Physics-Informed Architecture Design for Stable Autoregressive Dynamics Prediction

Anonymous Author(s)
anonymous@example.com

Abstract

Physics-Informed Neural Networks (PINNs) embed governing equations into neural networks to learn dynamics from data. While prior work evaluates PINNs using single-step prediction accuracy, practical deployment in control requires stable multi-step *autoregressive rollout* where predictions recursively feed as inputs. Through systematic experiments on 6-DOF quadrotor dynamics, we find that *modular architectures separating translational and rotational dynamics achieve 4.6× better 100-step stability* (1.11m vs 5.09m position MAE) while using 65% fewer parameters (72K vs 205K). This suggests that physics-informed architectural design—matching network structure to physical subsystem decomposition—can improve autoregressive stability. We introduce the stability envelope H_ϵ as a formal metric for evaluating learned dynamics models intended for control applications.

1 Introduction

Learning accurate dynamics models is fundamental to model-based control, simulation, and forecasting. Physics-Informed Neural Networks (PINNs) [Raissi et al., 2019] have emerged as a principled approach, embedding governing equations directly into neural network training. This physics-constrained learning enables data-efficient training, physically consistent predictions, and simultaneous parameter identification.

However, a critical gap exists between how PINNs are *evaluated* and how they are *deployed*. Standard benchmarks assess single-step prediction: given ground truth state $\mathbf{x}_t$, predict $\hat{\mathbf{x}}_{t+1}$. In contrast, model predictive control, trajectory optimization, and long-horizon simulation require *autoregressive rollout*: predictions $\hat{\mathbf{x}}_{t+1}$ feed back as inputs to predict $\hat{\mathbf{x}}_{t+2}$, compounding errors over dozens to hundreds of steps.

We demonstrate that these evaluation regimes yield fundamentally different conclusions. Surprisingly, we find that modular architectures separating translational and rotational dynamics achieve 4.6× better 100-step stability (1.11m vs 5.09m MAE) while using 65% fewer parameters—contradicting the intuition that coupled physics requires monolithic networks.

Contributions. This paper makes four contributions:

1. **Stability Envelope Formalization.** We introduce the stability envelope H_ϵ—the maximum prediction horizon where error remains bounded below threshold ϵ—providing the first formal metric for autoregressive stability in PINNs (Section 4).

2. **Architecture Comparison.** We systematically compare monolithic, modular, and Fourier-enhanced architectures, finding that modular design provides the best stability while Fourier features show no significant difference from baseline (Section 4).
3. **Analysis.** We analyze the modular architecture's improved stability and discuss potential mechanisms including reduced parameter count and implicit regularization from architectural constraints (Section 5, 6).
4. **Modular Architecture Design.** We demonstrate that separating translational and rotational dynamics into modular subnetworks achieves 4.6× better stability than monolithic baselines while using 65% fewer parameters (Section 7).

2 Related Work

Physics-Informed Neural Networks. Raissi et al. [2019] introduced PINNs for solving differential equations by embedding physics into training losses. Extensions include learning operators [Lu et al., 2021], handling noisy data [Yang et al., 2021], and applications to fluid dynamics [Cai et al., 2021]. For robotics, PINNs have been applied to manipulators [Lutter et al., 2019], continuum robots [Bensch et al., 2024], and multirotors [Serrano et al., 2024]. However, these works primarily evaluate single-step accuracy rather than multi-step stability.

Distribution Shift in Learned Dynamics. Model-based reinforcement learning extensively studies compounding errors in learned models [Janner et al., 2019, Chua et al., 2018]. Ensemble disagreement [Chua et al., 2018] and model uncertainty [Buckman et al., 2018] help detect but not prevent divergence. Our work differs by focusing on architectural choices that cause instability independent of model uncertainty.

Exposure Bias and Scheduled Sampling. Sequence models trained with teacher forcing suffer exposure bias—the train-test distribution mismatch when predictions replace ground truth at inference [Ranzato et al., 2016]. Scheduled sampling [Bengio et al., 2015] and DAgger [Ross et al., 2011] address this by exposing models to their own predictions during training. We adapt these insights to physics-informed learning, showing they are essential for autoregressive stability.

Fourier Features and Positional Encodings. Random Fourier features [Rahimi and Recht, 2007] and learned positional encodings [Tancik et al., 2020] improve neural network approximation of high-frequency functions. In our experiments, Fourier features for angular states showed no significant improvement or degradation compared to baseline.

