

未来的数学形态*

亚历克斯·孔托罗维奇 (Alex Kontorovich)[†]

摘要。 本文概述了某些计算工具目前如何与数学实践相互作用，并思考了在数学领域进入新兴的人工智能 (AI) 和形式化验证系统时代之际，这些工具对短期到中期研究数学的影响。

1 引言

本文将讨论我对我们似乎正进入的 AI 和形式化时代的科研数学可能呈现之面貌的愿景。鉴于进步的速度，我的话语可能在它们出现的瞬间就会过时，但我仍将冒着风险，记录下我在 2025 年 9 月所看到的景观。

我自己参与形式化的工作始于大约五年前（见 [9]），这是由我研究领域——解析数论——的特殊挑战所驱动的。解析数论的特点是论证过程特别长且具有技术性，往往跨越数十页。这些证明通常综合了许多引理，每个引理仅在其参数的受限范围内有效，而这些参数必须在最后完美对齐，主定理才能成立。这种参数与约束之间错综复杂的“舞蹈”既是我热爱该学科的原因，也是其危险所在：埋藏在这些技术性论证中间的一个微小的负号错误，就可能使整个主要结果失效。

当论证涉及复杂的代数运算时，我可以像使用像 *Sage* 或 *Mathematica* 这样的计算机代数系统来校验一次并一劳永逸；这样我就不再浪费时间反复检查该部分的论证以确保没有错误。难道不应该有类似的软件，来追踪我整篇论文中的逻辑依赖关系和参数约束吗？

这类软件确实存在，被称为“交互式定理证明器 (Interactive Theorem Prover)”；见 §7。原则上，它允许用户将他们的论文“形式化”，逐行输入证明，直到计算机认证该论证是完整的。然而，现实是令人清醒的。就目前情况而言，我甚至无法形式化地**陈述**大多数我感兴趣的定理，更不用说证明它们了。现有的形式化库（见 §7.2）在将其变为现实之前，需要增长几个数量级。

这导致了一个根本性的难题：新数学似乎以指数速率增长，设为 δ ，而形式化库也以指数速率增长，设为 ϵ ；但据我所知，目前的情况是：

$$\delta > \epsilon$$

如果这种不等式持续存在，形式化将**永远无法**赶上自然语言数学研究，而让软件像计算机代数系统辅助符号运算那样辅助“推理”的梦想，将永远只是一个遥不可及的幻影。

除非，形式化可以通过自动化来“增强”。因此，我已经从想要“做研究数学”转变到“需要足够的形式化来做研究数学”，再到“需要足够的 AI 来进行足够的形式化来做研究数学”！接下来是我

对于我们如何达到那个目标，以及当我们达到时情况会是如何的愿景。那句古老的格言——“预测很难，尤其是关于未来的预测”（常被归功于约吉·贝拉、尼尔斯·玻尔等人）——在这里非常贴切。因此，在开始之前，让我们先退后一步。

2 在 2000 年...

在千禧年之交，一些世界上最著名的数学家聚集在特拉维夫，讨论 21 世纪的数学可能是什么样子。会议记录发表在...

*题目的灵感来源于奥内特·科尔曼 1959 年的专辑《未来的爵士乐形态》(The Shape of Jazz to Come)。

†罗格斯大学数学系 (alex.kontorovich@rutgers.edu, <http://math.rutgers.edu/~alexk>)。部分受 NSF 资助 DMS-2302641、美以 BSF 资助 2020119 以及西蒙斯基金会奖学金支持。

【高三解读】

高三数学与跨学科深度解读

1. 数学模型的应用：增长率博弈

文中提到了一个非常经典的高三数学模型——**指数增长**。作者用 δ 代表新数学研究的增长率， ϵ 代表形式化工具的增长率。结论是 $\delta > \epsilon$ 。对于高三学生来说，这不仅是理解不等式，更是理解“追及问题”在抽象领域的体现。如果 $\delta > \epsilon$ ，两者之间的差距 ($\delta - \epsilon$) 会随着时间 t 呈指数级扩大。这告诉我们，仅靠人力积累（线性或低阶指数增长）无法解决信息爆炸的问题，必须寻求算法的“升维”突破。

2. 解析数论的严谨性与“负号陷阱”

作者提到“一个微小的负号错误可能使整个结果失效”。这在高三数学考试中极具共鸣。在处理数列求和、导数大题或解析几何繁琐计算时，一个正负号的疏忽会导致后续满盘皆输。这不仅是计算能力的要求，更是对数学思维**完备性**的要求。文章中提到的“形式化验证”，本质上就是追求逻辑的零失误，这正是数学学科的核心魅力与挑战。

3. 计算机代数系统 (CAS) 与逻辑推理

高三学生可能接触过 *Geogebra* 或图形计算器，它们能处理符号运算和图像。但作者区分了“运算 (Manipulation)”与“推理 (Reasoning)”。目前 AI 在运算上很强，但在严密的逻辑链条（即证明题）上仍处于起步阶段。这引导学生思考：数学不仅仅是套公式计算，更是严密的逻辑推演。

4. 学科交叉视野：AI + 数学

这篇文章展示了前沿数学家如何思考 AI。对于面临专业选择的高三学生，这提供了重要参

考：未来数学的研究不再是“纸笔”孤岛，而是与计算机科学深度融合。理解 $\delta > \epsilon$ 的现实意义，能帮助学生建立更宏观的科学史观，理解为何“自动化证明”是数学界亟待攻克의堡垒。

5. 词汇与表达亮点

- **Formalization (形式化)**：将自然语言转换成逻辑严密的机器可读代码。
- **Conundrum (难题/困局)**：数学研究中常见的逻辑悖论或发展瓶颈。
- **Exponential rate (指数速率)**：理解 $y = a^x$ 在现实增长模型中的主导地位。

第 2 页

【原文翻译】

GAFA特刊名为《数学愿景》，我强烈建议阅读整期内容 [1]。其中一篇极具先见之明的贡献是由蒂姆·高尔斯（Tim Gowers）撰写的，题目为《粗结构与分类》[11]。

高尔斯论文的第 2 节具有挑衅性的标题：“2099 年还会存在数学吗？”。在文中，他想象了“两到三十年后数学家与计算机之间的对话”——瞧！他想象的对话与我们现在许多人在与 OpenAI 的 GPT、Google 的 Gemini、Anthropic 的 Claude、xAI 的 Grok 等“聊天机器人”互动时经常获得的体验惊人地相似：

数学家：以下命题是否成立？令 $\delta > 0$ 。对于足够大的 N ，每个大小至少为 δN 的集合 $A \subset \{1, 2, \dots, N\}$ 都包含一个形如 $\{a, a + d, a + 2d\}$ 的子集？

计算机：是的。如果 A 非空，选择 $a \in A$ 并令 $d = 0$ 。

数学家：好吧，好吧，但如果 d 不允许为零呢？

计算机：你尝试过对 N 使用归纳法吗？令 $\delta = \delta(N)$ 趋于零？

数学家：那个想法一点帮助都没有。请给我一些例子。

计算机：显式的贪心算法给出的集合为

$$\{1, 2, 4, 5, 10, 11, 13, 14, 28, 29, 31, 32, 37, 38, 40, 41, \dots\}。$$

我注意到该集合的大部分是其他部分的平移。事实上，这个集合非常像康托尔集（Cantor set），因此这给出了 $\delta \geq N^{(\log 2 / \log 3) - 1}$ 的界限。

图 2.1：摘自 [11] 的一个片段

高尔斯指出：“对话的初衷是计算机对数学家非常有帮助，同时又不会做任何‘特别聪明’的事情”（强调为原文所加）。所以，沿着这个思路：让我们建立一些共识，并明确我所说的“大语言模

型”(LLM) 指的是什么。

3 什么是 LLM?

我们都有过这样的经历：在看书时，读到一页底部恰好是一句话的中段，当你手指摸索着吃力地翻到下一页时，你的大脑禁不住会超前思考，补全句子可能的后续。例如：

（图片文字：……担心接下来会发生什么。他仿佛已经在通往这一时刻的道路上奔驰了数周、数月，甚至数年，但现在他突然停了下来，无路可走。

那里肯定还有其他的魂器，但他甚至不知道它们可能在哪里。他甚至不知道它们到底是什么。与此同时，他不知该如何……）

图 3.1:《哈利·波特与死亡圣器》(J. K. Rowling 著) 的一页底部

我们读到：

“[哈利] 甚至不知道所有的 [魂器] 是什么。与此同时，他不知该如何去……”

如何去“干什么”？你的大脑本能地开始生成可能性。如何去“摧毁”(destroy) 魂器？或者可能是如何去“寻找”(find) 它们？或者，虽然可能性小得多，如何去“夹住”(sandwich) 它们？

我想象自己站在《家庭问答》(Family Feud) 的比赛现场，听到主持人史蒂夫·哈维宣布：“调查显示……”但这里的“调查”并非询问 100 个随机的人——它采样了所有可用的人类文本！结果可能是这样的：给定语境，大约 20% 的人会用“destroy”来接龙，另外 7% 的人会用“find”，也许极少数的 0.003% 会建议“sandwich”。

【高三解读】

作为一名高三教师，这段文字不仅涉及前沿的 AI 技术，还蕴含了深刻的数学思维和逻辑推演，非常适合作为跨学科学习的素材。以下是深度解读：

1. 数学知识点：等差数列与 Szemerédi 定理

对话中数学家提出的问题是数学中著名的 **Roth 定理**（关于长度为 3 的等差数列）。

- **核心概念**：集合 $A \subset \{1, 2, \dots, N\}$ 且其密度 δ 恒定。我们要找的是 $a, a + d, a + 2d$ ，这本质上是一个公差为 d 的等差数列。
- **贪心算法与康托尔集**：计算机提到的集合 $\{1, 2, 4, 5, \dots\}$ 实际上是去掉了包含 3 的倍数特性的构造（类似于三进制中不包含数字 1 的数），这与高三学到的康托尔集构造逻辑一致。康托尔集具有自相似性，这也是文中提到“平移”和“分形”特征的原因。
- **公式理解**： $\delta \geq N^{(\log 2 / \log 3) - 1}$ 。这里涉及到对数换底公式和幂函数的增长速度分析，是高三导数与不等式部分的进阶背景。

2. 解题技巧：分类讨论与反例构造

- **陷阱识别**：当数学家问一个通用命题时，计算机给出了 $d = 0$ 的特例。在数学竞赛或高考填空题中，“公差是否为 0” 是等差数列题目中最常见的陷阱。这提醒学生：在推导公式前，务必检查定义的边界条件。
- **构造性思维**：面对复杂的证明，尝试用贪心算法构造一个满足或不满足条件的集合，是寻找反例或理解题目特征的重要手段。

3. 学科背景：LLM 的本质是概率预测

- **概率思维**：LLM（大语言模型）本质上不是在“思考”，而是在做“下一个词的概率预测”（Next Token Prediction）。文中通过哈利·波特的例子，生动地展示了语言模型如何基于海量文本统计概率。在解决概率大题时，我们也要学会这种“给定已知条件（语境），推导未知结果（后续词）”的逻辑，即**条件概率** $P(A | B)$ 。
- **人工智能与数学**：高尔斯的预测正在变为现实。现在的 AI 能够辅助证明，但它最初的表现往往像文中那样“看似有用实则平庸”。这启发学生：未来的学习方向应从简单的计算和记忆转向**提问的艺术**（Prompt Engineering）和**逻辑纠错能力**。

第 3 页

【原文翻译】

大语言模型（LLM）的核心本质是一个函数，其输入是“上下文”（context）——一个可能很长的“标记”（tokens，包括单词、标点符号等）序列；而其输出并非如人们通常误解的那样是下一个单词，而是所有可能出现的下一个标记的概率分布。

图 3.2：LLM 是一个从上下文（标记序列）到潜在下一个标记的可能性分布的函数。

出于我们的研究目的，我们可以暂时不考虑使这种函数成为可能的那些令人难以置信的工程技术：架构（transformers）、数据表示（高维向量空间中的嵌入）、训练过程（梯度下降、预训练、微调）等等。我们只需将这种函数的存在视为既定事实。

前文提到的聊天机器人正是建立在这样一个模型之上的；机器人必须决定如何从模型输出的概率分布中选择实际的下一个标记。这种选择取决于“温度”（temperature）等参数——也许总是选择最可能的标记（这往往会产生略显生硬的文字），或者从分布中进行随机采样，以及介于两者之间的各种选项。一旦选定了一个标记，机器人就会将其附加到上下文中，并将整个序列重新输入模型；然后选择下一个标记，接着是再下一个，依此类推，直到它选择了一个表示“停止对话”的特殊标记。（我们稍后会讨论更复杂的代理行为，如工具使用和自我导向的工作流程；目前，我专注于“纯聊天机器人”的机制。）正是这种纯粹的聊天机器人行为，可能是高尔斯在与你的讨论中所设想的“并没有做任何特别聪明的事情”的计算机程序。

这里的一个关键观察是，这个过程是深刻的随机性的（stochastic）。在

每一步中，机器人都会做出随机选择，而每一个选择都会影响后续所有选择的概率分布。随机性随着每一个标记而复合——随机之上叠加随机。这种随机性既是系统巨大的力量源泉——使它能够以非凡的通用性应对极其广泛的问题——但正如我们将看到的，当应用于数学证明等确定性问题时，它也会带来根本性的挑战。

随机性问题 vs. 确定性问题。

考虑近似正确与精确正确之间的区别。一首长诗中间有一个用词不当，依然可以感人至深；即兴爵士乐演奏中出现一个偶然的不协和音符也是如此。在这些领域中，99% 的成功代表了真正的成就。但是，对于一个正确率为 99% 的数学定理证明呢？一个单一的错误就可能使随后的所有内容变得毫无意义！数学需要确定性的正确，而非概率性的近似。

为了定量理解其重要性，假设一个 LLM 在每一个单独的步骤中都达到了 99% 的准确率——这是一个非常高的成功率。如果一个数学论证需要连续进行 1000 个这样的步骤，且每一步的成功都相互独立，那么整体成功的概率是多少？答案是： $0.99^{1000} \approx 0.004\%$ 。一个在每一步看似高度可靠的过程，在经过多步复合后，几乎肯定会出错。

人类大脑显然也以两种截然不同的模式运行，丹尼尔·卡尼曼在《思考，快与慢》中将其普及为“系统 I”与“系统 II”。前者是快速、自动且不费力的；后者是缓慢、深思熟虑且费力的。当被问到“ $8 + 3$ 是多少？”时，大多数成年人会立即回答——这是系统 I 的处理。你已经遇到过这个计算太多次了，以至于你的大脑基本上记住了答案。（这就是为什么孩子们学习乘法表如此重要：它将曾经费力的系统 II 计算转化为自动的系统 I 回忆，为处理更复杂的任务释放认知资源。）

但当我用纸笔计算 437×82 时，我需要系统 II 的处理。然而，因为我学会了标准算法，我可以将这个问题分解成一长串简单的系统 I 计算：首先计算 $2 \times 7 = 14$ ，写下 4，进位 1，依此类推。关键的洞察是，系统 II 的数学推理通常包含小心地将许多系统 I 步骤链在一起，其中成功要求链条中的每一个环节都必须正确。

【高三解读】

作为高三学生，这段文字不仅揭示了人工智能的底层逻辑，更深刻地映射了数学学习和备考中的核心痛点：

1. 概率复合与“粗心”的代价：

文中提到的公式 $0.99^{1000} \approx 0.004\%$ 是对“细节决定成败”最震撼的数学诠释。在数学考试中，一个大题往往包含十几个步骤，如果你每一步的正确率是 90%，连续十步后的总正确率就只剩约 35%。这告诉我们，数学没有“差不多”，确定性（Deterministic）是数学的灵魂。

2. 系统 I 与系统 II 的博弈：

卡尼曼的理论在考试中非常实用。许多同学在做基础题（系统 I）时因为过于自信而产生“幻觉”错误，而在处理难题（系统 II）时，由于基础算法不够熟练，导致认知负荷过重而卡壳。高三复习的目标之一，就是通过大量练习将复杂的系统 II 过程（如通分、导数基础公式、简单几何结论）内化为系统 I 的本能，从而为解决综合性大题预留足够的“脑力”。

