**Abstract**

Over Bishop-style constructive mathematics (BISH), we identify the exact logical strength of a familiar non-reflexivity phenomenon in Banach space theory. Working in a classical meta-theory, we prove that the following are equivalent over BISH: the Weak Limited Principle of Omniscience (WLPO); the non-surjectivity of the canonical embedding $J_{\ell^\infty} : \ell^\infty \to (\ell^\infty)^{**}$; and the existence of a real Banach space with a bidual gap.

This equivalence builds on and refines earlier connections between variants of Hahn–Banach and omniscience principles in constructive reverse mathematics (e.g. work of Ishihara; Bridges–Richman; and surveys such as Diener) by isolating the *bare bidual-gap statement* and locating it precisely at WLPO. Our Lean 4 development separates the (meta-classical) witness-extraction step from the (constructive) analysis that derives WLPOfrom an abstract kernel, making the logical hygiene explicit. We also include a constructively careful account of the Boolean algebra of idempotents in $\ell^\infty/c_0$ (a "Stone window") and finite-dimensional surrogates via Cesàro means.

We view the contribution as modest but useful: to place a classically standard fact at an exact level in CRM, and to provide a machine-checked record that may serve future calibrations.

# The Bidual Gap and WLPO: A Constructive Calibration of Banach Space Non-Reflexivity

Paul Chun-Kit Lee[*]
New York University
`dr.paul.c.lee@gmail.com`

September 13, 2025

**AI-Generated Formalization Disclosure.** This entire Lean 4 formalization project was produced by multi-AI agents working under human direction. All proofs, definitions, and mathematical structures in this repository were AI-generated. This represents a case study in using multi-AI agent systems to tackle complex formal mathematics problems with human guidance on project direction. The mathematical content has been verified through Lean's proof checker. Users should be aware that the code was AI-generated as part of an experiment in AI-assisted formal mathematics. The author provided the mathematical goals, project direction, and verification oversight, and takes full responsibility for the content.

## Contents

[*]Lean 4 formalization available at `https://github.com/AICardiologist/FoundationRelativity`. Code archive: DOI `10.5281/zenodo.17107493`.

## 1 Introduction: Non-reflexivity, Constructivity, and Logical Strength

For a Banach space $X$, the canonical embedding $J_X : X \to X^{**}$ maps $x$ to evaluation at $x$. Classically, many spaces fail to be reflexive (i.e. $J_X$ is not surjective); for example, $\ell^\infty$ is non-reflexive via Hahn–Banach. In Bishop-style constructive mathematics (BISH), however, the status of such facts can depend on additional logical principles. It is therefore natural, in the spirit of constructive reverse mathematics (CRM), to ask for the *precise* logical strength of non-reflexivity statements over BISH. We emphasize that nothing here alters the classical status of these results; the point is to describe their exact logical strength *over BISH* and to record clean proofs with explicit axiom boundaries.

Building on prior work that relates forms of Hahn–Banach to omniscience principles (notably Ishihara's calibration results and subsequent surveys), we prove that, over BISH, the following are equivalent: WLPO; non-surjectivity of $J_{\ell^\infty}$; and the existence of a real Banach space with a bidual gap. Our aim is modest: to make explicit, and to formalize in Lean 4, a calibration that we have not found written down in this exact form, and to present it with careful separation between meta-classical witness extraction and constructive analysis.

**Definition 1.1 (Non-reflexivity and bidual gap)** *A Banach space $X$ is* ***non-reflexive*** *if the canonical embedding $J_X$ is not surjective. In this case, we say $X$* ***has a bidual gap****.*

### 1.1 Related work and positioning

This paper fits into a line of results locating analytic theorems over BISH at specific logical principles.

- **Hahn–Banach and omniscience.** Ishihara [7, 2] and others isolated forms of Hahn–Banach that imply or are equivalent to weak omniscience principles such as WLPO. Bridges and Richman [4] present the broader constructive context; Diener [6] surveys CRM calibrations.
- **Classical reverse mathematics.** Brown–Simpson [5] analyzed set-existence strength behind analytic theorems in the classical base theory. Our use of a two-stage argument (meta extraction, internal analysis) mirrors that methodology in a constructive setting.
- **Non-reflexivity folklore.** That $\ell^\infty$ is non-reflexive is a textbook fact (e.g. [14, 15, 3]). The *logical* content of the *bare non-reflexivity statement* over BISH appears not to have been recorded explicitly; our contribution is to formulate and verify the bi-implication with WLPO. To our knowledge, the equivalence between the bare non-reflexivity statements (for $\ell^\infty$ and "$\exists X$") and WLPOover BISH has not been recorded explicitly; we regard the present contribution as making that calibration precise and verifiable.

Compared with the literature above, our novelty lies in identifying the exact logical strength of *non-surjectivity of $J_{\ell^\infty}$ and the existence of a bidual gap* (rather than particular extension/separation schemes) exactly at WLPO, and in making the meta-classical/constructive separation explicit and machine-checked.

### 1.2 The Weak Limited Principle of Omniscience (WLPO)

**Definition 1.2** (WLPO) *For any binary sequence $\alpha : \mathbb{N} \to \{0,1\}$,*

$$(\forall n,\ \alpha(n) = 0) \quad \vee \quad \neg(\forall n,\ \alpha(n) = 0).$$

WLPO is strictly weaker than LEM but not provable in BISH. It captures the minimal decision strength needed to determine whether an infinite sequence is identically zero.

### 1.3 Main result and contributions

[Main Theorem] Over BISH, the following are equivalent:

1. WLPO.
2. **Gap$_{\ell^\infty}$:** The embedding $J_{\ell^\infty} : \ell^\infty \to (\ell^\infty)^{**}$ is not surjective.
3. **Gap$_\exists$:** There exists a real Banach space $X$ such that $J_X : X \to X^{**}$ is not surjective.

Moreover, this equivalence is fully formalized in Lean 4 (see Section 4).
*Scope.* Throughout, "over BISH" means that the implication is derivable in BISH; the meta-proof certifying the derivation may use classical reasoning (see Remark 2.2).

Our contributions are:

1. **Exact location over BISH.** Building on Ishihara's connections between Hahn–Banach and omniscience principles, we show that the *bare bidual-gap statements* (for $\ell^\infty$ and $\exists X$) are each equivalent to WLPO.

2. **Formal verification.** A Lean 4 development makes explicit the separation between meta-classical extraction and constructive analysis, and provides an axiom audit for the consumer lemma.

3. **Auxiliary tools.** We package a Prop-level kernel interface, a csSup treatment of partial sums suited to BISH, and a constructively careful "Stone window" for $\ell^\infty/c_0$ used as a tool rather than a headline result.

## 2 Main Mathematical Results: The Equivalence Theorem

We explain the equivalence between the bidual gap and WLPO.

### 2.1 Forward direction: Gap implies WLPO

To bridge a bidual-gap witness and a logical decision, we use an "Ishihara kernel", adapted from constructive reverse mathematics.

**Definition 2.1 (Ishihara kernel)** *An* Ishihara kernel *is a specific* CRM witness/test object *with a sharp dichotomy, designed so that purely constructive analysis of its properties decides* WLPO*. It consists of a normed space $X$, an element $y \in X^{**} \setminus J(X)$, a functional $f \in X^*$, a family $g : (\mathbb{N} \to \{0,1\}) \to X^*$, and a constant $\delta > 0$ such that for all binary sequences $\alpha$:*

1. *$|y(f + g(\alpha))| = 0$ or $|y(f + g(\alpha))| \geq \delta$ (dichotomy);*
2. *$(\forall n,\ \alpha(n) = 0) \ \Leftrightarrow\ y(f + g(\alpha)) = 0$ (decision property).*

[Kernel $\Rightarrow$ WLPO] If an Ishihara kernel exists, then WLPO holds.