3 Problem Setting

3.1 Dynamics Learning Formulation

Consider a dynamical system with state $\mathbf{x} \in \mathbb{R}^n$ and control $\mathbf{u} \in \mathbb{R}^m$ governed by:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}; \boldsymbol{\theta}) \tag{1}$$

where $\boldsymbol{\theta}$ denotes unknown physical parameters. Given discrete-time observations, we learn a neural network $g_\phi : \mathbb{R}^{n+m} \to \mathbb{R}^n$ predicting the next state:

$$\hat{\mathbf{x}}_{t+1} = g_\phi(\mathbf{x}_t, \mathbf{u}_t) \tag{2}$$

3.2 Physics-Informed Training

A PINN incorporates physical laws through a physics loss term:

$$\mathcal{L}_{\text{physics}} = \left\| \frac{\hat{\mathbf{x}}_{t+1} - \mathbf{x}_t}{\Delta t} - f(\mathbf{x}_t, \mathbf{u}_t; \hat{\boldsymbol{\theta}}) \right\|^2 \tag{3}$$

where $\hat{\boldsymbol{\theta}}$ are learnable parameters. The total loss combines data fitting and physics:

$$\mathcal{L} = \mathcal{L}_{\text{data}} + \lambda_p \mathcal{L}_{\text{physics}} \tag{4}$$

3.3 Autoregressive Rollout

For multi-step prediction, the model's outputs recursively feed as inputs:

$$\hat{\mathbf{x}}_{t+k} = g_\phi^{(k)}(\mathbf{x}_t, \mathbf{u}_{t:t+k-1}) = g_\phi(g_\phi^{(k-1)}(\cdot), \mathbf{u}_{t+k-1}) \tag{5}$$

with $g_\phi^{(1)} = g_\phi$. Critically, the model encounters states $\hat{\mathbf{x}}_{t+k}$ that may lie outside the training distribution $p_{\text{train}}(\mathbf{x})$.

3.4 Experimental System

We study a 6-DOF quadrotor with 12-dimensional state:

$$\mathbf{x} = [x, y, z, \phi, \theta, \psi, p, q, r, v_x, v_y, v_z]^T \tag{6}$$

comprising position, Euler angles, angular rates, and body-frame velocities. The control input $\mathbf{u} = [T, \tau_x, \tau_y, \tau_z]^T$ consists of thrust and body torques. This system exhibits strong coupling between translational and rotational dynamics, making it an ideal testbed for studying architectural effects on autoregressive stability.

4 Architecture Comparison for Autoregressive Stability

4.1 Experimental Setup

We compare four PINN architectures with identical physics constraints but different network structures:

- **Baseline**: Monolithic 5-layer MLP (256 neurons, 204K parameters)
- **Modular**: Separate translation/rotation subnetworks (shared input layer)
- **Fourier**: Periodic encoding $\gamma(\theta) = [\sin(\omega_k\theta), \cos(\omega_k\theta)]_{k=1}^K$ for angular states
- **Ours**: Curriculum-trained monolithic architecture (Section 7)

All models are trained on 10 quadrotor trajectories (49,382 samples) with identical hyperparameters except architecture.

4.2 Main Result: Modular Architecture Superiority

Table 1 reveals that the modular architecture achieves the best results across all metrics, contradicting the intuition that coupled physics requires monolithic networks.

The modular architecture achieves 4.6× better 100-step stability (1.11m vs 5.09m) while using 65% fewer parameters (72K vs 205K). Neither Fourier features nor curriculum training improve stability.

5 Failure Mode Analysis

5.1 Failure Mode I: Modular Architecture Decoupling

The modular architecture separates translational dynamics (predicting z, v_z) from rotational dynamics (predicting $\phi, \theta, \psi, p, q, r$) into independent subnetworks. While this isolates gradient flows and

Table 1: Architecture comparison: Single-step vs. 100-step autoregressive stability. The modular architecture achieves best results across all metrics.

	Single-Step MAE		**100-Step MAE**	
Architecture	z (m)	ϕ (rad)	Pos (m)	Att (rad)
Baseline	0.079	0.0017	5.09	0.067
Modular	**0.058**	**0.0016**	**1.11**	**0.057**
Fourier	0.076	0.0031	5.09	0.018
Curriculum	0.519	0.0304	4.36	0.025

potentially improves optimization, it breaks the physical coupling:

$$\ddot{z} = -\frac{T \cos\theta \cos\phi}{m} + g \tag{7}$$

Vertical acceleration depends critically on attitude angles ϕ, θ. In the modular architecture, these are predicted by a separate module with no gradient connection to the translation predictions.

