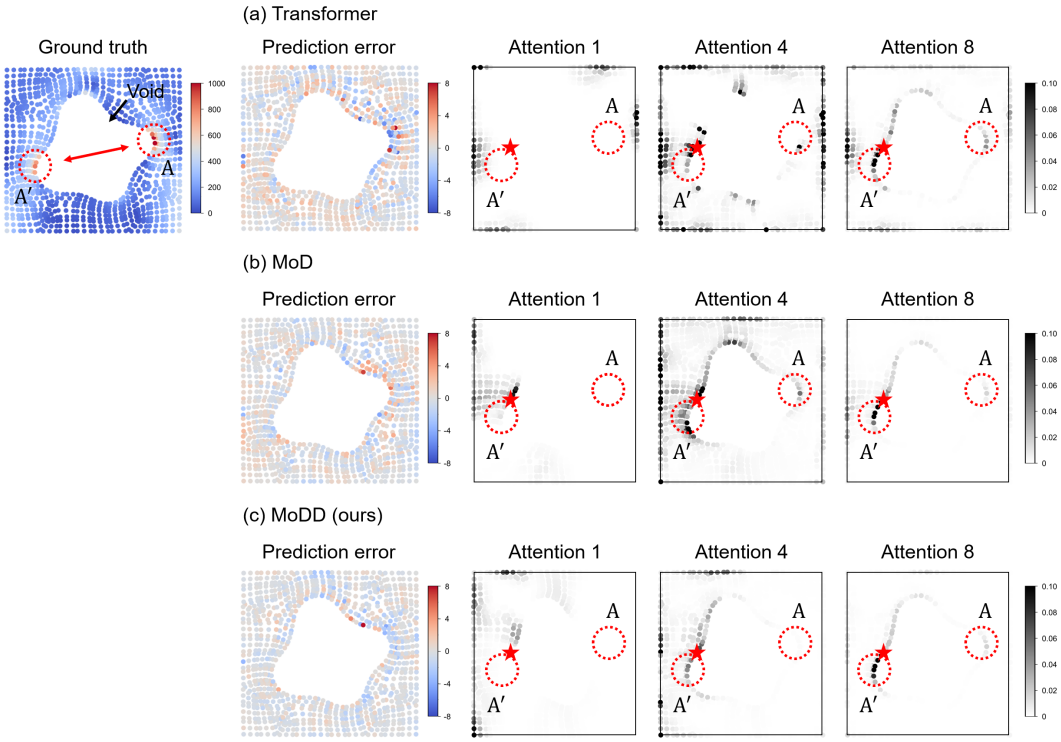


Figure 19: Ground truth stress field and prediction error maps of two trained models, i.e., (a) Transformer and (b) Transformer with MoD, (c) Transformer with MoDD (ours), for an Elasticity benchmark task. When predicting a solution at the specific input coordinates marked with a red star, the effective receptive fields of the selected attention layers are visualized. The maximum stress region A and its symmetric counterpart A′ are marked with red dashed circles. Each ERF is computed as the gradient magnitude of the output after applying attention with the residual connection.

G Theoretical Results

This section contains proofs of all theoretical statements in the paper. Our analysis follows the standard kernelized-attention framework used in FAVOR+ and Performer [8], and proceeds via a two-stage argument:

(i) **Scaled inner-product analysis.** We show that weak *cross-layer correlation of inter-layer difference features* implies weak correlation of the corresponding scaled inner products.

(ii) **Kernel-level transfer.** We relate the centered alignment between the exponential kernels $K^{(l)}(q,k) = \exp(S_l(q,k))$ and $K^{(m)}(q,k) = \exp(S_m(q,k))$ to the alignment of the *centered scores* $U_l = \mathcal{C}_\Pi S_l$ and $U_m = \mathcal{C}_\Pi S_m$. This step is based on a *first-order* Taylor linearization of the exponential, together with an explicit perturbation bound controlled by the remainder-to-linear ratio and a centering-bridge condition that connects U_l back to S_l.

All probabilistic assumptions and integrability conditions required for the analysis are stated explicitly.

G.1 Preliminaries and Notation

All random variables are defined on a common probability space $(\Omega, \mathcal{F}, \mathbb{P})$.

Query–key distribution. Let $(Q, K) \sim \Pi$ denote a random query–key pair drawn from an arbitrary joint distribution on $\mathbb{R}^d \times \mathbb{R}^d$. Unless otherwise specified, all expectations are taken with respect to Π. When needed, we use $\mathbb{E}[\cdot \mid Q]$ and $\mathbb{E}[\cdot \mid K]$ to denote conditional expectations under Π. Independent copies $(Q', K') \sim \Pi$ may be introduced for notational convenience, but no independence between Q and K is assumed.

Vector covariance and norms. For random vectors $X, Y \in \mathbb{R}^d$ with finite second moments, we define the (matrix) covariance by

$$\mathrm{Cov}(X, Y) := \mathbb{E}\big[(X - \mathbb{E}X)(Y - \mathbb{E}Y)^\top\big] \in \mathbb{R}^{d\times d}.$$

We use $\|\cdot\|_{\mathrm{HS}}$ to denote the Hilbert–Schmidt (Frobenius) norm.

Layer-wise and inter-layer features. For each layer $l \in \{1, \dots, L\}$, let $q^{(l)}(q) \in \mathbb{R}^d$ and $k^{(l)}(k) \in \mathbb{R}^d$ denote the layer-wise query and key features. For $l \geq 2$, we define the inter-layer difference (delta) features by

$$\Delta q^{(l)} := q^{(l)} - q^{(l-1)}, \qquad \Delta k^{(l)} := k^{(l)} - k^{(l-1)}.$$

Scaled inner products and exponential kernels. For $l \geq 2$, the scaled inner product is defined using the delta features as

$$S_l(q,k) := \frac{1}{\sqrt{d}} (\Delta q^{(l)})^\top \Delta k^{(l)}, \tag{9}$$

and the associated exponential kernel by

$$K^{(l)}(q,k) := \exp\big(S_l(q,k)\big). \tag{10}$$

which matches the exponential (softmax) dot-product kernel viewpoint commonly used in kernelized-attention analyses [8].

Positive semidefiniteness and the induced RKHS. Define the kernel

$$\kappa(u,v) := \exp\left(\frac{1}{\sqrt{d}} u^\top v\right), \qquad u, v \in \mathbb{R}^d,$$

so that $K^{(l)}(q,k) = \kappa(\Delta q^{(l)}, \Delta k^{(l)})$. We view both $\Delta q^{(l)}$ and $\Delta k^{(l)}$ as elements of the same representation space $\mathbb{R}^d$ for the kernel κ. The kernel κ is positive semidefinite (PSD). Indeed, using the power-series expansion

$$\kappa(u,v) = \sum_{n=0}^{\infty} \frac{1}{n!} \left(\frac{1}{\sqrt{d}} u^\top v\right)^n,$$

and defining the (possibly infinite-dimensional) feature map

$$\Phi(u) := \left(\sqrt{\frac{1}{n!}} \left(\frac{1}{d^{1/4}} \right)^n u^{\otimes n} \right)_{n \geq 0},$$

we have $(u^\top v)^n = \langle u^{\otimes n}, v^{\otimes n} \rangle$, and hence $\kappa(u, v) = \langle \Phi(u), \Phi(v) \rangle$. Therefore, κ induces a unique reproducing kernel Hilbert space (RKHS), denoted $\mathcal{H}_\kappa$. More generally, kernelized attention mechanisms can be studied in an RKHS framework whenever the attention nonlinearity corresponds to evaluation of a positive definite kernel, a perspective formalized in the Transformer setting by [7].

Kernel interpretation. We interpret $K^{(l)}(q, k) = \kappa(\Delta q^{(l)}, \Delta k^{(l)})$ as the evaluation of a positive semidefinite kernel $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ between two elements of the same ambient space. The two arguments correspond to the query-side and key-side delta features and are not required to coincide. Equivalently, one may view $(\Delta q^{(l)}, \Delta k^{(l)})$ as a single input in the product space $\mathbb{R}^d \times \mathbb{R}^d$, with $K^{(l)}$ induced by a kernel on this paired feature space.

Centering operator. For any integrable function $f : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, we define the joint-distribution centering operator $\mathcal{C}_\Pi$ by

$$(\mathcal{C}_\Pi f)(q, k) := f(q, k) - \mathbb{E}[f(Q, K) \mid K = k] - \mathbb{E}[f(Q, K) \mid Q = q] + \mathbb{E}[f(Q, K)], \tag{11}$$

where all expectations are taken with respect to the joint distribution $(Q, K) \sim \Pi$, and conditional expectations are understood in the almost-sure sense.

By construction, $\mathcal{C}_\Pi$ is linear and satisfies the conditional centering properties

$$\mathbb{E}\big[(\mathcal{C}_\Pi f)(Q, K) \mid K\big] = 0, \qquad \mathbb{E}\big[(\mathcal{C}_\Pi f)(Q, K) \mid Q\big] = 0 \quad \text{a.s.},$$

whenever the involved expectations are well-defined. This operator coincides with the standard population-level double-centering used in centered kernel alignment and HSIC-type dependence measures when Π is a product measure, and extends it naturally to general joint distributions [1].

