Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-26)
Early Career Spotlight Track

Verifiable PDE Reasoning and Modeling with Neurosymbolics

Wuyang Chen
School of Computing Science, Simon Fraser University
wuyang@sfu.ca

Abstract

Recent progress in Large Language Models (LLMs) has transformed text and code generation, yet models still falter on Partial Differential Equations (PDEs) where correctness, constraints, and physical consequences are critical. We explore how formal LLM reasoning can advance symbolic PDE modeling. First, our PDE-Controller formalizes informal PDEs, synthesizes solver-ready code, and plans subgoals to tackle nonconvex control via interactions with external solvers. Second, our Lean Finder accelerates PDE formalization via a semantics-aware search engine for Lean/Mathlib that retrieves relevant theorems, outperforming GPT models and gaining significant traction in the AI-for-math community. Through these efforts, we aim to design a semantics-first LLM that autoformalizes informal PDE problems into machine-checked specifications and synthesizes solver-ready code. This closes the loop between formal analysis and LLM reasoning, ultimately surpassing human heuristics across PDEs.

1 Introduction

My research is driven by a fundamental scientific question: ***how can we achieve verifiable Partial Differential Equation (PDE) reasoning and modeling for complex physical systems?*** PDEs power our most consequential models of the physical world, governing decisions across climate, aircraft, and plasmas, where we predict extremes, certify designs, and control high-stakes systems. Yet, despite decades of progress in analysis, numerics, and scientific computing, ***our PDE modeling remains largely unverifiable***. Formal properties (e.g., well-posedness and stability) are *proved by hand* and rarely linked to machine-checked verification; controllers are validated empirically rather than by formal guarantees [Dawson *et al.*, 2023]; and even widely used PDE solvers and surrogate models rarely make their guarantees explicit in the regimes where they are deployed [Riedmaier *et al.*, 2021]. This gap is increasingly pressing as scientific computing moves toward large-scale PDE-constrained optimization, neural PDE surrogates, and foundation models.

In parallel, Large Language Models (LLMs) have transformed text and code generation and exhibit strong commonsense reasoning, yet they still *falter in complex scientific domains* [Ahn *et al.*, 2024]. While AI-for-math has rapidly advanced formal reasoning in *pure* mathematics, from grade-school word problems [Cobbe *et al.*, 2021] to the International Mathematical Olympiad [Trinh *et al.*, 2024], with growing surveys of deep learning for mathematical reasoning and theorem proving [Lu *et al.*, 2023; Li *et al.*, 2024], progress in *applied* mathematics, and especially PDEs, remains limited. Unlike pure mathematics, which targets abstract theory, applied mathematics directly bridges theory and real-world needs: PDEs model physical dynamics across aerospace engineering, quantum mechanics, and fluid dynamics, providing the framework we use to understand and control such systems. Yet two bottlenecks persist. First, classical optimization and formal-method pipelines demand heavy manual formalization, requiring significant human effort to translate informal descriptions into specifications. Second, PDE control and analysis require highly specialized coding and physics expertise that is challenging even for seasoned mathematicians. Off-the-shelf LLMs trained on generic web corpora lack such scientific grounding, and the relevant formal languages and solver libraries are data-sparse compared to mainstream code [Chen *et al.*, 2021]. Autoformalization, translating informal mathematics into machine-checkable form [Wu *et al.*, 2022], is therefore both a key enabler and a major open challenge for PDEs.

