PLATONIC TRANSFORMERS: A SOLID CHOICE FOR EQUIVARIANCE

Mohammad Mohaiminul Islam [1] **Rishabh Anand** [2*] **David R. Wessels** [4*]
Friso de Kruiff [4] **Thijs P. Kuipers** [3] **Rex Ying** [2] **Clara I. Sánchez** [1,3]
Sharvaree Vadgama [4†] **Georg Bökman** [4†] **Erik J. Bekkers** [4]

[1] QurAI, Univ. of Amsterdam [2] Yale University, USA
[3] BMEP, Amsterdam UMC [4] AMLab, Univ. of Amsterdam

Open-source code: github.com/niazoys/PlatonicTransformers

ABSTRACT

Transformers lack inductive biases for geometric symmetries common across scientific domains. Existing equivariant methods often sacrifice the efficiency and flexibility that make Transformers so scalable through complex, computationally intensive designs. We introduce the Platonic Transformer to resolve this long-standing trade-off. By defining attention relative to reference frames from *Platonic solid symmetry groups*, our method induces a principled weight-sharing scheme. This enables combined $E(3)$ equivariance to continuous translations and Platonic symmetries, while preserving the exact architecture and computational cost of a standard Transformer. Platonic Transformers achieve highly competitive results on molecular property prediction and unconditional generation tasks (QM9, OMol25), leveraging geometric constraints at no additional cost.

1 INTRODUCTION

The widespread success of Transformers Vaswani et al. (2017) lies in simple, general-purpose mechanisms that have matured over the years and continue to offer remarkable gains in speed and flexibility, benefiting from vast datasets and computational resources (Dosovitskiy et al., 2021; Jumper et al., 2021; Devlin et al., 2019). Yet, this very generality implies they are not inherently equipped to handle specific symmetries present in many scientific domains, such as physics and molecular chemistry, where performance can be significantly enhanced by incorporating inductive bias (Fuchs et al., 2020; Ying et al., 2021; Zhao et al., 2021; Bekkers et al., 2024; Balla et al., 2024; Liao et al., 2024; Romero & Cordonnier, 2021; Wessels et al., 2024; Bose et al., 2024; Zhdanov et al., 2024; Nyholm et al., 2025). The principle of symmetry, for example, has given rise to highly data-efficient and robust group equivariant networks (Cohen & Welling, 2016; 2017; Cesa et al., 2022). However, scaling these symmetry-aware networks has been difficult, as their reliance on operations like group convolutions or Clebsch-Gordan tensor products introduces significant computational overhead compared to standard architectures (He et al., 2021; Luo et al., 2024). This raises the question: *how can we leverage powerful geometric inductive biases within the Transformer architecture without sacrificing the speed and flexibility integral to its success?*

Existing approaches often achieve this by making complex architectural changes to equivariant networks that poorly scale or settle for invariant attention mechanisms which sacrifice feature representations for simplicity and computational efficiency (Masters et al., 2022; Assaad et al., 2023; Thölke & Fabritiis, 2022; Brehmer et al., 2023; Kundu & Kondor, 2025; Joshi et al., 2025). Recent efforts have also explored *hybrid architectures* that resort to symmetry breaking (Qu & Krishnapriyan, 2024; Lawrence et al., 2025) to improve scalability but require a careful mix of modules to maximize downstream performance.

Our main contribution is the *Platonic Transformer*, a framework that achieves equivariance to continuous translations and discrete roto-reflections in Transformers *without changing the underlying*

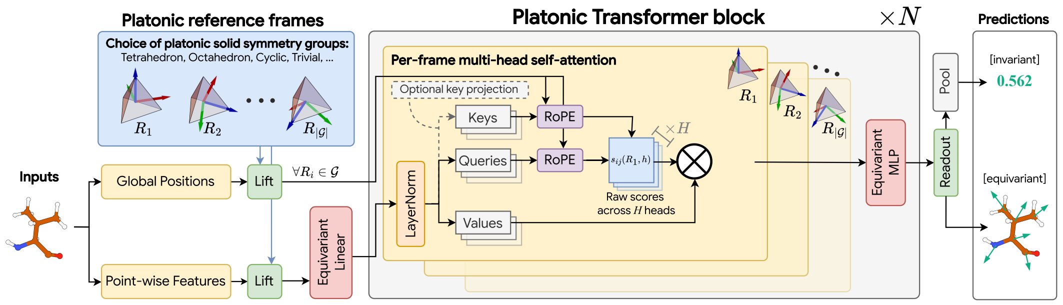


Figure 1: Visualization of Weight-Shared RoPE within the N-layer Platonic Transformer. Scalar and vector inputs are lifted to become functions on the *Platonic solid symmetry group* of choice (here, the Tetrahedral group). The same multi-head self-attention mechanism is applied in parallel, with each instance rotating the features according to a different reference frame $R_i \in \mathcal{G}$. Choosing the trivial group as $\mathcal{G}$ reduces this framework to a standard Transformer.

attention mechanism or computation graph. To achieve this, our method processes features relative to a collection of reference frames that form a Platonic symmetry group ($\mathcal{G} \subset O(3)$) and constrains all linear layers to be equivariant with respect to this choice of frame. As such we obtain general and efficient way to obtain equivariance to the semi-direct product group $T_3 \rtimes \mathcal{G} \subset E(3)$ of continuous translations T_3 and finite Platonic sub-groups $\mathcal{G}$ of $O(3)$. This principled scheme allows the standard attention block, including its unmodified Rotary Position Embeddings (RoPE), to operate in parallel across these frames, and effectively associates each reference frame with a distinct attention head. As a result, the model incorporates a geometric inductive bias without altering the architecture or computational footprint of a standard Transformer. This enables flexible usage at no additional cost, resolving the long-standing symmetry-awareness vs. scaling dilemma[1].

2 Background: Transformers with Position Embeddings

Vanilla Attention and Absolute Positioning Given a sequence of input features $\mathbf{f}_i \in \mathbb{R}^C$, the self-attention layer first computes query, key, and value vectors via linear projections: $\mathbf{q}_i = \mathbf{W}_Q \mathbf{f}_i$, $\mathbf{k}_j = \mathbf{W}_K \mathbf{f}_j$, $\mathbf{v}_j = \mathbf{W}_V \mathbf{f}_j$. Here, the learnable weight matrices are $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{C \times d}$ and $\mathbf{W}_V \in \mathbb{R}^{C \times C'}$. The output for the i-th feature, $\mathbf{y}_i \in \mathbb{R}^{C'}$, is a weighted sum of the value vectors, with weights determined by softmax-normalized dot products of queries and keys: $\mathbf{y}_i = \sum_{j=1}^{N} \mathrm{attn}(\mathbf{q}_i, \mathbf{k}_j)\mathbf{v}_j$, where $\mathrm{attn}(\mathbf{q}_i, \mathbf{k}_j) = \underset{j}{\mathrm{softmax}}\left(\mathbf{q}_i^\top \mathbf{k}_j\right)$. As this operation is permutation-equivariant, it is insensitive to the order of the inputs and must be modified to incorporate positional information for spatial tasks. A common approach is to use Absolute Positional Encodings (APE), where a unique vector $\mathbf{E}(\mathbf{p}_i)$ is added to each input feature, $\mathbf{f}_i' = \mathbf{f}_i + \mathbf{E}(\mathbf{p}_i)$, before the linear projections are applied. This encoding depends on absolute coordinates, making APE not translation-equivariant.

Rotary Position Embeddings (RoPE) RoPE (Su et al., 2024) takes a structured approach to positional encoding by modifying the query and key vectors with a position-dependent transformation, making the attention score explicitly dependent on relative positions. This transformation is constructed by stacking 2D rotation matrices, giving RoPE its name. To apply RoPE with positions $\mathbf{p}$ in dimension $n > 1$, we use a set of n-dimensional frequency vectors $\Omega = \{\boldsymbol{\omega}_k\}_{k=1}^{d/2}$, each defining a direction used to project $\mathbf{p}$ to 1D and a frequency used to apply 1D-RoPE in this direction. We obtain $d/2$ blocks, $\rho_{\boldsymbol{\omega}_k}(\mathbf{p}) = \begin{pmatrix} \cos(\boldsymbol{\omega}_k^\top \mathbf{p}) & -\sin(\boldsymbol{\omega}_k^\top \mathbf{p}) \\ \sin(\boldsymbol{\omega}_k^\top \mathbf{p}) & \cos(\boldsymbol{\omega}_k^\top \mathbf{p}) \end{pmatrix}$, which are stacked in a block-diagonal manner to form a single transformation matrix, $\boldsymbol{\rho}_\Omega(\mathbf{p}) = \mathrm{diag}(\rho_{\boldsymbol{\omega}_1}(\mathbf{p}), \ldots, \rho_{\boldsymbol{\omega}_{d/2}}(\mathbf{p}))$. Note that while $\boldsymbol{\rho}_\Omega(\mathbf{p})$ is a high-dimensional rotation, this rotation is not related to rotations of the position $\mathbf{p}$. In fact, $\boldsymbol{\rho}_\Omega$ is instead connected to translations of $\mathbf{p}$, formally discussed in Appendix A.3.

[1]We also show a formal connection between attention and dynamic group convolution that learns adaptive geometric filters, enabling a highly scalable, linear-time convolutional variant in Appendix E.

For a query $\mathbf{q}_i$ at position $\mathbf{p}_i$ and a key $\mathbf{k}_j$ at position $\mathbf{p}_j$, $\boldsymbol{\rho}_\Omega$ is applied before the dot product. As the operator $\boldsymbol{\rho}_\Omega$ is orthogonal and satisfies the homomorphism property[2] for translations, the interaction simplifies to depend only on relative positions: $(\boldsymbol{\rho}_\Omega(\mathbf{p}_i)\mathbf{q}_i)^\top (\boldsymbol{\rho}_\Omega(\mathbf{p}_j)\mathbf{k}_j) = \mathbf{q}_i^\top \boldsymbol{\rho}_\Omega(\mathbf{p}_i)^\top \boldsymbol{\rho}_\Omega(\mathbf{p}_j)\mathbf{k}_j = \mathbf{q}_i^\top \boldsymbol{\rho}_\Omega(\mathbf{p}_j - \mathbf{p}_i)\mathbf{k}_j$. This reveals the core property of RoPE: translation equivariance, a geometric bias that underlies its strong performance (Chen et al., 2023; Dai et al., 2019). We provide a formal group-theoretic construction of RoPE in Appendix A.[3]

3 The Platonic Transformer

We generalize RoPE from just being translation (T_3) equivariant to full $T_3 \rtimes \mathcal{G} \subset E(3)$ equivariance by redefining positional encodings relative to multiple local reference frames $\mathcal{G} \subset O(3)$, extending traditional single (trivial) frame RoPE without altering the attention mechanism and Transformer computation graph. See Figure 1.

Weight-sharing across RoPE Embeddings. To incorporate reference frames, input features—scalars or vectors—are lifted to become functions $\mathbf{f}_i : \mathcal{G} \to \mathbb{R}^C$ on the group $\mathcal{G}$, where with $\mathbf{f}_i(R)$ we denote the feature vector at point i viewed from the perspective of a reference frame $R \in \mathcal{G}$. Since we consider a finite collection of frames, this feature map is represented as a single tensor in $\mathbb{R}^{|\mathcal{G}| \times C}$, effectively adding a group axis of size $|\mathcal{G}|$. We achieve this by transforming the components of the original feature: vectors are projected onto the corresponding frame R, while invariant scalars are copied across all frames. Query, key, and value projections now operate on these group feature maps.

