Nested Rational Intervals for Non-Surjectivity of $\mathbb{N} \to [0, 1] \cap \mathbb{Q}$: A Coq Formalization with Minimal Axioms

Horsocrates
Independent Researcher
horsocrates@proton.me

January 2026

Abstract

We formalize in Coq that there is no surjection from $\mathbb{N}$ onto the rational interval $[0, 1] \cap \mathbb{Q}$, using only the Law of Excluded Middle (LEM) as an external axiom—without the Axiom of Infinity, Axiom of Choice, or function extensionality. The proof employs nested rational intervals with trisection and comprises 167 fully proven lemmas with 0 Admitted.

Additionally, we formalize ε-approximate versions of the Intermediate Value Theorem and Extreme Value Theorem for functions $\mathbb{Q} \to \mathbb{Q}$ (23 lemmas each, 0 Admitted).

Our main technical contributions are:

1. **Deterministic witness selection via order-preserving choice.** We resolve ambiguity in witness construction by selecting the leftmost candidate, yielding Leibniz equality instead of propositional equality.
2. **Index-based argmax for EVT.** By returning the index of a maximum rather than its value, we obtain definitional equality in witness lemmas.
3. **Trisection over bisection.** Our nested intervals use trisection, avoiding the "digit stability problem" where small perturbations change digit representations discontinuously.
4. **Executable extraction.** The Coq proof yields an extracted OCaml program that computes a witness for any given enumeration.

The full formalization comprises 397 proven lemmas across 10 modules. The 10 remaining Admitted lemmas require either completeness of reals (marking the $\mathbb{Q}/\mathbb{R}$ boundary) or concern universe-level type-theoretic constraints.

Important clarification: We also prove that $\mathbb{Q}$ is countable (explicit bijection $\mathbb{N} \leftrightarrow \mathbb{Q}$ via Calkin-Wilf tree). Our non-surjectivity result concerns Cauchy *processes* (functions $\mathbb{N} \to \mathbb{Q}$), not individual rationals.

All code is available at `https://github.com/Horsocrates/theory-of-systems-coq`.

Keywords: Coq formalization, nested intervals, rational arithmetic, non-surjectivity, finitistic methods, deterministic witnesses, minimal axioms

Contents

1 Introduction

1.1 Problem Statement

We address the following question: can the non-surjectivity of $\mathbb{N}$ onto $[0,1] \cap \mathbb{Q}$ be proven in Coq using only the Law of Excluded Middle, without the Axiom of Infinity or Axiom of Choice?

The classical diagonal argument proves that $\mathbb{R}$ is uncountable by constructing, for any enumeration $f : \mathbb{N} \to \mathbb{R}$, a real number differing from $f(n)$ in the n-th digit. This relies on treating infinite decimal expansions as completed objects. We ask whether a similar result holds over $\mathbb{Q}$ with weaker assumptions.

Clarification on terminology. Throughout this paper, "non-surjectivity of $\mathbb{N}$ onto $[0,1] \cap \mathbb{Q}$" means: for any function $f : \mathbb{N} \to \mathbb{Q}$ with range in $[0,1]$, there exists $q \in [0,1] \cap \mathbb{Q}$ such that $q \neq f(n)$ for all n. This is distinct from the classical uncountability of $\mathbb{R}$; we work entirely within $\mathbb{Q}$.

1.2 Main Results

Our formalization establishes:

Theorem 1.1 (Non-Surjectivity). *For any $f : \mathbb{N} \to \mathbb{Q}$, there exists $q \in [0,1] \cap \mathbb{Q}$ such that $q \neq f(n)$ for all n.*

In Coq:

```
Theorem unit_interval_uncountable_trisect_v2 : forall E :
    Enumeration,
  valid_regular_enumeration E ->
  exists D : RealProcess,
    is_Cauchy D /\ (forall m, 0 <= D m <= 1) /\ (forall n, not_equiv
        D (E n)).
```

Here `RealProcess := nat -> Q` represents a Cauchy sequence of rationals, and `not_equiv` asserts that two processes diverge by at least some fixed ε.

