The GNN as a Low-Pass Filter: A Spectral Perspective on Achieving Stability in Neural PDE Solvers

Peilin Rao
Department of Economics
University of California, Los Angeles
Los Angeles, CA 90024
jackrao@g.ucla.edu

Abstract

The choice of architecture in Graph Machine Learning (GML) presents a fundamental trade-off between the expressive power of universal approximators and the implicit regularization conferred by structured models like Graph Neural Networks (GNNs). This paper provides a principled framework for navigating this trade-off, using the challenging scientific domain of solving high-dimensional Hamilton-Jacobi-Bellman (HJB) partial differential equations as a testbed. Through a series of controlled experiments, we demonstrate that while flexible, unstructured networks excel for problems with smooth, globally-structured solutions, they fail catastrophically on problems with complex, non-smooth features. We connect this success to the GNN's established properties as a spectral low-pass filter, demonstrating how this provides the implicit Lipschitz regularization needed to learn stable and generalizable solutions. This prevents the numerical instabilities that plague unconstrained models. Our findings culminate in a framework that connects the mathematical properties of a problem's solution space to the optimal choice of GML architecture, offering a new perspective on the role of architectural bias as a powerful regularization tool.

1 Introduction

A central challenge in Graph Machine Learning (GML) is selecting an architecture with the appropriate inductive bias. The spectrum of choices ranges from unstructured, universal approximators like Feed-Forward Networks (FFNs) [Hornik et al., 1989] to highly structured models like Graph Neural Networks (GNNs). This choice dictates a fundamental trade-off between expressive power and implicit regularization, directly impacting a model's ability to generalize. When should an architectural constraint be viewed as a helpful regularizer versus a detrimental "straitjacket"?

To investigate this question, we turn to the challenging scientific domain of solving high-dimensional partial differential equations (PDEs), specifically the Hamilton-Jacobi-Bellman (HJB) equation, which arises in optimal control. Solving the HJB equation is hampered by the "curse of dimensionality" [Bellman, 1957], and neural network solvers represent a significant breakthrough [Han et al., 2018]. This context is an ideal testbed for GML, as HJB solutions can range from smooth and global to highly localized and non-smooth.

This paper demonstrates that the optimal architecture for this task is predictably determined by the mathematical structure of the PDE's solution. We develop our argument through a "three-act" empirical narrative, showing that while an FFN is superior for smooth problems, it fails catastrophically on non-smooth problems where the structural bias of a GNN is essential for learning stable, physically-plausible solutions.

We provide a formal theoretical foundation for the GNN's success by leveraging its established understanding as a spectral low-pass filter [Kipf and Welling, 2017, Wu et al., 2019]. Our contribution is to explicitly link this spectral property to the implicit Lipschitz regularization required for solving HJB equations, thereby explaining the mechanism behind the observed numerical stability. Section 2 formulates the problem, Section 3 presents our empirical results, Section 4 details the GNN's spectral bias, and Section 5 synthesizes these findings into a framework.

2 Problem Formulation

2.1 The Hamilton-Jacobi-Bellman Equation for Multi-Agent Systems

We consider a system of N interacting agents whose joint state $\mathbf{x}_t \in \mathbb{R}^N$ evolves according to a stochastic differential equation (SDE):

$$d\mathbf{x}_t = \boldsymbol{\mu}(\mathbf{x}_t, \boldsymbol{\alpha}_t)dt + \boldsymbol{\Sigma}(\mathbf{x}_t)d\mathbf{W}_t, \tag{1}$$

where $\boldsymbol{\mu}$ is the drift, $\boldsymbol{\Sigma}$ is the volatility, $\boldsymbol{\alpha}_t \in \mathbb{R}^N$ is the control, and $\mathbf{W}_t$ is a Wiener process. The solution to the optimal control problem is the value function, $V(t, \mathbf{x})$, which must satisfy a Hamilton-Jacobi-Bellman (HJB) PDE. The specific form depends on whether the problem is cooperative (a single PDE) or non-cooperative (a system of coupled PDEs), as detailed in Appendix B. Our goal is to train a neural network $V_{\boldsymbol{\theta}}$ to parameterize the value function by minimizing the HJB equation's mean squared residual.

2.2 Solver Architectures and Inductive Biases

We investigate three neural architectures for parameterizing $V_{\boldsymbol{\theta}}$:

- **Feed-Forward Network (FFN):** A universal approximator with no structural priors. It treats the input $\mathbf{x}$ as a flat vector, relying on the implicit spectral bias of gradient descent for regularization [Rahaman et al., 2019].
- **Symmetric MLP (SymmetricMLP):** Imposes a strong feature bias via permutation equivariance [Zaheer et al., 2017]. It processes per-agent features based on low-dimensional, hand-crafted statistics (e.g., agent state $\mathbf{x}_i$ and global mean $\bar{\mathbf{x}}$) using a shared MLP.
- **Graph Neural Network (GNN):** Imposes a structural bias by constraining information flow to a specified graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. We use a Graph Convolutional Network (GCN) [Kipf and Welling, 2017], with layer-wise updates $\boldsymbol{H}^{(l+1)} = \sigma(\hat{\boldsymbol{A}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$, where $\hat{\boldsymbol{A}}$ is the symmetrically normalized adjacency matrix.

3 The Empirical Investigation: A Tale of Three Games

All experiments were conducted on a single NVIDIA 4070 GPU. We design a sequence of three experiments to deconstruct the trade-offs between these architectures. Full mathematical specifications for each are in Appendix B.