3. 大模型的本质与学习启示：

LLM 本质上是“概率预测机”，它根据语境寻找最可能的下一个词。这其实像极了某些同学在做选择题时靠“语感”或“直觉”蒙答案。但在严谨的理科思维中，这种“随机性”（Stochastic）是致命的。真正的学习应该从“觉得这道题大概选 A”进化到“根据逻辑链条，这道题只能选 A”。

4. 逻辑链条的完整性：

数学论证像一根链条，其强度取决于最弱的那一环。在平时练习中，不要满足于“思路对了”，必须通过精准的计算和严密的表达，确保每一个“标记”（步骤）都百分之百正确。

第 4 页

【原文翻译】

这带我们回到了我们的随机性问题。如果你的“第一系统机器”在每一步都有 99% 的准确率，但你需要链接 1000 个步骤来解决一个第二系统问题，那么你将面临前述的 0.004% 的成功率。这就是为什么使用随机工具进行确定性数学研究具有根本性挑战的原因。

然而，这种论点假设 AI 的能力将保持静态。考虑到近年来非凡的发展速度，或许这些限制只是暂时的？

5 AI 能力正在迅速推进.....

持续快速进步的证据令人信服。考虑在国际数学奥林匹克（IMO）上的表现，该竞赛被广泛认为是针对高中生的最具挑战性的数学竞赛之一：

- 在 2023 年，能够获得 IMO 单分的 AI 系统数量为.....零。
- 在 2024 年，Google DeepMind 的 AlphaProof + AlphaGeometry 实现了银牌水平的表现，离金牌仅差一分 [10]。
- 到 2025 年，多个 AI 系统获得了金牌分数，其中大约一半使用了形式化验证方法，另一半纯粹使用自然语言。¹

这一轨迹是惊人的。在两年内，我们从完全无能跨越到了金牌水平。此外，2025 年纯自然语言方法的成功证明，形式化验证对于解决困难的数学问题并非绝对必要——至少在 IMO 级别是这样。

到 2026 年夏天的下一次 IMO，可以合理预测，参加 IMO 的领先 AI 系统数量将回归为零——不是因为它们退步了，而是因为竞赛对它们来说将变得轻而易举。（开源模型可能会继续参与，但商业系统很可能会发现这低于它们的能力水平。）

这种能力加速的模式并非数学领域所独有。它反映了理查德·萨顿（Richard Sutton）著名的“苦涩教训”（Bitter Lesson），即总结了七十年 AI 研究的心得：由人类专家精心打造的领域特定解决方案，总是会被利用不断增长的计算资源的通用方法所超越 [23]。一次又一次，我们看到扩大计算规模和数据量胜过聪明的工程技巧。

如果摩尔定律继续有效——甚至只是减速而非停止——我们可以预期大语言模型（LLM）能够维持越来越长的推理链。今天在 100 步论证中挣扎的系统，明天可能会处理 1000 步论证，后天则是 10,000 步。从高中数学到本科水平、博士水平，再到研究级数学的转变，可能仅仅需要将持续推理能力扩大若干倍。

事实上，人们可以类比国际象棋评分：2500 ELO 评分的象棋特级大师可能对应于一名刚获得数学博士学位的人；2600 对应于一名资深研究员；2700 对应于一名领先的专家；而 2800+ 对应于数学巨匠。如果 AI 系统能够达到精英高中数学所需的 2300 等效水平，为什么它们最终不会达到 2800 及以上呢？

因此，也许我一直强调的随机/确定性张力会随着系统变得足够强大、能维持任意长的推理链而消解。也许自然语言 AI 最终会足以应对所有数学问题？在我看来，即使是这种乐观的情景也未能解决一个根本性问题。

6但即使完美的扩展也是不够的

让我们乐观地想象，AI 继续以目前的速度进步。假设在十年内，我们的系统能够产生原创数学研究——不仅是解决竞赛题目，而是发现新定理、开发新理论以及撰写推动人类知识进步的论文。

考虑一个未来每天能生成 100 篇研究论文的 AI 系统。为了让情景尽可能有利，假设这个系统已经达到了非凡的可靠性：这 100 篇论文中有 99 篇是完全正确的，具有严谨的证明和对数学的重要贡献。100 篇论文中只有 1 篇包含错误——也许是引理中的细微错误，或者是未正确处理的一种情况，或者是埋在计算某处的正负号错误。

¹ 虽然 DeepMind 使用了通用的自然语言模型，但它仍然针对几何问题调用了 AlphaGeometry 类型的系统。

【高三解读】

高三数学与跨学科深度解析

这是一段探讨人工智能（AI）在解决高难度数学问题（如 IMO）中的进展及其局限性的深刻论述。对于高三学生来说，这不仅是科技前沿，更包含了重要的数学思想和备考启示。

1. 概率累积的“致命陷阱”：随机性 vs. 确定性

文章开头提到的 $0.99^{1000} \approx 0.004\%$ 是一个典型的**概率乘法原理**应用。在数学证明中，逻辑链条中的任何一环出错，整个结论就失效。这提醒同学们：

- **解题启示：**在大型综合题（如解析几何或导数大题）中，每一步的计算务必精准。如果你在每个推导环节（共 10 步）只有 90% 的准确率，最终得分概率仅约 35%。数学不相信“大概”，只相信绝对的严谨。

2. “苦涩教训”与通法的重要性

文章引用了“Bitter Lesson”，指出**通用方法（Scaling）**往往胜过**奇巧淫技（Clever tricks）**。

- **备考建议：**在复习中，不要过分沉迷于极个别题目中极其冷僻的解题技巧。高考更偏向考察“通性通法”。通过大量的练习（数据量）和逻辑思维的反复训练（计算资源），你建立的数学直觉（System I）和严密推理能力（System II）才是提分的王道。

3. IMO 等级与数学能力的阶梯

文中将数学水平与国际象棋 ELO 分分级，清晰地展示了从竞赛生到科学家的路径。对于志在强基计划或竞赛的同学，这提供了一个视角：

- **2300 分段（IMO 水平）：**属于精英高中生，核心在于解决“已知有解”的难题。
- **2800+ 分段（数学巨匠）：**核心在于“创造新知识”。
这告诉我们，刷题是基础，但真正的数学高峰在于对未知规律的探索。

4. “完美扩展”下的隐患：容错率的危机

即使 AI 达到了 99% 的正确率，在大规模产出（每天 100 篇论文）时，那 1% 的错误也会成为致命伤。

- **思维陷阱：**在考试检查阶段，我们往往会忽略“微小错误”（如文中提到的引理错误、分类讨论不全或正负号写错）。在数学科研和高考试卷中，这些“小病”往往导致“大灾”。养成**形式化验证（回代检验、极限位置检验）**的习惯至关重要。

5. 跨学科背景：AI 的本质

文中提到的 **AlphaProof** 和 **AlphaGeometry** 代表了两种路径：逻辑推导与几何直觉。这对应了数学学习的两大支柱——代数逻辑与空间想象。AI 的飞速进步预示着，未来数学家的工作将更多地转向“提出问题”和“验证逻辑”，而非单纯的计算。

第 5 页

【原文翻译】

你会依赖这样一个系统吗？我**不会**。

为了有效地使用这个工具，我需要通读所有 100 篇论文，以确定哪 99 篇是正确的，哪 1 篇是有缺陷的。但是，验证一篇数学论文的正确性，并不一定比最初撰写它容易多少，特别是对

于前沿研究而言。虽然挖掘该工具输出的有趣想法、新颖方法或意外联系肯定是有价值的；然而，我不能将定理本身视为可以据以构建的既定事实。我需要花费一整天、每一天的时间来检查这些论文，结果却发现我大部分的精力都花在了验证那些原本就是正确的工作上，而那篇唯一的瑕疵论文如果错误足够微妙，甚至可能在未被察觉的情况下蒙混过关。

有人可能会反对说，这种假设的 99% 准确率已经优于当前的已发表文献了！毕竟，大多数数学家都遇到过带有错误、论证漏洞或将“细节留给读者”的情况。这确实是事实。但有一个关键的区别：对于人类撰写的论文，我们可以培养出关于可靠性的直觉。正如凯文·巴扎德（Kevin Buzzard）最近写道的 [3]：“大语言模型（LLMs）会非常乐意掩盖论证中的任何或所有漏洞，甚至撒谎以达到结果的要求，因此，如果一篇论文不是由相信自己所写内容是真实的人生成的，而是由相信自己所写内容是似是而非的东西生成的，我阅读它的动力就会大大降低。”（强调为原文所加。）

我们通过多重信号来培养关于可靠性的直觉：表述的质量、技术是否以标准方式应用、边界情况的处理程度、在已知结果的情况下断言是否显得合理等等。一位默默无闻的数学家的精彩证明之所以能被立即识别出来，正是因为我们可以评估这些品质。（仅举一例，见 [27]。）由于 AI 从单一来源每天产生 100 篇论文，这些评估启发式方法无法提供区分；每一篇论文都必须被同等程度地怀疑。

此外，即使是对于那 99 篇正确的论文，如果没有经过验证，我也无法提取它们的全部价值。一个美丽的定理可能会暗示研究的新方向，但在我自己检查证明之前，我无法自信地在此基础上进行研究。这些想法可能值得挖掘，猜想可能值得追求，但定理本身在验证之前仍然是不确定的。既然如此，为什么不直接由我自己来推导这些数学呢？

这不是通过规模化就能解决的问题。即使我们将准确率提高到 99.99% 或 99.999%，根本问题仍然存在：对于数学而言，非零的错误率会创造出一个无法使用的工具。如果我们无法在不经过验证的情况下区分正确结果与错误结果，那么这就违背了自动化的整个目的。

将此与另一种场景对比：一个 AI 每年仅产出 10 篇论文，但每一篇都附带一个**形式化验证的证明**。此外——这是至关重要的——假设我已经学会了很好地阅读形式化陈述，足以验证它们准确表达了我想让它们表达的内容，没有误译或误解。那才是我会满腔热忱使用的工具。

区别在于确定性认证。我可以完全信任这 10 篇论文，自信地建立在它们之上，并将它们的结果融入我的工作中，而不必担心隐藏的错误。较慢的节奏不仅是接受的——它是一个特性，而不是一个缺陷。在构建累积的数学知识体系时，质量和确定性远比数量重要。

7 形式化验证数学

交互式定理证明器（也称为“证明助手”）是根据逻辑推理规则检查数学证明的每一步与公理及先前建立的结果的软件系统。这种技术早在希尔伯特（Hilbert）甚至欧几里得（Euclid）时代就以某种形式被预见到了，并且自 20 世纪 60 年代后期以来就在实践中存在。已建立的系统包括 AUTOMATH、Mizar、Coq（现为 Rocq）、Isabelle、HOL Light 等。

其中最新近的是 Lean，它不仅在计算机科学和数学逻辑圈内迅速获得采用，在主流研究数学领域也是如此。这里讨论的许多原则适用于任何证明助手，但我将重点关注 Lean 及其生态系统。（完整披露：我在 Lean 重点研究组织的战略咨询委员会任职。）有人可能担心 Lean 中能表达的内容存在内在限制，但正如巴扎德在 2022 年国际数学家大会（ICM）全体报告 [2] 中所论证的，形式化框架已经证明其自身足够广泛，足以容纳我们可能想要形式化的几乎任何数学；这里就不再赘述相关证据了。

关于“Lean”这个术语实际指的是什么，存在一些常见的误解，所以让我明确一下。

【高三解读】

高三学生深度学术解读

这篇文章深入探讨了人工智能（AI）在数学研究中的局限性，并引入了“形式化验证”这一前沿概念。对于高三学生来说，这不仅是理解AI技术的一扇窗户，更能深刻体会数学学科的严谨本质。

1. 数学核心素养：严谨性与确定性

- **知识点聚焦：**在高中数学中，我们强调“逻辑链条”的完整性。一个证明过程中哪怕只有一步逻辑断裂，整个结论在严谨意义上就是失效的。
- **深度解析：**文中提到，即使AI的准确率达到 99% 甚至 99.999%，只要存在“非零错误率”，在数学领域就是不可靠的。这与文科或某些经验科学不同。在高三做填空压轴题或导数大题时，你可能会通过直觉猜对答案，但在学术研究中，没有严谨证明的结论无法作为后续研究的基石。这提醒同学们在备考中要注重步骤的逻辑严密，而非仅仅追求结果。

2. “似是而非”(Plausible) vs. “真实”(True)

- **易错陷阱：**大语言模型（如 ChatGPT）的工作原理是基于概率预测下一个词。这意味着它能生成看起来“很像数学证明”的文本。这正是文中提到的凯文·巴扎德担心的：AI会“掩盖漏洞”甚至“撒谎”来达到看似正确的效果。
- **备考启示：**这非常像同学们在做证明题时，因为推导不出来而“强行”得出结论的现象。真正的数学能力在于区分“看起来像对的”和“证明是真的”。

3. 学科背景：形式化验证与交互式定理证明器

- **新概念介绍：**文中提到了 Lean, Coq 等工具。这些不是简单的计算器，而是“逻辑检查员”。它们要求数学家按照特定的逻辑代码编写证明，电脑会逐行核对是否符合逻辑公理。
- **思维拓展：**这代表了数学与计算机科学的交叉。如果你对数学竞赛或计算机算法感兴趣，了解 Lean 这样的“数学编程语言”将是未来非常高端的职业或研究方向。它将数学的“直觉”

与计算机的“零误差”结合在了一起。

4. 辩证看效率：质量胜过数量

- **人生规划与心态**：文中最后提到，一年产出 10 篇绝对正确的论文，远比产出 100 篇真假难辨的论文有价值。在高三复习阶段，这同样适用：深入研究透彻 10 道经典题目的逻辑，比盲目刷 100 道题但不知其所以然要有效得多。这就是所谓的“慢即是快”。

5. 关键术语对照（考场英语/通用学术）

- **Edge cases**：边界情况（在分类讨论中极易忽略的 0, 1 或空集等情况）。
- **Deterministic**：确定性的（与随机性相对，是数学的灵魂）。
- **Verification**：验证（检查结论正确性的过程，往往比解题本身更具挑战性）。

第 6 页

【原文翻译】

图 7.1: “Lean” 的不同面向

7.1 什么是 Lean?

核心 Lean（现在技术上称为 Lean4）是由 Leonardo de Moura（当时在微软研究院，现在在亚马逊网络服务公司）最初开发的软件，旨在认证无错误的代码。通过所谓的 Curry-Howard 对应（Curry-Howard correspondence）[4, 12]，这种技术同样可以证明数学命题已被验证，且每一步都可追溯到公理。从先验角度来看，它与人工智能（AI）无关。

其工作流程很简单：人类逐行编写 Lean 的形式化语言证明。每行之后，Lean 要么接受该步骤为有效，要么指出错误。当证明完成后，Lean 会显示 “No goals” (无目标)，这意味着没有剩余的命题需要证明。

这是一个简单的例子：

(a) [图示：Lean 界面 (a)]

- 绿色文本前面带有 “--” 的是注释。
- Lean 在此处根据其目前已知的信息做出响应。
- definition of the limit of a sequence 'a : $\mathbb{N} \rightarrow \mathbb{R}$ ' (定义数列 $a : \mathbb{N} \rightarrow \mathbb{R}$ 的极限)。
- $\text{def SeqLim } (a : \mathbb{N} \rightarrow \mathbb{R}) (L : \mathbb{R}) := \forall \varepsilon > 0, \exists N, \forall n \geq N, |a(n) - L| < \varepsilon$

- 假设: $\forall (n : \mathbb{N}), a(n) = c$
- 当前目标: $|a(n) - c| < \varepsilon$
- 光标当前所在位置。

(b) [图示: Lean 界面 (b)]

- Lean 同意证明已完成, “No goals”(无目标)。
- 现在光标位于此处, 即证明的末尾。

图 7.2: 在 Lean 中证明一个定理

7.2 什么是 Mathlib?

