

Post-Positional Numeracy: Finite-Adele Encoding and Product-Formula-Verified Exact Rational Arithmetic

Rowan Brad Quni-Gudzin

2026-08-26 · v1.0.1 · DOI: 10.5281/zenodo.22114495

Abstract. Exact computation on the rationals today inhabits a single completion. Digital arithmetic runs in the real numbers and accepts rounding; exact alternatives run in a single p -adic completion through Hensel codes and reconstruct through the Chinese remainder theorem with Farey bounds. This paper develops the multi-place realization that joins the two. A rational number is encoded by its residues at finitely many chosen primes together with a two-sided Archimedean window; the encoding is injective whenever $2B^2 < M$, where M is the product of the chosen prime powers. The adelic product formula — the identity that ties the places of $\mathbb{Q}$ together — becomes a machine-checkable invariant of the arithmetic: every correct encode-compute-decode round-trip of operands whose numerator and denominator factor over the chosen primes satisfies it exactly, and any violation localizes the failing place. All claims are computationally verified: golden values, exhaustive injectivity on small moduli, 10^5 seeded trials on a window of modulus $M = 810,000$, 8×10^4 componentwise arithmetic checks, and a reconstruction algorithm validated against exhaustive enumeration. A dependency-free reference implementation and a reproducibility statement accompany the paper.

1. Introduction

Numerical computation splits into two worlds. Digital arithmetic runs in the real numbers and accepts rounding; the classical alternative, exact arithmetic on the rationals, runs through p -adic Hensel codes and reconstructs rational answers from residues [krishnamurthy1975finite, gregory1984methods]. Both worlds are single-place: each computation commits to one completion of $\mathbb{Q}$. Ostrowski's theorem states that the choice is not a free one — every nontrivial absolute value on $\mathbb{Q}$ is equivalent to the real absolute value or to a p -adic one [ostrowski1916funktionalgleichung] — but the theorem also implies that no single completion is privileged. Computation has simply never carried more than one at a time.

The published record establishes the conceptual side of this observation in detail: positional notation is a tree whose topology the radix sets [qnfo:silent-radix, qnfo:nonlinear-tree, qnfo:ultrametric-foundation-v2], numeral systems admit multi-axis evaluation [qnfo:numerata], and a computation-ready framework exists for exact single-place arithmetic with Hensel codes [qnfo:hensel-framework]. What has not been built is the multi-place layer: an encoding that carries several places at once, and a global check that ties them. This paper supplies that layer.

We call the resulting representation *post-positional numeracy*: a rational number is represented not by one positional expansion but by its simultaneous images at finitely many chosen places, checked against the one global identity that constrains them all — the adelic product formula $\prod_v |x|_v = 1$ [tate1967fourier]. The contributions are three. First, the finite-adele encoding with its injectivity window: the encoding is injective on a two-sided Farey window whenever $2B^2 < M$ (Lemma 1), and the two-sided bound is essential (Remark 1). Second, the product formula as a verification invariant: every correct encode-compute-decode round-trip of operands whose numerator and denominator factor over the chosen primes satisfies the truncated product formula exactly, and a violation localizes the failing place (Theorem 1 and Corollary 1). Third, a dependency-free reference implementation whose claims are computationally verified (Section 5).

The results matter to three audiences. For exact-arithmetic practice, the product formula is a checksum with teeth: it verifies a multi-place computation against an identity independent of the computation itself, and it points at the place that failed. For number theory, the product formula — normally an abstract statement about all places at once — becomes runnable. For notation, the paper exhibits what a number looks like when no place is privileged.

2. Preliminaries

2.1 Absolute values and Ostrowski's theorem

For a rational number $x = a/b$ in lowest terms and a prime p , write $v_p(x)$ for the exponent of p in a minus its exponent in b . The p -adic absolute value is $|x|_p = p^{-v_p(x)}$, and the real absolute value is $|x|_\infty = |x|$. Ostrowski's theorem states that every nontrivial absolute value on $\mathbb{Q}$ is equivalent to one of these [ostrowski1916funktionalgleichung]; the completions are $\mathbb{R}$ and the fields $\mathbb{Q}_p$. For every nonzero $x \in \mathbb{Q}$ the product over all places satisfies

$$\prod_v |x|_v = 1,$$

with only finitely many factors different from 1 [tate1967fourier].

