Analytical Lyapunov Function Discovery: An RL-based Generative Approach

Haohan Zou[1] **Jie Feng**[1] **Hao Zhao**[2] **Yuanyuan Shi**[1]
[1]University of California San Diego
[2]École Polytechnique Fédérale de Lausanne (EPFL)
`{hazou, jif005, yyshi}@ucsd.edu, hao.zhao@epfl.ch`

Abstract

We propose an end-to-end framework using transformers to construct analytical (local) Lyapunov functions for addressing key challenges in current neural network-based approaches, namely scalability and interpretability. Our framework includes a transformer-based generator, which proposes candidate Lyapunov functions, and a falsifier that validates these candidates. The model is updated via risk-seeking policy gradient. We demonstrate the efficiency of our approach on a range of nonlinear dynamical systems with up to ten dimensions and show that it can discover Lyapunov functions not previously identified in the control literature. This work has been accepted by International Conference on Machine Learning 2025. Full implementation is available on Github.

1 Introduction

A Lyapunov function is an energy-like function used to certify the stability of dynamical systems. A sufficient condition for stability is that the Lyapunov function decreases along system trajectories. It is also central to controller design, offering formal guarantees of closed-loop stability and robustness [Khalil, 2002]. However, designing Lyapunov functions for nonlinear systems has long been considered more of an 'art' than a science, even for stable dynamics, due to its inherent complexities.

Existing techniques for Lyapunov function construction can be categorized into two kinds. The computational approaches, such as sum-of-squares (SOS) methods [Papachristodoulou and Prajna, 2005a,b, Ahmadi and Majumdar, 2016, Dai and Permenter, 2023], formulate the task as an optimization problem. However, the restriction of the candidates' form (e.g., polynomial formula) impedes its applicability to general nonlinear dynamics for real-world problems [Papachristodoulou and Prajna, 2005a,b]. The neural network-based approaches [Chang et al., 2019, Zhou et al., 2022, Wu et al., 2023, Edwards et al., 2024, Yang et al., 2024] parametrize the Lyapunov function as a neural network and optimize parameters with respect to Lyapunov conditions. Due to the black-box nature of NN, these methods offer limited insights into the system's dynamical behavior, and over-parameterization and nonlinear activations complicate the condition verification, leading to scalability issues.

In this work, we aim to address the following question: *Can neural networks effectively discover valid analytical Lyapunov functions directly from complex system dynamics?* To tackle this challenge, we introduce an end-to-end framework designed to find analytical Lyapunov functions for nonlinear dynamical systems given in analytical form. Building on the transformer's ability to model complex dependencies [Vaswani et al., 2017] and the success of deep symbolic regression methods [Holt et al., 2023], our framework deploys a symbolic transformer [Kamienny et al., 2022] for Lyapunov function discovery. Given the lack of high-quality (local) Lyapunov function datasets, particularly for high-dimensional systems, we propose a reinforcement learning (RL)-based approach to search for Lyapunov functions on a per-system basis, instead of pre-training like Alfarano et al. [2024].

Lyapunov conditions are verified by localized sampling in the neighborhoods of minimizers of candidate expressions, and identified counterexamples are incorporated into the training set for further optimization. We demonstrate the efficiency of our method on various systems, including non-polynomial dynamics like the pendulum, quadrotor, and power system frequency control. Notably, our approach scales to a 10-D system and discovers a valid local Lyapunov function for power system frequency control with lossy transmission lines, which is previously unknown in the literature.

2 Preliminary

Our framework searches for analytical Lyapunov functions of autonomous nonlinear dynamics at an equilibrium point. Without loss of generality, we assume the origin to be the equilibrium point.

Definition 1 (Dynamical systems)**.** An n-D autonomous nonlinear dynamical system is given by

$$\frac{dx}{dt} = f(x)\,, x(0) = x_0, \tag{1}$$

where $f : \mathcal{D} \to \mathbb{R}^n$ is a Lipschitz-continuous vector field, and $\mathcal{D} \subseteq \mathbb{R}^n$ is a set containing the origin that defines the state space. Each $x(t) \in \mathcal{D}$ is a state vector.

Definition 2 (Asymptotic stability)**.** A system of (1) is stable at the origin if $\forall\, \epsilon > 0$, there exists $\delta = \delta(\epsilon) > 0$ such that $\|x(0)\| < \delta \implies \|x(t)\| < \epsilon, \forall\, t \geq 0$. The origin is asymptotically stable if it is stable and δ can be chosen such that $\|x(0)\| < \delta \implies \lim_{t\to\infty} x(t) = 0$.

Definition 3 (Lie derivative)**.** The Lie derivative of a continuously differentiable scalar function $V : \mathcal{D} \to \mathbb{R}$ along the trajectory of (1) is given by

$$L_f V(x) = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i}\frac{dx_i}{dt} = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} f_i(x). \tag{2}$$

Proposition 1 (Lyapunov functions for asymptotic stability)**.** *Let* $x = 0$ *be an equilibrium point for* (1) *and* $\mathcal{D} \subseteq \mathbb{R}^n$ *be a domain containing the* $x = 0$*. Let* $V : \mathcal{D} \to \mathbb{R}$ *be a continuously differentiable function such that*

$$V(0) = 0 \textit{ and } V(x) > 0 \textit{ in } D\backslash\{0\}, \tag{3a}$$

$$L_f V(x) < 0 \textit{ in } D\backslash\{0\}, \tag{3b}$$

then the origin is asymptotically stable.

Definition 4 (Lyapunov risk)**.** Consider a candidate Lyapunov function $\tilde{V}$ for system $f(x)$ from Definition 1. For a dataset $\mathcal{X} = \{x_1, \cdots, x_N\}$ where $x_i \in \mathcal{D}$, the Lyapunov risk of $\tilde{V}$ Chang et al. [2019] over $\mathcal{D}$ is defined by

$$\mathcal{L}(\tilde{V}) = \frac{1}{N}\sum_{i=1}^{N}\Big(\max(0, L_f\tilde{V}(x_i)) + \max(0, -\tilde{V}(x_i))\Big)\,. \tag{4}$$

3 Proposed Method

Our framework consists of three components: 1) a symbolic transformer for candidate analytical Lyapunov functions generation, 2) a numerical verifier employing the Simplicial Homology Global Optimization (SHGO) [Endres et al., 2018] for Lyapunov conditions' checking, and 3) a risk-seeking policy gradient algorithm optimizing the transformer's parameters based on candidate expressions' rewards. Implementation details of these parts are in Appendix A. We denote $\mathcal{X} \subseteq \mathcal{D}$ as the training set for reward calculation.

Candidate Lyapunov Function Generation. We use a symbolic transformer model, parametrized as ϕ, as the backbone. The transformer takes a dynamical system $f(x)$ as input and aims to generate candidate analytical Lyapunov functions $\tilde{V}_\phi$ such that: $\tilde{V}_\phi(x) > 0$ and $L_f\tilde{V}_\phi < 0, \forall\, x \in \mathcal{D}\backslash\{0\}$. Following the representation rules in Lample and Charton [2020], our framework represents symbolic transformer models' input and output as sequences of symbolic tokens. In each epoch, we sample a batch of Q candidates $\tilde{\mathcal{V}}_\phi = \{\tilde{V}_\phi^i \sim p(\tilde{V}_\phi|\phi, f(x))\}_{i=1}^{Q}$, which will be verified according to the Lyapunov conditions and guide the optimization direction of parameters ϕ for further generation.

Verification and Falsification Feedback. SHGO is a constrained global optimization algorithm with theoretical guarantees for convergence (Proposition 2). Taking advantage of the theoretical results, we propose a global-optimization-based verification algorithm using SHGO that effectively checks the Lyapunov conditions and feedbacks counterexamples for further training. For a candidate $\tilde{V}_\phi$, SHGO is first applied to identify minimizers x_1^* and x_2^* of $\tilde{V}_\phi$ and $-L_f\tilde{V}_\phi$ in the state space $\mathcal{D}$. These minimizers highlight the regions where Lyapunov conditions are most likely to fail. Next, data points x from neighborhoods around these minimizers, $\mathcal{B}_r(x_1^*)$ and $\mathcal{B}_r(x_2^*)$ for some $r \ll \|\mathcal{D}\|_2$, are sampled to check Lyapunov conditions, i.e. $\tilde{V}_\phi(x) > 0$ and $-L_f\tilde{V}_\phi(x) > 0$ for $x \in \mathcal{D}\backslash\{0\}$. This localized sampling scheme effectively identifies violations within $\mathcal{D}$. Identified counterexamples $\mathcal{X}_{ce}$ are added to the training set $\mathcal{X}$ for reward calculations. Once a candidate Lyapunov function passes this verification process and does not encounter any violation in $\mathcal{X}$, it indicates a numerically valid solution is found, pending the final formal verification by SMT solver.

Proposition 2 (Convergence Gurantees of SHGO [Endres et al., 2018])**.** *For a given continuous objective function f that is adequately sampled by a sampling set of size N_s. If the size of the minimizer pool $\mathcal{M}$ extracted from the directed simplex (a convex polyhedron) $\mathcal{H}$ is $|\mathcal{M}|$. Then any further increase of the sampling size N_s will not increase $|\mathcal{M}|$.*

Risk-Seeking Policy Gradient. We define the proposed Lyapunov risk reward as:

$$R(\tilde{V}_\phi) = \frac{1}{1 + \mathcal{L}(\tilde{V}_\phi)}, \tag{5}$$

where the empirical Lyapunov risk $\mathcal{L}(\tilde{V}_\phi)$ in Equation (4) is measured over training set $\mathcal{X}$. This reward quantifies the violation degree of Lyapunov conditions over $\mathcal{X}$. Since the violation measure for $\tilde{V}_\phi$ is non-differentiable with respect ϕ, we employ the risk-seeking policy gradient to update ϕ end-to-end. In our task, final performance depends on finding *at least* one valid Lyapunov function fulfilling the conditions in Proposition 1, so standard policy gradient methods, which maximize the expected reward for candidates generated from current parameters, misalign with our objective. Instead, we adopt risk-seeking policy gradient [Petersen et al., 2020]. Let $R_\alpha(\phi)$ as the $1-\alpha$ quantile of the distribution of rewards of sampled candidates under the current policy ϕ. The learning objective of risk-seeking policy gradient, parameterized by α, is formulated as:

$$J_{\text{risk}}(\phi, \alpha) = \mathbb{E}_{\tilde{V}_\phi \sim p(\tilde{V}_\phi|\phi, f(x))}\left[-R(\tilde{V}_\phi) \mid R(\tilde{V}_\phi) \geq R_\alpha(\phi)\right]. \tag{6}$$

This objective aims to optimize only rewards of high-quality candidates from the top $1-\alpha$ quantile.