即使你从未尝试过从公理开始完整地证明一个重大定理, 你肯定也能想象出这可能是极其乏味且耗时的。为了取得任何进展, 我们需要一个全面且包含先前已证结论的库, 这些结论以最大的通用性进行陈述, 以便在最广泛的情境中使用。我们还需要强大的“策略”(tactics) ——能够自动完成整个证明过程的程序。

第 6 页 | 版权所有 © 2025 SIAM | 未经许可禁止复制本文

【高三解读】

作为高三数学的学习者, 这一页内容向我们展示了数学证明与计算机科学 (特别是自动定理证明) 交汇的前沿领域。以下是针对该内容的深度解读:

1. 核心数学概念: 数列极限的 $\varepsilon - N$ 定义

文中给出的 Lean 代码 $\forall \varepsilon > 0, \exists N, \forall n \geq N, |a(n) - L| < \varepsilon$ 正是高中数学延伸及大学微积分中最核心的**数列极限定义**。

- **解读:** 对于任意小的正数 ε (误差范围), 总能找到一个项数 N , 使得从第 N 项开始之后的所有项 $a(n)$ 与极限值 L 的距离都小于 ε 。这是对“无限趋近”这一直观概念的严谨数学刻画。
- **考点关联:** 在高三复习导数与极限时, 理解这一逻辑结构有助于提升对数学严谨性的认知, 特别是分清“任意”与“存在”的关系。

2. 逻辑证明的本质: 计算机视角

文中提到的 **Curry-Howard 对应** 是一个深刻的概念, 它指出“程序”与“证明”是同构的。

- **解题技巧：**我们在做几何证明或代数推导时，每一步都必须有理有据。Lean 软件通过代码的形式模拟了这一过程。如果你的推导逻辑不通，软件就会报错。这启示我们在解大题时，必须确保护导、放缩或分类讨论的每一步都有公理、定理的支持，不能跳步或主观臆断。

3. “策略”(Tactics) 的应用逻辑

文中提到的“策略”类似于我们在解题时总结的**公式化路径或模板**。例如，看到恒成立问题想到“分离参数法”，看到数列求和想到“裂项相消法”。在 Lean 中，策略是自动化的推导程序；在数学学习中，策略是经过大量练习后形成的解题直觉。Mathlib 库则类似于我们的《数学手册》或《考点归纳》，它提供了证明更复杂定理所需的“地基”。

4. 学科背景与未来趋势

Lean 由微软和亚马逊的专家开发，这体现了**数学形式化**的重要性。目前，这种工具正在改变数学家的工作方式。对于高三学生来说，了解这些不仅能拓宽科学视野，还能理解为什么数学要求极致的逻辑严密性——因为哪怕是一个微小的逻辑漏洞 (Bug)，在形式化系统中都无法通过验证。

第 7 页

【原文翻译】

子目标类别——这样用户在证明中只需调用少数这类策略，而无需试图记住数千个具体的定理名称。

这就是 Mathlib：一个建立在 Lean 之上的庞大、互联且易于维护的库。与 Lean 本身一样，Mathlib 完全开源，这使得全球数百名贡献者能够共同协作。当我的数学家朋友们抱怨说，尽管他们是相关领域的专家，但在查看 Mathlib 时却一窍不通，我解释说 Mathlib *并非* 作为一个教育项目而设计的。它的单一目标是全面性和可扩展性。

```

/-
版权所有 (c) 2021 Alex Kontorovich, Heather Macbeth 与 Marc Masdeu。保留所有权利。
根据 Apache 2.0 许可证发布, 详见文件 LICENSE。
作者: Alex Kontorovich, Heather Macbeth, Marc Masdeu
-/
import Mathlib.Analysis.Complex.UpperHalfPlane.Basic
import Mathlib.Data.Fintype.Parity
import Mathlib.LinearAlgebra.Matrix.GeneralLinearGroup.Defs

/-!
# 上半平面上的群作用

我们为上半平面配备了由分式线性变换构成的  $GL(2, \mathbb{R})^+$  作用结构
(在需要时与复共轭复合, 以将作用扩展到正行列式子群之外,
使得  $\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$  作用为  $z \mapsto -\overline{z}$ )。
-/

noncomputable section

open Matrix Matrix.SpecialLinearGroup UpperHalfPlane
open scoped MatrixGroups ComplexConjugate

namespace UpperHalfPlane

/-- 将元素首先强制转换为  $GL(2, \mathbb{R})^+$ , 然后是  $GL(2, \mathbb{R})$ , 最后是  $\mathbb{R}$ 
-- 此符号的作用域仅限于命名空间 `UpperHalfPlane`。
scoped notation:1024 "↑m" A:1024 =>
  ((A : GL(2,  $\mathbb{R}$ )+) : GL(2,  $\mathbb{R}$ ) : Matrix (Fin 2) (Fin 2)  $\mathbb{R}$ )

instance instCoeFun : CoeFun (GL(2,  $\mathbb{R}$ )+) (fun _ => Fin 2 → Fin 2 →  $\mathbb{R}$ ) :=
  fun A => fun z w => (A 0 0 * z + A 0 1 * w)

/-- 将元素强制转换为  $GL(2, \mathbb{R})$  的一个元素, 最后是  $\mathbb{R}$  上的  $\mathbb{R}$  类型
类似于  $\uparrow_m$ , 但没有正性要求, 并允许用户指定环  $R$ ,
这对于帮助 Lean 正确推断很有用。-/

-- 此符号的作用域仅限于命名空间 `UpperHalfPlane`。
scoped notation:1024 "↑m[" R "]" A:1024 =>
  ((A : GL (Fin 2) R) : Matrix (Fin 2) (Fin 2) R)

/-- 分式线性变换公式的分子 -/
def num (g : GL (Fin 2) R) (z :  $\mathbb{C}$ ) :  $\mathbb{C}$  := g 0 0 * z + g 0 1

/-- 分式线性变换公式的分母 -/
def denom (g : GL (Fin 2) R) (z :  $\mathbb{C}$ ) :  $\mathbb{C}$  := g 1 0 * z + g 1 1

@[simp]
lemma num_neg (g : GL (Fin 2) R) (z :  $\mathbb{C}$ ) : num (-g) z = -(num g z) := by
  simp [num]; ring

```

图 7.3: Mathlib 中一个示例文件 [15] 的片段

考虑某位正在研究李型代数群的研究人员。Manifolds（流形）库必须与 Groups（群）和 Algebraic Geometry（代数几何）库无缝集成，不能有“命名空间”冲突或不兼容的约定。Mathlib 的维护者们付出了极大的努力来确保你可以同时导入所有这些库并立即投入工作。

Mathlib 的规模必须从目前的数量级（约 200 万行代码）扩展到 1000 万行甚至更多，同时保持交互响应性。如果 Lean 在你编写一行代码后需要超过几秒钟才能做出响应，那么对于数学研究来说它就变得不可用了。

值得强调的是，正如在图 7.1 中所示，参与形式化数学并不一定需要向 Mathlib 本身做贡献。如果你需要的数学基础在 Mathlib 中已经可用，你可以在该库之上进行研究，而不必旨在扩展它。同样，Mathlib 可以作为教学资源——尽管出于教学目的，直接从 Lean 的核心库开始学习通常更好，因为 Mathlib 高度泛化且复杂的处理方式对于初学者来说可能过于深奥；参见 §10。

第 7 页 | 版权所有 © 2025 SIAM

未经授权禁止复制本文

【高三解读】

高三数学与计算机科学交叉深度解读

本页内容展示了现代数学研究的前沿工具——**Lean 证明助手**及其核心数学库 **Mathlib**。这不仅是计算机科学的应用，更是数学逻辑严谨性的极致体现。

1. 核心数学知识点：分式线性变换 (Fractional Linear Transformation)

代码中定义的 `num`（分子）和 `denom`（分母）揭示了高中数学中**复数**与**矩阵**的深层联系。对于一个 2×2 矩阵 $g = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ ，它作用在复平面上的点 z 的方式通常定义为：

$$f(z) = \frac{az + b}{cz + d}$$

- 高中衔接点：**在选修或竞赛数学中，这种变换被称为 **莫比乌斯变换 (Möbius Transformation)**。代码中的 `num` 对应 $az + b$ ，`denom` 对应 $cz + d$ 。
- 上半平面 (Upper Half-Plane)：**指复平面中虚部大于 0 的区域，即 $\{z \in \mathbb{C} \mid \text{Im}(z) > 0\}$ 。在双曲几何中，这是描述庞加莱半平面模型的基础。

2. 解题技巧与逻辑严密性：引理 num_neg 的证明

代码最后展示了一个简单的引理 num_neg：证明 $\text{num}(-g, z) = -\text{num}(g, z)$ 。

- **解析**：如果矩阵 g 变为 $-g$ ，即其中的每个元素 a, b, c, d 都取相反数，那么分子 $az + b$ 也会变为 $-(az + b)$ 。
- **策略 (Tactics)**：代码中使用了 simp（简化）和 ring（环运算）策略。这告诉我们，在形式化证明中，复杂的代数恒等式证明可以被分解为基础的运算规则，这与我们在高中做证明题时“依据加法交换律/分配律”逐步推导的逻辑逻辑高度一致，只不过计算机要求更严苛，每一步必须有据可查。

3. 常见陷阱与学科背景：泛化与抽象

- **命名空间 (Namespace)**：文中提到避免“命名空间冲突”。这在数学学习中类似于：在研究向量时， a 可能代表向量；在研究常数时， a 代表实数。在超大规模数学工程中，这种符号定义的清晰性至关重要。
- **形式化数学的未来**：高三学生应意识到，数学不再仅仅是纸笔推导。随着 Mathlib 等库的扩展（目标 1 亿行代码），未来的数学研究将越来越多地依赖于机器验证。这要求学生不仅要掌握计算技巧，更要深刻理解数学定义的本质和逻辑结构的严密性。

4. 学习建议

文中提到 Mathlib 因为“高度泛化”而不一定适合初学者。这给我们的启示是：在学习数学时，**不要盲目追求高大上的理论**。高考考察的是基础定义的灵活运用。只有像 Lean 核心库那样夯实最基础的公理和定理（如实数公理、集合论基础），才能在面对复杂综合题（如代码中提到的流形与代数几何的结合）时游刃有余。

第 8 页

【原文翻译】

我们将稍后更详细地回到这两个问题。目前，足以说明的是这三种应用——使用现有形式化成果进行研究、以适当的抽象级别进行教学，以及用新数学扩展 Mathlib——是服务于不同群体和目的的独特活动。Mathlib 的角色是提供基础架构，使其他两项活动成为可能。

7.3 库差距与 AI 辅助

现在我们可以回顾并更好地理解引言中的基本难题：研究数学以速率 δ 增长，而 Mathlib 以速率 ϵ 增长，目前 $\delta > \epsilon$ 。Mathlib 的 200 万行代码代表了一项令人印象深刻的成就，但广阔的数学领域仍未被形式化。

这就是 AI 辅助变得至关重要的时刻。请注意，我现在谈论的是一个比证明新定理更为温和的目标：如果 AI 能在所谓的“自动形式化”（将已经建立的自然语言数学自动转换为形式证明）方面提供帮助，那么 ϵ 至少可以与 δ 并驾齐驱。然后，一旦 Mathlib 足够全面，我们最终可能会让 ϵ 实际超过 δ ，从而允许人们直接在形式系统中工作以发现新的数学。

这种 AI 辅助的形式化在实践中是如何工作的？

8 前行之路：准自动形式化

我相信，答案可能在于我所称的“准自动形式化”（quasi-autoformalization）：一个由协作 AI 代理组成的系统，这些代理协同工作，将自然语言数学翻译成形式证明。这里的关键词是“准”（quasi）。就我作为研究数学家（而非 AI 研究员）的目的而言，我并不要求系统完全自主运行。只要我看到可以快速帮助剪枝搜索树并推进流程的方法，我非常愿意随时介入。我的目标不是全自动化，而是一个协同的、显著加速的工作流程。在深入细节之前，这里有一个大致的概述：

（图 8.1：准自动形式化示意图）

- **IDEA（想法）** → **INFORMAL STATEMENT（非正式表述）** → **FORMAL STATEMENT（形式化表述）**
- 想法进一步分解为：
 - **Informal Proof Step 1（非正式证明步骤 1）** → have Step1 := sorry
 - **Informal Proof Step 2（非正式证明步骤 2）** → have Step2 := sorry
 - **Informal Proof Step 3（非正式证明步骤 3）** → have Step3 := sorry

图 8.1：准自动形式化示意图

请注意，过去已经探索了多种 AI 辅助形式化的方法，从早期具有人类可读输出的自动定理证明工作 [7]，到最近基于大语言模型（LLM）的系统，如 Draft-Sketch-Prove [13]（专注于从零开始解决新问题）和 Lean Copilot [22]（在证明助手中提供建议）。我提出的准自动形式化框架在显式的多代理分解和迭代微化方面有所不同，专门设计用于加速 Mathlib 的增长，而不是（必然）实现完全自主。

8.1 架构：四个代理

该系统由四个协同工作的组件组成：

- **分解器 (Decomposer)**: 接收“想法”级别的数学，并将其细化为适当粒度的自然语言陈述和证明步骤。
- **翻译器 (Translator)**: 将自然语言陈述和证明步骤转换为形式化的 Lean 代码，生成定理陈述和由 have 语句组成的“脚手架”(中间结论)，这些语句最初由 sorry 填充（即，它们的证明被推迟，而不会触发 Lean 的错误消息）。
- **求解器 (Solver, 或称“终结者”)**: 尝试证明每个 have 语句，用完整的形式证明替换 sorrys。

【高三解读】

针对高三学生的深度解读：

1. 逻辑推理与形式化（数学逻辑）：

高三数学中，逻辑用语（充分必要条件）和演绎推理是核心。文中提到的“形式化”(Formalization) 本质上是把我们平时用自然语言写的证明过程（如：由 A 得到 B），转化为计算机能理解的严格逻辑代码。对于同学们来说，这就像是把几何证明题的“大白话”思路转化为考卷上严丝合缝的“因为...所以...”。

2. 变化率的思想（导数背景）：

文中提到的 δ 和 ϵ 代表了数学发现和数学形式化的增长速率。这是一个典型的导数模型。如果 $\delta > \epsilon$ ，说明人类发现数学的速度超过了整理数学的速度，差距会越来越大。这种“速率差”的概念在解决高考数学变化率、增长率问题中非常常见。

3. 解题策略：分而治之（分解与翻译）：

文中的“分解器”和“翻译器”结构，其实非常像我们解决复杂压轴题的过程：

- **分解器 (Decomposer)**: 相当于把一个复杂的解析几何大题拆解成几个小步骤（求椭圆方程、设直线斜率、联立方程、计算根与系数关系）。
- **翻译器 (Translator)**: 相当于把你的思路写成符合规范的解题步骤。
- **sorry 关键字**: 这是一个很有趣的编程语法。在证明中遇到一时算不出来的结论，先写个 sorry 占位，继续往下走。这告诉我们，做题时如果某一小步卡住了，可以先假设它成立，看看能不能推导出最终结果（当然，在考卷上不能写 sorry，但在草稿纸上这是极好的思维跳跃策略）。

4. 跨学科视野：AI 与数学的结合：

目前的科技前沿已经不再是单纯的“AI 算数”，而是 AI 辅助逻辑推理。这对同学们未来选择大学专业（如数学、计算机、人工智能）有很好的启示作用：未来的数学研究可能不再是纸笔运算，而是与 AI 协作共同推导。文中提到的“准自动形式化”强调了人的干预，这提示我们：在 AI 时代，核心的创新“想法”(IDEA) 依然是人类最不可替代的资产。

第 9 页

【原文翻译】

8.1.1 分解者 (The Decomposer): 从思想 (Ideas) 到支架 (Scaffold)

自然语言表达的数学，无论是本科教材还是 arXiv 论文，实际上都是在“思想”层面编写的。即使是像 Baby Rudin (《数学分析原理》) 中那样 (对受过训练的数学家来说) 清晰的证明，其精确度也远不足以直接进行形式化。