Building on my PhD thesis on principles of deep network architectures and rooted in my recent work on scientific machine learning (SciML) [Chen *et al.*, 2024; Ma *et al.*, 2026; Liu *et al.*, 2026], my research targets **verifiable PDE reasoning** through **hybrid neurosymbolics** (Figure 1). I design a *semantics-first* pipeline in which an LLM autoformalizes informal PDE problems into machine-checkable specifications, synthesizes solver-ready code, plans subgoals, and is tightly coupled with external solvers and proof assistants that execute, verify, and refine its outputs. This spans both *formal PDE control* (how to certify that a PDE system obeys machine-checkable specifications?) and *formal PDE analysis* (how to mechanize proofs of existence, uniqueness, and stability?). At the same time, verifiable reasoning must remain grounded in *PDE-based modeling of real phenomena*: high-dimensional flows demand reconstruction and genera-

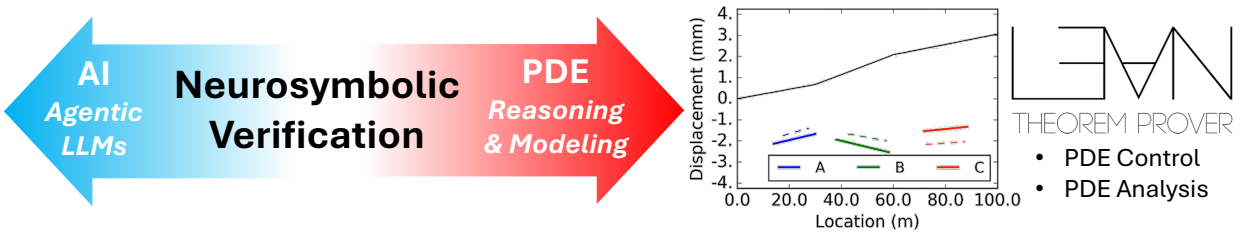

Figure 1: We address the core challenge of *verifiable PDE reasoning and modeling with neurosymbolics*, coupling LLM autoformalization, program synthesis, and subgoal planning with external solvers and the Lean proof assistant.

tion that connect abstract PDE descriptions to observable, editable physical behavior. Together, these efforts work toward a framework in which *mathematically certified PDE reasoning* and *high-fidelity physical modeling* co-evolve to support trustworthy decisions in science and engineering.

2 Verifiable PDE Reasoning with Neurosymbolics

Given informal PDE descriptions and safety requirements, the central problem is: ***how to obtain machine-checkable specifications, proofs, and controllers that support reliable decisions?*** This requires handling both spatio-temporal constraints (for control) and proofs of existence, uniqueness, and stability (for analysis). I pursue this agenda through two complementary lines of work, **PDE-Controller** and a **Lean-based formal-analysis stack**, linking informal PDE problems to machine-checked constraints, code, and theorems.

2.1 PDE-Controller: Formal PDE Control with Agentic LLMs

PDE-CONTROLLER [Soroco *et al.*, 2025] is a neurosymbolic framework for PDE control that autoformalizes natural-language tasks into spatio-temporal logic, synthesizes solver-ready programs, and plans subgoals for nonconvex PDE-constrained optimization, outperforming both human heuristics and black-box prompting. The motivating question is direct: ***can LLMs control PDEs with scientific reasoning?***

Background: PDE Control as Nonconvex Optimization
A PDE control problem takes a natural-language description of the initial state, system conditions (the governing PDE, geometry, initial and boundary conditions), and target constraints, and seeks a time-variant control trajectory. Following formal methods for cyber-physical systems, the spatio-temporal constraints are expressed in *Signal Temporal Logic* (STL) [Maler and Nickovic, 2004], whose satisfaction can be quantified by a real-valued *control utility* $r(\phi)$. The synthesized control task is then compiled into a Mixed-Integer Linear Program (MILP) and solved by Gurobi. Crucially, this utility is *nonconvex*, so naively optimizing from the given initial condition is frequently trapped in poor local optima due to loss-landscape barriers and bad initializations, a difficulty that motivates explicit reasoning.