Theorem 1.2 (ε-IVT). *If $f : \mathbb{Q} \to \mathbb{Q}$ is uniformly continuous on $[a,b]$ with $f(a) < 0 < f(b)$, then for any $\varepsilon > 0$, there exists $c \in [a,b]$ with $|f(c)| < \varepsilon$.*

Theorem 1.3 (ε-EVT). *If $f : \mathbb{Q} \to \mathbb{Q}$ is uniformly continuous on $[a,b]$, then for any $\varepsilon > 0$, there exists $c \in [a,b]$ such that $f(c) \geq f(x) - \varepsilon$ for all $x \in [a,b]$.*

1.3 Technical Contributions

Beyond the theorems themselves, we contribute techniques for formal verification over $\mathbb{Q}$:

1. **Deterministic witness selection.** When multiple candidates satisfy a specification (e.g., plateau in argmax), we select the leftmost, yielding unique witnesses with Leibniz equality. This avoids the pervasive `Qeq`/`=` mismatch in Coq's rational library.

2. **Index-based maximum.** For EVT, returning `argmax_idx` (the index of a maximum) rather than `argmax` (its value) gives definitional equality in witness lemmas.

3. **Trisection construction.** We use trisection rather than bisection or digit extraction. When avoiding a point p in interval $[a,b]$, at least $2/3$ of the interval remains available regardless of where p falls.

1.4 Method

The uncountability proof uses nested intervals rather than diagonal argument. Given an enumeration f, we construct a sequence of intervals $[a_n, b_n]$ such that:

1. Each interval is contained in the previous: $[a_{n+1}, b_{n+1}] \subseteq [a_n, b_n]$
2. Each interval excludes $f(n)$: $f(n) \notin [a_n, b_n]$
3. The intervals shrink: $b_n - a_n \to 0$

The construction proceeds by trisection: divide $[a_n, b_n]$ into thirds and select a third that excludes $f(n)$. This always succeeds because $f(n)$ can occupy at most one third.

The key observation is that this proof never requires "the limit point" to exist as a completed object. We prove that for any n, there exists a rational in $[a_n, b_n]$ distinct from $f(1), \ldots, f(n)$. The "limit" is a horizon we approach, not an object we reach.

1.5 Axioms Used

Our formalization uses exactly one axiom beyond Coq's core type theory:

```
Axiom classic : forall P : Prop, P \/ ~P.
```

This is the law of excluded middle (LEM). We use **no Axiom of Infinity**, **no Axiom of Choice**, and **no function extensionality**.

1.6 Paper Structure

Section 2 establishes preliminaries. Section 3 presents deterministic witness selection. Section 4 details the trisection construction. Section 5 covers ε-IVT and ε-EVT. Section 6 discusses proof-theoretic strength. Section 7 analyzes the Admitted lemmas. Section 8 covers related work. Section 9 concludes.

2 Preliminaries

2.1 The Coq Proof Assistant

Coq is an interactive theorem prover based on the Calculus of Inductive Constructions. Proofs in Coq are programs; verified theorems are type-checked terms. This provides a high degree of assurance: if Coq accepts a proof, it is correct relative to Coq's kernel.

We use Coq version 8.18.0 with the standard library's rational number implementation (`QArith`). Our proofs use standard tactics including `lia` and `nia` for linear and nonlinear integer arithmetic, `field` for rational field equations, and `setoid_rewrite` for reasoning up to rational equality (`Qeq`).

2.2 Rational Numbers in Coq

Coq's rationals are defined as pairs of integers with nonzero denominator:

```
Record Q : Set := Qmake { Qnum : Z ; Qden : positive }.
```

Equality on rationals is not definitional but propositional:

```
Definition Qeq (p q : Q) := Qnum p * Qden q = Qnum q * Qden p.
```

This means `1/2` and `2/4` are not identical (`=`) but are equal (`==`). Much of our technical work involves managing this distinction.

2.3 What "Without Axiom of Infinity" Means

In ZFC, the Axiom of Infinity asserts the existence of an inductive set—a set containing $\emptyset$ and closed under the successor operation $x \mapsto x \cup \{x\}$. This axiom is necessary to prove that $\mathbb{N}$ exists as a completed set.

Coq's type theory does not include ZFC's Axiom of Infinity. Natural numbers are defined inductively:

```
Inductive nat : Set :=
  | O : nat
  | S : nat -> nat.
```

This defines $\mathbb{N}$ as a type, not a set. Crucially, we never assert that all natural numbers exist simultaneously as a completed collection. Each natural number exists as a term; the type `nat` is a specification of how to form natural numbers, not a container holding infinitely many objects.