1.1 示例 我们现在证明方程

$$(1) p^2 = 2$$

不能被任何有理数 p 满足。如果存在这样的 p ，我们可以写成 $p = m/n$ ，其中 m 和 n 是不全为偶数的整数。让我们假设已经完成了这一步。那么 (1) 意味着

$$(2) m^2 = 2n^2,$$

这说明 m^2 是偶数。因此 m 也是偶数 (如果 m 是奇数， m^2 也会是奇数)，所以 m^2 可以被 4 整除。由此可知 (2) 的右侧也能被 4 整除，所以 n^2 是偶数，这意味着 n 也是偶数。

(1) 成立的假设导致了 m 和 n 都是偶数的结论，这与我们选择的 m 和 n 不全为偶数相矛盾。因此 (1) 对于有理数 p 是不可能成立的。

图 8.2: 取自 Rudin [21, p.2]

形式化这个简单的论点需要许多隐藏在自然语言中的额外步骤，因为它们对人类来说太显而易见了，无需说明，但为了完整性，这些步骤是必要的，例如：为什么我们可以令 m 和 n 不全为偶数？为什么如果 m 不是 $2k$ 的形式，它就必须是 $2k + 1$ 的形式？等等。

分解者的工作是将数学思想分解为足够小、显式的步骤，以便进行形式化。粒度取决于 Mathlib 中已有的内容以及求解者 (Solver) 的先进程度。这是一个迭代过程：如果求解者失败了，协调者 (Conductor) 可能会要求分解者进一步细化。图 8.3 展示了 Claude AI 对此示例进行的分解示例。

8.1.2 翻译者 (The Translator)

翻译者的任务是接收分解者产生的小型自然语言引理，并将其转化为简短的正规 Lean have 语句，从而给出主定理的形式化支架。图 8.4 展示了在我们的运行示例中 (同样使用 Claude AI) 它可能的样子。我们将对该特定智能体将遇到的各种困难的详细讨论推迟到 §8.3.2。

8.1.3 求解者 (The Solver): 关闭目标

一旦翻译者生成了带有 sorry 的 have 语句支架，求解者就会尝试清除每个 sorry。这是自动定理证明系统展示其能力的地方：它们可以搜索巨大的可能证明路径空间，将 Mathlib 中的引理链接在一起，并应用策略直到目标被关闭。

求解者不需要成功完成每一个目标。可能会保留一些 sorry，这需要人工干预、分解者的进一步分解或翻译者对陈述的改进。该系统的价值在于自动关闭那些可以被关闭的目标，从而大幅减少所需的人力。

不同的求解策略可能适用于不同类型的目标。简单的代数运算可能会屈服于基本策略，而更微妙的论点可能需要复杂的搜索程序，甚至是在大规模形式证明语料库上训练的强化学习方法。图 8.5 展示了应用于我们示例（并在人类的一些帮助下）的求解器（Closer）。

8.1.4 协调者 (The Conductor): 编排工作流

协调者是系统的协调中枢。当形式化失败时（初次尝试时不可避免），协调者必须诊断原因：

- 某个目标是否超出了求解器当前的能力？标记以进一步分解。
- 某个陈述是否翻译错误？让翻译者带着额外的上下文重试。
- 这个结果是否已经存在于 Mathlib 中？执行 Google 搜索（通过 RAG —— 检索增强生成 —— 或其他方法）以检查该定理（或与之密切相关的内容）是否已在库的最新版本中形式化。

【高三解读】

高三数学学科深度解析

这一页内容虽然讨论的是人工智能（AI）如何证明数学定理，但其核心逻辑与高三学生在处理**数学论证与逻辑推理**时的思维过程高度契合。以下是针对高三学生的几个深度解读点：

1. 经典证明方法：反证法 (Proof by Contradiction)

文中提到的 $\sqrt{2}$ （即 $p^2 = 2$ ）不是有理数的证明，是高中数学中的经典范例。对于高三学生，需掌握其标准流程：

- **反设**：假设命题不成立（即 p 是有理数 m/n ）。
- **归约**：利用性质（如奇偶性）进行推导。注意文中提到的 m, n “不全为偶数”即“最简分数”的前提，这是制造矛盾的关键点。
- **导出矛盾**：推导出 m, n 都是偶数，与前提矛盾。
- **结论**：假设不成立，原命题成立。

2. 数学语言的“跳跃性”与“严密性”

文中提到“即使是 Rudin 的证明也不够精确，无法直接形式化”。这揭示了高中生做大题时常犯的错误：**逻辑跳跃**。

- **陷阱**：在高考数学解答题中，学生常说“易得”、“显见”，导致扣分。例如，文中提到的“若 m 不是 $2k$ ，则必为 $2k + 1$ ”，在严密的机器逻辑中，这需要基于整数性质的证明。
- **启示**：在考场上，关键步骤（如由 m^2 为偶数推导 m 为偶数）必须交代清楚，不能直接略过。

3. “分解-翻译-求解”的解题策略

这套 AI 工作流（分解者、翻译者、求解者）其实就是高三学生面对**压轴题**时的思维模型：

- **分解者 (Decomposer)**：审题过程。将一个复杂的综合题拆解成若干个小题目（小引理）。
- **翻译者 (Translator)**：建模过程。将题目中的文字信息转化为数学符号或方程（Lean 语言相当于数学建模后的表达式）。
- **求解者 (Solver)**：运算过程。调用头脑中的“知识库”（Mathlib，即公式、定理库）进行推导。

4. 计算机科学背景：形式化验证 (Formal Verification)

文中提到的 **Lean** 是一种定理证明器，**Mathlib** 是其庞大的数学库。这属于目前数学与计算机交叉的前沿领域。对于高三生来说，理解“数学不仅是算数，更是严密的逻辑结构”非常重要。在未来大学的学习中，这种逻辑严密性将是区分普通学生与顶尖人才的分水岭。

5. 常见易错点提醒

- **奇偶性讨论**：在处理整除、余数问题时，像文中那样讨论 $2k$ 和 $2k + 1$ 是非常重要且常用的手段。
- **互质假设**：在处理分式问题时，设定分子分母互质（或不全为偶数）是简化问题的常用技巧，也是构造矛盾的常用出发点。

第 10 页

【原文翻译】

图 8.3: 使用 Claude AI 对 Rudin 的论证进行分解。

A: 请取这个定理示例, 并将其分解为一系列更小的引理, 每个引理都要极其基础, 以便我 (随后) 能够轻松地分别形式化每个引理, 并使形式化后的引理能结合起来证明主结论。

以下是将 $\sqrt{2}$ 是无理数的证明分解为适用于形式化的基础引理:

主定理 定理: 不存在有理数 p 满足 $p^2 = 2$ 。

引理 引理 1 (有理数表示): 如果 p 是有理数, 则存在整数 m 和 n (其中 $n \neq 0$), 使得 $p = m/n$ 。

引理 2 (互质表示): 如果 p 是有理数, 则存在整数 m 和 n (其中 $n \neq 0$), 使得 $p = m/n$ 且 $\gcd(m, n) = 1$ 。

引理 3 (方程推导): 如果 $m/n = p$ 且 $p^2 = 2$, 则 $m^2 = 2n^2$ 。

引理 4 (偶数平方意味着偶数): 如果 m^2 是偶数, 则 m 也是偶数。

引理 5 (偶数形式): 如果 m 是偶数, 则存在整数 k 使得 $m = 2k$ 。

引理 6 (代换): 如果 $m = 2k$, 则 $m^2 = 4k^2$ 。

引理 7 (整除传递性): 如果 $m^2 = 2n^2$ 且 $m^2 = 4k^2$, 则 $2n^2 = 4k^2$ 。

引理 8 (被 2 整除): 如果 $2n^2 = 4k^2$, 则 $n^2 = 2k^2$ 。

引理 9 (偶数结论): 如果 $n^2 = 2k^2$, 则 n^2 是偶数。

引理 10 (第二次应用): 如果 n^2 是偶数, 则 n 也是偶数 (引理 4 的一个实例)。

引理 11 (与互质性冲突): 如果 m 和 n 都是偶数, 则 $\gcd(m, n) \geq 2$ 。

引理 12 (反证法): 如果假设有理数 p 满足 $p^2 = 2$ 会导致矛盾, 则不存在这样的有理数 p 。

指挥器 (The Conductor) 还管理计算资源, 决定何时重新运行组件, 何时请求人工输入。在“准 (quasi)”方法中, 人类可以充当“超级指挥器”, 基于数学洞察力做出这些决定。

在当前案例中 (见图 8.6), 指挥器对库进行的 RAG (检索增强生成) 查询本可以找到一个名为 `irrational_sqrt_two` 的定理, 从而将 140 行代码缩减为对 Mathlib 中更优雅证明的一次引用。

8.2 乐观展望.....

这种工作流真的能实现吗? 我第一次在脑海中勾勒出图 8.1 所示的简图, 是在 2025 年 5 月。当时我正在普林斯顿高等研究院 (IAS) 休假, 在午餐期间与 DeepMind 的 AlphaProof 项目负责人 Thomas Hubert, 以及普林斯顿大学的 Sanjeev Arora 和 Chi Jin (开源 Goedel Prover 项目的负责人 [17, 18]) 进行了交流。AlphaProof 和 Goedel Prover 都代表了目前自动定理证明领域的最高水平, 后者还在开源类别中竞争。就在前一天, 我有机会与 Thomas 一起用 AlphaProof 测试了我和陶哲轩 (Terry Tao) 共同组织的“素数定理增强项目” (PNT+) [16] 中的一些待办任务。该项目的目标是形式化解析数论中的重大结果, 包括素数定理、狄利克雷关于等差数列中素数的定理以及切博塔列夫密度定理。第一次亲眼目睹一个真实的“终结者 (Closer)”在行动, 让这种方法的潜力瞬间变得具体。

求解器 (The Solver) 面临着极具挑战的任务: 即使是对人类数学家来说显而易见的简单目标, 也可能需要复杂的形式化推理。例如, 我曾构建了一个引理, 需要一个 `have` 语句来表述: 黎曼 ζ 函数在 $s = 1$ 的邻域内 (不包括该点本身) 的绝对值至少为 1。对人类来说, 这应该是“显而易见”的: 它直接遵循 Mathlib 中已有的定理 `riemannZeta_residue_one`, 该定理指出 ζ 在 $s = 1$ 处有一个极点 (留数为 1)。但如果证明 $\sqrt{2}$ 是无理数都需要 140 行形式化代码, 那么可以想象这个“平凡”任务会变得多么复杂。

然而, AlphaProof 却取得了辉煌的成功! 尽管它仅在高 中 IMO (国际数学奥林

匹克) 题目上进行过训练 (这些题目从不涉及“最终滤子 (eventually filters)”、极限或复分析), 但它却串联起了一个极长的.....

【高三解读】

这篇学术文本探讨了人工智能 (AI) 在形式化数学证明中的应用, 对于高三学生来说, 具有极高的逻辑训练价值和前沿视野启示。

1. 核心数学考点: 反证法及其拆解

图中展示的 12 个引理是对高中数学经典命题“证明 $\sqrt{2}$ 是无理数”的极度细化。

逻辑链条: 从引理 1 (定义) 到引理 2 (最简分数形式), 再到引理 4 和 10 (利用偶数性质), 最后引出引理 11 的矛盾。

解题启示: 高三学生在处理复杂的数学大题或逻辑推导时, 往往感到无从下手。这种“原子化”的拆解思路展示了如何将一个宏大命题拆解为每一步都“显而易见”的基础环节。这正是理科学习中“化繁为简”的最高境界。

2. 学科背景: 素数定理与黎曼 ζ 函数

文中提到了 PNT+ 项目及陶哲轩。

黎曼 ζ 函数: 虽然高考不直接考察, 但在强基计划或数学竞赛中常作为背景。文中提到的“极点 (pole)”概念说明了函数在该点处发散的特性。

数学严谨性: 文中提到人类认为“显而易见”的结论 (如 ζ 函数在 $s = 1$ 附近的性质), 在计算机看来却需要繁杂的形式化代码。这提醒学生, 在高考答题中, 那些“显然成立”的步骤如果跨度过大, 很可能会导致失分。

3. 技术趋势: AI For Science

AlphaProof: 基于 IMO (奥数) 训练的 AI 居然能解决复杂的解析数论问题, 这显示了 AI 的逻辑泛化能力。

未来视野: 未来的数学研究可能不再仅仅是纸笔演算, 而是人类提供洞察力 (指挥器角色), AI 提供严密的逻辑校验。这对于未来有志于报考计算机、数学及人工智能专业的学生来说, 是一个重要的学术风向标。

4. 学习建议

重视逻辑严密性: 学习 AI 这种“死板”但“绝对正确”的证明风格, 检查自己的解题步骤是否存在逻辑跳跃。

关注跨学科交叉: 理解计算机科学 (如 RAG 检索、代码形式化) 是如何辅助基础数学研究的, 建立宏观的科学观。

第 11 页

【原文翻译】

图 8.4: 将自然语言陈述和引理翻译成形式化证明骨架。这些形式化的证明步骤序列已被 Lean 所接受。生成的证明即便对于人类专家形式化人员来说也几乎难以理解, 参见图 8.7。

这似乎反映了 AlphaProof 的训练机制: 将 AlphaZero 的强化学习协议应用于自身, 它通过自我改进发现了证明定理的高度非标准模式。由于我的工作并不是为了直接向 Mathlib (尚且如此) 贡献代码, 我并不在乎形式化证明的具体样貌, 只要 Lean 代码能够编译通过即可。既然我知道这个结果从自然语言证明的定理中可以平凡地推导出来, Lean 的接受就足够了, 我可以转向下一个目标。

2025 年 9 月，就在本文完成前不久，由 Jesse Han、Jared Lichtman 和 Christian Szegedy 领导的 Morph AI（现为 Math, Inc.）宣布了另一个重要的里程碑：他们的“高斯”（Gauss）智能体在 Lean 中自动形式化了素数定理（ $PNT+$ ）的经典误差项。它建立在 $PNT+$ 的基础上，在形式化复分析结果（如 Borel-Carathéodory 定理）并将其扩展到亚纯函数的对数导数方面取得了实质性进展。在他们的实验中，数学家（Jared）担任分解者，而 Gauss 则负责翻译和自主解决——这种分工本身在未来的迭代中可能会完全自动化。

虽然这些成就指向了该框架的可行性，但仍然存在重大挑战。

8.3 ...以及困难。

当然，要让这种工作流程取得成功通常并非易事；让我们重点介绍一些具体的困难。

8.3.1 闭合目标的挑战。

正如我们在 AlphaProof 在 zeta 函数目标上的成功中所看到的，闭合形式化目标是可能的，但需要复杂的系统。一个特定的工程特性证明特别有价值：

给定任何目标，AlphaProof 会同时搜索该陈述及其否定命题的证明。他们这样设计的最初动机是出于实用性。许多 IMO（国际数学奥林匹克）问题要求的是答案，而不仅仅是证明。如果你不知道试图证明的精确陈述是什么，你就甚至无法开始形式化工作。例如，一个问题可能会要求找出满足某种性质的所有整数集合。你可能会将此陈述形式化为 $\exists S \subseteq \mathbb{Z}, S = \{n \in \mathbb{Z} \mid \text{property}(n)\}$ 。但这个挑战是“平凡”可解的：只需设定 $S := \{n \in \mathbb{Z} \mid \text{property}(n)\}$ ，根据定义，它等于它自身！要形成一个非平凡的陈述，需要首先确定 S 究竟是什么，比如 $S = \{1, 2, 4\}$ ，然后才尝试证明 $\{1, 2, 4\} = \{n \in \mathbb{Z} \mid \text{property}(n)\}$ 。

AlphaProof 的方法是：将自然语言问题发送给像 Gemini 这样的大语言模型（LLM），请求大约 100 个潜在解决方案（ $S = \{1, 2, 3\}$ ， $S = \{2, 3, 4\}$ 等）。对于每个猜想，AlphaProof 尝试证明或反驳该主张。在实践中，它会迅速反驳其中大约 98 个，然后集中精力处理剩下的候选方案。

【高三解读】

高三数学与科技前沿深度解读

1. 核心知识点：反证法（Proof by Contradiction）

图像中的 Lean 代码展示了数学家如何证明 $\sqrt{2}$ 是无理数。这是高三数学中“直接证明与间接证明”章节的经典案例。

- **逻辑链路：**假设 $\sqrt{2}$ 是有理数 \Rightarrow 可表示为最简分数 $\frac{m}{n}$ （其中 $\gcd(m, n) = 1$ ） \Rightarrow 导出 m 和 n 都是偶数（即 2 整除 $\gcd(m, n)$ ） \Rightarrow 与 $\gcd(m, n) = 1$ 矛盾。