Principled Data Synthesis
Since formal languages and solver libraries are data-sparse and existing math-reasoning corpora lack large-scale PDE data, we construct a dataset by principled synthesis with augmentations. We organize eligible STL syntax (e.g., the temporal operators $\{\mathbf{G}, \mathbf{F}\}$ combined with comparisons $\{<, >, =\}$, connected by $\wedge/\vee$ with parenthesized precedence for up to three constraints) to enumerate diverse, well-formed specifications, then pair each with system descriptions and natural-language paraphrases. This supplies supervised signals that generic LLMs lack while guaranteeing formal correctness.

Autoformalization and Program Synthesis
PDE-CONTROLLER orchestrates three cooperating LLMs, all fine-tuned from a MathCoder2 checkpoint with LoRA, around external symbolic tools. A *Translator* LLM performs **autoformalization**: it separates constraints from system descriptions, aligns informal concepts to formal STL syntax, and connects multiple constraints with the correct logical operators, trained by supervised fine-tuning that predicts the formal sequence given the informal input. A *Coder* LLM then performs **program synthesis**: taking both the prompt and the formalized STL, it generates Python that drives the PDE simulator and the Gurobi optimizer. Notably, the LLMs do *not* solve the PDE themselves; they *understand and orchestrate* external solvers that provide ground-truth correctness.

PDE Reasoning via the Controller LLM
Our most important contribution is to ask whether an LLM can *reason and plan* on scientific problems where pretrained commonsense knowledge offers little help. Inspired by subgoal decomposition in robotics and AI-for-math, we adopt the strategy that *a PDE control problem can be improved by decomposing it into subgoals optimized sequentially*: given a target "anchor" problem ϕ, the *Controller* LLM proposes an intermediate STL subgoal ϕ' that reshapes the state, so the system first solves ϕ' and then attacks ϕ from a better starting point. We collect win–lose subgoal pairs by ranking candidates with symbolic feedback from the PDE solver (preferring subgoals that raise the anchor's utility), and fine-tune the Controller with Direct Preference Optimization, shifting its distribution toward preferred subgoals while eliminating ad hoc human design.

Results
On 1D heat and wave problems, PDE-CONTROLLER achieves accurate autoformalization and program synthesis, and its Controller *consistently improves control utility* across easy, medium, and hard difficulty levels, surpassing human heuristics, random subgoal sampling, and black-box prompting of frontier models (GPT-4o, o1-mini). The result is an

agentic, tool-using LLM that translates informal specifications into solver-ready code and subgoals, pointing toward automated, accelerated engineering design verification.

2.2 Lean Finder: Accelerating PDE Formalization

Formal PDE analysis requires expressing definitions, theorems, and proofs inside an interactive theorem prover and machine-checking them. Lean and its `mathlib` library [de Moura *et al.*, 2015; Moura and Ullrich, 2021] offer a powerful foundation, but two obstacles limit their use for PDEs: (i) *missing PDE primitives* (differential operators; heat/wave/Laplace equations; Sobolev spaces), and (ii) a *severe retrieval gap*: `mathlib` exposes over 230K theorems and 110K definitions, far more "entry points" than typical programming platforms, and existing tools (#find, `library_search`, Loogle) depend on exact names or goal states and fail when naming conventions drift. Recent LLM-based search engines [Yang *et al.*, 2023] embed informal queries or proof goals, but optimize for embedding similarity rather than the nuanced, shifting *intents* of mathematicians. This raises the question: ***how can Lean retrieval become aware of mathematicians' intents so that formal PDE analysis becomes practical at scale?***