Automated Expert Guidance. Inspired by Mundhenk et al. [2021], we incorporate a Genetic Programming (GP) component [Fortin et al., 2012] into the training paradigm to further enhance the training efficiency. In each epoch, we feed $\{\tilde{V}_\phi^i \sim p(\tilde{V}_\phi|\phi, f(x))\}_{i=1}^Q$ into the GP module, which refines these expressions through evolutionary operations with respect to $R(\tilde{V}_\phi)$. We select an 'elite set' of the refined expressions $\tilde{\mathcal{V}}_{gp} = \{\tilde{V}_{gp}^i \sim \text{GP}(\tilde{V}_\phi)\}_{i=1}^G$, regard them as target expressions, and optimize the transformer with the following expert guidance loss:

$$\mathcal{L}(\tilde{\mathcal{V}}_{gp}) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{k_i}R(\tilde{V}_{gp}^i)\sum_{j=1}^{k_i} -\log\left(p(\tilde{V}_{gp_j}^i|\tilde{V}_{gp_{1:(j-1)}}^i, \phi, f(x))\right), \tag{7}$$

where G is the number of expressions in $\tilde{\mathcal{V}}_{gp}$, and k_i is the complexity (number of symbolic tokens in the pre-order traversal) of $\tilde{V}_{gp}^i$. The GP solutions explore the characteristics of Lyapunov functions that have not been captured by the transformer yet and effectively guide the transformer.

4 Experiment Results

We validate our algorithm across a variety of nonlinear dynamics. The test dynamics fall into two kinds: 1). Polynomial Systems, and 2). Non-polynomial Systems, where detailed information is summarized in Appendices E & F. *dReal* [Gao et al., 2013] SMT solver is used for final verification of found Lyapunov functions. We compare against four baseline algorithms on our test dynamics. *Neural methods*: 1) Augmented Neural Lyapunov Control (ANLC) [Grande et al., 2023], 2) FOSSIL 2.0 [Edwards et al., 2024]. *Analytical methods*: 3) the transformer-based global Lyapunov search of Alfarano et al. [2024], 4) SOS methods via SOSTOOLS (Matlab) [Papachristodoulou et al., 2013].

Table 1: Comparison between ours and neural baselines. Succ % is the successful rate of finding a valid Lyapunov function in 5 hours out of 5 random seeds. Runtime is the average training time for a successful trial.

Frameworks	2-D Dynamics Succ %	2-D Dynamics Runtime	3-D Dynamics Succ %	3-D Dynamics Runtime	6-D Dynamics Succ %	6-D Dynamics Runtime	8-D Dynamics (App. E.5) Succ %	8-D Dynamics (App. E.5) Runtime
Ours	**100**	165s	**100**	124s	**80**	6290s	**80**	14358s
ANLC	53.3	**1.409s**	46.7	**63.91s**	0	-	0	-
FOSSIL 2.0	80	7.708s	66.7	221s	0	-	0	-

Performance Analysis. Table 5 summarizes the runtime, success rate, and discovered Lyapunov functions for a selection of tested nonlinear systems, ranging from 2-D to 10-D, demonstrating the robustness and scalability of our framework. Unlike existing methods that produce neural Lyapunov functions, our framework yields interpretable *analytical* candidates. For example, it correctly identifies the energy function as a valid Lyapunov function for the simple pendulum. Likewise, for the 3-bus power system, it discovers the commonly used energy-based storage function for incremental [Weitenberg et al., 2018]. The simplicity of analytical formulations allow SMT solvers to efficiently verify Lyapunov conditions over the identified expressions.

Newly Discovered Lyapunov Function. The lossy frequency dynamics in power systems [Chiang, 1989, Cui and Zhang, 2022] is a system that lacks a valid Lyapunov function to directly certify its stability. In this paper, we focus on a 2-bus (4-D) lossy system with the equilibrium point at the origin, with detailed descriptions in Appendix F.5. Using the proposed method, we successfully discovered two local Lyapunov functions valid in the considered region $\mathcal{D} = \{(\delta_1, \delta_2, \omega_1, \omega_2) \in \mathbb{R}^4 \mid |\delta_i| \leq 0.75 \text{ and } |\omega_i| \leq 2 \text{ for } i = 1, 2\}$:

$$V_1(\delta_1, \delta_2, \omega_1, \omega_2) = \sum_{i=1}^{2} \omega_1^2 + \Big(\sin(\delta_2) - \sin(\delta_1) + \omega_2\Big)^2, \quad V_2(\delta_1, \delta_2, \omega_1, \omega_2) = \sum_{i=1}^{2} \omega_1^2 + \Big(\sin(\delta_2) - \sin(\delta_1) - \omega_1\Big)^2.$$

To the best of our knowledge, these are the first analytical Lyapunov functions used to certify the local stability of a 2-bus lossy power system.

Neural Lyapunov Function Baselines. Table 1 compares ours with two neural methods. While the baselines train faster on low-dimensional systems, their restricted small networks fail to converge on harder tasks (e.g., the pendulum and 3-D Trig), yielding lower success rates overall. As dimensionality increases, verification of lie derivative becomes a bottleneck for neural approaches. Even networks with fewer than 15 neurons per layer requires hours for formal checks. In contrast, our method scales robustly to 8-D systems: its analytical forms enable fast numerical verification and allow SMT solvers to certify candidates within milliseconds through term cancellations and algebraic restructuring.

Analytical Lyapunov Function Baselines. Solely trained on globally stable systems with fewer than 6 states, the pre-trained model of Alfarano et al. [2024] can only produce valid Lyapunov functions for the low-dimensional examples in Appendices E.1, E.2, E.3, & F.1, which are globally stable. For polynomial systems, as shown in Table 2, though SOS is more efficient for low-dimensional examples, it fails on higher-dimensional locally stable cases (Appendices E.4, E.5, & E.6), where additional Lie derivative constraints render verification intractable. For non-polynomial dynamics, we applied the recasting scheme [Papachristodoulou and Prajna, 2005b], replacing non-poly with auxiliary variables linked by extra (in)equalities. While SOS recovered certificates for the pendulum and 3-D trig systems, runtimes were far higher than ours (over one hour vs. 157s for 3-D trig dynamics shown in Table 3). This approach suffers from three major drawbacks - heavy reliance on domain expertise, significant computational overhead, and the ability to certify only stability, not asymptotical stability — limiting its practicality for high-dimensional, non-polynomial systems.

Table 2: Training/solving time of ours and sum-of-squares (SOS) on polynomial systems, averaged over successful trials. The stable regions (local or global) of the considered dynamics are indicated in smaller font.

Test Systems	2-D Systems (App. E.1, App. E.2)	3-D Systems (App. E.3)	6-D System (App. E.4)	8-D System (App. E.5)	10-D System (App. E.6)
Ours	97s	108s	**1667s**	**14358s**	**64223s**
SOS	**0.765s**	**1.503s**	-	-	-

Table 3: Training/solving time of ours and SOS on two non-poly systems.

Test Systems	App. F.1	App. F.2
Ours	288s	**157s**
SOS	**19.58s**	6163s

5 Conclusion

We present an end-to-end framework for discovering analytical Lyapunov functions for nonlinear dynamics. Symbolic transformer, trained with a risk-seeking policy gradient and enhanced by genetic programming, proposes candidate expressions; the SHGO optimizer verifies candidates and generates counterexamples during training; and an SMT solver certifies the final candidates. The method scales to 10-D dynamics and has discovered previously unknown Lyapunov functions for lossy power systems, with potential extension to other certificate functions such as control barrier functions.

Acknowledgments and Disclosure of Funding

The authors thank Dr. Huan Zhang at the University of Illinois Urbana-Champaign for helpful discussions. J. Feng is supported by the UC–National Laboratory In-Residence Graduate Fellowship L24GF7923. Y. Shi acknowledges support from NSF ECCS-2200692, DOE grant DE-SC0025495, and the Schmidt Sciences AI2050 Early Career Fellowship.

References

Amir Ali Ahmadi and Anirudha Majumdar. Some applications of polynomial optimization in operations research and real-time decision making. *Optimization Letters*, pages 709–729, 2016.

Alberto Alfarano, Francois Charton, and Amaury Hayat. Global lyapunov functions: a long-standing open problem in mathematics, with symbolic transformers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=kOMrm4ZJ3m`.

Zachary Bastiani, Mike Kirby, Jacob Hochhalter, and Shandian Zhe. Complexity-aware deep symbolic regression with robust risk-seeking policy gradients, 2024. URL `https://openreview.net/forum?id=krJ73n4Pma`.

S. Bouabdallah, P. Murrieri, and R. Siegwart. Design and control of an indoor micro quadrotor. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 5, pages 4393–4398 Vol.5, 2004. doi: 10.1109/ROBOT.2004.1302409.

Ya-Chien Chang, Nima Roohi, and Sicun Gao. Neural lyapunov control. *Advances in neural information processing systems*, 32, 2019.

H.-D. Chiang. Study of the existence of energy functions for power systems with losses. *IEEE Transactions on Circuits and Systems*, 36(11):1423–1429, 1989. doi: 10.1109/31.41298.

Wenqi Cui and Baosen Zhang. Lyapunov-regularized reinforcement learning for power system transient stability. *IEEE Control Systems Letters*, 6:974–979, 2022. doi: 10.1109/LCSYS.2021.3088068.

Wenqi Cui, Yan Jiang, and Baosen Zhang. Reinforcement learning for optimal primary frequency control: A lyapunov approach. *IEEE Transactions on Power Systems*, 38(2):1676–1688, 2023. doi: 10.1109/TPWRS.2022.3176525.

Hongkai Dai and Frank Permenter. Convex synthesis and verification of control-lyapunov and barrier functions with input constraints. In *2023 American Control Conference (ACC)*, pages 4116–4123. IEEE, 2023.