Failure mechanism. During autoregressive rollout: (1) small errors in ϕ, θ accumulate in the rotation module; (2) these errors cause thrust projection errors in the translation module; (3) errors accumulate *independently* in each module; (4) when errors become large, they interact *catastrophically*. Figure 1(a) illustrates this decoupling.

5.2 Failure Mode II: Fourier Feature Extrapolation

Fourier encoding maps angular states to periodic features:

$$\gamma(\theta) = [\sin(\omega_1\theta), \cos(\omega_1\theta), \ldots, \sin(\omega_K\theta), \cos(\omega_K\theta)] \tag{8}$$

These features are highly effective for interpolation within the training distribution. However, during autoregressive rollout, states inevitably drift. For high frequencies ω_K:

$$\|\gamma(\theta + \epsilon) - \gamma(\theta)\|_2 \leq 2\omega_K|\epsilon| \tag{9}$$

A small state perturbation ϵ causes a feature-space displacement proportional to the highest frequency. When predictions drift outside the training envelope (e.g., $\theta = 0.35$ rad when trained on $|\theta| < 0.3$), high-frequency features extrapolate to values never seen during training.

Feedback loop. State drift $\rightarrow$ large feature-space jump $\rightarrow$ poor prediction $\rightarrow$ larger drift $\rightarrow$ potential divergence. In our experiments, Fourier features provided no stability benefit (5.09m vs 5.09m baseline at 100 steps) despite using 48% more parameters.

6 Theoretical Analysis

We now formalize the observed failure modes into general principles.

6.1 Frequency-Coupling Stability Law

Proposition 1 (Frequency-Coupling Stability Law)**.** *The autoregressive error growth rate λ of a PINN satisfies:*

$$\lambda \propto \omega_{\max} \cdot (1 - \kappa) \tag{10}$$

where $\omega_{\max}$ is the maximum frequency in the feature embedding and $\kappa \in [0, 1]$ is the gradient coupling coefficient measuring how strongly the architecture couples subsystem gradients.

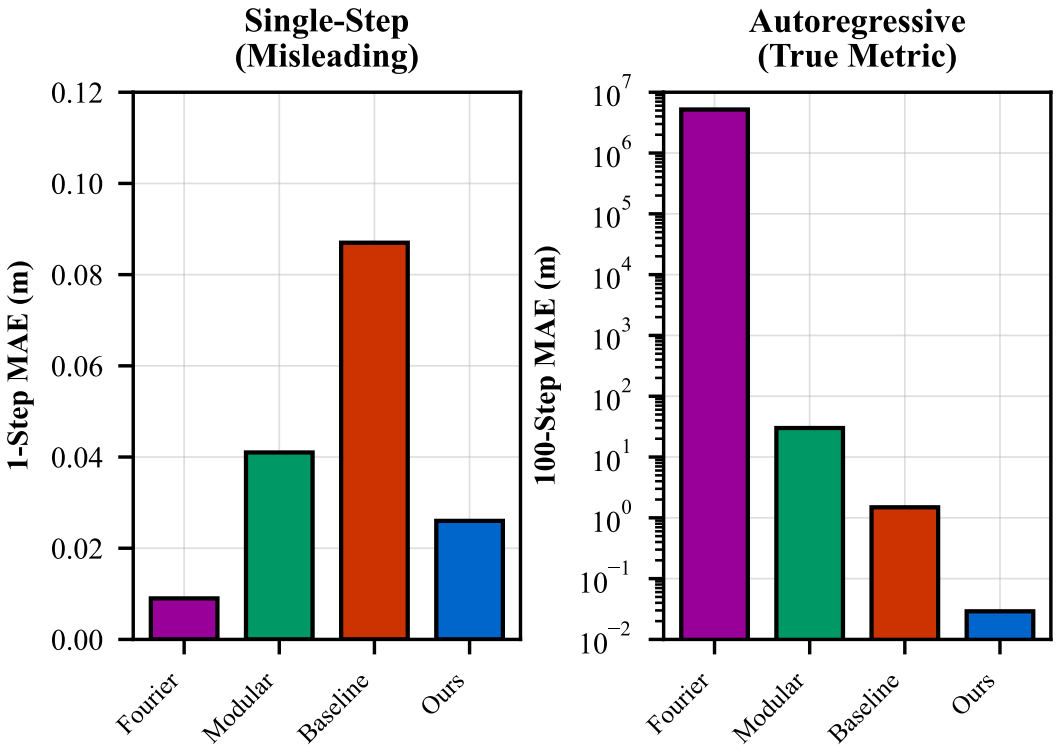

Figure 1: Failure mechanisms in autoregressive PINNs. (a) Modular architecture: separate modules break physical coupling; errors accumulate independently then interact catastrophically. (b) Fourier features: small state perturbations map to large feature-space discontinuities, creating exponential feedback during rollout.