The key step for achieving equivariance to Euclidean transformations is making the RoPE operator itself dependent on the reference frame. This is achieved simply by projecting the positions $\mathbf{p}_i$ on R to obtain the positions $\mathbf{p}(R)_i = R^{-1}\mathbf{p}_i$ corresponding to $\mathbf{f}(R)_i$ that are defined relative to each frame. Similarly, queries $\mathbf{q}_i(R)$, keys $\mathbf{k}_j(R)$, and values $\mathbf{v}_j(R)$ are now also functions on the group. We can then compute the unnormalized attention scores, $s_{ij}(R)$, from the perspective of frame R:

$$s_{ij}(R) = \mathbf{q}_i(R)^\top \rho_\Omega(\mathbf{p}_j(R) - \mathbf{p}_i(R))\mathbf{k}_j(R) = \mathbf{q}_i(R)^\top \rho_\Omega((\mathbf{p}_j - \mathbf{p}_i)(R))\mathbf{k}_j(R)\,. \tag{1}$$

Scores for each frame are computed in parallel as their own independent attention head. Note that we can also obtain $s_{ij}(R)$ from the perspective of steering the base set of frequencies Ω instead of the positions $\mathbf{p}_i$, which we show in Appendix E.3. However, from our current perspective, the RoPE-attention mechanism itself remains completely unchanged from its traditional formulation described in Section 2; only the relative positions $\mathbf{p}_i - \mathbf{p}_j$ are now defined relative to each reference frame R. The attention coefficients are obtained by applying the softmax to the scores $s_{ij}(R)$. The output $\mathbf{y}_i(R)$ for each input feature $\mathbf{f}_i(R)$ is then given as $\mathbf{y}_i(R) = \sum_{j=1}^{N} \mathrm{attn}_{ij}(R)\mathbf{v}_j(R)$, where $\mathrm{attn}_{ij}(R) = \underset{j}{\mathrm{softmax}}(s_{ij}(R))$. This process naturally results in an output tensor in $\mathbb{R}^{|\mathcal{G}| \times C}$, where the features are defined relative to each frame. The projection matrices for queries, keys, and values $(\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V)$, and RoPE base frequencies (Ω) are shared across all reference frames, allowing the model to learn a single, generalizable attention mechanism applicable from any orientation.

Frame Selection via Platonic Solids. We select a suitable subgroup $\mathcal{G} \subset O(3)$ to serve as the reference frames: the discrete symmetry groups of regular polygons and polyhedra. We restrict our frames to the finite *rotational* symmetry groups ($\mathcal{G} \subset SO(3)$) of these Platonic solids: the *tetrahedral* (12 rotations), *octahedral* (24 rotations), and *icosahedral* (60 rotations) groups. The advantage of working with a *finite* group $\mathcal{G}$ is that its operations can be handled discretely and efficiently using *Cayley tables*. We assign a unique index $i \in \{0, \dots, |\mathcal{G}| - 1\}$ to each rotation $R_i \in \mathcal{G}$. The group product $R_i R_j = R_k$ can then be precomputed and stored in the Cayley table, a simple look-up table where $\mathrm{Cayley}[i, j] = k$. This discrete formalism makes the group action on our feature maps, which are functions on the group $f : \mathcal{G} \to \mathbb{R}^C$, extremely efficient. A detailed explanation of how we retrieve the group product is provided in Appendix C.2.

[2]The operator $\boldsymbol{\rho}_\Omega(\mathbf{p})$ being orthogonal means its inverse is its transpose: $\boldsymbol{\rho}_\Omega(\mathbf{p})^{-1} = \boldsymbol{\rho}_\Omega(\mathbf{p})^\top$. The homomorphism property for the translation group satisfies $\boldsymbol{\rho}_\Omega(\mathbf{p_i} + \mathbf{p_j}) = \boldsymbol{\rho}_\Omega(\mathbf{p_i})\boldsymbol{\rho}_\Omega(\mathbf{p_j})$.

[3]We analyze the formal connection between RoPE-attention and convolution to highlight its underlying inductive bias, in Appendix E. In this convolutional setting, the attention operator's complexity scales linearly with the number of tokens.

3.1 Equivariant Linear Layers and Fixed Computation Graph

For such a structure to be maintained, all linear transformations, including the query, key, and value projections ($\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$), and any MLP blocks, must also be equivariant. To satisfy the equivariance constraint, we replace standard matrix-vector multiplication with a *group convolution* (Cohen et al., 2019, Thm. 3.1). Such an equivariant linear layer acts on $\mathbf{f}_i(R)$ at each orientation $R \in \mathcal{G}$ as follows: $(\Phi(\mathbf{f}_i))(R) = \sum_{\tilde{R} \in \mathcal{G}} \mathbf{W}_{\text{group}}(R^{-1}\tilde{R})\, \mathbf{f}_i(\tilde{R})$. Here, $\mathbf{W}_{\text{group}} : \mathcal{G} \to \mathbb{R}^{C' \times C}$ is a learnable kernel defined on the group. We can write the equivariant linear layer as a standard linear layer from $|\mathcal{G}|C$ to $|\mathcal{G}|C'$ channels with weight-sharing. For a specific effective embedding dimension $|\mathcal{G}|C$, a larger group $\mathcal{G}$ therefore leads to fewer learnable parameters, but no extra computational cost compared to a standard layer. We provide a detailed explanation of this weight-sharing structure in Appendix C.1, and its amenability to GPU speed-ups in Appendix I.

With all components of the architecture now defined as equivariant operations, we formally state the key property of the full model, proven in Appendix B.

Proposition 1 (End-to-End Equivariance). *Our proposed Transformer architecture is an equivariant model. A global roto-reflection $R \in \mathcal{G}$ applied to the input point cloud results in a corresponding roto-reflection L_R of the output feature maps, which is a rotation of the reference frames.*

4 Experiments

We validate our proposed architecture on tasks with inherent symmetry, QM9 (Ramakrishnan et al., 2014) and OMol25 (Levine et al., 2025), where the molecular systems therein have no canonical orientation. Since the physical laws governing these molecular properties are $E(3)$-symmetric, incorporating equivariance naturally allows for efficient generalization (Fuchs et al., 2020; Bronstein et al., 2021; Pacini et al., 2025; Vadgama et al., 2025). Experimental details are provided in Appendices G and H. Crucially, we maintain a computational cost comparable to standard non-equivariant baselines under matched effective dimension and head-packing assumptions. We detail the specific architectural alignment strategy used to achieve this parity extensively in Appendix D.

QM9. Our results on the QM9 benchmark are summarized in Table 1. The first group of results identifies the most effective Platonic group and model variant for this task, showing a performance gain from incorporating Platonic symmetries over the translation-only ($\emptyset$ group) baseline. Both the Tetrahedron and Octahedron groups achieve strong performance, delivering results on par with state-of-the-art methods like EquiformerV2 (Liao et al., 2024). Our end-to-end $SE(3)$-equivariant model is superior to other baselines: G-Hyena[4] (Moskalev et al., 2025) and FAFormer Huang et al. (2024), a Frame Averaging Puny et al. (2022) method with a Transformer backbone.

Table 1: QM9 Property Prediction MAE (↓).

Platonic Transformer (end-to-end)		
Group	$\boldsymbol{\mu}$	$\boldsymbol{\alpha}$
$\emptyset$	0.028	0.064
Tetrahedron	0.012	0.049
Octahedron	**0.010**	**0.048**
Reference Method	*(re)produced herein [*]	
EquiformerV2 [31]	**0.010**	0.050
FAFormer [24]*	0.122	0.252
G-Hyena [34]*	0.018	0.066
Rapidash [49]	**0.010**	**0.040**

Our use of reference frames relates to the popular class of "frame-based methods," such as Frame Averaging (FA) (Puny et al., 2022), a framework that achieves group equivariance by averaging predictions over reference frames constructed from the input. While FA is theoretically expressive, it can be computationally expensive in practice, as it requires a separate forward pass for each frame element (e.g., 8 passes for full $E(3)$). In Appendix G.4, we demonstrate that the Platonic Transformer offers a highly scalable alternative. By combining just a single PCA-derived frame element with a native axis-aligned roto-reflection group (specifically $C_2 \times C_2 \times C_2$, a subgroup of the octahedral group), our model achieves exact $E(3)$ equivariance at an approximately $8\times$ lower computational cost.As shown in our extended QM9 evaluation, this efficient single-frame variant outperforms standard FA methods such as FAFormer (Huang et al., 2024), highlighting that the geometric representation learned directly is fundamentally more effective compared to external *symmetrization*.

Molecule generation on QM9. We replace the standard Transformer in the DiffusionTransformer (DiT) (Peebles & Xie, 2022) with our Platonic Transformer, which we call DiP, and follow the training

[4]The codebase for G-Hyena was obtained through private correspondence with the authors.

regimen and guidelines from Joshi et al. (2025). We operate at an all-atom resolution, including explicit Hydrogen atoms. We use the same evaluation procedure detailed in Joshi et al. (2025), computing metrics such as molecular validity, molecular stability, and uniqueness.

We present results in Table 2. DiP is on par with jointly-trained and QM9-only variants of ADiT (Joshi et al., 2025) in terms of validity and outperforms other baselines like GeoLDM (Xu et al., 2023), Equivariant Diffusion (EDM) (Hoogeboom et al., 2022), and Symphony Daigavane et al. (2024). These baselines all model explicit Hydrogen atoms as well.

Table 2: Molecule generation results on QM9.

Group	Validity % (↑)	Stability (↑)
DiP-$\emptyset$	93.4	90.7
DiP-Tetrahedron	93.0	93.0
Reference Method	**results taken from Joshi et al. (2025)*	
Jointly trained ADiT [25]*	94.45	–
QM9-only ADiT [25]*	92.19	–
Symphony [16]*	83.50	–
GeoLDM [52]*	93.80	–
EDM [23]*	91.90	–

Our results present a nuanced view of equivariance in generative diffusion. While the trivial (DiP-$\emptyset$) and equivariant (DiP-Tetrahedron) models achieve near-identical validity scores, the equivariant variant yields significantly higher molecular stability. This suggests that standard validity metrics may be overly lenient, whereas stability—accounting for proximity, valency, and charge—offers a more rigorous measure of physical plausibility. Furthermore, the equivariant model exhibited superior training stability at large hidden dimensions and converged faster, consistent with the learning dynamics observed by Vadgama et al. (2025). Notably, the validity gap between our models is minimal, whereas the convolution-based framework in Vadgama et al. (2025) showed a pronounced 11.5% difference between non-equivariant and equivariant variants. This suggests that Transformers may more effectively learn equivariant tasks from unconstrained parameterizations, particularly at scale. Since performance gaps often diminish with increased model size, our use of a large-scale architecture may explain this similarity.

OMol25. We evaluate our model with the best hyperparameters on the large-scale OMol25 dataset to validate scalability and performance. For a fair comparison under a constrained computational budget, we compare the Platonic Transformer with eSEN (Levine et al., 2025), the current state-of-the-art method on OMol25, for 120 hours on a node with 4 NVIDIA 6000 Ada GPUs; note that we use the hyperparameters for eSEN indicated in the original work. The results are presented in Table 3, demonstrating that our model significantly outperforms eSEN under identical conditions for energy prediction and achieves highly competitive performance in force prediction.

Table 3: OMol25 Energy/Force MAE (↓).

(Re)produced results (4 GPUs - 120 hrs)			
Method	**Force**	**Energy**	**E/Atom**
Platonic Transformer (Tetrahedron)	24.25	**74.00**	**2.63**
eSEN [30]	**23.92**	120.0	3.37
From Levine et al. (2025) (estimated 4 GPUs - 475 hrs)			
eSEN [30]	**10.11**	**29.80**	**0.88**
MACE [30]	16.83	54.09	1.55

This performance is noteworthy when contextualized against current literature benchmarks, which we estimate from Levine et al. (2025), utilized a computational budget nearly 4× larger.

