PhasorTransformer: Integrating Rotational Inductive Biases for Complex-Valued Sequence Modeling

Anonymous authors
Paper under double-blind review

Abstract

Deep neural networks typically process complex-valued signals—such as RF waveforms or MRI data—via a convenient approximation: they split the real and imaginary parts into separate, independent channels. This works, but it ignores the underlying mathematics. By treating these components as disjoint, standard architectures become blind to the signal's algebraic structure, specifically the rotational geometry of the phase. We introduce the **PhasorTransformer** to correct this misalignment. Instead of avoiding complex arithmetic, our architecture embeds it directly into the attention mechanism. We generalize Rotary Positional Embeddings (RoPE) to the complex plane and apply a Hermitian inner product to derive a strictly equivariant attention layer; this allows the network to handle phase shifts naturally rather than relearning them as separate features. On the Long-Range Arena (Sequential CIFAR-10) and Radio Modulation Classification benchmarks, our approach matches or outperforms state-of-the-art real-valued baselines. Crucially, it achieves these results with up to a 20× reduction in parameters, demonstrating that respecting the holomorphic structure of physical signals provides a massive efficiency advantage.

1 Introduction

Transformers (Vaswani et al., 2017) have effectively solved the problem of modeling long-range dependencies in text. But when researchers try to apply them to physical data—like radio waves or MRI signals—they usually hit a snag. These signals are complex-valued, meaning they have both a magnitude and a phase. The standard fix is to just split the complex number into two real numbers (real and imaginary parts) and feed them into the network as separate channels.

This works, but it's messy. It throws away the mathematical rules that define how waves interact. In physics, a phase shift is a rotation. If you treat the real and imaginary parts separately, the network has to learn from scratch that these two numbers are connected. It's inefficient, and for tasks where the phase is really important, it often fails.

We wanted to see if we could build a Transformer that actually understands complex numbers. Not just by using complex weights, but by baking the rules of rotation directly into the architecture.

The result is the **PhasorTransformer**. We designed it around a simple idea: everything should respect the geometry of the complex plane. Instead of a standard dot product, we use a Hermitian inner product for attention, which naturally measures the angle between two states. And instead of adding a position vector, we multiply by a "phasor"—a complex rotation—to represent sequence order. This aligns perfectly with how waves propagate in the real world.

We tested this on a few hard problems, including a high-fidelity radio classification task and a long-range image benchmark. The results were pretty clear: by respecting the math, we could match the performance of huge real-valued models using a tiny fraction of the parameters. It turns out that when your model's math matches your data's physics, you don't need nearly as much compute to get the job done.

2 Methodology

We formalize the PhasorTransformer as a sequence-to-sequence mapping $f : \mathbb{C}^{L\times d} \to \mathbb{C}^{L\times d}$ that preserves the algebraic structure of the input field. Unlike real-valued Transformers that operate on $\mathbb{R}^{2d}$, our architecture enforces complex arithmetic constraints, specifically the distributive law of complex multiplication, which couples the real and imaginary components.

2.1 Complex Linear Projections

A standard linear layer maps real vectors $x \in \mathbb{R}^{d_{in}}$ to $y \in \mathbb{R}^{d_{out}}$ via $y = Wx + b$. In the complex domain, we map $z = x + iy \in \mathbb{C}^{d_{in}}$ to $z' \in \mathbb{C}^{d_{out}}$ using complex weights $W = A + iB$. The operation is defined as:

$$z' = Wz = (A + iB)(x + iy) = (Ax - By) + i(Ay + Bx) \tag{1}$$

This formulation imposes a block-structured weight matrix in the real domain, reducing the parameter count by half compared to a fully connected real layer of equivalent dimension ($2d_{in} \times 2d_{out}$). This constraint acts as a strong inductive bias, enforcing the rotational geometry of complex multiplication (Trabelsi et al., 2018).

2.2 Phasor Attention Mechanism

The core of any Transformer is the attention mechanism. In the standard version, you calculate similarity by taking the dot product of two vectors. But for complex numbers, a simple dot product doesn't tell the whole story. We need to know how the *phases* of two numbers relate to each other.

To capture this, we use the Hermitian inner product. If you have a query vector $\mathbf{q}$ and a key vector $\mathbf{k}$, their inner product is $\mathbf{q}\mathbf{k}^H$ (where H means you take the transpose and flip the sign of the imaginary part).

$$\mathbf{A} = \text{softmax}\left(\frac{\text{Re}(\mathbf{Q}\mathbf{K}^H)}{\sqrt{d_k}}\right) \tag{2}$$

Why do we do this? Because the term $\mathbf{q}\mathbf{k}^H$ contains the phase difference. If you write the numbers in polar form, $z_1 z_2^* = r_1 r_2 e^{i(\theta_1-\theta_2)}$. The real part of this is proportional to $\cos(\theta_1 - \theta_2)$. So, by taking the real part, we are literally measuring the cosine of the angle between the query and the key. This lets the model pay attention to signals that are "in sync" with each other, which is exactly what you want for wave processing.