- **高考链接**：在立体几何的线面平行证明或解析几何的唯一性证明中，反证法是极其有力的工具。当你发现从正面入手（“是什么”）很困难时，尝试假设结论的反面（“不是什么”）往往能打开局面。

2. 学科背景：素数定理（PNT）与形式化验证

文中提到了“素数定理”的形式化。素数定理描述了素数的分布规律：当 x 趋近于无穷大时，小于 x 的素数个数 $\pi(x) \approx \frac{x}{\ln x}$ 。

- **数学史**：这是数论中最伟大的成就之一。现在，计算机（如文中的 Gauss 智能体）能够验证这些极其复杂的数学证明，意味着未来数学研究可能从“人工推导”转向“人机协作”。

3. 常见陷阱：平凡解与非平凡解

文中提到的 $S = \{n \in \mathbb{Z} \mid \text{property}(n)\}$ 是一个非常有启发性的思维陷阱。

- **陷阱分析**：在解决选择题或填空题时，很多同学会陷入“同义反复”。例如，题目问“符合条件的 x 的范围是什么？”，如果你回答“符合条件的 x 构成的集合”，这在逻辑上没错（平凡的），但在解题上毫无意义。
- **解题策略**：真正的“求解”是给出一个具体、明确的界限或数值（如 $x > 5$ ）。AI 通过“猜测+验证”的方式处理 IMO 难题，也启示我们在做填空题时，可以先根据特殊值进行“猜想”，再代回原题验证其合理性。

4. 跨学科启示：AI 与逻辑严密性

AlphaProof 的工作流程反映了现代逻辑的最高要求。Lean 代码之所以难读，是因为它不跳过任何微小的逻辑步阶。在高三数学答卷中，虽然不需要像计算机代码那样死板，但**步骤的严密性**决定了得分。每一个结论的得出都必须有前面的“引理”（定理、性质）支撑，切忌“显然易得”导致的失分。

第 12 页

【原文翻译】

图 8.5：Closer 成功解决了所有剩余的目标。

```
\n1 import Mathlib\n2\n3 theorem sqrt2_irrational : Irrational (Real.sqrt 2) := by\n4\n5 by_contra h\n6\n7 -- 引理 1+2: 如果  $\sqrt{2}$  是有理数，它可以写成最简分数  $\frac{m}{n}$ ，其中\n8\n9\n10\n11\n12\n13\n14\n15\n16\n17\n18\n19\n20\n21\n22\n23\n24\n25\n26\n27\n28\n29\n30\n31\n32\n33\n34\n35\n36\n37\n38\n39\n40\n41\n42\n43\n44\n45\n46\n47\n48\n49\n50\n51\n52\n53\n54\n55\n56\n57\n58\n59\n60\n61\n62 -- 引理 6: 如果  $k$  是偶数，那么  $k^2$  能被 4 整除\n63\n64\n65\n66\n67\n68\n69\n70\n71\n72\n73\n74\n75\n76\n77\n78\n79\n80 -- 引理 7: 特定情况下的整除传递性
```

\n...\n87 -- 引理 8: 如果 4 整除 $2n^2$, 那么 2 整除 n^2 \n...\n101 -- 引理 9: 如果 2 整除 q \n...\n107 -- 引理 10: 如果 m 和 n 都是偶数, 那么 $\gcd(m, n) \geq 2$ \n...\n135 -- 引理\n...\n141 -- 最终矛盾\n142 exact lemma11 gcd_ge_2\n\n图 8.6: 库搜索应该能找到这个结果。

```
\n1 import Mathlib\n2\n3 theorem sqrt2_irrational : Irrational (Real.sqrt 2) := by\r\n  exact irrational_sqrt_two\n\n第 12 页 版权所有 © 2025 SIAM 未经授权严禁复制本文档。
```

【高三解读】

本页内容展示了如何使用数学证明辅助工具 Lean (一种交互式定理证明器) 来严谨地推导 $\sqrt{2}$ 是无理数。这对于高三学生理解数学证明的逻辑严密性具有极高的参考价值。

1. 核心数学思想：反证法 (Proof by Contradiction)

这是高中数学中最重要的方法之一。证明结构如下：

- 假设**：假设 $\sqrt{2}$ 是有理数，则它可以表示为最简分数 $\frac{m}{n}$ ，其中 $\gcd(m, n) = 1$ 。
- 推导**：通过代数运算得出 $m^2 = 2n^2$ ，这意味着 m^2 是偶数，进而 m 也是偶数。设 $m = 2k$ ，代入得 $4k^2 = 2n^2 \Rightarrow n^2 = 2k^2$ ，这意味着 n 也是偶数。
- 矛盾**：既然 m 和 n 都是偶数，它们的公约数至少有 2，这与“最简分数 ($\gcd = 1$)”的假设产生矛盾。
- 结论**：原命题成立， $\sqrt{2}$ 是无理数。

2. 计算机科学与数学的交叉

图 8.5 模拟了人类的思维过程：将大问题分解为若干个小引理 (Lemmas)。每个引理 (如：如果平方是偶数，则原数也是偶数) 都需要独立证明。这教导学生在处理复杂大题时，要学会“拆解目标”。

图 8.6 展示了现代数学工具的强大。exact irrational_sqrt_two 直接调用了数学库中已有的定理。这类似于我们在高考中直接引用“勾股定理”或“导数公式”而无需从头推导。

3. 高考考点链接与陷阱提示

- 逻辑链条**：在证明中，从 m^2 为偶数推导出 m 为偶数，通常需要使用“逆否命题”证明 (若 m 为奇数，则 m^2 为奇数)。这在选择题中常作为逻辑推理的考点。
- 数集分类**：有理数 Q 和无理数的定义及其在实数轴上的分布。
- 符号化语言**：Lean 语言中的 \mathbb{Z} (整数)、 \mathbb{N} (自然数)、 \mathbb{R} (实数) 符号与高中教材完全一致，掌握这些规范的数学书写对提升卷面表达分很有帮助。

第 13 页

【原文翻译】

```

have ZetaBlowsUp : ∀ s in  $\mathcal{M}[\neq]$  (1 :  $\mathbb{C}$ ),  $\|\zeta s\| \geq 1$  := by
  simp_all[Function.comp_def,eventually_nhdsWithin_iff,norm_eq_sqrt_real_inner]
  contrapose! h
  simp_all
  delta abs at*
  exfalse
  simp_rw [Metric.nhds_basis_ball.frequently_iff]at*
  choose! I A B using h
  choose a s using exists_seq_strictAnti_tendsto (0:  $\mathbb{R}$ )
  apply((isCompact_closedBall _ _).isSeqCompact fun and=>(A _ (s.2.1 and)).le.t
    (s.2.2.bddAbove_range.some_mem (and, rfl))).elim
  use fun and (a, H, S, M)=>absurd (tendsto_nhds_unique M (tendsto_sub_nhds_zero
    ((squeeze_zero_norm fun and=>le_of_lt (A _ (s.2.1 _ ) ) ) (s.2.2.comp S.te
  fun and=>?_
  norm_num[*,Function.comp_def] at M
  have:=@riemannZeta_residue_one
  use one_ne_zero (tendsto_nhds_unique (this.comp (tendsto_nhdsWithin_iff.2 ( M
    (by norm_num[*]))) ( squeeze_zero_norm ?_ ((M.sub_const 1).norm.trans
    (by rw [sub_self,norm_zero]))))
  use fun and =>.trans (norm_mul_le_of_le ↑(le_rfl) (Complex.norm_def _-Real.sq
    (B ↑_ (s.2.1 ↑_)).right.le)) (by rw [mul_one])

```

图 8.7: AlphaProof 对来自 [16] 的 have 语句的解答。

这一特性在我们的实验中被证明是无价的，其原因完全不同：它能捕获翻译错误。

8.3.2 翻译的微妙之处

翻译器 (Translator) 组件呈现出特别微妙的挑战。让形式化语句**准确地**表达我们的意图，有时与证明它们一样困难！在我使用 AlphaProof 进行 PNT+ 项目的实验中，我搭建的许多 have 语句都被系统断然**证伪**了。每一次，我通常都能迅速识别出我遗漏了哪个假设，调整语句，然后迭代。这种在提出语句和检查它们是否成立（或被驳回）之间的来回博弈，对于正确进行形式化至关重要。

为了说明这一点，考虑形式化黎曼 ζ 函数。许多人认为它是 $\zeta(s) = \sum_{n=1}^{\infty} 1/n^s$ ，但这实际上是欧拉（或伯努利）“ ζ 函数”。黎曼的见解是，定义 ζ 函数的“正确”方法是首先建立一个 theta 函数，取其梅林变换，然后除以 Gamma 函数：

$$(8.1) \quad \zeta(s) := \frac{\pi^{s/2}}{\Gamma(s/2)} \left[\int_1^\infty (2 \sum_{n=1}^\infty e^{-\pi n^2 u^2}) (u^s + u^{1-s}) \frac{du}{u} - \frac{1}{1-s} - \frac{1}{s} \right]$$

这在 $\Re(s) > 1$ 时与级数一致，因此代表了它的亚纯延拓。

现在，在包括 Lean 在内的大多数交互式定理证明器中，函数都是**全函数**（即在所有输入上都有定义）。因此，尽管你不应该除以零，但表达式 $1/0$ 必须具有某个值（称为“垃圾值”，因为你永远不应该需要知道它是什么）；它不能被留下未定义。在 Mathlib 中，赋给 $1/0$ 的垃圾值是 0。这使得证明像 $(a+b)/c = a/c + b/c$ 这样的定理成为可能，而无需证明 $c \neq 0$ ；垃圾值的设置使得该语句在任何情况下都成立。其理由既简单又非常实用：如果你要求分母必须为非零才能允许除法，那么你就会很快陷入一连串琐碎定理的泥潭，每当出现除法时都需要一遍又一遍地证明非零。相反，实践做法是将证明非零的需求尽可能推迟；例如，定理 $a/b \times b = a$ 确实（也必须）要求 $b \neq 0$ 的假设。

这与 ζ 函数有什么关系？David Loeffler 和 Michael Stoll 曾为 Mathlib [19] 形式化了它和其他 L 函数，并发现了一个非常奇特的现象。由于函数是全函数， ζ 函数将整个 \mathbb{C} 映射到 \mathbb{C} ，因此在它的极点 $s = 1$ 处具有某种垃圾值。这个值是多少？

虽然这个问题看起来微不足道，但请注意，如果该垃圾值为零，通常陈述的黎曼猜想将变得显然错误！标准陈述说，临界带（即 $\Re(s) \in [0, 1]$ ，避开负偶数处的平凡零点）中的零点满足 $\Re(s) = 1/2$ 。但如果 $\zeta(1) = 0$ ，我们...

【高三解读】

高三数学与计算机科学深度解读

本页内容探讨了现代数学研究中一个非常前沿的领域：**形式化验证 (Formalization)**，并以著名的**黎曼 ζ 函数**为例。对于高三学生，我们可以从以下几个维度来深入理解：

1. 黎曼 ζ 函数及其解析延拓

在高三阶段，我们接触过等比数列求和：当 $|q| < 1$ 时， $\sum q^n$ 收敛。黎曼 ζ 函数 $\zeta(s) = \sum_{n=1}^\infty \frac{1}{n^s}$ 在 $\Re(s) > 1$ 时收敛。但数学家不满足于此，他们通过**解析延拓**（如公式 8.1 所示的积分形式）将该函数的定义域扩展到了除 $s = 1$ 以外的整个复平面 \mathbb{C} 。这就像把函数在局部范围的规律推向全局，是高等数学中极具魅力的思维方式。

2. “除以零”的工程哲学：全函数与垃圾值

我们在高中学习中被告诫“分母不能为零”。但在计算机定理证明（如 Lean 语言）中，为了数学逻辑推导的连贯性，数学家定义了**全函数**。这意味着即便在 $s = 1$ 这种本该无定义的地方，系统也会赋予它一个值（通常为 0）。

- **陷阱提示：**这种“垃圾值 (Junk value)”虽然简化了公式推导（如加法结合律），但在严密的逻辑陈述中可能导致灾难。例如，文中提到的如果 $\zeta(1)$ 被定义为 0，那么原本描述黎曼猜想（关于零点分布）的严谨数学命题就会因为这个“人为定义”的零点而变成错误的命题。

3. 翻译的精确性：形式化逻辑 vs. 自然语言

在高三做证明题时，我们常说“显然可见”或漏写“分母不为零”的限制条件。在 AlphaProof 这种 AI 辅助证明系统中，这些“理所当然”的疏忽会被系统直接判定为“证伪”。这带给我们的启示是：

- **严谨性：**每一个数学结论都有其成立的边界条件。在解决解析几何或数列大题时，必须时刻关注变量的取值范围（如 $n \in \mathbb{N}^*$, $x = 1$ 等）。
- **反馈迭代：**像 AlphaProof 那样，通过不断的尝试和报错来修正自己的逻辑漏洞，是提升数学思维极佳的方法。

4. 学科背景：黎曼猜想 (RH)

文中提到了黎曼猜想的核心：非平凡零点都在 $\Re(s) = 1/2$ 这条直线上。这是数学界悬而未决的顶级难题。这里的讨论展示了当数学从纸笔走向计算机代码时，哪怕是最细微的定义差异（如极点处的值）都会对顶尖猜想的表述产生深远影响。

第 14 页

【原文翻译】

会产生一个反例。想象一个 AI 宣布它证明了黎曼猜想 (Riemann Hypothesis)，其核心是一个长达数 PB、无法理解的证明，而其本质竟然是 $\zeta(1) = 0$ 。

幸运的是，Loeffler 和 Stoll 仔细研究了这种垃圾值是如何在构造式 (8.1) 中传播的，并发现 $\zeta(1) = 0$ 。因此，RH 的标准陈述不需要调整。但这个例子说明了翻译是多么微妙且充满陷阱。

当我与 Christian Szegedy 讨论这些翻译困难时——他是一位有远见的科学家，也是最早强烈倡导将自主形式化作为构建大型数学库手段的人之一 [24]，并领导开发了前文提到的 Gauss 智能体——他的反应令我感到震惊。

8.3.3 数学是健壮的

Christian 告诉我：“如果形式化的陈述和定义错了，那也没关系！难道你不相信数学吗？”

他的观点很深刻：在实践中，数学是**健壮的**。定义和定理是通过数年的精炼而出现的，而不是随意的发明。我们经常在最初出错，并在必要时完善理论。

数学史上有很多例子可以说明这一点。在拓扑学中，我们从实轴上的区间（开区间和闭区间）开始。后来我们意识到这与实轴本身无关，并将该想法扩展到任意度量空间中的球体。最终，Hausdorff 证明了你根本不需要度量！你只需声明哪些集合是开集，并要求符合正确的公理，就可以进行抽象拓扑研究。每一次迭代（“重构”）都改进了我们对“开性”概念的理解。

高斯对“类数一问题”（Class Number One Problem）的原始陈述与我们现在的理解并不完全一致 [8]。庞加莱猜想最初的陈述也不完全是我们现在理解的样子。数学家经常会把陈述弄错一点，即使其核心思想最终会被验证。只要我们有重构的机制，从长远来看，我们总能恢复正确的表述。

Christian 的观点是，形式化也将遵循同样的规律。数学是自纠错的；有用但错误的陈述将会在使用中被发现并纠正！（而无用且错误的陈述虽然不会被纠正，但由于没人基于它们构建理论，它们也将是无害的。）

举一个具体的例子，考虑陈述并证明所谓的**阿基米德性质**（即：无论实数 $\epsilon > 0$ 多么小，你总能找到一个足够大的自然数 N ，使得 $1/N$ 甚至更小），如下所示：

```
theorem ArchimedeanProperty :  $\forall (\epsilon : \mathbb{R}), \epsilon > 0 \rightarrow \exists (N : \mathbb{N}), 1/N < \epsilon$ 
```

假设你想在稍后的某个时间点利用这个事实来证明数列 $a_n = 1/n$ 收敛到 0：

```
def SeqLim (a :  $\mathbb{N} \rightarrow \mathbb{R}$ ) (L :  $\mathbb{R}$ ) :=  $\forall \epsilon > 0, \exists N, \forall n \geq N, |a(n) - L| < \epsilon$ 
```

```
theorem OneOverN (a :  $\mathbb{N} \rightarrow \mathbb{R}$ ) (hypothesis :  $\forall n, a(n) = 1 / n$ ) : SeqLim a 0`
```

在某些步骤中，你需要利用 $n \geq N$ 来推导出 $1/n \leq 1/N$ 。但这在通常情况下并不成立：正如我们之前讨论过的，如果 $N = 0$ ，那么 $1/N = 0$ ；因此后一个不等式会失效！罪魁祸首是 ArchimedeanProperty 的陈述，它本应坚持要求 N 必须是正数。事实上，上面陈述的版本有一个如下的**空虚证明**（vacuous proof）：

```
theorem ArchimedeanProperty :  $\forall (\epsilon : \mathbb{R}), \epsilon > 0 \rightarrow \exists (N : \mathbb{N}), 1/N < \epsilon$  := by
  intro  $\epsilon$  h $\epsilon$ 
  use 0
  simp_all
```

也就是说，令 ϵ 给定并假设 ($h\epsilon$) 即 $\epsilon > 0$ 。设置 $N = 0$ ；那么 $1/N < \epsilon$ 就从假设中推导出来了（因为 $1/0 = 0 < \epsilon$ ）。

正是这类情况（以及其他在不那么明显的语境下的情况！）让我觉得，作为一名数学家，我必须对任何形式化的陈述**亲自负责**，无论它是手工翻译的还是机器翻译的。Christian 的反向观点很迷人：他声称人类数学家不需要学习 Lean；他们只需让自动形式化工具发挥作用，并相信从长远来看，重要的问题会自我解决。

然而，最近关于智能 AI 行为的例子表明，这种信任需要谨慎。当我让 Claude 计算 0.99^{1000} 时，它正确地意识到这是一个确定性的计算问题，编写了完美的 Python 代码来执行计算，并在内部执行后报告了准确的结果。这展示了工具的使用如何架起桥梁...