We design LEAN FINDER [Lu *et al.*, 2026a], a semantics-aware code-search LLM for Lean trained via *reversed data synthesis*. We first cluster real queries from forums (Zulip, GitHub) to characterize how mathematicians actually phrase their needs, then prompt LLMs to generate *intent-biased* queries for each Lean statement, building the largest Lean code-search dataset to date: over 1.4M query–code pairs (582K synthesized user queries, 245K informalized statements, and 338K proof states). We train LEAN FINDER with both contrastive learning and reinforcement learning from mathematicians' feedback, aligning retrieval with user intent rather than surface form. In an LMArena-style human evaluation [Chiang *et al.*, 2024], LEAN FINDER is upvoted **81.6%** of the time, versus 56.9% for prior Lean search and 54.1% for GPT-4o, and it delivers **30%+** and **16%+** relative Recall@1 gains on informalized statements and proof states, respectively. Beyond a paper, we *release it as a public LLM service* at https://leanfinder.github.io/, now receiving 100+ daily HTTP requests. Practicing mathematicians in academia (CMU, MIT) and industry (XTX, DeepMind, Meta, Amazon, Mistral AI) rate it helpful, and as a *prover-agnostic* retriever it boosts LLM-based provers [Xin *et al.*, 2024; Lin *et al.*, 2025] via in-context retrieval augmentation and integration with Lean-MCP (Model Context Protocol).

2.3 Formalizing Maintainable PDE Proofs in Lean

LEAN FINDER is a retrieval engine; turning it into verified PDE analysis additionally requires *writing* the formalization and *maintaining* it as the underlying library evolves. We address these two needs in turn.

PDE Formalization in Lean4

Using LEAN FINDER to locate the relevant `mathlib` primitives, we have begun building a public Lean4 formalization of PDE theory [Sun *et al.*, 2025]. This fills the missing-primitive gap from the ground up: we formalize the *heat operator* and its *fundamental solutions*, and develop the *Galerkin approximation* that underpins existence and convergence arguments for weak solutions. Each definition and theorem is machine-checked by Lean's kernel, so well-posedness and stability properties that were previously argued by hand become *mechanically verified* artifacts. We maintain the development as an open repository so that the community can extend it toward broader classes of elliptic, parabolic, and hyperbolic PDEs.

Lean Refactor: Controllable, Version-Robust Proofs

Formalization alone is not enough: LLM-generated proofs are notoriously *correct-but-verbose* and brittle across library versions, which makes a growing PDE formalization expensive to maintain. We therefore develop LEAN REFACTOR [Lu *et al.*, 2026b], a plug-and-play, retrieval-augmented agentic framework for *multi-objective, controllable, and version-robust* proof optimization. The problem is natively multi-objective, with three competing axes: *proof length*, *compilation-time cost*, and *Lean/`mathlib` version compatibility*. These are in tension, and two structural obstacles make them hard to balance: Lean repositories have fragile compatibility (weekly releases rename declarations and break imports), while frontier LLMs are *never aligned* with a specific version since their training cutoff predates recent releases; and training-based pipelines would require repeated fine-tuning with every new model or library release. Instead of fine-tuning, LEAN REFACTOR *externalizes* refactoring knowledge into a densely annotated *strategy bank*: over 200K long–short proof pairs distilled into $\sim$9K reusable strategies, each annotated with *when* and *how* to apply it, expected compilation-cost reduction, and validated Lean/`mathlib` versions. At inference, a multi-objective retrieve-and-rerank pipeline filters strategies to the user's target toolchain and trade-off, steering a *frozen* agentic LLM (a planner–refactorer–debugger loop instantiated by any frontier model) without retraining. LEAN REFACTOR achieves over **70% token-level compression** on competition benchmarks and over **20%** on research repositories, up to **60% compilation-time reduction**, outperforming prior work and Claude Code; version-filtered retrieval further improves compression on the target version, and refactored miniF2F proofs exhibit stronger *zero-shot transfer* to future Lean releases. Together, LEAN FINDER, our Lean4 PDE formalization, and LEAN REFACTOR make formal PDE analysis *discoverable*, *writable*, and *maintainable* at scale.

3 Future Research

My future research advances scalable, interactive, human-in-the-loop AI for physical-world modeling (Figure 2).

