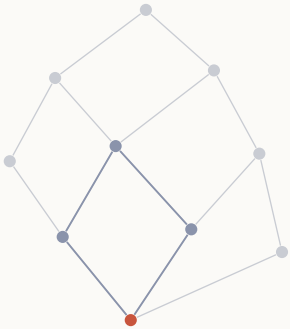
AI4MATH SEMINAR · OCTOBER 2026

Agentic Systems for Research Mathematics

Rethlas, Danus/Pharos, and OSVerify



Shurui Liu

Stanford University

Why research mathematics is different

AI already performs strongly on math competitions (IMO, Putnam). But we care much more about research level mathematics, not simply because it is actually impactful on advancing math research and reducing human effort.

Resources

arXiv and journals may contain essential errors (data); need to keep update with frontier literature (test-time learning).

Long horizon

Arguments of 5–100 pages; explorations without guarateed feedback that take hours to days.

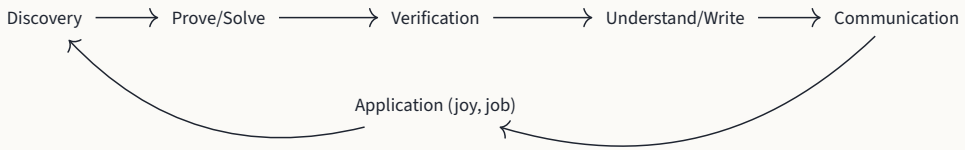
Open scope

Competition problems are solvable within a fixed syllabus. But most research problems do NOT have the properties

- "locally finite many" possible moves.
- guaranteed to be solvable.

Plan: This page is made by hand, so please forgive it is ugly.

Roughly, one can decompose mathematical research into the following loop



Today we will focus on Prove/Solve and Verification, which (used to) constitute most (visible) part of math research.

Disclaimer: Different people have different philosophy viewpoints. I only focus on the techniques and the facts to the best of my knowledge. One should always use (AI) tools with good intentions for humanity.

OVERVIEW

Plan

PART I · THREE SYSTEMS (all open-sourced) and Design Philosophy

1 Rethlas (light-weight problem solver)

A generation agent and a verification agent in a loop.

arXiv:2604.03789; open source

2 Danus/Pharos (scaled problem solver)

Orchestration of many Rethlas-style workers over a *fact graph*.

arXiv:2607.06447; open source

3 OSVerify (verification)

One-sided verification, intended to be more strict.

arXiv:2610.18089; open source

II. Why the harness matters

III. Future work

Summary. Design philosophy

PART I

Three systems

01

Rethlas, Danus/Pharos, and OSVerify: reduce human effort in problem sovling and verification

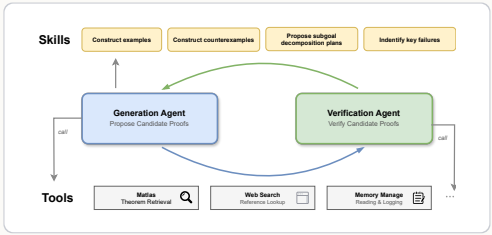
SYSTEM 1

Rethlas

A generate–verify loop to persistently attach a conjecture

Rethlas in one slide

- A **generation agent** (skills = research habits, tool = Matlas, memory = notebook) writes an informal proof *blueprint*.
- A **verification agent** referees it and returns a strict verdict with repair hints.
- The loop repeats until acceptance: zero critical errors and zero gaps.
- Built on coding-agent CLIs (Codex / Claude Code) under a supervisor script.



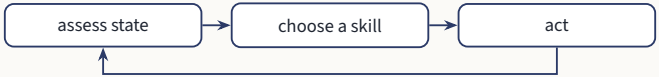
> Two design principles: **mimic the human workflow**, and **context engineering** (in particular, separate the author from the referee).

Habits become skills

What does a mathematician do to attack a hard problem?

toy examples | counterexamples | literature search | decomposition plans

direct proving | recursive proving | identify key failures



- Each **skill** mimics a human expert habbit; the generation agent loops over *assess, choose, act*.
- Competing strategies run in parallel as **one subagent**, with shared memory of failures and partial results. If all fail, their common obstructions seed the next round of plans.

Tools mimic resources; memory mimics the notebook

The library

Matlas (`matlas.ai`): semantic theorem search over ~13.6M statements extracted from arXiv. Dependencies are unfolded, so retrieved results are self-contained.

