DOCUMENT 4 : RESEARCH PROPOSAL

AIM²P — Proposition de recherche v0.1 : de la question aux premières mesures

Phase : Phase 0 (cadrage scientifique) · Projet : AIM²P Initiative · Version : 0.1

Synthèse

Cette proposition transforme le cadrage de la Phase 0 en un plan de mesure pré-enregistré pour les 90 jours qui vont du 17 novembre 2026 à la mi-février 2027. Elle s'appuie sur l'audit de la littérature arrêté au 17 novembre 2026. Cet audit montre que l'association LLM + calcul formel + Lean existe déjà pour des domaines précis : AIM²P ne revendique donc pas une première, mais des **mesures** que personne n'a encore publiées. Le protocole repose sur **AMP-Bench-mini**, 40 items audités et pondérés vers la preuve formelle et la physique mathématique. Il compare le prototype à quatre baselines, dont un « meilleur-de-5 filtré par le même vérificateur », complète la comparaison par deux ablations et juge les hypothèses H1 à H3 selon des critères de décision fixés avant toute exécution. Le cas d'étude phare, la complexité de Krylov, fournit la tranche N5 : des problèmes à réponse exacte, revérifiés avec SymPy. Sur 40 items, seuls des écarts d'environ 20 points sont détectables, et nous le disons d'emblée. **Aucun résultat n'existe encore : tous les chiffres de performance restent à mesurer.**

1. PROBLÈME ET MOTIVATION

Les LLM de pointe produisent des solutions convaincantes, mais **noter la seule réponse finale ne certifie aucune étape intermédiaire**. En physique, une recorrection experte de septembre 2026 a examiné 250 rejets dans six benchmarks. Elle les attribue pour 57,2 % à des erreurs du benchmark, pour 38,0 % à des erreurs du correcteur et pour seulement 4,8 % à des erreurs du modèle ; une fois les clés réparées, un modèle de pointe atteint 87 à 95 % sur quatre de ces benchmarks (arXiv:2609.13009). Sur des preuves de l'USAMO 2025, les modèles affirment avec assurance avoir résolu des problèmes dont le raisonnement est faux, et les correcteurs LLM surestiment les scores jusqu'à 20 fois (arXiv:2503.21934). Un score fondé sur la réponse finale sature donc et perd son pouvoir discriminant ; la vérification pas à pas est la mesure qui le conserve.

La vérification formelle déplace le problème sans le supprimer, car **le noyau de Lean ne certifie que l'énoncé formel, pas sa fidélité au problème posé**. Un audit de juin 2026 portant sur cinq benchmarks Lean relève 4 833 signalements, dont 398 défauts certifiés par une preuve, et mesure que les juges LLM détectent les défauts sémantiques avec une précision de 0,24 à 0,30 seulement (arXiv:2606.29493). ProofNet# a dû corriger 118 des 371 énoncés des portages Lean 4 de ProofNet, soit 31,8 % (arXiv:2406.07222). En autoformalisation, MathForm-8B produit 88,06 % d'énoncés qui compilent, mais seulement 72,37 % d'énoncés fidèles (arXiv:2608.14221). En physique formelle, TNLean désigne comme principal obstacle le fait de « faire respecter l'intention mathématique » (arXiv:2607.07857). AIM²P traite donc **la fidélité des énoncés comme son risque principal**. La promesse d'un « taux d'hallucination nul » cède la place à une chaîne d'audit explicite (§5.2) et à des taux d'échec mesurés.

2. QUESTION, SOUS-QUESTIONS ET HYPOTHÈSES PRÉ-ENREGISTRÉES

> **Question principale**
>
> « Dans quelle mesure une boucle neuro-symbolique ouverte, où un LLM raisonne, SymPy calcule et critique, et Lean 4 certifie, augmente-t-elle, à budget de calcul égal, la part de solutions correctes et intégralement vérifiées, et réduit-elle la part d'affirmations de succès non vérifiées, par rapport au même LLM utilisé seul ou filtré par le même vérificateur, sur des problèmes de mathématiques et de physique mathématique (cas d'étude : dynamique d'opérateurs en dimension finie et complexité de Krylov) ? »

- **SQ1 (interface et fidélité) :** comment passer d'un raisonnement informel à des énoncés et à des tactiques Lean 4 sans dérive d'énoncé ? Dans quels cas un résultat de SymPy peut-il devenir un certificat vérifiable par Lean, plutôt qu'un simple indice ?
- **SQ2 (boucle de rétroaction) :** comment les messages d'erreur de Lean 4 et les verdicts de SymPy, réinjectés dans le contexte du LLM, guident-ils une boucle de révision (agentique, éventuellement arborescente) vers une preuve certifiée, et à quel coût par itération ?
- **SQ3 (physique mathématique) :** comment formaliser les objets de la dynamique d'opérateurs en dimension finie (liouvillien, récursion de Lanczos, complexité de Krylov) ? Quelles erreurs physiques un vérificateur formel détecte-t-il mieux qu'un juge LLM ou que SymPy ?

> **Pré-enregistrement**
>
> Les définitions, seuils et critères de décision de cette section sont **gelés dans la présente version 0.1 (17 novembre 2026), avant toute exécution**. Toute modification ultérieure fera l'objet d'une version datée et justifiée, et les résultats seront aussi rapportés selon les définitions d'origine. Chaque hypothèse peut être *soutenue*, *réfutée* ou *non tranchée*, et les trois issues seront publiées de la même façon.

Définitions communes. La *réussite vérifiée* dépend du type d'item. Pour un item à réponse fermée, c'est une réponse équivalente à la clé auditée, à la fois selon SymPy et selon un test numérique. Pour un item formel, c'est une preuve acceptée par le noyau de Lean contre l'énoncé gelé, avec les seuls axiomes standard (`propext`, `Quot.sound`, `Classical.choice`). Une *itération* est un cycle complet « compilation Lean → message d'erreur (et verdict SymPy) → révision » ; le premier essai, qui précède tout retour, correspond à k = 0. Un *lemme* est un énoncé intermédiaire produit par une étape de décomposition exécutée une seule fois, avant la boucle (plan de preuve figé). Il est *certifié* s'il est accepté par le noyau, avec les seuls axiomes standard et sans modification de son énoncé.

H1 : gain de la boucle à budget égal. H1 porte sur AMP-Bench-mini v1.0, dont les 40 items N1–N5 et le correcteur sont gelés avant toute exécution. **H1a :** la boucle LLM + SymPy + Lean 4, limitée à 5 tours de révision (B3), atteint un taux de réussite vérifiée supérieur d'**au moins 20 points de pourcentage absolus** à celui du même LLM utilisé seul en un essai (B0). **H1b :** B3 dépasse aussi le même LLM lorsqu'il dispose de cinq essais indépendants filtrés par le même vérificateur (B2), à budget de tokens comparable. Le seuil de 20 points correspond au plus petit écart (environ 8 problèmes nets) qu'un test apparié peut détecter sur 40 items quand peu de problèmes basculent dans l'autre sens (§5.5). Le gain relatif en pourcentage n'est rapporté qu'à titre descriptif.

