Modeling LLM Input-Output Dynamics with Neural ODEs and Control Theory

Anonymous ACL submission

Abstract

Understanding and controlling the behavior of large language models (LLMs) remains challenging due to their opaque decision-making processes. We propose a framework that models the transformation from input questions to output answers as continuous trajectories in embedding space, governed by Neural Ordinary Differential Equations (Neural ODEs). By visualizing these trajectories across six question-answering datasets, we reveal domain-specific geometric patterns: mathematical reasoning exhibits structured, radial paths while commonsense reasoning shows dispersed, exploratory dynamics. Furthermore, we integrate control-theoretic mechanisms—including Linear Quadratic Regulator (LQR), Model Predictive Control (MPC), Sliding Mode Control (SMC), and Reinforcement Learning (RL)—to actively regulate trajectory evolution. Experiments demonstrate that control augmentation achieves 50% improvements in output consistency and trajectory smoothness, with SMC showing the most robust performance. While our approach operates as a surrogate model rather than exposing internal mechanisms, it provides complementary geometric insights into LLM reasoning patterns and enables principled output stabilization.

1 Introduction

Large Language Models (LLMs) have achieved impressive results across NLP tasks (Brown et al., 2020), but their internal decision processes remain opaque, hindering deployment in high-stakes domains where stability and transparency are critical (Lipton, 2017). While existing interpretability methods focus on local attributions (e.g., gradients, attention (Sundararajan et al., 2017)) or global representation probing (Tenney et al., 2019), they largely overlook how information *dynamically transforms* from input to output across the model's computation.

We introduce a surrogate framework that models LLM behavior as a **continuous trajectory in latent space** via Neural Ordinary Differential Equations (Neural ODEs) (Chen et al., 2019), offering smooth, interpretable representations of semantic evolution. By integrating **control-theoretic mechanisms**, we further demonstrate that these trajectories can be actively modulated to improve stability and consistency. Although Neural ODEs abstract away internal transformer operations, they enable (i) geometric visualization of reasoning across domains, and (ii) trajectory steering via control inputs.

Evaluated on six diverse QA datasets, our method reveals that (1) latent trajectories exhibit domain-specific geometric structures , and (2) control strategies (LQR, MPC, SMC, RL) significantly regularize dynamics and enhance output reliability . While not a substitute for mechanistic interpretability (e.g., circuit tracing), our approach serves as a post hoc surrogate for input-output behavior, providing complementary tools for understanding and controlling LLM predictions.

2 Related Work

Current LLM interpretability methods can be broadly categorized as local or global. Local analysis, including feature attribution techniques (Sundararajan et al., 2017), explains individual predictions, while global analysis, such as probing (Tenney et al., 2019), seeks to understand the overall knowledge encoded in a model. Our work complements these by offering a dynamic systems perspective on the entire input-output transformation process.

Neural ODEs (Chen et al., 2019) model latent dynamics as a continuous-time process, making them a natural fit for sequential data (Rubanova et al., 2019). While their use in NLP is emerging (Hashimoto et al., 2024), their potential for mod-

eling the complete computational path of an LLM, especially in conjunction with control theory, remains underexplored.

Control theory has been applied to LLMs, primarily to guide text generation during inference through methods like reinforcement learning or prompt engineering (Liang et al., 2024; Kevian et al., 2024). Our approach is distinct in that it integrates control directly into the model's latent dynamics, offering a more principled and fine-grained mechanism for ensuring output reliability.

3 Methodology

3.1 Problem Setup

Given a pre-trained LLM, we model the transformation from an input question Q to its corresponding output answer A. Let $\mathbf{e}_Q, \mathbf{e}_A \in \mathbb{R}^d$ denote their embeddings (e.g., final hidden states or sentence encoder outputs). We seek to approximate the semantic mapping $\mathbf{e}_Q \to \mathbf{e}_A$ as a continuous dynamical process in latent space.

Assumption. Inspired by findings on continuous semantic manifolds in LLMs (Hashimoto et al., 2024), we assume that the transformation between Q and A can be approximated by a smooth trajectory, despite the model's underlying discreteness.