Given $\alpha$, compute $s = |y(f + g(\alpha))|$. By dichotomy, either $s = 0$ or $s \geq \delta > 0$. The decision property equates $s = 0$ with $(\forall n, \alpha(n) = 0)$, deciding the WLPO instance.

[Gap implies WLPO (meta-classical)] Working in a classical meta-theory: if some Banach space $X$ has $J : X \to X^{**}$ not surjective, then WLPO holds over BISH.

Choose $y \in X^{**} \setminus J(X)$ with $y \neq 0$. A half-norm attainment lemma yields $h \in X^*$ with $\|h\| \leq 1$ and $|y(h)| > \|y\|/2$. Let $f = 0$, $\delta = |y(h)|/2$ (so $\delta > 0$ by $0 \leq \|y\|/2 < |y(h)|$), and

$$g(\alpha) = \begin{cases} 0 & \text{if } \forall n, \alpha(n) = 0, \\ h & \text{otherwise.} \end{cases}$$

Then the dichotomy and decision properties hold, so the kernel exists.

### 2.2 Foundation-theoretic interlude: Classical logic in reverse mathematics

Before proceeding to the reverse direction, we must address a subtle but crucial point about the use of classical logic in constructive reverse mathematics.

[Classical meta-logic vs. object logic] The definition of $g(\alpha)$ in the proof case-splits on the undecidable proposition $(\forall n, \alpha(n) = 0)$. In BISH, we cannot constructively perform this case split. However, in reverse mathematics, we work in a *classical meta-logic* to prove statements *about* BISH.

Concretely, we prove the implication:

(classically) [there exists a Banach space with a bidual gap (a BISH-provable statement)] $\implies$ [BISH $\vdash$ WLPO

This is the standard proof strategy in Constructive Reverse Mathematics (CRM), which follows a two-part structure: *meta-classical witness extraction* followed by *constructive analysis of the witness.* First, operating in a meta-theory, we use non-constructive reasoning to extract a concrete witness—in our case, an Ishihara kernel—from a classical theorem. Second, we analyze this witness using only the logic of the constructive base theory (BISH) to derive a non-constructive principle like WLPO.

In the formalization, we fence the meta-classical reasoning (witness extraction and the case split in $g$) inside a dedicated block, and keep the kernel analysis (*Ishihara kernel* $\Rightarrow$ WLPO) intuitionistic. No classical axiom is added to the object theory. This approach follows Ishihara's original pattern for extracting logical principles from functional analysis results [7].

[Terminology: *witness/test object* in CRM] In constructive reverse mathematics (CRM), the standard phrasing for the two-stage pattern used here is: *(meta-classical) witness extraction* followed by *(constructive) analysis of the witness.* The specific witness we use is an *Ishihara-style* test object with a sharp dichotomy $|y(f+g(\alpha))| = 0 \ \vee\ |y(f+g(\alpha))| \geq \delta$; this is a well-known pattern for calibrating results at WLPO (see Ishihara [7] and the survey [6]).[1] A closely related separation of concerns is common in classical reverse mathematics, e.g. the work of Brown–Simpson [5] on the set-existence strength behind analytic theorems.

**Special note (differences from an earlier draft).** Two technical changes make the argument robust and CRM-compliant:

1. **Gap parameter.** We set
$$\delta \ := \ \frac{|y(h^\star)|}{2},$$
where $h^\star \in X^*$ satisfies $\|h^\star\| \leq 1$ and $\frac{\|y\|}{2} < |y(h^\star)|$. This yields $\delta > 0$ by elementary order facts ($0 \leq \frac{\|y\|}{2} < |y(h^\star)|$ implies $0 < |y(h^\star)|$, hence $0 < \delta$). In particular, we do *not* need to derive $\|y\| > 0$ from $y \neq 0$, which would otherwise force norm-positivity lemmas at the bidual type and introduce foundational friction in a constructive setting.

2. **Fencing classical reasoning.** The only classical steps are: picking $y \in X^{**} \backslash j(X)$, obtaining $h^\star$, and defining $g(\alpha)$ by a global case split. The kernel consumer (from the separation statement and the zero-characterization to WLPO) is intuitionistic and does not depend on classical principles.

Both changes are invisible at the level of classical mathematics, but they are important for constructive reverse mathematics: the consumer no longer relies on bidual-specific norm facts or undecidable case splits.

## 2.3 Reverse direction: WLPO implies gap

[WLPO implies Gap] If WLPO holds, then $J : \ell^\infty \to (\ell^\infty)^{**}$ is not surjective.

[Proof sketch] We reduce a generic instance of WLPO to separation in $\ell^\infty/c_0$ by a uniform *coding* of binary sequences. Given $\alpha \in \{0,1\}^{\mathbb{N}}$, define a bounded sequence $v^\alpha$ whose membership in $c_0$ is equivalent to $(\forall n, \alpha(n) = 0)$ (e.g. by sparse "pulses" placed far apart, or via a bounded alternating prefix with a single switch triggered by the first 1). Then WLPO decides whether $v^\alpha \in c_0$. Using these codes one constructs an $\mathbb{R}$-linear functional $\Phi : \ell^\infty \to \mathbb{R}$ with $\Phi|_{c_0} = 0$ and $\Phi(v^\alpha) \neq 0$ for

[1] In an earlier draft we informally called this a "producer/consumer" split. We keep that metaphor out of the formal presentation and use the standard CRM terms *witness extraction* and *constructive analysis.*

some $\alpha$. The induced functional on the quotient $\ell^\infty/c_0$ yields $G \in (\ell^\infty)^{**}$ with $G \notin J_{\ell^\infty}(\ell^\infty)$, giving the bidual gap.

### 2.4 A structural byproduct

[Gap structure] Assuming WLPO, the quotient $\ell^\infty/c_0$ is nontrivial and carries a Boolean algebra of idempotents with rich structure.

This connects naturally with the "Stone window" theorem in Section 5.

## 3 Constructive Algorithms: Finite Approximations

Finite-dimensional surrogates clarify the constructive boundary: they are computable and illuminating, but their infinite limit would decide WLPO-type predicates.

### 3.1 Cesàro mean surrogates

**Definition 3.1 (Cesàro mean)** *For $n \geq 1$, define $f_n : \mathbb{R}^n \to \mathbb{R}$ by $f_n(x) = \frac{1}{n}\sum_{i=1}^n x_i$.*

[Constructive finite Hahn–Banach]With the sup norm on $\mathbb{R}^n$, the average $f_n(x) = \frac{1}{n}\sum_{i=1}^n x_i$ is the unique linear functional of norm 1 satisfying $f_n(1,\dots,1) = 1$ and $\ker(f_n) = M_n := \{x \in \mathbb{R}^n : \sum_{i=1}^n x_i = 0\}$.

If $\|x\|_\infty \leq 1$, then $|f_n(x)| \leq \frac{1}{n}\sum_{i=1}^n |x_i| \leq 1$, with equality at $x = (1,\dots,1)$, so $\|f_n\| = 1$. The kernel claim is immediate from linearity and the definition of $M_n$; uniqueness follows because any linear functional with this kernel and value on $(1,\dots,1)$ must agree with $f_n$ on a direct-sum decomposition $\mathbb{R}^n = M_n \oplus \operatorname{span}\{(1,\dots,1)\}$.

### 3.2 Computational implementation

We provide $O(n)$ implementations (see the repository) to verify Theorem 3.1 numerically.