3.1 Act I: The Primacy of Feature Bias in a Mean-Field Game

Setup: We implement a linear-quadratic (LQ) mean-field game where interactions are all-to-all (a complete graph). The analytical solution for each agent's value function is a simple quadratic form of its deviation from the global mean (see Appendix B.2). This provides a clear test for the feature-engineered SymmetricMLP.

Result: As shown in Table 1, while the GNN achieves the lowest HJB loss, the most critical metric—True L2 Error against the known solution—shows the SymmetricMLP dramatically outperforming both alternatives by nearly an order of magnitude.

Insight: When a problem's structure can be distilled into low-dimensional sufficient statistics, explicit feature engineering is the most effective approach. The SymmetricMLP's direct access to the mean feature was a more powerful bias than the GNN's general structural constraint. The GNN's low HJB loss but high L2 error indicates it found a function that locally satisfied the PDE but missed the correct global structure.

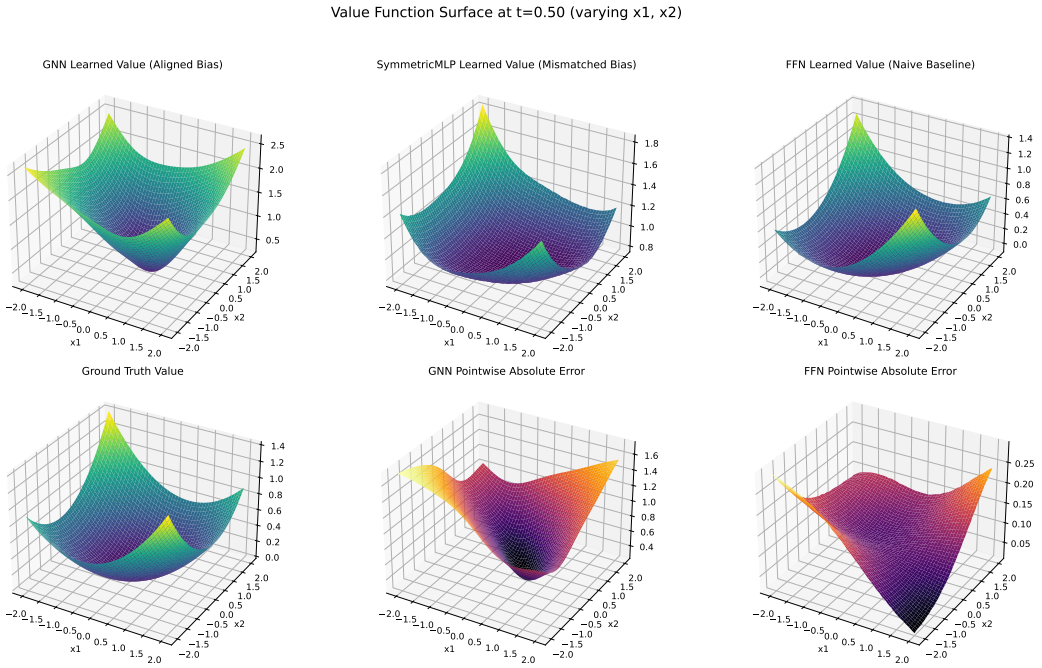


Figure 1: Learned value function surfaces for the Sparse LQ Game (Act II). The FFN (top right) learns a smooth surface that closely matches the Ground Truth (bottom left), while the GNN (top left) learns an incorrect function, visually demonstrating the failure of its mismatched bias.

3.2 Act II: The Peril of Mismatched Bias in a Sparse LQ Game

Setup: We pivot to a social planner LQR problem on a sparse ring graph. Agent dynamics depend only on their immediate neighbors, a structure seemingly aligned with the GNN's bias. However, the true value function $V(\mathbf{x}, t) = \frac{1}{2}\mathbf{x}^T \boldsymbol{P}_t \mathbf{x}$ is governed by a matrix Riccati equation whose solution, $\boldsymbol{P}_t$, is a dense, globally-structured matrix (see Appendix B.3).

Result: Against expectations, the GNN fails catastrophically. The winner, as shown in Table 1 and Figure 1, is the unstructured FFN, which learns a near-perfect approximation of the true quadratic value function.

Insight: The GNN's architectural bias, while aligned with the system's local dynamics, acted as a detrimental "straitjacket," preventing it from approximating the globally-structured, dense quadratic solution. The FFN, free from structural priors, easily discovered the smooth function. This reveals a critical lesson: the architectural bias must match the functional form of the solution, not just the system's one-step dynamics.

Table 1: Consolidated results for Act I and Act II. The primary success metric (True L2 Error) is bolded.

Experiment	Model	Parameters	Final HJB Loss	Final True L2 Error
Act I: Mean-Field Game	**SymmetricMLP**	4,481	0.0719	**0.0040**
	FFN	18,053	0.3079	0.0481
	GNN	3,417	*0.0248*	0.1529
Act II: Sparse LQ Game	**FFN**	**18,177**	**0.2016**	**0.0263**
	SymmetricMLP	21,793	0.6101	0.1424
	GNN	1,185	3.2144	3.7266

Note: In Act I, the GNN's low HJB loss is misleading, as the L2 error reveals it failed to find the true solution.

3.3 Act III: The Vindicating Power of Regularization

Setup: We introduce complexity by adding a localized, non-linear cost term, $\lambda(\mathbf{x}^0)^4$, to a single agent in the ring graph (the "Clog in the Ring" game, specified in Appendix B.4). This modification makes the true value function non-quadratic and non-smooth, with sharp local gradients. No analytic solution exists, so we evaluate models on test-set HJB residual, Monte Carlo rollout cost, and critically, the stability of the learned policy.