3.1 Scalable and Interpretable SciML Models

SciML promises efficient PDE surrogates that bridge accurate but costly numerical solvers and flexible yet fragile neural operators [Li *et al.*, 2021; Kovachki *et al.*, 2023], including physics-informed approaches [Raissi *et al.*, 2019]. I pursue two thrusts. For *scalability*, building on data-efficient operator learning via unsupervised pretraining and in-context learning [Chen *et al.*, 2024], I will pretrain neural operators on large unlabeled simulations using physics-inspired re-

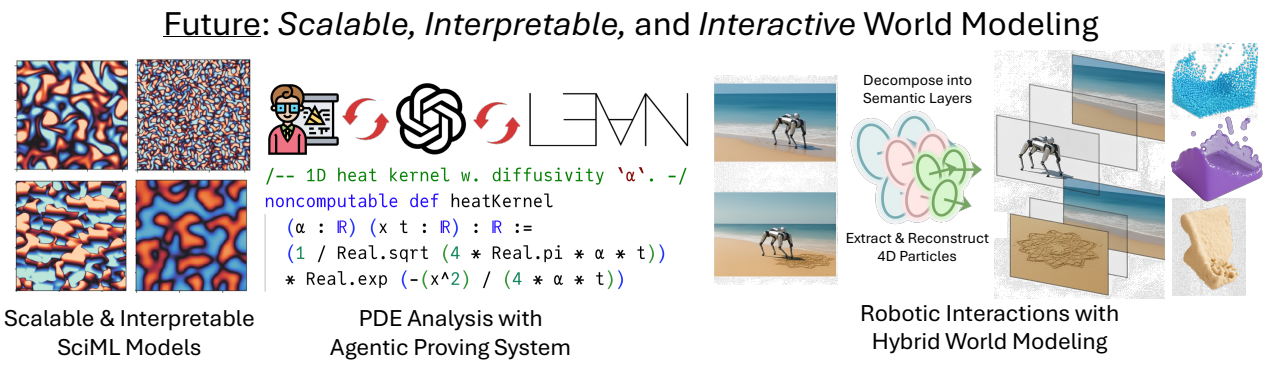

Figure 2: My future research builds *scalable, interactive, human-in-the-loop* AI for physical-world modeling: scalable and interpretable SciML, agentic proving for PDE analysis, and robotic interaction via hybrid world modeling.

construction proxy tasks, and deploy training-free, test-time adaptation that flexibly leverages similar in-context examples to cut simulation costs while avoiding fine-tuning. For *interpretability*, building on learning operators from fundamental physics knowledge [Ma *et al.*, 2026], I will decompose PDEs into compositional terms and jointly learn from both the original equations and their simplified basic forms; this multiphysics view improves data efficiency, reduces predictive error, and strengthens out-of-distribution generalization under parameter shifts and synthetic-to-real transfer across 1D/2D/3D problems. Long term, I aim to unify scale and interpretability in *PDE foundation models*: pretrained on multi-domain corpora, adapted via in-context learning, and aware of conservation laws and symmetries, while analyzing the physics-informed versus data-driven trade-off through modern learning theory.

3.2 PDE Analysis with an Agentic Proving System

Modern neurosymbolic systems pair LLMs with solvers and proof assistants, yet they often "prove the wrong thing": syntactically correct but semantically misaligned with the intended specification. I will develop *semantics-aware coding agents* for interpretable heuristics and Lean proofs with intent guarantees. On the *training* side, this combines contrastive language–code embeddings that align informal prompts with formal artifacts, fine-tuning on high-quality plan-to-code pairs, and preference optimization balancing correctness, complexity, and readability. On the *inference* side, agents will follow a "specifications first, then code" discipline: elicit and refine specifications, synthesize solutions that satisfy them, and enforce cycle-consistency between intent and formal results using Lean retrieval [Lu *et al.*, 2026a] and refactoring [Lu *et al.*, 2026b]. The target is an end-to-end, dependable proving system that autonomously formalizes and proves existence, uniqueness, and stability results for broad classes of PDEs.