H2 : courbe de certification des lemmes. H2 porte sur les lemmes produits pour les items N3–N5, avec un LLM, un contexte et un budget figés : **au moins 70 % des lemmes sont certifiés en au plus 5 itérations**. On publie la courbe cumulée succès(k) pour k = 0 à 10, comparée à un rééchantillonnage indépendant de même coût et

ventilée par type d'erreur Lean : identifiant inconnu, incompatibilité de types, classes de types, délai dépassé. Pour cette mesure, la boucle est prolongée jusqu'à k = 10 ; B3 reste plafonnée à 5 tours pour H1. La littérature soutient la *forme* de H2, puisque 84 à 96 % du gain de réparation arrive dès le premier tour (arXiv:2610.26942), mais son *niveau* n'a aucun précédent. La version conditionnelle (« parmi les lemmes finalement certifiés, au moins 70 % le sont en 5 itérations au plus ») n'est qu'un résultat secondaire, car elle est presque certainement vraie et donc peu informative.

H3 : détection et correction d'erreurs physiques par injection contrôlée. Le corpus est formé de dérivations correctes de physique mathématique, tirées des items N5 et de la mécanique hamiltonienne. On y injecte des erreurs documentées : inversion de signe, terme de bord supprimé, condition aux limites erronée, terme non invariant, unités incohérentes, erreur d'indexation ou de normalisation. On compare ensuite trois critiques : un juge LLM sans accès aux artefacts du générateur, une vérification pas à pas par SymPy, et Lean 4.

- **H3a (typage) :** les erreurs dimensionnelles et d'invariance sont rejetées dès l'élaboration lorsque les grandeurs sont typées (`Units` de Physlib, formalisation de l'analyse dimensionnelle en Lean 4 arXiv:2509.13142, indices tensoriels).
- **H3b (obligations de preuve) :** les hypothèses manquantes, les cas limites ou dégénérés et les conditions aux limites erronées (au sens d'une frontière de domaine de paramètres comme au sens d'une EDP) apparaissent comme des buts non prouvables ou des contre-exemples formels, avec un taux de détection supérieur à celui des deux autres critiques.
- **H3c (correction, exploratoire) :** la boucle propose un énoncé ou une dérivation corrigés, acceptés par Lean et jugés fidèles par un relecteur humain.

Les énoncés formels de référence sont relus par un humain avant toute mesure. Le catalogue d'erreurs est préparé avec la tranche N5 (§6.3). La campagne H3 est prévue après la V0.1, sauf si le calendrier le permet plus tôt.

Hypothèse	Mesure	Soutenue si	Réfutée si
H1a	Différence appariée B3 − B0 du taux de réussite vérifiée	Borne inférieure de l'IC à 95 % > 0 **et** estimation ponctuelle ≥ 20 points	IC à 95 % entièrement sous 20 points
H1b	Différence appariée B3 − B2	Borne inférieure de l'IC à 95 % > 0	IC à 95 % entièrement sous 0
H2	Part des lemmes N3–N5 certifiés en ≤ 5 itérations (IC tenant compte de la corrélation des lemmes d'un même problème)	Borne inférieure de l'IC à 95 % ≥ 70 %	Borne supérieure de l'IC à 95 % < 70 %
H3a	Taux de rejet à l'élaboration des erreurs injectées dans des grandeurs typées	Toutes rejetées ; on rapporte aussi la part du catalogue exprimable en grandeurs typées	Une erreur typée injectée passe l'élaboration
H3b	Différence appariée des taux de détection, Lean − meilleur des deux autres critiques, sur les mêmes erreurs	Borne inférieure de l'IC > 0, avec un taux de faux positifs de Lean sur les dérivations correctes qui n'excède pas celui de ce critique	Borne supérieure de l'IC < 0
H3c	Corrections acceptées par Lean et jugées fidèles	Exploratoire : rapport descriptif, sans critère de décision	—

3. POSITIONNEMENT ET TRAVAUX LES PLUS PROCHES

Ce qu'AIM²P ne revendique pas. AIM²P ne revendique **ni la première physique formelle, ni la première combinaison LLM + calcul formel + Lean, ni la première boucle d'auto-correction guidée par Lean**. Le programme ne promet pas non plus un taux d'hallucination nul ni une vérification « à 100 % » au sens de la fidélité des énoncés. Il ne cherche pas davantage à concourir sur les benchmarks de concours, déjà saturés : plus de 99 % sur miniF2F, 88,8 % sur PutnamBench pour un système fondé sur un modèle à poids ouverts (arXiv:2606.06468). Ses comparaisons portent sur des systèmes construits sur **le même LLM**, à budget égal.

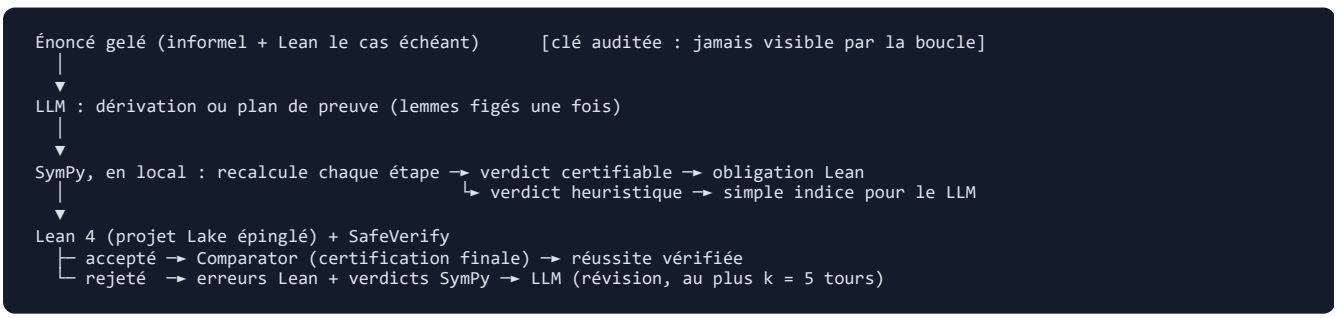
Quatre manques précis, à notre connaissance au 17 novembre 2026 :

1. **Une boucle ouverte et généraliste qui mesure ses propres critiques.** Aucune architecture ouverte et généraliste ne réunit encore quatre propriétés. (i) SymPy y critique systématiquement les étapes intermédiaires du LLM. (ii) Lean 4 y certifie les résultats finaux et, lorsqu'un certificat existe, les calculs symboliques eux-mêmes. (iii) La fiabilité du critique symbolique y est mesurée plutôt que présumée. (iv) Des ablations y mesurent la contribution de chaque composant.
2. **La physique mathématique à forte composante d'analyse.** Les évaluations Lean en physique sont récentes (134 à 1 019 tâches) et portent surtout sur l'information quantique et la physique de premier cycle. Une recherche directe sur les dépôts ne trouve aucun fichier sur « Lanczos » ou la croissance d'opérateurs dans Mathlib, et aucun sur « Krylov » ou « Lanczos » dans Physlib. Aucun pipeline publié ne couple calcul formel et réparation guidée par Lean pour des dérivations de physique mathématique.
3. **Des courbes de certification par lemme.** Aucun article ne publie la distribution du nombre de tours par lemme pour un système à décomposition, ni de courbe de réparation sur des énoncés de physique.
4. **Un foyer africain pour le raisonnement formellement vérifié.** Aucune plateforme ni aucun benchmark de raisonnement *formellement vérifié* n'est porté depuis l'Afrique, et aucun événement de la communauté Lean ne s'y est tenu (Lean community). Des benchmarks africains ouverts existent en revanche pour le raisonnement informel, comme AfriMGSM dans IrokoBench (arXiv:2406.03368) et Uhura.