Centered kernel alignment. For any integrable kernel $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, we denote its centered version (with respect to the joint distribution Π) by

$$K_c := \mathcal{C}_\Pi K,$$

where $\mathcal{C}_\Pi$ is the joint-distribution centering operator defined in Equation (11). For kernels K and K' with finite second moments and nonzero denominators, the population centered alignment is defined as

$$\rho(K, K') := \frac{\mathbb{E}_\Pi[K_c(Q, K)\, K'_c(Q, K)]}{\sqrt{\mathbb{E}_\Pi[K_c(Q, K)^2]\, \mathbb{E}_\Pi[(K'_c(Q, K))^2]}}.$$

G.2 Assumptions for Theorem 3.1

Throughout this appendix, $\mathrm{Tr}(\cdot)$ denotes the matrix trace.

Assumption G.1 (Centered features and finite second moments)**.** For all $l \in \{1, \dots, L\}$, $\mathbb{E}[q^{(l)}] = \mathbb{E}[k^{(l)}] = 0$ and $\mathbb{E}\|q^{(l)}\|_2^2, \mathbb{E}\|k^{(l)}\|_2^2 < \infty$.

Assumption G.2 (Layer-shared component decomposition)**.** Fix $L \geq 2$. For each $l \in \{1, \dots, L\}$, assume

$$q^{(l)} = a_l f + r^{(l)}, \qquad k^{(l)} = b_l g + s^{(l)},$$

where $f, g \in \mathbb{R}^d$ are zero-mean layer-shared random vectors, $a_l, b_l \in \mathbb{R}$ are deterministic gains, and $r^{(l)}, s^{(l)} \in \mathbb{R}^d$ are zero-mean innovations. Assume $\mathrm{Cov}(f, r^{(l)}) = 0$ and $\mathrm{Cov}(g, s^{(l)}) = 0$ for all l, and $\mathbb{E}\|f\|_2^2, \mathbb{E}\|g\|_2^2, \mathbb{E}\|r^{(l)}\|_2^2, \mathbb{E}\|s^{(l)}\|_2^2 < \infty$. By linearity, $\mathrm{Cov}(f, \Delta r^{(l)}) = 0$ and $\mathrm{Cov}(g, \Delta s^{(l)}) = 0$ for all $l \geq 2$.

Assumption G.3 (Slow variation of layer gains)**.** There exist constants $\eta_q, \eta_k \in [0, 1)$ such that for all $l \geq 2$, $|a_l - a_{l-1}| \leq \eta_q |a_l|$ and $|b_l - b_{l-1}| \leq \eta_k |b_l|$.

Assumption G.4 (Weak dependence of innovation increments)**.** There exist constants $\Gamma_q, \Gamma_k, \Gamma_{qk} < \infty$, independent of d, and a non-increasing function $\psi : \mathbb{N} \to \mathbb{R}_+$ such that for all $l \neq m$ with $l, m \geq 2$,

$$\begin{aligned}
\|\mathrm{Cov}(\Delta r^{(l)}, \Delta r^{(m)})\|_{\mathrm{HS}} &\leq \Gamma_q \, \psi(|l - m|), \\
\|\mathrm{Cov}(\Delta s^{(l)}, \Delta s^{(m)})\|_{\mathrm{HS}} &\leq \Gamma_k \, \psi(|l - m|), \\
\|\mathrm{Cov}(\Delta r^{(l)}, \Delta s^{(m)})\|_{\mathrm{HS}} &\leq \Gamma_{qk} \, \psi(|l - m|),
\end{aligned}$$

where $\Delta r^{(l)} := r^{(l)} - r^{(l-1)}$ and $\Delta s^{(l)} := s^{(l)} - s^{(l-1)}$.

Assumption G.5 (Common-mode dominance of layer-wise covariances)**.** There exists $\delta \in [0, 1)$ such that for all $l \neq m$,

$$\begin{aligned}
\|\mathrm{Cov}(r^{(l)}, r^{(m)})\|_{\mathrm{HS}} &\leq \delta \, |a_l a_m| \, \|\mathrm{Cov}(f, f)\|_{\mathrm{HS}}, \\
\|\mathrm{Cov}(s^{(l)}, s^{(m)})\|_{\mathrm{HS}} &\leq \delta \, |b_l b_m| \, \|\mathrm{Cov}(g, g)\|_{\mathrm{HS}}, \\
\|\mathrm{Cov}(r^{(l)}, s^{(m)})\|_{\mathrm{HS}} &\leq \delta \, |a_l b_m| \, \|\mathrm{Cov}(f, g)\|_{\mathrm{HS}}.
\end{aligned}$$

Assumption G.6 (Non-degenerate common-mode scale)**.** Assume $\|\mathrm{Cov}(f, f)\|_{\mathrm{HS}} > 0$, $\|\mathrm{Cov}(g, g)\|_{\mathrm{HS}} > 0$, $\|\mathrm{Cov}(f, g)\|_{\mathrm{HS}} > 0$, and that $|a_l|, |b_l|$ are bounded away from 0 on the layer indices of interest. (Otherwise, the corresponding ratio bounds are trivial.)

Assumption G.7 (Centered delta features and finite moments)**.** For all layers l, $\mathbb{E}[\Delta q^{(l)}] = 0$, $\mathbb{E}[\Delta k^{(l)}] = 0$, $\mathbb{E}\|\Delta q^{(l)}\|_2^2 < \infty$, and $\mathbb{E}\|\Delta k^{(l)}\|_2^2 < \infty$.

Assumption G.8 (Fourth-moment integrability)**.** For all $l \neq m$, $\mathbb{E}\big[\|\Delta q^{(l)}\|_2 \|\Delta q^{(m)}\|_2 \|\Delta k^{(l)}\|_2 \|\Delta k^{(m)}\|_2\big] < \infty$.

Assumption G.9 (Gaussian or weakly non-Gaussian initialization)**.** At initialization, the stacked random vector $(\Delta q^{(l)}, \Delta q^{(m)}, \Delta k^{(l)}, \Delta k^{(m)})$ is jointly Gaussian and centered, or more generally satisfies $\frac{1}{d}|\mathrm{Tr}(E_{lm})| \leq C_E$ for some constant C_E independent of d, l, m.

Assumption G.10 (Score integrability and non-degeneracy)**.** Fix $l \neq m$. Assume:

(a) $S_l, S_m \in L^2(\Pi)$ and $\|U_l\|_{L^2} > 0$, $\|U_m\|_{L^2} > 0$, where $U_i := \mathcal{C}_\Pi S_i$.

(b) $\exp(S_l), \exp(S_m) \in L^2(\Pi)$.

(c) $\mathbb{E}[S_l^4 e^{2|S_l|}] < \infty$ and $\mathbb{E}[S_m^4 e^{2|S_m|}] < \infty$.

Assumption G.11 (Score centering bridge)**.** For $i \in \{l, m\}$, assume $S_i \in L^2(\Pi)$ and define $\beta_i := \|\mathbb{E}[S_i(Q, K) \mid K]\|_{L^2(\Pi)} + \|\mathbb{E}[S_i(Q, K) \mid Q]\|_{L^2(\Pi)} + |\mathbb{E}[S_i(Q, K)]| < \infty$.

G.3 Additional Assumptions for Theorem 3.2

Assumption G.12 (Conditionally independent bounded-variance noise)**.** Conditional on $x_{1:n}$, the noises $\xi_1, \ldots, \xi_n$ are independent and satisfy

$$\mathbb{E}[\xi_i^2 \mid x_i] \le \tau^2 \quad \text{a.s. for all } i.$$

Equivalently, writing $\xi = (\xi_1, \ldots, \xi_n)^\top$, we have

$$\mathbb{E}[\xi\xi^\top \mid x_{1:n}] \preceq \tau^2 I_n.$$

Assumption G.13 (Uniform boundedness of factor kernels)**.** There exists $B \ge 1$ such that for all ℓ and all $x, x' \in \mathcal{X}$,

$$|K^{(\ell)}(x, x')| \le B.$$

Consequently, $|K_\Pi(x, x')| \le B^L$ for all x, x'.

Assumption G.14 (Non-degenerate factor variation)**.** For all ℓ, $\sigma_\ell^2 = \mathbb{E}[\widetilde{K}_\ell^2] > 0$.