### 3.3 Convergence and the constructive boundary

[Non-constructive limit] Viewing $(f_n)$ as functionals on $\ell^\infty$ via projection, any constructive pointwise limit that separates the encodings described below would decide a WLPO-level predicate, and hence does not exist in BISH.

[Proof sketch] Encode a WLPO instance $\alpha$ by $v^\alpha \in \ell^\infty$; in a simple restricted case (at most one 1) let $v_n^\alpha = (-1)^{\sum_{k \leq n} \alpha_k}$. Then $f_n(v^\alpha)$ converges to different limits depending on whether $\alpha$ is all zeros or has a single 1. A constructive limit would decide that dichotomy.

## 4 Formalization in Lean 4

### 4.1 Overview and axiom profile

Our Lean 4 formalization contains about 4,500 lines of verified code.

- **Bidirectional proof:** both Gap $\Rightarrow$ WLPO and WLPO $\Rightarrow$ Gap.
- **Auxiliaries:** near-constructive development of $(c_0)^* \cong \ell^1$ and the Stone window.
- **Axioms:**

```
#print axioms gap_equiv_wlpo
-- [propext, Classical.choice, Quot.sound]
```

Here, `Quot.sound` underlies quotients like $\ell^\infty/c_0$; `propext` is used extensively at Prop level; and `Classical.choice` (with `open Classical`) provides classical case splits in the meta-logic, as discussed in Remark 2.2.

## 4.2 CRM methodology and axiom hygiene

Following standard CRM (Constructive Reverse Mathematics) methodology, our formalization separates the *classical producer* from the *constructive consumer*:

- **Meta-classical witness extraction:** The construction of the Ishihara kernel from the bidual gap, including extraction of $y \in X^{**} \setminus J(X)$, finding $h^\star$ with $\|y\|/2 < \|y(h^\star)\|$, and defining $g(\alpha)$ via case-splitting on the undecidable predicate $(\forall n, \alpha(n) = 0)$. This is fenced in `section ClassicalMeta` in `Ishihara.lean`.
- **Constructive analysis of the witness:** The theorem `WLPO_of_kernel` that derives WLPO from the abstract kernel properties. This is placed *outside* any `noncomputable` or `Classical` sections, ensuring pure intuitionistic reasoning.

This separation can be mechanically verified in Lean:

```
#print axioms WLPO_of_kernel
-- (no classical axioms)

#print axioms WLPO_of_gap
-- [propext, Classical.choice, Quot.sound]
```

The result properly demonstrates that in a classical meta-theory, $\mathsf{BISH} \vdash \mathsf{BidualGapStrong} \Rightarrow$ WLPO.

This mechanical separation is not mathematically deep, but it reduces the risk of inadvertently importing classical reasoning into the constructive core.

## 4.3 Reproducibility information

**Reproducibility Box**

- **Repository**: https://github.com/AICardiologist/FoundationRelativity
- **Lean toolchain**: `leanprover/lean4:v4.22.0-rc4`
- **mathlib4 commit**: `59e4fba0c656457728c559a7d280903732a6d9d1`
- **Project tag**: `p2-crm-v0.2` / commit `85f69aa`
- **Build**: `lake exe cache get && lake build`
  top-level target: `Papers.P2_BidualGap.gap_equiv_wlpo`
- **Code Archive DOI**: `10.5281/zenodo.17107493`
- **Status**: The main equivalence builds with 0 errors. Three WLPO-conditional lemmas remain in the optional module `DualIsometriesComplete.lean` concerning completeness of certain dual spaces; they do not affect the main equivalence.

- **AI Generation**: See the front-page disclosure regarding the AI-generated nature of this formalization.

## 4.4 Key technical solutions

**Prop-level kernel technique.** We implement the Ishihara kernel entirely at Prop level.

```
structure IshiharaKernel where
  -- fields omitted
  kernel_property : ∀ α, |y (f + g α)| = 0 ∨ δ ≤ |y (f + g α)|
  decision_property : ∀ α, (∀ n, α n = false) ↔ y (f + g α) = 0

lemma WLPO_of_kernel (K : IshiharaKernel) : WLPO := by
  intro α
  cases K.kernel_property α with
  | inl h => left; exact (K.decision_property α).mpr h
  | inr h => right; exact mt (K.decision_property α).mp (ne_of_gt h)
```

Listing 1: Ishihara kernel (illustrative Lean snippet)

**Robust csSup for partial sums.** We avoid fragile complete-lattice instance resolution by working directly with conditional suprema:

```
private lemma tsum_eq_csSup_sum_of_nonneg
  {ι : Type*} (u : ι → ℝ) (h0 : ∀ i, 0 ≤ u i) (hs : Summable u) :
  (∑' i, u i) = sSup (Set.range (fun s : Finset ι => ∑ i ∈ s, u i)) := by
  have nonempty : (Set.range _).Nonempty :=   0  , ∅, by  s i m p
  have bdd : BddAbove (Set.range _) := by
    use ∑' i, u i
    rintro x   s  ,  r f l
    exact sum_le_tsum s (fun i _ => h0 i) hs
  apply le_antisymm
  · apply tsum_le_of_sum_le hs; intro s; exact le_csSup bdd   s  ,  r f l
  · apply csSup_le nonempty; rintro x   s  ,  r f l
    exact sum_le_tsum s (fun i _ => h0 i) hs
```

Listing 2: tsum equals csSup of finite partial sums

**HasWLPO architecture.** A lightweight typeclass separates WLPO-dependent arguments from constructive cores.

```
class HasWLPO : Prop :=
  (wlpo : ∀ (α : ℕ → Bool), (∀ n, α n = false) ∨ ¬(∀ n, α n = false))

lemma gap_exists_of_WLPO [HasWLPO] : ...
instance [Classical] : HasWLPO :=   fun   α => em  _
```

Listing 3: WLPO typeclass sketch

## 4.5 The bidirectional theorem (Lean)

```
theorem gap_equiv_wlpo : BidualGapStrong.{0} ↔ WLPO := by
  constructor
  · intro h_gap
    classical
    -- construct kernel from gap witness and apply WLPO_of_kernel
    exact WLPO_of_kernel (kernel_from_gap h_gap)
  · intro hwlpo
    -- construct gap from HasWLPO instance
    exact gap_from_WLPO hwlpo
```

Listing 4: WLPO Gap (top-level equivalence)

## 5 Stone Window: The Boolean Algebra Connection

We relate logic and analysis via a concrete Boolean algebra in the quotient.

### 5.1 Almost-equality and idempotents

**Definition 5.1 (Almost equality)** *For $A, B \subseteq \mathbb{N}$, write $A \sim B$ if the symmetric difference $A \triangle B$ is finite.*

**Definition 5.2** *Write* $\mathrm{Idem}(B)$ *for the set of idempotents of a (commutative) Banach algebra $B$ under the induced Boolean operations $e \wedge f := ef$, $e \vee f := e + f - ef$, and $\neg e := 1 - e$.*

[Stone window]Equip $\ell^\infty/c_0$ with the quotient Banach algebra structure induced by pointwise operations. The map $\Phi : \mathcal{P}(\mathbb{N})/{\sim} \to \mathrm{Idem}(\ell^\infty/c_0)$, $[A] \mapsto [\chi_A]$, is a Boolean algebra isomorphism.

[Proof sketch] (1) $[\chi_A] = [\chi_B]$ iff $\chi_A - \chi_B \in c_0$, i.e. $A \sim B$. (2) Boolean operations are respected. (3) Every idempotent is equivalent (mod $c_0$) to an indicator via thresholding at $1/2$.

[Finite distributive lattices]Every finite distributive lattice embeds into $\mathrm{Idem}(\ell^\infty/c_0)$.

