Automatisation par LLM de preuves formelles pour l'analyse réelle et numérique

mai 2025

1 Informations générales

Direction **Co-direction** **co-encadrements**	Micaela Mayero : mayero [@] lipn.univ-paris13.fr Sylvie Boldo : sylvie.boldo [@] inria.fr François Clément : francois.clement [@] inria.fr Vincent Martin : vincent.martin [@] utc.fr
Lieu	Equipe LoCal ; LIPN - CNRS UMR 7030 Université Sorbonne Paris Nord (USPN) 99 av. JB Clément ; 93430 VILLETANEUSE *Des groupes de travail réguliers auront lieu à Orsay et à Paris*

2 Résumé

Cette proposition de thèse s'inscrit dans la continuité des travaux qui consistent à développer et mettre en application des méthodes permettant de démontrer formellement en Rocq (ex Coq) la correction de programmes issus du domaine de l'analyse numérique.

Les formalisations et preuves de correction de ces programmes nécessitent un travail important de formalisation des mathématiques (analyse réelle et algèbre). Nous pensons que les techniques d'IA basées sur des LLMs peuvent nous permettre d'accélérer nos formalisations mathématiques en Rocq.

3 Contexte et motivations

Notre objectif à long terme est de fournir plus de garanties aux programmes de simulation numérique, car ils sont largement utilisés, de la santé à la construction. En particulier nous nous intéresserons à l'une des méthodes les plus populaires pour résoudre les équations aux dérivées partielles, la méthode des éléments finis (MEF).

De manière informelle, la méthode des éléments finis est utilisée pour approcher les solutions d'équations aux dérivées partielles (EDP). Ces équations sont utilisées pour modéliser un large éventail de problèmes, notamment la mécanique des fluides (équations d'Euler et de Navier-Stokes), l'électromagnétisme (équation de Maxwell), l'équation de la chaleur (Fourier), la mécanique (élasticité), la mécanique quantique (Schrödinger et Heisenberg), les prévisions météorologiques et l'aéronautique, pour n'en citer que quelques-uns.

Les solutions exactes de ces EDP ne sont généralement pas calculables. Pour pouvoir calculer une solution approximative, l'idée est de discrétiser, c'est-à-dire de découper le problème à résoudre en petits morceaux. Nous créons alors un maillage composé d'éléments "simples" sur lesquels nous

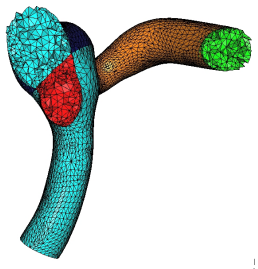

FIGURE 1 – Maillage 3D d'un anévrisme cérébral (les couleurs permettent uniquement de distinguer les zones).

savons comment effectuer les calculs. Nous passons ainsi d'un problème continu à un problème discret. Ces éléments sont appelés "éléments finis". Un maillage est alors constitué de ces éléments finis qui couvrent des surfaces plus petites que la surface ou le volume total, voir la Figure 1.

Pour effectuer des calculs d'approximation sur ces différents éléments, appelés "éléments courants", un changement de cadre de référence est nécessaire. Il s'agit de changer la représentation de "l'élément de référence", par exemple, en 2D, un triangle rectangle de côté 1 (voir Figure 2), où les calculs sont faciles. La méthode pour les utiliser afin d'approcher solutions au problème global est la "méthode des éléments finis".

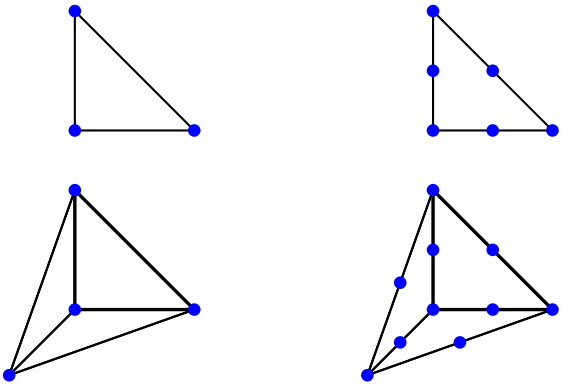

FIGURE 2 – Élément fini de Lagrange ${}^{Lag}\mathbb{P}_k^d$ en dimention $d = 2$ et $d = 3$ avec comme degré d'approximation $k = 1$ (à gauche) où les points d'approximation sont les sommets et $k = 2$ (à droite) où l'on ajoute le milieu des arrêtes.

Les preuves de programmes numériques présentent certaines spécificités récurrentes. Elles utilisent intensivement les nombres réels et les nombres flottants, des notions de dérivation, d'intégration, d'espaces et de sous-espaces, de polynômes, de combinaisons linéaires, etc.

Mais tout cela a un coût important de développement : la bibliothèque coq-num-analysis (`https://lipn.univ-paris13.fr/coq-num-analysis/`) que nous développons correspond à plus de 150 fichiers, 9000 définitions et lemmes pour un total d'environ 85 kLoC et 3 publications [BCF+17, BCF+22, BCM+23].

Ce coût est commun à toutes les grandes bibliothèques formelles. Il est donc naturel que l'interaction entre IA et preuves formelles existe. Des tactiques automatiques généralistes ou dédiées existent depuis longtemps en Rocq [DM01]. Mais l'essor récent des LLMs a conduit également à des environnements ou à des tactiques les utilisant. Ainsi rien qu'en Rocq, CoqPilot [KSKP] offre une expérience d'aide dans VSCode similaire à Copilot en proposant des morceaux de preuves. Tactician [BUG] propose également des début de preuves et est capable de synthétiser des preuves en prenant en compte les bibliothèques importées. Beaucoup d'autres outils basés sur des LLMs ou de l'IA existent dans tous les assistants de preuves récents. C'est même l'objet d'une conférence annuelle depuis 2016 `https://aitp-conference.org/`. La référence la plus récente est en Lean [OY]

qui mélange habilement les preuves automatiques usuelles avec la génération de sous-preuves par LLM.