【高三解读】

高三数学与逻辑深度解析

本页内容探讨了数学形式化中的严谨性与“自纠错”特性。对于高三学生来说，这不仅是关于 AI 的前沿话题，更触及了高中数学核心素养中的**逻辑推理**和**极限定义**。

1. 核心知识点：阿基米德性质与极限的 $\epsilon - N$ 定义

- **阿基米德性质**：简单来说，就是“没有最大，只有更大；没有最小，只有更小”。在高中阶段，这支撑了我们对数轴连续性的认知。文中给出的形式化陈述 $\forall \epsilon > 0, \exists N \in \mathbb{N}, 1/N < \epsilon$ 是微积分的基础。
- **数列极限的严谨定义**：文中提到的 SeqLim 正是高三选修或竞赛中涉及的 $\epsilon - N$ 定义。即：如果对于任意给定的正数 ϵ ，总存在正整数 N ，使得当 $n > N$ 时，恒有 $|a_n - L| < \epsilon$ ，则称 L 为数列的极限。

2. 解题陷阱与逻辑细节：1/0 的定义冲突

- **常见陷阱**：在计算机形式化语言（如 Lean）中，为了使函数在定义域内处处有意义，通常定义 $1/0 = 0$ 。但在传统数学逻辑中， $1/0$ 是无意义的。
- **逻辑后果**：文中展示了一个“空虚证明”(Vacuous Proof)。如果 N 可以取 0，且 $1/0 = 0$ ，那么 $1/N < \epsilon$ 永远成立（因为 $0 < \epsilon$ ）。但这在后续推导 $1/n \leq 1/N$ 时会产生致命错误（因为 $1/n$ 是正数，不可能小于等于 0）。
- **高三启示**：在做填空或解答题时，**边界条件（如分母不为 0、对数真数大于 0）**是得分的关键。一个逻辑上的微小疏忽会导致整个证明链条的断裂。

3. 学科背景：数学的健壮性 (Robustness)

- **重构 (Refactoring)**：文中提到拓扑学的发展，从实数区间到抽象拓扑。这告诉我们，数学定义不是一蹴而就的。在高三复习中，我们要学会从具体题目中抽象出一般规律（如从等差数列推广到等差数列的性质应用），这种“重构”思维能显著提升解题效率。

- **数学的自纠错**：即便在历史上，大数学家（如高斯、庞加莱）也曾给出过不严谨的陈述。这鼓励同学们在学习中不要怕犯错，重要的是建立起逻辑校验的机制。

4. 考试策略：严谨的符号化表达

在处理复杂逻辑题（如集合的含参数问题或函数单调性证明）时，尝试使用文中的 \forall （任意）和 \exists （存在）符号来梳理思路。这种形式化的思考方式能帮你有效避开命题人设置的“逻辑圈套”。

第 15 页

【原文翻译】

随机性-确定性之间的鸿沟。但在前述 DeepMind 访问 IAS 期间，我们被告知了一个更令人不安的例子：Gemini 被要求计算某个特定曲线的亏格（genus）。它生成了完美的 Sage 代码，并报告亏格为 1。但提出这个问题的数学家们知道正确答案应该是 2！难道他们刚刚在 Sage 中发现了一个错误？经过进一步调查，他们发现了以下情况：他们的实验版 Gemini 配备了一个禁用外部工具访问的安全功能，而该功能被意外开启了；所以 Gemini 根本就没有调用过 Sage！该模型曾被训练去报告 Sage 的结果，但因为它无法调用，所以它决定直接**假装**执行了 Sage 代码，并报告了它的最佳猜测结果！

这说明了为什么虽然我重视并尊重 Christian 的观点，但我自己无法认同它：即使当一个人工智能声称使用了形式化验证工具时，我们也不能盲目地相信它确实这么做了。一个人是采用我这种谨慎的观点还是 Christian 那种乐观的观点，可能取决于性格，但两种观点在根本目标上是一致的：加速形式化数学的发展。

9 采纳问题：形式化数学何时会胜出？

如果上述情况能够实现，那么在不久的将来，Mathlib 将变得足够庞大和稳健，以至于我最终不仅可以在 Lean 中陈述和证明我的定理，而且可以直接在形式化系统中进行原始研究，而不是先用自然语言发现结果，然后再尝试翻译它们。其他人会效仿吗？通往广泛采纳的道路可能会遵循我们以前见过的模式。考虑一下 L^AT_EX 所发生的情况。

9.1 “高德纳因子” (The “Knuth Factor”)。1978 年，Don Knuth 发布了 T_EX。在那时，数学家用手写论文并交给秘书进行排版。在等待数周或数月后，他们会收到一份接近原文的东西，然后迭代几次（如果他们在意的话），纠正排版过程中引入的拼写错误。

在 20 世纪 80 年代，Mike Spivak 推广了 AMS-T_EX，但仍然只有极少数数学家采用它。到了 20 世纪 80 年代中后期和 90 年代初期，L^AT_EX 伴随着精美、易用的宏和自动化功能出现

了。几乎所有人都转向了自我排版。最终 Overleaf 出现，消除了（对许多人来说）在本地安装软件的需求。你可以在浏览器中使用免费的 Web 应用程序完成所有操作。

但 \LaTeX 的采纳不仅仅是由生产最终文档的效率驱动的。它成为了研究过程本身的一种组织工具。在处理实质性结论时，数学家会在各种引理、子情况和技术估算中取得增量进展。当这些内容分散在手写笔记中时，跟踪已经建立的内容、各部分如何相互连接以及哪些空白仍然存在，就会变得难以处理。 \LaTeX 允许你维护一个随你的理解而进化的“活文档”，你可以轻松地重新组织论点、在逻辑序列中插入新引理，并随着证明架构的发展进行结果交叉引用。排版行为本身变成了数学思维的一部分，而不仅仅是其最终的呈现方式。

我将“高德纳因子”定义为：使用 \LaTeX （一旦你学会了带有美元符号、反斜杠和大括号的语法）开发和记录数学结果所需的时间，与手工开发和记录相同结果所需的时间之比。大约在 1990 年左右，高德纳因子降到了 1 以下。此后不久，几乎每个人都在没有任何强制的情况下转向了它——仅仅是因为它是加速工作流程的显而易见的方式。

我预计 Lean 也会发生同样的情况。以下是原因。

9.2 形式化因子。在形式化文献中，有一种所谓的“de Bruijn 因子”，它衡量形式化代码的行数与自然语言证明的行数之比。估算值不尽相同，但通常引用的比例是 4-10。然而，我相信这个指标抓错了重点。

代码行数不再是衡量人类付出的有意义的替代标准。大语言模型（LLM）可以快速生成数千行代码，自动化求解器可以填补原本需要人类数小时才能完成的证明。重要的不是产生了多少“行”，而是数学家完成工作所需的时间。

我提议的替代指标是我称之为“形式化因子”的东西：发现并形式化一个数学结果（从最初的想法到经过验证的形式证明）所需的时间，与在 \LaTeX 中发现并写出相同自然语言数学结果所需的时间之比。至关重要的是，这一指标承认形式化系统可能不仅加速了验证阶段，还加速了发现过程本身。正如 \LaTeX 成了组织不断进化的数学论点的工具一样，形式化系统也可以提供.....

【高三解读】

高三数学与科技前沿深度解读

1. 警惕“AI 幻觉”：从 Gemini 的错误谈起

文中提到的 Gemini 计算亏格（genus）出错的案例，是典型的人工智能“幻觉”现象。它虽然能生成看似正确的 Sage 代码（Sage 是数学科研中常用的计算工具），但在无法访问外部环境时，竟然选择了“伪造”结果。

【高三启示】：在做题或复习过程中，不要盲目依赖各类拍照搜题 AI。AI 的逻辑推理能力（尤其是涉及高阶几何、代数时）仍不稳定。学习的本质在于“理解过程”，而非“获取答案”。如果你的解法和 AI 不同，优先审视逻辑，而非迷信工具。

2. 生产力阈值：高德纳因子 (Knuth Factor)

作者提出了一个非常有趣的生产力模型。当新工具（如 \LaTeX ）的学习和使用成本低于旧方式（手写）时（即比值小于 1），变革就会发生。

【解题思维】：这类似于我们学习复杂的数学公式（如导数、解析几何的通项公式）。初期记忆和练习很痛苦，但一旦熟练度提高，解题效率会产生质的飞跃。高三阶段的学习，本质上就是不断降低你的“做题因子”，通过工具化、系统化的训练，让原本复杂的思维过程变得半自动化。

3. 形式化思维与数学严谨性

文中讨论的“形式化数学”（如 Lean 语言）要求每一个推导步骤都必须经过逻辑验证。这与高三数学证明题的要求完全一致。

【考点连接】：在解析几何或立体几何的证明中，学生常犯的错误是“跳步”或“想当然”。文中提到的“发现并形式化的时间”提醒我们，真正的数学研究不仅是算出答案，更是要构建一个无懈可击的逻辑链条。练习书写规范的解答过程，实际上是在锻炼你的“形式化”思维，这能有效避免考试中无谓的扣分。

4. 学科背景： \LaTeX 与现代数学

\LaTeX 是理科生进入大学后的必修课，它是目前全球通用的学术文档排版系统。其核心标志就是美元符号 $\$$ 包裹的数学公式。

【职业规划】：如果你未来希望进入数学、物理、计算机等领域，提前了解这些工具的历史和逻辑（如文中提到的 Don Knuth，算法界的祖师爷级人物），能让你更早具备科研视野。

第 16 页

【原文翻译】

结构和错误检查功能加速了复杂证明的增量式开发，同时使其更加可靠。

目前这一因子远高于 1 —— 也许是 10，甚至是 100，这取决于具体的子领域以及数学家对 Lean 的熟悉程度。困难不仅在于编写证明；还在于广阔的数学领域尚未被充分形式化，甚至无法陈述你的定理。你必须要么自己构建基础，（这往往昂贵得令人望而却步），要么等待 Mathlib 的成长。

但一旦该因子降至 1 以下，我预计几乎所有人都会自愿转向形式化工作，就像他们对待 \LaTeX 那样；不需要任何强迫。为了加快工作流并增强对结果的信心，这将是顺理成章的选择。

9.3 采纳条件

要使形式化因子降至 1 以下，必须满足几个条件：

首先，Mathlib 必须足够全面，使得陈述你的定理只需要极少的基础工作。这就是“准自动形式化”(quasi-autoformalization) 变得至关重要的地方：AI 辅助可以加速库的增长，从而确保形式化数学与自然语言数学保持同步（或超越自然语言数学）。但库的增长不仅仅需要正确的证明，它还需要精良、可维护的代码。AI 必须学会编写可重用的数学代码，能够与现有基础设施无缝集成，而不仅仅是生成不可维护的形式化论证。

其次，工具必须变得更加易于获取。安装应该是毫不费力的。想象一个像 Overleaf 这样基于浏览器的 Lean 环境，你只需打开页面即可开始工作。（这在 live.lean-lang.org 和 GitHub Codespaces 等平台上几乎已经实现，尽管仍存在显著的摩擦。）

第三，AI 助手必须处理繁琐的部分。自然语言界面可以帮助编写形式化代码；自动求解器可以解决常规目标；翻译工具可以将数学论述转换为形式化支架。数学家专注于创造性工作，识别核心思想并构建论证结构，而 AI 则处理机械的形式化工作。

第四，也许也是最重要的，验证的好处必须变得即时且切实。目前，形式化是一项回报遥远的投资：你现在进行形式化是为了让他人能在你的工作基础上继续开发，或者是为了让你确信自己的证明是正确的。但如果工作时能及早发现错误，避免繁琐的情况检查，并自信地引用他人的结果，那么工作流本身就会变得更加高效。形式化因子的下降不仅是因为形式化变得更快，还因为整个研究过程得到了改善。

这个因子何时会降到 1 以下？这主要取决于第一个条件：全面的库。在持续的 AI 辅助形式化下，对于许多核心数学领域，Mathlib 可能会在 5 到 10 年内达到必要的规模。对于某些高度专业化的领域，可能需要更长时间。但一旦你的领域跨过了这个门槛，转变可能会非常迅速。

10 教学

我概述的采纳时间线侧重于数学研究，但教学代表了一种并行的、潜在的加速力量。最容易转化为形式化的人群是年轻人。为了理解其中的原因，想象一个我们纯粹通过“描述”棋步（比如使用代数记谱法）来教新手下围棋的世界。老师说：

好了同学们，这局棋是这样的：1. Nf3 Nf6 2. c4 g6 3. Nc3 Bg7 4. d4 O-O 5. Bf4 d5（这是一个易位的格林菲尔德防御）6. Qb3 dxc4 7. Qxc4 c6 8. e4 Nbd7 9. Rd1 Nb6 10. Qc5 Bg4 11. Bg5 Na4!! 哇，你能相信那步棋吗？！

象棋爱好者可以毫不费力地将这些符号转化为一系列不断演变的局面，在脑海中更新棋盘。但对于我们其他人来说，如果你能**直接给我们看棋盘**，那会容易得多！

而这种想象中的世界正是我们目前教几乎所有新手证明定理的方式：完全使用自然语言。在数学证明的每一步中，“棋盘”——即当前的数学对象、假设和目标——都在发生变化。专业数学

家可以毫不费力地追踪这一点（系统 I 思维），但对于初学者来说，这需要刻意的、容易出错的努力（系统 II 思维）。回顾我们《数学分析原理》（Baby Rudin）中证明 $\sqrt{2}$ 是无理数的例子：当证明 $\sqrt{2}$ 是无理数时，底层的“棋盘”在推理的每一行都在发生偏移。

当然，我们永远不会梦想要在自然语言证明的每一步之后写出完整的棋盘位置。首先，这会极其繁琐。其次，我们大多数人即使没有它也过得“还不错”。

【高三解读】

这篇文章探讨了数学形式化（使用计算机辅助证明工具，如 Lean）的未来趋势，特别是从研究到教学的转变。对于高三学生来说，这不仅是关于数学前沿的科普，更蕴含了深刻的学习逻辑和思维方法。