Assumption G.15 (Fourth-moment control for squared centered values)**.** There exists $c_0 \ge 0$ such that for all $\ell \ne m$,

$$\mathbb{E}[\widetilde{K}_\ell^2\, \widetilde{K}_m^2] \le \big(1 + c_0 \rho_{\ell m}^2\big)\, \mathbb{E}[\widetilde{K}_\ell^2]\, \mathbb{E}[\widetilde{K}_m^2].$$

Assumption G.16 (Vanishing mixed third moments across distinct layers)**.** For all $\ell \ne m$,

$$\mathbb{E}[\widetilde{K}_\ell\, \widetilde{K}_m^2] = 0 \qquad \text{and} \qquad \mathbb{E}[\widetilde{K}_m\, \widetilde{K}_\ell^2] = 0.$$

G.4 Lemma for Theorem 3.1

Lemma G.1 (Basic L^2 stability of joint centering)**.** *For any $f \in L^2(\Pi)$,*

$$\|\mathcal{C}_\Pi f\|_{L^2(\Pi)} \le 4\|f\|_{L^2(\Pi)}.$$

Proof. Let

$$\alpha(K) := \mathbb{E}[f(Q,K) \mid K], \qquad \beta(Q) := \mathbb{E}[f(Q,K) \mid Q], \qquad \gamma := \mathbb{E}[f(Q,K)],$$

so that $f_c = \mathcal{C}_\Pi f$ can be written (a.s.) as

$$f_c(Q,K) = f(Q,K) - \alpha(K) - \beta(Q) + \gamma.$$

By the L^2 contraction property of conditional expectation,

$$\|\alpha(K)\|_{L^2(\Pi)} \le \|f(Q,K)\|_{L^2(\Pi)}, \qquad \|\beta(Q)\|_{L^2(\Pi)} \le \|f(Q,K)\|_{L^2(\Pi)}.$$

Moreover, $\|\gamma\|_{L^2(\Pi)} = |\gamma| \le \|f(Q,K)\|_{L^2(\Pi)}$ by Cauchy–Schwarz. Therefore, by the triangle inequality,

$$\|f_c\|_{L^2(\Pi)} \le \|f\|_{L^2(\Pi)} + \|\alpha(K)\|_{L^2(\Pi)} + \|\beta(Q)\|_{L^2(\Pi)} + \|\gamma\|_{L^2(\Pi)} \le 4\|f\|_{L^2(\Pi)}.$$

□

Lemma G.2 (Cross-moment perturbation)**.** *Let $l \neq m$ and assume $U_l, U_m, R_l, R_m \in L^2(\Pi)$. Then*

$$\left|\mathbb{E}[K_c^{(l)} K_c^{(m)}] - \mathbb{E}[U_l U_m]\right| \le \|U_l\|_{L^2}\|R_m\|_{L^2} + \|U_m\|_{L^2}\|R_l\|_{L^2} + \|R_l\|_{L^2}\|R_m\|_{L^2}.$$

Proof. Using the decomposition $K_c^{(i)} = U_i + R_i$, $K_c^{(l)} K_c^{(m)} = (U_l + R_l)(U_m + R_m)$ and expand the product. Apply Cauchy–Schwarz to each mixed term. □

Lemma G.3 (Denominator stability)**.** *If $\|U_i\|_{L^2} > 0$ and $\|R_i\|_{L^2} \le \eta_i \|U_i\|_{L^2}$ with $\eta_i \in [0,1)$, then*

$$(1-\eta_i)\|U_i\|_{L^2} \le \|K_c^{(i)}\|_{L^2} \le (1+\eta_i)\|U_i\|_{L^2}.$$

In particular, $\|K_c^{(i)}\|_{L^2} \ge (1-\eta_i)\|U_i\|_{L^2} > 0$.

Proof. This is the reverse triangle inequality applied to $K_c^{(i)} = U_i + R_i$: $\left|\|U_i + R_i\|_{L^2} - \|U_i\|_{L^2}\right| \le \|R_i\|_{L^2}$. □

Corollary G.1 (Score-alignment control of centered-score alignment)**.** *Assume Assumption G.11 and that $\|U_l\|_{L^2}, \|U_m\|_{L^2} > 0$. Then*

$$|\rho(U_l, U_m)| \le \frac{\|S_l\|_{L^2}\|S_m\|_{L^2}}{\|U_l\|_{L^2}\|U_m\|_{L^2}}\,|\rho(S_l, S_m)| + \frac{\beta_l}{\|U_l\|_{L^2}} + \frac{\beta_m}{\|U_m\|_{L^2}} + \frac{\beta_l\beta_m}{\|U_l\|_{L^2}\|U_m\|_{L^2}}. \tag{12}$$

Proof. By definition, $\rho(U_l, U_m) = \mathbb{E}[U_l U_m]/(\|U_l\|_{L^2}\|U_m\|_{L^2})$. Decompose $\mathbb{E}[U_l U_m] = \mathbb{E}[S_l S_m] + (\mathbb{E}[U_l U_m] - \mathbb{E}[S_l S_m])$. Using the bridge inequality

$$\left|\mathbb{E}[U_l U_m] - \mathbb{E}[S_l S_m]\right| \le \beta_l\|U_m\|_{L^2} + \beta_m\|U_l\|_{L^2} + \beta_l\beta_m,$$

and $|\mathbb{E}[S_l S_m]| = \|S_l\|_{L^2}\|S_m\|_{L^2}|\rho(S_l, S_m)|$, we obtain the desired bound after dividing by $\|U_l\|_{L^2}\|U_m\|_{L^2}$. □

G.5 Additional Lemma for Lemma 3.1

Lemma G.4 (Products of kernels are kernels)**.** *Let k_1 be a kernel on X_1 and k_2 be a kernel on X_2. Then the pointwise product*

$$(k_1 \cdot k_2)\big((x_1, x_2), (x_1', x_2')\big) := k_1(x_1, x_1')\, k_2(x_2, x_2')$$

defines a kernel on $X_1 \times X_2$. In particular, if $X_1 = X_2 = X$, then

$$k(x, x') := k_1(x, x')\, k_2(x, x'), \qquad x, x' \in X,$$

defines a kernel on X.

Proof. Let H_i be a Hilbert space and $\Phi_i : X_i \to H_i$ be a feature map associated with the kernel k_i, $i = 1, 2$, so that

$$k_i(x_i, x_i') = \langle \Phi_i(x_i), \Phi_i(x_i') \rangle_{H_i}.$$

Consider the Hilbert tensor product space $H_1 \widehat{\otimes} H_2$. For $(x_1, x_2), (x_1', x_2') \in X_1 \times X_2$, we compute

$$\begin{aligned} k_1(x_1, x_1')\, k_2(x_2, x_2') &= \langle \Phi_1(x_1), \Phi_1(x_1') \rangle_{H_1} \langle \Phi_2(x_2), \Phi_2(x_2') \rangle_{H_2} \\ &= \langle \Phi_1(x_1) \otimes \Phi_2(x_2),\ \Phi_1(x_1') \otimes \Phi_2(x_2') \rangle_{H_1 \widehat{\otimes} H_2}. \end{aligned}$$

Hence, the map

$$(x_1, x_2) \mapsto \Phi_1(x_1) \otimes \Phi_2(x_2)$$

is a feature map for $k_1 \cdot k_2$, which proves that $k_1 \cdot k_2$ is a kernel. The diagonal restriction yields the second claim. □

G.6 Additional Lemma for Theorem 3.2

Lemma G.5 (Reference-based effective-dimension control of sample-to-population and noise variance)**.** *Assume Assumptions G.12 and G.13. Then there exists a constant $C_{\mathrm{ed}} > 0$ (depending only on the kernel boundedness and a standard noise-moment condition) such that for all $\lambda > 0$,*

$$\mathbb{E}\big[(\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2\big] \le \frac{C_{\mathrm{ed}}}{n}\, d_\Pi(\lambda), \tag{13}$$

$$\mathbb{E}\Big[k(X)^\top (K + n\lambda I_n)^{-2} k(X)\Big] \le \frac{C_{\mathrm{ed}}}{n\lambda}\, d_\Pi(\lambda). \tag{14}$$

Moreover, one may replace $d_\Pi(\lambda)$ by $\mathbb{E}[\widehat{d}_\Pi(\lambda)]$ (up to adjusting C_{ed}) when working directly with the empirical operator.

Lemma G.6 (Alignment control of $\mathrm{Cov}(K_\ell^2, K_m^2)$)**.** *Assume Assumptions G.13, G.14, G.15, and G.16. Then for $\ell \neq m$,*

$$\big|\mathrm{Cov}(K_\ell^2, K_m^2)\big| \le c_0\, \rho_{\ell m}^2\, \sigma_\ell^2 \sigma_m^2 + 4B^2\, \sigma_\ell \sigma_m\, |\rho_{\ell m}|. \tag{15}$$

Proof. Write $K_\ell = \widetilde{K}_\ell + \mathbb{E}[K_\ell]$. Then $K_\ell^2 = \widetilde{K}_\ell^2 + 2\mathbb{E}[K_\ell]\widetilde{K}_\ell + \mathbb{E}[K_\ell]^2$. Expanding $\mathrm{Cov}(K_\ell^2, K_m^2)$ yields $\mathrm{Cov}(\widetilde{K}_\ell^2, \widetilde{K}_m^2)$, a term $4\mathbb{E}[K_\ell]\mathbb{E}[K_m]\mathrm{Cov}(\widetilde{K}_\ell, \widetilde{K}_m)$, and two mixed third-moment terms. Assumption G.16 kills the mixed terms, Assumption G.15 controls $\mathrm{Cov}(\widetilde{K}_\ell^2, \widetilde{K}_m^2)$, and boundedness yields $\mathbb{E}[K_\ell], \mathbb{E}[K_m] \le B$. □

Lemma G.7 (Second moment of a product: exact degree-2 expansion + bounded remainder)**.** *Under Assumption G.13, let $Z_\ell := K_\ell^2$, $\mu_\ell := \mathbb{E}[Z_\ell] \in [0, B^2]$, and $\delta_\ell := Z_\ell - \mu_\ell$. Then*

$$\mathbb{E}[K_\Pi(X, X')^2] = \mathbb{E}\Big[\prod_{\ell=1}^{L} Z_\ell\Big] = \prod_{\ell=1}^{L} \mu_\ell + \sum_{\ell<m} \mathrm{Cov}(Z_\ell, Z_m) \prod_{r \neq \ell, m} \mu_r + \mathcal{E}_L, \tag{16}$$

where the remainder term satisfies the universal bound

$$|\mathcal{E}_L| \le (B^2)^L \sum_{s=3}^{L} \binom{L}{s} 2^s. \tag{17}$$

Proof. Expand $\prod_{\ell=1}^{L}(\mu_\ell + \delta_\ell)$ and take expectations. Degree-1 terms vanish since $\mathbb{E}[\delta_\ell] = 0$. Degree-2 terms give $\mathbb{E}[\delta_\ell \delta_m] = \mathrm{Cov}(Z_\ell, Z_m)$. For $s \ge 3$, use $|\delta_\ell| \le |Z_\ell| + |\mu_\ell| \le B^2 + B^2 = 2B^2$ and count subsets. □

G.7 Proof of Theorem 3.1

Fix $l \neq m$ with $l, m \geq 2$.