Inference Times. A single Platonic Transformer layer runs at the same order of magnitude as a standard Transformer layer (implemented using a single `TransformerEncoderLayer` module in PyTorch) on a batch size of 64 molecules from QM9 on a single H200 GPU, averaged over 10 batches, as shown in Table 4.

Table 4: Inference wall-clock times.

Platonic Transformer	
Group	Avg. Time (ms) (↓)
$\emptyset$	2.87 ± 0.29
Tetrahedron	**2.79** $\pm$ 0.21
Octahedron	2.85 ± 0.25
Reference Method	
Standard Transformer	**2.01** $\pm$ 3.74
G-Hyena [34]	44.06 ± 60.05
TFN [47]	590.45 ± 269.25

5 Conclusion

By combining Rotary Position Embeddings (RoPE) with a new frame-dependent attention mechanism, where attention is computed relative to reference frames from Platonic solid symmetry groups, we integrate a powerful geometric inductive bias while preserving the original computation graph and cost. The Platonic transformer achieves approximate $E(n)$ equivariance without compromising the flexibility and scalability of the standard Transformer architecture.

References

Serge Assaad, Carlton Downey, Rami Al-Rfou', Nigamaa Nayakanti, and Benjamin Sapp. VN-transformer: Rotation-equivariant attention for vector neurons. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=EiX2L4sDPG.

Julia Balla, Siddharth Mishra-Sharma, Carolina Cuesta-Lazaro, Tommi Jaakkola, and Tess Smidt. A cosmic-scale benchmark for symmetry-preserving data processing, 2024. URL https://arxiv.org/abs/2410.20516.

Erik J Bekkers. B-spline cnns on lie groups. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1gBhkBFDH.

Erik J Bekkers, Sharvaree Vadgama, Rob Hesselink, Putri A Van der Linden, and David W. Romero. Fast, expressive \$\mathrm{SE}(n)\$ equivariant networks through weight-sharing in position-orientation space. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dPHLbUqGbr.

Georg Bökman, David Nordström, and Fredrik Kahl. Flopping for flops: Leveraging equivariance for computational efficiency. In *Forty-second International Conference on Machine Learning*, 2025.

Avishek Joey Bose, Tara Akhound-Sadegh, Guillaume Huguet, Kilian Fatras, Jarrid Rector-Brooks, Cheng-Hao Liu, Andrei Cristian Nica, Maksym Korablyov, Michael Bronstein, and Alexander Tong. Se(3)-stochastic flow matching for protein backbone generation, 2024. URL https://arxiv.org/abs/2310.02391.

Johann Brehmer, Pim de Haan, Sönke Behrends, and Taco Cohen. Geometric algebra transformer, 2023. URL https://arxiv.org/abs/2305.18415.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021. URL https://arxiv.org/abs/2104.13478.

Gabriele Cesa, Leon Lang, and Maurice Weiler. A program to build e (n)-equivariant steerable cnns. In *International conference on learning representations*, 2022.

Nan Chen and Soledad Villar. Se (3)-equivariant self-attention via invariant features. In *Machine Learning for Physics NeurIPS Workshop*, 2022.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation, 2023.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pp. 2990–2999. PMLR, 2016.

Taco S. Cohen and Max Welling. Steerable cnns. In *International Conference on Learning Representations (ICLR)*, 2017.

Taco S Cohen, Mario Geiger, and Maurice Weiler. A general theory of equivariant cnns on homogeneous spaces. *Advances in neural information processing systems*, 32, 2019.

Zihang Dai, Zhilin Yang, Yiming Yang, William W. Cohen, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context, 2019.

Ameya Daigavane, Song Kim, Mario Geiger, and Tess Smidt. Symphony: Symmetry-equivariant point-centered spherical harmonics for 3d molecule generation, 2024. URL https://arxiv.org/abs/2311.16199.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

J Thorben Frank, Stefan Chmiela, Klaus-Robert MÃžller, and Oliver T Unke. Euclidean fast attention: Machine learning global atomic representations at linear cost. *arXiv preprint arXiv:2412.08541*, 2024.

Fabian B. Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. Se(3)-transformers: 3d roto-translation equivariant attention networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

Lingshen He, Yuxuan Chen, Zhengyang Shen, Yiming Dong, Yisen Wang, and Zhouchen Lin. Efficient equivariant network. In *Advances in Neural Information Processing Systems*, 2021.

Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d, 2022. URL https://arxiv.org/abs/2203.17003.

Tinglin Huang, Zhenqiao Song, Rex Ying, and Wengong Jin. Protein-nucleic acid complex modeling with frame averaging transformer, 2024. URL https://arxiv.org/abs/2406.09586.

Chaitanya K. Joshi, Xiang Fu, Yi-Lun Liao, Vahe Gharakhanyan, Benjamin Kurt Miller, Anuroop Sriram, and Zachary W. Ulissi. All-atom diffusion transformers: Unified generative modelling of molecules and materials, 2025. URL https://arxiv.org/abs/2503.03965.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russell Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

David M Knigge, David R Wessels, Riccardo Valperga, Samuele Papa, Jan-Jakob Sonke, Efstratios Gavves, and Erik J Bekkers. Space-time continuous pde forecasting using equivariant neural fields. *Advances in Neural Information Processing Systems*, 37:76553–76577, 2024.

Soumyabrata Kundu and Risi Kondor. Steerable transformers for volumetric data. In *Forty-second International Conference on Machine Learning*, 2025.

Hannah Lawrence, Vasco Portilheiro, Yan Zhang, and Sékou-Oumar Kaba. Improving equivariant networks with probabilistic symmetry breaking, 2025. URL https://arxiv.org/abs/2503.21985.

Daniel S. Levine, Muhammed Shuaibi, Evan Walter Clark Spotte-Smith, Michael G. Taylor, Muhammad R. Hasyim, Kyle Michel, Ilyes Batatia, Gábor Csányi, Misko Dzamba, Peter Eastman, Nathan C. Frey, Xiang Fu, Vahe Gharakhanyan, Aditi S. Krishnapriyan, Joshua A. Rackers, Sanjeev Raja, Ammar Rizvi, Andrew S. Rosen, Zachary Ulissi, Santiago Vargas, C. Lawrence Zitnick, Samuel M. Blau, and Brandon M. Wood. The open molecules 2025 (omol25) dataset, evaluations, and models, 2025. URL https://arxiv.org/abs/2505.08762.

Yi-Lun Liao, Brandon Wood, Abhishek Das, and Tess Smidt. Equiformerv2: Improved equivariant transformer for scaling to higher-degree representations, 2024. URL https://arxiv.org/abs/2306.12059.

Shengjie Luo, Tianlang Chen, and Aditi S. Krishnapriyan. Enabling efficient equivariant operations in the fourier basis via gaunt tensor products. In *International Conference on Learning Representations*, 2024. Spotlight.

Dominic Masters, Josef Dean, Kerstin Klaser, Zhiyi Li, Sam Maddrell-Mander, Adam Sanders, Hatem Helal, Deniz Beker, Ladislav Rampášek, and Dominique Beaini. Gps++: An optimised hybrid mpnn/transformer for molecular property prediction. *arXiv preprint arXiv:2212.02229*, 2022.

Artem Moskalev, Mangal Prakash, Junjie Xu, Tianyu Cui, Rui Liao, and Tommaso Mansi. Geometric hyena networks for large-scale equivariant learning, 2025. URL `https://arxiv.org/abs/2505.22560`.

David Nordström, Johan Edstedt, Fredrik Kahl, and Georg Bökman. Stronger vits with octic equivariance. *arXiv preprint arXiv:2505.15441*, 2025.

Elias Nyholm, Oscar Carlsson, Maurice Weiler, and Daniel Persson. Equivariant non-linear maps for neural networks on homogeneous spaces, 2025. URL `https://arxiv.org/abs/2504.20974`.

Marco Pacini, Gabriele Santin, Bruno Lepri, and Shubhendu Trivedi. On universality classes of equivariant networks, 2025. URL `https://arxiv.org/abs/2506.02293`.

William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.

Omri Puny, Matan Atzmon, Heli Ben-Hamu, Ishan Misra, Aditya Grover, Edward J. Smith, and Yaron Lipman. Frame averaging for invariant and equivariant network design, 2022. URL `https://arxiv.org/abs/2110.03336`.

Eric Qu and Aditi S. Krishnapriyan. The importance of being scalable: Improving the speed and accuracy of neural network interatomic potentials across chemical domains, 2024. URL `https://arxiv.org/abs/2410.24169`.

Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1:140022, 2014.

B Srinivasa Reddy and Biswanath N Chatterji. An fft-based technique for translation, rotation, and scale-invariant image registration. *IEEE transactions on image processing*, 5(8):1266–1271, 1996.

David W. Romero and Jean-Baptiste Cordonnier. Group equivariant stand-alone self-attention for vision. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=JkfYjnOEo6M`.

Jean-Pierre Serre et al. *Linear representations of finite groups*, volume 42. Springer, 1977.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Philipp Thölke and Gianni De Fabritiis. Equivariant transformers for neural network based molecular potentials. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=zNHzqZ9wrRB`.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds, 2018. URL `https://arxiv.org/abs/1802.08219`.

Richard Tran, Janice Lan, Muhammed Shuaibi, Brandon M Wood, Siddharth Goyal, Abhishek Das, Javier Heras-Domingo, Adeesh Kolluru, Ammar Rizvi, Nima Shoghi, et al. The open catalyst 2022 (oc22) dataset and challenges for oxide electrocatalysts. *ACS Catalysis*, 13(5):3066–3084, 2023.

Sharvaree Vadgama, Mohammad Mohaiminul Islam, Domas Buracas, Christian Shewmake, Artem Moskalev, and Erik Bekkers. Probing equivariance and symmetry breaking in convolutional networks, 2025. URL `https://arxiv.org/abs/2501.01999`.

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.

David R Wessels, David M Knigge, Samuele Papa, Riccardo Valperga, Sharvaree Vadgama, Efstratios Gavves, and Erik J Bekkers. Grounding continuous representations in geometry: Equivariant neural fields. *arXiv preprint arXiv:2406.05753*, 2024.

Minkai Xu, Alexander Powers, Ron Dror, Stefano Ermon, and Jure Leskovec. Geometric latent diffusion models for 3d molecule generation, 2023. URL `https://arxiv.org/abs/2305.01140`.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16259–16268, 2021.

Maksim Zhdanov, David Ruhe, Maurice Weiler, Ana Lucic, Johannes Brandstetter, and Patrick Forré. Clifford-steerable convolutional neural networks. In *International Conference on Machine Learning (ICML)*, 2024.

Appendix

A Rotary Position Embeddings from a Group Theoretical Perspective

A fundamental challenge in geometric deep learning is creating position representations that respect underlying symmetries. For data in $\mathbb{R}^d$, our goal is to define a high-dimensional position embedding, $\mathbf{E} : \mathbb{R}^d \to \mathbb{R}^{d'}$, that is equivariant to translations. This requires that for any translation vector $\mathbf{p}$, the embedding transforms predictably: $\mathbf{E}(\mathbf{p}_0 + \mathbf{p}) = \boldsymbol{\rho}(\mathbf{p})\mathbf{E}(\mathbf{p}_0)$, where $\boldsymbol{\rho}(\mathbf{p})$ is a linear transformation. Group representation theory provides the formal tools to construct such embeddings.

A.1 The Theoretical Toolkit

To proceed, we first define the essential concepts required for our construction.

Definition 1 (Representation)**.** *A linear representation of a group $\mathcal{G}$ on a vector space V is a group homomorphism $\rho : \mathcal{G} \to GL(V)$, where $GL(V)$ is the general linear group of invertible linear transformations on V.*

To ensure that the positional encoding does not arbitrarily amplify or diminish feature magnitudes, which would destabilize learning, we require the representations to be length-preserving. This leads to the concept of a unitary representation.