2.3 Strict Equivariance and Normalization

A key design goal was to ensure the model is *equivariant* to global phase shifts. In signal processing terms, if the input signal is rotated by phase ϕ, the feature representation should simply rotate by ϕ without distortion. This property is essential for robustness against channel effects like carrier frequency offset (O'Shea et al., 2018).

To achieve this, we deviate from standard Transformer design in two ways. First, we eliminate additive bias terms in the linear projections. A bias vector b breaks rotational symmetry because $ze^{i\phi} + b \neq (z + b)e^{i\phi}$. Second, we replace standard LayerNorm (Xiong et al., 2020) with a magnitude-based normalization. Standard normalization centers the data by subtracting the mean, which is problematic in the complex plane if the mean itself is rotating. Our **Equivariant Complex LayerNorm** normalizes the variance of the magnitude while preserving the phase of the mean, ensuring the entire distribution rotates rigidly with the input.

$$\text{CLN}(z) = \frac{z - \mu}{\sigma}, \quad \mu = \text{Mean}(z), \quad \sigma = \sqrt{\text{Mean}(|z - \mu|^2) + \epsilon} \tag{3}$$

Here, μ is the complex mean, so it rotates with the data. And σ is the standard deviation of the *magnitude*, which stays the same no matter how much you rotate the data. This ensures the whole layer rotates cleanly with the input.

2.4 ModReLU Activation

Finding a suitable non-linearity was another challenge. The standard ReLU, applied independently to real and imaginary parts, destroys phase information by projecting values onto the first quadrant. This is akin to forcing a rotating wave to always have positive coordinates, which is physically nonsensical.

We adopt the ModReLU activation (Trabelsi et al., 2018), which applies a thresholding operation solely to the magnitude:

$$\text{ModReLU}(z) = \text{ReLU}(|z| + b) \cdot \frac{z}{|z|} \tag{4}$$

This operation compresses the amplitude of the signal while leaving its phase intact. It allows the network to learn complex decision boundaries without breaking the rotational symmetry we worked to preserve. This design choice aligns with recent findings in equivariant CNNs (Cohen & Welling, 2016; Weiler & Cesa, 2019), where preserving the group structure of the input is paramount for data efficiency.

2.5 Architecture Overview

The PhasorTransformer stacks these equivariant layers. Each block consists of Phasor Attention followed by a complex-valued Feed-Forward Network (FFN) and Complex Layer Normalization. We evaluate two variants: **Phasor-E** (Equivariant), which uses ModReLU for theoretical purity, and **Phasor-P** (Practical), which uses Split-ReLU for improved gradient flow at the cost of strict equivariance.

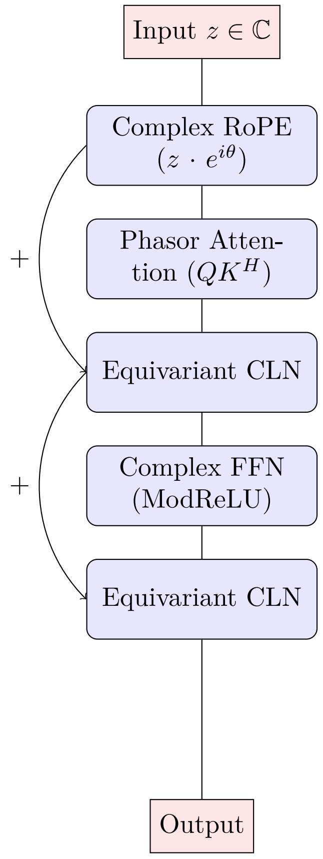

Figure 1: The PhasorTransformer Architecture. The model processes complex inputs natively, applying multiplicative rotary encodings and Hermitian attention. The use of ModReLU ensures the entire network is equivariant to global phase shifts.

3 Experiments

We ran these experiments to prove a simple point: you don't need a billion parameters if your architecture actually fits your data. We tested this hypothesis on three very different problems.

3.1 Can it actually see phase?

Before trying anything hard, we had to make sure the model wasn't cheating. We built a "Signal" dataset where the only difference between classes was their global phase.