2.4 Formal Specification

The following table summarizes our formal setup:

Component	Coq Representation	Significance
Classical logic	`classic` axiom	LEM without Choice
Inductive naturals	`Inductive nat`	No completed $\mathbb{N}$
Decidable comparison	`Qlt_le_dec`	Computable ordering
Leftmost selection	First witness in list	Deterministic choice
Convergence as process	`RealProcess := nat -> Q`	No completed limits

3 Deterministic Witness Selection

3.1 The Problem: Qeq vs Leibniz Equality

A persistent challenge in Coq's rational arithmetic is the mismatch between propositional equality (`Qeq`, denoted `==`) and Leibniz equality (`=`). When a lemma produces a witness q satisfying some property, subsequent lemmas may require the *same* q—but `Qeq` only guarantees an *equivalent* rational.

Example 3.1. Consider finding a maximum of a function $f : \mathbb{Q} \to \mathbb{Q}$ on a finite grid. The standard approach returns some x with $f(x) = \max_i f(x_i)$. But if f has a plateau (multiple maxima), the choice of x is arbitrary. Different proof branches may select different representatives, breaking later proofs that assume a unique witness.

3.2 The Solution: Leftmost Selection

We resolve ambiguity by selecting the **leftmost** candidate—the one with minimal index in the enumeration.

```
Fixpoint find_max_idx_acc (f : Q -> Q) (l : list Q)
  (curr_idx best_idx : nat) (best_val : Q) : nat :=
  match l with
  | [] => best_idx
  | x :: xs =>
      if Qle_bool best_val (f x)
      then find_max_idx_acc f xs (S curr_idx) curr_idx (f x)
      else find_max_idx_acc f xs (S curr_idx) best_idx best_val
  end.
```

```
Definition argmax_idx (f : Q -> Q) (l : list Q) : nat :=
  match l with
  | [] => 0
  | x :: xs => find_max_idx_acc f xs 1 0 (f x)
  end.
```

The key insight: `Qle_bool best_val (f x)` uses $\leq$, not $<$. When $f(x)$ equals the current best, we do *not* update. Since we traverse left-to-right, the *first* occurrence of the maximum wins.

3.3 Benefits

1. **Leibniz equality.** The index is a natural number; `argmax_idx f l = n` is definitional equality.
2. **Determinism.** Given the same inputs, the same index is always returned.
3. **Proof simplification.** Witness lemmas can use `reflexivity` instead of `Qeq` reasoning.

4 Trisection Construction

4.1 Why Trisection?

Classical uncountability proofs often use digit extraction: the diagonal real differs from $f(n)$ in the n-th digit. This fails over $\mathbb{Q}$ because:

1. Rationals have finite or repeating decimal expansions
2. Digit extraction is discontinuous: $\lfloor 10^n x \rfloor \mod 10$ can change drastically with small perturbations of x
3. The "diagonal" may not be rational

Bisection improves on digits but has its own problem: if $f(n)$ lands exactly on the midpoint, both halves contain $f(n)$ in their closure.

Trisection resolves this. Dividing $[a, b]$ into thirds:

- $[a, a + w/3]$ (left third)
- $[a + w/3, b - w/3]$ (middle third)
- $[b - w/3, b]$ (right third)

where $w = b - a$. Given a point p to exclude, at least two thirds are entirely free of p.

4.2 The Avoid Function

```
Definition avoid_third (p a b : Q) : Q * Q :=
  let w := b - a in
  let third := w / 3 in
  let m1 := a + third in
  let m2 := b - third in
  if Qlt_le_dec p m1 then (m1, b)       (* p in left -> take middle-
      right *)
  else if Qlt_le_dec m2 p then (a, m2) (* p in right -> take left-
      middle *)
  else (a, m1).                          (* p in middle -> take LEFT
      *)
```

The crucial choice: when p is in the middle third, we take the **left** third. This ensures:

- The left endpoint is non-decreasing across iterations
- The sequence of left endpoints is monotone
- Determinism: the same enumeration always produces the same intervals

4.3 Nested Intervals

```
Fixpoint trisect_iter (E : Enumeration) (s : Bisection) (n : nat) :
    Bisection :=
  match n with
  | O => s
  | S n' =>
      let s' := trisect_iter E s n' in
      let ref := 12 * (3 ^ n') in
      let p := E n' ref in
      mkBisection (fst (avoid_third p (bis_left s') (bis_right s')))
                  (snd (avoid_third p (bis_left s') (bis_right s')))
  end.
```