Think, then look it up

Odd iterations block search, even iterations re-enable it, to encourage deep thinking.

The notebook

Raw logs are scratch paper. Structured, append-only notes per problem (`toy_examples`, `counterexamples`, `subgoals`, `failed_paths`, `...`) are retrieved by BM25.

Failed paths are mandatory entries

Every dead end is written down and stays queryable, so it is not revisited.

The author is not the referee

- An author tends to believe their own proof; subtle gaps survive proofreading by the person who wrote them. Models behave similarly: self-checking suppresses hallucination only mildly.
- The community's answer is the **referee**: someone who did not write the proof and has no stake in it being true.
- **Generator: continuous context.** Intuition, partial progress, and dead ends persist across iterations.
- **Verifier: fresh context.** A new process per submission sees the blueprint and nothing of how it was produced.

Separate roles also get separate crafts. The verifier's ends in a strict report: every error and gap, a verdict, and repair hints.

Smallest deduction steps

Check each step for validity, applicability, and missing hypotheses.

Reference correctness

Does the cited theorem exist, and does its *statement* give what this proof needs, under these hypotheses?

An open problem of D. D. Anderson

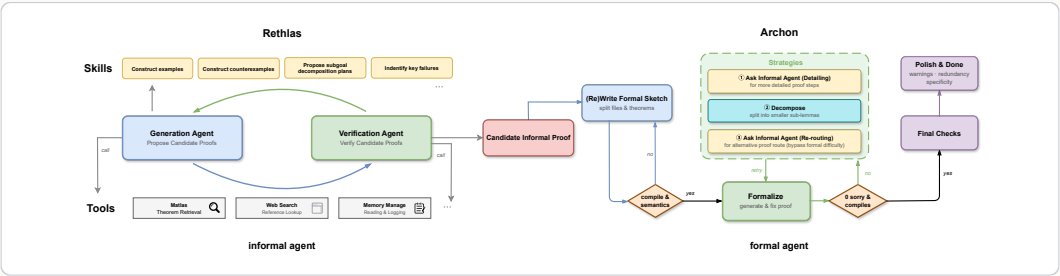
QUESTION (ANDERSON, 2014)

Does weak quasi-completeness imply quasi-completeness for Noetherian local rings?

45 min	19,448	∼80 h	26 lines
Rethlas: counterexample, no proof hints	lines of Lean 4, 42 files	Archon agent runtime	human-reviewed specification

- The key ingredient, Jensen's 2006 theorem on completions of local UFDs, was retrieved through Matlas from the generic-formal-fiber literature.
- Comparator checks that the Lean project proves exactly the 26-line specification, with only the standard axioms.

From informal to formal: Rethlas–Archon



Rethlas's generate–verify loop produces the informal proof; Archon turns it into a kernel-checked Lean 4 project.

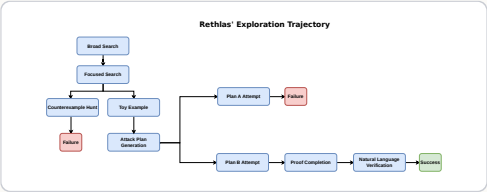
Rational conjugacy classes of cocharacters

QUESTION (SUGGESTED BY B. CONRAD)

G a connected smooth affine group over a field k, $\mathbb{X}_*(G) = \mathrm{Hom}_{k\text{-gp}}(\mathrm{GL}_1, G)$. For K/k, is $\mathbb{X}_*(G)/G(k) \to \mathbb{X}_*(G_K)/G(K)$ injective?

Special cases (split reductive, reductive, perfect k) are known; the general case is not in the literature.

- ✗ GPT-5.5 Pro (web): a wrong proof, little progress.
- ✓ Rethlas (GPT-5.4/5.5): a correct proof in 44 minutes, through the quasi-reductive case.
- One citation substitution: the needed result is in a copyrighted book the agent could not retrieve; corrected by the expert.



Rethlas results

Area	Results
Commutative algebra	Anderson's question; eight conjectures from the CFFG collection; Boij–Söderberg questions (Erman–Sam)
Algebraic groups	rational conjugacy of cocharacters
p-adic Hodge theory	refined lift-independence theorem; genus-separation counterexample
Combinatorics	counterexample to real-rootedness of matroid KL polynomials
Analysis, probability	Brezis problems (Sobolev circle maps); injectivity of phase retrieval

frenzymath/
Rethlas_results

A partial, growing list. Each result was checked by human experts before release.