Definition 2 (Unitary Representation)**.** *A representation ρ is **unitary** if it maps group elements to unitary operators, i.e., $\rho : \mathcal{G} \to U(V)$. For real-valued representations, this corresponds to orthogonality, $\rho(g)^{-1} = \rho(g)^\top$.*

Just as a complex signal can be decomposed into pure frequencies, a general representation can be broken down into fundamental building blocks known as irreducible representations (irreps).

Definition 3 (Irreducible Representation)**.** *An **irreducible representation** (irrep) is a representation acting on a vector space V that has no non-trivial invariant subspaces.*

A.2 Constructing the RoPE Operator

With these formal tools, we can now build the RoPE operator. The irreps of the translation group $(\mathbb{R}^d, +)$ are indexed by a frequency vector $\boldsymbol{\omega} \in \mathbb{R}^d$ and are given by one-dimensional, unitary representations:

$$\rho_{\boldsymbol{\omega}}(\mathbf{p}) = e^{i\boldsymbol{\omega}^\top \mathbf{p}}. \tag{2}$$

This exponential form is the unique continuous solution to the group's homomorphism property, $\rho(\mathbf{p}_1 + \mathbf{p}_2) = \rho(\mathbf{p}_1)\rho(\mathbf{p}_2)$, where the imaginary exponent ensures unitarity.

However, neural networks typically operate on real numbers. We can obtain a real-valued irrep by combining pairs of conjugate frequencies, $\boldsymbol{\omega}_k$ and $-\boldsymbol{\omega}_k$. This yields a 2D irreducible representation that takes the familiar form of a rotation matrix:

$$\rho_{\boldsymbol{\omega}_k}(\mathbf{p}) = \begin{pmatrix} \cos(\boldsymbol{\omega}_k^\top \mathbf{p}) & -\sin(\boldsymbol{\omega}_k^\top \mathbf{p}) \\ \sin(\boldsymbol{\omega}_k^\top \mathbf{p}) & \cos(\boldsymbol{\omega}_k^\top \mathbf{p}) \end{pmatrix}. \tag{3}$$

To create a high-dimensional embedding, we simply select a set of frequencies $\Omega = \{\boldsymbol{\omega}_k\}_{k=1}^{d'/2}$ and stack these 2D rotation blocks along the diagonal of a larger matrix. This results in a single, block-diagonal transformation that correctly and equivariantly updates the entire embedding for a given translation $\mathbf{p}$:

$$\boldsymbol{\rho}_\Omega(\mathbf{p}) = \text{diag}(\rho^{\boldsymbol{\omega}_1}(\mathbf{p}), \ldots, \rho_{\boldsymbol{\omega}_{d'/2}}(\mathbf{p})). \tag{4}$$

This is the core mechanism behind Rotary Position Embeddings. Its structure guarantees both equivariance and computational efficiency, as each 2D component can be rotated independently.

Definition 4 (Rotary Position Embedding (RoPE) Operator)**.** *The RoPE operator $\boldsymbol{\rho}_\Omega(\mathbf{p})$ for a position $\mathbf{p} \in \mathbb{R}^d$ is the block-diagonal rotation matrix defined above (Equation 4), constructed from a set of frequencies Ω. The application of RoPE to a feature vector $\mathbf{f} \in \mathbb{R}^{d'}$ is defined as the matrix-vector product: $\boldsymbol{\rho}_\Omega(\mathbf{p})\mathbf{f}$. For this operation to be well-defined, the feature dimension d' must be even.*

A.3 Translation Invariance in Attention

While the RoPE operator provides an *equivariant* transformation for feature vectors, its crucial benefit within the Transformer architecture is that it makes the attention score *invariant* to global translations. This property ensures that the attention mechanism only considers the relative positions of tokens, which is the inductive bias we seek. We formalize this key result below.

Proposition 2 (Translation Invariance of the RoPE Attention Score)**.** *The attention score computed using RoPE, $\mathrm{attn}(\mathbf{q}, \mathbf{k}, \Delta\mathbf{p}) = \mathbf{q}^\top \boldsymbol{\rho}_\Omega(\Delta\mathbf{p})\mathbf{k}$, where $\Delta\mathbf{p} = \mathbf{p}_j - \mathbf{p}_i$, is invariant to a global translation of the coordinate system.*

Proof. Let the positions $\mathbf{p}_i$ and $\mathbf{p}_j$ be translated by an arbitrary vector $\mathbf{t} \in \mathbb{R}^d$, resulting in new positions $\mathbf{p}'_i = \mathbf{p}_i + \mathbf{t}$ and $\mathbf{p}'_j = \mathbf{p}_j + \mathbf{t}$. The new relative displacement vector, $\Delta\mathbf{p}'$, is:

$$\Delta\mathbf{p}' = \mathbf{p}'_j - \mathbf{p}'_i = (\mathbf{p}_j + \mathbf{t}) - (\mathbf{p}_i + \mathbf{t}) = \mathbf{p}_j - \mathbf{p}_i = \Delta\mathbf{p}. \tag{5}$$

Since the relative displacement vector is unchanged by the global translation, the RoPE operator applied to it also remains unchanged: $\boldsymbol{\rho}_\Omega(\Delta\mathbf{p}') = \boldsymbol{\rho}_\Omega(\Delta\mathbf{p})$. Consequently, the attention score, which depends only on the content vectors and this operator, is invariant to the translation:

$$\mathbf{q}^\top \boldsymbol{\rho}_\Omega(\Delta\mathbf{p}')\mathbf{k} = \mathbf{q}^\top \boldsymbol{\rho}_\Omega(\Delta\mathbf{p})\mathbf{k}. \tag{6}$$

This formally demonstrates that RoPE imparts translation invariance to the attention mechanism. □

A.4 A Fourier Perspective

The principle of constructing equivariant functions from irreducible representations is deeply connected to Fourier analysis. The Fourier transform provides a way to decompose any function on a group into a weighted sum (or integral) over its irreps. For the translation group on $\mathbb{R}^d$, these irreps are precisely the complex exponentials we used as our building blocks. Therefore, RoPE can be understood as a practical application of Fourier theory, using a discrete basis of Fourier modes (the chosen frequencies Ω) to represent the positional signal.

Definition 5 (Fourier Transform on $\mathbb{R}^d$)**.** *The forward Fourier transform $\mathcal{F} : L^2(\mathbb{R}^d) \to L^2(\mathbb{R}^d)$ maps a function f to its frequency-space representation $\hat{f}$. The coefficient for a frequency $\boldsymbol{\omega}$ is the projection of f onto the corresponding irrep $\rho_{\boldsymbol{\omega}}$:*

$$\hat{f}(\boldsymbol{\omega}) = \mathcal{F}\{f\}(\boldsymbol{\omega}) = \int_{\mathbb{R}^d} f(\mathbf{p})\overline{\rho_{\boldsymbol{\omega}}(\mathbf{p})}\, d\mathbf{p} = \int_{\mathbb{R}^d} f(\mathbf{p}) e^{-i\boldsymbol{\omega}^\top \mathbf{p}}\, d\mathbf{p}. \tag{7}$$

The inverse transform reconstructs the function by integrating over all irreps:

$$f(\mathbf{p}) = \mathcal{F}^{-1}\{\hat{f}\}(\mathbf{p}) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{f}(\boldsymbol{\omega})\rho_{\boldsymbol{\omega}}(\mathbf{p})\, d\boldsymbol{\omega} = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{f}(\boldsymbol{\omega}) e^{i\boldsymbol{\omega}^\top \mathbf{p}}\, d\boldsymbol{\omega}. \tag{8}$$

B Equivariance Properties of Platonic Transformers

We formally establish the equivariance of our proposed architecture. We consider a point cloud $\{\mathbf{p}_i, \mathbf{v}_i, s_i\}_{i=1}^N$ consisting of positions, vectors, and scalars. A global rotation $R \in \mathcal{G}$ acts on these inputs as $\mathbf{p}_i \mapsto R\mathbf{p}_i$, $\mathbf{v}_i \mapsto R\mathbf{v}_i$, and $s_i \mapsto s_i$.

Equivariant Feature Lifting. Input features are first lifted to functions on the group $\mathcal{G}$. The lifting operator, Lift, maps the input point cloud to a set of feature maps $\{\mathbf{f}_i : \mathcal{G} \to \mathbb{R}^C\}_{i=1}^N$. Scalar components are copied to each frame, while vector components (from $\mathbf{p}_i, \mathbf{v}_i$) are lifted by projecting them onto each reference frame. This projection means expressing the vector's coordinates in the local basis of a given frame $\tilde{R} \in \mathcal{G}$, which is achieved by the transformation $\tilde{R}^{-1}\mathbf{v}$. This lifting procedure is equivariant by construction: a global rotation R of the input point cloud results in the lifted feature maps transforming via the left regular representation, L_R. That is:

$$(\mathrm{Lift}(R \cdot \mathrm{cloud}))_i(\tilde{R}) = (\mathrm{Lift}(\mathrm{cloud}))_i(R^{-1}\tilde{R}) \triangleq (L_R \mathbf{f}_i)(\tilde{R}). \tag{9}$$

Equivariant Linear Layers. All linear layers Φ in our network are implemented as point-wise group convolutions, as shown in Eq. 10. These layers are equivariant to the action of the group by construction (Cohen et al., 2019, Thm. 3.1), satisfying $\Phi(L_R \mathbf{f}_i) = L_R(\Phi(\mathbf{f}_i))$.

This leads to our main proposition regarding the attention mechanism.

Proposition 3 (Equivariant Attention). *Let the queries Q_i, keys K_i, and values V_i be equivariant feature maps produced by the equivariant linear layers. The RoPE-enhanced attention mechanism detailed in Section 3, which computes outputs $\mathbf{y}_i$, is an equivariant operation. That is, if the inputs transform as $\mathbf{f}_i \mapsto L_R \mathbf{f}_i$, the outputs transform as $\mathbf{y}_i \mapsto L_R \mathbf{y}_i$.*

Proof. Let the inputs to the attention layer (Q_i, K_i, V_i) transform under a global rotation R as $Q'_i = L_R Q_i$, $K'_i = L_R K_i$, and $V'_i = L_R V_i$. We analyze the transformation of each component of the attention calculation.

The score function $s_{ij}(\tilde{R})$, which depends on $Q_i(\tilde{R})$ and $K_j(\tilde{R})$ (and potentially RoPE terms derived from lifted positions), will transform as:

$$s'_{ij}(\tilde{R}) = \mathrm{score}(Q'_i(\tilde{R}), K'_i(\tilde{R}), \dots) = \mathrm{score}(Q_i(R^{-1}\tilde{R}), K_j(R^{-1}\tilde{R}), \dots) = s_{ij}(R^{-1}\tilde{R}).$$

This means the score function itself is equivariant, $s'_{ij} = L_R s_{ij}$. Since the softmax operator is applied point-wise for each frame $\tilde{R}$ over the index j, the attention weights also transform equivariantly:

$$\mathrm{attn}'_{ij}(\tilde{R}) = \underset{j}{\mathrm{softmax}}(s'_{ij}(\tilde{R})) = \underset{j}{\mathrm{softmax}}(s_{ij}(R^{-1}\tilde{R})) = \mathrm{attn}_{ij}(R^{-1}\tilde{R}).$$

Finally, the output feature map $\mathbf{y}_i$ transforms as:

$$\begin{aligned}\mathbf{y}'_i(\tilde{R}) &= \sum_{j=1}^{N} \mathrm{attn}'_{ij}(\tilde{R}) V'_j(\tilde{R}) \\ &= \sum_{j=1}^{N} \mathrm{attn}_{ij}(R^{-1}\tilde{R}) V_j(R^{-1}\tilde{R}) = \mathbf{y}_i(R^{-1}\tilde{R}).\end{aligned}$$

Thus, the output transforms as $\mathbf{y}'_i = L_R \mathbf{y}_i$, proving the attention mechanism is equivariant. □

C Architectural Details

In this section, we provide additional details about the architecture of the Platonic Transformer and the various model configurations used in our experiments. Our framework is designed to be equivariant to roto-translation groups, primarily finite subgroups of $SE(n)$ and, through specific configurations, the Euclidean group $E(n)$.