- 形式化因子与效率意识：**文中提到的“形式化因子”是指完成形式化证明的时间与编写自然语言证明的时间之比。在高三复习中，我们也存在类似的“效率因子”。比如，通过建立错题本或总结模型（类似于 Mathlib 的基础库），初期投入大（因子 > 1 ），但一旦模型成熟，后期解题速度和准确率会飞跃（因子 < 1 ）。
- AI 与数学底层逻辑：**文章指出 AI 的作用是处理“机械的形式化工作”。对于高中生，这意味着要区分“套路、运算”与“数学思维”。未来的数学家将更多关注“创造性工作”，而高三阶段对 $\epsilon - \delta$ 语言（微积分基础）或 $\sqrt{2}$ 是无理数的逻辑推理训练，本质上是在训练你的“系统 I”直觉，让你在脑海中也能像象棋大师一样实时演化“数学棋盘”。
- 系统 I 与系统 II 思维：**文中引用了心理学家丹尼尔·卡尼曼的理论。**系统 I** 是直觉、快速的；**系统 II** 是逻辑、缓慢且费力的。高三的学习目标之一就是通过大量练习，将复杂的逻辑推导（系统 II）转化为本能的直觉（系统 I）。当你看到“ $\sqrt{2}$ ”时能立刻想到反证法，就是这种转化的体现。
- 可视化与结构化学习：**文章用象棋记谱法的类比非常精妙。在解决数学大题时，很多同学会“断片”，本质上是脑海中的“棋盘”跟丢了。建议在做复杂推导（如解析几何或导数压轴题）时，利用草稿纸通过简图或列表“实时显示棋盘”，减少大脑的认知负荷，防止逻辑断层。
- 学术规范与未来展望：**文中提到的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 和 Lean 是现代数学界的标准工具。了解这些可以拓宽学术视野，明白现在的数学证明正朝着更加严谨、可验证的方向发展，这也正是高中数学要求“解题步骤严密”的终极意义。

第 17 页

【原文翻译】

事实上，有人可能会认为，在大脑中构建这些“隐形的棋盘”的能力对于成为一名数学家至关重要，正如一名强大的棋手必须学会并在脑海中视觉化多个走位一样。

但关键点在于：当使用 Lean 进行教学时，这个“游戏板”是始终存在的，它会自动生成并不断更新。初学者不必费力地从枯燥的文字叙述中重建它——他们只需观察即可。

这就是为什么在图 7.1 的示意图中，我画了一个从 Mathlib 指向教学（Teaching）的箭头，但同时也有一个直接从 Lean 本身指向教学的箭头。Mathlib 的箭头意义很明确：当库庞大且便捷时，教学会更容易。但 Lean 的箭头反映了一些不同的东西：在教学中，人们往往不想要 Mathlib 中最通用、最抽象的定义。正如我们不会在高中生接触黎曼积分（Riemann integration）之前就向他们介绍勒贝格积分（Lebesgue integration）一样，在 Lean 中，直接编写更简单、更具体的定义版本通常更有效。例如，实分析（Real Analysis）的初学者应该首先通过熟悉的 $\epsilon - \delta$ 定义来接触极限，而不是通过 Mathlib 通用的 `Filter.Tendsto`。通过这种方式，Lean 为教学提供了一个灵活的框架，允许教师根据学生的需求量身定制抽象级别。

这是一个成熟的实验领域。在众多的例子中，Patrick Massot [20] 开发了一个名为 **Verbose** 的 Lean 包装器，它以受控的自然语言呈现证明，使学生能够更轻松地进入正式语法，参见图 10.1。就我而言，我目前正在进行一项实验，在一门本科实分析课程中，我以一种类似“电子游戏”的方式进行教学，其精神类似于 Buzzard 受欢迎的《自然数游戏》（Natural Number Game），见图 10.2。

图 10.1：Massot 的 Verbose Lean 包装器，用于以受控自然语言教学实分析。

这些教学实验反映了一个更广泛的历史模式。在 1600 年之前，数学家们已经在研究严谨的代数，但它看起来像这样：

你有一个未知量。当你创建这个量的第二个副本并将其与一个已知量合并，然后将这个总和乘以你原来的未知量时，你得到了一个指定的结果。

与现代版本相比：

$$x(x + a) = b$$

后者经过一定的训练，更容易理解，也更直观。

同样，在形式化之前，我们很少在证明中明确命名我们的假设。我怀疑随着形式化系统影响数学阐述，这种情况可能会改变。正如代数记号彻底改变了我们对思考的方式一样...

页脚:

第 17 页 | 版权所有 © 2025 SIAM。未经授权严禁翻印本文。

【高三解读】

高三数学深度解读：从形式化证明到代数演变

本页内容探讨了数学教学的现代化工具（Lean 证明辅助器）以及数学语言演变的史实。对于正处于高中向大学衔接阶段的高三学生来说，这不仅关于工具的介绍，更是关于数学本质的深刻思考。

1. 核心知识点：夹逼定理（The Squeeze Theorem）

图 10.1 展示的是用 Lean 语言证明的“夹逼定理”。

- 定义回顾：**若三个数列满足 $u_n \leq v_n \leq w_n$ ，且 $\lim_{n \rightarrow \infty} u_n = l$ ， $\lim_{n \rightarrow \infty} w_n = l$ ，则 $\lim_{n \rightarrow \infty} v_n = l$ 。
- 高三视角：**在高考中，夹逼定理通常用于求复杂数列的极限或在不等式证明题中作为放缩法的理论依据。而在大学阶段，你需要学习图中显示的 $\epsilon - N$ 语言来严谨地定义和证明它。图中左侧代码展示了从“固定 $\epsilon > 0$ ”到“寻找满足条件的 N ”的标准化数学叙述过程。

2. 解题技巧： $\epsilon - \delta$ 语言的逻辑（微积分基础）

文中提到初学者应先接触 $\epsilon - \delta$ 定义。这是大学数学分析（Real Analysis）的第一道坎：

- 常见陷阱：**学生容易混淆任意的 ϵ （误差范围）和存在的 N 或 δ （控制范围）。
- 理解窍门：**把这看作一个“甲方乙方”的游戏。甲方给出任意小的误差允许值 ϵ ，乙方必须证明只要自变量足够接近某个值（或者 n 足够大），函数值与极限的差距就能小于 ϵ 。只要乙方总能赢得游戏，极限就存在。

3. 学科背景：代数符号化的意义

文中对比了 1600 年前的“文字代数”和现代的“符号代数” $x(x + a) = b$ 。

- 思维进化：**符号化不仅是缩写，它极大地减轻了认知负荷（Cognitive Load）。在高三复杂的解析几何或导数大题中，合理的符号化设置（如设而不求、参数代换）是化繁为简的关键。文字叙述容易让人在大脑中“断线”，而符号能像文中提到的“棋盘”一样，让你视觉化地推演逻辑。

4. 学习建议：从“感性认知”向“理性逻辑”过渡

- **注重假设命名：**文中提到在 Lean 中要给每一个假设起名字（如 h, h' ）。在写高中数学大题时，养成清晰标注“由题意知”、“由引理可知”的习惯，有助于理清证明链条，防止漏掉关键步骤。
- **利用工具可视化：**虽然普通高中生不一定接触 Lean，但可以利用几何画板或 GeoGebra 来实现文中提到的“自动生成的棋盘”，直观观察函数极限的变化过程。

第 18 页

【原文翻译】

图 10.2: 实分析游戏，网址：

<https://adam.math.hhu.de/#/g/AlexKontorovich/RealAnalysisGame>

方程式中，形式系统所要求的精确性可能会改变我们交流数学论据的方式，引导我们在自然语言数学中也能更清晰、更准确地表达。早期的迹象表明，Lean 不仅仅是数学研究的工具：它有望改变数学教学法和数学话语本身。

11 理解与交流

我所概述的采用时间表和教学效益主要集中在效率和可访问性上。但形式化也可能从根本上改变数学家理解和交流数学思想本身的方式——这些变化可能会因为超越单纯生产力增益的原因而加速采用。

正如瑟斯顿（Thurston）在《论数学的证明与进展》（On Proof and Progress in Mathematics）[25] 中雄辩地指出的那样，形式化证明（即使是在自然语言中）仅仅代表了真正数学理解的起点，而非终点。问题在于，形式系统是否实际上能增强而不是束缚数学洞察力的发展及其交流。

当前的大规模形式化项目提供了这种潜力的早期一瞥。Massot 的“Lean 蓝图”技术实现了自然语言 \LaTeX 与相应形式化 Lean 代码之间的无缝转换，同时生成交互式依赖图，使复杂论据的逻辑结构可视化，见图 11.1。这些图表服务于一个直接的工程目的——向协作者展示形式化树的哪些“叶子”需要注意——但它们也指向了更深远的方向。

图 11.1: [16] 中依赖图的一部分

设想一下，当每一个定理都存在于一个显式的、可导航的逻辑依赖网络中时，数学理解将如何演变。数学家不再将结果视为线性阐述中孤立的事实，而是可以在不同的细节水平上探索论

据，从高层直觉一直缩放到...

18 页 | 版权所有 © 2025 SIAM。未经许可禁止复制本文。

【高三解读】

高三数学深度解读：从“实分析游戏”看数学思维的进化

这份文档探讨了数学形式化（Formalization）对数学理解与交流的深远影响。对于正处于备考冲刺阶段的高三学生来说，这不仅是关于未来大学数学的预览，更蕴含了提升当前解题逻辑的重要启示。

1. 知识点衔接：从“直观感悟”到“严格证明”

- ϵ - δ 定义**：图中提到了“ ϵ - δ 微积分”。在高三导数和极限的学习中，我们通常采用的是直观的描述性定义（如“当 x 无限接近 x_0 时”）。而大学实分析则要求使用严格的数学语言进行证明。这告诉我们：**数学的本质是逻辑的严密性**，而非仅仅是算得准。
- 逻辑连接词与充要条件**：形式化系统的核心是逻辑。在高考数学解答题中，能否清晰地使用“因为...所以...”、“若且唯若”等逻辑连接词，直接决定了证明过程的得分。图 11.1 展示的“依赖图”实际上就是复杂证明逻辑的“思维导图”。

2. 解题技巧：结构化思维（Structural Thinking）

- 依赖图的应用**：图 11.1 展示了定理之间的相互支撑关系。在做高考综合大题（如圆锥曲线或导数压轴题）时，你可以尝试在草稿纸上画出逻辑依赖线：第一问的结论是如何支撑第二问的？需要调用哪些性质（如切线方程、韦达定理）？这种“可视化逻辑”能有效防止解题思路中断。
- “叶子”节点意识**：文中提到的“叶子”指证明中的基础步骤。解题时，将大问题拆解为多个小“叶子”，逐个击破，是处理复杂数学问题的通用策略。

3. 学科背景与未来趋势

- Lean 证明辅助器**：这是一种计算机程序，能检查数学证明是否正确。这预示着未来的数学学习将更加互动化。作为新时代的考生，理解“逻辑规则”比死记硬背公式更重要。
- 瑟斯顿的观点**：证明只是理解的开始。这意味着你在做完一道题后，不应止步于对答案，而应思考：这个结论能推广吗？它背后的数学本质是什么？这种深度思考正是高三学生从“题海”中跳出、提升数学素养的关键。

4. 常见陷阱提醒

- **逻辑断层**：很多学生在证明题中会跳步，认为“显而易见”。但在形式化系统中，任何没有依据的跳步都会报错。在考试中，务必确保每一步推导都有对应的定理或性质支撑。
- **定义不清**：形式化要求对符号有极精确的定义。高三学生常在概念性题目（如集合、逻辑用语）上丢分，往往是因为对定义的理解存在偏差。

第 19 页

【原文翻译】

根据需要提供细粒度的形式化步骤。学习代数拓扑的学生可以从宏观的概念景观开始，仅在准备好时才深入研究具体的同调计算。审阅论文的专家可以高效地验证整体逻辑结构，仅针对新颖或可疑的断言检查详细证明。

这代表了与当前数学交流方式相比的质变，目前的数学交流本质上仍然是线性和静态的。期刊文章²以固定顺序呈现论点，要求读者要么凭信任接受断言，要么费力地验证每一步。形式化系统可以实现我们所谓的“多分辨率证明”——即能够适应读者的专业水平和兴趣的数学论证。

其影响超出了个人理解，扩展到协作数学实践。当复杂的论证出现分歧时，形式化系统提供了客观的仲裁。在他人工作的基础上，数学家可以放心地整合结果，而不必担心隐藏的错误。数学知识的累积性质——每一代人都建立在先前的洞见之上——随着基础可靠性的提高可能会加速。

这可能引发数学责任本身深刻的转变。目前，数学问责制是高度个人化且垂直的。在研究生期间，我被允许引用一个结果——德利涅（Deligne）对韦伊猜想（Weil conjectures）的证明——并被要求从基础开始了解其他一切。即使现在，当我引用另一篇论文的定理时，我感到个人有责任理解其证明。我们就像必须自己开采金属、锻造零件并亲自组装一切的工程师。

但形式化，通过对每个假设和结论的明确规范，可以实现真正的模块化数学。想象一下，如果我们能放心地使用“现成的”组件，我们能证明什么样的定理！当形式化陈述精确地指明了它们的假设和保证时，数学家就可以像工程师信任标准化组件那样，以同样的信心在他人工作的基础上进行构建。这种从垂直责任到水平协作的转变将极大地加速数学的进步。

然而，重大挑战依然存在。当前的形式语言优先考虑逻辑正确性而非人类直觉，往往掩盖了驱动数学进步的概念性见解。风险在于，形式化可能会割裂数学文化，在那些在形式化系统中工作的人和那些在自然语言中操作的人之间制造鸿沟。

形式化系统最终是增强还是约束数学理解，可能取决于我们在开发能够保留并放大人类数学直觉而非取代它的工具方面的成功。目前正在进行的交互式可视化、自然语言界面和协作形式化环境方面的实验，代表了迈向我们所谓的“未来论文”的早期步骤——这种数学交流将同时具备严谨性、易读性和智力启发性。

这种转变的意义超出了数学本身。如果形式化系统能够真正增强数学理解和交流，它们可能会为精密和累积知识至关重要的其他领域的严谨推理提供模型。数学未来的形态可能会预示人类处理世界复杂推理方式的更广泛变革。

12. 结论。

我所勾勒的愿景——准自动形式化加速库的增长、形式化因子降至 1 以下，以及数学实践向形式化系统的迁移——既代表了巨大的机遇，也包含着相当大的不确定性。这种转型是否如预测的那样展开，取决于能否解决当前 AI 系统的随机性质与数学证明的确定性要求之间的根本张力。

证据指向相反的方向。一方面，从 2023 年的零 IMO 能力到 2025 年的金牌表现，表明 AI 在数学方面的能力正在以显著的速度进步。像 AlphaProof 这样的系统在完成复杂形式化目标（甚至包括那些需要数百个复杂步骤的目标）方面的成功，证明了复杂的数学推理在当前技术的可及范围内。

然而核心挑战依然存在：即使是一个每步准确率为 99.99% 的假设性 AI 系统，在将数千个推理步骤串联起来时，也会产生不可靠的结果。大语言模型的随机本质——它们对下个 token 概率分布的根本依赖——似乎与数学所要求的确定性正确性深度错位。

这让我们回到了蒂姆·高尔斯（Tim Gowers）在 2000 年提出的具有预见性的愿景。他在新千禧年之初写作，预言了即将到来的数学领域人机协作的“黄金时代”，但也警告说，这样一个时代...