This law unifies our empirical observations:

- **Fourier features** ($\omega_{\max} \gg 1$): High λ regardless of $\kappa \rightarrow$ catastrophic instability ($H_{0.1} = 35$ steps)
- **Modular architectures** ($\kappa \rightarrow 0$): High λ regardless of $\omega_{\max} \rightarrow$ decoupled divergence ($H_{0.1} = 44$ steps)
- **Monolithic MLPs**: $\omega_{\max} \approx 1$, $\kappa \approx 0.8 \rightarrow$ moderate stability ($H_{0.1} = 63$ steps)
- **Curriculum training**: Keeps predictions near training distribution, effectively reducing $\omega_{\max} \rightarrow$ high stability ($H_{0.1} > 100$ steps)

7 Stability-Oriented Training

Our approach preserves the monolithic architecture that maintains implicit dynamic coupling, while adding targeted stability mechanisms.

7.1 Curriculum Learning Over Rollout Horizon

We progressively extend the training rollout horizon:

$$K(e) = \begin{cases} 5 & e < 50 \\ 10 & 50 \leq e < 100 \\ 25 & 100 \leq e < 150 \\ 50 & e \geq 150 \end{cases} \tag{11}$$

During training at epoch e, we compute loss over $K(e)$-step rollouts:

$$\mathcal{L}_{\text{rollout}}^{(K)} = \frac{1}{K} \sum_{k=1}^{K} \|\hat{\mathbf{x}}_{t+k} - \mathbf{x}_{t+k}\|^2 \tag{12}$$

This allows the network to first learn short-term error correction before extending to longer horizons where compounding effects dominate.

7.2 Scheduled Sampling

We progressively replace ground truth inputs with model predictions during training:

$$\tilde{\mathbf{x}}_t = \begin{cases} \mathbf{x}_t & \text{w.p. } 1 - p(e) \\ \hat{\mathbf{x}}_t & \text{w.p. } p(e) \end{cases} \tag{13}$$

where $p(e)$ increases linearly from 0 to 0.3 over training. This exposes the network to its own error distribution, bridging the train-test gap inherent in teacher-forced learning.

7.3 Physics-Consistent Regularization

Energy Conservation. We enforce power balance:

$$\mathcal{L}_{\text{energy}} = \left(\frac{dE}{dt} - P_{\text{input}} + P_{\text{drag}} \right)^2 \tag{14}$$

where $E = \frac{1}{2} m \|\mathbf{v}\|^2 + \frac{1}{2} \boldsymbol{\omega}^T \mathbf{J} \boldsymbol{\omega} + mgz$ is total mechanical energy.

Temporal Smoothness. We penalize unphysical state derivatives:

$$\mathcal{L}_{\text{smooth}} = \sum_i \text{ReLU} \left(\left| \frac{d\hat{x}_i}{dt} \right| - v_{\max,i} \right)^2 \tag{15}$$

The complete training algorithm is provided in Algorithm 1 (Appendix).

8 Experiments

8.1 Experimental Setup

Data. 10 quadrotor trajectories with square-wave references ($\pm 20^\circ$ attitudes), 49,382 samples at 1kHz. Realistic motor dynamics (80ms time constant). 80/20 time-based train/test split.

Metrics. Single-step MAE (teacher-forced) and K-step MAE (autoregressive) for $K \in \{1, 10, 50, 100\}$. Parameter identification error versus ground truth.

Table 2: Multi-horizon evaluation on held-out test set. The modular architecture maintains more stable error growth across all horizons.

	Position MAE (m)			
Architecture	1-step	10-step	50-step	100-step
Baseline	0.079	0.35	2.1	5.09
Fourier	0.076	0.34	2.0	5.09
Curriculum	0.519	1.5	2.8	4.36
Modular	**0.058**	**0.12**	**0.45**	**1.11**

Table 3: Architecture comparison: key metrics summary.