Result: The roles dramatically reverse. The FFN, previously the victor, now learns an erratic policy (Figure 2b) leading to catastrophic performance across all metrics (Table 2). More tellingly, a conflict emerges between the remaining models: both the Monte Carlo cost and the HJB residual suggest the SymmetricMLP is the superior model, as it achieves lower values than the GNN. This sets the stage for a deeper analysis to uncover the true nature of the learned solutions.

Insight: This experiment reveals the potential for standard metrics to be misleading and confirms our central thesis: for problems with complex, non-smooth solutions, implicit architectural regularization is paramount. To demonstrate this, we perform a local stability analysis by computing the eigenvalues of the learned closed-loop system's Jacobian at the $\mathbf{x} = \mathbf{0}$ equilibrium (see Appendix B.4 for the formulation).

The results, summarized in Table 2, are definitive.

- The **FFN** learns a catastrophically unstable policy, with a maximum real eigenvalue of $+6.63$. Its failure is corroborated by its massive HJB residual and rollout cost. This instability is visualized in the bottom row of Figure 2, where the norm of the value function's gradient explodes.
- The **SymmetricMLP** appears optimal based on cost and HJB residual. However, stability analysis reveals its max real eigenvalue is $+3.50 \times 10^{-8}$. It achieves its low cost by reverting to a trivial, passive, and ultimately unstable policy that fails to actively control the non-linear dynamics.
- Only the **GNN** leverages its structural bias to learn a provably stable policy, with a max real eigenvalue of -1.66. It finds a valid, active control strategy, even at the expense of slightly higher cost and residual metrics.

The GNN's architectural constraint acts as an essential regularizer, preventing the instabilities that plague the FFN and guiding the solver to a physically-meaningful solution that the other, more misleading metrics would have missed. For complex dynamic systems, stability is the decisive measure of success, and the GNN's implicit regularization is the key to achieving it.

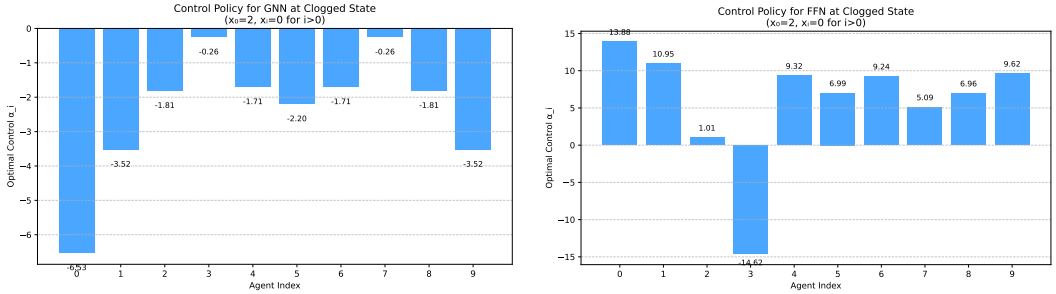
Table 2: Final evaluation for the "Clog in the Ring" Game (Act III).

Model	Final Test HJB Residual	MC Mean Cost	MC Std Error	Max Real Eigenvalue	Stability
GNN	12,079.68	15.29	0.22	**-1.66**	**Stable**
SymmetricMLP	*11,020.24*	*13.08**	0.79	+3.50e-08	Unstable
FFN	40,400.03	18,242.22	289.26	+6.63	Unstable

Note: *The SymmetricMLP's low cost and HJB residual are misleading; they result from a passive policy that fails to actively control the system's non-linearities. Only the GNN learns an active and provably stable control strategy.

4 Theoretical Foundation: GNN as a Low-Pass Filter

We now formalize why the GNN succeeds in Act III, proving that the GCN layer functions as a spectral low-pass filter. This filtering promotes smoother functions, effectively regularizing the solution's Lipschitz constant and preventing the instabilities seen in the FFN. We use Graph Fourier Analysis, where eigenvectors $\boldsymbol{U}$ of the normalized adjacency matrix $\hat{\boldsymbol{A}} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$ form an orthonormal basis representing graph frequencies [Chung, 1997].



(a) GNN Policy

(b) FFN Policy

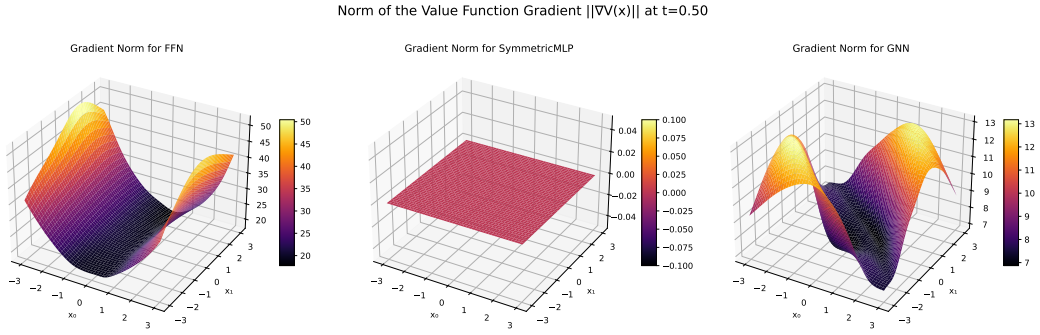


Figure 2: Analysis for the "Clog" game. **(Top Row)** Learned policies $\hat{\boldsymbol{\alpha}} = -\nabla_{\mathbf{x}} V$ at a test state. The GNN (a) learns a plausible policy while the FFN's (b) is nonsensical. **(Bottom Row)** Norm of the value function gradient, $||\nabla_{\mathbf{x}} V||_2$, across a 2D slice of the state space. The FFN's gradient exhibits exploding magnitudes, visually explaining its instability, while the GNN's gradient is smooth and controlled as a direct consequence of its implicit regularization.