(I) Baseline-relative bounds for cross-layer delta covariances. We work under Assumption F.1 so that all covariances appearing below are well-defined.

Step 1: delta-covariance decomposition.

By Assumption G.2, for $t \geq 2$,

$$\Delta q^{(t)} = (a_t - a_{t-1})f + \Delta r^{(t)}, \qquad \Delta k^{(t)} = (b_t - b_{t-1})g + \Delta s^{(t)}.$$

Since $\mathrm{Cov}(f, r^{(t)}) = 0$ and $\mathrm{Cov}(g, s^{(t)}) = 0$ for all t, linearity implies $\mathrm{Cov}(f, \Delta r^{(t)}) = 0$ and $\mathrm{Cov}(g, \Delta s^{(t)}) = 0$ for all $t \geq 2$. Hence,

$$\begin{aligned}
\mathrm{Cov}(\Delta q^{(l)}, \Delta q^{(m)}) &= (a_l - a_{l-1})(a_m - a_{m-1})\,\mathrm{Cov}(f, f) + \mathrm{Cov}(\Delta r^{(l)}, \Delta r^{(m)}),\\
\mathrm{Cov}(\Delta k^{(l)}, \Delta k^{(m)}) &= (b_l - b_{l-1})(b_m - b_{m-1})\,\mathrm{Cov}(g, g) + \mathrm{Cov}(\Delta s^{(l)}, \Delta s^{(m)}),\\
\mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(m)}) &= (a_l - a_{l-1})(b_m - b_{m-1})\,\mathrm{Cov}(f, g) + \mathrm{Cov}(\Delta r^{(l)}, \Delta s^{(m)}).
\end{aligned}$$

Taking Hilbert–Schmidt norms and applying the triangle inequality, the innovation terms are controlled by Assumption G.4, which introduces the constants Γ_q, Γ_k, and Γ_{qk} together with the decay function $\psi(\cdot)$. Moreover, the gain increments are bounded using the slow-gain condition (Assumption G.3), yielding the factor $\eta := \max\{\eta_q, \eta_k\}$. As a result, we obtain:

$$\begin{aligned}
\|\mathrm{Cov}(\Delta q^{(l)}, \Delta q^{(m)})\|_{\mathrm{HS}} &\leq |a_l - a_{l-1}|\,|a_m - a_{m-1}|\,\|\mathrm{Cov}(f, f)\|_{\mathrm{HS}} + \Gamma_q\,\psi(|l - m|)\\
&\leq \eta^2 |a_l a_m|\,\|\mathrm{Cov}(f, f)\|_{\mathrm{HS}} + \Gamma_q\,\psi(|l - m|), \qquad (18\text{-}1)\\
\|\mathrm{Cov}(\Delta k^{(l)}, \Delta k^{(m)})\|_{\mathrm{HS}} &\leq |b_l - b_{l-1}|\,|b_m - b_{m-1}|\,\|\mathrm{Cov}(g, g)\|_{\mathrm{HS}} + \Gamma_k\,\psi(|l - m|)\\
&\leq \eta^2 |b_l b_m|\,\|\mathrm{Cov}(g, g)\|_{\mathrm{HS}} + \Gamma_k\,\psi(|l - m|), \qquad (18\text{-}2)\\
\|\mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(m)})\|_{\mathrm{HS}} &\leq |a_l - a_{l-1}|\,|b_m - b_{m-1}|\,\|\mathrm{Cov}(f, g)\|_{\mathrm{HS}} + \Gamma_{qk}\,\psi(|l - m|)\\
&\leq \eta^2 |a_l b_m|\,\|\mathrm{Cov}(f, g)\|_{\mathrm{HS}} + \Gamma_{qk}\,\psi(|l - m|), \qquad (18\text{-}3)
\end{aligned}$$

Step 2: baseline covariance lower bounds.

By Assumption G.2, we have

$$\begin{aligned}
\mathrm{Cov}(q^{(l)}, q^{(m)}) &= a_l a_m\,\mathrm{Cov}(f, f) + \mathrm{Cov}(r^{(l)}, r^{(m)}),\\
\mathrm{Cov}(k^{(l)}, k^{(m)}) &= b_l b_m\,\mathrm{Cov}(g, g) + \mathrm{Cov}(s^{(l)}, s^{(m)}),\\
\mathrm{Cov}(q^{(l)}, k^{(m)}) &= a_l b_m\,\mathrm{Cov}(f, g) + \mathrm{Cov}(r^{(l)}, s^{(m)}).
\end{aligned}$$

Applying the reverse triangle inequality and Assumption G.5 (which bounds the residual covariance by $\delta \in (0, 1)$ fraction of the common-mode term), we obtain the lower bounds

$$\begin{aligned}
\|\mathrm{Cov}(q^{(l)}, q^{(m)})\|_{\mathrm{HS}} &\geq |a_l a_m|\,\|\mathrm{Cov}(f, f)\|_{\mathrm{HS}} - \|\mathrm{Cov}(r^{(l)}, r^{(m)})\|_{\mathrm{HS}}\\
&\geq (1 - \delta)\,|a_l a_m|\,\|\mathrm{Cov}(f, f)\|_{\mathrm{HS}}, \qquad (19\text{-}1)\\
\|\mathrm{Cov}(k^{(l)}, k^{(m)})\|_{\mathrm{HS}} &\geq |b_l b_m|\,\|\mathrm{Cov}(g, g)\|_{\mathrm{HS}} - \|\mathrm{Cov}(s^{(l)}, s^{(m)})\|_{\mathrm{HS}}\\
&\geq (1 - \delta)\,|b_l b_m|\,\|\mathrm{Cov}(g, g)\|_{\mathrm{HS}}, \qquad (19\text{-}2)\\
\|\mathrm{Cov}(q^{(l)}, k^{(m)})\|_{\mathrm{HS}} &\geq |a_l b_m|\,\|\mathrm{Cov}(f, g)\|_{\mathrm{HS}} - \|\mathrm{Cov}(r^{(l)}, s^{(m)})\|_{\mathrm{HS}}\\
&\geq (1 - \delta)\,|a_l b_m|\,\|\mathrm{Cov}(f, g)\|_{\mathrm{HS}}, \qquad (19\text{-}3)
\end{aligned}$$

Step 3: ratio bound.

By Assumption G.6, all denominators in Step 2 are strictly positive, so division is legitimate. Dividing the upper bounds in Equations (18-1)–(18-3) by the corresponding lower bounds in Equations (19-1)–(19-3), we obtain the ratio bounds

$$\frac{\|\mathrm{Cov}(\Delta q^{(l)}, \Delta q^{(m)})\|_{\mathrm{HS}}}{\|\mathrm{Cov}(q^{(l)}, q^{(m)})\|_{\mathrm{HS}}} \le \frac{\eta^2}{1-\delta} + \frac{\Gamma_q\, \psi(|l-m|)}{(1-\delta)\,|a_l a_m|\,\|\mathrm{Cov}(f,f)\|_{\mathrm{HS}}} = \frac{\eta^2}{1-\delta} + \mathcal{R}_{qq}(l,m), \tag{20-1}$$

$$\frac{\|\mathrm{Cov}(\Delta k^{(l)}, \Delta k^{(m)})\|_{\mathrm{HS}}}{\|\mathrm{Cov}(k^{(l)}, k^{(m)})\|_{\mathrm{HS}}} \le \frac{\eta^2}{1-\delta} + \frac{\Gamma_k\, \psi(|l-m|)}{(1-\delta)\,|b_l b_m|\,\|\mathrm{Cov}(g,g)\|_{\mathrm{HS}}} = \frac{\eta^2}{1-\delta} + \mathcal{R}_{kk}(l,m), \tag{20-2}$$

$$\frac{\|\mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(m)})\|_{\mathrm{HS}}}{\|\mathrm{Cov}(q^{(l)}, k^{(m)})\|_{\mathrm{HS}}} \le \frac{\eta^2}{1-\delta} + \frac{\Gamma_{qk}\, \psi(|l-m|)}{(1-\delta)\,|a_l b_m|\,\|\mathrm{Cov}(f,g)\|_{\mathrm{HS}}} = \frac{\eta^2}{1-\delta} + \mathcal{R}_{qk}(l,m), \tag{20-3}$$

where $\mathcal{R}_{qq}(l,m)$, $\mathcal{R}_{kk}(l,m)$, and $\mathcal{R}_{qk}(l,m)$ denote residual terms arising from the dependence of innovation increments, defined by

$$\mathcal{R}_{qq}(l,m) := \frac{\Gamma_q\, \psi(|l-m|)}{(1-\delta)\,|a_l a_m|\,\|\mathrm{Cov}(f,f)\|_{\mathrm{HS}}},$$
$$\mathcal{R}_{kk}(l,m) := \frac{\Gamma_k\, \psi(|l-m|)}{(1-\delta)\,|b_l b_m|\,\|\mathrm{Cov}(g,g)\|_{\mathrm{HS}}},$$
$$\mathcal{R}_{qk}(l,m) := \frac{\Gamma_{qk}\, \psi(|l-m|)}{(1-\delta)\,|a_l b_m|\,\|\mathrm{Cov}(f,g)\|_{\mathrm{HS}}}.$$

Equivalently, Equations (20-1)–(20-3) can be rewritten as

$$\|\mathrm{Cov}(\Delta X^{(l)}, \Delta Y^{(m)})\|_{\mathrm{HS}} \le \Big(\frac{\eta^2}{1-\delta} + \mathcal{R}_{XY}(l,m)\Big)\|\mathrm{Cov}(X^{(l)}, Y^{(m)})\|_{\mathrm{HS}},$$

for $(X,Y) \in \{(q,q),(k,k),(q,k)\}$.