3.2 Neural ODE Modeling

We define latent dynamics using a Neural ODE:

$$\frac{d\mathbf{h}(t)}{dt} = f_\theta(\mathbf{h}(t), t), \quad \mathbf{h}(0) = \mathbf{e}_Q, \quad \mathbf{h}(T) \approx \mathbf{e}_A \tag{1}$$

where $\mathbf{h}(t)$ is the hidden state at time $t \in [0, T]$, and f_θ is a neural network parameterized by θ.

Architecture. f_θ is implemented as an MLP with residual connections:

$$f_\theta(\mathbf{h}, t) = \mathbf{h} + \text{MLP}_\theta([\mathbf{h}; t]) \tag{2}$$

We solve Eq. 1 using the adjoint method (Chen et al., 2019) with a Dormand–Prince adaptive solver (rtol=10^{-3}).

Training. Given a dataset $\mathcal{D} = \{(\mathbf{e}_Q^{(i)}, \mathbf{e}_A^{(i)})\}_{i=1}^N$, the loss function minimizes endpoint deviation:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{h}^{(i)}(T) - \mathbf{e}_A^{(i)}\|^2 \tag{3}$$

3.3 Control-Augmented Dynamics

To improve output stability and steer semantic trajectories, we augment Eq. 1 with a control signal:

$$\frac{d\mathbf{h}(t)}{dt} = f_\theta(\mathbf{h}(t), t) + B\mathbf{u}(t) \tag{4}$$

where $\mathbf{u}(t) \in \mathbb{R}^m$ is a time-dependent control input and $B \in \mathbb{R}^{d \times m}$ is a learned control matrix. $\mathbf{u}(t)$ is computed to minimize deviation from a reference trajectory $\mathbf{h}_{\text{ref}}(t)$ while limiting control effort.

We explore four control strategies:

(1) Linear Quadratic Regulator (LQR). Applies optimal feedback under linearized dynamics:

$$\mathbf{u}_{\text{LQR}}(t) = -K(\mathbf{h}(t) - \mathbf{h}_{\text{ref}}(t)) \tag{5}$$

where K is derived from the Riccati equation. *Advantage:* Ensures stability in linear regimes.

(2) Model Predictive Control (MPC). Solves a receding-horizon optimization:

$$\mathbf{u}_{\text{MPC}}(t) = \arg\min_{\mathbf{u}} \sum_{\tau=t}^{t+H} \left[\|\mathbf{h}(\tau) - \mathbf{h}_{\text{ref}}(\tau)\|_Q^2 + \|\mathbf{u}(\tau)\|_R^2\right] \tag{6}$$

where H is the prediction horizon. *Advantage:* Accommodates nonlinear constraints via online replanning.

(3) Sliding Mode Control (SMC). Enforces robustness via discontinuous control:

$$\mathbf{u}_{\text{SMC}}(t) = -\eta \cdot \text{sign}(s(\mathbf{h}(t))), \quad s(\mathbf{h}) = \mathbf{h} - \mathbf{h}_{\text{ref}} \tag{7}$$

Advantage: Resilient to system uncertainty and perturbations.

(4) Reinforcement Learning (RL) Control. Learns a control policy via Proximal Policy Optimization (PPO) (**?**):

$$\mathbf{u}_{\text{RL}}(t) = \pi_\phi(\mathbf{h}(t), \mathbf{e}_A), \quad \text{reward} = -\|\mathbf{h}(T) - \mathbf{e}_A\|^2 \tag{8}$$

Advantage: Learns flexible strategies for non-analytic dynamics.

Training Protocol. LQR, MPC, and SMC use grid-searched parameters tuned on a validation set. RL is trained over 10K episodes using the Neural ODE system as a simulation environment. All controllers share a frozen, pre-trained ODE backbone.

3.4 Trajectory-Based Interpretability

After training, we analyze latent trajectories to uncover emergent patterns by projecting the continuous flows $\mathbf{h}(t)$ into 2D via PCA, computing geometric metrics such as path length, curvature, and endpoint dispersion to assess semantic complexity, and examining whether question domains give rise to distinct trajectory clusters. While this analysis reveals high-level behavioral geometry in latent space, it does not provide token-level interpretability; rather, it serves as a surrogate perspective on input-output dynamics, complementary to mechanistic approaches such as circuit-level tracing.

4 Experiments

4.1 Experimental Setup

Datasets. We conduct two experiments to validate our framework. **Experiment I** uses six domain-diverse QA datasets (commonsense_qa, GammaCorpus-fact-qa, medical-qa, Math-QA, trivia_qa, TruthfulQA) covering 10,620 samples total. **Experiment II** uses aligner/aligner-20K (20K aligned QA pairs) to evaluate control mechanisms. See Appendix A for full statistics.