## 6 Significance and Conclusion

### 6.1 Precise logical calibration

The result places a familiar analytic phenomenon exactly at WLPO, demonstrating how constructive reverse mathematics can sharpen vague "non-constructive" labels into precise equivalences.

### 6.2 Foundation relativity

| Foundation | Statement about the gap ($\exists y \notin \mathrm{im}(J)$) | Comment |
|---|---|---|
| ZFC | *Holds* | Hahn–Banach yields witnesses |
| BISH | *Equivalent to* WLPO | (Main Theorem) |
| BISH + DC | *Equivalent to* WLPO | DC does not imply WLPO |
| BISH + WLPO | *Holds* | by equivalence |
| BISH + ¬WLPO | *Not provable* | fails in models of BISH + ¬WLPO |

### 6.3 The role of formalization

The Lean development clarified axiom usage, suggested robust approaches to conditional suprema, and yielded reusable patterns (Ishihara kernel, HasWLPO) for future formalized reverse mathematics.

### 6.4 Conclusion

We establish and formalize that detecting the bidual gap has exactly the logical strength of WLPO over BISH. Finite constructive surrogates illuminate the obstruction at the infinite limit. The approach integrates constructive analysis, reverse mathematics, and mechanized verification. We do not address stronger choice principles, nor do we attempt to calibrate other classical non-reflexivity phenomena; our focus here is solely the bidual-gap statements above.

## Acknowledgments

We are indebted to the constructive analysis and reverse mathematics communities for the conceptual groundwork: in particular E. Bishop, D. S. Bridges, F. Richman, and H. Ishihara, and to surveys such as Diener's overview of constructive reverse mathematics. Discussions in the Lean and mathlib communities shaped the formalization style. Any remaining mistakes are the author's alone.

**AI note.** The Lean formalization was AI-generated; see the front-page disclosure for details. All results have been machine-checked in Lean 4.

## References

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

[2] H. Ishihara. Reverse mathematics in Bishop's constructive mathematics. *Philosophia Scientiae*, Cahier Spécial 6:43–59, 2006.

[3] F. Albiac and N. J. Kalton. *Topics in Banach Space Theory.* Springer, 2nd edition, 2016.

[4] D. S. Bridges and F. Richman. *Varieties of Constructive Mathematics.* Cambridge University Press, 1987.

[5] D. K. Brown and S. G. Simpson. Which set existence axioms are needed to prove the separable Hahn–Banach theorem? *Annals of Pure and Applied Logic*, 31(2–3):123–144, 1986.

[6] H. Diener. Constructive Reverse Mathematics. *arXiv:1804.05495*, 2018.

[7] H. Ishihara. An omniscience principle, the König lemma and the Hahn–Banach theorem. *Mathematical Logic Quarterly*, 36(3):237–240, 1990.

[8] The Lean 4 theorem prover. https://leanprover.github.io/

[9] The mathlib4 mathematical library. https://github.com/leanprover-community/mathlib4

[10] A. Pous and S. Zdancewic. A linear/producer/consumer model of classical linear logic. In *Proceedings of the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10. IEEE, 2014.

[11] D. S. Bridges and L. Vîţă. *Techniques of Constructive Analysis.* Springer, 2006.

[12] M. Beeson. *Foundations of Constructive Mathematics.* Springer, 1985.

[13] A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics, Vols. I–II.* North-Holland, 1988.

[14] J. B. Conway. *A Course in Functional Analysis.* Springer, 2nd ed., 1990.

[15] R. E. Megginson. *An Introduction to Banach Space Theory.* Springer, 1998.

## A Selected Lean Snippets

This appendix presents key excerpts from our Lean 4 formalization, demonstrating the witness extraction/analysis architecture and axiom hygiene discussed in the main text. These snippets are illustrative but align with the actual implementation available at https://github.com/AICardiologist/FoundationRelativity.

### A.1 Constructive analysis (witness ⇒ WLPO)

The following shows the purely constructive analysis that derives WLPO from an Ishihara kernel witness without using classical axioms:

```
/-! Constructive analysis: no ‘open Classical‘, no ‘noncomputable‘. -/
structure IshiharaKernel (X : Type _) [NormedAddCommGroup X] [NormedSpace ℝ X]
    where
  y   : (X →L[ℝ] ℝ) →L[ℝ] ℝ
  f   : X →L[ℝ] ℝ
  g   : (ℕ → Bool) → (X →L[ℝ] ℝ)
  δ   : ℝ
  δpos : 0 < δ
  sep  : ∀ α, |y (f + g α)| = 0 ∨ δ ≤ |y (f + g α)|
  zero_iff_allFalse : ∀ α, (∀ n, α n = false) ↔ y (f + g α) = 0

theorem WLPO_of_kernel
  {X : Type _} [NormedAddCommGroup X] [NormedSpace ℝ X]
  (K : IshiharaKernel X) : WLPO := by
  intro α
  rcases K.sep α with h0 | hpos
  · have : K.y (K.f + K.g α) = 0 := (abs_eq_zero.mp h0)
    exact Or.inl ((K.zero_iff_allFalse α).mpr this)
  · have pos : 0 < |K.y (K.f + K.g α)| := lt_of_lt_of_le K.δpos hpos
    have hne : K.y (K.f + K.g α) ≠ 0 := by
      intro hz; have : |K.y (K.f + K.g α)| = 0 := by simp [hz]
      exact (ne_of_gt pos) this
    have : ¬ (∀ n, α n = false) := by
      intro hall
      have hz : K.y (K.f + K.g α) = 0 := (K.zero_iff_allFalse α).mp hall
      exact hne hz
    exact Or.inr this
```

Listing 5: Constructive analysis of a witness: no classical axioms

### A.2 Meta-classical witness extraction (Gap ⇒ witness)

The meta-classical witness extraction obtains an Ishihara kernel from a bidual gap. Note the explicit `noncomputable` and `open Classical` declarations:

```
/-! Meta-classical witness extraction: fenced. Extract a kernel from a bidual gap.
    -/
noncomputable section
section ClassicalMeta
open Classical

lemma exists_on_unitBall_gt_half_opNorm
  {E} [NormedAddCommGroup E] [NormedSpace ℝ E]
  (T : E →L[ℝ] ℝ) (hT : T ≠ 0) :
  ∃ x : E,    x    ≤ 1 ∧ (   T    / 2) <   T    x    := by
  by_contra h; push_neg at h
  have bound_all : ∀ x,   T    x    ≤ (   T    / 2) *    x    := by
    intro x
    by_cases hx : x = 0
    · simp [hx]
    · have hxpos : 0 <    x    := norm_pos_iff.mpr hx
      let u : E := (   x    )          x
      have hu_le :    u    ≤ 1 := by
        field_simp; rw [norm_smul, Real.norm_of_nonneg (inv_nonneg.mpr (le_of_lt
   hxpos))]
        simp [ne_of_gt hxpos]
      have hu_ball :   T    u    ≤    T    / 2 := h u hu_le
      calc   T    x    =    x    *   T    u        := by rw [← T.map_smul]; simp [u]
                  _ ≤ (   T    / 2) *    x    := mul_le_mul_of_nonneg_left hu_ball (
   norm_nonneg x)
  have hle :    T    ≤    T    / 2 := by
    apply ContinuousLinearMap.opNorm_le_bound
    · exact div_nonneg (norm_nonneg T) (by norm_num)
    · intro x; exact bound_all x
  have :    T    = 0 := by linarith
  exact hT (ContinuousLinearMap.norm_eq_zero.mp this)
```