(II) Exact fourth-moment decomposition for $\mathbb{E}[S_l S_m]$ and its bound. Under Assumption F.7, all covariance matrices below coincide with the corresponding raw second moments. In this part, we show that the baseline-relative decorrelation established for the inter-layer differences $(\Delta q^{(l)}, \Delta k^{(l)})$ in Part (I) propagates through the scaled inner-product

$$S_l := \frac{1}{\sqrt{d}} (\Delta q^{(l)})^\top \Delta k^{(l)}.$$

Specifically, we analyze the mixed score moment $\mathbb{E}[S_l S_m]$ and show that it admits an exact fourth-moment decomposition into terms depending only on second-order cross-layer covariances of the delta features, together with a residual term capturing non-Gaussian fourth-order effects [38].

To this end, we first rewrite the mixed-layer product in trace form:

$$S_l S_m = \frac{1}{d} (\Delta q^{(l)})^\top \Delta k^{(l)} (\Delta q^{(m)})^\top \Delta k^{(m)} = \frac{1}{d}\,\mathrm{Tr}\Big(\Delta q^{(l)} (\Delta q^{(m)})^\top \Delta k^{(m)} (\Delta k^{(l)})^\top\Big).$$

Taking expectations (justified by Assumption G.8), and applying the standard fourth-moment decomposition for products of inner products [38], we obtain

$$\begin{aligned}\mathbb{E}[S_l S_m] &= \frac{1}{d}\,\mathrm{Tr}\Big(\mathbb{E}\big[\Delta q^{(l)} (\Delta q^{(m)})^\top \Delta k^{(m)} (\Delta k^{(l)})^\top\big]\Big) \\ &= \frac{1}{d}\,\mathrm{Tr}\Big(\mathrm{Cov}(\Delta q^{(l)}, \Delta q^{(m)})\mathrm{Cov}(\Delta k^{(m)}, \Delta k^{(l)})^\top + \mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(m)})\mathrm{Cov}(\Delta q^{(m)}, \Delta k^{(l)})^\top \\ &\qquad + \mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(l)})\mathrm{Cov}(\Delta q^{(m)}, \Delta k^{(m)})^\top + E_{lm}\Big).\end{aligned} \tag{21}$$

This yields an exact decomposition of $\mathbb{E}[S_l S_m]$ into three second-order covariance pairing terms involving delta features and a fourth-moment remainder $\mathrm{Tr}(E_{lm})/d$. Here the matrix $E_{lm} \in \mathbb{R}^{d\times d}$ denotes the fourth-moment decomposition residual guaranteed by Assumption F.9, defined by

$$\begin{aligned} E_{lm} := \mathbb{E}\Big[\Delta q^{(l)}(\Delta q^{(m)})^\top \Delta k^{(m)}(\Delta k^{(l)})^\top\Big] &- \mathrm{Cov}(\Delta q^{(l)}, \Delta q^{(m)})\,\mathrm{Cov}(\Delta k^{(m)}, \Delta k^{(l)})^\top \\ &- \mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(m)})\,\mathrm{Cov}(\Delta q^{(m)}, \Delta k^{(l)})^\top \\ &- \mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(l)})\,\mathrm{Cov}(\Delta q^{(m)}, \Delta k^{(m)})^\top. \end{aligned}$$

Applying the Cauchy–Schwarz inequality for the Hilbert–Schmidt inner product, $|\,\mathrm{Tr}(AB^\top)| \le \|A\|_{\mathrm{HS}}\|B\|_{\mathrm{HS}}$, to each trace term in Equation(21), and using the baseline-relative covariance bounds (Equations(20-1)–(20-3)), we obtain

$$\begin{aligned} |\mathbb{E}[S_l S_m]| &\le \frac{1}{d}\Big(\|\mathrm{Cov}(\Delta q^{(l)}, \Delta q^{(m)})\|_{\mathrm{HS}}\,\|\mathrm{Cov}(\Delta k^{(l)}, \Delta k^{(m)})\|_{\mathrm{HS}} \\ &\qquad + \|\mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(m)})\|_{\mathrm{HS}}\,\|\mathrm{Cov}(\Delta q^{(m)}, \Delta k^{(l)})\|_{\mathrm{HS}}\Big) \\ &\quad + \underbrace{\frac{1}{d}\|\mathrm{Cov}(\Delta q^{(l)}, \Delta k^{(l)})\|_{\mathrm{HS}}\,\|\mathrm{Cov}(\Delta q^{(m)}, \Delta k^{(m)})\|_{\mathrm{HS}} + \frac{1}{d}|\,\mathrm{Tr}(E_{lm})|}_{=:\,\mathcal{T}_{lm}} \\ &\le \Big(\frac{\eta^2}{1-\delta} + \mathcal{R}_{qq}(l,m)\Big)\Big(\frac{\eta^2}{1-\delta} + \mathcal{R}_{kk}(l,m)\Big)\cdot\frac{1}{d}\|\mathrm{Cov}(q^{(l)}, q^{(m)})\|_{\mathrm{HS}}\|\mathrm{Cov}(k^{(l)}, k^{(m)})\|_{\mathrm{HS}} \\ &\quad + \Big(\frac{\eta^2}{1-\delta} + \mathcal{R}_{qk}(l,m)\Big)\Big(\frac{\eta^2}{1-\delta} + \mathcal{R}_{qk}(m,l)\Big)\cdot\frac{1}{d}\|\mathrm{Cov}(q^{(l)}, k^{(m)})\|_{\mathrm{HS}}\|\mathrm{Cov}(q^{(m)}, k^{(l)})\|_{\mathrm{HS}} \\ &\quad + \mathcal{T}_{lm} \\ &\le \alpha_{lm}\,\mathcal{B}_{lm} + \mathcal{T}_{lm}, \end{aligned} \tag{22}$$

where

$$\alpha_{lm} := \max\left\{\Big(\frac{\eta^2}{1-\delta}+\mathcal{R}_{qq}(l,m)\Big)\Big(\frac{\eta^2}{1-\delta}+\mathcal{R}_{kk}(l,m)\Big),\ \Big(\frac{\eta^2}{1-\delta}+\mathcal{R}_{qk}(l,m)\Big)\Big(\frac{\eta^2}{1-\delta}+\mathcal{R}_{qk}(m,l)\Big)\right\},$$

and

$$\mathcal{B}_{lm} := \frac{1}{d}\max\left\{\|\mathrm{Cov}(q^{(l)}, q^{(m)})\|_{\mathrm{HS}}\,\|\mathrm{Cov}(k^{(l)}, k^{(m)})\|_{\mathrm{HS}},\ \|\mathrm{Cov}(q^{(l)}, k^{(m)})\|_{\mathrm{HS}}\,\|\mathrm{Cov}(q^{(m)}, k^{(l)})\|_{\mathrm{HS}}\right\}.$$

This bound shows that the reduction in cross-layer delta-feature covariances established in Part (I) propagates to a corresponding reduction in the upper bound on the mixed score moment, up to the residual term $\mathcal{T}_{lm}$.

Remark G.1 (Interpretation). Part (II) shows that the cross-moment $|\mathbb{E}[S_l S_m]|$ admits an explicit upper bound in terms of the same cross-layer covariance quantities that appear in Part (I), up to a fourth-moment residual. Consequently, whenever inter-layer differencing reduces these covariance bounds, the corresponding upper bound on the score cross-moment decreases accordingly.

(III) From $\mathbb{E}[S_l S_m]$ to population centered kernel alignment. We transfer the mixed score moment bound obtained in Part (II) to a bound on the population centered alignment between the exponential kernels $\widetilde{K}^{(l)} := e^{S_l}$ and $\widetilde{K}^{(m)} := e^{S_m}$.

For each $t \in \{l, m\}$, consider the exponential kernel $\widetilde{K}^{(t)} := e^{S_t}$ and its population-centered version $\widetilde{K}_c^{(t)} := \mathcal{C}_\Pi(\widetilde{K}^{(t)})$. Introduce the exact first-order Taylor remainder $r(x) := e^x - 1 - x$. so that the identity $e^x = 1 + x + r(x)$ holds for all real x. By Taylor's theorem, $|r(x)| \le \frac{x^2}{2}e^{|x|}$ for all $x \in \mathbb{R}$. Define $U_t := \mathcal{C}_\Pi S_t$ and $R_t := \mathcal{C}_\Pi(r(S_t))$.

