

Compléments à la section 5 de la note au sujet de la preuve dite primegaps d’une société d’ia : les pièges de relecture d’un énoncé en lean 4 (avec import de la librairie mathématique mathlib)

ia claude pour Denise Vella-Chemla, 19 octobre 2026

Le paragraphe 5 de la note ici

[1] <https://denisevellachemla.eu/verif-depot-oai-primegaps-claude-dvc-20261018.pdf> , dans le cadre de la conversation qui semble s’être engagée entre mathématiciens et informaticiens sur la possibilité de faire démontrer informatiquement des théorèmes mathématiques en utilisant à la fois des llm (en l’occurrence le dépôt simultané le plus important de preuves a été fait par la société openai et donc, le llm a dû être une version de chatgpt ou bien similaire à ce llm) et des assistants de preuve (dans le dépôt en question, l’assistant de preuve utilisé est lean).

Mon grain de sel est d’avoir fait générer par l’ia claude la liste de points de vigilance ci-dessous.

1. Réponse courte

Les trois points de la section 5 de la note [1] ci-dessus (soustraction tronquée, division par zéro, borne supérieure) ne sont pas les seuls. Ils appartiennent à une même famille : en théorie des types, toute fonction doit être *totale*, donc lean (et mathlib) fixe une valeur par convention là où la mathématique usuelle dit “non défini”. Une preuve peut alors être parfaitement acceptée par le noyau tout en parlant d’une valeur-poubelle.

Nombre approximatif. Il n’existe pas de recensement officiel que l’ia claude pourrait citer. Dans la suite de ce texte, le pronom personnel “je” doit être associé à l’ia claude qui a intégralement rédigé la note¹. D’après ma connaissance de mathlib, je compte de l’ordre de **cinquante** conventions de ce genre (davantage si l’on sépare les variantes), dont une **trentaine** rencontrées couramment dans des énoncés de théorie des nombres et d’analyse. Les 31 premières sont listées ci-dessous. C’est une estimation de ma part, à compléter par la lecture de la documentation de mathlib pour chaque notion employée.

La dernière colonne indique ce que l’on peut dire pour l’énoncé de PrimeGaps, à partir du texte de la note uniquement.

n°	Convention	Effet à surveiller	Dans PrimeGaps
1	Soustraction dans $\mathbb{N}$	Tronquée : $3 - 5 = 0$. Le transtypage de $\mathbf{a} - \mathbf{b}$ n’est pas la différence des transtypages.	Évité (conversion en réels avant la soustraction). $\mathbf{n} - 1$ dans prime sans effet pour $n \geq 1$.
2	Division par zéro dans $\mathbb{R}$ (et tout corps)	$\mathbf{a} / 0 = 0$ et $0^{-1} = 0$.	Écarté : on peut toujours prendre $N_0 \geq 1$.
3	Division dans $\mathbb{N}$	Arrondie par défaut : $7 / 2 = 3$, et $\mathbf{n} / 0 = 0$.	Non utilisée (quotients faits dans $\mathbb{R}$).

1. sauf la partie bleue introductive.

n°	Convention	Effet à surveiller	Dans PrimeGaps
4	Reste modulo 0	$n \% 0 = n$.	Non utilisé.
5	Division dans $\mathbb{Z}$	Euclidienne : le reste est toujours ≥ 0 , $-7 / 2 = -4$.	Non utilisée.
6	sSup et sInf sur $\mathbb{R}$	Ensemble vide ou non borné : valeur 0.	Contrôlé (ensemble non vide, majoré par 1).
7	Nat.nth	Commence à l'indice 0 ; si l'ensemble est fini et l'indice trop grand, valeur 0. Ici $\text{prime } 0 = \text{prime } 1 = 2$, à cause de $0 - 1 = 0$.	Ensemble infini ; indices restreints à $n \geq 1$ dans les deux parties de l'énoncé.
8	Indexation et intervalles	$\text{Finset.range } n = \{0, \dots, n - 1\}$; $\text{Icc}, \text{Ico}, \text{Ioc}$; $\text{Icc } 1 \ 0$ est vide.	$\text{Icc } 1 \ N$: cohérent avec la lecture " $n \leq N$ ".
9	Placement des coercions	$((n+1 : \mathbb{N}) : \mathbb{R})$ diffère de $(n : \mathbb{R}) + 1$ seulement en apparence, mais le transtypage d'une division ou d'une soustraction naturelle perd l'information.	Cohérent dans le fichier de défi.
10	Real.log	$\log 0 = 0$, $\log (-x) = \log x$.	Arguments : des nombres premiers, donc ≥ 2 .
11	Real.sqrt	Racine d'un réel négatif = 0.	Non utilisée.
12	Puissances x^y	Un exposant numéral comme $1/2$ peut être interprété dans $\mathbb{N}$ (donc $= 0$, d'où $x^0 = 1$) ; il faut écrire $(1/2 : \mathbb{R})$. Pour les réels : $0^0 = 1$, base négative traitée par $\exp(y \log x) \cos(\pi y)$.	Non utilisées.
13	Branches complexes	Complex.log , Complex.cpow : branche principale ; $0^0 = 1$.	Non utilisées.
14	Trigonométrie réciproque, tangente	Real.arcsin et Real.arccos écrêtent hors de $[-1, 1]$; Real.tan en $\pi/2$ vaut 0.	Non utilisées.
15	Fonctions arithmétiques en 0 ou hors domaine	$\text{Nat.divisors } 0 = \emptyset$ $\text{Nat.totient } 0 = 0$, $\text{Nat.factorization } 0 = 0$, $\text{padicValNat } p \ 0 = 0$, $\text{Nat.gcd } 0 \ 0 = 0$, $\text{Nat.choose } n \ k = 0$ si $k > n$	Non utilisées dans l'énoncé.
16	Nat.log, Nat.sqrt, Real.logb	Parties entières ; $\text{Nat.log } b \ n = 0$ si $b \leq 1$; Real.logb de base 1 ou 0 : valeur-poubelle.	Non utilisés.
17	Parties entières	$\text{Nat.floor } x = 0$ pour $x < 0$; ne pas confondre Nat.floor , Int.floor , Nat.ceil .	Non utilisées.
18	Cardinaux	Nat.card et Set.ncard d'un ensemble infini valent 0 ; Set.encard vit dans $\mathbb{N}_\infty$.	On emploie Finset.card sur un Finset borné : sans piège.