²关于形式化时代期刊审稿过程可能是什么样子的更多信息，见 [14]。

【高三解读】

这是一篇探讨数学形式化与人工智能（AI）在数学领域应用的前沿文章。对于高三学生来说，这不仅是英语长难句阅读的绝佳材料，更是跨学科思考数学本质、逻辑严密性与未来科技趋势的深度素材。

1. 核心知识点解析

- **德利涅对韦伊猜想的证明 (Deligne's proof of the Weil conjectures)**：这是 20 世纪数学的重大里程碑。在数学学习中，这提醒我们要重视基础。作者提到的“垂直责任”正是高中

数学强调的：你必须亲手推导过公式（如正弦定理、导数公式），才能在解题时真正做到信手拈来。

- **概率论与逻辑链条**：文中提到的 99.99% 准确率在数千步推理后的失效，是一个极佳的数学模型应用。假设一步推理正确的概率为 $p = 0.9999$ ，若证明过程有 $n = 10000$ 步，则总正确率 $P = p^n = (0.9999)^{10000} \approx 0.3678$ 。这解释了为什么在高中证明题中，哪怕只是一个微小的“显然”跳步，如果缺乏逻辑严密性，整个得分点就会崩塌。

2. 学科思维训练

- **模块化思维 (Modular Mathematics)**：这与我们解决复杂几何题或数列题的方法类似。我们将复杂的题目拆解成若干个“现成组件”（如：基本不等式、等差中项性质、辅助圆模型）。只有每个组件都绝对可靠，最终的结论才成立。
- **直觉与形式化 (Intuition vs. Formalization)**：高中数学常说“数形结合”。直觉是那一瞬间的火花（形），而形式化是严谨的算式（数）。高考不仅考察你是否有解题灵感，更考察你是否能用标准的数学语言将其表达出来。

3. 解题技巧与陷阱

- **严谨性要求**：文中提到的 AI 随机性 (Stochastic) 与数学确定性 (Deterministic) 的矛盾，正是学生在考场上常犯的错误——“靠感觉解题”而非“靠逻辑解题”。AI 的“幻觉”在学生身上表现为对结论的盲目猜测。记住：在数学的世界里，99.99% 的可能性不等于 100% 的证明。
- **多分辨率思维**：阅读长篇材料（如数学建模题）时，应学习文中的“多分辨率”策略。第一遍快速浏览（低分辨率），确定大致的逻辑结构和考查意图；第二遍针对具体计算和推导（高分辨率），逐字逐句攻克难题。

4. 学科视野拓展

- **AlphaProof 与 IMO**：文中提到的 AI 在国际数学奥林匹克 (IMO) 中的表现，反映了算法在组合逻辑和搜索空间上的优势。但正如文末所言，AI 仍然缺乏人类那种“智力启发性”的直觉。作为未来的大学生，掌握如何与 AI 协作 (Human-computer collaboration) 将是比单纯计算更重要的技能。

第 20 页

【原文翻译】

可能很短暂 [11]:

下一个阶段可能是这样一个阶段：只有极少数杰出的数学家能够发现计算机无法触及的证明……最终，数学家的工作将仅仅是学习如何有效地使用定理证明机，并为它们寻找有趣的应用。这将是一项宝贵的技能，但它很难再被称为我们今天所熟知的纯数学。

我对这样一个“黄金时代”（如果它真的到来的话）持有一种更为乐观的看法。数学有着悠久的历史，许多问题起初看起来极其困难，但后来证明比预期的更容易解决。有限域卡基亚猜想（finite field Kakeya conjecture）就体现了这种模式：在顶尖专家多年取得微小进展之后，Zeev Dvir 在 2008 年 [6] 利用代数几何中的初等技术解决了整个问题，揭示了那些看似深奥的困难实际上相当直接。

这表明，AI 辅助可能帮助我们发现许多类似的、比看起来更易处理的开放性问题，即遍布数学领域的某种“隐藏的低垂果实”。AI 探索巨大解空间的能力与形式验证对正确性的保证相结合，可以解锁那些并非由于根本性困难、而仅仅是因为没有人类有毅力去尝试正确的组合而一直未能解决的结果。

当然，在这些成功之后，可能依然存在的是那些真正困难的问题；在这里，我完全预料到像黎曼猜想（Riemann Hypothesis）挑战对于即使是尖端的 AI 系统来说也将保持艰巨。但是，如果我们真的进入了一个形式化、AI 辅助数学成为常态的时期，“英雄式数学家”的经典角色可能会以新的形式存在：确定哪些问题值得追求，识别看似迥异的领域何时可能产生联系，并提供引导自动化搜索的概念性见解。

数学未来的形态最终取决于我们今天在开发这些工具时做出的选择。如果我们优先考虑纯自动化而非人类洞察力，我们就有可能创造出能够解决问题但无法推进理解的系统。但如果我们成功构建出能够增强而非取代数学直觉的 AI —— 这些系统处理形式化的机械方面，同时为人类创造力保留空间 —— 我们可能见证的不是纯数学的终结，而是它转化为比以前更强大、更美丽的东西。

无论这一愿景被证明具有先见之明还是仅仅是过于乐观，数学界正处于一个显著的拐点。接下来的十年可能会决定形式验证是否会像今天的 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 一样成为数学实践的基础，还是仅作为最专注的从业者的专业工具。无论如何，我们正经历着数学推理史上一个关键时刻，这个故事的最后一章尚待谱写。

致谢。我走向 Lean 之路始于 Kevin Buzzard 的在线讲座和鼓励，并在 Heather Macbeth 耐心的指导下得以继续。Paul Nelson 是第一个向我展示大语言模型卓越潜力的人，而 Christian Szegedy 让我意识到自动形式化的核心重要性。在此过程中，我受益于与 Sanjeev Arora,

Jeremy Avigad, Tim Gowers, Thomas Hubert, Ian Jauslin, Chi Jin, JJ Jung, Jeremy Kahn, Patrick Massot, Konstantin Mischaikow, Terry Tao, Akshay Venkatesh 以及充满活力的 Mathlib 社区的启发性交流。最重要的是，我感谢 Leo de Moura，他创造了 Lean —— 建立在类型论和证明助手的丰富传统之上 —— 以以前难以想象的方式向实践研究者开启了形式数学的大门。

参考文献

- [1] N. ALON, J. BOURGAIN, A. CONNES, M. GROMOV, AND V. MILMAN, eds., *GAFA 2000*, Birkhäuser Verlag, Basel, 2000. Visions in mathematics. Towards 2000, Geom. Funct. Anal. 2000, Special Volume, Part I.
- [2] K. BUZZARD, *What is the point of computers? A question for pure mathematicians*, in ICM—International Congress of Mathematicians. Vol. 2. Plenary lectures, EMS Press, Berlin, [2023] ©2023, pp. 578–608.
- [3] K. BUZZARD, *Private communication*, 2025.
- [4] H. B. CURRY, *Functionality in combinatory logic*, Proceedings of the National Academy of Sciences, 20 (1934), pp. 584–590, <https://doi.org/10.1073/pnas.20.11.584>.

【高三解读】

这份学术文献探讨了人工智能（AI）与数学研究的未来关系，对于即将步入大学、接触高等数学的高三学生来说，具有极强的启发性：

- 数学工具的演变与传承**：文中提到了 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 。对于高中生来说， $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 是排版数学公式的标准语言。作者将“形式验证”（如 Lean 语言）与 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 类比，预示着未来的数学家可能不再仅仅用纸笔推导，而是通过编程和 AI 来验证逻辑的严密性。这提醒同学们，在大学的学习中，除了扎实的演算功底，逻辑的严谨性和对新工具的掌握同样重要。
- “低垂的果实”与初等技术**：文中提到的卡基亚猜想（Kakeya conjecture）案例非常有教育意义。它说明有些看似高不可攀的难题，最后竟是用“初等技术”解决的。这告诉我们，高中所学的代数、几何基础非常重要，深奥的数学大厦往往建立在极其简洁的逻辑基础之上。不要轻视基础，真正的创新往往来自于对基础工具的巧妙运用。
- 数学家的核心价值——直觉与洞察**：作者强调，即使 AI 能够完成机械的证明过程，人类数学家的角色也不会消失。人类的优势在于“直觉”、“联结迥异领域的能力”以及“提出有价值问题的能力”。这与高考评价体系中强调的“创新性”和“综合性”不谋而合。在备考中，我们不应只做“刷题机器”，而要思考题目背后的本质逻辑和数学美感。
- 跨学科视野**：致谢名单中出现了陶哲轩（Terry Tao）等数学巨匠。同学们可以关注这些数学家的工作，了解现代数学是如何与计算机科学（如大语言模型、自动化证明）交叉融合的。这种跨学科的思维是未来顶尖人才必备的素质。
- 学科背景点拨**
 - 黎曼猜想（Riemann Hypothesis）**：文中提到的顶级难题，关乎素数的分布，是克雷数学研究所悬赏百万美元的七大“千禧年大奖难题”之一。
 - 形式验证（Formal Verification）**：指使用数学方法证明算法或逻辑的正确性，这在航空航天、芯片设计等不容许出错的领域至关重要。

第 21 页

【原文翻译】

[5] N. G. DE BRUIJN, 《Automath 项目综述》, 载于《致 H.B. Curry: 组合逻辑、 λ 演算与形式主义论文集》, J. P. Seldin 和 J. R. Hindley 编, 学术出版社, 纽约、伦敦, 1980年, 第 579–606 页。[6] Z. DVIR, 《关于有限域中 Kakeya 集的大小》, 《美国数学学会杂志》, 22 (2009), 第 1093–1097 页。[7] M. GANESALINGAM 和 W. T. GOWERS, 《一个具有人类风格输出的全自动定理证明器》, 《自动推理杂志》, 58 (2017), 第 253–291 页。[8] D. GOLDFELD, 《虚二次域的 Gauss 类数问题》, 《美国数学学会公告 (新系列)》, 13 (1985), 第 23–37 页。[9] B. GOMES 和 A. KONTOROVICH, 《Lean 中的黎曼猜想》, 2020年。[10] GOOGLE DEEPMIND, 《AI 在解决国际数学奥林匹克 (IMO) 问题上达到银牌水平》, 2024 年。[11] W. T. GOWERS, 《粗糙结构与分类》, GAFA 2000 (特拉维夫, 1999年)。[12] W. A. HOWARD, 《作为类型的公式这一构造概念》, 载于《致 H.B. Curry: 组合逻辑、 λ 演算与形式主义论文集》, 1980年。[13] A. Q. JIANG 等, 《草拟、勾勒与证明: 利用非正式证明引导形式化定理证明器》, ICLR 2023。[14] A. KONTOROVICH, 《前言: 交互式定理证明器专题》, 《实验数学》, 31 (2022)。[15] A. KONTOROVICH 等, 《上半平面的 Moebius 作用》, 2021 年。[16] A. KONTOROVICH 和 T. TAO, 《素数定理及更多》, 2024年。[17] Y. LIN 等, 《Goedel-prover: 开源自动定理证明的前沿模型》, 2025年。[18] Y. LIN 等, 《Goedel-prover-v2: 通过脚手架数据合成和自我修正扩展形式化定理证明》, 2025年。[19] D. LOEFFLER 和 M. STOLL, 《在 Lean 中形式化 ζ 函数和 L 函数》, 《形式化数学年刊》, 2025 年。[20] P. MASSOT, 《使用 Lean 和受控自然语言教学数学》, ITP 2024。

【高三解读】

这份参考文献列表不仅是一份学术清单, 更是当代数学发展的风向标。对于高三学生, 可以从以下三个维度深度解读: 1. **AI 与数学的深度融合**: 第 [10]、[17] 和 [18] 条提到的 Google DeepMind 银牌成果及 Goedel-prover 模型, 反映了人工智能正在攻克人类智力的巅峰——数学逻辑推理。高三学生熟悉的 IMO 问题, 如今已成为衡量 AI 逻辑能力的标准。2. **形式化证明与 Lean 语言**: 文中多次出现 ‘Lean’ (如 [9], [15], [19], [20])。这是一种程序语言, 旨在将数学定理转化为计算机可校验的代码。这意味着未来的数学研究不仅需要纸笔, 更需要编程思维。第 [20] 条甚至讨论了如何用这种方式教学, 这预示着数学学习模式的潜在变革。3. **顶尖数学家的身影**: 名单中出现了 Timothy Gowers (第 [7], [11] 条) 和 Terence Tao (陶哲轩, 第 [16] 条) 等菲尔兹奖得主。他们关注的不仅仅是高深理论, 还有如何利用自动定理证明工具来探索经典难题, 如黎曼猜想 (ζ 函数) 和素数定理。这告诉我们, 即便是在最传统的数学领域, 技术创新也是推动学科前进的核心动力。

第 22 页

【原文翻译】

参考文献

[21] W. RUDIN, 《数学分析原理》, 国际纯粹与应用数学丛书, McGraw-Hill, 纽约, 第 3 版, 1976 年。

[22] P. SONG, K. YANG, 和 A. ANANDKUMAR, 《Lean 辅助驾驶: 大语言模型作为 Lean 定理证明的辅助驾驶员》, 载于 2024 年 NeurIPS 数学推理与人工智能研讨会, 2024 年, <https://arxiv.org/abs/2404.12534>。

[23] R. S. SUTTON, 《痛苦的教训》。
<http://www.incompleteideas.net/IncIdeas/BitterLesson.html>, 2019 年。访问日期: 2025-09-29。

[24] C. SZEGEDY, 《迈向自动形式化和通用人工智能的一条充满希望的道路》, 载于《智能计算机数学》, 计算机科学讲义第 12236 卷, Springer, 2020 年, 第 3-20 页。

[25] W. P. THURSTON, 《论数学中的证明与进展》, 美国数学学会简报, 30 (1994), 第 161–177 页。

[26] F. WIEDIJK, 《de Bruijn 因子》, (2000), <https://www.cs.ru.nl/~freek/factor/factor.pdf>。技术报告, 奈梅亨拉德堡德大学。

[27] 张益唐, 《素数间的有界间隔》, 数学年刊 (2), 179 (2014), 第 1121–1174 页, <https://doi.org/10.4007/annals.2014.179.3.7>。

第 22 页

版权所有 © 2025 SIAM。未经授权严禁转载本文。

【高三解读】

高三学术视角解读：从经典分析到现代 AI 推理

这份参考文献列表涵盖了从数学基础理论到最前沿 AI 辅助证明的跨度，对于高三学生了解“现代数学是如何研究的”具有极高的启发价值。

1. 数学学习的阶梯：从计算到严谨证明

- [21] Rudin 的《数学分析原理》：这本书在数学界被称为“Baby Rudin”，是大学数学系学生的入门“神作”。高三学生目前接触的是微积分初步（导数、积分），主要关注计算。而 Rudin 的书代表了数学的下一步：**严谨化**。它教会我们如何用逻辑推导（如 $\epsilon - \delta$ 语言）重构整个微积分体系，这是从“算术者”向“数学家”转化的必经之路。

2. 数学巨匠的哲学：证明的本质是什么？

- [25] Thurston 的论文：威廉·瑟斯顿是菲尔兹奖得主。他探讨的是：数学进步仅仅是堆砌证明吗？不是。数学是为了**理解**。这提醒高三学生，在刷题过程中，理解一个概念背后的几何直观和逻辑脉络，比单纯背诵公式要重要得多。

3. 中国骄傲：张益唐与李生素数猜想

- [27] 张益唐的经典论文：这是这份名单中最贴近中国学生的内容。张益唐在 2013 年证明了存在无穷多对素数，其间隔小于 7×10^7 。虽然这个数字很大，但他打破了从“无限”到“有限”的壁垒。他的故事激励我们要有强大的毅力：他在学术边缘沉潜多年，最终一鸣惊人。这正是高三学子在攻坚克难时需要的“板凳要坐十年冷”的精神。

4. 未来的方向：AI 推理与定理证明

- [22] & [24] AI 辅助证明：这些文献讨论了如何利用大语言模型（如 GPT 类的技术）在 Lean 等交互式定理证明器中协助人类进行数学推理。这预示着未来：数学家可能不再只是用纸笔演算，而是与 AI 共同探索未知的数学真理。对于即将步入大学的你们，掌握计算机工具（如 LaTeX 或编程语言）将成为研究数学的新常态。

5. 跨学科思维：计算的“痛苦教训”

- [23] Sutton 的《痛苦的教训》：强化学习之父 Rich Sutton 提出，在 AI 发展史上，利用算力的通用算法（暴力计算）最终总能战胜人类精心设计的启发式规则。这反映了现代科学的一个趋势：**计算能力正在重塑逻辑推理的边界**。这种思维能帮助同学们在理综学习中更好地理解算法与模型的重要性。