Under Assumption G.10(c), the Taylor remainder satisfies $r(S_t) \in L^2(\Pi)$, and hence $R_t \in L^2(\Pi)$ is well defined. Since $\mathcal{C}_\Pi$ is linear and satisfies $\mathcal{C}_\Pi(1) = 0$, the centered kernel admits the exact decomposition

$$\widetilde{K}_c^{(t)} = \mathcal{C}_\Pi(e^{S_t}) = \mathcal{C}_\Pi(1 + S_t + r(S_t)) = U_t + R_t. \tag{23}$$

Assume $\|U_l\|_2, \|U_m\|_2 > 0$ and define

$$\eta_{\exp} := \max\left\{ \frac{\|R_l\|_2}{\|U_l\|_2}, \frac{\|R_m\|_2}{\|U_m\|_2} \right\} \in (0,1).$$

Recalling the decomposition in Equation (23), Lemma G.2 yields the numerator bound

$$\left|\mathbb{E}[\widetilde{K}_c^{(l)} \widetilde{K}_c^{(m)}] - \mathbb{E}[U_l U_m]\right| \le \|U_l\|_2 \|R_m\|_2 + \|U_m\|_2 \|R_l\|_2 + \|R_l\|_2 \|R_m\|_2.$$

Moreover, by Lemma G.3,

$$\|\widetilde{K}_c^{(l)}\|_2 \, \|\widetilde{K}_c^{(m)}\|_2 \ge (1-\eta_{\exp})^2 \|U_l\|_2 \|U_m\|_2.$$

Combining these bounds with the definition of population centered alignment, we obtain the alignment perturbation inequality

$$\left|\rho(\widetilde{K}^{(l)}, \widetilde{K}^{(m)}) - \rho(U_l, U_m)\right| \le \frac{6}{(1-\eta_{\exp})^2} \left(\frac{\|R_l\|_2}{\|U_l\|_2} + \frac{\|R_m\|_2}{\|U_m\|_2} \right). \tag{24}$$

Next, define the conditional means with respect to the joint distribution Π:

$$a_t(k) := \mathbb{E}[S_t(Q,K) \mid K = k], \qquad b_t(q) := \mathbb{E}[S_t(Q,K) \mid Q = q], \qquad \mu_t := \mathbb{E}[S_t(Q,K)],$$

and set

$$\beta_t := \|a_t\|_2 + \|b_t\|_2 + |\mu_t|.$$

Under Assumption G.11, these quantities are finite and satisfy $U_t = S_t - a_t - b_t + \mu_t$ almost surely. Then, apply Corollary G.1 to upper bound $|\rho(U_l, U_m)|$, which leads

$$\begin{aligned} |\rho(U_l, U_m)| &= \frac{|\mathbb{E}[U_l U_m]|}{\|U_l\|_2 \|U_m\|_2} \\ &\le \frac{|\mathbb{E}[S_l S_m]|}{\|U_l\|_2 \|U_m\|_2} + \frac{\beta_l}{\|U_l\|_2} + \frac{\beta_m}{\|U_m\|_2} + \frac{\beta_l \beta_m}{\|U_l\|_2 \|U_m\|_2}. \end{aligned} \tag{25}$$

Combining Equations (24) and (25), and substituting the mixed moment bound from Part (II),

$$|\mathbb{E}[S_l S_m]| \le \alpha_{lm} \mathcal{B}_{lm} + \mathcal{T}_{lm},$$

we obtain the following conditional bound for the population centered alignment between the exponential kernels:

$$\begin{aligned} |\rho(\widetilde{K}^{(l)}, \widetilde{K}^{(m)})| &\le \frac{\alpha_{lm} \mathcal{B}_{lm} + \mathcal{T}_{lm}}{\|U_l\|_2 \|U_m\|_2} + \mathcal{R}_{lm}^{(\text{bridge+exp})} \\ &= \alpha_{lm} \underbrace{\frac{\mathcal{B}_{lm}}{\|U_l\|_2 \|U_m\|_2}}_{=: \widetilde{\mathcal{B}}_{lm}} + \underbrace{\frac{\mathcal{T}_{lm}}{\|U_l\|_2 \|U_m\|_2} + \mathcal{R}_{lm}^{(\text{bridge+exp})}}_{=: \widetilde{\mathcal{R}}_{lm}}, \end{aligned} \tag{26}$$

$$= \alpha_{lm} \widetilde{\mathcal{B}}_{lm} + \widetilde{\mathcal{R}}_{lm}, \tag{27}$$

where

$$\mathcal{R}_{lm}^{(\text{bridge+exp})} := \frac{\beta_l}{\|U_l\|_2} + \frac{\beta_m}{\|U_m\|_2} + \frac{\beta_l \beta_m}{\|U_l\|_2 \|U_m\|_2} + \frac{6}{(1-\eta_{\exp})^2} \left(\frac{\|R_l\|_2}{\|U_l\|_2} + \frac{\|R_m\|_2}{\|U_m\|_2} \right).$$

Remark G.2 (What the bound does and does not claim). Theorem 3.1 is *conditional* and fully explicit. It states that the exponential-kernel CKA is small provided: (i) the baseline-relative contraction coefficient α_{lm}, which governs the mixed scaled dot-product moment $\mathbb{E}[S_l S_m]$, is small; (ii) the centering bridge terms β_l, β_m are small relative to $\|U_l\|_{L^2}, \|U_m\|_{L^2}$; and (iii) the exponential remainder ratios $\|R_i\|_{L^2} / \|U_i\|_{L^2}$ are small. Importantly, it does *not* assert that small $\mathbb{E}[S_l S_m]$ alone implies small kernel alignment, nor does it claim monotone decrease of the true alignment across layers; it provides an explicit upper bound whose right-hand side scales down as the controlled quantities scale down.

G.8 Proof of Lemma 3.1

This subsection proves that the scaled dot product of MoDD, defined as a weighted combination of layer-wise delta inner products, induces a positive semidefinite kernel that admits an explicit finite product representation in terms of exponential factor kernels. Specifically, we show that the attention mechanism in MoDD can be interpreted as a product-kernel construction over layer-wise delta features. This result characterizes the attention kernel induced by MoDD's gated query–key construction. It should not be interpreted as a characterization of the full Transformer block, whose value mixing, output projection, residual connections, normalization, and MLP layers lie outside the theorem and are evaluated empirically in the full architecture.

To clarify the scope of the PSD result, Lemma 3.1 establishes positive semidefiniteness of K_{MoDD} on its representation-space domain and does not assume identical query and key projections. In the actual attention mechanism, this symmetric PSD kernel function is cross-evaluated on distinct sets of query and key representations, $K_{ij} = K_{\mathrm{MoDD}}(q_i, k_j)$; consequently, the resulting cross-kernel matrix need not itself be symmetric. Thus, distinct learned query and key projections do not conflict with the PSD property of the representation-space kernel. The PSD claim is therefore restricted to this representation-space kernel and does not extend directly to a kernel on the original input space.

Proof. By definition and bilinearity of the inner product,

$$\begin{aligned} S_{\mathrm{MoDD}}(q,k) &:= \frac{1}{\sqrt{d}}\Big(\sum_{i=2}^{l} \lambda^{(i)}\,\Delta q^{(i)}\Big)^{\top}\Big(\sum_{j=2}^{l} \lambda^{(j)}\,\Delta k^{(j)}\Big) \\ &= \sum_{i=2}^{l}\sum_{j=2}^{l} \frac{\lambda^{(i)}\lambda^{(j)}}{\sqrt{d}}\,(\Delta q^{(i)})^{\top}(\Delta k^{(j)}). \end{aligned}$$

Using $\exp(\sum_r a_r) = \prod_r \exp(a_r)$,

$$\begin{aligned} K_{\mathrm{MoDD}}(q,k) &= \exp\big(S_{\mathrm{MoDD}}(q,k)\big) \\ &= \prod_{i=2}^{l}\prod_{j=2}^{l} \exp\!\left(\frac{\lambda^{(i)}\lambda^{(j)}}{\sqrt{d}}\,(\Delta q^{(i)})^{\top}(\Delta k^{(j)})\right). \end{aligned}$$

For each pair $(i,j) \in \{2,\dots,l\}^2$, define the factor kernel $K^{(i,j)}$ on $\mathbb{R}^d \times \mathbb{R}^d$ by

$$K^{(i,j)}(u,v) := \exp\!\left(\frac{\lambda^{(i)}\lambda^{(j)}}{\sqrt{d}}\,u^{\top}v\right), \qquad u,v \in \mathbb{R}^d.$$

We claim that each $K^{(i,j)}$ is positive semidefinite. Indeed, for any $\alpha \geq 0$, the function $(u,v) \mapsto \exp(\alpha\, u^{\top}v)$ is positive semidefinite on $\mathbb{R}^d$, since

$$\exp(\alpha\, u^{\top}v) = \sum_{n=0}^{\infty} \frac{\alpha^n}{n!}(u^{\top}v)^n,$$

and $(u^{\top}v)^n$ is positive semidefinite for each n (it is the kernel induced by the feature map $u \mapsto u^{\otimes n}$), while all coefficients $\alpha^n/n!$ are nonnegative. Taking $\alpha = \lambda^{(i)}\lambda^{(j)}/\sqrt{d} \geq 0$ yields the claim.