4.4 Main Theorem

```
Theorem unit_interval_uncountable_trisect_v2 : forall E :
    Enumeration,
  valid_regular_enumeration E ->
  exists D : RealProcess,
    is_Cauchy D /\
    (forall m, 0 <= D m <= 1) /\
    (forall n, not_equiv D (E n)).
Proof.
  intros E Hvalid.
  exists (diagonal_trisect_v2 E).
  split; [| split].
  - apply diagonal_trisect_v2_is_Cauchy.
  - intro m. apply diagonal_trisect_v2_in_unit.
  - intro n. apply diagonal_trisect_v2_differs_from_E_n. exact
     Hvalid.
Qed.
```

5 ε-IVT and ε-EVT

5.1 ε-Approximate Theorems

Over $\mathbb{Q}$, exact versions of IVT and EVT fail: there may be no rational c with $f(c) = 0$ or $f(c) = \max f$. However, ε-approximate versions hold.

Theorem 5.1 (ε-IVT, formal)

```
Theorem IVT_epsilon : forall (f : Q -> Q) (a b eps : Q),
  a < b -> eps > 0 ->
  uniform_continuous f a b ->
  f a < 0 -> f b > 0 ->
  exists c : Q, a <= c <= b /\ Qabs (f c) < eps.
```

The proof uses bisection: at each step, take the half where f changes sign. After n steps, the interval has width $(b-a)/2^n$. By uniform continuity, $|f(c)|$ can be made arbitrarily small.

Theorem 5.2 (ε-EVT, formal).

```
Theorem EVT_epsilon : forall (f : Q -> Q) (a b eps : Q),
  a < b -> eps > 0 ->
  uniform_continuous f a b ->
  exists c : Q, a <= c <= b /\
    forall x, a <= x <= b -> f c >= f x - eps.
```

The proof uses grid sampling with the index-based argmax from Section 3.

6 Discussion: Proof-Theoretic Strength

6.1 Axiomatic Strength and Reverse Mathematics

Our formalization uses:

- Coq's Calculus of Inductive Constructions (CIC)
- Law of Excluded Middle (`classic`)

It does **not** use:

- Axiom of Infinity (`nat` is inductively defined)
- Axiom of Choice
- Function extensionality
- Propositional extensionality

Proof-theoretic classification. The proof-theoretic strength is approximately **PRA** + **LEM** (Primitive Recursive Arithmetic with classical logic) or equivalently $\mathbf{I\Sigma_1}$ + **LEM**. All functions definable in our system are primitive recursive; all quantification is bounded or over inductively defined types.

Significance for foundations. Our formalization demonstrates that the non-surjectivity theorem—traditionally seen as requiring "actual infinity" to state and prove—is in fact provable in systems without any infinite sets. The key insight is separating:

1. **Unbounded iteration** ($\forall n \in \mathbb{N}, P(n)$)—provable in weak arithmetic
2. **Completed infinite sets** ($\exists S$ such that $S = \{n : n \in \mathbb{N}\}$)—requires Axiom of Infinity

Classical proofs of uncountability conflate these notions by treating $\mathbb{R}$ as a completed set. Our proof uses only (1): for any enumeration f and any n, we construct an interval excluding $f(n)$. The "diagonal" is never a completed object—it's a procedure that, given n, outputs a rational approximation.

6.2 Comparison with Simpson's Hierarchy

In the framework of *Subsystems of Second Order Arithmetic* [7], our results inhabit the following position:

System	Infinity	Choice	Our theorems
$\mathbf{RCA}_0$	No	No	✓ Countability of $\mathbb{Q}$
$\mathbf{RCA}_0$ + **LEM**	No	No	✓ Non-surjectivity, ε-IVT, ε-EVT
$\mathbf{WKL}_0$	No	Weak König	Not needed
$\mathbf{ACA}_0$	Arithmetical compr.	—	Not needed

The non-surjectivity theorem is often assumed to require ACA_0 (because "the diagonal real" seems to require comprehension). Our proof shows this is not so: the diagonal is computed step-by-step, never formed as a completed object.

6.3 Hilbert's Program, Partially Realized

Hilbert sought to reduce infinitary mathematics to finitary reasoning. Our formalization provides a concrete example: the "uncountability of the continuum" (in its non-surjectivity form) is finitistically reducible. The proof uses only:

- Primitive recursive arithmetic on $\mathbb{Q}$
- Classical logic (LEM)
- Induction on $\mathbb{N}$

No transfinite methods, no completed infinities, no choice principles.