Travail	Ce qu'il fait	Différence avec AIM²P
LIPS (ICLR 2025)	GPT-4o, tactiques de sommes de carrés fondées sur SymPy et Lean 4 : 76,3 % de 161 inégalités olympiques	Le plus proche par la pile logicielle ; un seul domaine, SymPy y sert de tactique et non de critique
MechGeo (2026)	Certificats Singular/SymPy vérifiés par le noyau de Lean, réparation par contre-exemples : 29 des 43 énoncés IMO de géométrie prouvés, les 14 autres réfutés	Le plus proche par le mécanisme ; géométrie euclidienne seulement
NSPI (ICML 2026)	D'après son résumé, le LLM devine une décomposition SOS approchée, le calcul symbolique la rend exacte et Lean la certifie	Inégalités polynomiales seulement
Moxia (2026)	Le LLM reformule l'énoncé, un pipeline déterministe de calcul formel résout ou s'abstient, Lean certifie les dérivations qui entrent dans un fragment de Mathlib ; 95,6 % sur MATH sans réponse fausse affirmée, selon l'auteur	Le plus proche parmi les architectures générales ; le calcul formel remplace le raisonnement du LLM au lieu de le critiquer ; pas de boucle d'auto-correction ni de physique

Travail	Ce qu'il fait	Différence avec AIM²P
SPL (2026)	Langage déclaratif qui compose appels au LLM et outils symboliques (SymPy, SageMath, Lean) ; 82 à 93 % de correction vérifiée selon les modèles	Couche d'orchestration générique ; d'après son résumé, l'évaluation porte sur la correction des réponses, pas sur une boucle de révision guidée par les erreurs
Aristotle (Harmonic, 2025)	LLM, exécution de code, moteur algébrique et Lean : niveau or à l'IMO 2025 en preuves formelles	Orienté mathématiques de concours
Ax-Prover	Agent Lean à outils : 96 % sur QuantumTheorems (134 problèmes de mécanique quantique) contre 40 % pour Claude Sonnet seul	Aucun outil de calcul formel
Lean4PHYS / LeanPhysBench	200 énoncés de physique : 35 % pour Claude-Sonnet-4 contre 16 % pour le meilleur prouveur mathématique spécialisé	Benchmark, sans boucle de calcul formel
AxProverBase	Agent minimal ouvert (proposeur ReAct, compilateur, relecteur d'énoncé, jusqu'à 50 itérations) : 54,7 % en pass@1 sur PutnamBench pour environ 12,6 $ par échantillon	Squelette réutilisable ; aucun critique symbolique
Goedel-Prover-V2	Auto-correction par messages d'erreur : avec 5 révisions, 92,7 % en pass@32 sur miniF2F, contre 92,2 % en pass@8192 sans correction	Référence pour H2 ; mathématiques de concours
Physlib (HepLean)	Environ 451 000 lignes sur 20 domaines au 17 novembre 2026, dont environ 245 000 dans PhyslibAlpha, à relecture allégée	Cible amont naturelle ; ni Krylov ni Lanczos
Erreur du 2HDM (Tooby-Smith, 2026)	Une formalisation montre qu'une condition de stabilité publiée en 2006 comme nécessaire et suffisante n'est pas suffisante	Précédent de H3 (détection), obtenu sans usage extensif de l'IA
lanzarini-krylov-lean	Complexité de Krylov pour un hamiltonien de Jacobi à quatre sites et une borne de vitesse	Seule formalisation Krylov trouvée ; mono-auteur, non relue, sans licence ouverte

4. MÉTHODE : LE PROTOTYPE V0.1



- **Rôles.** Le LLM lit l'énoncé gelé et propose soit une réponse fermée avec sa dérivation, soit un plan de preuve, fixé une fois, puis les preuves Lean de chaque lemme. SymPy recalcule et critique chaque étape, toujours en local : la tactique `polyrith` de Mathlib est en pratique hors service depuis la fermeture de son serveur Sage

externe (Zulip Lean), ce qui illustre le risque d'une dépendance distante. Lean 4 vérifie dans un projet Lake épinglé (§5.2).

- **Réinjection.** Les messages d'erreur de Lean et les verdicts de SymPy entrent dans le contexte du LLM, avec **au plus k = 5 tours de révision** et un arrêt au premier succès vérifié. La clé de correction n'est jamais visible par la boucle : SymPy y critique la cohérence interne des étapes, pas leur conformité à la clé.
- **Pas de MCTS en V0.1.** L'état de l'art est dominé par des boucles agentiques. Utiliser le compilateur comme simple récompense scalaire expose au piratage de récompense (arXiv:2608.28639). La recherche arborescente reste une option pour P6.
- **Réutiliser plutôt qu'écrire.** La boucle s'appuie sur `lean-lsp-mcp` pour l'accès à Lean, sur AxProverBase ou Numina-Lean-Agent comme squelette d'agent, sur Math-Verify pour les réponses fermées et sur SafeVerify et Comparator pour l'audit ; LeanSage peut servir d'oracle Sage local. Les licences de ces dépôts seront vérifiées en S2.

Le critique symbolique, un signal faillible que l'on mesure. Le calcul formel peut produire un résultat faux mais bien formé. Mathematica a ainsi calculé faux des déterminants de matrices entières (arXiv:1312.3270), et le traqueur de SymPy maintient une étiquette `simplify` qui recense des problèmes de correction ouverts. L'audit n'a trouvé **aucune étude empirique systématique du taux d'erreur de SymPy**. Chaque sortie SymPy est donc classée. Elle est *certifiable* quand elle porte un témoin que Lean peut vérifier : identité polynomiale, combinaison linéaire, somme de carrés, racine explicite, contre-exemple. Elle est *heuristique* sinon : elle informe alors le LLM sans jamais être citée comme preuve. Les pièges connus sont surveillés : symboles complexes par défaut, hypothèses `positive=True` ajoutées à tort, coupures de branche de `sqrt` et `log`, simplifications qui effacent une division par zéro, `solve` incomplet. Enfin, un échec de `simplify(a - b) == 0` ne prouve pas que a ≠ b. Un test différentiel, qui confronte SymPy à Sage ou Singular et à une évaluation numérique, est appliqué de manière ponctuelle. **La fiabilité du critique devient un résultat de recherche** : on mesure la part de ses verdicts contredits par Lean, par le numérique ou par le test différentiel, puis la part de ces erreurs que Lean rattrape.

Périmètre de la V0.1.