Next, view K_{MoDD} as a function on $X \times X$ via stacked features

$$X := \prod_{i=2}^{l} \mathbb{R}^d.$$

Define $x := q_{\mathrm{stk}} = (x_{l-1},\dots,x_2) \in X$ and $y := k_{\mathrm{stk}} = (y_{l-1},\dots,y_2) \in X$ with

$$x_i := \Delta q^{(i)}, \qquad y_j := \Delta k^{(j)}, \quad i,j \in \{2,\dots,l\}.$$

For each pair $(i,j) \in \{2,\dots,l\}^2$, define the kernel $K^{(i,j)}$ on X by

$$K^{(i,j)}(x,y) := \exp\!\left(\frac{\lambda^{(i)}\lambda^{(j)}}{\sqrt{d}}\,x_i^{\top}y_j\right).$$

Each $K^{(i,j)}$ is a kernel on X, since it is the composition of the positive semidefinite kernel on $\mathbb{R}^d$ with the coordinate projections $(x, y) \mapsto (x_i, y_j)$. Finally, by closure of kernels under pointwise products (Lemma G.4), the finite product

$$\prod_{i=2}^{l} \prod_{j=2}^{l} K^{(i,j)}(x, y)$$

is a kernel on X. By the factorization above, this product equals $K_{\mathrm{MoDD}}(q, k)$ when $x = q_{\mathrm{stk}}$ and $y = k_{\mathrm{stk}}$. Thus K_{MoDD} is a positive semidefinite kernel on X. □

G.9 Proof of Theorem 3.2

This subsection studies kernel ridge regression with product kernels. We show that when the base kernels have weak mutual alignment, the product kernel admits an explicit variance upper bound that is smaller, yielding a statistical advantage in learning.

This analysis is based on a fixed-kernel KRR estimator associated with the representation-space MoDD kernel. It should not be interpreted as an end-to-end generalization guarantee for the full architecture.

We give a complete proof by first deriving a finite-sample bias–variance decomposition and then refining each term separately using the product-kernel structure, effective-dimension control, and alignment-sensitive second-moment bounds.

(I) Learning setup and KRR estimator. Let $\mathcal{X}$ denote the representation-space domain of K_{MoDD}, and ρ a probability measure on $\mathcal{X}$. We observe i.i.d. samples $(x_i, y_i)_{i=1}^n$ generated by

$$x_i \overset{\text{i.i.d.}}{\sim} \rho, \qquad y_i = f^*(x_i) + \xi_i, \qquad \mathbb{E}[\xi_i \mid x_i] = 0. \tag{28}$$

Define the MoDD product kernel

$$K_{\mathrm{MoDD}}(x, x') := \prod_{i=2}^{l} \prod_{j=2}^{l} K^{(i,j)}\big(x^{(i)}, x'^{(j)}\big), \qquad K^{(i,j)}(u, v) = \exp\left(\frac{\lambda^{(i)} \lambda^{(j)}}{\sqrt{d}}\, u^\top v \right). \tag{29}$$

Then K_{MoDD} is PSD by closure of kernels under pointwise products, and its RKHS is denoted $\mathcal{H}_{\mathrm{MoDD}}$.

Let $K \in \mathbb{R}^{n \times n}$ be the Gram matrix of K_{MoDD} on $x_{1:n}$: $K_{ij} = K_{\mathrm{MoDD}}(x_i, x_j)$. For a test point $x \in \mathcal{X}$ let $k(x) := (K_{\mathrm{MoDD}}(x, x_1), \ldots, K_{\mathrm{MoDD}}(x, x_n))^\top$. For $\lambda > 0$, the kernel ridge regression estimator is

$$\hat{f}_{\lambda,n} = \arg\min_{f \in \mathcal{H}_{\mathrm{MoDD}}} \left\{ \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \|f\|^2_{\mathcal{H}_{\mathrm{MoDD}}} \right\}.$$

By the representer theorem, $\hat{f}_{\lambda,n}(\cdot) = \sum_{i=1}^n \alpha_i K_{\mathrm{MoDD}}(\cdot, x_i)$ with

$$\alpha = (K + n\lambda I_n)^{-1} y. \tag{30}$$

(II) Population ridge solution and bias–variance decomposition. Define the population risk $R(f) = \mathbb{E}[(Y - f(X))^2]$ and the population ridge solution for $\lambda > 0$,

$$f_\lambda := \arg\min_{f \in \mathcal{H}_{\mathrm{MoDD}}} \left\{ \mathbb{E}[(Y - f(X))^2] + \lambda \|f\|^2_{\mathcal{H}_{\mathrm{MoDD}}} \right\}.$$

By $R(f) - R(f^*) = \mathbb{E}[(f(X) - f^*(X))^2]$, inserting f_λ and using $(a+b)^2 \le 2a^2 + 2b^2$ yield

$$\begin{aligned} \mathbb{E}[R(\hat{f}_{\lambda,n}) - R(f^*)] &= \mathbb{E}[(\hat{f}_{\lambda,n}(X) - f^*(X))^2] \\ &\le 2\,\mathbb{E}[(f_\lambda(X) - f^*(X))^2] + 2\,\mathbb{E}[(\hat{f}_{\lambda,n}(X) - f_\lambda(X))^2], \end{aligned} \tag{31}$$

To control the second term, define the noise-averaged predictor

$$\bar{f}_{\lambda,n}(X) := \mathbb{E}[\hat{f}_{\lambda,n}(X) \mid x_{1:n}, X] = k(X)^\top (K + n\lambda I_n)^{-1} f^*(x_{1:n}). \tag{32}$$

Then

$$\hat{f}_{\lambda,n}(X) = \bar{f}_{\lambda,n}(X) + k(X)^\top (K + n\lambda I_n)^{-1} \xi.$$

Conditioning on $(x_{1:n}, X)$ and applying the variance identity yields

$$\begin{aligned} \mathbb{E}[(\hat{f}_{\lambda,n}(X) - f_\lambda(X))^2 \mid x_{1:n}, X] &= (\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2 + \mathrm{Var}(\hat{f}_{\lambda,n}(X) \mid x_{1:n}, X) \\ &\le (\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2 + \tau^2\, k(X)^\top (K + n\lambda I_n)^{-2} k(X), \end{aligned} \tag{33}$$

where the inequality follows from Assumption G.12 (conditional independence and bounded conditional variance of the noise).

Taking expectations gives

$$\mathbb{E}[(\hat{f}_{\lambda,n}(X) - f_\lambda(X))^2] \le \mathbb{E}[(\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2] + \tau^2\,\mathbb{E}\Big[k(X)^\top (K + n\lambda I_n)^{-2} k(X)\Big]. \quad (34)$$

Substituting this bound into Equation (31) yields the finite-sample decomposition for all $\lambda > 0$

$$\mathbb{E}\big[R(\hat{f}_{\lambda,n}) - R(f^*)\big] \le 2\,\underbrace{\mathbb{E}[(f_\lambda(X) - f^*(X))^2]}_{\text{population approximation (bias)}} + 2\,\underbrace{\mathbb{E}[(\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2]}_{\text{sample-to-population error}} + 2\tau^2\,\underbrace{\mathbb{E}\Big[k(X)^\top (K + n\lambda I_n)^{-2} k(X)\Big]}_{\text{noise variance proxy}}, \quad (35)$$

where

- $\mathbb{E}[(f_\lambda(X) - f^*(X))^2]$ is the *population approximation (bias)* term, measuring the approximation error of the population ridge solution f_λ relative to the target f^*;
- $\mathbb{E}[(\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2]$ is the *sample-to-population error*, capturing the discrepancy between the finite-sample estimator and its population counterpart;
- $\mathbb{E}\big[k(X)^\top (K + n\lambda I_n)^{-2} k(X)\big]$ is the *noise variance proxy*, which quantifies the contribution of observation noise through the stability of the kernel ridge estimator.

Step 1: Population approximation (bias).

We first control the population approximation term $\mathbb{E}[(f_\lambda(X) - f^*(X))^2]$. By definition, the population ridge solution f_λ minimizes

$$\mathbb{E}[(f(X) - f^*(X))^2] + \lambda \|f\|^2_{\mathcal{H}_{\mathrm{MoDD}}}$$

over $f \in \mathcal{H}_{\mathrm{MoDD}}$. Evaluating this objective at $f = f^*$ yields

$$\mathbb{E}[(f_\lambda(X) - f^*(X))^2] + \lambda \|f_\lambda\|^2_{\mathcal{H}_{\mathrm{MoDD}}} \le \lambda \|f^*\|^2_{\mathcal{H}_{\mathrm{MoDD}}},$$

and therefore

$$\mathbb{E}[(f_\lambda(X) - f^*(X))^2] \le \lambda \|f^*\|^2_{\mathcal{H}_{\mathrm{MoDD}}}. \quad (36)$$

We now bound $\|f^*\|_{\mathcal{H}_{\mathrm{MoDD}}}$ under the assumed product structure $f^*(x) = \prod_{\ell=1}^L f_\ell^*(x)$ with $f_\ell^* \in \mathcal{H}_\ell$. For each $\ell \in \{1, \dots, L\}$, since $K^{(\ell)}$ is a positive semidefinite kernel, there exists a Hilbert space $\mathcal{F}_\ell$ and a feature map $\phi_\ell : \mathcal{X} \to \mathcal{F}_\ell$ such that

$$K^{(\ell)}(x, x') = \langle \phi_\ell(x), \phi_\ell(x') \rangle_{\mathcal{F}_\ell}.$$

Define the Hilbert tensor-product space $\mathcal{F} = \widehat{\otimes}_{\ell=1}^L \mathcal{F}_\ell$ and the feature map $\Phi(x) = \bigotimes_{\ell=1}^L \phi_\ell(x) \in \mathcal{F}$. Then the product kernel satisfies

$$K_\Pi(x, x') = \prod_{\ell=1}^L K^{(\ell)}(x, x') = \prod_{\ell=1}^L \langle \phi_\ell(x), \phi_\ell(x') \rangle_{\mathcal{F}_\ell} = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{F}},$$

so $\mathcal{H}_{\mathrm{MoDD}}$ is the reproducing kernel Hilbert space induced by Φ.