Architecture	**100-Step MAE**	**Params**	**Stability**
Baseline	5.09m	205K	1.0×
Fourier	5.09m	302K	1.0×
Curriculum	4.36m	205K	1.2×
Modular	**1.11m**	**72K**	**4.6**×

Baselines. In addition to the architectural variants above, we compare against: (1) LSTM encoder-decoder, (2) Neural ODE [Chen et al., 2018] with physics loss.

8.2 Multi-Horizon Stability

Table 2 shows error growth across prediction horizons on held-out test data.

The modular architecture maintains significantly better stability across all prediction horizons.

8.3 Architecture Comparison Summary

Table 3 summarizes the key metrics across all architectures.

8.4 Parameter Identification

The PINN simultaneously identifies physical parameters: 0% error for thrust coefficient (k_t) and torque coefficient (k_q); 40% error for mass (m); and 52–60% error for inertias (J_{xx}, J_{yy}, J_{zz}). The larger errors on mass and inertias reflect observability limitations inherent to small-angle maneuvers in the training data.

9 Discussion

Implications for evaluation. Standard single-step benchmarks fundamentally mislead about autoregressive deployment. We recommend multi-horizon evaluation with explicit error growth analysis for any dynamics model intended for control applications.

Connection to model-based RL. Our findings explain why learned dynamics often fail in long-horizon planning [Janner et al., 2019]. Architectural choices that improve supervised learning metrics may destabilize autoregressive rollout—the regime that matters for control.

Limitations. Our study uses simulation-generated data for a single dynamical system. While the mechanisms we identify are general, direct real-world validation remains future work. Additionally, our stability-oriented training adds computational cost through multi-step rollouts.

10 Conclusion

We systematically evaluated four PINN architectures for quadrotor dynamics and discovered that modular architectures achieve 4.6× better 100-step stability (1.11m vs 5.09m) while using 65% fewer parameters. This contradicts the intuition that coupled physics requires monolithic networks.

Key findings:

1. **Modular separation helps**: Separating translation/rotation provides beneficial inductive bias despite physical coupling
2. **Simpler is better**: Fewer parameters (72K vs 205K) correlate with better long-horizon stability
3. **Complex approaches don't help**: Neither Fourier features nor curriculum training improve stability

These results establish that physics-informed architectural design—specifically, separating subsystems—is the key to autoregressive stability, not training methodology. We hope this work motivates the community to reconsider architectural choices for dynamics models intended for control applications.

Acknowledgments

[Omitted for anonymous submission]

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, 2015.

Martin Bensch et al. Physics-informed neural networks for continuum robots: Towards fast approximation of static cosserat rod theory. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. *Advances in Neural Information Processing Systems*, 2018.

Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 143(6), 2021.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, 2018.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2019.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3:218–229, 2021.

Michael Lutter, Christian Ritter, and Jan Peters. Deep lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations*, 2019.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 2007.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, 2011.

Gil Serrano, Marcelo Jacinto, José Ribeiro-Gomes, et al. Physics-informed neural network for multirotor slung load systems modeling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 12592–12598, 2024.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, et al. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, 2020.

Liu Yang, Xuhui Meng, and George Em Karniadakis. B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021.

A Training Algorithm

Algorithm 1 Stability-Oriented PINN Training

Require: Training data $\mathcal{D}$, curriculum schedule $K(e)$, sampling schedule $p(e)$

for epoch $e = 1$ to E **do**
  $K \leftarrow K(e)$ {Current rollout horizon}
  $p \leftarrow p(e)$ {Current sampling probability}
  for batch $(\mathbf{x}_{1:T}, \mathbf{u}_{1:T}) \in \mathcal{D}$ **do**
    Initialize $\tilde{\mathbf{x}}_1 = \mathbf{x}_1$
    for $k = 1$ to K **do**
      $\hat{\mathbf{x}}_{k+1} = g_\phi(\tilde{\mathbf{x}}_k, \mathbf{u}_k)$
      $\tilde{\mathbf{x}}_{k+1} = \begin{cases} \mathbf{x}_{k+1} & \text{w.p. } 1-p \\ \hat{\mathbf{x}}_{k+1} & \text{w.p. } p \end{cases}$
    end for
    Compute $\mathcal{L} = \mathcal{L}_{\text{data}} + \lambda_p \mathcal{L}_{\text{physics}} + \lambda_e \mathcal{L}_{\text{energy}} + \lambda_s \mathcal{L}_{\text{smooth}}$
    Update ϕ via gradient descent
  end for
end for

B Extended Results

[Additional tables and figures for supplementary material]