Listing 6: Meta-classical witness extraction from a bidual gap

## A.3 Gap ⇒ WLPO (meta-classical bridge)

This shows the complete meta-classical construction:

```
/-- Meta-classical bridge: from a bidual gap, extract a witness and apply the
    constructive analysis. -/
theorem WLPO_of_gap (hGap : BidualGapStrong) : WLPO := by
  classical
  rcases hGap with   X  , Xng, Xns, Xc, _dualBan, _bidualBan, hNotSurj
  letI : NormedAddCommGroup X := Xng
  letI : NormedSpace ℝ X := Xns
  letI : CompleteSpace X := Xc
  let j := NormedSpace.inclusionInDoubleDual ℝ X
  -- Extract witness y ∉ range(j)
  have : ∃ y : (X →L[ℝ] ℝ) →L[ℝ] ℝ, y ∉ Set.range j := by
    have : ¬ (∀ y, y ∈ Set.range j) := by
      simpa [Function.Surjective, Set.mem_range] using hNotSurj
    push_neg at this
    exact this
  rcases this with   y  , hy
  have hy0 : y ≠ 0 := by intro hz; subst hz; exact hy   0  , by simp
  -- Find h* with   y  h*    >    y   /2
  obtain    h   , -, hbig  := exists_on_unitBall_gt_half_opNorm y hy0
  let δ : ℝ :=   y     h     / 2
  have δpos : 0 < δ := by
```

```
    have : 0 <   y     h      := lt_of_le_of_lt (div_nonneg (norm_nonneg y) (by
   norm_num)) hbig
    exact half_pos this
  -- Build kernel components
  let f : X →L[ℝ] ℝ := 0
  let g : (ℕ → Bool) → (X →L[ℝ] ℝ) := fun α => if (∀ n, α n = false) then 0 else
    h
  have sep : ∀ α, |y (f + g α)| = 0 ∨ δ ≤ |y (f + g α)| := by
    intro α
    by_cases hall : ∀ n, α n = false
    · left; simp [f, g, hall]
    · right
      have : δ ≤   y     h      := by
        have hnn : 0 ≤   y     h      := norm_nonneg _
        simpa [δ] using half_le_self hnn
      -- unfold to |y(f+g α)| and rewrite     ·     = |·|
      simpa [f, g, hall, zero_add, Real.norm_eq_abs]
  have ziff : ∀ α, (∀ n, α n = false) ↔ y (f + g α) = 0 := by
    intro α; constructor
    · intro hall; simp [f, g, hall]
    · intro hz; by_contra hall
      have : y (f + g α) = y  h    := by simp [f, g, hall]
      rw [this] at hz; norm_num at hz
      exact absurd hz (ne_of_gt (by exact δpos))
  exact WLPO_of_kernel
    { y := y, f := f, g := g, δ := δ, δpos := δpos,
      sep := sep, zero_iff_allFalse := ziff }
end ClassicalMeta
```

Listing 7: Meta-classical: from bidual gap to WLPO

## A.4 WLPO → Gap (coding-based sketch)

The reverse direction uses WLPO to construct a gap via sequence coding:

```
/-- Abbreviation: a "code" turning α into a bounded sequence v^α. -/
def codedSeq (α : ℕ → Bool) : ℕ → ℝ :=
  fun n => if (∃ k ≤ n, α k = true) then (1 : ℝ) else 0

lemma codedSeq_in_c0_iff_allFalse (α : ℕ → Bool) :
  (Filter.Tendsto (codedSeq α) Filter.atTop (nhds 0)) ↔ (∀ n, α n = false) := by
  constructor
  · intro htend hall
    -- If α has a true value, sequence stays at 1 eventually
    rcases hall with  m  ,  h m
    have : ∀ n ≥ m, codedSeq α n = 1 := by
      intro n hn; simp [codedSeq]; use m, le_trans (le_of_eq rfl) hn, hm
    -- This contradicts convergence to 0
    have : Filter.Tendsto (codedSeq α) Filter.atTop (nhds 1) := by
      apply tendsto_atTop_of_eventually_const
      use m; intros; apply this; assumption
    exact absurd (tendsto_nhds_unique htend this) (by norm_num)
  · intro hall
    -- If all false, sequence is constantly 0
    have : ∀ n, codedSeq α n = 0 := by
      intro n; simp [codedSeq]
      intro k hk hkt; exact hall k hkt
    simp [this]; exact tendsto_const_nhds

```

```
/-- Uses WLPO to separate coded sequences in    ∞/ c    -/
theorem gap_from_WLPO : WLPO → BidualGapStrong := by
  intro hwlpo
  -- Construction uses the coding to build a functional on    ∞/ c
  -- that separates some codedSeq α from  c
  -- Full proof in Papers.P2_BidualGap.HB.WLPO_to_Gap_pure
  sorry  -- See repository for complete construction
```
Listing 8: WLPO implies gap via coding

## A.5 OpNorm core (classical LUB)

The operator norm construction shows the classical use of suprema:

```
/-- Classical: existence of operator norm via sSup of values on unit ball -/
namespace OpNorm
variable {X : Type*} [NormedAddCommGroup X] [NormedSpace ℝ X]

def valueSet (h : X →L[ℝ] ℝ) : Set ℝ :=
  { r | ∃ x,     x     ≤ 1 ∧ r =   h    x    }

lemma valueSet_nonempty (h : X →L[ℝ] ℝ) : (valueSet h).Nonempty :=
    0  , 0, by  simp

lemma valueSet_bddAbove (h : X →L[ℝ] ℝ) : BddAbove (valueSet h) :=
      h     , by rintro r   x  , hx,  rfl  ; exact h.le_opNorm_of_le  h x

def HasOpNorm (h : X →L[ℝ] ℝ) : Prop :=
  ∃ N, IsLUB (valueSet h) N

lemma hasOpNorm_classical (h : X →L[ℝ] ℝ) : HasOpNorm h := by
  classical
  use sSup (valueSet h)
  exact isLUB_csSup (valueSet_nonempty h) (valueSet_bddAbove h)
end OpNorm
```
Listing 9: Classical operator norm via supremum

## A.6 Stone window (idempotents in the quotient)

The Stone window construction shows the Boolean algebra structure:

```
/-- Almost-equality of subsets of ℕ -/
def AlmostEq (A B : Set ℕ) : Prop := (A \ B ∪ B \ A).Finite

/-- Indicator function in    ∞ -/
def chi (A : Set ℕ) : ℕ → ℝ := fun n => if n ∈ A then 1 else 0

/-- The quotient    ∞/ c    has a Banach algebra structure -/
instance : Algebra ℝ (   ∞       c   ) where
  -- Multiplication is pointwise, well-defined modulo   c
  mul := Quotient.map   (fun f g n => f n * g n) (by sorry)
  one := Quotient.mk (fun _ => 1)
  -- Other fields omitted

/-- Boolean algebra of idempotents -/
def Idem (B : Type*) [Mul B] := { e : B // e * e = e }

```

```
instance [Mul B] [One B] [Add B] [Sub B] : BooleanAlgebra (Idem B) where
  inf e f :=  e .val * f.val, by simp [e.property, f.property]
  sup e f :=  e .val + f.val - e.val * f.val, by ring_nf; simp [e.property, f.
    property]
  compl e :=  1  - e.val, by ring_nf; simp [e.property]
  -- Other fields follow from these operations

/-- Stone window isomorphism -/
def StoneWindow : (Set ℕ)    AlmostEq     Idem (  ∞      c  ) where
  toFun := Quotient.map (fun A =>   Quotient  .mk (chi A), by s o r r y ) (by sorry)
  invFun := sorry  -- Inverse via support extraction
  left_inv := by sorry
  right_inv := by sorry
```

Listing 10: Stone window: Boolean algebra of idempotents

### A.7 Axiom hygiene verification

Finally, we can verify the axiom usage as discussed in Section 4.1:

```
#print axioms WLPO_of_kernel
-- (no axioms)

#print axioms WLPO_of_gap
-- [propext, Classical.choice, Quot.sound]

#print axioms gap_from_WLPO
-- [propext, Quot.sound]  -- uses WLPO hypothesis, not Classical.choice

#print axioms gap_equiv_wlpo
-- [propext, Classical.choice, Quot.sound]
```

Listing 11: Axiom audit commands

The output confirms that the constructive consumer (`WLPO_of_kernel`) uses no classical axioms, while the meta-classical producer (`WLPO_of_gap`) requires classical choice. This validates our CRM methodology claim that the consumer remains purely constructive.