We denote the core embedding dimension per group element as d_{hidden}. Since our features are functions on a group $\mathcal{G}$ of order $|\mathcal{G}|$, the total feature dimension of a layer is $d_{\text{model}} = |\mathcal{G}| \times d_{\text{hidden}}$. The specific group is determined by the `solid_name` parameter.

For the initial feature processing, input scalars and vectors are first embedded and then lifted into a group-equivariant feature space using an initial lifting operation. This creates a tensor where the channel dimension is expanded by a factor of $|\mathcal{G}|$. An initial group-equivariant linear layer then projects these lifted features to the model's working dimension, d_{model}. Optionally, an equivariant Absolute Positional Encoding (APE), parameterized by `ape_sigma`, can be added at this stage.

The main body of the network consists of a stack of equivariant transformer blocks. Each block contains two main sub-modules: a group-equivariant interaction layer and a feed-forward network (FFN), connected with residual connections. Normalization is applied either before each sub-module or after.

For the group-equivariant interaction layer, we denote the number of attention heads per group element as n_{head}. The total number of effective parallel heads is therefore $|\mathcal{G}| \times n_{\text{head}}$. The dimension of each head, d_{head}, is calculated as $d_{\text{hidden}}/n_{\text{head}}$. The input features are first projected to query, key, and value representations using group-equivariant linear layers. To encode relative spatial information, group-equivariant Rotary Position Embeddings, parameterized by `rope_sigma` and `learned_freqs`, are applied to the query and key vectors. The interaction can then be performed either as a full softmax-based attention mechanism or as a linear-time dynamic group convolution via the `attention` flag. For the Feed Forward Networks (FFNs), we denote the hidden feature dimension as $d_{\text{ffn}} = d_{\text{model}} \times f_{\text{factor}}$. The FFN consists of two group-equivariant linear layers with a GELU activation function in between.

For the final output, two separate readout heads project the features to the desired scalar and vector output dimensions. For graph-level tasks, a pooling operation performs a mean aggregation over the node and group dimensions to produce a final invariant prediction. For node-level tasks, an averaging operation over the group axis projects the features back to standard invariant scalar and equivariant vector representations. Following standard Transformer practices, we apply dropout to the attention weights and FFN activations, and stochastic depth to the outputs of the equivariant transformer blocks.

Particular values for all the important hyperparameters used for the experiments are in the Table 5.

Table 5: Hyperparameter for all datasets

Hyperparameter	**QM9**	**OMol25**
Architecture		
`hidden_dim`	1152	1152
`layers`	14	14
`num_heads`	72	72
Positional encoding		
`freq_sigma`	0.5	0.5
`ape_sigma`	0.5	None
`learned_freqs`	True	True
Attention / readout		
`dropout`	0.0	0.0
`drop_path_rate`	0.0	0.0
`mean_aggregation`	False	False
Training		
`lr`	5e-4	2e-4
`batch_size`	96	64
`epochs`	1000	22
`warmup`	10	5
`weight_decay`	1e-8	1e-6
`lambda_F`	-	12.0
`cosine_scheduler`	True	True
`precision`	32	32
`gpus`	1	4

C.1 Equivariant Linear Layers and Fixed Computation Graph

All linear transformations, including the query, key, and value projections ($\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$), and any MLP blocks, must be equivariant. Since our feature maps have a group axis, any linear layer Φ must respect the group action on this axis. This means that for any group element $R \in \mathcal{G}$, the transformation must satisfy the equivariance constraint: $\Phi(L_R\mathbf{f}_i) = L_R(\Phi(\mathbf{f}_i))$, where L_R is the action of rotating the reference frames, i.e., $L_R\mathbf{f}_i(\tilde{R}) = \mathbf{f}_i(R^{-1}\tilde{R})$.

This constraint is satisfied if and only if we replace the standard matrix-vector multiplication with a *group convolution* (Cohen et al., 2019, Thm. 3.1). An equivariant linear layer acts on the feature vector $\mathbf{f}_i(R)$ at each orientation $R \in \mathcal{G}$ as follows:

$$(\Phi(\mathbf{f}_i))(R) = \sum_{\tilde{R} \in \mathcal{G}} \mathbf{W}_{\text{group}}(R^{-1}\tilde{R})\, \mathbf{f}_i(\tilde{R})\,. \tag{10}$$

Here, $\mathbf{W}_{\text{group}} : \mathcal{G} \to \mathbb{R}^{C' \times C}$ is a learnable kernel defined on the group. This equation imposes a specific weight-sharing structure where the interaction between input and output frames depends only on their *relative pose*, $R^{-1}\tilde{R}$. The layer is thus constrained to learn patterns from the geometric arrangement of features, rather than their absolute orientation.

This group convolution is a principled *weight-sharing scheme* that preserves the computation graph of a standard linear layer. While Eq. 10 is written as a sum, it is equivalent to a single matrix-vector multiplication with a large, structured weight matrix. This matrix is constructed by sampling a much smaller learnable kernel, $\mathbf{W}_{\text{group}}$, which reduces the parameter count from the $(|\mathcal{G}| \cdot C') \times (|\mathcal{G}| \cdot C)$ of an unconstrained layer to just $|\mathcal{G}| \cdot C' \cdot C$—a reduction by a factor of $|\mathcal{G}|$.

Crucially, by choosing the number of channels C such that the effective feature dimension $C \times |\mathcal{G}|$ is held constant, the overall matrix dimensions are identical regardless of the group size. The trivial group $\mathcal{G} = \{e\}$ illustrates the base case, where the operation collapses to a standard linear layer with a weight matrix $\mathbf{W}_{\text{group}}(e)$ of size $C' \times C$. The geometric inductive bias is therefore not introduced by adding new, complex modules, but by imposing a structure on the weights of existing ones.[5]

C.2 Computing group products via Cayley tables

We restrict our 3D frames to the finite *rotational* symmetry groups ($\mathcal{G} \subset SO(3)$) of the Platonic solids: the *tetrahedral* (12 rotations), *octahedral* (24 rotations), and *icosahedral* (60 rotations) groups. While these solids have larger full symmetry groups that include reflections (e.g., 24 total symmetries for the tetrahedron), we focus on the purely rotational subgroups for a more tractable structure.

A rotation of this feature map by an element R_i, defined by the action $(L_{R_i} f)(R_j) = f(R_i^{-1} R_j)$, simplifies to a permutation of the feature tensor's entries. With the Cayley table, the new feature at position j is simply copied from the old feature at position $k = \text{Cayley}[\text{inverse}[i], j]$.

D Hyperparameter Tuning and Model Selection Strategy

This section outlines the full procedure used to configure and train our models.

Baseline Optimization. To establish a fair point of comparison, we first optimized the general training protocol using only the translation-only equivariant ($T(n)$) models. This initial phase involved tuning the optimizer, learning rate schedule, weight decay, and data augmentations to ensure the baseline models were as competitive as possible. This fixed protocol was then used for all subsequent experiments.

Hyperparameter Sweep for Model Selection. With the training protocol fixed, we performed an extensive hyperparameter sweep for both $SE(n)$ and $T(n)$ model classes. This sweep was designed to find the optimal architectural parameters while maintaining an equal computational budget between model families. The parameters and their swept values are summarized in Table 6.

The hyperparameter sweep was conducted across multiple layers and three random seeds for a *moderate number of epochs* to efficiently explore the configuration space. It should be noted that some configurations are not applicable for the Octahedral group, as its 24 symmetry elements require a minimum of 24 total effective heads (i.e., at least one head per group element). After identifying the best-performing hyperparameters for both the $SE(n)$ and $T(n)$ model families from this sweep, we proceeded to a final, full-length training run. These selected models were trained for a *large number*

[5]This structure can even give computational benefits, by implementing the linear layers in the Fourier domain of $\mathcal{G}$ (Bökman et al., 2025). In Appendix I, we find that at marginally higher channel counts than used in this paper, a Fourier implementation leads to greatly improved training throughput, indicating that this is a promising direction for future research.

Table 6: Hyperparameter Sweep Configurations.

Parameter	**Configuration**	**Values**
Hidden Dim	-	$[384, 576, 768, 1152]$
Number of Heads	$T(n)$ **model** (HS=16)	$[24, 36, 48, 72]$
	$T(n)$ **model** (HS=32)	$[12, 18, 24, 36]$
	$SE(n)$ **model** (HS=16)	$[2, 3, 4, 6]$ heads per group element
	$SE(n)$ **model** (HS=32)	$[1, 2, 3]$ heads per group element
Rope Sigma (σ_{rope})	RoPE frequency scaling	$[0.5 - 2.0]$
Attention	Full Attention / Linear Conv	[True, False]
Solid Group ($\mathcal{G}$)	Symmetry group	[Octahedron, Tetrahedron, C_{2-8}, D_{4-8}]
Lambda F (λ_F)	OMol Force loss weight	$[1.0 - 25.0]$
Batch Size	Samples per batch	$[64 - 512]$
Weight Decay		$[1e^{-3} - 1e^{-7}]$

of epochs to ensure convergence again with fixed compute budget, producing the final results reported in the main paper.

E Inductive Bias of Platonic Transformers

This section examines the Platonic Transformer's structural inductive biases. We highlight its interpretation as a dynamic group convolution and its fully equivariant attention, contrasting these with approaches based on invariant attention.

E.1 Platonic Transformer as Dynamic Group Convolution

The use of RoPE in a linear attention setting establishes a deep connection to convolution. Specifically, the mechanism implements an adaptive convolution where the kernel is synthesized on-the-fly. This dynamic kernel is expressed as an expansion in a sparse Fourier basis, defined by the RoPE frequencies, and the coefficients for this basis expansion are provided by the query vectors. This makes the convolution content-aware. We formalize this as follows (proof in Appendix F.1).

Proposition 4 (Linear RoPE Attention as Dynamic Convolution)**.** *Consider a standard linear attention layer using RoPE with constant key vectors ($\mathbf{k}_j = \mathbf{1}$). The layer's output $\mathbf{y}_i$ is mathematically equivalent to a dynamic convolution:*

$$\mathbf{y}_i = \sum_{j=1}^{N} \phi_{\mathbf{q}_i}(\mathbf{p}_j - \mathbf{p}_i)\mathbf{v}_j \, , \tag{11}$$

where the dynamic kernel $\phi_{\mathbf{q}_i}$ is given by the inverse sparse Fourier transform:

$$\phi_{\mathbf{q}_i}(\Delta\mathbf{p}) = \sum_{k=1}^{d/2} \left[a_k(\mathbf{q}_i)\cos(\boldsymbol{\omega}_k^\top \Delta\mathbf{p}) + b_k(\mathbf{q}_i)\sin(\boldsymbol{\omega}_k^\top \Delta\mathbf{p})\right] \, . \tag{12}$$

The Fourier coefficients are given by the linear projections $a_k(\mathbf{q}_i) = q_{i,2k-1} + q_{i,2k}$ and $b_k(\mathbf{q}_i) = q_{i,2k} - q_{i,2k-1}$, where $q_{i,m}$ is the m-th element of the query vector $\mathbf{q}_i$.