Alec Edwards, Andrea Peruffo, and Alessandro Abate. Fossil 2.0: Formal certificate synthesis for the verification and control of dynamical models. In *Proceedings of the 27th ACM International Conference on Hybrid Systems: Computation and Control*, pages 1–10, 2024.

Stefan C. Endres, Carl Sandrock, and Walter W. Focke. A simplicial homology algorithm for lipschitz optimisation. *Journal of Global Optimization*, 72(2):181–217, 2018. doi: 10.1007/s10898-018-0645-y.

Jie Feng, Wenqi Cui, Jorge Cortés, and Yuanyuan Shi. Online event-triggered switching for frequency control in power grids with variable inertia. *IEEE Transactions on Power Systems*, pages 1–13, 2024. doi: 10.1109/TPWRS.2024.3523262.

Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner Gardner, Marc Parizeau, and Christian Gagné. Deap: Evolutionary algorithms made easy. *The Journal of Machine Learning Research*, 13(1):2171–2175, 2012.

Sicun Gao, Soonho Kong, and Edmund M. Clarke. dreal: An smt solver for nonlinear theories over the reals. In Maria Paola Bonacina, editor, *Automated Deduction – CADE-24*, pages 208–214, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

Davide Grande, Andrea Peruffo, Enrico Anderlini, and Georgios Salavasidis. Augmented Neural Lyapunov Control. *IEEE Access*, 11:67979–67986, 2023. doi: 10.1109/ACCESS.2023.3291349.

Lars Grüne. Computing lyapunov functions using deep neural networks. *Journal of Computational Dynamics*, 8, 01 2019. doi: 10.3934/jcd.2021006.

Samuel Holt, Zhaozhi Qian, and Mihaela van der Schaar. Deep generative symbolic regression. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=o7koEEMA1bR`.

Pierre-Alexandre Kamienny, Stéphane d'Ascoli, Guillaume Lample, and Francois Charton. End-to-end symbolic regression with transformers. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=GoOuIrDHG_Y`.

Hassan K Khalil. *Nonlinear systems*. Prentice Hall, 3nd edition, 2002.

John R Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, volume 1. MIT press, 1992.

Guillaume Lample and François Charton. Deep learning for symbolic mathematics. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1eZYeHFDS`.

Terrell N. Mundhenk, Mikel Landajuela, Ruben Glatt, Claudio P. Santiago, Daniel faissol, and Brenden K. Petersen. Symbolic regression via deep reinforcement learning enhanced genetic programming seeding. In *Advances in Neural Information Processing Systems*, 2021.

A. Papachristodoulou and S. Prajna. A tutorial on sum of squares techniques for systems analysis. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 2686–2700 vol. 4, 2005a. doi: 10.1109/ACC.2005.1470374.

A. Papachristodoulou, J. Anderson, G. Valmorbida, S. Prajna, P. Seiler, and P. A. Parrilo. *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*. `http://arxiv.org/abs/1310.4716`, 2013. Available from `http://www.cds.caltech.edu/sostools`.

Antonis Papachristodoulou and Stephen Prajna. *Analysis of Non-polynomial Systems Using the Sum of Squares Decomposition*, page 23–43. Springer, Aug 2005b. ISBN 978-3-540-23948-2. doi: 10.1007/10997703_2.

Brenden K Petersen, Mikel Landajuela Larma, Terrell N Mundhenk, Claudio Prata Santiago, Soo Kyung Kim, and Joanne Taery Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

Erieke Weitenberg, Claudio De Persis, and Nima Monshizadeh. Exponential convergence under distributed averaging integral frequency control. *Automatica*, 98:103–113, 2018.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Junlin Wu, Andrew Clark, Yiannis Kantaros, and Yevgeniy Vorobeychik. Neural lyapunov control for discrete-time systems. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Lujie Yang, Hongkai Dai, Zhouxing Shi, Cho-Jui Hsieh, Russ Tedrake, and Huan Zhang. Lyapunov-stable neural control for state and output feedback: A novel formulation. In *Forty-first International Conference on Machine Learning*, 2024. URL `https://openreview.net/forum?id=3xPMW9JURD`.

Ruikun Zhou, Thanin Quartz, Hans De Sterck, and Jun Liu. Neural lyapunov control of unknown nonlinear systems with stability guarantees. In *Advances in Neural Information Processing Systems*, 2022.

A Implementation Details of Framework

The proposed framework, visualized in Figure 1, consists of three components: 1) a symbolic transformer for candidate analytical Lyapunov functions generation, where ζ and θ denote the parameters of encoder and decoder, 2) a numerical verifier employing the SHGO [Endres et al., 2018] global optimization algorithm for Lyapunov conditions' checking (Proposition 1) and counterexamples' feedback, and 3) a risk-seeking policy gradient algorithm optimizing the transformer's parameters based on candidate Lyapunov functions' rewards. To tackle the challenges posed by the exponentially growing search space of complex, high-dimensional systems, our framework integrates Genetic Programming as expert guidance to improve expression quality and training efficiency. We denote $\mathcal{X} \subseteq \mathcal{D}$ as the training set for reward calculation. Algorithm 1 summarizes the overall training process.

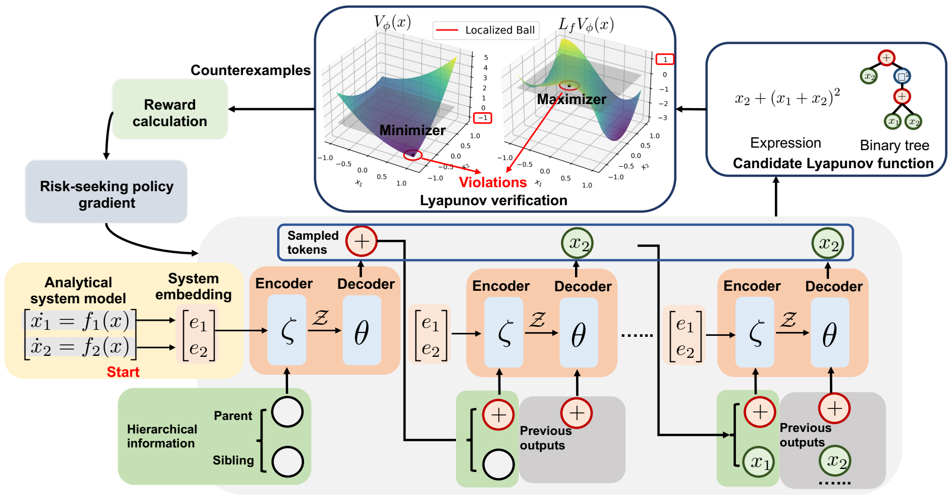

Figure 1: **Framework overview**: The transformer takes embeddings of the dynamical system model as input and generates candidate Lyapunov functions in an autoregressive manner. Hierarchical information is deployed to enhance the model input. For example, when generating the last token x_2, its parent is $+$, and its sibling is x_1. The output is the pre-order traversal of the expression's binary tree. Candidates are verified using a global-optimization-based verification process, with counterexamples added to the training set for reward calculation. The transformer is updated via the risk-seeking policy gradient. The program terminates once a valid expression is found.

A.1 Symbolic Transformer Model

We outline the details of our symbolic transformer model, a conditional generator for analytical candidate expression generation, comprising two components: 1) an encoder, and 2) a decoder. $\phi = \{\zeta, \theta\}$ denotes the transformer parameters, where ζ and θ denote the parameters of encoder and decoder respectively.

Expression Representation. Inspired by the deep symbolic regression frameworks, we use a symbolic transformer model as the backbone. The transformer takes a dynamical system $f(x)$ as input and generates candidate analytical Lyapunov functions $\tilde{V}_\phi$ such that: $\tilde{V}_\phi(x) > 0$ and $L_f \tilde{V}_\phi < 0, \forall\, x \in \mathcal{D} \backslash \{0\}$. Following the expression representation rules in Lample and Charton [2020], our framework represents symbolic transformer models' input and output as sequences of symbolic tokens. Each mathematical expression can be converted into a symbolic expression tree, a binary tree where internal nodes are symbolic operators and terminal nodes (leaves in the tree) are variables or constants. Symbolic operators can be either unary (i.e., one child), such as sin, cos, or binary (i.e., two children), such as $+, \times$. Furthermore, each symbolic expression tree can be represented as a sequence of node values, either symbolic tokens or numerical coefficients, by its pre-order traversal (i.e., first visiting the parent, then traversing the left child and right child). In this way, each expression obtains a pre-order traversal representation, which can uniquely reconstruct the original expression [Petersen et al., 2020].

Algorithm 1 Training Framework for Analytical Lyapunov Function Discovery via Reinforcement Learning

Input: Dynamics $f(x)$, state space $\mathcal{D}$, quantile α, symbolic library $\mathcal{L}$, batch size Q, max complexity k, radius r.
Output: Valid Lyapunov function $\mathcal{V}^*$ for system $f(x)$.

```
 1: Initialize the conditional generator with parameters $\phi$,
 2: Randomly sample training datapoints $\mathcal{X} = \{x_1, \cdots, x_N\}$ where $x_i \in \mathcal{D}$,
 3: **while** no valid candidates found **do**
 4:     $\tilde{\mathcal{V}}_\phi \leftarrow \{\tilde{V}_\phi^i \sim p(\tilde{V}_\phi | \phi, f(x))\}_{i=1}^Q$,
 5:     $\tilde{\mathcal{V}}_{gp} \leftarrow$ Genetic Programming$(\tilde{\mathcal{V}}_\phi)$,
 6:     $\tilde{\mathcal{V}} \leftarrow \tilde{\mathcal{V}}_\phi \cup \tilde{\mathcal{V}}_{gp}$,
 7:     $\mathcal{V}^*, \mathcal{X}_{ce} \leftarrow$ verification$(\tilde{\mathcal{V}}, r, \mathcal{D})$, {Verify candidates $\tilde{\mathcal{V}}$, return the valid Lyapunov function $\mathcal{V}^*$
        (if any) and counterexamples $\mathcal{X}_{ce}$. Details in App. A.2. }
 8:     **if** $\mathcal{V}^*$ is not empty **then**
 9:         **Return** $\mathcal{V}^*$.
10:     **end if**
11:     $\mathcal{R} \leftarrow \{R(\tilde{V}^i) \ \forall \ \tilde{V}^i \in \tilde{\mathcal{V}}\}$,
12:     $R_\alpha(\phi) \leftarrow (1-\alpha)$-quantile of $\mathcal{R}$,
13:     $\phi \leftarrow \phi - \nabla_\phi J_{\text{risk}}(\phi, \alpha)$ , {risk-seeking policy gradient update. Equation (6) }
14:     $\phi \leftarrow \phi - \nabla_\phi \mathcal{L}(\tilde{\mathcal{V}}_{gp})$ , {expert guidance loss based on the genetic programming refined
        Lyapunov functions $\tilde{\mathcal{V}}_{gp}$ to update policy. Equation (7).}
15:     Concatenate counterexamples $\mathcal{X}_{ce}$ to dataset $\mathcal{X}$.
16: **end while**
```