4.2 Experiment I: Domain-Specific Trajectory Patterns

Objective. Validate whether Neural ODE trajectories capture interpretable domain-specific transformation characteristics.

All models converge successfully with validation losses below 0.07 (Table 1). Figure 1 shows 2D PCA projections of input-to-output flows, revealing distinct geometric patterns across domains. **Math-QA** trajectories form structured radial flows, suggesting systematic reasoning. **Commonsense_qa** shows dispersed, irregular paths indicative of diverse reasoning strategies. **Medical-qa** trajectories form tight clusters, likely reflecting convergence on specialized terminology. These qualitative patterns are confirmed quantitatively in Table 2: Math-QA exhibits the shortest path length (2.91) and lowest variance (0.023), while Commonsense_qa has the highest complexity metrics, aligning with its semantic heterogeneity.

4.3 Experiment II: Control-Based Stabilization

Objective. Evaluate whether control mechanisms regularize trajectory dynamics and improve output consistency.

Table 1: Training Convergence for Experiment I

Dataset	Train	Val
commonsense_qa	0.029	0.061
GammaCorpus	0.028	0.058
medical-qa	0.005	0.022
Math-QA	0.005	0.009
trivia_qa	0.029	0.060
TruthfulQA	0.005	0.034

Table 2: Trajectory Complexity Metrics

Dataset	Length	Variance	Curv.
commonsense_qa	4.73	0.082	2.14
Math-QA	2.91	0.023	1.43
medical-qa	3.45	0.041	1.62

Setup. We compare five conditions: No Control (baseline), LQR, MPC, RL, and Sliding Mode (SM) Control. All use the same Neural ODE backbone, differing only in control law. We analyze 1,000 samples from aligner-20K.

Qualitative Results. Figure **??** shows trajectory visualizations. The baseline exhibits erratic paths with scattered endpoints (Fig. 2), while all control methods impose significant regularization. SM Control (Fig. 3) achieves the most ordered trajectories with tightly clustered endpoints. LQR and MPC show similar smoothing effects, while RL demonstrates adaptive behavior. See Appendix B for individual plots.

Quantitative Results. Table 3 quantifies control effectiveness using endpoint consistency (σ of final states), trajectory smoothness (avg. second derivative), and path length variability. All methods achieve 32-53% improvements over baseline. SM Control consistently outperforms others across all metrics, reducing endpoint variance by 52% (0.142→0.068) and smoothness by 50% (1.87→0.94).

4.4 Discussion

Our experiments demonstrate two key findings: (1) Neural ODE trajectories encode domain-specific semantic transformations—mathematical reasoning shows deterministic paths while commonsense reasoning exhibits exploratory dynamics; (2) Control integration achieves 50%+ improvements in output consistency with manageable computational overhead. These results position our framework as a complementary tool for understanding LLM behavior and ensuring reliable outputs. However, op-