Remark 1 (Purely Geometric vs. Mixed Kernels)**.** *This result recasts the query's role: rather than simply probing for content, $\mathbf{q}_i$ enables the parameters to construct a unique geometric filter. The formulation of the key vector is a design choice. The constant-key formulation ($\mathbf{k}_j = \mathbf{1}$) forces the model to learn purely geometric, content-adaptive convolution operators. In contrast, a learned key ($\mathbf{k}_j = \mathbf{W}_K\mathbf{f}_j$) results in a mixed kernel whose coefficients depend on both query and key features, and thus entangles geometry and signal.*

Our choice of the purely geometric formulation was motivated by a key experimental observation: training with a mixed kernel from learned keys was highly unstable on the QM9 and OMol25 datasets. We hypothesize this instability arises because these tasks require learning universal *physical principles* that are purely functions of geometry. A mixed kernel *entangles* these universal principles with instance-specific features, creating an unstable optimization problem with *conflicting gradients* as the model attempts to learn a general physical law while simultaneously fitting unique local chemical environments. Fixing the keys thus acts as a crucial regularizer for physical systems by forcing the model to prioritize the disentangled geometric principles, and thus stabilizing the training process. The convolution perspective further leads to a key practical advantage.

Corollary 1 (Linear-Time Complexity)**.** *The dynamic convolution in Proposition 4 can be computed in $O(N)$ time, where N is the number of tokens or points in the point cloud. This offers a scalable alternative to standard attention, which has a quadratic complexity of $O(N^2)$.*

Within our Platonic Transformer, this entire mechanism is lifted to operate over the reference frames defined by a group $\mathcal{G}$. Consequently, the operator becomes an adaptive *group convolution* (proof in Appendix F.3), where the kernel is steered by the group elements/reference frames.

E.2 Invariant vs. Equivariant Attention Score

Our approach implements a fully *equivariant attention* mechanism, where the attention pattern is orientation-dependent. This contrasts with methods using an *invariant attention* score, which applies the same pattern from all orientations (Fuchs et al., 2020; Chen & Villar, 2022; Assaad et al., 2023; Frank et al., 2024; Knigge et al., 2024; Kundu & Kondor, 2025; Nordström et al., 2025).

For multi-head attention with H heads, let $\mathbf{q}_i(R,h)$, $\mathbf{k}_j(R,h)$, and $\mathbf{v}_j(R,h)$ denote the projected query, key, and value vectors for head h from the perspective of frame R. In our equivariant approach, the raw scores $s_{ij}(R,h)$ are passed directly to the softmax. This allows the model to learn orientation-dependent attention patterns, making it a more expressive formulation that retains the rich geometric information in the features. The output is an equivariant feature map on the group:

$$\mathbf{y}_i(R,h) = \sum_{j=1}^{N} \operatorname{softmax}_j(\underbrace{s_{ij}(R,h)}_{R-\text{dependent}})\mathbf{v}_j(R,h), \quad s_{ij}(R,h) = \mathbf{q}_i(R,h)^\top \boldsymbol{\rho}_{\Omega_h}((\mathbf{p}_j - \mathbf{p}_i)(R))\mathbf{k}_j(R,h)\,. \tag{13}$$

In practice, this is efficiently implemented by treating the $|\mathcal{G}|$ perspectives as an independent set of attention heads. Tensors are reshaped so that the group and head dimensions are merged, e.g., to a shape of $[B, N, |\mathcal{G}| \cdot H, C_h]$, before the dot product calculation.

In an invariant attention score, a single attention pattern is created by pooling the raw scores over the group axis before the softmax, akin to the *symmetrization* in the RoPE-based approach of Frank et al. (2024). These invariant attention scores are then applied to the original equivariant value vectors. The resulting output is still equivariant, but it is derived from an *orientation-agnostic attention pattern*:

$$\mathbf{y}_i(R,h) = \sum_{j=1}^{N} \operatorname{softmax}_j(\underbrace{s_{ij}^{\text{inv}}(h)}_{R-\text{agnostic}})\mathbf{v}_j(R,h)\,, \quad \text{where} \quad s_{ij}^{\text{inv}}(h) = \sum_{R\in\mathcal{G}} s_{ij}(R,h)\,. \tag{14}$$

Although simpler, this formulation sacrifices the model's ability to attend to features in an orientation-dependent manner. Implementing Eq. 14 can be done by reshaping tensors so that the group and channel dimensions are merged, to shape $[B, N, H, |\mathcal{G}| \cdot C_h]$, as then the dot-product in s_{ij} and the sum in s_{ij}^{inv} are simultaneously computed when taking the dot-product between queries and keys. For a fully invariant output, one could additionally average the value vectors $\mathbf{v}_j(R,h)$ over the group to further collapse the geometric representation.

E.3 Equivalent Attention via RoPE Base Frequency Steering

We achieve full equivariance to Euclidean transformations by making the RoPE operator dependent on a local reference frame R by projecting positions $\mathbf{p}_i$ on R to obtain positions $\mathbf{p}_i(R) := R^{-1}\mathbf{p}_i$. The attention scores $s_{ij}(R)$ for a query $\mathbf{q}(R)_i$ and key $\mathbf{k}(R)_j$ are computed as:

$$s_{ij}(R) = \mathbf{q}_i(R)^\top \rho_\Omega((\mathbf{p}_j - \mathbf{p}_i)(R))\mathbf{k}_j(R), \tag{15}$$

An equivalent approach is to steer the set of base RoPE frequencies Ω for each frame, creating a frame-specific set $\Omega_R = \{R\boldsymbol{\omega}_k \mid \boldsymbol{\omega}_k \in \Omega\}$ (Reddy & Chatterji, 1996). The attention scores are then computed as:

$$\hat{s}_{ij}(R) = \mathbf{q}_i(R)^\top \boldsymbol{\rho}_{\Omega_R}(\mathbf{p}_j - \mathbf{p}_i)\mathbf{k}_j(R). \tag{16}$$

Proof. For $s_{ij}(R)$ and $\hat{s}_{ij}(R)$ to be equivalent, we require that $\rho_\Omega((\mathbf{p}_j - \mathbf{p}_i)(R)) = \boldsymbol{\rho}_{\Omega_R}(\mathbf{p}_j - \mathbf{p}_i)$. For this, we need to show that $\boldsymbol{\omega}_k^\top \Delta\mathbf{p}(R) = (R\boldsymbol{\omega})_k^\top \Delta\mathbf{p}$, where $\Delta\mathbf{p} = \mathbf{p}_j - \mathbf{p}_i$. Let $\mathbf{R}$ be the orthogonal matrix corresponding to R. Then we have:

$$\begin{aligned}
\boldsymbol{\omega}_k^\top \Delta\mathbf{p}(R) &= \boldsymbol{\omega}_k^\top R^{-1}\Delta\mathbf{p} && (17)\\
&= \boldsymbol{\omega}_k^\top \mathbf{R}^\top \Delta\mathbf{p} && (18)\\
&= (\mathbf{R}\boldsymbol{\omega}_k)^\top \Delta\mathbf{p} && (19)\\
&= (R\boldsymbol{\omega}_k)^\top \Delta\mathbf{p} && (20)
\end{aligned}$$

Thus, projecting global positions or steering the base frequencies are equivalent. □

By projecting the global positions, the RoPE attention mechanism remains identical to its traditional formulation. Steering the base frequencies, however, is often more computationally efficient, since the number of base frequencies is typically much smaller than the input sequence length.

F PROOFS

F.1 PROOF OF PROPOSITION 4

We seek to show that the unnormalized attention score, which defines the kernel $\phi_{\mathbf{q}_i}(\Delta\mathbf{p})$, takes the form of a sparse Fourier series whose coefficients are linear projections of the query $\mathbf{q}_i$.

Let the relative position be $\Delta\mathbf{p} = \mathbf{p}_j - \mathbf{p}_i$. With a constant key vector $\mathbf{k}_j = \mathbf{1}$, the kernel is defined by the attention score:

$$\phi_{\mathbf{q}_i}(\Delta\mathbf{p}) = (\boldsymbol{\rho}(\mathbf{p}_i)\mathbf{q}_i)^\top(\boldsymbol{\rho}(\mathbf{p}_j)\mathbf{1})$$

Using the properties of the RoPE operator $\boldsymbol{\rho}$, this simplifies to:

$$\phi_{\mathbf{q}_i}(\Delta\mathbf{p}) = \mathbf{q}_i^\top \boldsymbol{\rho}(\mathbf{p}_i)^\top \boldsymbol{\rho}(\mathbf{p}_j)\mathbf{1} = \mathbf{q}_i^\top \boldsymbol{\rho}(\Delta\mathbf{p})\mathbf{1}$$

The RoPE matrix $\boldsymbol{\rho}(\Delta\mathbf{p})$ is block-diagonal, consisting of $d/2$ independent 2D rotation blocks. We can therefore analyze the contribution from a single block k and sum the results. Let $\theta_k = \boldsymbol{\omega}_k^\top \Delta\mathbf{p}$. The contribution from block k is:

$$\phi_k = \begin{pmatrix} q_{2k-1} & q_{2k} \end{pmatrix} \begin{pmatrix} \cos(\theta_k) & -\sin(\theta_k) \\ \sin(\theta_k) & \cos(\theta_k) \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Performing the matrix-vector multiplications, we get:

$$\begin{aligned}
\phi_k &= \begin{pmatrix} q_{2k-1} & q_{2k} \end{pmatrix} \begin{pmatrix} \cos(\theta_k) - \sin(\theta_k) \\ \sin(\theta_k) + \cos(\theta_k) \end{pmatrix} \\
&= q_{2k-1}(\cos(\theta_k) - \sin(\theta_k)) + q_{2k}(\sin(\theta_k) + \cos(\theta_k))
\end{aligned}$$

Grouping terms by $\cos(\theta_k)$ and $\sin(\theta_k)$ reveals the linear projections:

$$\phi_k = \underbrace{(q_{2k-1} + q_{2k})}_{a_k(\mathbf{q}_i)} \cos(\theta_k) + \underbrace{(q_{2k} - q_{2k-1})}_{b_k(\mathbf{q}_i)} \sin(\theta_k)$$

The Fourier coefficients $a_k(\mathbf{q}_i)$ and $b_k(\mathbf{q}_i)$ are thus simple linear combinations of the query vector's elements. Summing over all $k = 1, \ldots, d/2$ yields the complete kernel $\phi_{\mathbf{q}_i}(\Delta\mathbf{p})$, which has the exact form stated in the proposition.

F.2 Proof of Corollary 1

The linear-time complexity is achieved by expressing the operation in matrix form and re-ordering the computation. Let $\mathbf{q}'_i = \boldsymbol{\rho}_\Omega(\mathbf{p}_i)\mathbf{q}_i$ and $\mathbf{k}'_j = \boldsymbol{\rho}_\Omega(\mathbf{p}_j)\mathbf{1}$. Let $\mathbf{Q}' \in \mathbb{R}^{N\times d'}$ be the matrix with rows $(\mathbf{q}'_i)^\top$, $\mathbf{K}' \in \mathbb{R}^{N\times d'}$ be the matrix with rows $(\mathbf{k}'_j)^\top$, and $\mathbf{V} \in \mathbb{R}^{N\times d_v}$ be the matrix of value vectors. The output matrix $\mathbf{Y} \in \mathbb{R}^{N\times d_v}$ is given by:

$$\mathbf{Y} = (\mathbf{Q}'(\mathbf{K}')^\top)\mathbf{V}.$$

By the associativity of matrix multiplication, this can be computed as $\mathbf{Y} = \mathbf{Q}'((\mathbf{K}')^\top\mathbf{V})$. The term $(\mathbf{K}')^\top\mathbf{V}$ costs $O(Nd'd_v)$ to compute, resulting in a $d' \times d_v$ matrix. Multiplying this by $\mathbf{Q}'$ costs an additional $O(Nd'd_v)$. The total complexity is therefore $O(Nd'd_v)$, linear in the sequence length N.