6.4 The Contrast: Countable Points vs Uncountable Processes

A potential objection: "$\mathbb{Q}$ is countable, so how can $[0,1] \cap \mathbb{Q}$ be uncountable?"

The resolution lies in distinguishing **objects** from **processes**:

Theorem 6.1 ($\mathbb{Q}$ is countable)**.** *There exists a bijection $\mathbb{N} \to \mathbb{Q}$. Proof: Calkin-Wilf tree [4]. Fully constructive—no axioms required.*

Theorem 6.2 (Cauchy processes are uncountable)**.** *For any enumeration $E : \mathbb{N} \to (\mathbb{N} \to \mathbb{Q})$ of Cauchy sequences, there exists a Cauchy sequence D not in the enumeration. Proof: nested trisection intervals (this paper). Requires LEM.*

These are not contradictory because they enumerate different things:

What is enumerated	Cardinality	Axioms needed
$\mathbb{Q}$ as pairs (p, q)	Countable	None
Cauchy sequences $\mathbb{N} \to \mathbb{Q}$	Uncountable	LEM

7 The Unproven Lemmas

Our formalization contains 10 lemmas marked `Admitted`. We categorize them to show they are not gaps but boundaries.

7.1 Completeness of Reals (2 lemmas)

`Heine_Borel` and `continuity_implies_uniform` require that nested intervals converge to a *point*—a completed real number. Over $\mathbb{Q}$, nested intervals may "converge" to an irrational, which does not exist in our domain.

These lemmas mark the **boundary between $\mathbb{Q}$ and $\mathbb{R}$**. They are not provable in our framework because our framework uses $\mathbb{Q}$, not $\mathbb{R}$.

7.2 Universe-Level Constraints (3 lemmas)

`update_increases_size`, `no_self_level_elements`, and `cantor_no_system_of_all_L2_systems` concern the hierarchy of types in Coq's universe system. They formalize that systems cannot contain themselves—but this lives at the meta-level.

These lemmas confirm that **hierarchical structure is enforced by the type system**.

7.3 Superseded Approaches (3 lemmas)

`extracted_equals_floor`, `diagonal_Q_separation`, and `diagonal_differs_at_n` belong to a digit-extraction approach we abandoned. The interval approach makes them unnecessary.

7.4 Countability Round-Trip (2 lemmas)

The bijection proofs in `Countability_Q.v` have 2 Admitted lemmas for round-trip properties of the Calkin-Wilf encoding. The core injectivity and GCD-preservation theorems are fully proven.

8 Related Work

Constructive analysis. Bishop [1] develops analysis without LEM but with Countable Choice. Our approach retains LEM but rejects completed infinities.

Reverse mathematics. Simpson [7] classifies theorems by set-theoretic strength. Our results live in fragments below RCA_0, suggesting very low proof-theoretic strength.

Coq formalizations. The Mathematical Components library and Coquelicot provide extensive real analysis in Coq. Both use the Axiom of Infinity. Our contribution is demonstrating what can be achieved without it.

Calkin-Wilf tree. Calkin and Wilf [4] introduced the tree enumeration of $\mathbb{Q}^+$ that we formalize.

9 Conclusion

We have formalized in Coq:

1. Non-surjectivity of $\mathbb{N} \to [0,1] \cap \mathbb{Q}$ (167 lemmas, 0 Admitted)

2. ε-approximate IVT and EVT (23 lemmas each, 0 Admitted)

3. Countability of $\mathbb{Q}$ via Calkin-Wilf bijection (12 lemmas, 2 Admitted)

All using only LEM as an external axiom—no Axiom of Infinity, no Choice.

9.1 Future Work

1. **ε-Bolzano-Weierstrass.** Every bounded sequence has an ε-accumulation point.

2. **Comparison with Cauchy reals.** Quantify precisely what our approach gains/loses vs. Coq's stdlib.

3. **Measure theory.** Can Lebesgue measure be developed finitistically?

The broader point: significant mathematics can be formalized with minimal axiomatic commitments. Whether this matters philosophically is debatable; that it works technically is verified.

All code, including extracted OCaml, is available at `https://github.com/Horsocrates/theory-of-systems-coq`.

References

[1] E. Bishop. *Foundations of Constructive Analysis.* McGraw-Hill, 1967.