Theorem 1 (GCN as a Low-Pass Filter). *Let a GCN layer be defined by $\boldsymbol{H}_{out} = \sigma(\hat{\boldsymbol{A}}\boldsymbol{H}_{in}\boldsymbol{W})$. The graph convolution operator $\hat{\boldsymbol{A}}$ decomposes any input signal into low-frequency and high-frequency components and has the following effect:*

1. *The low-frequency component (in the eigenspace of $\lambda_1 = 1$) is preserved.*
2. *The norm of any high-frequency component is strictly contracted by a factor of at least $\lambda_{\max}^{\perp} = \max(|\lambda_2|, |\lambda_N|) < 1$.*

Since the gain on the lowest frequency is 1 while the gain on all higher frequencies is strictly less than 1, the operator acts as a low-pass filter.

Proof Sketch. The proof follows by linearity. The low-frequency component is an eigenvector with eigenvalue 1 and is unchanged. Any high-frequency component is a linear combination of other eigenvectors, and its norm contracts because all other eigenvalues have magnitude strictly less than 1. A 1-Lipschitz activation preserves this attenuation. ■.

A full proof, including necessary preliminaries on graph spectral theory, is provided in Appendix C.

From Spectral Bias to Implicit Lipschitz Regularization

Theorem 1 explains the GNN's stability. The Lipschitz constant of a multi-layer network is bounded by the product of its layers' Lipschitz constants. For an FFN, this product can grow, permitting solutions with massive gradients, as seen in Act III. For a GNN, the graph convolution operator $\hat{\boldsymbol{A}}$ is not merely non-expansive (Lipschitz constant of 1); it is an *active low-pass filter*. By systematically contracting high-frequency components at every layer, the GNN architecture penalizes high-frequency oscillations. This spectral bias regularizes the solution space, inherently limiting the Lipschitz constant of both $V_{\boldsymbol{\theta}}$ and its gradient $\nabla_{\mathbf{x}} V$, providing the implicit regularization that ensures a stable learned policy.

5 A Principled Framework for Solver Selection

Our results culminate in a framework for selecting a neural HJB solver based on the expected characteristics of the problem's solution, summarized in Table 3.

Table 3: A Principled Framework for Neural HJB Solver Selection

Problem Property	Example Solution Character	Optimal Solver	Governing Principle	Primary Validation Criterion
Mean-Field Interaction	Global, simple aggregates	**SymmetricMLP**	Feature Bias	True L2 Error vs. Analytic
Local, Simple Cost	Global, smooth, quadratic	**FFN**	Approximation Power	True L2 Error vs. Riccati
Local, Complex Cost	Non-smooth, sharp gradients	**GNN**	Structural Regularization	**Stable Closed-Loop Dynamics** (via max real eigenvalue), which serves as the decisive criterion over misleadingly low MC cost and HJB residuals.

Our empirical and theoretical results culminate in the framework summarized in Table 3. The choice of architecture should be dictated by the expected smoothness of the solution: feature bias for simple aggregates, approximation power for global smoothness, and the GNN's structural regularization for non-smooth functions requiring stability.

Limitations

The primary limitation of this work is the scope of its empirical validation. Our experiments, while carefully designed as controlled testbeds to isolate architectural trade-offs, are based on synthetic HJB problems. A crucial next step is to apply and validate this framework on more challenging, realistic HJB test cases from scientific domains such as mathematical finance or multi-agent robotics. Such validation would be necessary to demonstrate the practical utility of our proposed insights. Furthermore, our study employs a standard GCN; future work should also investigate whether the insights generalize to more advanced GNN architectures (e.g., with attention) and other classes of PDEs where solution stability is paramount.

6 Conclusion

We established a principled framework for architectural selection in GML, grounded in the trade-off between approximation power and implicit regularization. Using HJB equations as a testbed, we showed that the optimal architecture depends on the mathematical properties of the target function. Our central contribution is to explain and apply the GNN's role as an implicit regularizer in a new scientific domain. By connecting the theory of GCNs as spectral low-pass filters to the practical challenge of solving dynamic systems, we established a concrete mechanism by which architectural bias enforces smoothness and confers stability. This insight is key to understanding GNN success on complex scientific problems where more flexible models fail. While our controlled experiments provide a clear theoretical narrative, the next stage of this research is to validate this framework on larger-scale, real-world control problems. Doing so would fulfill the promise of a GML paradigm that deliberately selects architectures whose spectral biases align with the expected structure of a problem's solution.

Broader Impacts Statement

This foundational research aims to improve the reliability of GML models in scientific applications. While it has no direct societal impact, its advances could enable more stable solutions to optimal control problems in economics and engineering. Potential negative impacts are not from malicious use but from misapplication; deploying a control policy from an unstable or misspecified model in a safety-critical domain could have dangerous consequences. This underscores the need for careful validation and domain-expert oversight when applying machine learning to high-stakes problems.

References

Richard E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

Rene A. Carmona, Jean-Pierre Fouque, and Li-Hsien Sun. Mean field games and systemic risk. *Communications in Mathematical Sciences*, 13(4):911–933, 2015. doi: 10.4310/CMS.2015.v13.n4.a4.