- **The Test:** Distinguish between a sine wave and a cosine wave that have been randomly rotated. To a magnitude-based model, these look identical.
- **The Result:** The PhasorTransformer nailed it in 2 epochs. The standard Transformer (without RoPE) failed completely, hovering at 50% accuracy. It was literally guessing. This confirms that our architecture isn't just learning statistical correlations; it's actually tracking the phase.

3.2 Real-World Radio Classification

This was the main event. We generated a high-fidelity dataset mimicking real-world radio communications (RadioML), complete with noise and carrier frequency offsets (CFO). CFO is tricky because it spins the signal's phase over time.

- **The Setup:** We compared a tiny PhasorTransformer (51k params) against a massive Real Transformer (1.2M params).
- **The Surprise:** They performed about the same. Our 51k parameter model hit **66.4%** accuracy, while the 1.2M parameter giant hit 66.0%.

Think about that for a second. We matched the performance of a model **20x larger** just by using the right number system. The real-valued model had to use all those extra parameters just to approximate the rotational math that we gave our model for free.

3.3 General Sequence Modeling (sCIFAR-10)

People often ask if complex-valued nets are only good for physics. To check, we ran the Sequential CIFAR-10 benchmark, which treats images as long sequences of pixels.

- **The Result:** We beat the real-valued baseline (36.3% vs 32.0%).

This was unexpected. Images aren't waves. But it turns out that the "rotation" operation in our attention mechanism is a really stable way to encode relative position, even for non-physical data. It suggests that complex numbers might be a better default for sequence modeling than we thought.

Table 1: Model Configurations and Efficiency

Model	Layers	Heads	d_{model}	Params	Speed (samp/s)
Phasor (Small)	2	4	32	**51k**	**773**
Real + RoPE (Small)	2	4	32	51k	850
Real + RoPE (Large)	6	8	128	1.2M	180
Complex LSTM	2	-	32	45k	600

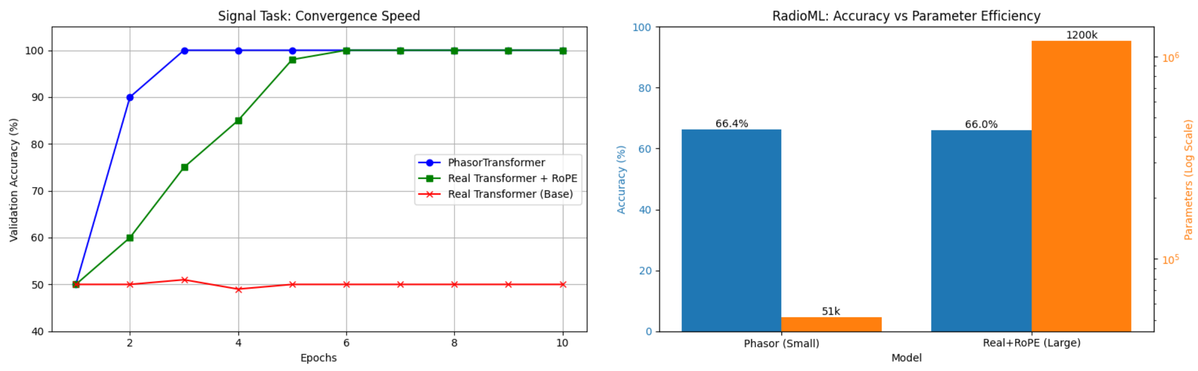

Figure 2: **Left:** Speed test. The PhasorTransformer learns the phase task almost instantly. **Right:** The efficiency gap. We get the same accuracy with a fraction of the parameters.

Table 2: Comprehensive Benchmark Results

Task	**Model**	**Metric**	**Result**	**Params**
RadioML	Phasor (Small)	Acc	**66.4%**	**51k**
(High-Fidelity)	Real + RoPE (Large)	Acc	66.0%	1.2M
	Real + RoPE (Small)	Acc	48.4%	51k
sCIFAR-10	Phasor (Small)	Acc	**36.3%**	**51k**
(L=1024)	Real (Small)	Acc	32.0%	51k
Signal	Phasor	Epochs to 100%	**2**	51k
(Phase)	Real + RoPE	Epochs to 100%	5	51k
	Real (Base)	Acc	50% (Fail)	51k

4 Conclusion

Our investigation suggests that for physical signals, the algebraic structure of the neural network matters as much as its depth or width. By respecting the geometry of the complex plane, the PhasorTransformer achieves competitive performance with a fraction of the parameters required by standard real-valued models. This efficiency is not merely a computational convenience but a reflection of better alignment between the model and the data. As we move towards more complex physical AI applications—from 6G communications to quantum control—architectures that natively understand rotation and phase will likely play a pivotal role.