- **Dans la V0.1 :** une boucle linéaire d'au plus 5 tours, avec une décomposition en lemmes exécutée une fois. Elle utilise 2 ou 3 LLM, dont au moins un à poids ouverts, et SymPy en local comme critique, complété par un test différentiel ponctuel. Les énoncés formels sont écrits, relus et gelés par un humain. Le domaine de physique est la dynamique d'opérateurs en dimension finie, avec Lean 4 + Mathlib (+ Physlib).
- **Hors V0.1 (reporté) :** la recherche arborescente MCTS (P6) et la re-planification dynamique ; l'entraînement, le *fine-tuning* et l'apprentissage par renforcement ; Sage ou Singular comme critique principal ; l'autoformalisation des énoncés par le modèle. Sont aussi reportés les chaînes semi-infinies, les liouvilliens non bornés, SYK, la QFT et l'holographie (P7 et au-delà), ainsi que Rocq et Isabelle.

5. PROTOCOLE EXPÉRIMENTAL

5.1 AMP-Bench-mini v1.0

La répartition donne plus de poids aux niveaux où la baseline n'est pas au plafond. N1 et N2 sont proches de la saturation pour les modèles de pointe, et N3 l'est pour les prouveurs spécialisés (99,6 % sur miniF2F-test, arXiv:2507.23726). N5 porte la contribution scientifique. Cette pondération est un choix de conception, pas une pratique publiée.

Niveau	Items	Contenu	Source et licence	Vérification
N1	6	Calcul élémentaire, réponse fermée	Rédigés par AIM²P (variantes paramétriques à la manière de GSM1k), privés jusqu'à l'évaluation	Math-Verify (SymPy) + équivalence numérique en points aléatoires + audit manuel de chaque désaccord
N2	6	Calcul avancé et physique de premier cycle, réponse fermée	Idem	Idem, plus une seconde vérification experte de la clé
N3	8	Preuve formelle, niveau olympiade	google-deepmind/miniF2F (Apache-2.0) ou miniF2F-v2c (licence du jeu de données à confirmer), recoupés avec l'audit de 2026	Lean : SafeVerify dans la boucle, Comparator pour la certification finale
N4	8	Preuve formelle, premier cycle et Putnam	ProofNet# (MIT) et PutnamBench Lean (Apache-2.0)	Idem ; preuves PutnamBench non publiées
N5	12	Physique mathématique : Krylov, dynamique d'opérateurs, mécanique hamiltonienne, unités	Rédigés par AIM²P (§6), privés jusqu'à l'évaluation	SymPy exact + numérique ou Lean selon l'item ; clé dérivée indépendamment et relue par un second expert

Chaque item est stocké dans un fichier qui contient : id, niveau, énoncé informel, énoncé Lean le cas échéant, clé, mode de vérification, conventions (pour N5), source et licence. Les licences Apache-2.0 et MIT autorisent la redistribution avec attribution, à condition de conserver la licence et la notice. PutnamBench pose deux contraintes supplémentaires. **Ses mainteneurs demandent de ne pas publier de preuves formelles, pour limiter la contamination : AIM²P ne publiera donc aucune preuve PutnamBench produite par ses systèmes.** Par ailleurs, ses énoncés informels sont utilisés avec la permission de la MAA : AIM²P ne redistribue que les énoncés Lean et renvoie à la source. ProverBench, qui n'affiche pas de licence de jeu de données, est écarté. Chaque item repris est confronté aux signalements de l'audit de 2026 et relu à la main sur quatre points : énoncé contre source informelle, `autoImplicit false`, soustraction et division dans $\mathbb{N}$, hypothèses vacues. Les 40 énoncés sont portés dans **un seul projet Lake, avec `lean-toolchain` et commit Mathlib épinglés, en Lean ≥ 4.20.0**. ProofNet# n'est compatible qu'avec Lean 4.7.0 à 4.16.0-rc2 et devra donc être porté. Avec 6 à 12 items par niveau, les résultats par niveau restent descriptifs : un score de 4/8 a un intervalle de Wilson à 95 % de 21,5 % à 78,5 %.

5.2 Chaîne de vérification

1. **Énoncés gelés.** Le théorème prouvé est toujours comparé au fichier d'énoncés gelé, jamais à la reformulation du modèle.
2. **Version épinglée.** La version Lean est ≥ 4.20.0, car avant cette version un bogue de `apply?` faisait passer pour réussies des preuves sans déclaration vérifiée par le noyau. Le projet impose `autoImplicit false`. L'environnement de travail dispose de Lean 4.35.0 et de SymPy 1.14 ; les versions définitives seront fixées au gel.
3. **Axiomes et interdits.** Seuls `propext`, `Quot.sound` et `Classical.choice` sont admis, ce que contrôle `#print axioms` (SafeVerify). Sont exclus `sorry` et `admit`, ainsi que `native_decide`, qui depuis Lean 4.29 ajoute un axiome et fait confiance au compilateur (native-trust). Le sont aussi `debug.skipKernelTC`, les

déclarations `axiom` personnalisées et les métaprogrammes qui modifient l'environnement (arXiv:2609.25050).

4. **Vérification en deux temps.** SafeVerify tourne dans la boucle (médiane de 10,1 s par requête), Comparator une seule fois pour la certification finale (médiane de 95,7 s, avec une traîne au-delà de 10 minutes) (arXiv:2606.26442).
5. **Réponses fermées.** Math-Verify est utilisé avec une clé mise sous forme canonique, car sa fonction `verify` est asymétrique. On épingle son runtime ANTLR4 et on ajoute une équivalence numérique en points aléatoires. La méthode qui a tranché chaque item est consignée, et chaque désaccord est audité à la main.
6. **Humains et juges LLM.** Les juges LLM servent au tri, jamais à la notation. Tous les énoncés et toutes les clés N5 sont relus par un humain avant le gel.

5.3 Baselines et ablations

Code	Système	Rôle
B0	Même LLM seul, un essai	Baseline de H1a
B1	Même LLM, 5 échantillons indépendants, succès si l'un est correct (pass@5)	Indicatif seulement, car il suppose un oracle
B2	Même LLM, 5 échantillons indépendants, sélection par le même vérificateur SymPy/Lean	Contrôle de H1b : sépare la valeur de l'itération de celle de la vérification
B3	Boucle AIM²P complète, au plus 5 tours de révision	Système évalué
A1	Boucle sans SymPy (LLM + Lean)	Ablation, au moins sur N4–N5
A2	Boucle sans Lean dans la boucle (LLM + SymPy)	Idem ; la vérification finale reste la même pour tous les systèmes

B0, B1 et B2 se calculent sur les mêmes cinq échantillons indépendants. Chaque système tourne **au moins 3 fois par problème**, y compris à température nulle, car les API ne garantissent pas le déterminisme ; MathArena moyenne par exemple 4 essais par question (arXiv:2505.23281). Par LLM, cela représente 600 appels simples pour B0–B2 et au plus 360 exécutions agentiques pour B3, A1 et A2, ou 240 si les ablations se limitent aux 20 items N4–N5. Les tests portent sur deux ou trois LLM, dont **au moins un à poids ouverts**, conformément au principe de neutralité technologique ; Leanstral, par exemple, a publié ses premiers poids sous Apache-2.0 (Mistral). L'analyse principale de H1–H2 porte sur un LLM désigné au gel ; les autres sont rapportés à titre descriptif. Le budget réel de B2 et de B3 est comparé en tokens.