3.3 Robotic Interactions with Hybrid World Modeling

Verifiable reasoning must ultimately be grounded in the modeling of real phenomena, where high-dimensional flows can be simulated, edited, and observed. Modern simulators (Habitat, Isaac Gym, MuJoCo) provide fast, controllable physics but oversimplify contacts and interactions, while generative world models (Sora, Genie-3, Cosmos) render rich, long-horizon dynamics yet drift from physical laws. My recent work already couples learning with simulation from both directions: HYBRID NEURAL MPM [Xu *et al.*, 2025] trains a diffusion model that predicts step- and particle-wise external forces to steer an in-the-loop Material Point Method, turning user sketches into stable, physically faithful 2D/3D fluid interactions (in collaboration with Meta Reality Labs); and WILDSMOKE [Liu *et al.*, 2025] reconstructs simulation-ready 4D smoke assets (density and velocity) from a single in-the-wild video, enabling realistic re-simulation and editing. FLUIDGAUSSIAN [Liu *et al.*, 2026] further propagates simulation-based uncertainty toward functionally-intelligent 3D reconstruction by tightly coupling geometry reconstruction with fluid–structure interactions, yielding up to $+8.6\%$ visual PSNR and -62.3% velocity divergence. Going forward, I propose a hybrid loop that unifies these strengths: compose a mixture of controllable world models, extract simulation-ready particle representations from video, couple particle simulation with generative models under physics guidance when motion is complex, and extend simulators with multiphysics couplings (soft bodies, fluid–rigid, thermal). To close the Sim2Real gap, I will blend reconstructed 3D/4D assets, emulate multimodal sensors, and integrate text-to-motion interfaces, working toward a *foundation simulator* for certifiable, low-latency robotic interaction.

4 Conclusion

My research establishes verifiable PDE reasoning and modeling as a unified neurosymbolic agenda. Through PDE-CONTROLLER, LEAN FINDER, and LEAN REFACTOR, I close the loop between informal PDE problems and machine-checked specifications, code, and theorems; through scalable SciML, agentic proving, and hybrid world modeling, I extend this loop toward physically grounded, trustworthy decision-making. Ultimately, I aim to build semantics-first AI systems that reason over PDEs with mathematical guarantees, surpassing human heuristics across diverse physical systems.

References

[Ahn *et al.*, 2024] Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*, 2024.

[Chen *et al.*, 2021] Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[Chen *et al.*, 2024] Wuyang Chen, Jialin Song, Pu Ren, Shashank Subramanian, Dmitriy Morozov, and Michael W Mahoney. Data-efficient operator learning via unsupervised pretraining and in-context learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:6213–6245, 2024.

[Chiang *et al.*, 2024] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, et al. Chatbot arena: An open platform for evaluating LLMs by human preference. In *International Conference on Machine Learning (ICML)*, 2024.

[Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[Dawson *et al.*, 2023] Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics*, 39(3):1749–1767, 2023.

[de Moura *et al.*, 2015] Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris van Doorn, and Jakob von Raumer. The Lean theorem prover (system description). *International Conference on Automated Deduction (CADE)*, pages 378–388, 2015.

[Kovachki *et al.*, 2023] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[Li *et al.*, 2021] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *International Conference on Learning Representations (ICLR)*, 2021.

[Li *et al.*, 2024] Zhaoyu Li, Jialiang Sun, Logan Murphy, Qidong Su, Zenan Li, Xian Zhang, Kaiyu Yang, and Xujie Si. A survey on deep learning for theorem proving. *Conference on Language Modeling (COLM)*, 2024.

[Lin *et al.*, 2025] Yong Lin, Shange Tang, Bohan Lyu, Jiayun Wu, et al. Goedel-Prover: A frontier model for open-source automated theorem proving. *arXiv preprint arXiv:2502.07640*, 2025.