Fan R. K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997. doi: 10.1090/cbms/092.

Jiequn Han, Arnulf Jentzen, and Weinan E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018. doi: 10.1073/pnas.1718942115.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. doi: 10.1016/0893-6080(89)90020-8.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=SJU4ayYgl`.

Nasim Ullah Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron C. Courville. On the Spectral Bias of Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5301–5310. PMLR, 2019. URL `http://proceedings.mlr.press/v97/rahaman19a.html`.

Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr., Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6861–6871. PMLR, 2019. doi: 10.48550/arXiv.1902.07153.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R. Salakhutdinov, and Alexander J. Smola. Deep Sets. In *Advances in Neural Information Processing Systems 30*, 2017. URL `http://papers.nips.cc/paper/6931-deep-sets.pdf`.

7 Appendix: Supplementary Materials

A Hyperparameters and Implementation Details

This section provides the hyperparameters used for training the models in each of the three empirical acts. All models were implemented in JAX and Flax.

Table 4: Model and Training Hyperparameters

Parameter	Act I	Act II	Act III
Learning Rate	1×10^{-4}	3×10^{-4}	1×10^{-4}
Training Steps	20,000	50,000	60,000
Batch Size	256	256	256
Terminal Weight	100.0	100.0	100.0
Optimizer	AdamW	AdamW	AdamW
FFN Hidden Dims	[128, 128]	[128, 128]	[128, 128]
SymmetricMLP Hidden Dims	[64, 64]	Ag:[32,32], Hd:[64]	Ag:[32,32], Hd:[64]
GNN Hidden Dim	56	32	32

B Detailed Problem Formulations

B.1 The General Hamilton-Jacobi-Bellman Equation

We consider a system of N interacting agents whose joint state $\mathbf{x}_t \in \mathbb{R}^N$ evolves according to the stochastic differential equation (SDE) in Equation 1. The specific form of the HJB partial differential equation (PDE) that the value function must solve depends on the nature of the multi-agent interaction.

- **Social Planner (Cooperative) Problem:** A central planner seeks to minimize a single, joint cost. This yields a single, global value function $V(t, \mathbf{x})$ that solves the HJB equation:

$$-\frac{\partial V}{\partial t} = \inf_{\boldsymbol{\alpha} \in \mathbb{R}^N} \left\{ (\nabla_{\mathbf{x}} V)^T \boldsymbol{\mu}(\mathbf{x}, \boldsymbol{\alpha}) + L(\mathbf{x}, V) + C(\mathbf{x}, \boldsymbol{\alpha}) \right\} \tag{2}$$

  where $L(\mathbf{x}, V) = \frac{1}{2}\text{Tr}\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T \text{Hess}_{\mathbf{x}}(V)\right)$ is the second-order diffusion term and $C(\cdot)$ is the joint running cost.

- **Nash Equilibrium (Non-Cooperative) Problem:** Each agent i seeks to minimize its own cost, taking the policies of other agents as given. This results in a system of N coupled PDEs, one for each agent's value function $V^i(t, \mathbf{x})$:

$$-\frac{\partial V^i}{\partial t} = \inf_{\alpha_i \in \mathbb{R}} \left\{ \mathcal{L}^{\hat{\boldsymbol{\alpha}}_{-i}} V^i(\mathbf{x}) + C_i(\mathbf{x}, \alpha_i) \right\} \tag{3}$$

  where $\mathcal{L}^{\hat{\boldsymbol{\alpha}}_{-i}}$ is the infinitesimal generator of the SDE under the optimal policies of all other agents, $\hat{\boldsymbol{\alpha}}_{-i}$, and C_i is the running cost for agent i.

B.2 Act I: Mean-Field LQ Game

This model is based on the canonical mean-field game formulation for systemic risk presented in Carmona et al. [2015]. It is a linear-quadratic game of N agents on a complete graph.

B.2.1 System Dynamics and Cost

The state of agent i, $\mathbf{x}_t^i$, evolves according to:

$$d\mathbf{x}_t^i = \left[a(\bar{\mathbf{x}}_t - \mathbf{x}_t^i) + \alpha_t^i\right] dt + \text{noise}_t^i \tag{4}$$

where $\bar{\mathbf{x}}_t = \frac{1}{N}\sum_j \mathbf{x}_t^j$. In matrix form, the drift is $\boldsymbol{\mu}(\mathbf{x}_t, \boldsymbol{\alpha}_t) = \boldsymbol{F}\mathbf{x}_t + \boldsymbol{B}\boldsymbol{\alpha}_t$, with:

- **Dynamics Matrix:** $\boldsymbol{F} = a(\frac{1}{N}\boldsymbol{J} - \boldsymbol{I}) = -\frac{a}{N}\boldsymbol{L}_{\text{complete}}$, where $\boldsymbol{J}$ is the matrix of ones and $\boldsymbol{L}_{\text{complete}}$ is the standard graph Laplacian for the complete graph.
- **Control Matrix:** $\boldsymbol{B} = \boldsymbol{I}$.

The noise has common and idiosyncratic components, with covariance $(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T)_{jk} = \sigma^2(\rho^2 + \delta_{jk}(1 - \rho^2))$. Each agent i seeks to minimize a cost functional $J_i(\boldsymbol{\alpha})$ with running cost f_i and terminal cost g_i:

$$f_i(\mathbf{x}, \alpha_i) = \frac{1}{2}(\alpha_i)^2 - q\alpha_i(\bar{x} - x_i) + \frac{\epsilon}{2}(\bar{x} - x_i)^2 \tag{5}$$

$$g_i(\mathbf{x}) = \frac{c}{2}(\bar{x} - x_i)^2 \tag{6}$$

This defines a Nash Equilibrium problem where each agent's value function V^i must satisfy a coupled HJB equation.