5.4 Métriques

Métrique	Définition	Usage
Taux de réussite vérifiée	Part des items en réussite vérifiée (§2), moyennée sur les répétitions	Primaire (H1)
Part certifiée par Lean	Part des items dont la solution passe Comparator avec la liste blanche d'axiomes	Le « Z % » de la règle d'or

Métrique	Définition	Usage
Itérations, courbe succès(k)	Tours jusqu'au succès ; courbe cumulée par lemme pour k = 0 à 10, comparée au rééchantillonnage, par type d'erreur	H2
Fausses déclarations de succès	Part des tentatives où le système affirme « résolu » ou « vérifié » alors que le vérificateur rejette (H-d)	Définition opérationnelle de l'hallucination
Fiabilité du critique SymPy	Part des verdicts SymPy contredits par Lean, par le numérique ou par le test différentiel	Résultat de recherche (§4)
Coût	Tokens d'entrée et de sortie, temps mur, coût par problème ; succès en fonction des tokens consommés, à la manière du pass@t d'Olausson et al.	Comparaison à budget égal

Taxonomie d'échecs. Chaque tentative est classée selon une taxonomie synthétisée à partir de l'audit de 2026, de « Proof or Bluff? » et de MA-ProofBench (arXiv:2606.13782). Elle compte cinq catégories. **H-a** est une preuve non valide (`sorry`, axiome hors liste blanche, `native_decide`, métaprogramme), détectée par `#print axioms`, SafeVerify et Comparator. **H-b** est une dérive d'énoncé, détectée par comparaison avec le fichier gelé. **H-c** regroupe les identifiants inventés (erreurs « unknown identifier/constant »), comptés par tour. **H-d** est une fausse déclaration de succès, démentie par le verdict ; c'est la définition la plus nette de l'hallucination. **H-e** est une réponse fermée fausse, jugée par SymPy et l'équivalence numérique contre la clé auditée.

5.5 Statistiques

Pour chaque item et chaque système, le résultat est la proportion d'essais réussis parmi les R ≥ 3 répétitions. Deux systèmes se comparent par **différences appariées, item par item** (arXiv:2411.00640). Les proportions reçoivent des intervalles de Wilson, et les différences appariées des intervalles exacts ou bayésiens (bibliothèque `bayes_evals`). Aucun intervalle n'est fondé sur le théorème central limite, qui sous-estime l'incertitude en dessous de quelques centaines d'items (arXiv:2503.01747). En analyse de sensibilité, chaque item reçoit une issue binaire (réussite pour au moins 2 essais sur 3), et les items discordants sont soumis à un test de McNemar exact. Les items publics (N3–N4) sont rapportés séparément des items privés (N1, N2, N5).

Pourquoi « +40 % » relatif est indétectable sur 40 items. Autour d'une exactitude de 50 %, l'intervalle de Wilson va de 35,2 % à 64,8 %, soit environ ±15 points ; autour de 30 %, il va de 18,1 % à 45,4 %. Un gain relatif de 40 % sur une baseline à 30 % mène à 42 %, soit environ 5 problèmes de plus. Or, avec un test du signe exact, même 5 problèmes gagnés contre 0 perdu donnent p = 0,0625. Pour conclure à 5 %, il faut 6 contre 0 (p = 0,031), 8 contre 1 (p = 0,039) ou 10 contre 2 (p = 0,039). D'où le seuil absolu de 20 points retenu pour H1a.

5.6 Reproductibilité et gel horodaté

Avant toute exécution, on gèle les 40 items, le correcteur, les identifiants exacts et les dates des modèles, les paramètres de décodage, les prompts système et les graines. On gèle aussi les versions de SymPy, de Math-Verify, de la chaîne Lean, de Mathlib et du REPL. Le choix du correcteur peut à lui seul déplacer un score d'environ 18 points, d'après un billet de blog qui n'a pas été revérifié sur une source primaire.

Le gel se fait en deux temps. Le premier fige le benchmark et le correcteur en fin de S5, puis la configuration des baselines en S8. Le second fige la configuration et les prompts de la boucle avant les exécutions de B3, A1 et A2 (S12). La boucle est mise au point sur des items de développement (réserve Krylov et items analogues), jamais sur les 40 items gelés. Chaque gel est horodaté publiquement par un tag du dépôt. Les items privés y figurent par leurs empreintes SHA-256 ; leurs fichiers sont publiés après l'évaluation, à l'exception des preuves

PutnamBench. Une sonde de mémorisation teste, sur les items publics, si le modèle reproduit un énoncé à partir d'un préfixe. Enfin, une seule commande, `python -m engine.run --bench bench/ --model <id>`, reproduit le tableau final, et aucun chiffre n'est publié sans son script.

6. CAS D'ÉTUDE KRYLOV (TRANCHE N5)

6.1 Contexte et conventions

La complexité de Krylov est la position moyenne $K(t) = \Sigma_n\, n{\cdot}\varphi_n(t)^2$ d'un opérateur en évolution sur la « chaîne de Krylov », dont les amplitudes de saut sont les coefficients de Lanczos b_n. La fonction d'autocorrélation C(t), les moments μ_{2k}, les b_n et la fonction spectrale portent la même information. Selon l'hypothèse de croissance universelle de Parker et al., dans les systèmes chaotiques génériques de dimension $d > 1$, les b_n croissent linéairement ($b_n \approx \alpha n + \gamma$), avec une correction logarithmique en dimension 1. La borne $\lambda_L \leq 2\alpha$ sur l'exposant de Lyapunov n'est démontrée qu'à température infinie (arXiv:1812.08657 ; revue de Nandy et al.). **La croissance exponentielle de K(t) n'est pas un diagnostic de chaos.** On la trouve dans toutes les théories conformes testées, y compris libres (arXiv:2104.09514), et dans le modèle intégrable LMG (arXiv:2203.03534). Aucune formalisation relue de la théorie des coefficients de Lanczos (en Lean, Coq ou Isabelle) et aucun benchmark de raisonnement LLM sur Krylov n'ont été trouvés.

Les conventions sont celles de Caputa–Magán–Patramanis (arXiv:2109.03824) : $b_0 = 0$, $\varphi_n(0) = \delta_{n0}$, $\partial_t\varphi_n = b_n\varphi_{n-1} - b_{n+1}\varphi_{n+1}$, $O(t) = \Sigma\, i^n\, \varphi_n\, O_n$, $C(t) = \varphi_0(t)$, et produit scalaire normalisé $\mathrm{Tr}(A^\dagger B)/\mathrm{Tr}\,\mathbb{1}$. Dans ces conventions, les φ_n sont réels. Chaque item embarque ses conventions, car les réponses en dépendent.

6.2 Problèmes candidats

La colonne « Vérification » indique le contrôle effectué lors de l'audit (SymPy 1.14, mpmath) et la faisabilité Lean proposée. Les identifiants U1 et C1 sont attribués ici.