## B Code–Theorem Correspondence for the Direct Dual Construction

This appendix explains how the Lean file

`Papers/P2_BidualGap/HB/DirectDual.lean`

formalizes the standard "direct construction" used in the reverse direction (WLPO $\Rightarrow$ Gap) and in the classical non–reflexivity argument for $c_0$. Concretely, the file builds a bounded linear functional

$$G : (c_0^*) \to \mathbb{R} \qquad \text{via} \qquad G(f) \;=\; \sum_{n=0}^{\infty} f(e_n),$$

and proves that $G(\delta_m) = 1$ for all $m$. From this, one derives that $G$ cannot be represented by evaluation at any $x \in c_0$, whence the canonical embedding $J_{c_0} : c_0 \to c_0^{**}$ is not surjective. The last step is carried out in the file that imports this construction (see remark at the end of this appendix).

## A. The target theorem (mathematical form)

**Theorem (Direct dual witness for non-surjectivity of $J_{c_0}$).** Define $G : (c_0^*) \to \mathbb{R}$ by $G(f) = \sum_n f(e_n)$, where $e_n \in c_0$ is the $n$-th standard basis vector and the series is absolutely convergent. Then for every $m$,

$$G(\delta_m) = 1.$$

Consequently, if $G = J_{c_0}(x)$ for some $x \in c_0$, then $x_m = G(\delta_m) = 1$ for all $m$, contradicting $x \in c_0$. Hence $G \notin \operatorname{im}(J_{c_0})$ and $J_{c_0}$ is not surjective.

This is the classical "basis/coordinate" argument: the values of $G$ on the coordinate functionals force the representing vector to be the constant sequence $(1, 1, \dots)$, which is not in $c_0$.

## B. Modeling choices and core definitions (Lean ↔ mathematics)

We model $c_0$ over the discrete space $\mathbb{N}$ using the standard mathlib type:

| **Lean artifact** | **Mathematical meaning** |
|---|---|
| `abbrev c := ZeroAtInftyContinuousMap` | $c_0 = \{x : \mathbb{N} \to \mathbb{R} : x_n \to 0\}$ with $\|\cdot\|_\infty$ |
| `def e (n : ) : c := ...` | Basis vector $e_n$ (1 at $n$, 0 elsewhere); tends to 0 outside the singleton $\{n\}$ |
| `def  (n : ) : c →L[]  := ...` | Coordinate evaluation $\delta_n(x) = x_n$, a bounded linear functional |
| `def coeff ...` | Scalar $\operatorname{coeff}(f, n) = f(e_n)/\|f(e_n)\|$ (or 0 if $f(e_n) = 0$) |
| `def signVector ...` | Finite "sign vector" $\sum_{n \in F} \operatorname{coeff}(f, n)\, e_n \in c_0$ |
| `noncomputable def G ...` | $G(f) = \sum_n f(e_n)$ as a continuous linear functional on $c_0^*$ |

The choice `ZeroAtInftyContinuousMap` is mathlib's canonical $c_0$; it is isometrically the usual Banach space of real sequences tending to 0. The proof uses the built–in sup–norm via the map `toBCF` into `BoundedContinuousFunction`.

## C. The "finite signs" inequality $\sum_{n \in F} \|f(e_n)\| \le \|f\|$

The core analytic estimate is proved exactly as on paper, using "sign vectors":

$$x_F := \sum_{n \in F} \operatorname{coeff}(f, n)\, e_n, \qquad \|x_F\|_\infty \le 1, \qquad f(x_F) = \sum_{n \in F} \|f(e_n)\|.$$

In Lean this is the lemma:

- `lemma finite_sum_bound ... :  nF ‖f (e n)‖  ‖f‖`,

which follows from:

- `lemma abs_coeff_le_one ...` (the sign coefficients have absolute value $\le 1$),
- `lemma signVector_norm_le_one ...` ($\|x_F\| \le 1$),
- `lemma coeff_mul_eval_abs ...` ($\operatorname{coeff}(f, n) \cdot f(e_n) = \|f(e_n)\|$).

The proof path is the textbook argument and matches the handwritten derivation line–by–line.

## D. Absolute summability and definition of $G$

From the finite–sum bound, Lean derives the absolute summability of $n \mapsto f(e_n)$:

- `lemma summable_abs_eval ... : Summable ( n, ‖f (e n)‖)`,

and hence summability of $n \mapsto f(e_n)$:

- `lemma summable_eval ... : Summable ( n, f (e n))`.

With this, the series defining $G$ exists and is used to build a continuous linear map:

- `noncomputable def G : (c →L[] ) →L[]  := ...`,

together with the operator–norm bound

$$\|G(f)\| \;\le\; \sum_n \|f(e_n)\| \;\le\; \|f\|,$$

encoded in the Lipschitz proof obligation for `LinearMap.mkContinuous`.

## E. Computing $G(\delta_m) = 1$

The key computation is:

- `lemma G_delta (m : ) : G ( m) = 1`.

Mathematically, $(\delta_m)(e_n) = \mathbf{1}_{n=m}$, so the series is the sum of a single 1. The Lean proof implements exactly this: it identifies the (finite) support of $n \mapsto \delta_m(e_n)$, reduces the `tsum` to a finite sum over $\{m\}$, and computes it to be 1.

## F. From $G(\delta_m) = 1$ to the non-surjectivity of $J_{c_0}$

The last step is a purely formal consequence. If $G = J_{c_0}(x)$ for some $x \in c_0$, then

$$x_m \;=\; J_{c_0}(x)(\delta_m) \;=\; G(\delta_m) \;=\; 1 \quad \text{for all } m,$$

so $x = (1, 1, \dots) \notin c_0$. Therefore $G \notin \operatorname{im}(J_{c_0})$. In the repository this implication is proven in the file that imports `DirectDual.lean` (see the lemma named along the lines of `c0_not_reflexive_via_direct` in `HB/WLPO_to_Gap_HB.lean`), where the contradiction is discharged using the evaluation identity $J_{c_0}(x)(\delta_m) = x_m$.

*Conclusion.* The sequence of Lean lemmas `finite_sum_bound` $\Rightarrow$ `summable_eval` $\Rightarrow$ definition of `G` $\Rightarrow$ `G_delta` formalizes the standard analytical argument and yields the intended witness $G \in c_0^{**} \setminus \operatorname{im}(J_{c_0})$ once paired with the short evaluation argument above.

## G. Axiom profile and scope

The file begins with `noncomputable section` and `open Classical` because mathlib's infinite–sum API (`tsum`) and some norm inequalities live in the classical namespace. No additional axioms are introduced by the definitions and lemmas in `DirectDual.lean`; the construction uses only:

- the discrete–space model for $c_0$ (`ZeroAtInftyContinuousMap`),
- basic sup–norm bounds via `BoundedContinuousFunction`,

- standard series facts (`Summable`, `tsum`),
- linearity and operator–norm estimates for continuous linear maps.