F.3 Proof of Platonic Transformers Implementing Group Convolutions

The dynamic convolution from Proposition 4 becomes a dynamic *group convolution* within the Platonic Transformer. This is a direct consequence of applying the operation to lifted coordinates $\mathbf{p}_i(R) = R^{-1}\mathbf{p}_i$ for each reference frame $R \in \mathcal{G}$. Since the relative position vector becomes $R^{-1}(\mathbf{p}_j - \mathbf{p}_i)$, the kernel's input is transformed accordingly. The resulting output for each frame takes the form of a group cross-correlation[6]:

$$\mathbf{y}_i(R) = \sum_{j=1}^{N} \phi_{\mathbf{q}_i(R)}\left(R^{-1}(\mathbf{p}_j - \mathbf{p}_i)\right)\mathbf{v}_j(R). \tag{21}$$

Here, the kernel $\phi_{\mathbf{q}_i(R)}$ is steered by the group element R, defining an equivariant dynamic group convolution.

G Details of experiments on QM9 Regression

G.1 Description of the dataset

The QM9 dataset (Ramakrishnan et al., 2014) contains up to 9 heavy atoms and 29 atoms, including hydrogens. We use the train/val/test partitions introduced in (Gilmer et al., 2017), which consist of 100K/18K/13K samples, respectively, for each partition.

G.2 Training Details for the Regression Experiment

For the QM9 regression task, we train the Platonic Transformer to predict molecular properties. Before being fed to the model, the input molecular geometries are centered by subtracting the mean coordinate of each molecule. To stabilize training, we normalize the target property values by subtracting their mean and dividing by their standard deviation, with these statistics computed over the training set. We employ data augmentation in the form of random $SO(3)$ rotations applied to the coordinates during training.

The model is trained for a total of 1000 epochs using a batch size of 96. We utilize the Adam optimizer with a learning rate of 5×10^{-4} and a weight decay of 10^{-8}. A cosine annealing schedule with a 10-epoch linear warmup adjusts the learning rate throughout training. To prevent exploding gradients, we apply gradient clipping with a maximum norm of 0.5. The training objective is the Mean Absolute Error (MAE) on the normalized target values, while validation and testing are performed by calculating the MAE on the original, unnormalized scale. Our experiments explore different Platonic Transformer configurations, specifically by varying the symmetry group among `trivial_3`, `tetrahedron`, and `octahedron`. Further hyperparameter details are available in Table 5.

We choose FAFormer (Huang et al., 2024) to compare the contribution of Frame Averaging (with a standard Transformer backbone) with our end-to-end trained Platonic Transformer with different

[6]Following common convention, we refer to this operation as a group convolution, though it is technically a cross-correlation Cohen & Welling (2016); Bekkers (2020).

platonic solid symmetry groups. Recent efforts in long-context sequence modeling have also pit Attention-based methods with state-space methods. To provide a similar analysis for molecular modeling, we thus choose G-Hyena (Moskalev et al., 2025), a recent state-space model that relies on long convolutions for 3D molecular point clouds. We hyperparameter-tune FAFormer[7] and G-Hyena to have similar parameter counts and representational capacity as our best-performing Platonic Transformer on QM9. The baselines are also trained over 300 epochs with a batch size of 96 on a single H200 GPU.

G.3 Additional details on wall-clock timings

To produce the wall-clock timing for the standard Transformer in Table 4, node features from $B = 64$ QM9 molecules were projected from $d_{\text{in}} = 11$ to $d_{\text{model}} = 512$ using a Linear layer and fed as tokens into a `TransformerEncoderLayer` module provided by PyTorch with 16 heads. We measure wall-clock timings for a forward pass over 10 batches on a single H200 GPU. This offers a reference timing to compare the inference speed of the Platonic Transformer and other geometric baselines like G-Hyena (Moskalev et al., 2025) and the Tensor Field Network (Thomas et al., 2018).

G.4 Connection to Frame Averaging

Our discussion on the use of reference frames from platonic solid symmetry groups is also reminiscent of Frame Averaging (FA) (Puny et al., 2022), a framework that incorporates group equivariance to non-equivariant neural network backbones via *symmetrization*.

A backbone neural network $\Phi : V \to W$ on normed spaces can be made equivariant (or invariant) to a group $\mathcal{G}$. Specifically, for two group representations $\rho_1(g)$ and $\rho_2(g)$ that $g \in \mathcal{G}$ induces on V and W, $\Phi(\rho_1(g) \cdot \mathbf{x}) = \rho_2(g) \cdot \Phi(\mathbf{x})$, where $\mathbf{x} \in V$ and $\cdot$ is the group action on elements from V and W.

FA first constructs a *frame* $\mathcal{F}(\mathbf{x}) : V \to 2^{\mathcal{G}}$ with the following properties:

- A frame is $\mathcal{G}$-equivariant if $\mathcal{F}(\rho_1(g)\mathbf{x}) = g\mathcal{F}(\mathbf{x})$, where $g\mathcal{F}(\mathbf{x}) = \{gh | h \in \mathcal{F}(\mathbf{x})\}$, and
- A frame is bounded over a domain $K \subset V$ if $\exists c > 0$ such that the operator norm $\|\rho_2(g)\|_{\text{op}} \leq c, \forall g \in \mathcal{F}(\mathbf{x}), \forall \mathbf{x} \in K$.

To make the backbone Φ $\mathcal{G}$-equivariant, *symmetrization* is applied on $\mathbf{x}$ via the group averaging operator:

$$\langle \Phi \rangle_{\mathcal{F}}(\mathbf{x}) = \frac{1}{|\mathcal{F}(\mathbf{x})|} \sum_{g \in \mathcal{F}(\mathbf{x})} \rho_2(g)\Phi(\rho_1(g)^{-1}(\mathbf{x})). \tag{22}$$

For equivariance to 3-dimensional Euclidean rigid-body transformations (translations, rotations, reflections), i.e., $\mathcal{G} := E(3)$ and $V = \mathbb{R}^3$, $\mathcal{F}(\mathbf{x})$ is constructed using PCA on $\mathbf{x}$; this involves computing the centroid $\mathbf{t} = \frac{1}{n}\mathbf{x}^\top \mathbf{1} \in \mathbb{R}^3$ and covariance matrix $\mathbf{C} = (\mathbf{x} - \mathbf{1}\mathbf{t}^\top)^\top(\mathbf{x} - \mathbf{1}\mathbf{t}^\top)$ for a point cloud $\mathbf{x} \in \mathbb{R}^{n \times 3}$ with n points. Suppose we obtain eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3 \in \mathbb{R}^3$ of $\mathbf{C}$, we construct 3×3 orthogonal matrices by concatenating them together. Depending on the collection of these matrices, we achieve equivariance to different motion groups: for $E(3)$ equivariance, we use $\mathbf{U} = [\pm\mathbf{u}_1, \pm\mathbf{u}_2, \pm\mathbf{u}_3] \subset E(3)$, and if we restrict this collection to contain only orthogonal, positive orientation matrices, we achieve $SE(3)$ equivariance. The frame then looks like,

$$\mathcal{F}(\mathbf{x}) = \{(\mathbf{U}, \mathbf{t}) : \mathbf{U} = [\pm\mathbf{u}_1, \pm\mathbf{u}_2, \pm\mathbf{u}_3])\} \subset E(3).$$

For equivariant predictions on atomistic point clouds, we set $\rho_1(g)\mathbf{x} = \mathbf{x}\mathbf{U}^\top + \mathbf{1}\mathbf{t}^\top$ and $\rho_2(g)\mathbf{x} = \mathbf{x}\mathbf{U}^\top$ for each frame element, followed by the application of the averaging operator in Eq. 22. Furthermore, Puny et al. (2022) show that if $\mathcal{F}(\mathbf{x})$ is bounded, FA preserves the expressiveness of the underlying backbone, making $\langle \Phi \rangle_{\mathcal{F}}$ maximally expressive even for non-compact groups like $E(n)$.

FA can be extended to full $E(3)$ by including roto-reflections (e.g., roto-reflecting PCA axes) but comes at a cost, requiring a separate forward pass for each frame element. Our Platonic Transformer offers a more scalable drop-in replacement. For instance, using just a single orthogonal matrix from

[7]We use the open-source implementation of FAFormer provided at `https://github.com/Graph-and-Geometric-Learning/Frame-Averaging-Transformer`.

PCA (i.e., just one frame element) along with an axis-aligned roto-reflection group (here, we use $C_2 \times C_2 \times C_2$, a subgroup of the octahedral group) achieves native $E(3)$ equivariance at an ~ 8x lower cost.

We evaluate the effectiveness of FA by comparing to FAFormer (Huang et al., 2024), which applies FA to a standard Transformer backbone with 8 frame elements. As shown in Table 7, our single-frame variant (`8-Refl + 1 frame`) outperforms FAFormer with a smaller compute cost. This underscores the need for a geometrically expressive backbone, such as the Platonic Transformer.

Table 7: QM9 Property Prediction MAE (↓).

Platonic Transformer (end-to-end)		
Group	$\boldsymbol{\mu}$	$\boldsymbol{\alpha}$
$\emptyset$	0.028	0.064
Tetrahedron	0.012	0.049
Octahedron	**0.010**	**0.048**
8-Refl + 1 frame	0.039	0.155
Reference Method	**(re)produced herein*	
FAFormer [24]*	0.122	0.252

H Details of experiments on OMol25

H.1 Description of the dataset

For large-scale molecular experiments, we use the Open Molecules 2025 (OMol25) dataset (Levine et al., 2025), a comprehensive collection of over 100 million Density Functional Theory (DFT) calculations performed at the wB97M-V/def2-TZVPD level of theory. This dataset is notable for its vast chemical and structural diversity, encompassing 83 elements and systems up to 350 atoms. The structures are drawn from a wide range of chemical domains, including small molecules, biomolecules, metal complexes, and electrolytes, and feature varied charges, spin states, conformers, and reactive geometries.

The OMol25 dataset is organized into several training sets and splits for validation and testing to ensure consistent and robust model evaluation. The full training set, "All," contains over 100 million DFT calculations. For more computationally efficient training and development, a smaller, uniformly sampled "4M" split is provided, containing approximately 4 million structures. Our work primarily utilizes the "Neutral" split, which consists of approximately 34 million charge-neutral, singlet structures drawn from established community datasets like ANI-2X, GEOM, and SPICE2. This split is designed to benchmark model performance on familiar organic chemistry space without the added complexity of variable charge and spin.

For validation and testing, OMol25 provides several out-of-distribution (OOD) splits designed to evaluate model generalizability. The primary validation set ("Val Comp") consists of structures with compositions held out from the training set. Further specialized test sets include held-out organic and metal-complex reactions ("Test Reactivity"), experimental crystal structures from the Crystallography Open Database ("Test COD"), and unique anion structures ("Test Anions"), among others. The core task is Structure to Energy and Forces (S2EF), where models are evaluated on their ability to predict the total energy of a structure and the per-atom forces, with Mean Absolute Error (MAE) being the primary metric.

H.2 Training Details

We partition the "Neutral" split of the OMol25 dataset into an 80% training set and a 20% validation set, and report the final results on the official test set. The model is trained using an AdamW optimizer with a learning rate of 2×10^{-4} and a weight decay of 10^{-6}. The learning rate is managed by a cosine decay schedule, which includes a linear warmup period over the first 5 epochs. Training is conducted for a total of 22 epochs using a batch size of 64.

The training objective is a weighted sum of two components: a **Mean Squared Error (MSE)** loss for the total energy and a **Mean Absolute Error (MAE) on the force vectors**. The force loss is calculated as the average L2 norm (Euclidean distance) of the error between the predicted and target force vectors for each atom. To balance these two targets, the force loss component is weighted by a factor of $\lambda_F = 12.0$.