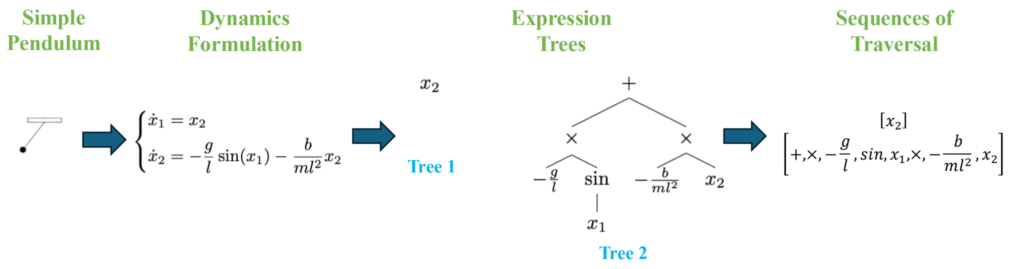

Figure 2: We visualize the dynamics tokenization process of the simple pendulum system. Each analytical formula in the ODE representation of input dynamics is first converted into an expression tree and then represented by the pre-order traversal in symbolic tokens and constant coefficients.

Encoder Structure. Our framework employs the vanilla transformer encoder from Vaswani et al. [2017] to encode two types of input information: 1) the input system dynamics $f(x)$, encoded into a latent vector $\mathcal{F} \in \mathbb{R}^p$, where $p \in \mathbb{R}$, and 2) the hierarchical tree state representation [Petersen et al., 2020] of the selected tokens, introduced below, encoded as a latent vector $\mathcal{W} \in \mathbb{R}^k$, where $k \in \mathbb{R}$. The resulting representations are concatenated as $\mathcal{Z}$. Both inputs are expressed as sequences of symbolic tokens and numerical coefficients when fed into the encoder. The numerical coefficients are tokenized in two schemes: an integer is represented as a sequence of digits in base $b = 10$ (e.g. 123 is tokenized as $[1, 2, 3]$), and a real number is represented in scientific notation rounded to 4 significant digits (e.g. 3.14 is tokenized as $[3, 1, 4, 0, 10^0]$). For damped pendulum example in Figure 2, suppose we have $m = 0.5\,\text{kg}$, $l = 1\,\text{m}$, $g = 9.81$, and $b = 0.1$, the dynamics can be tokenized as: $[\texttt{SOS}, x_2, \texttt{EOS}, \texttt{SOS}, +, \times, -, 9, 8, 1, 0, 10^0, \sin, x_1, \times, -, 2, 0, 0, 0, 10^{-1}, x_2, \texttt{EOS}]$. In this work, we set the embedding dimension to 128, attention head to 2, and applied a 2-layer transformer encoder for system dynamics $f(x)$ encoding and a 3-layer transformer encoder for hierarchical tree state representation encoding.

Decoder Structure. The decoder of the symbolic transformer model also uses the vanilla transformer decoder, with an additional linear layer to output the token probability ψ over the symbolic library $\mathcal{L}_s$ for token selection. Candidate Lyapunov functions $\tilde{V}_\phi$ are sampled as sequences of symbolic tokens in pre-order traversal. Each symbolic token $\tilde{V}_{\phi_i}$ is sampled autogressively from conditional distribution $p(\tilde{V}_{\phi_i}|\tilde{V}_{\phi_{1:(i-1)}}, \phi, f(x))$. Since analytical expression in its pre-order traversal is inherently hierarchical, we also deploy the hierarchical tree state representation [Petersen et al., 2020, Holt et al., 2023]. This method enhances the decoder input by concatenating the representations of the parent and sibling nodes with previously selected outputs and the dynamics. Upon token sampling is complete for $\tilde{V}_\phi$, we subtract $\tilde{V}_\phi(0)$ from the candidate expression to enforce the Lyapunov condition $\tilde{V}_\phi(0) = 0$. In each epoch, a batch of candidate Lyapunov functions $\{\tilde{V}_\phi^i \sim p(\tilde{V}_\phi|\phi, f(x))\}_{i=1}^{Q}$ is sampled as candidates, which are verified by global-optimization-based numerical verification. In this work, we set the embedding dimension to 128, attention head to 2, and applied a 6-layer transformer decoder for the candidate expression generation. In each epoch, we sample $Q = 500$ expressions as candidates.

A.2 Global-optimization-based Numerical Verification

For a given dynamics $f(x)$, suppose $\tilde{V}_\phi$ is an invalid analytical candidate Lyapunov function. According to Lyapunov conditions defined in Proposition 1, for $x_1^*, x_2^* \in \mathcal{D}$, where x_1^*, x_2^* are the global minimizers of $\tilde{V}_\phi$ and $-L_f\tilde{V}_\phi$ in the state space $\mathcal{D}$, the following two inequalities hold: $\tilde{V}_\phi(x_1^*) \leq 0$ and $L_f\tilde{V}_\phi(x_2^*) \geq 0$. This implies that if $\tilde{V}_\phi$ is invalid, the neighborhoods of x_1^* and x_2^* are highly likely to capture significant violations. Based on this observation, we propose a global-optimization-based numerical verification. This verification identifies minimizers x_1^* and x_2^* by Simplicial Homology Global Optimization (SHGO), verifies Lyapunov conditions on localized samples in neighborhoods $\mathcal{B}_r(x_1^*)$ and $\mathcal{B}_r(x_2^*)$, and feeds counterexamples back into the training set $\mathcal{X}$. We detail the sampling and condition-checking procedures in Algorithm 2. Figure 3 illustrates this verification process on a sampled candidate, $\tilde{V}_\phi = (x_1 + x_2)^2 + x_2$, for the Van der Pol Oscillator. In the implementation, we initiate with 2048 starting points and iterate 3 times in the SHGO algorithm for the minimizer detection. We tested the number of starting points with values $[1024, 2048, 4096, 8196]$ on various high-dimensional continuous functions, and the setting with 2048 starting points achieves the best efficiency. In datapoint sampling for counter-example identification, for each candidate expression, 800 data points are sampled from each of $\mathcal{B}_r(x_1^*)$ and $\mathcal{B}_r(x_2^*)$, and additional 800 data points are randomly sampled across the state space $\mathcal{D}$.

Algorithm 2 Global-optimization-based Numerical Verification

```
Input: A set of analytical expressions $\mathcal{V} = \{V^i| i = 1, \cdots, Q\}$, radius $r$, and state space $\mathcal{D}$.
Output: a set of numerically valid candidate $\mathcal{V}^*$, a set of encountered counterexample $\mathcal{X}_{ce}$.
 1: $\mathcal{V}, \mathcal{X}_{ce} \leftarrow \{\}, \{\}$,
 2: for $i = 1$ to $Q$ do
 3:     $x_1^*,\ x_2^* \leftarrow$ SHGO$(V^i, \mathcal{D})$, SHGO$(-L_f V^i, \mathcal{D})$, {Identify global minimizers within the state space $\mathcal{D}$}
 4:     $\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3 \leftarrow \{x_i|\ x_i \in \mathcal{B}_r(x_1^*)\}, \{x_j|\ x_j \in \mathcal{B}_r(x_2^*)\}, \{x_k|\ x_k \in \mathcal{D}\}$
 5:     Check Lyapunov conditions on $\mathcal{X}_1 \cup \mathcal{X}_2 \cup \mathcal{X}_3$,
 6:     if $R(V^i) = 1$ and no counter example found in $\mathcal{X}_1 \cup \mathcal{X}_2 \cup \mathcal{X}_3$ then
 7:         $\mathcal{V}^* \leftarrow \mathcal{V}^* \cup \{V^i\}$.
 8:     else
 9:         $\mathcal{X}_{ce} \leftarrow \mathcal{X}_{ce} \cup$ identified counterexamples in $\mathcal{X}_1 \cup \mathcal{X}_2 \cup \mathcal{X}_3$. {Gather falsification}
10:     end if
11: end for
12: Return $\mathcal{V}^*, \mathcal{X}_{ce}$.
```

A.3 Risk-seeking Policy Gradient

Objective. The standard policy gradient $J_{\text{std}}(\phi) = \mathbb{E}_{\tilde{V}_\phi \sim p(\tilde{V}_\phi|\phi, f(x))}[R(\tilde{V}_\phi)]$ aims to optimize the average performance of a policy given the reward function $R(\cdot)$. However, for the task of Lyapunov function construction, the final performance is measured by identifying a single or a few valid analytical Lyapunov functions that satisfy the Lyapunov conditions. Thus, $J_{\text{std}}(\phi)$ is not an appropriate objective, as there is a mismatch between the objective being optimized and the

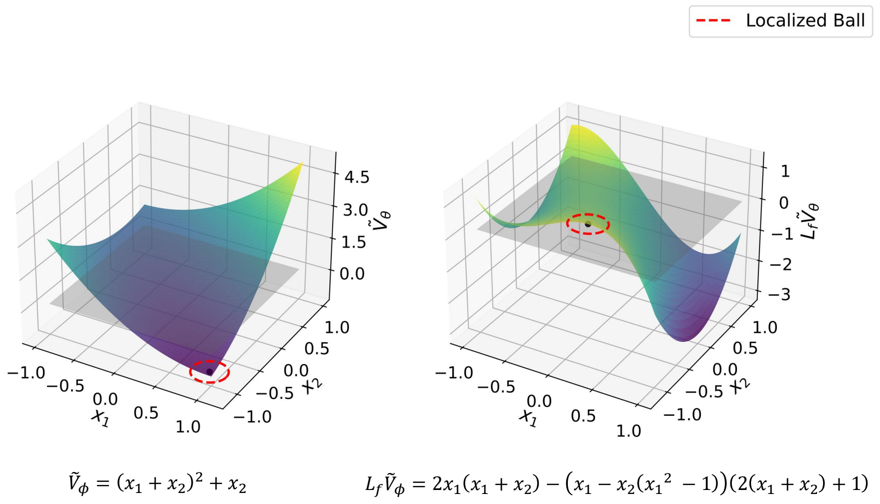

Figure 3: This plot visualizes our proposed verification process on a sampled candidate. $\tilde{V}_\phi = (x_1 + x_2)^2 + x_2$ is a sampled candidate during the training for Van Der Pol Oscillator. Using Simplicial Homology Global Optimization, we first identify the minimizer of the Lyapunov function and the maximizer of the Lie Derivative, the black dots in each graph. Next, data points are sampled in the neighborhoods of the two points, the regions in red circles. For sampled data points that violate the Lyapunov conditions, we feed them into the training set $\mathcal{X}$.

final performance evaluation metric. To address this misalignment, we adopt risk-seeking policy gradient [Petersen et al., 2020], optimizing the best-case performance via the objective $J_{\text{risk}}(\phi, \alpha)$, as defined in Equation (6). In implementation, we choose $\alpha = 0.1$ in the training for all tested dynamics.