Limitations of the linear loop (from Math perspective)

From viewpoint of Math:

1. A mathematician accumulates many small facts and adjusts strategy;
2. A real projects may need a research team of several collaborators on different routes. But Rethlas cannot be naively parallelized.
3. Need to improve generator capability (Danus/Pharos) and verifier capability (OSVerify).

Limitations of the linear loop (from CS perspective)

From viewpoint of CS:

naive DFS philosophy

The search is essentially DFS (no "decomposition of a problem" or "mixture of expertise").

context engineering

The blueprint, verified partial results *with their proofs*, and the notes, even when only statements are needed.

Edits collide: needs orchestration

Two agents cannot safely improve the same Markdown document.

What is needed: a modular deliverable, shared truth, and rules about who may write what (orchestration).

SYSTEM 2

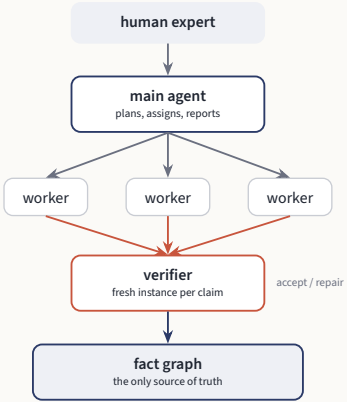
Danus → Pharos

An organization of agents over a verifier-gated fact graph

A research group of agents

This time the unit is not one mathematician but a research group.

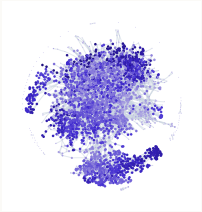
- **Group leader** (main agent): plans routes, assigns and monitors, reports. Does no mathematics itself.
- **Students and postdocs** (workers, Rethlas generation agents): each owns one claim at a time. Lower effort for breadth, higher for depth.
- **Referee** (Rethlas verifier): a fresh instance per claim; the only way into the fact graph.
- **The expert**: poses the problem, instructs, decides.



Tangent classes of matroids

PROBLEM (POSED BY R. CHENG)

Construct a tangent class $T_{M,\mathcal{G}}$ in the combinatorial K-ring of any loopless matroid, extending the tangent bundle of the wonderful compactification, with HRR and Chern-number properties. Cheng's own preprint was withheld from all systems.



Same verbatim prompt		Outcome
GPT-5.5-pro (web)	model	✗ no solution
Rethlas, 3 runs	loop	✗ 0/3 passed the verifier
Danus	organization	✓ verified solution

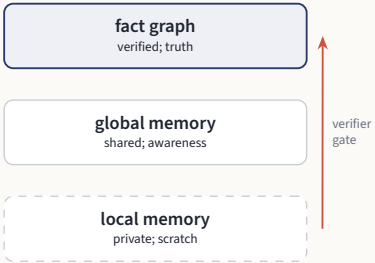
3,157
verified facts

54
deepest chain

7 workers, ~5 days (~50 h active), 13 consultations of GPT-5.5-pro; 499 facts in the theorem's closure. R. Cheng, S. Liu, G. Gao, *Tangent classes of matroids and wonderful compactifications*, arXiv:2607.05835.

Three-tier memory and the fact graph

- A worker submits **one claim at a time**: a lemma, a counterexample, a computation. A fresh verifier judges it.
- **Fact graph** (DAG): node = statement + checked proof; edge = logical dependency. The only source of truth.
- **Global memory**: plans, examples, counterexamples, dead ends, obstacles. Shared awareness, explicitly *not* truth.
- **Local memory**: each worker's own scratch paper.
- A worker loads only the facts it needs, by statement, and reads a proof only when its technique is needed.



Compaction at every boundary



What any single agent reads stays bounded, while what the system knows keeps growing.

A worker starting its tenth task inherits the lessons of the first nine by reading a summary, not the full logs.

Separation of powers, enforced by tools

At scale, separation of roles becomes separation of powers. It is enforced by the MCP tool table rather than by instructions.

	main	worker	verifier
read facts	✓	✓	✓
write memory	✓	✓	✗
`fact_submit`	✗	✓	✗
decide truth	✗	✗	✓

Human expert and main agent only steers the direction/strategy.