[2] L. E. J. Brouwer. Intuitionism and formalism. *Bulletin of the American Mathematical Society*, 20(2):81–96, 1913.

[3] G. Cantor. Über eine Eigenschaft des Inbegriffs aller reellen algebraischen Zahlen. *Journal für die reine und angewandte Mathematik*, 77:258–262, 1874.

[4] N. Calkin and H. S. Wilf. Recounting the rationals. *American Mathematical Monthly*, 107(4):360–363, 2000.

[5] S. Feferman. Predicativity. In S. Shapiro, editor, *The Oxford Handbook of Philosophy of Mathematics and Logic*, pages 590–624. Oxford University Press, 2005.

[6] P. Martin-Löf. *Intuitionistic Type Theory.* Bibliopolis, 1984.

[7] S. G. Simpson. *Subsystems of Second Order Arithmetic.* Cambridge University Press, 2nd edition, 2009.

[8] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics.* Institute for Advanced Study, 2013.

[9] The Coq Development Team. *The Coq Reference Manual, Version 8.18.0.* INRIA, 2023. `https://coq.inria.fr/refman`

A Executable Extraction

A.1 How to Run the Demo

The complete demonstration is available as `diagonal_demo.ml` in the repository:

```
# Native compilation (fast)
ocamlopt -o diagonal diagonal_demo.ml
./diagonal

# Bytecode compilation
ocamlc -o diagonal diagonal_demo.ml
./diagonal

# Interactive (no compilation)
ocaml diagonal_demo.ml
```

Requirements: OCaml 4.x or later. No external dependencies.

A.2 Core Extracted Code

```
(* Rationals as int pairs *)
type q = { num : int; den : int }

let q_add a b = { num = a.num * b.den + b.num * a.den; den = a.den *
    b.den }
let q_sub a b = { num = a.num * b.den - b.num * a.den; den = a.den *
    b.den }
let q_div a n = { num = a.num; den = a.den * n }
let q_lt a b = a.num * b.den < b.num * a.den
```

```
(* Trisection: avoid point p in interval [a,b] *)
let avoid_third p a b =
  let width = q_sub b a in
  let third = q_div width 3 in
  let m1 = q_add a third in
  let m2 = q_sub b third in
  if q_lt p m1 then (m1, b)          (* p in left -> take middle-right
      *)
  else if q_lt m2 p then (a, m2)    (* p in right -> take left-middle
      *)
  else (a, m1)                       (* p in middle -> take LEFT *)

(* Calkin-Wilf enumeration (no axioms!) *)
let rec cw_node n =
  if n = 1 then (1, 1)
  else if n mod 2 = 0 then
    let (a, b) = cw_node (n / 2) in (a, a + b)
  else
    let (a, b) = cw_node (n / 2) in (a + b, b)

let enum_qpos n =
  let (a, b) = cw_node (n + 1) in
  { num = a; den = b }
```

A.3 Sample Output

```
=== Calkin-Wilf Enumeration (Q is countable) ===
  enum_qpos( 0) = 1/1
  enum_qpos( 1) = 1/2
  enum_qpos( 2) = 2/1
  enum_qpos( 3) = 1/3
  ...

=== Diagonal Construction (Cauchy processes are uncountable) ===
  Depth 1: diagonal = 1/6,    interval = [0/1, 1/3]
  Depth 2: diagonal = 1/18,   interval = [0/1, 1/9]
  ...
```

B Statistics Summary

File	Qed	Admitted	Status
ShrinkingIntervals_uncountable_ERR.v	167	0	100%
Countability_Q.v	12	2	86%
EVT_idx.v	23	0	100%
IVT_ERR.v	23	0	100%
Archimedean_ERR.v	14	0	100%
SchroederBernstein_ERR.v	14	0	100%
TernaryRepresentation_ERR.v	52	2	96%
DiagonalArgument_integrated.v	41	1	98%
HeineBorel_ERR.v	22	2	92%
TheoryOfSystems_Core_ERR.v	29	3	91%
Total	**397**	**10**	**98%**

Countability as consistency check. The proof of $\mathbb{Q}^+ \cong \mathbb{N}$ via Calkin-Wilf bijection uses **0 axioms**—fully constructive, not even requiring LEM. This confirms that our non-surjectivity result specifically targets functional processes ($\mathbb{N} \to \mathbb{Q}$) rather than discrete point-sets ($\mathbb{Q}$ itself).