[Liu *et al.*, 2025] Yuqiu Liu, Jialin Song, Manolis Savva, and Wuyang Chen. Wildsmoke: Ready-to-use dynamic 3d smoke assets from a single video in the wild. *arXiv preprint arXiv:2509.11114*, 2025.

[Liu *et al.*, 2026] Yuqiu Liu, Jialin Song, Marissa Ramirez de Chanlatte, Rochishnu Chowdhury, Rushil Paresh Desai, Wuyang Chen, Daniel Martin, and Michael W Mahoney. FluidGaussian: Propagating simulation-based uncertainty toward functionally-intelligent 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15421–15431, 2026.

[Lu *et al.*, 2023] Pan Lu, Liang Qiu, Wenhao Yu, Sean Welleck, and Kai-Wei Chang. A survey of deep learning for mathematical reasoning. *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023.

[Lu *et al.*, 2026a] Jialin Lu, Kye Emond, Kaiyu Yang, Swarat Chaudhuri, Weiran Sun, and Wuyang Chen. Lean finder: Semantic search for mathlib that understands user intents. *International Conference on Learning Representations (ICLR)*, 2026.

[Lu *et al.*, 2026b] Jialin Lu, Soonho Kong, Rodrigo Stehling, Kaiyu Yang, Zhangyang Wang, Weiran Sun, and Wuyang Chen. Lean refactor: Multi-objective controllable proof optimization via agentic strategy search. *arXiv preprint arXiv:2605.20244*, 2026.

[Ma *et al.*, 2026] Siying Ma, Mehrdad M Zadeh, Mauricio Soroco, Wuyang Chen, Jiguo Cao, and Vijay Ganesh. Learning data-efficient and generalizable neural operators via fundamental physics knowledge. *International Conference on Learning Representations (ICLR)*, 2026.

[Maler and Nickovic, 2004] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, pages 152–166, 2004.

[Moura and Ullrich, 2021] Leonardo de Moura and Sebastian Ullrich. The Lean 4 theorem prover and programming language. *International Conference on Automated Deduction (CADE)*, pages 625–635, 2021.

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

[Riedmaier *et al.*, 2021] Stefan Riedmaier, Daniel Schneider, Daniel Watzenig, Frank Diermeyer, and Bernhard Schick. Non-deterministic methods and surrogates in the assessment of an automated driving function. *Algorithms*, 14(4):101, 2021.

[Soroco *et al.*, 2025] Mauricio Soroco, Jialin Song, Mengzhou Xia, Kye Emond, Weiran Sun, and Wuyang Chen. PDE-Controller: LLMs for autoformalization and reasoning of PDEs. *International Conference on Machine Learning (ICML)*, 2025.

[Sun *et al.*, 2025] Weiran Sun, Wuyang Chen, et al. PDE formalization in Lean4. https://reservoir.lean-lang.org/@weiran-sun/PDE, 2025.

[Trinh *et al.*, 2024] Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. volume 625, pages 476–482, 2024.

[Wu *et al.*, 2022] Yuhuai Wu, Albert Q Jiang, Wenda Li, Markus N Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[Xin *et al.*, 2024] Huajian Xin, Daya Guo, Zhihong Shao, Zhizhou Ren, Qihao Zhu, Bo Liu, Chong Ruan, Wenda Li, and Xiaodan Liang. DeepSeek-Prover: Advancing theorem proving in LLMs through large-scale synthetic data. *arXiv preprint arXiv:2405.14333*, 2024.

[Xu *et al.*, 2025] Jingxuan Xu, Hong Huang, Chuhang Zou, Manolis Savva, Yunchao Wei, and Wuyang Chen. Hybrid neural-mpm for interactive fluid simulations in real-time. *arXiv preprint arXiv:2505.18926*, 2025.

[Yang *et al.*, 2023] Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan J Prenger, and Anima Anandkumar. LeanDojo: Theorem proving with retrieval-augmented language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.