Worker and verifier skills are inherited from Rethlas essentially unchanged. What changed is the orchestration.

Scaling in width and depth

↔ Width

Naively adding Rethlas workers fails: they overwrite one another's blueprint. The fact graph splits the proof into units a worker can own, so contributions accumulate; the main agent diversifies directions.

↕ Depth

Per-fact verification and the memory hierarchy extend the reasoning horizon beyond one context window: chains of 54 facts.

$$0.99^{499} < 1\%$$

probability that a 499-fact closure is sound at a 1% per-fact error rate

base assumption: verifier is reliable

A verifier engineered for essentially no false positives; revocation of a wrong fact with all its dependents; summaries written under prohibitions; citations stored as data; the run stops only when the target is a verified fact; the paper itself must pass the verifier.

From Danus to Pharos

	Danus (July 2026)	**Pharos v3.1** (Sept. 2026)
Main agent	Claude Code, Opus 4.8	resident Codex session, `ultra` effort
Strategy	GPT-5.5-pro consulted every 1–2 h; reply broadcast to all workers	consultant and broadcast removed; one `TASK.md` briefing per worker
Planning	periodic summaries	prior-work survey; route board; 4-hour checkpoints graded by the expert's criteria
Expert input	ad hoc	recorded verbatim; a "do not cite" list
Workers	per-round rhythm	"a fact is a lemma": no quota; multi-round work on one result
Verifier	one shared service	per project; probes "exhaustive" case claims; demands version and hypotheses of cited papers
Writing	one writing loop	outline ⟶ per-section writer–verifier pairs ⟶ whole-paper verifier
Operations	shared homes; full scans	per-actor homes; incremental index; compute caps; monitor
Models	GPT-5.5 workers	GPT-6 Astra throughout

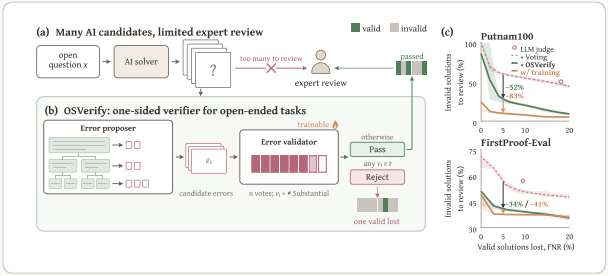
SYSTEM 3

OSVerify

Verification is now the expensive step

Motivation

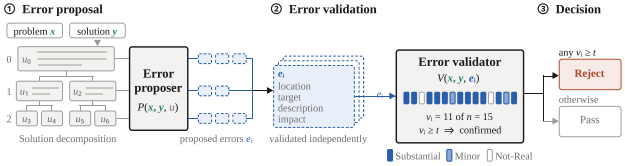
- **Math**: review is the new bottleneck, so we need to reduce human effort on referee.
- **CS**: after generation is scaled (e.g. Danus/Pharos), need to improve the verifier to avoid "DOS attack", so that the system can stably run without crashing due to false lemmas. So, scale test-time compute for the verifier.
- Goal: a verifier that *filters* for the expert. Maximize the rejection rate of invalid solutions (TNR) subject to a small false-rejection rate of valid ones (FNR $\leq \tau$).



Finding an error vs. confirming one

Mimic Human workflow: A referee first *suspects* a step, then *checks* whether it is a real, fatal error.

0.68 ⟶ 0.90
AUROC, judging solutions vs. judging proposed errors; same model, FirstProof-Eval



Error proposer. Decompose the solution hierarchically; propose errors at every granularity. Goal: coverage.

Error validator. Is each error SUBSTANTIAL, MINOR, or NOT-REAL? Repeated votes; reject if any is confirmed.

Design rationale

Scale test-time compute on verification

More proposal calls and finer decomposition raise the coverage of real errors; more validator votes lower false rejections.

Context engineering

The proposer focuses on one piece of the solution at a time, with the whole solution available. The validator sees the problem, the solution, and focuses on a single proposed error.

The same principle as in Rethlas, applied inside the verification: separate the role that searches from the role that judges. This "2nd order expansion of generate-verify" scales the compute and encourages a stricter referee.

OSVerify on benchmarks

- Putnam100, H2V, FirstProof-Eval. Reference solutions and Lean withheld from the verifier.
- Deliberately weak backbones (`gpt-oss-120b`, `gpt-5-mini`), to simulate problems beyond the model's reliable ability.
- The validator's RL training (GRPO) transfers to FirstProof-Eval, unseen in training.