Proposition 3 (Petersen et al. [2020])**.** *Let $J_{risk}(\phi, \alpha)$ denote the conditional expectation of rewards above the $(1-\alpha)$-quantile $R_\alpha(\phi)$ as in Equation* (6)*. Then the gradient of $J_{risk}(\phi, \alpha)$ is given by:*

$$\nabla_\phi J_{risk}(\phi, \alpha) = \mathbb{E}_{\tilde{V}_\phi \sim p(\tilde{V}_\phi | \phi, f(x))} \left[\left(R_\alpha(\phi) - R(\tilde{V}_\phi) \right) \cdot \nabla_\phi \log p(\tilde{V}_\phi \mid \phi, f(x)) \,\middle|\, R(\tilde{V}_\phi) \geq R_\alpha(\phi) \right]. \tag{8}$$

The proposition suggests a Monte Carlo estimate of the gradient of $J_{\text{risk}}(\phi, \alpha)$ from a batch of N samples:

$$\nabla_\phi J_{\text{risk}}(\phi, \alpha) \approx \frac{1}{\alpha N} \sum_{i=1}^{N} \left[\tilde{R}_\alpha(\phi) - R(\tilde{V}_\phi^{(i)}) \right] \cdot \mathbf{1}_{R(\tilde{V}_\phi^{(i)}) \geq \tilde{R}_\alpha(\phi)} \nabla_\phi \log p(\tilde{V}_\phi^{(i)} \mid \phi, f(x)), \tag{9}$$

where $\tilde{R}_\alpha(\phi)$ is the empirical $(1-\alpha)$-quantile of the batch of rewards, and 1_x returns 1 if condition x is true and 0 otherwise. Compared to standard REINFORCE algorithm [Williams, 1992], Equation (9) has two distinct features: (1) it has a specific baseline, $\tilde{R}_\alpha(\phi)$, instead of an arbitrary baseline in standard policy gradients chosen by user; (2) the gradient computation only uses the top α fraction of samples.

Reward Design. To optimize the symbolic transformer parameters ϕ such that the decoder generates a valid candidate Lyapunov function $\tilde{V}_\phi$ satisfying Lyapunov conditions, we employ empirical Lyapunov risk as the fitness metric to measure the violation degree of Lyapunov conditions within the state space following Chang et al. [2019]. However, directly using the unbounded empirical Lyapunov risk as the reward for risk-seeking policy gradient might introduce bias. To address this issue, we adopt a bounded reward function using the continuous mapping $g(x) = \frac{1}{x}$ [Petersen et al., 2020, Bastiani et al., 2024], defined as:

$$R(\tilde{V}_\phi) = g(\mathcal{L}(\tilde{V}_\phi)) = \frac{1}{1 + \mathcal{L}(\tilde{V}_\phi)},$$

where $\mathcal{L}(\tilde{V}_\phi)$ measures the violation degree over the training set $\mathcal{X}$. This design ensures the reward is bounded in $[0, 1]$, avoiding bias in the risk-seeking policy gradient.

A.4 Genetic Programming

In the field of symbolic regression, given the large, combinatorial search space, traditional approaches commonly utilize evolutionary algorithms, especially genetic programming (GP) [Koza, 1992], to retrieve analytical expressions that approximate the output values y given input data x. The GP-based symbolic regression operates by evolving the input population of mathematical expressions through evolutionary operations such as selection, crossover, and mutation. A pre-defined fitness metric serves as the objective function to guide the optimization of the population over successive generations. However, for analytical Lyapunov function construction, GP algorithms lack the capability to directly generate Lyapunov functions from the given dynamics and require an initial population that represents potential Lyapunov functions.

As the search space grows exponentially with the expression complexity and the number of states in input dynamics, it is a challenging task even for the symbolic transformer model to search a valid Lyapunov function for complex, high-dimensional systems. Inspired by Mundhenk et al. [2021], we incorporate a GP component into the training framework to complement the symbolic transformer model - the symbolic transformer model outputs a well-behaved initial populations of expressions $\tilde{V}_\phi$, which serve as the starting points for the GP component, and GP component refines $\tilde{V}_\phi$ through evolutionary operations to explore the characteristics of Lyapunov functions that might be overlooked by symbolic transformer. The fitness metric for the GP component is the same as the reward function used in the risk-seeking policy gradient. After each refinement, we select an 'elite set' of the top-performing refined expressions, $\tilde{\mathcal{V}}_{gp}$, based on fitness values. These expressions are treated as ground-truth solutions for the transformer decoder, and transformer parameters ϕ are optimized through the expert guidance loss introduced. In the implementation, the size of 'elite set' $\tilde{\mathcal{V}}_{gp}$ is chosen to be $0.1Q$, where Q is the number of sampled candidate expressions $\tilde{V}_\phi$ in each epoch.

In our framework, we employ three evolutionary operations: mutation, crossover, and selection, within our Genetic Programming component (DEAP [Fortin et al., 2012]). A mutation operator introduces random variations to an expression, such as replacing a subtree of one expression with another randomly generated subtree. A crossover operator exchanges content between two expressions, e.g., by swapping a subtree of one expression with a subtree of another expression, enabling the combination of their features. A selection operator determines which expressions persist into the next population. A common method is tournament selection [Koza, 1992], where a set of l candidate expressions is randomly sampled from the population, and the expression with the highest fitness value is selected. In each iteration of GP evolution, each expression has a probability of undergoing mutation and a probability of undergoing crossover; selection is performed until the new generation's population has the same size as the current generation's population. In empirical experiments, we set the probability of undergoing mutation and crossover to be 0.5, and we adjust the size of the tournament and number of evolutions proportional to the dimension of the input system.

B Experiment Settings

We validate the proposed algorithm across a variety of nonlinear dynamics by finding their local Lyapunov functions at the equilibrium point to verify their stability, where the systems are autonomous (or closed-loop systems with known feedback control laws). We use *dReal* [Gao et al., 2013] SMT solver for final verification of found Lyapunov functions, with a numerical tolerance error $\epsilon = e^{-3}$ and precision $\delta = e^{-12}$, over the state space, i.e. $V(x) > \delta$ and $L_f V(x) < -\delta$ over $\mathcal{D} \backslash \mathcal{B}_\epsilon(0)$. The excluded ball $\mathcal{B}_\epsilon(0)$ is to avoid numerical issues, which is a common practice for SMT-based formal verification [Chang et al., 2019]. Global stability can be determined through further expert analysis; for instance, if the Lie derivative is a negation of SOS, it is sufficient to establish global stability. The symbolic library $\mathcal{L}_s$ is defined as $\{+, -, \times, \sin, \cos, x_i\}$ in all tests.

Algorithm 3 Expert Guidance Loss

```
Input: 'Elite set' of analytical expressions $\tilde{\mathcal{V}}_{gp}$, input dynamics $f(x)$, and transformer parameters $\phi$.
Output: The weighted cross-entropy loss between the transformer model output probability distribution and given refined expressions $\tilde{\mathcal{V}}_{gp}$.
1: $G \leftarrow |\tilde{\mathcal{V}}_{gp}|$, {Get the size of 'elite set'}
2: $\mathcal{L} \leftarrow 0$,
3: for i = 1 to $G$ do
4:     $\mathcal{L} \leftarrow \mathcal{L} + \frac{1}{k_i} R(\tilde{V}^i_{gp}) \sum_{j=1}^{k_i} -\log\left(p(\tilde{V}^i_{gp_j} | \tilde{V}^i_{gp_{1:(j-1)}}, \phi, f(x))\right)$, {Calculate the expert guidance loss based on 'elite set' $\tilde{\mathcal{V}}_{gp}$. Equation (7)}
5: end for
6: $\mathcal{L}(\tilde{\mathcal{V}}_{gp}) \leftarrow \frac{1}{G}\mathcal{L}$,
7: Return $\mathcal{L}(\tilde{\mathcal{V}}_{gp})$.
```

C Baseline Descriptions

C.1 Augmented Neural Lyapunov Control (ANLC)

The Augmented Neural Lyapunov Control (ANLC) [Grande et al., 2023] combines Artificial Neural Networks (ANNs) with Satisfiability Modulo Theories (SMT) solvers to synthesize stabilizing control laws for the input dynamics $f(x)$ with formal guarantees. The neural network is trained over a dataset of state-space samples to generate candidate control laws and Lyapunov functions, while the SMT solvers are tasked with certifying the Lyapunov conditions of the neural Lyapunov function over a continuous domain and returning a counterexample if the function is invalid. To ease the computationally inefficient verification process in the SMT module, ANLC proposed a discrete falsifier, which discretized the state space for sample selection and evaluation, employed before the SMT call to avoid the frequent calling of the time-consuming SMT falsifier. As the previous learning-based Lyapunov function construction approaches usually initialized the parameters of control policy with pre-computed gains from state-feedback controllers, e.g. Linear-Quadratic Regulators, which requires user time and control expertise to properly perform the initialization process, ANLC instead removes the need of control initialization by its proposed compositional control architecture containing both linear and nonlinear control laws so that the proposed method allows the synthesis of nonlinear (as well as linear) control laws with the sole requirement being the knowledge of the system dynamics. For empirical experiments, we tested the ANLC algorithm for all system dynamics in Appendices E and F. We tested on the Van Der Pol Oscillator and 3D Trig dynamics to get the best hyperparameter setting. In bold, we show the chosen parameters, selected to have the best success discovery rate on Van Der Pol Oscillator and 3D Trig Dynamics.