n°	Convention	Effet à surveiller	Dans PrimeGaps
19	Sommes et produits	Somme vide = 0, produit vide = 1 ; <code>tsum</code> ($\sum'$) d'une famille non sommable vaut 0.	Non utilisés.
20	<code>iSup</code> , <code>iInf</code> , <code>limsup</code> , <code>liminf</code> , <code>Finset.sup</code>	Même régime que <code>sSup</code> : valeur-poubelle si vide ou non borné ; <code>Finset.sup</code> d'un ensemble vide vaut $\perp$.	Seul <code>sSup</code> est employé (point 6).
21	Dérivée et intégrale	<code>deriv f x = 0</code> si f n'est pas dérivable en x ; $\int f = 0$ si f n'est pas intégrable.	Non utilisées.
22	Arithmétique de <code>ENNReal</code>	$\top - \top = 0$, $\top \cdot 0 = 0$, <code>ENNReal.toReal</code> $\top = 0$.	Non utilisée.
23	<code>ZMod</code> , <code>Fin</code>	<code>ZMod 0</code> est $\mathbb{Z}$; <code>Fin 0</code> est vide ; arithmétique modulaire dans <code>Fin n</code> ; <code>ZMod.val</code> .	Non utilisés.
24	Fonctions spéciales aux pôles	Valeurs fixées par convention (par exemple <code>Gamma</code> aux entiers négatifs ou nul : 0) ; à lire dans la documentation pour <code>riemannZeta</code> en 1.	Non utilisées.
25	Choix et description	<code>Classical.choose</code> , <code>Classical.epsilon</code> , <code>Nat.find</code> , <code>Function.invFun</code> : valeur arbitraire si l'existence n'est pas prouvée dans l'énoncé.	Non utilisés.
26	Vacuité	Une quantification sur un ensemble vide, ou des définitions qui rendent une hypothèse contradictoire, rendent le théorème trivial. Contrôle : exhiber un exemple concret qui satisfait les hypothèses.	Non discuté dans la note ; les deux parties de l'énoncé portent sur des ensembles infinis d'indices.
27	Quantificateurs et in-égalités	Ordre $\exists N_0 \forall N \geq N_0$, inégalité stricte ou large, “pour N assez grand” exprimé par un filtre (<code>atTop</code>) ou à la main.	Relu dans la section 8 de la note.
28	Variables implicites automatiques	Avec <code>autoImplicit</code> , une variable mal orthographiée devient silencieusement un argument universellement quantifié.	Non vérifié dans la note.
29	Notations et espaces de noms	<code>local notation</code> , <code>open ... in</code> , noms qui masquent ceux de Mathlib ; deux définitions de même contenu sous deux noms (ici <code>prime</code> et <code>primeAt</code>).	À contrôler : la note signale les deux définitions et les préfixes omis.
30	Code exécutable dans la base de confiance	<code>native_decide</code> ajoute un axiome qui fait confiance au compilateur ; <code>implemented_by</code> , <code>extern</code> , <code>unsafe</code> , <code>partial</code> .	Recherche textuelle faite (aucune occurrence de <code>native_decide</code>).
31	Métaprogrammation et options du noyau	Une commande <code>elab</code> , <code>macro</code> ou <code>run_cmd</code> peut produire des déclarations sans le mot <code>axiom</code> ; des options de débogage (par exemple <code>debug.skipKernelTC</code>) peuvent désactiver le contrôle du noyau. <code>#print axioms</code> reste le contrôle à garder, et une recherche de <code>set_option</code> dans les 56 fichiers le complète.	<code>#print axioms</code> fait ; recherche de <code>set_option</code> et de métaprogrammes non mentionnée dans la note.

Remarque. Seules les lignes 1, 2, 6, 7, 8, 9, 10 et 27 concernent réellement les définitions de la démonstration dite **primeGaps** du dépôt d'openai du 5 octobre 2026.

L'ensemble des lignes peuvent servir de grille pour relire tout énoncé écrit en lean ; pour un énoncé donné, la méthode est la même : repérer chaque symbole de mathlib employé, chercher sa convention aux cas limites, puis vérifier que l'énoncé n'atteint jamais ces cas.