TNR (%) at FNR ≤ 5%, `gpt-oss-120b`

	Putnam	H2V	FirstProof
LLM judge	0	54.4	0
Voting	38.4	69.2	35.9
OSVerify	**70.4**	**84.7**	**58.0**

−46%
review load vs. voting

−57%
with validator training

PART II

Why the harness matters

02

With GPT-5.6 and GPT-6, the harness still separates success from failure

Kazhdan–Lusztig polynomials of matroids

CONJECTURES

$P_M(t)$ is real-rooted (Gedeon–Proudfoot–Young, 2017) and log-concave (Elias–Proudfoot–Wakefield, 2016); evidence from uniform, q-niform, fan, wheel, whirl matroids.

Prompt: the definition of P_M, and "prove or disprove that all zeros of $P_M(t)$ lie on the negative real axis."

✔ Rethlas (GPT-5.6 Sol, `max`): explicit counterexample in 4 h 59 min, by deleting points from a finite projective geometry.

✖ GPT-5.6 Sol Ultra (web), two prompts: incomplete.

Then the humans

Cheng and Liu developed the method (Thm1), provided geometric interpretation (Section 3) and proved P_M need not be **unimodal**. This refutes both conjectures at once.

R. Cheng, S. Liu, *Kazhdan–Lusztig polynomials of matroids need not be unimodal*, arXiv:2607.24186.

Rational curves via jets

QUESTION (DU–GUO–XIE, APPENDIX F)

For smooth projective X in characteristic zero and a curve $f : C \to X$, does $-K_X \cdot f_*[C] > 0$ alone give a rational curve through every point of $f(C)$, under a strict degree bound?

19 h 57 min
to the verified target, no human intervention

273
verified facts; 1 revoked and re-proved

11 h 38 min
with expert hints at launch; same proof

>300k
lines of Lean 4, within a week

- Result: an algebraic, characteristic-zero proof of the Miyaoka–Mori criterion via weighted jets, with an effective decomposition of the curve class.
- GPT-6 Astra throughout: main agent `ultra`; 3 workers `xhigh`, 4 `max`. Manuscript written and checked in a further 1 h 52 min.

B. Dong, G. Gao, B. Guo, Z. Sun, B. Wu, S.-Y. Xie, arXiv:2609.28465, Appendix B.

OSVerify on research mathematics

AI-generated papers

11 recent OpenAI preprints on central problems (Kakeya in dimension 4, planar unit distances, trilinear Hilbert transform, quantum geometric Langlands, …). Mistakes found in each, overlooked by OpenAI's internal review, confirmed by experts.

Human papers

Papers published in major journals: errors found, repairs suggested, reported to authors.

Solving, with Rethlas

Automorphic commutator relation (Rewritten by Liu). Flatness of the loop-rotated Plancherel algebra, commutativity of Plancherel algebra (expected to be doable by experts; checked not very carefully by Liu).

F. Nie, H. Ye, S. Liu, Z. Wang, S. Ermon, J. Zou, arXiv:2610.18089; S. Liu, arXiv:2610.05781.

The Manin constant

THEOREM (CHI–LIU–NIE, arXiv:2610.25672)

The Manin constant conjecture is true. The Stevens' Conjectures I and II are true.

Base models, expert prompting

ChatGPT-6 Pro, Claude Fable 5.1, Codex 6-Astra first proved only $c_\varphi \mid \deg \varphi$. Reaching $c_E \mid 2$, then $c_E = 1$, took many rounds of prompts from the authors.

Review

12 independent checks (Codex 6-Astra, Claude Opus 5.5) passed. OSVerify found one mistake: the claim $L_E \in \mathbb{Q}[[w]]$ for every $E/\mathbb{Q}_p$. It suggested a fix for the three affected places.

Harness

Rethlas (GPT-6 Astra) with OSVerify, given only the statement, reproduced essentially the same proof, rejecting ineffective strategies automatically.

Key idea: modularity of the normalized lift via Calegari–Dimitrov–Tang. The logs show no access to the closest prior use (Yin, 2026), but pretraining exposure cannot be excluded.