B.2.2 Analytical Solution

The true value function for agent i is quadratic:

$$V^i(t, \mathbf{x}) = \frac{\eta_t}{2}(\bar{x} - x_i)^2 + \mu_t \tag{7}$$

This can be written as $V^i(t, \mathbf{x}) = \frac{1}{2}\mathbf{x}^T\boldsymbol{P}_t^i\mathbf{x} + \mu_t$, where the state-dependent part is determined by a time-dependent, rank-one matrix:

$$\boldsymbol{P}_t^i = \eta_t \left(\frac{1}{N}\mathbf{1} - \mathbf{e}_i\right)\left(\frac{1}{N}\mathbf{1} - \mathbf{e}_i\right)^T \tag{8}$$

Here, $\mathbf{e}_i$ is the i-th standard basis vector and the scalar function η_t solves the Riccati ODE:

$$\dot{\eta}_t = 2(a + q)\eta_t + \left(1 - \frac{1}{N^2}\right)\eta_t^2 - (\epsilon - q^2), \quad \eta_T = c \tag{9}$$

B.3 Act II: Sparse LQ Social Planner

This model is a cooperative linear-quadratic regulator problem on a ring graph.

B.3.1 System Dynamics and Cost

The state of agent i evolves according to local diffusion dynamics:

$$d\mathbf{x}_t^i = \left[a(\mathbf{x}_t^{i-1} - 2\mathbf{x}_t^i + \mathbf{x}_t^{i+1}) + \alpha_t^i\right] dt + \text{noise}_t^i \tag{10}$$

In matrix form, $\boldsymbol{\mu}(\mathbf{x}_t, \boldsymbol{\alpha}_t) = \boldsymbol{F}\mathbf{x}_t + \boldsymbol{B}\boldsymbol{\alpha}_t$, with:

- **Dynamics Matrix:** $\boldsymbol{F} = a\boldsymbol{L}_{\text{ring}}$, where $\boldsymbol{L}_{\text{ring}}$ is the sparse graph Laplacian for the ring graph.
- **Control Matrix:** $\boldsymbol{B} = \boldsymbol{I}$.

The social planner minimizes a single cost functional $J(\boldsymbol{\alpha})$ with quadratic running cost f and terminal cost g:

$$f(\mathbf{x}, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T\boldsymbol{\alpha} + \frac{1}{2}\mathbf{x}^T\boldsymbol{Q}\mathbf{x}, \quad \text{where} \quad \boldsymbol{Q} = \epsilon(-\boldsymbol{L}_{\text{ring}}) \tag{11}$$

$$g(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\boldsymbol{Q}_T\mathbf{x}, \quad \text{where} \quad \boldsymbol{Q}_T = c(-\boldsymbol{L}_{\text{ring}}) \tag{12}$$

This defines a social planner problem with a single value function $V(t, \mathbf{x})$ that must solve the corresponding HJB PDE.

B.3.2 Analytical Solution

The true value function has a global quadratic structure:

$$V(t, \mathbf{x}) = \frac{1}{2}\mathbf{x}^T \boldsymbol{P}_t \mathbf{x} + \mu_t \tag{13}$$

The matrix $\boldsymbol{P}_t \in \mathbb{R}^{N \times N}$ solves the continuous-time matrix Differential Riccati Equation (DRE):

$$-\dot{\boldsymbol{P}}_t = \boldsymbol{Q} + \boldsymbol{F}^T \boldsymbol{P}_t + \boldsymbol{P}_t \boldsymbol{F} - \boldsymbol{P}_t \boldsymbol{B}\boldsymbol{B}^T \boldsymbol{P}_t \tag{14}$$

With the specific matrices for this model, this becomes:

$$-\dot{\boldsymbol{P}}_t = \epsilon(-\boldsymbol{L}_{\text{ring}}) + a(\boldsymbol{L}_{\text{ring}}\boldsymbol{P}_t + \boldsymbol{P}_t \boldsymbol{L}_{\text{ring}}) - \boldsymbol{P}_t^2 \tag{15}$$

solved backwards from the terminal condition $\boldsymbol{P}_T = c(-\boldsymbol{L}_{\text{ring}})$. Although the system matrices $\boldsymbol{F}$ and $\boldsymbol{Q}$ are sparse, the solution $\boldsymbol{P}_t$ to the Riccati equation is a dense, non-local matrix.

B.4 Act III: The "Clog in the Ring" Game

This model modifies the sparse LQ problem of Act II to include a localized, non-linear cost.