- lr = $[0.1, \mathbf{0.01}, 0.001]$
- activations = $[(x^2, x^2, x^2), (\tanh, \tanh, x^2), \boldsymbol{(x^2, x^2, \tanh)}, (\tanh, \tanh, \tanh)]$
- hidden neurons = $[6, \mathbf{12}, 15, 20]$
- data = $[500, \mathbf{1000}, 2000]$
- iteration = $[500, \mathbf{1000}, 2000]$

C.2 FOSSIL 2.0

FOSSIL 2.0 [Edwards et al., 2024] is a software tool for robust formal synthesis of certificates (e.g., Lyapunov and barrier functions) for dynamical systems modelled as ordinary differential and difference equations. FOSSIL 2.0 implements a counterexample-guided inductive synthesis (CEGIS) for the construction of certificates alongside a feedback control law. In the loop of CEGIS, the learner, based upon neural network templates, acts as a candidate to satisfy the conditions over a finite set $\mathcal{D}$ of samples, while the verifier (formal verification tools) works in a symbolic environment that either confirms or falsifies whether the candidate from learner satisfies the conditions over the whole dense domain $\mathcal{X}$. If the verifier falsifies the candidate, one or more counterexamples identified by the verifier are added to the sample set, and the network is retrained. This loop repeats until the verification proves that no counterexamples exist or until a timeout is reached. Similar to the ANLC, in the empirical experiment, we set the hyperparameters based on the Van Der Pol Oscillator and 3-D Trig dynamics and tested for all other dynamics in Appendices E and F.

- lr = $[\mathbf{0.1}, 0.01, 0.001]$
- activations = $[(x^2, x^2), (\tanh, \tanh, x^2), \boldsymbol{(\tanh, x^2)}]$
- hidden neurons = $[\mathbf{6}, 10, 12]$
- data = $[500, \mathbf{1000}, 2000]$
- iteration = $[25, \mathbf{50}, 100]$

C.3 Global Lyapunov Function Discovery by Pre-trained Transformer

Alfarano et al. [2024] pre-trained a transformer on backward-generated and forward-generated global Lyapunov function datasets. The backward-generated datasets involve sampling arbitrary positive definite functions and deriving corresponding stable dynamics through some specific symbolic designs, while the forward-generated polynomial datasets contain randomly generated dynamics with corresponding Lyapunov functions identified by SOS methods if the system is inherently globally stable. Candidate Lyapunov expressions are sampled using beam search in a token-by-token manner. However, their method cannot adaptively refine the candidate Lyapunov functions if the beam search fails on specific dynamics, and it requires a dataset that is expensive to generate (e.g., thousands of CPU hours for a 5-D dynamics dataset) to achieve adequate generalization during inference. Furthermore, its emphasis on global stability limits its applicability to real-world, nonpolynomial control systems, which typically only admit local stability. Due to the lack of resources of multiple industrial-level GPUs, we contacted the authors of Alfarano et al. [2024] to conduct the evaluation of their pre-trained model on our test systems, which is shown in Section 4. Trained solely on globally stable systems with fewer than six states, it produced valid Lyapunov functions only for the low-dimensional examples in Appendices E.1, E.2, E.3, & F.1, which have global stability guarantees, and failed on every benchmark that is only locally stable.

C.4 Sum-of-Squares (SOS) Methods

SOS methods formulate Lyapunov functions discovery of given dynamics as a semi-definite programming task, where the coefficients of a pre-defined SOS candidate expressions are optimized to satisfy the Lyapunov conditions (hard constraints in the optimization problem) using convex optimization tools. SOS methods are generally applied to polynomial systems for stability analysis. With proper recasting techniques, SOS methods can also be applied to non-polynomial systems.

Definition 5 (Sum of Squares [Papachristodoulou and Prajna, 2005a]). For $x \in \mathbb{R}^n$, a multivariate polynomial $p(x)$ is a sum of squares (SOS) if there exist some polynomials $f_i(x), i = 1, \cdots, M$ such that

$$p(x) = \sum_{i=1}^{M} f_i(x)^2.$$

Polynomial systems. By Papachristodoulou and Prajna [2005a], for a given n-dimensional polynomial dynamics $f(x)$ and an integer degree $2d$, to check the globally asymptotical stability of $f(x)$, SOS method aims to find a polynomial $V(x)$ of degree $2d$, such that

1. $V(x) - \sum_{i=1}^n \sum_{j=1}^d \epsilon_{ij} x_i^{2j}$ is a SOS, where $\sum_{j=1}^d \epsilon_{ij} > \gamma, \forall i = 1, ..., n$ with $\gamma > 0$, and $\epsilon_{ij} \geq 0 \; \forall \; i$ and j,
2. $-\frac{\partial V}{\partial x} f(x)$ is a SOS.

For local stability analysis, consider a ball of radius r centered at origin $\mathcal{B}_r(0)$, which can be represented by the semialgebraic set $S = \{x : g(x, r) \geq 0$, where $g(x, r) = r - \sum_{i=1}^n x_i^2\}$. We require that the stability condition holds in S. Retaining the same optimization objective and constraints on $V(x)$ as before, a modified constraint on Lie derivative is imposed: $-\frac{\partial V}{\partial x} f(x) - s(x)g(x, r)$ is a SOS for some SOS $s(x)$. If such an $s(x)$ exists, we can establish local stability.

In Section 4, we develop our code based on the `findlyap` function from SOSTOOLS (MATLAB) and issue-16 of SOSTOOLS' official GitHub repo to examine the SOS method on polynomial systems in Appendix E. Table 4 summarizes the experiment results of SOS approach on our polynomial test dynamics.

Table 4: Training\solving time of sum-of-squares (SOS) on test polynomial systems.

Systems	App. F.1	App. F.2	App. F.3 - I	App. F.3 - II	App. F.4	App. F.5	App. F.6
Degree 2d	2	2	2	4	2	2	2
Region	$\mathcal{B}_1(0)$	Global	Global	Global	$\mathcal{B}_1(0)$	$\mathcal{B}_1(0)$	$\mathcal{B}_1(0)$
Runtime	0.697s	0.832s	0.497s	2.509s	-	-	-

D Experiment Results Summary

We summarized the runtime, success rate, and discovered Lyapunov functions for a selection of tested nonlinear systems, ranging from 2-D to 10-D, demonstrating the robustness and scalability of our framework in Table 5. As dimensionality increases, runtime grows with the exponentially expanding search space, reward calculations, SHGO optimization, and genetic programming. Detailed experiment results for each test dynamics are in Appendices E & F.

Unlike existing methods that produce neural Lyapunov functions, our framework yields interpretable *analytical* candidates. For example, it correctly identifies the energy function as a valid Lyapunov function for the simple pendulum. Likewise, for the 3-bus power system (Appendix F.3), it discovers the commonly used energy-based storage function for incremental passive systems Weitenberg et al. [2018].

Analytical Lyapunov functions can potentially bypass the need for formal verification. In the 3-D Trig system (Appendix F.2), over the state space $\mathcal{D} = \{(x_1, x_2, x_3) \in \mathbb{R}^3 \mid |x_i| \leq 1.5, \forall\, i \in \{1, 2, 3\}\}$, the positive definiteness of the identified Lyapunov function is evident from its formulation. Moreover, the Lie derivative $L_f V = -2x_2^2 - x_3 \sin(2x_3)$ is directly identifiable as non-positive in $\mathcal{D}$, since $x \sin(x) > 0$ for all $x \in (-\pi, \pi)$. By the invariance principle [Khalil, 2002], the discovered function certifies the asymptotic stability of the origin in state space $\mathcal{D}$. When direct identification is non-trivial, SMT solvers can efficiently verify Lyapunov conditions given analytical formulations' simplicity.

Table 5: Performance and time consumption of our method on test dynamics. 'App.' refers to Appendix.

Dynamics	**Runtime**	**Ver.**^{a}	**Found Lyapunov Functions**	**Stab**[‡]	**Succ %** [‡]
2-D Polynomial Sys (App. E.2)	68s	2 ms	$V = 9x_1^2 + 2x_2^2$	l.a.s.	100
2-D Van Der Pol (App. E.1)	126s	1 ms	$V = x_1^2 + x_2^2$	l.a.s.	100
2-D Simple Pendulum (App. F.1)	288s	1 ms	$V = 2(1 - \cos(x_1)) + x_2^2$	l.a.s.	100
3-D Polynomial Sys (App. E.3)	112s	1 ms	$V = 9x_1^2 + x_2^2 + x_3^2$	l.a.s.	100
3-D Trig Dynamics (App. F.2)	157s	1 ms	$V = 1 - \cos(x_1)^2 + x_2^2 + \sin(x_3)^2$	l.a.s.	100
4-D Lossy Power Sys (App. F.5)	3632s	621s	$V = \omega_1^2 + \omega_2^2 + (\omega_2 - \sin(\delta_1) + \sin(\delta_2))^2$	l.a.s.	100
6-D Polynomial Sys (App. E.4)	1667s	7 ms	$V = \sum_{i=1}^{6} x_i^2$	l.a.s.	100
6-D Quadrotor (App. F.4)	3218s	1 ms	$V = \sum_{i=1}^{6} x_i^2$	g.a.s.	80
6-D Lossless Power Sys (App. F.3)	18094s	2 ms	$V = (\sum_{i=1}^{3} \omega_i^2) - 0.5 \left(\sum_{i=1}^{3} \sum_{j=1, i\neq j}^{3} \cos(\delta_i - \delta_j) - 1 \right)$	l.a.s.	60
9-D Synthetic Sys (App. F.6)	27047s	6.6 s	$V = (\sum_{i=1}^{6} x_i^2) + \sin(x_7)^2 + x_8^2 - \cos(x_9) + 1$	l.a.s.	60
10-D Polynomial Sys (App. E.6)	64223s	2 ms	$V = \sum_{i=1}^{10} x_i^2$	l.a.s.	60

a. 'Ver.' presents the time consumption for the final verification of the found Lyapunov functions. All found Lyapunov functions passed SMT solver's verification.