2.2 Hensel codes and rational reconstruction

A Hensel code of length k at the prime p is the residue of a rational modulo p^k [hensel1908theorie, krishnamurthy1975finite]. Addition, subtraction, multiplication, and — when the denominator is coprime to p — division are exact integer operations modulo p^k . Given a residue r modulo a composite M together with a bound B with $2B^2 < M$, the rational a/b with $\gcd(b, M) = 1$, $|a| \leq B$, $|b| \leq B$, and $ab^{-1} \equiv r \pmod{M}$ is unique, and a two-step Euclidean algorithm recovers it [wang1982padic, dixon1982exact, miola1982conversion, kornrup1983mapping, krishnamurthy1983conversion, rao1984conversion]. Primes at which a modular computation degenerates can be detected and handled [boehm2015badprimes]; production systems implement the full stack [doris2021exact].

2.3 Notation

Throughout, S denotes a finite set of primes, $k \geq 1$ a precision, and

$$M = \prod_{p \in S} p^k, \quad B = \lfloor \sqrt{M/2} \rfloor.$$

The window W is the set of rationals $x = a/b$ in lowest terms with $\gcd(b, M) = 1$, $|a| \leq B$, and $|b| \leq B$. A rational is S -smooth when both its numerator and its denominator factor over S .

2.4 A remark on terminology

"Ostrowski numeration systems" denotes a different subject: numeration built from continued-fraction expansions [hieronymi2014ostrowski]. That literature is unrelated to Ostrowski's theorem on absolute values, which is the theorem used here.

3. The finite-adele encoding

3.1 Definition

Definition 1 (finite-adele encoding). The encoding of $x \in W$ is its residue vector

$$\varphi(x) = (x \bmod p^k)_{p \in S} \in \prod_{p \in S} \mathbb{Z}/p^k\mathbb{Z},$$

computed componentwise as $a \cdot b^{-1} \bmod p^k$ for $x = a/b$. The vector is the truncated restricted product of x 's local components: its finite-adele image.

3.2 Injectivity

Lemma 1. The encoding φ is injective on the window W .

Proof. Let $x = a/b$ and $y = c/d$ in W with $\varphi(x) = \varphi(y)$. For each $p \in S$, $ab^{-1} \equiv cd^{-1} \pmod{p^k}$, hence $ad \equiv bc \pmod{p^k}$. The moduli p^k for $p \in S$ are pairwise coprime, so by the Chinese remainder theorem $ad \equiv bc \pmod{M}$. Both $|ad|$ and $|bc|$ are at most B^2 , so $|ad - bc| \leq 2B^2 < M$. A multiple of M strictly smaller than M in absolute value is zero: $ad = bc$, and $x = y$. $\square$

Remark 1 (the two-sided bound is essential). An Archimedean bound on x alone does not suffice. Since $7 \cdot 13 = 91 \equiv 1 \pmod{30}$, the rationals $1/7$ and 13 share the image $(1, 1, 3)$ modulo $(2, 3, 5)$; both satisfy $|x| \leq 15$, and both denominators are coprime to 30 . The injectivity window must bound numerator and denominator separately.

3.3 Reconstruction

Algorithm 1 (two-step Euclidean reconstruction). Given r with $0 \leq r < M$ and the bound B : run the Euclidean algorithm on the pair (M, r) ; at the first step where the remainder does not exceed B , the pair (remainder, coefficient) is a candidate; if the coefficient exceeds B , take one further step and read the pair there. Normalize the sign so the denominator is positive and reduce by gcd. If no candidate satisfies both bounds, report failure. Correctness follows from the standard analysis of the reconstruction algorithm [wang1982adic, dixon1982exact]; Section 5 validates the implementation against exhaustive enumeration.