For reproducibility, an axiom audit can be inspected with:

```
#print axioms Papers.P2.HB.G
#print axioms Papers.P2.HB.G_delta
#print axioms Papers.P2.HB.finite_sum_bound
```

which (on our build) report no extra axioms beyond classical logic used by mathlib's series machinery.

### H. Minimal crosswalk (code ↔ paper)

| **Lean name (this file)** | **Paper step** |
|---|---|
| `abbrev c` | $c_0$ as sequences $\to 0$ with $\|\cdot\|_\infty$ |
| `def e (n)` | Basis vector $e_n$ |
| `def  (n)` | Coordinate functional $\delta_n$ |
| `def coeff`, `def signVector` | Sign–vector device for Hölder–type bound |
| `lemma finite_sum_bound` | $\sum_{n \in F} \|f(e_n)\| \le \|f\|$ |
| `lemma summable_abs_eval`, `summable_eval` | Absolute summability of $n \mapsto f(e_n)$ |
| `noncomputable def G` | $G(f) = \sum_n f(e_n)$, linear and bounded |
| `lemma G_delta` | $G(\delta_m) = 1$ |
| *(importing file)* | Conclude $G \notin \mathrm{im}(J_{c_0})$ and so $J_{c_0}$ not surjective |

**Where the final contradiction is discharged.** The contradiction "$G = J_{c_0}(x) \Rightarrow x = (1, 1, \dots) \notin c_0$" is proved in the file that imports this module (Section 4; see the lemma titled $c_0 isnotreflexiveviadirectconstruction).Thatlemmausesonlythetwofactsexportedhere : thedefinitionofGandG($
1 *for all* $m$.

*This establishes that the Lean development in* `DirectDual.lean` *exactly represents the handwritten theorem and proof structure, with each mathematical step mirrored by a named Lean definition or lemma and no elided reasoning.*

## C Meta-Classical Witness Extraction and the Producer/Consumer Analogy

*This appendix explains the proof architecture in standard Constructive Reverse Mathematics (CRM) terms and clarifies the intended role of the producer/consumer metaphor.*

### A. CRM terminology and proof shape

*In CRM one typically proves implications of the form*

$$\textit{(classically)} \qquad [\mathsf{BISH} \vdash \mathsf{Thm}] \implies [\mathsf{BISH} \vdash \mathsf{Prin}],$$

*where the* meta-proof *is classical, but the* object-level *target theory is* BISH *(intuitionistic). The standard two-stage pattern is:*

1. ***Meta-classical witness extraction.*** *Working in classical logic* about BISH, *use the assumed theorem to* produce *a concrete mathematical* witness *(sometimes called a* test object*) whose existence is provable in* BISH *and whose properties will be sufficient to derive the target principle.*

2. ***Constructive consumption.*** *Still inside* BISH *(no classical axioms),* consume *only the abstract properties of that witness to derive the logical principle.*

*Formally, the meta-argument composes two* BISH*-internal arrows:*

$$\mathsf{BISH} \vdash (\mathsf{Thm} \Rightarrow \exists W\,.\,\mathsf{Adequate}(W)) \quad \textit{and} \quad \mathsf{BISH} \vdash (\mathsf{Adequate}(W) \Rightarrow \mathsf{Prin}).$$

*The first arrow is* proved *meta-classically; the second arrow is proved intuitionistically. (See [2, 6, 7] for surveys and examples of this pattern.)*

## B. The witness schema used in this paper

*In our setting the witness is an* Ishihara-style test object *with a sharp* dichotomy *and a* decision *property:*

- *A Banach-analytic package $W = (y, f, g, \delta)$ such that for each binary sequence $\alpha$,*

$$|y(f + g(\alpha))| = 0 \ \ \textit{or} \ \ \delta \le |y(f + g(\alpha))|$$

$$\textit{and} \qquad (\forall n,\, \alpha(n) = 0) \ \Leftrightarrow \ y(f + g(\alpha)) = 0.$$

- *The consumer (§4, App. B) shows* constructively *that this suffices to decide each WLPO instance $\alpha$.*

*The only logically delicate step is the* extraction *of $W$ from a gap witness in the bidual, where one (i) chooses $y \in X^{**} \setminus J(X)$, (ii) obtains $h^\star \in X^*$ with $|y(h^\star)| > \frac{1}{2}|y|$, and (iii) defines $g(\alpha)$ by a global case split on the undecidable predicate $(\forall n, \alpha(n) = 0)$. As is standard in CRM, that case split happens in the* meta-classical *part, while the consumer uses only the abstract properties of $W$ and remains intuitionistic. Compare [7] for earlier uses of such dichotomy-style witnesses and [6] for a general CRM perspective; see also [5] for related calibrations around Hahn–Banach.*

## C. How this is organized in Lean (axiom hygiene)

*We make this separation* mechanical*:*

- ***Producer (meta-classical block).*** *In the file `CRM_MetaClassical/Ishihara.lean`, the witness $W$ is extracted inside a fenced section*

  ```
  noncomputable section  section ClassicalMeta  open Classical
  ```

  *which licenses classical reasoning needed for choice and global case splits.*

- ***Consumer (constructive block).*** *The theorem `WLPO_of_kernel` (App.B, Listing5) lives* outside *any classical section and uses no classical axioms:*

  ```
  #print axioms WLPO_of_kernel
  -- (no axioms)
  ```

- ***Audit (meta composition).*** *The bridge `WLPO_of_gap` composes the two* BISH*-internal arrows and reports exactly the expected meta-classical footprint:*

  ```
  #print axioms WLPO_of_gap
  -- [propext, Classical.choice, Quot.sound]
  ```

*This confirms the intended CRM architecture: only the extraction uses classical logic; the consumption is purely constructive.*

## D. The producer/consumer analogy—useful, but just an analogy

*The terms "producer" and "consumer" are* not *standard CRM jargon; they are metaphors we use to communicate the separation of concerns:*

- *The producer is the meta-classical routine that* produces *a witness $W$ from the assumed theorem (here, the bidual gap).*
- *The consumer is the constructive routine that* consumes *only the abstract API of $W$ to derive the target principle (here, WLPO).*

*This is an explanatory overlay on the standard witness-extraction pattern in CRM; our paper and code use the precise terms* meta-classical witness extraction *and* constructive consumption *as the primary terminology and "producer/consumer" only as a helpful guide.*

**Relation to linear-logic intuitions (optional).** *There is a useful conceptual parallel to resource sensitivity in linear logic: meta-classical inferences (strong resources) are confined to the extraction phase, while the consumer manipulates only the* type *of $W$ (its API) using constructive rules. We do not rely on any linear-logic machinery; the connection is heuristic and pedagogical.*

## E. How others use the same mechanism

- ***Ishihara (1990):*** *Uses an Ishihara-style dichotomy witness to calibrate variants of Hahn–Banach and omniscience principles, deriving WLPO-level consequences from analytic hypotheses [7].*
- ***Brown–Simpson (1986):*** *Calibrate set-existence axioms needed for separable Hahn–Banach, a closely related program in classical reverse mathematics [5].*
- ***Diener (CRM overview):*** *Describes the general two-stage CRM pattern (extract a witness/test object classically; analyze it constructively) across many examples [6].*

*Our development follows this well-trodden schema: the specific* shape *of $W$ is tailored to the bidual-gap setting (the "Ishihara-style kernel" used here), but the methodology is the standard CRM pattern of witness extraction and constructive consumption.*

### F. Minimal formal schema (for readers comparing paper and code)

***Meta-classical step (witness extraction).***

$$\textit{(classically)}\quad \mathsf{BISH} \vdash \Big(\mathsf{Gap} \;\Rightarrow\; \exists W\,.\,\mathsf{Adequate}(W)\Big)$$