Id	Énoncé	Réponse exacte	Vérification	Statut
B1	Moments gaussiens $\mu_{2k} = (2k-1)!!\ \alpha^{2k}$; trouver $b_1 \ldots b_5$	$b_n = \alpha\sqrt{n}$; $C(t) = e^{(-\alpha^2 t^2/2)}$	SymPy ; Lean : `ring` pour $n \leq 2$, déterminants explicites au-delà	Réserve
B2	$\mu_{2k} = 1, 1, 5, 61, 1385, 50521, 2702765$ (nombres d'Euler en valeur absolue, C = sech t) ; trouver $b_1 \ldots b_6$	$b_n = n$; variante sech(3t) : $b_n = 3n$	SymPy ; Lean faisable (`norm_num`)	N5
B3	$C(t) = \cos^3(\alpha t)$; trouver tous les b_n et la dimension de Krylov	$b_1^2 = 3\alpha^2$, $b_2^2 = 4\alpha^2$, $b_3^2 = 3\alpha^2$, $b_4 = 0$; dimension $4 = 2j + 1$ ($j = 3/2$)	SymPy ; Lean faisable (déterminant de Hankel nul)	N5
B4	Montrer que $\varphi_n = \sqrt{((\eta)_n/n!)}\ \tanh^n(\alpha t)\ \mathrm{sech}^\eta(\alpha t)$ résout la chaîne avec $b_n = \alpha\sqrt{(n(n-1+\eta))}$	Se réduit à deux identités de Pochhammer	SymPy à n général ; Lean faisable (`ascPochhammer`)	N5
B5	Montrer $\Sigma_n\, C(2j,n)\, \sin^{2n}x\, \cos^{(2(2j-n))}x = 1$ et $\Sigma_n\, n\, C(2j,n)\, \sin^{2n}x\, \cos^{(2(2j-n))}x = 2j\, \sin^2 x$	$K(t) = 2j\, \sin^2(\alpha t)$, de maximum $2j$	SymPy pour $2j = 1, 2, 3, 5$; Lean faisable (`add_pow`)	N5

Id	Énoncé	Réponse exacte	Vérification	Statut
B6	$K(t) = k_2 t^2 + k_4 t^4 + O(t^6)$; trouver b_1^2 et b_2^2	$b_1^2 = k_2$; $b_2^2 = 2k_2 + 6k_4/k_2$	SymPy ; Lean faisable (identité algébrique)	N5
B7	Pour $b_n = \alpha\sqrt{(n(n-1+\eta))}$, trouver l'asymptotique et l'exposant	$b_n - \alpha n \to \alpha(\eta-1)/2$; $\lambda_K = 2\alpha$	SymPy ; Lean moyen (`Filter.Tendsto`)	Réserve
B8	$a_n = \gamma(h+n)$, $b_n = \alpha\sqrt{(n(2h+n-1))}$: complexité d'étalement	$K_S(t) = [2h\alpha^2/(\alpha^2-\gamma^2/4)] \sinh^2(\sqrt{(\alpha^2-\gamma^2/4)}\, t)$: exponentielle si $\gamma < 2\alpha$, égale à $2h\alpha^2 t^2$ si $\gamma = 2\alpha$, oscillante si $\gamma > 2\alpha$	Accord numérique à 10 chiffres ; source bibliographique à confirmer ; Lean difficile	N5
B9	Calculer $b_1 \ldots b_{40}$ à partir des moments de sech (item piège)	$b_n = n$ exactement ; un calcul en float64 donne $b_{40} \approx 39{,}954$	mpmath à 80 chiffres ; sans objet en Lean	Réserve (robustesse numérique)
B10	$b_1 = \alpha$, $b_2 = 0$ (spin ½)	$\varphi_0 = \cos \alpha t$, $\varphi_1 = \sin \alpha t$; $K = \sin^2(\alpha t)$	SymPy ; Lean faisable (`HasDerivAt`)	N5
F1	Pour H hermitienne, O(t) = e^(itH) O e^(−itH) vérifie dO/dt = i[H, O(t)] et conserve la norme de Frobenius	— (preuve)	Lean (`Matrix.exp`)	N5
F2	$\mathcal{K}_k$ = vect{v, Lv, …, L^(k−1)v} est de dimension ≤ N, devient L-invariant dès qu'il se stabilise, et Lanczos termine en ≤ N pas ($N = D^2$)	— (preuve)	Lean ; recherche Loogle préalable	N5
F3	Pour un liouvillien hermitien, la base de Gram–Schmidt de la suite de Krylov est tridiagonale, et $a_n = 0$ pour un opérateur initial hermitien	— (preuve)	Lean (`gramSchmidt`)	Réserve (livrable amont PhyslibAlpha)
F4	$\Sigma_n \varphi_n(t)^2 = 1$, $K(0) = 0$ et $0 \le K(t) \le \dim \mathcal{K} - 1$	— (preuve)	Lean, sur la base de F1–F3	Réserve (idem)
F5	Pour H indépendant du temps, l'énergie est conservée le long des équations de Hamilton et {H, H} = 0 ; variantes à erreur injectée	— (preuve)	Lean, réutilise Physlib	N5
U1	Cohérence dimensionnelle de la période du pendule	— (énoncé typé)	Lean, `Units` / `WithDim` de Physlib (test direct de H3a)	N5
C1	$[x_i, p_j] = i\hbar\delta_{ij}$, déjà présent dans Physlib	— (étalonnage)	Lean	N5

> **Énoncés formels : des propositions, pas encore du code compilé**
>
> Les faisabilités Lean et les noms de lemmes Mathlib cités (`ascPochhammer`, `add_pow`, `HasDerivAt`, `Filter.Tendsto`, `Matrix.exp`, `gramSchmidt` …) sont des **propositions non compilées**, à confirmer sur la version épinglée. Leur difficulté est estimée, pas mesurée. Avant toute revendication de nouveauté, une recherche Loogle doit vérifier que F1 et F2 n'existent pas déjà sous d'autres noms. Tout item dont l'énoncé Lean ne compile pas avant le gel est remplacé par un item de réserve.

6.3 Pièges à contrôler

Ces pièges servent de liste de contrôle pour les correcteurs et de catalogue d'erreurs à injecter pour H3 :

- décalage d'indice (on contrôle que $b_1^2 = \mu_2 = -C''(0)$) ;
- normalisation $\mu_0 \neq 1$;
- confusion entre η et $2h$;
- $\lambda_K = \alpha$ au lieu de 2α ;
- oubli de la terminaison $b_{2j+1} = 0$ en SU(2) ;
- croissance $e^{(2\alpha t)}$ affirmée en dimension 1, où $K \sim e^{(\sqrt{(4\alpha t)})}$;
- changement de produit scalaire à température finie.