Un sujet relié est l'autoformalisation, qui consiste à transformer un texte en langage naturel en langage formel (parfois vice versa) [WJL+22, Pat24]. Cela pourrait permettre de traduire des livres mathématiques en bibliothèques formelles, que ce soient les énoncés ou les démonstrations.

Nous allons commencer par nous intéresser à des mathématiques plus simples. En particulier, nous avons des exemples de théorèmes Coq de niveau baccalauréat que nous avons développé pour des questions didactiques [BCH+24] au sein du projet LiberAbaci. Ces feuilles d'exercices feront d'excellents premiers exemples pour tester des LLMs : sont-ils meilleurs que des élèves de terminale ?

4 Principales étapes de la thèse

La thèse pourra s'articuler de la manière suivante :

1. Le stage en cours de Suryansh Shrivastlava se concentre sur l'étude des méthodes et des outils existants dans le domaine de l'utilisation de l'IA pour la recherche de preuves dans les théorèmes prouveurs. À ce jour, Suryansh est en train de tester les outils tels que CoqPilot, Tactician. À la fin du stage, des exemples de difficultés moyennes issus du projet LiberAbaci pourront être traités et ils nous donneront une idée précise des avancées du domaine.
   Une autre piste est en cours également, il s'agit d'étudier les possibles développements de tactiques spécifiques aux éléments finis, mais en utilisant une méthode plus usuelle qu'est le langage de tactiques Ltac [1].
2. La thèse sera exactement la suite du stage. Le début pourra se concentrer sur l'automatisation de preuves choisies issues de Coquelicot sur les dérivées, par exemple.
3. Toujours par rapport à Coquelicot, une étape très intéressante pour la suite serait d'arriver à montrer automatiquement des exercices d'analyse du baccalauréat de mathématiques, comme fait à la main en 2013 par Catherine Lelay avec Coquelicot [BLM15].
4. En parallèle l'autre piste d'automatisation pourra être approfondie, en utilisant des méthodes usuelles adaptées au contexte.
5. Ensuite, il devrait être possible de généraliser l'automatisation du développement et des preuves autour de la méthode des éléments finis. Mettre l'accent sur les preuves d'éléments finis utilisants des cuboïdes nous semble un objectif atteignable.

Une difficulté dont nous sommes conscients est de veiller à avoir des preuves portables et déterministes.

Références

[BCF+17] Sylvie Boldo, François Clément, Florian Faissole, Vincent Martin, and Micaela Mayero. A Coq formal proof of the Lax–Milgram theorem. In *Proc. of the 6th ACM SIGPLAN Internat. Conf. on Certified Programs and Proofs (CPP 2017)*, pages 79–89. Association for Computing Machinery, New York, 2017.

[BCF+22] Sylvie Boldo, François Clément, Florian Faissole, Vincent Martin, and Micaela Mayero. A Coq formalization of Lebesgue integration of nonnegative functions. *J. Autom. Reason.*, 66(2) :175–213, 2022.

[BCH+24] Sylvie Boldo, François Clément, David Hamelin, Micaela Mayero, and Pierre Rousselin. Teaching divisibility and binomials with Coq. In *Proc. of the 13th International Workshop on Theorem proving components for Educational software (ThEdu 2024)*, volume 419, pages 124–139. EPTCS, 2024.

1. `https://coq.inria.fr/doc/V8.20.0/refman/proofs/creating-tactics/index.html`

[BCM+23] S. Boldo, F. Clément, V. Martin, M. Mayero, and H. Mouhcine. A Coq formalization of Lebesgue induction principle and Tonelli's theorem. In M. Chechik, JP. Katoen, and M. Leucker, editors, *Proc. of the 25th Internat. Symp. on Formal Methods (FM 2023)*, volume 14000 of *LNCS*, pages 39–55. Springer, Cham, 2023.

[BLM15] Sylvie Boldo, Catherine Lelay, and Guillaume Melquiond. Coquelicot : A user-friendly library of real analysis for Coq. *Math. Comput. Sci.*, 9(1) :41–62, 2015.

[BUG] Lasse Blaauwbroek, Josef Urban, and Herman Geuvers. The Tactician (extended version) : A Seamless, Interactive Tactic Learner and Prover for Coq. `https://arxiv.org/abs/2008.00120`.

[DM01] David Delahaye and Micaela Mayero. `Field` : une procédure de décision pour les nombres réels en Coq. In *Journées Francophones des Langages Applicatifs, Pontarlier*. INRIA, Janvier 2001.

[KSKP] Andrei Kozyrev, Gleb Solovev, Nikita Khramov, and Anton Podkopaev. CoqPilot, a plugin for LLM-based generation of proofs. `https://arxiv.org/abs/2410.19605`.

[OY] Azim Ospanov and Roozbeh Yousefzadeh. APOLLO : Automated LLM and Lean Collaboration for Advanced Formal Reasoning. `https://arxiv.org/abs/2505.05758`.

[Pat24] Shashank Pathak. Gflean : An autoformalisation framework for lean via GF. *CoRR*, abs/2404.01234, 2024.

[WJL+22] Yuhuai Wu, Albert Qiaochu Jiang, Wenda Li, Markus N. Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35 : Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.