***Constructive step (witness consumption).***

$$\mathsf{BISH} \vdash \Big(\mathsf{Adequate}(W) \;\Rightarrow\; \mathsf{WLPO}\Big)$$

***Composition (CRM conclusion).***

$$\textit{(classically)}\quad \mathsf{BISH} \vdash (\mathsf{Gap} \;\Rightarrow\; \mathsf{WLPO}).$$

*In our formalization, the first arrow is implemented in `Ishihara.lean` (producer block) and the second arrow is `WLPO_of_kernel` (consumer block). App. B explains how the direct dual construction (`DirectDual.lean`) supplies the complementary witness $G$ used for the reverse direction.*

## D WLPO ⇒ Bidual Gap via Direct Dual Witness $G$

*This appendix documents the reverse direction of the equivalence—the direct construction of a non-representable element of $c_0^{**}$—and shows that the Lean code in `Papers/P2_BidualGap/HB/WLPO_to_Gap_HB.lean` faithfully implements the informal proof.*

### A. Mathematical statement and proof sketch

*Over* BISH *with* WLPO *as a hypothesis, we claim that $J_{c_0} : c_0 \to c_0^{**}$ is not surjective.*

***Witness $G$.*** *For $f \in c_0^*$, write $f(e_n)$ for the evaluation of $f$ on the standard basis $e_n \in c_0$ (the function that is 1 at $n$ and 0 elsewhere). Define*

$$G(f) \;:=\; \sum_{n=0}^{\infty} f(e_n).$$

*Two key facts:*

1. *The series $\sum_n f(e_n)$ is absolutely summable (uniform bound on partial sums), hence $G$ is a well-defined bounded linear functional on $c_0^*$, i.e. $G \in (c_0^*)^* = c_0^{**}$.*

2. *For the coordinate functionals $\delta_m \in c_0^*$ (evaluation at $m$), we have $G(\delta_m) = 1$ for all $m$.*

***Contradiction.*** *If $G = J_{c_0}(x)$ for some $x \in c_0$, then $x_m = J_{c_0}(x)(\delta_m) = G(\delta_m) = 1$ for all $m$, so $x$ is the constant sequence 1, which is not in $c_0$ (does not vanish at infinity). Hence $G \notin \mathrm{im}(J_{c_0})$, i.e. $J_{c_0}$ is not surjective.*

### B. Lean ↔ Math crosswalk

*The following table matches the informal proof objects to their Lean definitions.*

| ***Mathematics*** | ***Lean symbol / fact*** | ***Where*** |
|---|---|---|
| *The space $c_0$ on $\mathbb{N}$* | `abbrev c := ZeroAtInftyContinuousMap` | *DirectDual.lean* |
| *Basis $e_n \in c_0$* | `def e (n : ) : c` | *DirectDual.lean* |
| *Coordinate $\delta_m \in c_0^*$* | `def  (m : ) : c →L[]` | *DirectDual.lean* |
| *Witness $G(f) = \sum_n f(e_n)$* | `def G : (c →L[] ) →L[]` | *DirectDual.lean* |
| *Key property $G(\delta_m) = 1$* | `lemma G_delta (m) : G ( m) = 1` | *DirectDual.lean* |
| *Non-surjectivity of $J_{c_0}$* | `lemma c0_not_reflexive_via_direct :` `¬ Surjective (inclusionInDoubleDual  c)` | *WLPO_to_Gap_HB.lean* |
| *Packaging as BidualGapStrong* | `lemma wlpo_implies_gap :` `WLPO → BidualGapStrong.{0}` | *WLPO_to_Gap_HB.lean* |
| *Equivalence* | `theorem gap_equiv_wlpo :` `BidualGapStrong.{0}  WLPO` | *WLPO_to_Gap_HB.lean* |

***Notes on the core steps***

- ***Absolute summability:*** *DirectDual.lean proves* `summable_abs_eval` *and* `summable_eval` *for the sequence $n \mapsto f(e_n)$, then uses* `norm_tsum_le_tsum_norm` *to bound $\|G\|$. This is the formal counterpart of the textbook estimate $\sum_{n\in F} |f(e_n)| \le \|f\|$ for any finite $F$.*
- ***Computing** $G(\delta_m)$**:*** `G_delta` *reduces the tsum to a finite sum over the support, which is just $\{m\}$, and evaluates $(\delta_m)(e_m) = 1$.*
- ***The contradiction:*** *In* `c0_not_reflexive_via_direct`*, assuming surjectivity gives $x$ with $J(x) = G$. Evaluating both sides at $\delta_m$ yields $x_m = 1$ for all $m$, contradicting* `x.zero_at_infty'` *(the vanishing-at-infinity property). This is exactly the informal argument.*

## C. Where WLPO appears (and where it does not)

*The* contradiction *that $G \notin \mathrm{im}(J_{c_0})$ uses only:*

- *the concrete model of $c_0$,*
- *the basis $e_n$, coordinates $\delta_m$,*
- *the definition and summability of $G$, and*
- *the topological fact that elements of $c_0$ tend to 0 at infinity.*

*It does not use any instance of* WLPO. *In* `wlpo_implies_gap` *the hypothesis WLPO is threaded in only to satisfy the packaging requirements of the predicate* `BidualGapStrong` *(see the two declarations below). The core non-representability of G is independent of WLPO.*

## D. On the two auxiliary axioms

*The file currently contains*

```
axiom dual_is_banach_c0_from_WLPO :
  WLPO → DualIsBanach c

axiom dual_is_banach_c0_dual_from_WLPO :
  WLPO → DualIsBanach (c →L[] )
```

*These are* packaging stubs *for the fields that* `BidualGapStrong` *asks for (completeness/Banachness of the space and its duals). They are deliberately placed here to avoid entangling the direct G-construction with library-level completeness proofs. Two remarks:*

1. ***They do not touch the core argument.*** *The contradiction $J_{c_0}(x) = G \Rightarrow x = 1$ is proved in* `c0_not_reflexive_via_direct` without *these axioms.*
2. ***They are dischargeable.*** *In mathlib, one has completeness of $E \to_L F$ once $F$ is complete, and $c_0$ is complete as a closed subspace of bounded continuous functions on a discrete space. Thus these stubs can be replaced by ordinary lemmas/instances (or the* `BidualGapStrong` *record can be relaxed to require only the standard* `CompleteSpace` *instances). We keep them localized here solely to decouple the direct dual argument from auxiliary typeclass plumbing.*

*We highlight this so that readers can see that the formalization mirrors the informal proof structure: the hard analytic content is in DirectDual.lean; the "gap witness packaging" around it is orthogonal.*

## E. Relation to $G = S \circ \Phi_1$ in the text

*Analytically, one often identifies $c_0^* \cong \ell^1$ via the coordinate map $\Phi_1(f) = (f(e_n))_n$ and takes $S$ to be the summation functional on $\ell^1$. Our Lean definition of*

$$G(f) = \sum_n f(e_n)$$

*is exactly the composition $S \circ \Phi_1$ at the level of behavior, but avoids committing to the explicit $c_0^* \cong \ell^1$ isomorphism. This keeps the construction lightweight: we use only the unconditional summability proved in DirectDual.lean.*

## F. CRM positioning (standard terms)

*Unlike the Gap $\Rightarrow$ WLPO direction—where we used a meta-classical* witness extraction *followed by a constructive* consumer*—this WLPO $\Rightarrow$ Gap direction is a direct explicit construction: we build $G \in c_0^{**}$, prove $G \notin J(c_0)$, and then package $c_0$ as the witness space in* `BidualGapStrong`*. The producer/consumer terminology is still a helpful mental model for the other direction, but here the proof is purely "producer-style": we produce the object $G$ outright.*