For each ℓ, since $f_\ell^* \in \mathcal{H}_\ell$, there exists $w_\ell \in \mathcal{F}_\ell$ such that $f_\ell^*(x) = \langle w_\ell, \phi_\ell(x) \rangle_{\mathcal{F}_\ell}$ and $\|f_\ell^*\|_{\mathcal{H}_\ell} = \inf\{\|w\|_{\mathcal{F}_\ell} : f_\ell^* = \langle w, \phi_\ell(\cdot) \rangle\}$. Defining $w = \bigotimes_{\ell=1}^L w_\ell \in \mathcal{F}$, we have for all $x \in \mathcal{X}$

$$\langle w, \Phi(x) \rangle_{\mathcal{F}} = \prod_{\ell=1}^L \langle w_\ell, \phi_\ell(x) \rangle_{\mathcal{F}_\ell} = \prod_{\ell=1}^L f_\ell^*(x) = f^*(x).$$

Moreover, $\|w\|_{\mathcal{F}} = \prod_{\ell=1}^L \|w_\ell\|_{\mathcal{F}_\ell}$, so

$$\|f^*\|_{\mathcal{H}_{\mathrm{MoDD}}} \le \|w\|_{\mathcal{F}}.$$

Combining this inequality with Equation (36) gives

$$\mathbb{E}[(f_\lambda(X) - f^*(X))^2] \le \lambda \prod_{\ell=1}^L \|f_\ell^*\|^2_{\mathcal{H}_\ell}. \quad (37)$$

Step 2: Sample-to-population error via effective dimension.

Let $T_{\mathrm{MoDD}} : L^2(\rho) \to L^2(\rho)$ denote the population integral operator associated with K_{MoDD},

$$(T_{\mathrm{MoDD}} f)(x) := \int K_{\mathrm{MoDD}}(x, x')\, f(x')\, d\rho(x').$$

Under Assumption G.13, T_{MoDD} is Hilbert–Schmidt and compact. We define the effective dimension

$$d_{\mathrm{MoDD}}(\lambda) = \mathrm{Tr}\big(T_{\mathrm{MoDD}}(T_{\mathrm{MoDD}} + \lambda I)^{-1}\big).$$

We now control the sample-to-population error term $\mathbb{E}[(\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2]$. Under Assumptions G.12 and G.13, the reference-based effective-dimension control stated in Lemma G.5 applies, yielding

$$\mathbb{E}[(\bar{f}_{\lambda,n}(X) - f_\lambda(X))^2] \le \frac{C_{\mathrm{ed}}}{n}\, d_{\mathrm{MoDD}}(\lambda). \tag{38}$$

Step 3(a): Noise variance proxy (spectral bounds).

We first control the noise variance proxy appearing in Equation (35). Under Assumptions G.12 and G.13, the reference-based effective-dimension control stated in Lemma G.5 yields that there exists a constant $C_{\mathrm{ed}} > 0$ such that, for all $\lambda > 0$,

$$\mathbb{E}\Big[k(X)^\top (K + n\lambda I_n)^{-2} k(X)\Big] \le \frac{C_{\mathrm{ed}}}{n\lambda}\, \mathrm{Tr}\big(T_{\mathrm{MoDD}}(T_{\mathrm{MoDD}} + \lambda I)^{-1}\big) = \frac{C_{\mathrm{ed}}}{n\lambda}\, d_{\mathrm{MoDD}}(\lambda). \tag{39}$$

Moreover, since the product kernel K_{MoDD} is positive semidefinite, it follows that for all $\lambda > 0$,

$$\mathbb{E}\Big[k(X)^\top (K + n\lambda I_n)^{-2} k(X)\Big] \le \frac{1}{n\lambda^2}\, \mathbb{E}_{X,X'}\big[K_{\mathrm{MoDD}}(X, X')^2\big], \tag{40}$$

where $X, X' \sim \rho$ are independent.

Step 3(b): Alignment-sensitive control of the second moment.

It remains to control the second moment $\mathbb{E}[K_{\mathrm{MoDD}}(X, X')^2]$ appearing in Equation (40) in a manner that explicitly reflects cross-layer alignment. Let $X, X' \sim \rho$ be independent and write

$$K_{\mathrm{MoDD}}(X, X') = \prod_{i=2}^{l} \prod_{j=2}^{l} K_{ij}, \qquad K_{ij} := K^{(i,j)}(X_i, X'_j).$$

Define $\mu_{ij} = \mathbb{E}[K_{ij}^2]$, $\widetilde{K}_{ij} = K_{ij} - \mathbb{E}[K_{ij}]$, $\sigma_{ij}^2 = \mathbb{E}[\widetilde{K}_{ij}^2]$, and pairwise alignments

$$\rho_{(i,j),(i',j')} = \frac{\mathbb{E}[\widetilde{K}_{ij}\widetilde{K}_{i'j'}]}{\sigma_{ij}\sigma_{i'j'}}.$$

Under Assumption G.13, Lemma G.7 yields the exact degree-2 expansion

$$\mathbb{E}[K_{\mathrm{MoDD}}(X, X')^2] = \prod_{i,j} \mu_{ij} + \sum_{(i,j)\neq(i',j')} \mathrm{Cov}(K_{ij}^2, K_{i'j'}^2) \prod_{(r,s)\neq(i,j),(i',j')} \mu_{rs} + \mathcal{E}. \tag{41}$$

where $\mu_{ij} = \mathbb{E}[K_{ij}^2]$ and the remainder term $\mathcal{E}_L$ is bounded as in Equation (17).

Furthermore, under Assumptions G.14, G.15, and G.16, Lemma G.6 provides the alignment-sensitive bound

$$\big|\mathrm{Cov}(K_{ij}^2, K_{i'j'}^2)\big| \le c_0\, \rho_{(i,j),(i',j')}^2\, \sigma_{ij}^2 \sigma_{i'j'}^2 + 4B^2\, \sigma_{ij}\sigma_{i'j'}\, |\rho_{(i,j),(i',j')}|. \tag{42}$$

Substituting Equation (42) into Equation (16) yields an explicit alignment-sensitive upper bound on $\mathbb{E}[K_{\mathrm{MoDD}}(X, X')^2]$. Moreover, the resulting right-hand side is nondecreasing in each $|\rho_{(i,j),(i',j')}|$ (and hence in each $\rho^2_{(i,j),(i',j')}$).

Define the alignment-sensitive second-moment upper bound

$$\mathcal{V}_{\mathrm{MoDD}} := \prod_{i,j} \mu_{ij} + \sum_{(i,j)\neq(i',j')} \Big(c_0 \rho^2_{(i,j),(i',j')} \sigma^2_{ij}\sigma^2_{i'j'} + 4B^2 \sigma_{ij}\sigma_{i'j'} |\rho_{(i,j),(i',j')}|\Big) \prod_{(r,s)\neq(i,j),(i',j')} \mu_{rs} + |\mathcal{E}|. \tag{43}$$

which satisfies

$$\mathbb{E}[K_{\mathrm{MoDD}}(X, X')^2] \le \mathcal{V}_{\mathrm{MoDD}}$$

(III) Final aggregation. For all $\lambda > 0$,

$$\mathbb{E}[R(\hat{f}_{\lambda,n}) - R(f^*)] \leq 2\lambda \|f^*\|^2_{\mathcal{H}_{\mathrm{MoDD}}} + \frac{2C_{\mathrm{ed}}}{n}\, d_{\mathrm{MoDD}}(\lambda) + 2\tau^2 \min\left\{ \frac{C_{\mathrm{ed}}}{n\lambda}\, d_{\mathrm{MoDD}}(\lambda),\ \frac{1}{n\lambda^2}\, \mathcal{V}_{\mathrm{MoDD}} \right\}. \tag{44}$$

This proves the theorem.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction are supported by the theoretical motivation in Section 3 and the empirical results in Section 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The conclusion discusses limitations and future directions.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The assumptions and full proofs for Theorem 3.1, Lemma 3.1, and Theorem 3.2 are provided in Appendix G. The main text summarizes the role of these results as theoretical motivation for the proposed design.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The architecture, model integrations, datasets, metrics, and training protocols are described in Section 4 and Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The experiments use public benchmark datasets and publicly available baseline implementations. MoDD is added as an attention module, with its operations and equations fully specified in the paper, making it straightforward to reproduce from the description.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Dataset details, train/test splits, model configurations, and implementation details are provided in Section 4 and Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The main results report means and standard deviations over three random seeds, as stated in the corresponding tables.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Parameter counts, GPU memory usage, and training time are reported in the appendix, with additional scaling analysis in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and are not aware of any violations. The work uses standard scientific benchmark datasets and does not involve human subjects, private data, or sensitive attributes.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

Justification: The paper does not include a dedicated broader impact discussion. The work focuses on neural operator architecture design and is evaluated on standard scientific benchmarks without human subjects, private data, or sensitive attributes.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk models or scraped datasets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The datasets and baseline implementations used in this work are publicly available research assets, and the original papers are cited in the main text and appendix. We follow the terms of use of the corresponding benchmark datasets and baseline repositories.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset or benchmark asset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human-subject research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects or identifiable personal data.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as part of the core method development.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.