4. The product formula as a verification invariant

4.1 The S -smooth invariant

Theorem 1. If $x \in \mathbb{Q}^\times$ is S -smooth, then

$$\prod_{v \in S \cup \{\infty\}} |x|_v = 1.$$

Proof. Write $x = \pm \prod_{p \in S} p^{e_p}$ with integer exponents. Then $|x|_p = p^{-e_p}$ for each $p \in S$, and $|x|_\infty = \prod_{p \in S} p^{e_p}$. The product is 1. $\square$

4.2 General operands

For x not S -smooth the truncated product does not equal 1. Writing $x = \pm \left(\prod_{p \in S} p^{e_p} \right) \left(\prod_{q \notin S} q^{f_q} \right)$ and applying the same computation gives the general identity

$$\prod_{v \in S \cup \{\infty\}} |x|_v = \left(\prod_{q \notin S} |x|_q \right)^{-1}.$$

Both identities are derived from unique factorization and are verified numerically in Section 5.

4.3 Failure localization

Corollary 1. After a multi-place encode-compute-decode round-trip on S -smooth operands, the deviation of the truncated product from 1 is a rational of the form $\prod_{p \in S} p^{d_p}$; the primes with nonzero exponent identify the failing places.

Argument. The truncated product is multiplicative across components, and each component contributes a pure prime power. A wrong component at the prime p changes exactly the factors at p , so the deviation factors over the primes whose components failed. If the result of a round-trip is checked and the product equals 1, the computation is consistent at every place in S at once.

5. Computational verification

5.1 Method

A single dependency-free script executes the following checks. Product-formula checks: the golden values $x \in \{6, 2/3, 12\}$ over $S = \{2, 3\}$; the boundary value $x = 5/2$, where the truncated product is $5 = 1/|5/2|_5$; 10^4 seeded random S -smooth trials over $S = \{2, 3, 5, 7\}$; and 10^4 seeded trials of the general identity with an outside prime factor. Ostrowski checks: the strong triangle inequality on 2×10^4 random integer pairs, the bound $|n|_2 \leq 1$ for integers, and the full product formula for four values. Injectivity checks: exhaustive enumeration of the window for $M = 36$ ($B = 4$), $M = 216$ ($B = 10$), and $M = 30$ ($B = 3$), counting distinct images; and 10^5 seeded random window pairs for $S = \{2, 3, 5\}$, $k = 4$ ($M = 810,000$, $B = 636$), encoded, reconstructed, and compared. The reconstruction algorithm is validated against exhaustive enumeration on $M = 216$. Round-trip checks: 2×10^4 random window operand pairs under addition, subtraction, multiplication, and division, each compared componentwise, with a subset carried through full reconstruction.

5.2 Results

All checks pass. The seeded trial on $M = 810,000$ accepted 94,998 pairs with zero collisions and zero reconstruction failures; the componentwise arithmetic checks passed 80,000 of 80,000; full round-trips reconstructed 338 results exactly.

Check	Result
Product formula, golden values (6, 2/3, 12, $S = \{2, 3\}$)	exactly 1 each
Product formula, boundary ($x = 5/2$, $S = \{2, 3\}$)	$5 = 1/ 5/2 _5$
Product formula, S -smooth trials (10^4)	$10^4/10^4$; largest deviation 4.4×10^{-16}
General identity, trials with outside prime (10^4)	$10^4/10^4$
Strong triangle inequality (2×10^4 pairs)	20,000/20,000
Injectivity, exhaustive ($M = 36, 216, 30$)	9/9, 55/55, 7/7 distinct images
Injectivity, seeded (10^5 trials, $M = 810,000$, $B = 636$)	94,998 accepted; 0 collisions
Reconstruction vs. enumeration ($M = 216$)	63/63
Componentwise $+$, $-$, $\times$, $\div$	80,000/80,000
Full reconstruct round-trips	338 exact

5.3 Reproducibility

Seed 20260826; Python 3.8 or later; standard library only; single-threaded runtime under one minute. The script `verify_ppn.py` and its machine-readable output accompany this paper. Re-running the script reproduces every number in the table.