B.4.1 System Dynamics and Cost

The system dynamics and noise structure are identical to those in Act II, governed by $\boldsymbol{F} = a\boldsymbol{L}_{\text{ring}}$. The social planner's cost functional is modified with a quartic penalty on the state of agent 0:

$$f(\mathbf{x}, \boldsymbol{\alpha}) = \underbrace{\frac{1}{2}\boldsymbol{\alpha}^T \boldsymbol{\alpha} + \frac{1}{2}\mathbf{x}^T \boldsymbol{Q}\mathbf{x}}_{\text{Original LQ Cost}} + \underbrace{\lambda(\mathbf{x}^0)^4}_{\text{The Clog}} \tag{16}$$

$$g(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T \boldsymbol{Q}_T \mathbf{x} + \lambda(\mathbf{x}^0)^4 \tag{17}$$

The closed-loop HJB equation for the value function $V(t, \mathbf{x})$ is:

$$\begin{aligned} -\frac{\partial V}{\partial t} = (\nabla_{\mathbf{x}} V)^T \boldsymbol{F}\mathbf{x} - \frac{1}{2}||\nabla_{\mathbf{x}} V||_2^2 \\ + \left(\frac{1}{2}\mathbf{x}^T \boldsymbol{Q}\mathbf{x} + \lambda(\mathbf{x}^0)^4\right) \\ + \frac{1}{2}\text{Tr}\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T \text{Hess}_{\mathbf{x}}(V)\right) \end{aligned} \tag{18}$$

Due to the non-quadratic $(\mathbf{x}^0)^4$ term, this HJB equation does not admit an analytical solution.

B.4.2 Stability Analysis Formulation

The stability of the learned policy is assessed by linearizing the learned closed-loop dynamics around the equilibrium point $\mathbf{x} = \mathbf{0}$. The Jacobian matrix of the closed-loop system is given by:

$$\boldsymbol{A}_{\text{cl}}(\mathbf{x}) = \boldsymbol{F} - \boldsymbol{B}\boldsymbol{B}^T \text{Hess}_{\mathbf{x}}(V_{\boldsymbol{\theta}}(\mathbf{x})) \tag{19}$$

A stable policy requires that all eigenvalues of $\boldsymbol{A}_{\text{cl}}(\mathbf{0})$ have negative real parts.

C Full Theoretical Proofs

C.1 Preliminaries: Graph Fourier Analysis

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a connected, non-bipartite graph. The symmetrically normalized adjacency matrix $\hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-1/2}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-1/2}$ is real and symmetric, admitting an eigendecomposition $\hat{\boldsymbol{A}} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$.

- The columns of the orthogonal matrix $\boldsymbol{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_N]$ form the Graph Fourier Basis.

- The diagonal matrix $\mathbf{\Lambda}$ contains the real eigenvalues, ordered $1 = \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_N \geq -1$. The eigenvector $\mathbf{u}_1$ is the constant vector, representing the lowest frequency (DC component), while subsequent eigenvectors correspond to increasingly higher frequencies of oscillation on the graph.

This eigendecomposition serves as the basis for graph Fourier analysis, where the eigenvectors of the graph operator are interpreted as frequency components [Chung, 1997].

Any signal $\mathbf{h} \in \mathbb{R}^N$ on the graph can be decomposed into two orthogonal subspaces relative to this basis:

- **Low-Frequency Subspace:** $\mathcal{H}_{\|} = \text{span}(\mathbf{u}_1)$.
- **High-Frequency Subspace:** $\mathcal{H}_{\perp} = \text{span}(\mathbf{u}_2, \ldots, \mathbf{u}_N)$.

The following lemma, which establishes that $\hat{\boldsymbol{A}}$ is a contraction mapping on the high-frequency subspace, is central to our main result.

Lemma 1 (High-Frequency Contraction).

Let $\lambda_{\max}^{\perp} = \max(|\lambda_2|, |\lambda_N|) < 1$. For any signal $\mathbf{h}_{\perp} \in \mathcal{H}_{\perp}$, the operator $\hat{\boldsymbol{A}}$ is a strict contraction:

$$||\hat{\boldsymbol{A}}\mathbf{h}_{\perp}||_2 \leq \lambda_{\max}^{\perp} \cdot ||\mathbf{h}_{\perp}||_2$$

Proof. Any $\mathbf{h}_{\perp} \in \mathcal{H}_{\perp}$ is a linear combination $\mathbf{h}_{\perp} = \sum_{i=2}^{N} c_i \mathbf{u}_i$. Applying the operator yields $\hat{\boldsymbol{A}}\mathbf{h}_{\perp} = \sum_{i=2}^{N} c_i \lambda_i \mathbf{u}_i$. Due to the orthonormality of $\boldsymbol{U}$, the squared norm is $||\hat{\boldsymbol{A}}\mathbf{h}_{\perp}||_2^2 = \sum_{i=2}^{N} c_i^2 \lambda_i^2 \leq (\lambda_{\max}^{\perp})^2 \sum_{i=2}^{N} c_i^2 = (\lambda_{\max}^{\perp})^2 ||\mathbf{h}_{\perp}||_2^2$. Taking the square root completes the proof. ■

C.2 Full Proof of Theorem 1 (GCN as a Low-Pass Filter)

Theorem 1 (GCN as a Low-Pass Filter). *Let a single GCN layer be defined by the transformation $\boldsymbol{H}_{out} = \sigma(\hat{\boldsymbol{A}}\boldsymbol{H}_{in}\boldsymbol{W})$, where σ is a 1-Lipschitz activation function with $\sigma(0) = 0$. Let $\boldsymbol{v} = \boldsymbol{H}_{in}\boldsymbol{w}_k$ be the aggregated feature vector for an output channel k, and decompose it into its orthogonal frequency components, $\boldsymbol{v} = \boldsymbol{v}_{\|} + \boldsymbol{v}_{\perp}$. The pre-activation vector $\tilde{\boldsymbol{h}}_k = \hat{\boldsymbol{A}}\boldsymbol{v}$ exhibits the following spectral properties:*

1. *The low-frequency component is preserved: $\hat{\boldsymbol{A}}\boldsymbol{v}_{\|} = \boldsymbol{v}_{\|}$.*
2. *The norm of the high-frequency component is strictly contracted: $||\hat{\boldsymbol{A}}\boldsymbol{v}_{\perp}||_2 \leq \lambda_{\max}^{\perp} ||\boldsymbol{v}_{\perp}||_2$.*

Since the gain on the low-frequency component is 1 while the gain on all higher-frequency components is strictly less than 1, the graph convolution operator $\hat{\boldsymbol{A}}$ acts as a low-pass filter.