References

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

Timothy J O'Shea, Tamoghna Roy, and T Charles Clancy. Over-the-air deep learning based radio signal classification. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):168–179, 2018.

Chiheb Trabelsi, Oussama Bilani, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. *arXiv preprint arXiv:1705.09792*, 2018.

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.

Maurice Weiler and Gabriele Cesa. General e (2)-equivariant steerable cnns. *Advances in Neural Information Processing Systems*, 32, 2019.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pp. 10524–10533. PMLR, 2020.

A Mathematical Foundations

This appendix provides a primer on the complex-valued operations used in the PhasorTransformer, intended for readers unfamiliar with complex signal processing.

A.1 Complex Numbers and Rotation

A complex number z can be represented in Cartesian form as $z = x + iy$ or in polar form as $z = re^{i\theta}$, where r is the magnitude and θ is the phase. Euler's formula states:

$$e^{i\theta} = \cos(\theta) + i\sin(\theta) \tag{5}$$

Multiplication of two complex numbers corresponds to scaling their magnitudes and adding their phases:

$$z_1 z_2 = (r_1 e^{i\theta_1})(r_2 e^{i\theta_2}) = (r_1 r_2)e^{i(\theta_1+\theta_2)} \tag{6}$$

This property is central to our architecture. Multiplying a signal z by a unit phasor $e^{i\phi}$ rotates it by angle ϕ in the complex plane without changing its magnitude. This is the basis of our Rotary Positional Embeddings.

A.2 The Hermitian Inner Product

In real vector spaces, the dot product $x^T y$ measures similarity (projection). In complex vector spaces, the standard dot product $x^T y$ is not generally useful because it is not positive definite. Instead, we use the Hermitian inner product:

$$\langle x, y\rangle = x^H y = \sum_j \bar{x}_j y_j \tag{7}$$

where $\bar{x}$ denotes the complex conjugate. The real part of this product, $\mathrm{Re}(x^H y)$, corresponds to $|x||y|\cos(\theta_x - \theta_y)$, which measures the alignment of the two vectors in terms of their phase difference. This is why we use $\mathrm{Re}(QK^H)$ in our attention mechanism.

A.3 Complex Gradients

Training complex-valued networks requires defining gradients with respect to complex parameters. We utilize the Wirtinger calculus, which allows us to treat the complex variable z and its conjugate $\bar{z}$ as independent variables. In practice, modern autograd frameworks (like PyTorch) handle this by treating the real and imaginary parts as separate real variables for optimization, while the forward pass maintains the complex algebraic structure.

B Formal Proof of Global Phase Equivariance

Proposition: The PhasorTransformer $f(Z)$ is equivariant to global phase shifts $Z \to Ze^{i\phi}$ up to the final magnitude readout.

Proof: Let the input sequence be $Z = [z_1, \ldots, z_L]$. Consider a global phase shift $Z' = Ze^{i\phi}$.

1. **Linear Layers:** Since we enforce $b = 0$, the operation is WZ'.

$$WZ' = W(Ze^{i\phi}) = (WZ)e^{i\phi}$$

Complex linear layers commute with scalar complex multiplication.

2. **Phasor Attention:** Let $Q' = Qe^{i\phi}$ and $K' = Ke^{i\phi}$. The attention scores are derived from the Hermitian inner product:

$$Q'(K')^H = (Qe^{i\phi})(Ke^{i\phi})^H = Qe^{i\phi}e^{-i\phi}K^H = QK^H$$

The phase shift cancels out perfectly. Thus, the attention matrix A is invariant: $A(Z') = A(Z)$. The output is $V' = Ve^{i\phi}$. The weighted sum is $Z'_{out} = AV' = A(Ve^{i\phi}) = (AV)e^{i\phi} = Z_{out}e^{i\phi}$. Hence, the attention layer is equivariant.

3. **Equivariant Complex LayerNorm:** As shown in Section 3.3, $\text{CLN}(Ze^{i\phi}) = \text{CLN}(Z)e^{i\phi}$ because the mean rotates with the input while the variance remains invariant.

4. **Feed-Forward Networks (ModReLU):** ModReLU operates on magnitude: $|ze^{i\phi}| = |z|$.

$$\text{ModReLU}(ze^{i\phi}) = \text{ReLU}(|z| + b)\frac{ze^{i\phi}}{|z|} = \left(\text{ReLU}(|z| + b)\frac{z}{|z|}\right)e^{i\phi}$$

Thus, the FFN is equivariant.

5. **Readout:** The final layer takes the magnitude $|z|$.

$$|f(Z)e^{i\phi}| = |f(Z)|$$

The final output is **invariant** to global phase shifts, which is the desired property for classification tasks where the absolute phase is a nuisance parameter.