‡. 'Stab' means stability. In this column, 'g.a.s' represents globally asymptotically stable, and 'l.a.s.' represents locally asymptotically stable.

‡. 'Succ %' denotes the successful rate of finding a valid Lyapunov function out of 5 random seeds.

E Polynomial Nonlinear Dynamical System

E.1 Van Der Pol Oscillator

Van Der Pol Oscillator is a nonconservative, oscillating system with nonlinear damping [Zhou et al., 2022]. The dynamics of the Van Der Pol Oscillator have two state variables, formulated as follows:

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= -x_1 - \mu(1 - x_1^2) \cdot x_2,\end{aligned}$$

where x_1 and x_2 represent the object's position in the Cartesian coordinate, parameter $\mu \in \mathbb{R}^+$ indicates the strength of the damping. Under the state space $\mathcal{D} = \{(x_1, x_2) \in \mathbb{R}^2 \mid |x_i| \leq 1\}$ and setting $\mu = 1$, our proposed method found valid local Lyapunov function $V(x_1, x_2) = x_1^2 + x_2^2$. Other forms of Lyapunov functions for Van Der Pol Oscillator, for example, $V(x_1, x_2) = x_1^2 + x_2(x_1 + x_2)$, are also recovered during the experiments.

E.2 Two-variable-polynomial-system with higher degree

Here we have a polynomial system of two variables with a higher degree, adopted from Alfarano et al. [2024], formulated as:

$$\begin{cases}\dot{x}_1 &= -5x_1^3 - 2x_1 \cdot x_2^2,\\ \dot{x}_2 &= -9x_1^4 + 3x_1^3 \cdot x_2 - 4x_2^3.\end{cases}$$

Under the state space $\mathcal{D} = \{(x_1, x_2) \in \mathbb{R}^2 \mid |x_i| \leq 1\}$, our proposed method successfully found valid local Lyapunov function $V(x_1, x_2) = 9x_1^2 + x_2^2$.

E.3 Three-variable-polynomial-systems with higher degree

Table 6 describes two polynomial systems of three variables with a higher degree, adopted from Alfarano et al. [2024]. Our framework successfully retrieves valid local Lyapunov functions on both examples under the state space $\mathcal{D} = \{(x_1, x_2, x_3) \in \mathbb{R}^3 \mid |x_i| \leq 1\}$.

Table 6: Three-Dimensional Polynomial Example with Higher Degree.

System	**Lyapunov function**
$\begin{cases}\dot{x}_1 = -3x_1^3 + 3x_1 \cdot x_3 - 9x_1\\ \dot{x}_2 = -x_1^3 - 5x_2 + 5x_3^2\\ \dot{x}_3 = -9x_3^3\end{cases}$	$V(x_1, x_2, x_3) = 9x_1^2 + x_2^2 + x_3^2$
$\begin{cases}\dot{x}_1 = -8x_1 \cdot x_2^2 - 10x_2^4\\ \dot{x}_2 = -8x_2^3 + 3x_2^2 - 8x_2\\ \dot{x}_3 = -x_3\end{cases}$	$V(x_1, x_2, x_3) = x_1^8 \cdot x_2^2 \cdot x_3^2 + x_2^2$

E.4 6-D Polynomial Nonlinear System

This 6-D dynamics consists of three two-dimensional asymptotically stable linear subsystems that are coupled by three nonlinearities with small gains adopted from Grüne [2019]. The dynamics are written as:

$$\begin{aligned}\dot{x}_1 &= -x_1 + 0.5x_2 - 0.1x_5^2,\\ \dot{x}_2 &= -0.5x_1 - x_2,\\ \dot{x}_3 &= -x_3 + 0.5x_4 - 0.1x_1^2,\\ \dot{x}_4 &= -0.5x_3 - x_4,\\ \dot{x}_5 &= -x_5 + 0.5x_6,\\ \dot{x}_6 &= -0.5x_5 - x_6 + 0.1x_2^2.\end{aligned}$$

Our proposed method is trained over the state space $\mathcal{D} = \{(x_1, x_2, x_3, x_4, x_5, x_6) \in \mathbb{R}^6 \mid |x_i| \leq 1, \forall i \in \{1, 2, .., 6\}\}$, and is able to find a valid Lyapunov function $V(x) = x_1^2+x_2^2+x_3^2+x_4^2+x_5^2+x_6^2$. This dynamics is not globally asymptotically stable since if $x_1, x_2,$ or x_5 has a significantly large value, the perturbations introduced by the small gains will shift the object by a significant amount away from the equilibrium point. By empirical checking, our found Lyapunov function certifies the asymptotical stability of this system over the region $\mathcal{D}' = \{(x_1, x_2, x_3, x_4, x_5, x_6) \in \mathbb{R}^6 \mid \sum_{i=1}^{6} x_i^2 \leq 500\}$.

E.5 8-D Polynomial Nonlinear System

This 8-D dynamics consists of four two-dimensional asymptotically stable linear subsystems that are coupled by four nonlinearities with small gains, modified from the above 6D polynomial dynamics. The dynamics are written as:

$$\begin{aligned}
\dot{x}_1 &= -x_1 + 0.5x_2 - 0.1x_5^2, \\
\dot{x}_2 &= -0.5x_1 - x_2, \\
\dot{x}_3 &= -x_3 + 0.5x_4 - 0.1x_1^2, \\
\dot{x}_4 &= -0.5x_3 - x_4, \\
\dot{x}_5 &= -x_5 + 0.5x_6 + 0.1x_7^2, \\
\dot{x}_6 &= -0.5x_5 - x_6, \\
\dot{x}_7 &= -x_7 + 0.5x_8, \\
\dot{x}_8 &= -0.5x_7 - x_8 - 0.1x_4^2.
\end{aligned}$$

Our proposed method is trained over state space $\mathcal{D} = \{(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8) \in \mathbb{R}^8 \mid |x_i| \leq 1, \forall i \in \{1, 2, .., 8\}\}$, and is able to find a valid Lyapunov function $V(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2 + x_8^2$. This Lyapunov function certifies the asymptotical stability of this system over the region $\mathcal{D} = \{(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8) \in \mathbb{R}^8 \mid \sum_{i=1}^{8} x_i^2 \leq 450\}$. This dynamics is not globally asymptotically stable since if $x_1, x_4, x_5,$ or x_7 has a significantly large value, the perturbations introduced by the small gains will shift the object by a significant amount away from the equilibrium point.

E.6 10-D Polynomial Nonlinear System

Finally, we extend to the original 10-D polynomial dynamics proposed in Grüne [2019]. This 10-D dynamics consists of five two-dimensional asymptotically stable linear subsystems that are coupled by four nonlinearities with small gains. The dynamics are written as:

$$\begin{aligned}
\dot{x}_1 &= -x_1 + 0.5x_2 - 0.1x_5^2, \\
\dot{x}_2 &= -0.5x_1 - x_2, \\
\dot{x}_3 &= -x_3 + 0.5x_4 - 0.1x_1^2, \\
\dot{x}_4 &= -0.5x_3 - x_4, \\
\dot{x}_5 &= -x_5 + 0.5x_6 + 0.1x_9^2, \\
\dot{x}_6 &= -0.5x_5 - x_6, \\
\dot{x}_7 &= -x_7 + 0.5x_8, \\
\dot{x}_8 &= -0.5x_7 - x_8. \\
\dot{x}_9 &= -x_9 + 0.5x_{10}, \\
\dot{x}_{10} &= -0.5x_9 - x_{10} - 0.1x_4^2.
\end{aligned}$$

Our proposed method is trained over the state space $\mathcal{D} = \{(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}) \in \mathbb{R}^{10} \mid |x_i| \leq 1, \forall i \in \{1, 2, .., 10\}\}$, and is able to find a valid Lyapunov function $V(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2 + x_7^2 + x_8^2 + x_9^2 + x_{10}^2$. This Lyapunov function certifies the asymptotic stability of this system over the region $\mathcal{D} = \{(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}) \in \mathbb{R}^{10} \mid \sum_{i=1}^{10} x_i^2 \leq 400\}$. This dynamics is not globally asymptotically stable since if $x_1, x_4, x_5,$ or x_9 has a significantly large value, the perturbations introduced by the small gains will shift the object by a significant amount away from the equilibrium point.

F Non-polynomial Nonlinear Dynamical Systems

F.1 Simple Pendulum

The simple pendulum is a well-known classical nonlinear system that contains two state variables. The dynamics are formulated as follows,

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= -\frac{g}{l}\sin(x_1) - \frac{b}{m}x_2,\end{aligned}$$

where x_1 is the angular position from the inverted position, x_2 is the angular velocity, and parameters g, m, l, b are the acceleration of gravity, the mass of the inverted object, the length of the string, and the coefficient of friction, respectively. In experiments, since we don't incorporate the constant generation capability within the training framework, we set $g = 1$, $m = 1\,\mathrm{kg}$, $l = 1\,\mathrm{m}$, and $b = 0.1$. Our proposed method finds the valid Lyapunov function $V = 2 - 2\cos(x_1) + x_2^2$ over the state space: $\mathcal{D} = \{(x_1, x_2) \in \mathbb{R}^2 \mid |x_1| \leq \pi$ and $|x_2| \leq 6\}$. This found Lyapunov function has the same analytical structure as the energy function of the inverted pendulum.

F.2 3-D Trigonometric System

3-D trig dynamics comes from exercise problems in textbook [Khalil, 2002] whose dynamics are written as follows,

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= -h(x_1) - x_2 - h(x_3),\\ \dot{x}_3 &= x_2 - x_3,\end{aligned}$$

where $h(x) = \sin(x)\cdot\cos(x)$. When the state space is $\mathcal{D} = \{(x_1, x_2, x_3) \in \mathbb{R}^3 \mid |x_i| \leq 1.5, \forall\, i \in \{1, 2, 3\}\}$, the valid local Lyapunov function found by our proposed method is $V(x_1, x_2, x_3) = 1 - \cos(x_1)^2 + x_2^2 + \sin(x_3)^2$, which is consistent to the textbook solution of Lyapunov function for this particular dynamics.