6. Crosswalk

The same objects carry different names in three communities. The table translates the paper's terms into their adjacent-domain equivalents.

Number theory	Exact computation	Notation
place of $\mathbb{Q}$ (a choice of metric on the rationals)	the arithmetic the computation is faithful under	which distance the notation presumes
Hensel code	finite p -adic segment (residue mod p^k)	digits at a non-Archimedean place
finite adeles (truncated restricted product)	the record carrying all chosen places at once	multi-place numeracy: one number, many simultaneous local images
product formula $\prod_v x _v = 1$	global integrity checksum of a multi-place computation	the identity that ties the places: no place may win
rational reconstruction (CRT + Farey bounds)	exact decode from multi-place residues	recovering one number from its many local appearances
Ostrowski's theorem	why $\mathbb{R}$ and $\mathbb{Q}_p$ are the only completions	why "the" number line was always a choice

7. Practitioner relevance

A practitioner can build exact multi-place rational arithmetic from Section 3 and Section 4 directly. Encode a rational as its residue vector at a chosen prime set and precision; perform addition, subtraction, multiplication, and division componentwise as integer operations modulo the prime powers; reconstruct with Algorithm 1; and check the truncated product formula at every round-trip as the global invariant that localizes any failing place. The deliverable is a dependency-free module in Python with a reproducible test suite, suitable as the arithmetic core of exact linear algebra, financial computation, simulation, and teaching tools. The invariant check is the piece standard modular methods do not provide: a single identity that audits all places of a computation simultaneously rather than per-operation assertions.

8. Related work

The Hensel-code lineage spans five decades: the origin of finite-segment p -adic arithmetic [krishnamurthy1975finite], its use for exact computation [gregory1978finite, gregory1984methods], reconstruction from residues [wang1982padic, dixon1982exact], conversion methods [miola1982conversion, krishnamurthy1983conversion, rao1984conversion], and the Farey-fraction view of the window [korerup1983mapping]. Bad-prime handling is classical [boehm2015badprimes], and exact p -adic computation is available in mainstream systems [doris2021exact]. Closest to the multi-place setting are simultaneous rational number codes, which decode beyond half the minimum distance using multiplicity codes and bad-prime detection [abbondati2026simultaneous]; they do not use the product formula as an invariant, and the present paper's checksum complements their decoding guarantees. A published single-place framework implements Hensel codes end to end with tests and benchmarks [qnfo:hensel-framework]. The conceptual line this paper extends

reads positional notation as a tree [qnfo:silent-radix, qnfo:nonlinear-tree], evaluates numeral systems on multiple axes [qnfo:numerata], and traces the role of notation in how mathematics is grounded [qnfo:ten-fingered-trap, qnfo:decimal-fingers, qnfo:embodied-math, qnfo:explicit-frame].

9. Discussion

The paper's map-territory hygiene is worth stating. The encoding is a map: a notation for rationals, not the rationals themselves. Lemma 1 quantifies the map's faithfulness — on the window W the map is injective, and Remark 1 shows the faithfulness fails if the window is stated one-sided. The product formula is the constraint that the territory imposes on every faithful map across places; using it as a checksum converts that constraint into machinery.

An open question the verification suggests is the cost of exactness: what a multi-place round-trip costs against a single-place one in cycles, memory, and energy, and where the crossover lies for practical workloads. The measurement is straightforward to set up and is left for future work.

10. Declarations

Funding. This research received no external funding. **Conflicts of interest.** None. **Author contributions.** The named author conceived the encoding and the invariant, and authored this paper. **Use of artificial intelligence.** This paper was drafted with AI assistance under the disclosure and audit principles of the published methodology [qnfo:uia, qnfo:iaps]; all mathematical claims and every numerical result were verified by the executable reproducibility suite described in Section 5. **Code availability.** The reference implementation and the verification script accompany this paper in the same archive.