Proof. The aggregated feature vector for channel k is $\boldsymbol{v} = \boldsymbol{H}_{\text{in}}\boldsymbol{w}_k$. The pre-activation is $\tilde{\boldsymbol{h}}_k = \hat{\boldsymbol{A}}\boldsymbol{v}$. We analyze the action of $\hat{\boldsymbol{A}}$ on each orthogonal component of $\boldsymbol{v}$ by linearity: $\tilde{\boldsymbol{h}}_k = \hat{\boldsymbol{A}}(\boldsymbol{v}_{\|} + \boldsymbol{v}_{\perp}) = \hat{\boldsymbol{A}}\boldsymbol{v}_{\|} + \hat{\boldsymbol{A}}\boldsymbol{v}_{\perp}$.

1. **Effect on Low Frequencies:** By definition, $\boldsymbol{v}_{\|}$ lies in the eigenspace of $\hat{\boldsymbol{A}}$ corresponding to the eigenvalue $\lambda_1 = 1$. Therefore, $\hat{\boldsymbol{A}}\boldsymbol{v}_{\|} = \lambda_1 \boldsymbol{v}_{\|} = \boldsymbol{v}_{\|}$. The low-frequency component passes through the filter unchanged, with its norm perfectly preserved.
2. **Effect on High Frequencies:** The component $\boldsymbol{v}_{\perp}$ lies entirely within the high-frequency subspace $\mathcal{H}_{\perp}$. Applying Lemma 1 directly gives $||\hat{\boldsymbol{A}}\boldsymbol{v}_{\perp}||_2 \leq \lambda_{\max}^{\perp} ||\boldsymbol{v}_{\perp}||_2$. Since $\lambda_{\max}^{\perp} < 1$, the high-frequency component is strictly contracted.
3. **Impact of Activation:** The final output for the channel is $\boldsymbol{h}_{k,\text{out}} = \sigma(\tilde{\boldsymbol{h}}_k)$. As σ is 1-Lipschitz with $\sigma(0) = 0$, it is a non-expansive mapping: $||\sigma(\mathbf{z})||_2 \leq ||\mathbf{z}||_2$. This ensures the activation cannot reverse the relative attenuation of high-frequency components achieved in the linear step. ■

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction (Section 1) claim to provide a framework for architecture selection based on the trade-off between expressive power and implicit regularization, demonstrated through HJB solvers. The paper's contributions—including the empirical investigation in Section 3, the theoretical link to spectral filtering in Section 4, and the resulting framework in Section 5—directly and accurately fulfill these claims.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper includes a dedicated "Limitations" section before the Conclusion. This section addresses the scope of the study regarding the specific class of PDEs and GNN architecture used, framing these as avenues for future research.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: The main theoretical result, Theorem 1, is presented in Section 4 with a proof sketch. A complete and formal proof, along with all necessary assumptions and a supporting lemma, is provided in Appendix C.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: The paper provides sufficient detail for reproducibility. Appendix B gives the complete mathematical formulations for all three experiments. Appendix A lists all key hyperparameters, model architectures, and training details in Table 4.

   Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [No]

   Justification: The paper does not provide a link to the source code for the experiments. However, the experiments are based on synthetic data generated from the mathematical formulations provided in Appendix B, and detailed hyperparameters are given in Appendix A, which would facilitate a re-implementation.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: All necessary experimental details are provided. The experimental setups are described in Section 3, with full mathematical details in Appendix B. Training details, including optimizers, learning rates, and model hyperparameters, are specified in Appendix A.

   Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: The paper reports standard error for the Monte Carlo cost estimates in Table 2, which is the primary stochastic evaluation metric. Other reported metrics, such as True L2 Error against an analytical solution or eigenvalues of a learned system, are deterministic for a given model and do not require error bars.

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The paper specifies the hardware used (a single NVIDIA 4070 GPU) at the beginning of Section 3. While specific memory usage and execution times are not detailed, this information is sufficient to understand the computational scale required for the experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research is theoretical and computational, involving synthetic experiments to analyze model properties. It does not involve human subjects, personal data, or other areas that would conflict with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper includes a dedicated "Broader Impacts Statement" section. It discusses potential positive impacts in scientific applications like economics and engineering, as well as potential negative impacts arising from the misapplication of unstable models in safety-critical domains.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: This research is foundational and does not release any data, models, or other assets that would pose a high risk of misuse. The experiments are synthetic and the models are small-scale for illustrative purposes.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [NA]

    Justification: The paper does not use any third-party datasets, models, or specialized codebases that would require explicit licensing information. The experiments are built from first principles based on cited theoretical work, and implementation was done using standard open-source libraries (JAX, Flax).

    Guidelines:

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

13. **New assets**

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

    Answer: [NA]

    Justification: The paper does not introduce or release any new assets, such as datasets, models, or software packages.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: The research does not involve any crowdsourcing or experiments with human subjects.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: The research does not involve any experiments with human subjects, and therefore IRB approval was not required.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: Large Language Models (LLMs) were not used as part of the core research methodology, experimentation, or analysis presented in this paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.