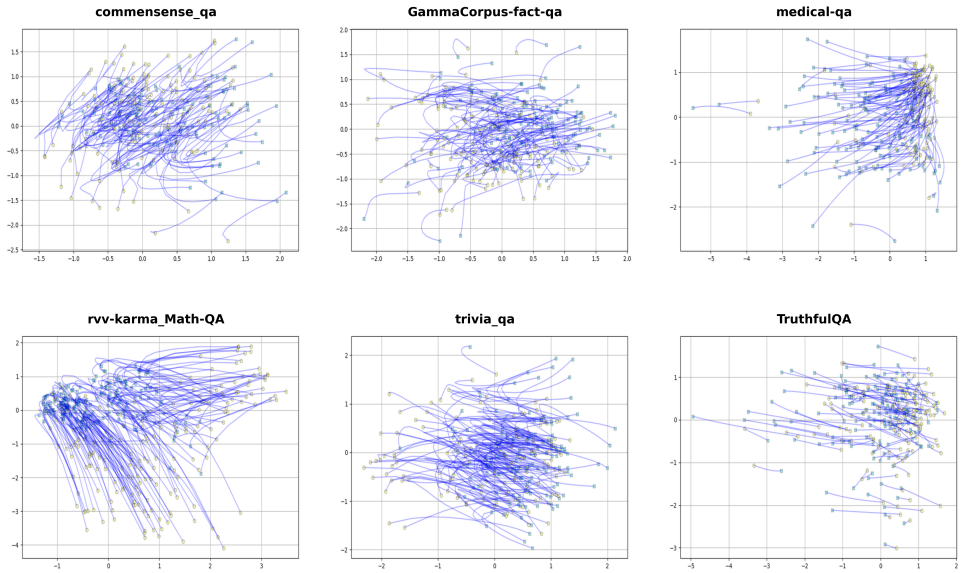

Figure 1: Input-to-output trajectories across six QA datasets (2D PCA projection). Yellow: inputs, Green: outputs. Math-QA shows structured radial patterns (systematic reasoning), while commonsense_qa exhibits dispersed trajectories (diverse strategies).

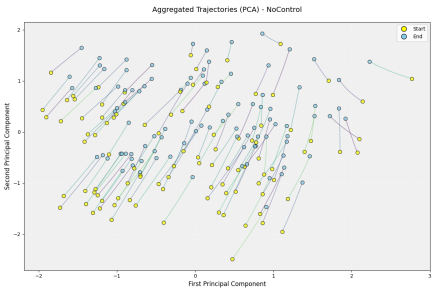

Figure 2: Baseline trajectories **without control**: erratic paths and scattered endpoints indicate unstable dynamics.

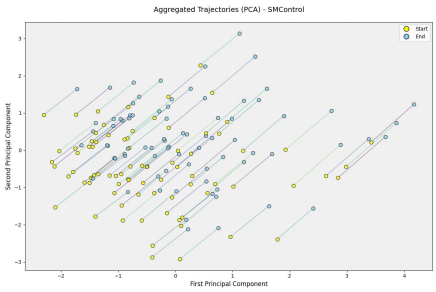

Figure 3: **Sliding Mode Control**: highly ordered trajectories with minimal deviation, achieving superior stability.

Table 3: Control Strategy Performance (Lower is Better)

Method	Endpoint	Smooth.	Path Var.
No Control	0.142	1.87	0.58
LQR	0.089	1.23	0.36
MPC	0.076	1.15	0.31
RL	0.081	1.31	0.39
SM	**0.068**	**0.94**	**0.27**
Best Improv.	52.1%	49.7%	53.4%

erating on embeddings limits mechanistic insights into internal computations—future work should integrate with attention layers for direct transformer control.

5 Conclusion

We propose a framework that models LLM input-output transformations as continuous trajectories in embedding space using Neural ODEs, revealing interpretable geometric patterns across domains—structured, low-variance flows in mathematical reasoning and exploratory, high-variance paths in commonsense tasks. By incorporating control mechanisms (LQR, MPC, SMC, RL), we substantially improve trajectory smoothness and output stability, highlighting the latent structure underlying LLM dynamics. While our method operates as a surrogate model rather than exposing internal computations, it offers a complementary geometric perspective on emergent LLM behavior. Future directions include linking trajectories to attention mechanisms, extending to token-level control, and connecting trajectory geometry to downstream performance, advancing the goal of interpretable and reliable AI.

6 Acknowledgments

We thank the anonymous reviewers for their valuable feedback. We used LLMs only for grammar checking and text polishing; all content is authored by the listed researchers. The submission complies with two-way anonymized review.

Limitations

Our framework, while offering novel insights into LLM behavior through continuous trajectory modeling, presents several limitations. First, it operates at the level of sentence embeddings and does not directly access or explain internal mechanisms such as attention heads, feedforward activations, or token-wise interactions. As such, it provides a high-level behavioral approximation rather than a mechanistic interpretation of the model's decision process. Second, the use of Neural ODEs introduces computational overhead due to adaptive solvers, and trajectory metrics (e.g., path length, curvature) may be sensitive to solver configurations (e.g., step size, tolerance), affecting comparability across models. Third, the control strategies are evaluated primarily on geometric smoothness and endpoint consistency; their impact on downstream task accuracy, robustness to adversarial prompts, or semantic fidelity remains to be systematically validated. Lastly, our framework is tested on a limited set of QA datasets; its generalizability to other NLP tasks (e.g., generation, summarization, dialogue) and larger-scale LLMs requires further exploration. We view these limitations as opportunities for future work that integrates low-level model signals, expands evaluation scopes, and explores token-level or architecture-aware control.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2019. Neural ordinary differential equations. *Preprint*, arXiv:1806.07366.