F.3 N-bus Lossless Power System

We test our proposed framework on the N-bus power lossless system [Cui et al., 2023, Feng et al., 2024] to examine its ability to handle complex high-dimensional dynamics. Consider θ_i, ω_i as the phase angle and the frequency of bus i, respectively, the dynamics for each bus are formulated as follows,

$$\begin{aligned}\dot{\theta}_i &= \omega_i,\\ m_i\dot{\omega}_i &= p_i - d_i\omega_i - u_i(\omega_i) - \sum_{j=1}^{N} B_{ij}\cdot\sin(\theta_i - \theta_j),\end{aligned}$$

where m_i is the generator inertia constant, d_i is the combined frequency response coefficient from synchronous generators and frequency sensitive load, and p_i is the net power injection, for each bus $i = 1, \cdots, N$. $B \in \mathbb{R}^{N\times N}$ is the susceptance matrix with $B_{ij} = 0$ for every pair $\{i, j\}$ such that bus i and bus j are not connected, and $u_i(\omega_i)$ is the controller at bus i that adjusts the power injection to stabilize the frequency.

Since the frequency dynamics of the system depends only on the phase angle differences, so we change the coordinates:

$$\delta_i = \theta_i - \frac{1}{N}\sum_{i=1}^{N}\theta_i$$

where δ_i can be understood as the center-of-inertia coordinates of each bus. In our experiment, we test the proposed framework on the 3-bus power system. For simplicity, we set $p_i = 0$, $m_i = 2$, $d_i = 1$, $u_i(\omega_i) = \omega_i$, and $B_{ij} = 1 \, \forall \, i \neq j, B_{ii} = 0$. In this case, the equilibrium point for our system is at the origin, i.e. $\delta_i^* = \omega_i^* = 0, \; i = 1, 2, 3$. The state space for our experiment is defined as: $\mathcal{D} = \{(\delta_1, \delta_2, \delta_3, \omega_1, \omega_2, \omega_3) \in \mathbb{R}^6 \mid |\delta_i| \leq 0.75 \text{ and } |\omega_i| \leq 1.2 \text{ for } i = 1, 2, 3\}$. Through our method, we retrieved a valid Lyapunov function $V(\delta_1, \delta_2, \delta_3, \omega_1, \omega_2, \omega_3) = (\sum_{i=1}^{3} \omega_i^2) - 0.5 \left(\sum_{i=1}^{3} \sum_{j=1, i \neq j}^{3} \cos(\delta_i - \delta_j) - 1 \right)$, which is consistent to the known Lyapunov function presented in Cui et al. [2023]. The Lie derivative of the identified Lyapunov function can be simplified as $L_f V = -2(\omega_1^2 + \omega_2^2 + \omega_3^2)$. The analytical structure of this found Lyapunov function and invariance principle allows us to easily identify it as a valid Lyapunov function by hand.

F.4 Indoor Micro Quadrotor

For the angular rotations subsystems of the quadrator from Bouabdallah et al. [2004], it has 6 states to describe the angular motion of the quadrotor. The states $x_1, x_3,$ and x_5 describe the roll, pitch, and yaw of the quadrator, and states $x_2, x_4,$ and x_6 represent their time derivatives. With perturbation terms Ω and control inputs $U_1, U_2,$ and U_3, the subsystem can be formulated as follows,

$$\begin{aligned}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= x_4 x_6 (\frac{I_y - I_z}{I_x}) - \frac{J_R}{I_x} x_4 \Omega + \frac{l}{I_x} U_1, \\
\dot{x}_3 &= x_4, \\
\dot{x}_4 &= x_2 x_6 (\frac{I_z - I_x}{I_y}) + \frac{J_R}{I_y} x_2 \Omega + \frac{l}{I_y} U_2, \\
\dot{x}_5 &= x_6, \\
\dot{x}_6 &= x_2 x_4 (\frac{I_x - I_y}{I_z}) + \frac{l}{I_z} U_3,
\end{aligned}$$

where $I_x, I_y,$ and I_z represents the body inertia, l denotes the lever, and J_R is the rotor inertia. With the control policy

$$\begin{aligned}
U_1 &= -\frac{I_x}{l}(x_1 - x_1^d) - k_1 x_2, \\
U_2 &= -\frac{I_y}{l}(x_3 - x_3^d) - k_2 x_4, \\
U_3 &= -I_z(x_5 - x_5^d) - k_3 x_6,
\end{aligned}$$

and restricting $I_x = I_y$, the angular rotations subsystems is stabilized to the chosen equilibrium point $X_d = \{x_1^d, 0, x_3^d, 0, x_5^d, 0\}$. In empirical experiments, we set $X_d = \{0, 0, 0, 0, 0, 0\}$, state space $\mathcal{D} = \{(x_1, x_2, x_3, x_4, x_5, x_6) \in \mathbb{R}^6 \mid |x_i| \leq 3, \forall i \in \{1, 2, .., 6\}\}$, $I_x = I_y = 2$, $I_z = 5$, $k_1 = 5$, $k_2 = 20$, $k_3 = 4$, $l = 1$, $J_R = 1$, and $\Omega = \sin(x_2)\cos(x_4)$, our framework successfully found Lyapunov function $V(x_1, x_2, x_3, x_4, x_5, x_6) = \sum_{i=1}^{6} x_i^2$. By examining the Lie derivative $L_f V = -2.5x_2^2 - 10x_4^2 - 0.8x_6^2$ and using the Invariance principle, we conclude that this subsystem is globally asymptotically stable. Under other parameter settings, our framework can also retrieve valid analytical Lyapunov functions for this dynamics.

F.5 N-bus Lossy Power System

Unlike N-bus lossless power systems in Appendix F.3 which has a well-known energy-based storage function served as a valid Lyapunov function for asymptotical stability guarantee, the N-bus lossy power system [Cui and Zhang, 2022] does not have a known analytical Lyapunov function to certify stability, though by passivity the system should be asymptotically stable at origin. Utilizing the proposed framework, we aim to discover a valid analytical Lyapunov function for an N-bus lossy power system to formally certify its asymptotic stability.

The θ_i and δ_i are the angle and frequency deviation of bus i, the dynamics of an N-bus lossy power system is represented by the swing equation, formulated as:

$$\begin{aligned}
&\dot{\theta}_i = \omega_i, \\
&m_i \dot{\omega}_i = p_i - d_i \omega_i - u_i(\omega_i) - \sum_{j=1}^{N} B_{ij} \cdot \sin(\theta_i - \theta_j) - \sum_{j=1}^{N} G_{ij} \cdot \cos(\theta_i - \theta_j),
\end{aligned}$$

where m_i is the generator inertia constant, d_i is the combined frequency response coefficient from synchronous generators and frequency-sensitive load, and p_i is the net power injection, for each bus $i = 1, \cdots, N$. The susceptance and conductance of the line (i, j) are $B_{ij} = B_{ji}$ and $G_{ij} = G_{ji}$, respectively. The value is 0 if the buses are not connected. In this work, we consider input u_i to be a static feedback controller where only its local frequency measurement ω_i is available. Like the lossless power system, since the frequency dynamics of the system depends only on the phase angle differences, we change the coordinates:

$$\delta_i = \theta_i - \frac{1}{N} \sum_{i=1}^{N} \theta_i$$

where δ_i can be understood as the center-of-inertia coordinates of each bus.

We test the proposed framework on a 2-bus lossy power system. In experiment, we set $p_i = 1$,$m_i = 2$, $d_i = 1$, $u_i(w_i) = w_i$, $B_{ij} = 1, G_{ij} = 1 \; \forall \; i \neq j$, $B_{ii} = 0, G_{ij} = 0$. By this setting, the equilibrium point for this system is at the origin, i.e. $\delta_i^* = \omega_i^* = 0$. The state space for the experiment is defined as: $\mathcal{D} = \{(\delta_1, \delta_2, \omega_1, \omega_2) \in \mathbb{R}^4 \mid |\delta_i| \leq 0.75 \text{ and } |\omega_i| \leq 2 \text{ for } i = 1, 2\}$. The proposed method found two valid Lyapunov function $V(\delta_1, \delta_2, \omega_1, \omega_2) = \omega_1^2 + \omega_2^2 + (\omega_2 - \sin(\delta_1) + \sin(\delta_2))^2$ and $V(\delta_1, \delta_2, \omega_1, \omega_2) = \omega_1^2 + \omega_2^2 + (-\omega_1 - \sin(\delta_1) + \sin(\delta_2))^2$. Both Lyapunov functions pass the formal verification by SMT solver in the state space $\mathcal{D} \backslash \mathcal{B}_\epsilon(0)$, where precision δ is set to be e^{-12} and $\epsilon = e^{-3}$ to avoid tolerable numerical error.

F.6 9-D Synthetic Dynamics

Consider the synthetic dynamics adapted from Appendices E.4 & F.2 with linear interactions between two subsystems:

$$\begin{aligned}
\dot{x}_1 &= -x_1 + 0.5x_2 - 0.1x_5^2, \\
\dot{x}_2 &= -0.5x_1 - x_2 + 0.1x_8, \\
\dot{x}_3 &= -x_3 + 0.5x_4 - 0.1x_1^2, \\
\dot{x}_4 &= -0.5x_3 - x_4, \\
\dot{x}_5 &= -x_5 + 0.5x_6, \\
\dot{x}_6 &= -0.5x_5 - x_6 + 0.1x_2^2, \\
\dot{x}_7 &= x_8, \\
\dot{x}_8 &= -\sin(x_7)\cos(x_7) - x_8 - \sin(x_9)\cos(x_9) - 0.1x_2, \\
\dot{x}_9 &= x_8 - x_9.
\end{aligned}$$

To properly address the trigonometric terms in $\dot{x}_8$, the Lyapunov function for this dynamics can't be a simple form like $\sum_{i=1}^{n} x_i^2$ and should include some trigonometric terms. Setting the state space $\mathcal{D} = \{x \in \mathbb{R}^9 || x_i| \leq 1.5, \forall \; i = 1, \cdots, 9\}$, our method successfully identifies a valid Lyapunov function $V = \sum_{i=1}^{6} x_i^2 + \sin(x_7)^2 + x_8^2 - \cos(x_9) + 1$, which passes formal verification following settings in Section 4.