To ensure stable training on this large-scale task, we normalize the target energies. We apply a linear referencing scheme to the raw DFT energies. This method normalizes the total energy by subtracting the pre-computed DFT energies of the constituent isolated atoms:

$$E_{\text{ref}} = E_{\text{DFT}} - \sum_{i=1}^{N} E_i^{\text{atom}} \tag{23}$$

where E_{ref} is the target value for the model, E_{DFT} is the system's total energy, N is the number of atoms, and E_i^{atom} is the pre-computed DFT energy of an isolated atom of the same species as atom i. This procedure is consistent with the methodology used for the OC22 dataset (Tran et al., 2023) and helps maintain comparability with other large-scale models.

Given the significant computational requirements for training on OMol25, we established a fixed compute budget to ensure a fair comparison between models. Each model was trained for a maximum duration of 5 days on a node equipped with 4x NVIDIA 6000 Ada GPUs. The detailed hyperparameters for our model configuration on this dataset are summarized in Table 5.

I Implementing Platonic Transformers in the Fourier Domain of Finite Groups

With increasing hidden dimension (while not increasing sequence length), transformer blocks spend more and more of their total compute time in the pointwise linear layers. To improve speed it can then be worthwhile to implement the pointwise equivariant linear layers in the Fourier domain of the rotation group, a technique that has recently been successfully employed in computer vision (Bökman et al., 2025; Nordström et al., 2025). Considering the Fourier domain also sheds light on the connections between Platonic Transformers and equivariant networks with general steerable feature spaces (Cesa et al., 2022).

In this section we demonstrate how a Fourier domain implementation can improve computational efficiency in Platonic Transformers. In the Fourier domain, equivariant linear layers are block-diagonal, drastically reducing the required number of FLOPs for both forward and backward passes. We will see that with the number of hidden dimensions considered in this paper, a naive PyTorch implementation is not efficient enough to realize the reduction in FLOPs in terms of a substantial reduction in training throughput, but at a moderately higher number of hidden dimensions, there are throughput gains. This suggests that future scaling of Platonic Transformers will benefit from being implemented in the Fourier domain, and that more efficient implementations than our current one would be able to improve throughput even at smaller number of hidden dimensions.

We will use the tetrahedral symmetry group as a running example in this section. The reader is cautioned that the representations discussed in this section are representations of the rotation group, in contrast to the representations of the translation group discussed in Appendix A.

I.1 Introduction to the Fourier Theory of Finite Groups

The representation theory of finite groups is a well studied topic with many good text books. We recommend (Serre et al., 1977) for more detailed background than given here. Note that we consider vector spaces over the real numbers, which leads to a slightly more involved representation theory than complex numbers, see (Serre et al., 1977, Section II.12).

Recall from Appendix A.1 that a representation of a group $\mathcal{G}$ is a group homomorphism $\rho : \mathcal{G} \to GL(V)$, where V is a vector space. We will here consider finite real vector spaces $V = \mathbb{R}^n$ so that $\rho(g)$ can be considered real-valued invertible matrices. An irreducible representation is one where the matrices $\{\rho(g)\}_{g\in\mathcal{G}}$ can not be simultaneously block-diagonalized. Any finite group $\mathcal{G}$ has a finite number (up to ismorphisms) of irreducible representations (irreps) $\{\rho_i\}$ and they can be computed given the multiplication table of the group. Irreps are important because we can decompose any finite representation ρ into a direct sum of irreps by performing a change of basis, so statements about general representations often reduce to statements about irreps.

The features in Platonic Transformers are functions from $\mathcal{G}$ to $\mathbb{R}^C$, that transform under the left regular representation as explained in Appendix B. In order words, the representation that acts on them is a direct sum of C copies of the regular representation of $\mathcal{G}$. Let this representation be denoted $\tilde{\rho}$. Decomposing $\tilde{\rho}$ into irreps, we obtain

$$\tilde{\rho}(g) = Q \left(\bigoplus_i \rho_i(g)^{\oplus m_i} \right) Q^{-1} \tag{24}$$

for some multiplicities m_i of each irrep and a change of basis matrix Q that can be taken to be orthogonal.

Now, Schur's lemma says that any equivariant linear map between non-isomorphic irreps $\rho_i \neq \rho_j$ must be constant zero. Further, the space of equivariant linear maps between ρ_i and itself is 1-, 2-, or 4-dimensional and isomorphic (as a division algebra over $\mathbb{R}$) to the real numbers, complex numbers, or quaternions depending on whether ρ_i is of so-called real, complex or quaternion type. (The type of ρ_i can be computed.) This means that any linear map that is equivariant from $\tilde{\rho}$ to $\tilde{\rho}$ is actually block diagonal after having performed the change of basis in equation 24, in particular so are the group convolutions used in Platonic Transformers.

For cyclic groups, the block-diagonalization corresponds to the fact that convolutions are pointwise multiplications in the Fourier domain[8].

I.2 Fourier Theory of the Tetrahedral Group

Let us now consider the Tetrahedral rotation group as $\mathcal{G}$, consisting of the twelve rotational symmetries of a regular tetrahedron. This group is isomorphic to the alternating group A_4 and has three real irreps. The real irreps of the tetrahedral group are given by the one-dimensional trivial representation

$$\rho_1(R) = 1, \tag{25}$$

the three-dimensional standard representation

$$\rho_3(R) = R \tag{26}$$

and a two-dimensional representation ρ_2 that is defined as follows. Note that any element in $\mathcal{G}$ is either the identity, a rotation by $2\pi/3$ radians (there are 8 of these) or a rotation by π radians (there are 3 of these). For the identity and rotations by π,

$$\rho_2(R) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{27}$$

The rotations by $2\pi/3$ fall into two conjugacy classes of four elements each, where one conjugacy class contains the inverses of the second. We can arbitrarily choose one of the conjugacy classes and define

$$\rho_2(R) = \begin{pmatrix} \cos(2\pi/3) & -\sin(2\pi/3) \\ \sin(2\pi/3) & \cos(2\pi/3) \end{pmatrix} \tag{28}$$

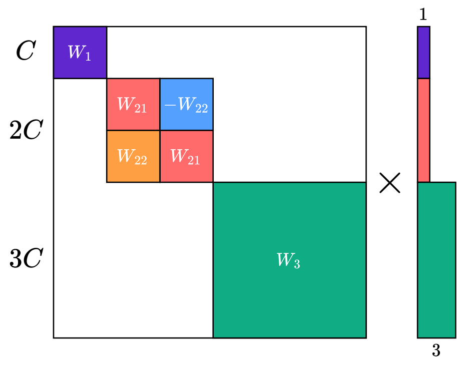
there, which implicitly defines the values for the second conjugacy class to be the inverse of the above.

It can be computed that ρ_1 and ρ_3 are both of real type, while ρ_2 is of complex type. Hence, equivariant linear maps from ρ_1 to ρ_1 are parameterized by one value, and the same for ρ_3. Equivariant linear maps from ρ_2 to ρ_2 are instead parameterized by two values (this is because ρ_2 splits into two irreps over the complex numbers).

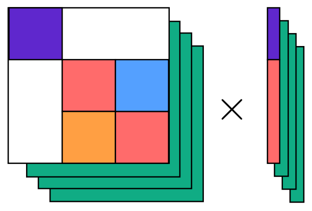
It can also be computed (or recovered from general facts of the Fourier transform over finite groups) that the representation $\tilde{\rho}$ acting on features with C channels in a tetrahedral Platonic Transformer splits into C copies of ρ_1, C copies of ρ_2 and $3C$ copies of ρ_3 (as a sanity check, we recover all $C + C \cdot 2 + 3C \cdot 3 = 12C$ dimensions).

As mentioned, Schur's lemma now implies that equivariant linear maps from $\tilde{\rho}$ to itself are block-diagonal. The map from copies of ρ_1 to copies of ρ_1 is parameterized by a $C \times C$ matrix, the map

[8]This requires working over the complex numbers, over the real numbers the pointwise multiplications turn into 2×2 matrix multiplications, again a block-diagonal structure.



(a) The block-diagonal structure of an equivariant weight matrix in the Fourier domain.



(b) We can implement the linear layer as a batched matrix-vector multiplication with four batches.

Figure 2: We visualize the weight matrices for linear layers that are equivariant under the tetrahedral rotation group, implemented in the Fourier domain. Each subfigure shows weights to the left and features to the right. Purple features transform according to ρ_1 (or technically $\rho_1 \otimes I_C$ since there are C copies of ρ_1), red features according to ρ_2 (by multiplication by $\rho_2(g) \otimes I_C$ from the left) and green features according to ρ_3 (by multiplication by $\rho_3(g)^\top$ from the right (if we flattened the green features, they would transform by $\rho_3(g) \otimes I_{3C}$ from the left)). The weight matrix is parameterized by the $C \times C$ matrices W_1, W_{21}, W_{22} and the $3C \times 3C$ matrix W_3, yielding a total of $12C^2$ learnable parameters. The total number of multiplications to compute the linear layer implemented as a batched matrix-multiplication in 2b is $4 \cdot (3C)^2 = 36C^2$, yielding a $4\times$ FLOP reduction versus an ordinary layer from $12C$ to $12C$ dimensions ($144C^2$ multiplications).

from copies of ρ_2 to copies of ρ_2 is parameterized by two $C \times C$ matrices (because ρ_2 is of complex type) and the map from copies of ρ_3 to copies of ρ_3 is parameterized by a $3C \times 3C$ matrix. Again, a sanity check gives that the full equivariant layer is then parameterized by $C^2 + 2C^2 + (3C)^2 = 12C^2$ values, which is the same as the group convolution discussed in Appendix C.1.

We visualize the weight structure in Figure 2a.

I.3 Implementation

We implement a version of the Platonic Transformer with tetrahedral equivariance and all linear layers (i.e. in the MLP and projections in multi-head attention) in the Fourier domain. We transform back to the spatial domain at each non-linearity and at the RoPE-attention layers and to the Fourier domain after these layers. This transforming back-and-forth incurs an overhead that goes to zero as the hidden dimension increases (since it is just the 12×12 matrix Q applied to each channel C), however it is non-negligible at low–medium number of hidden dimensions, because it involves non-contiguous reshapes.

The maximum FLOP saving that can be obtained from changing a linear layer to be in the Fourier domain is going from $(12C)^2 = 144C^2$ operations to $C^2 + (2C)^2 + 3 \cdot (3C)^2 = 32C^2$, i.e. a saving of 4.5 times. However, in order to make the implementation more efficient in pure PyTorch, we opt to implement the mappings for ρ_1 and ρ_2 as one single $3C \times 3C$ matrix, enabling the whole linear layer to be implemented as a batched matrix multiplication with four $3C \times 3C$ weight matrices, as illustrated in Figure 2b. This batched implementation uses $4 \cdot (3C)^2 = 36C^2$ operations, yielding a maximum potential compute saving of 4 times.

I.4 Throughput Benchmarking

We benchmark the training time per epoch on a subset of 20k molecules on the OMol25 task, using PyTorch's `torch.compile`. These timing runs are on a single NVIDIA RTX6000 GPU. We keep all hyperparameters constant as in the main experiments, except for varying the number of hidden dimensions. The results are presented in Table 8. It is clear that as we increase the number of hidden dimensions, a Fourier implementation starts paying off more and more. Notably, since the standard spatial implementation is equal to non-equivariant Transformers in computational cost, the efficiency improvement of the Fourier implementation is a benefit of equivariant architectures over non-equivariant ones. We emphasize that our Fourier implementation is not well-optimized, so further throughput improvements should be available.

Table 8: Training times per epoch (seconds) on a subset of OMol25 with 20k examples. We compare a tetrahedral Platonic Transformer implemented in the spatial domain with one implemented in the Fourier domain.

	Hidden dimension					
Implementation	576	864	1152	1440	1728	2016
Spatial (standard)	18	23	29	40	49	63
Fourier	19	22	27	32	38	45