Les choix $e^{(+i\mathscr{L}t)}$ ou $e^{(-i\mathscr{L}t)}$, i^n ou $(-i)^n$, changent les phases des φ_n mais pas leurs modules, ni K, ni les b_n : les correcteurs comparent donc des quantités invariantes. Côté numérique, le passage des moments aux b_n est exponentiellement mal conditionné. Pour sech t, la récursion en float64 donne une erreur relative de $1{,}8 \cdot 10^{-10}$ à n = 20 mais de $1{,}2 \cdot 10^{-3}$ à n = 40, alors qu'en mpmath à 80 chiffres l'erreur tombe à 10^{-70}. Sans réorthogonalisation, un Lanczos float64 en dimension 400 dévie de plus de 10^{-8} dès n = 98. Les clés sont donc calculées en arithmétique exacte ou en haute précision.

7. CALENDRIER

Le calendrier suppose environ 10 heures de travail par semaine, soit à peu près 130 heures sur 90 jours.

Sem.	Dates	Phase	Livrable
S1	17–23 nov.	P0	Cette proposition, recherche arXiv finale, site en ligne, tag `v0.1-phase0`, 1 ou 2 demandes de financement ou de crédits
S2	24–30 nov.	P1	`docs/05_Domain_Map.md` (LLM, Lean 4, SymPy, Rocq, Isabelle) ; licences des composants réutilisés ; choix de la version Lean et Mathlib
S3	1–7 déc.	P2	`bench/` : schéma des items, `bench/validate.py`, projet Lake épinglé ; 16 items N3–N4 sélectionnés, audités, portés
S4	8–14 déc.	P2	12 items N5 : clés indépendantes, énoncés Lean compilés, recherche Loogle ; catalogue d'erreurs pour H3
S5	15–21 déc.	P2	12 items N1–N2 ; `bench/PROTOCOL.md` ; **gel horodaté d'AMP-Bench-mini v1.0** et du correcteur

Sem.	Dates	Phase	Livrable
S6–S7	22 déc.–4 janv.	*tampon*	Fêtes : rattrapage ou repos
S8	5–11 janv.	P3	`harness/` ; choix des 2 ou 3 LLM ; pilote de coût sur 5 items de développement ; `results/baselines/config.yaml` gelé
S9	12–18 janv.	P3	Exécutions B0–B2 (3 répétitions)
S10	19–25 janv.	P3	`docs/06_Baseline_Report.md` : premiers chiffres réels, taxonomie des échecs, coûts
S11	26 janv.–1er févr.	P4	`engine/` V0.1, mise au point sur les items de développement
S12	2–8 févr.	P4	Gel de la boucle ; exécutions B3, A1, A2
S13	9–15 févr.	P4 (+ début de P5)	Analyse de H1 et H2, courbe succès(k), release `v0.2` , mise à jour du site avec les seuls chiffres mesurés

> **Règle anti-dérive**
>
> « Si une tâche dépasse le double de son estimation, on réduit le périmètre, pas le calendrier. » Les réductions possibles sont déclarées d'avance, dans cet ordre : ablations A1 et A2 limitées aux 20 items N4–N5 ; deux LLM au lieu de trois, avec B3 sur le seul LLM principal ; remplacement d'un item N5 non compilable par un item de réserve ; report de la campagne H3. Toute réduction est annoncée avant les exécutions concernées. Ne sont jamais réduits : le gel avant exécution, la chaîne de vérification, les 3 répétitions et la publication des résultats négatifs.

8. LIVRABLES

Livrable	Contenu	Échéance
`docs/04_Research_Proposal_v0.1.md`	Ce document (le PDF est généré automatiquement)	S1
`docs/05_Domain_Map.md`	Matrice d'une page justifiant les choix d'outils, licences des composants réutilisés	S2
`bench/`	40 fichiers d'items, projet Lake épinglé, `validate.py` , `PROTOCOL.md` , catalogue d'erreurs H3 ; items privés publiés après l'évaluation	S3–S5
`harness/`	Soumission aux modèles, chaîne de vérification (§5.2), journalisation des échecs H-a à H-e	S8
`results/`	`baselines/config.yaml` , journaux bruts (sans preuves PutnamBench), tableaux, coûts	S9–S13
`docs/06_Baseline_Report.md`	Premiers chiffres B0–B2, taxonomie des échecs, coûts	S10

Livrable	Contenu	Échéance
`engine/`	Boucle V0.1, reproductible en une commande	S11–S13
Tags et releases	`v0.1-phase0` (S1), tag de gel d'AMP-Bench-mini v1.0 (S5), `v0.2` (S13) ; DOI Zenodo en option	—

9. RISQUES ET PARADES

Risque	Parade
Coût ou accès aux API	Pilote de coût avant tout lancement ; au moins un modèle à poids ouverts ; demandes de crédits dès S1 ; réductions de périmètre pré-déclarées (§7)
Infidélité des énoncés, défauts du benchmark	Énoncés gelés, recoupement avec l'audit de 2026, `autoImplicit false` , relecture humaine, clés N5 dérivées indépendamment. Un défaut découvert après le gel est publié en erratum ; l'item est exclu dans une analyse de sensibilité, jamais corrigé en silence
Pas de second expert disponible pour N5	Recherche d'un relecteur (AIMS, communauté Lean) ; à défaut, double dérivation (SymPy exact et numérique), en signalant explicitement l'absence de seconde relecture
Évolutions de Lean et Mathlib	Un seul projet Lake épinglé, Lean ≥ 4.20.0, aucune mise à jour pendant une campagne ; compatibilité de SafeVerify et de Comparator vérifiée au gel
Calcul formel faux mais bien formé	Distinction certifiable/heuristique, tests numérique et différentiel, SymPy jamais cité comme preuve, fiabilité mesurée (§4)
Preuves invalides acceptées	Liste blanche d'axiomes, interdits (§5.2), certification finale par Comparator
Contamination	Items N1, N2 et N5 privés jusqu'à l'évaluation ; N3–N4 rapportés à part ; sonde de mémorisation ; preuves PutnamBench non publiées
Temps d'un fondateur seul (10 h par semaine)	Un livrable minimal par bloc, réutilisation de composants ouverts, tampon S6–S7, règle anti-dérive
Antériorité	Positionnement étroit (§3), recherche arXiv finale avant diffusion, recherche Loogle pour F1–F2
Résultat négatif	**Publié tel quel.** Si B3 ne bat pas B2, le gain vient de la vérification et de l'échantillonnage, pas de l'itération : c'est un résultat légitime, et le pré-enregistrement le rend informatif

10. BUDGET ET FINANCEMENT

Poste	Besoin pour les 90 jours	Estimation
Appels LLM	Par LLM : 600 appels simples (B0–B2) et 240 à 360 exécutions agentiques (B3, A1, A2), plus un pilote de 5 items	**À estimer sur le pilote avant tout lancement.** L'audit avance un ordre de grandeur de quelques centaines à quelques milliers de dollars pour l'ensemble. Cette estimation s'appuie sur des coûts publiés pour des agents bien plus longs que 5 tours, de l'ordre de 4 $ (chiffre de source secondaire) à 12,6 $ (AxProverBase) par problème Putnam