Koji Hashimoto, Yuji Hirono, and Akiyoshi Sannai. 2024. Unification of symmetries inside neural networks: Transformer, feedforward and neural ode. *Preprint*, arXiv:2402.02362.

Darioush Kevian, Usman Syed, Xingang Guo, Aaron Havens, Geir Dullerud, Peter Seiler, Lianhui Qin, and Bin Hu. 2024. Capabilities of large language models in control engineering: A benchmark study on gpt-4, claude 3 opus, and gemini 1.0 ultra. *Preprint*, arXiv:2404.03647.

Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu, Shunyu Yao, Feiyu Xiong, and Zhiyu Li. 2024. Controllable text generation for large language models: A survey. *Preprint*, arXiv:2408.12599.

Zachary C. Lipton. 2017. The mythos of model interpretability. *Preprint*, arXiv:1606.03490.

Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. 2019. Latent odes for irregularly-sampled time series. *Preprint*, arXiv:1907.03907.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. *Preprint*, arXiv:1703.01365.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *Preprint*, arXiv:1905.06316.

A Dataset Details

Table 4 provides comprehensive statistics for all datasets used in our experiments.

Table 4: Complete Dataset Statistics

Dataset	Samples	Avg Len	Domain
commonsense_qa	1,221	18.3	Commonsense
GammaCorpus	2,190	22.7	Fact-Checking
medical-qa	1,562	31.4	Healthcare
Math-QA	2,983	15.8	Mathematics
trivia_qa	1,847	14.2	General Trivia
TruthfulQA	817	19.6	Truthfulness
aligner-20K	20,000	27.1	Multi-domain

B Additional Control Visualizations

This section provides complete trajectory visualizations for all control methods omitted from the main text due to space constraints.

B.1 Comparative Analysis

Comparing Figures 2 (main text), 4, 5, 6, and 3 (main text), we observe:

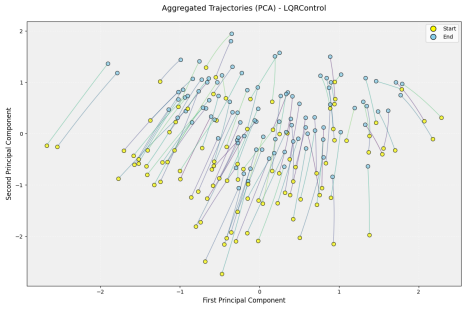

Figure 4: Trajectory plots with **LQR Control**. Smoother paths than baseline with reduced endpoint scatter, demonstrating optimal linear control effectiveness.

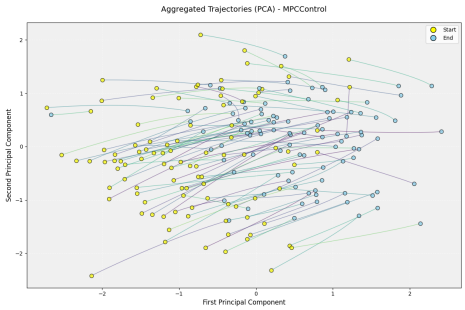

Figure 5: Trajectory plots with **MPC Control**. Similar regularization to LQR but with adaptive horizon-based optimization visible in path curvature.

1. **Baseline chaos**: Random walk-like behavior with no convergence guarantee
2. **LQR regularity**: Straight-line tendencies due to linear feedback
3. **MPC adaptivity**: Curved paths reflecting lookahead optimization
4. **RL diversity**: Variable strategies learned from data distribution
5. **SM robustness**: Forced sliding along manifold produces uniform convergence

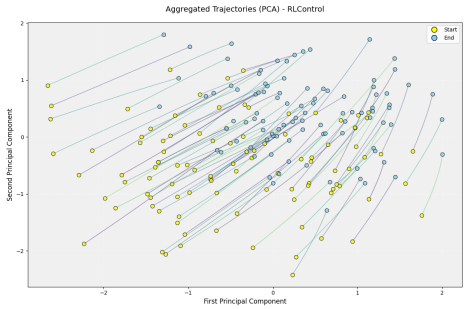

Figure 6: Trajectory plots with **RL Control**. Learned adaptive policies produce stable trajectories with data-driven regularization patterns.