SUMMARY OF THE COMPARISONS

Base model vs. harness

Problem	Model	Base model alone	With harness
Cocharacters	GPT-5.5	✗ wrong proof	✓ Rethlas: correct, 44 min
Matroid tangent classes	GPT-5.5	✗ no solution	✓ Danus: solved (Rethlas 0/3)
KL polynomials	GPT-5.6	✗ incomplete	✓ Rethlas: counterexample, 5 h
Manin, solving	GPT-6	✗ many expert prompts	✓ Rethlas + OSVerify: statement only
Manin, review	GPT-6	✗ 12 AI checks passed	✓ OSVerify: one error found with GPT5.6Sol
OpenAI 11 preprints	Bel	✗ mistakes overlooked	✓ OSVerify: found mistakes with Astra

The models are the same or comparable. What differs is the harness.

PART III

Future work

03

Simplification, understanding, and open problems

FUTURE WORK

Simplifying the harness as models improve

Components that compensate for weaknesses

Optimistic summaries, prompt-shaped language, retrieval spirals. Danus needed an outside strategist (GPT-5.5-pro); with GPT-6 Astra the main agent plans on its own, and the consultant was removed.

Components that encode structure

Separation of roles, verifier-gated truth, bounded context per agent. These do not obviously depend on model strength.

Open question: which parts of the harness remain essential for the next model generation?

FUTURE WORK · ONGOING

From verification to understanding/writing

- AI systems now produce correct proofs faster than mathematicians can read them. A verified fact graph of 3,000 facts, or a 300k-line Lean library, is correct but not yet *understood*.
- Mathematics is a human activity of understanding: why a proof works, which idea is essential, how it connects to other results.

Ongoing project

AI that helps humans understand mathematics.

Joint with Nancy Jiang, Travor Liu, Zhuo Zhang, Fred Rajasekaran, et al.

FUTURE WORK

Other directions

Creativity and theory building

Current systems work largely within the "convex closure of known techniques" (WARN: not well-defined!). The strict verifier makes workers take small steps; non-rigorous exploration, which may favor new ideas, rarely happens.

Human-AI collaboration

Improve writing skills and communication skills of agents.

Evaluating agentic systems

From 30% to 80% is easy, 80% to 90% is hard, 90% to 99% is very hard. Benchmarking is still largely empirical.

SUMMARY

Design philosophy

Discovery → **Problem Solving** → **Verification** → *Understanding* → Education/Communication → Application.

MATHEMATICAL HEURISTICS

Mimic the human workflow

Habits become skills, resources become tools, the notebook becomes memory (Rethlas); at scale, mimic a research group (Danus/Pharos).

Separation of roles

The author is not the referee (Rethlas); separation of powers through capabilities (Danus/Pharos); suspecting an error is separate from confirming it (OSVerify).

CS HEURISTICS

Context engineering

Fresh context for judgment; bounded reads for workers; compaction at every boundary; one source of truth; one error per validation call.

Scaling test-time compute

Iteration (Rethlas); width and depth (Danus/Pharos); verification (OSVerify). All rest on a verifier with essentially no false positives.

References

SYSTEMS

H. Ju, G. Gao, J. Jiang, B. Wu, Z. Sun, S. Liu, et al. *Automated Conjecture Resolution with Formal Verification*. arXiv:2604.03789.

J. Liu, G. Gao, Z. Sun, B. Wu, S. Liu, et al. *Danus: Orchestrating Mathematical Reasoning Agents with Fact-Graph Memory*. arXiv:2607.06447.

F. Nie, H. Ye, S. Liu, Z. Wang, S. Ermon, J. Zou. *One-Sided Verification Under Limited Expert Review*. arXiv:2610.18089.

Code: `github.com/frenzymath/{Rethlas, Archon, Pharos}`

MATHEMATICS

R. Cheng, S. Liu, G. Gao. *Tangent classes of matroids and wonderful compactifications*. arXiv:2607.05835.

R. Cheng, S. Liu. *Kazhdan–Lusztig polynomials of matroids need not be unimodal*. arXiv:2607.24186.

B. Dong, G. Gao, B. Guo, Z. Sun, B. Wu, S.-Y. Xie. *Constructing rational curves via jets on projective varieties with non-nef canonical bundle*. arXiv:2609.28465.

Z. Chi, S. Liu, P. Nie. *On the Manin constant*. arXiv:2610.25672.

S. Liu. *Automorphic commutator relations in relative geometric Langlands*. arXiv:2610.05781.

Thank you

Shurui Liu · Stanford University