Poste	Besoin pour les 90 jours	Estimation
Vérification Lean et SymPy	Calcul CPU local ; Comparator, de l'ordre de 1,5 minute médiane par item	Pas de coût direct au-delà de la machine existante
Hébergement	Site (GitHub Pages), DOI (Zenodo), données (Hugging Face)	Gratuit
Temps	Environ 130 heures	Bénévole
Seconde relecture experte	Clés et énoncés N5	Non budgétée : à obtenir par collaboration

Priorité	Source	Sans affiliation ?	Apport	Statut au 17/11/2026
1	AI for Math Fund, *seed grants* (Renaissance Philanthropy × XTX Markets)	Oui, « individus et équipes du monde entier »	Jusqu'à 100 000 $; catégories « jeux de données et benchmarks » et « outils et infrastructure » ; diffusion ouverte exigée	Examen au fil de l'eau d'après la page officielle ; échéance à confirmer auprès d'aiformath@renphil.org
2	Hugging Face ZeroGPU + *community GPU grant*	Oui	2 Spaces ZeroGPU gratuits et un quota quotidien de GPU (de l'ordre de 5 minutes), pour une démonstration publique	Ouvert
3	Google TPU Research Cloud	Historiquement oui ; conditions actuelles à revérifier	TPU gratuits, utiles seulement pour du *fine-tuning*	Sur candidature
4	Anthropic AI for Science, OpenAI Researcher Access, Google Cloud	Non : affiliation requise	Respectivement jusqu'à 20 000 $ de crédits API sur 6 mois, jusqu'à 1 000 $ et de 1 000 à 5 000 $	Ouverts, mais conditionnés à une affiliation

Le *seed grant* de l'AI for Math Fund est la seule source d'argent réellement ouverte à un fondateur sans affiliation. Une demande de 15 000 à 50 000 $ paraît proportionnée à un pilote ; c'est une suggestion de l'audit, pas une consigne du fonds, et elle reste à valider. Le grand tour 2026 du fonds est clos depuis le 30 mars 2026, et le tour 2027 n'est pas annoncé.

C'est l'affiliation qui débloque les crédits d'API. Le partenaire le plus naturel est AIMS : chaque étudiant du master AI for Science d'AIMS South Africa mène un projet de recherche, et AIM²P peut proposer un ou deux sujets bien bornés, comme formaliser un lemme de Krylov ou construire une tranche vérifiée du benchmark. AIMS RIC (Kigali) organise déjà des ateliers « AI in Science » avec Imperial College. **Aucun partenariat n'existe à ce jour : c'est une démarche à engager.** Plusieurs pistes du document directeur sont fermées : le tour « Claude Science » (clos le 15 juillet 2026), Microsoft AFMR (terminé) et Modal for Academics (signalé en pause par une source secondaire).

11. RÈGLE DE COMMUNICATION

> **Règle d'or**
>
> Ne jamais écrire « Notre IA est intelligente » ou « Notre IA résout les mathématiques ».
>
> Toujours écrire : « Notre système obtient X % de réussite sur le benchmark AMP-Bench contre Y % pour la baseline, avec Z % de solutions intégralement vérifiées par Lean 4. »

Pour cette proposition, la règle d'or se traduit en quatre consignes :

- aucun chiffre de performance n'est publié, sur le site comme ailleurs, avant d'avoir été mesuré ; chaque chiffre est accompagné de son script, de sa configuration, de sa graine et de ses versions ;
- « vérifié » signifie accepté par le noyau contre l'énoncé gelé, avec les seuls axiomes de la liste blanche ; la fidélité de l'énoncé relève de la relecture humaine et est présentée comme telle ;
- chaque score indique la classe de système, le budget (essais, tours, tokens) et la baseline à budget égal ; sur N3, on rappelle que les prouveurs spécialisés atteignent environ 99,6 % sur miniF2F-test ;
- les résultats négatifs ou non tranchés sont publiés avec la même visibilité que les autres.

RÉFÉRENCES

- **Source principale :** audit bibliographique de clôture de la phase P0 (17 novembre 2026), dont la substance est publiée dans les documents 02 État de l'art et 03 Verrous scientifiques, version 0.2.
- **Systèmes LLM + calcul formel + Lean :** LIPS (arXiv:2502.13834) ; MechGeo (arXiv:2608.02295) ; NSPI (arXiv:2605.15445) ; Aristotle (arXiv:2510.01346) ; architectures plus générales : Moxia (arXiv:2606.00671), SPL (arXiv:2607.07727) ; calcul formel et Lean sans LLM (arXiv:2604.13514 ; arXiv:2101.07758).
- **Boucles de réparation guidées par Lean :** DeepSeek-Prover-V1.5 (arXiv:2408.08152) ; Goedel-Prover-V2 (arXiv:2508.03613) ; OProver (arXiv:2605.17283) ; Fix, Not Just Resample (arXiv:2610.26942) ; AxProverBase (arXiv:2602.24273) ; Goedel-Architect (arXiv:2606.06468) ; Reward-Oracle MCTS (arXiv:2608.28639).
- **Physique formelle :** HepLean / Physlib (arXiv:2405.08863) ; Ax-Prover (arXiv:2510.12787) ; Lean4PHYS (arXiv:2510.26094) ; erreur du 2HDM (arXiv:2603.08139) ; TNLean (arXiv:2607.07857) ; SYK à haute température en Lean (arXiv:2610.18440) ; Physics as Code SU(5) (arXiv:2603.28406).
- **Évaluation et validité :** Faults in Our Formal Benchmarking (arXiv:2606.29493) ; ProofNet# (arXiv:2406.07222) ; MathForm (arXiv:2608.14221) ; Proof or Bluff? (arXiv:2503.21934) ; MA-ProofBench (arXiv:2606.13782) ; recorrection experte en physique (arXiv:2609.13009) ; AXLE (arXiv:2606.26442) ; FrontierMath Erdős (arXiv:2609.25050).
- **Statistiques :** Olausson et al. (arXiv:2306.09896) ; Bowyer et al. (arXiv:2503.01747) ; Miller (arXiv:2411.00640) ; MathArena (arXiv:2505.23281).
- **Outils et données :** SafeVerify ; native-trust ; Math-Verify ; bayes_evals ; google-deepmind/miniF2F ; ProofNet# ; PutnamBench ; Mathlib4.
- **Krylov :** Parker et al. (arXiv:1812.08657) ; Caputa–Magán–Patramanis (arXiv:2109.03824) ; Nandy et al. (arXiv:2405.09628) ; croissance exponentielle en CFT (arXiv:2104.09514) et dans le modèle LMG (arXiv:2203.03534) ; lanzarini-krylov-lean.
- **Financement et écosystème :** AI for Math Fund ; Hugging Face ZeroGPU ; TPU Research Cloud ; AIMS AI for Science ; Imperial–AIMS ; IrokoBench (arXiv:2406.03368).

HISTORIQUE DES VERSIONS

- **v0.1 (17 novembre 2026) :** première version, rédigée à partir du rapport de clôture de P0. Elle pré-enregistre les hypothèses H1 à H3 et leurs critères de décision, ainsi que le protocole d'AMP-Bench-mini v1.0 et le calendrier des 90 jours. Intègre la recherche arXiv de vérification finale du 17 novembre 2026 (notamment Moxia et SPL).