The Modular Spectrum of π: Theoretical Unification, DSP Isomorphism, and Exascale Validation

Algorithmic Hybridization Architecture in $\mathbb{Z}/6\mathbb{Z}$

José Ignacio Peinador Sala
Independent Researcher
joseignacio.peinador@gmail.com

1 de febrero de 2026

Resumen

This article presents the definitive consolidation of the ”Modular Spectrum”theoretical framework, which proposes a solution to the discontinuity between the discrete nature of integers and the transcendental periodicity of trigonometric functions via the ring $\mathbb{Z}/6\mathbb{Z}$. An isomorphism between this arithmetic structure and the **Polyphase Decomposition** used in Digital Signal Processing (DSP) is demonstrated. The research culminates with the implementation of the ”Modular Hyper-Computer”(Hybrid Stride-6 algorithm), validated experimentally by calculating 10^8 digits of π in a resource-constrained environment (12GB RAM), achieving a parallelization efficiency of 95 % and a sustained speed of 83,000 digits/s. Finally, the spectral rigidity of the Riemann zeros under this modular filter is confirmed.

1. Introduction: The Continuum Crisis

The intersection between Computational Number Theory and Harmonic Analysis has historically sought to reconcile the discrete nature of $\mathbb{Z}$ with continuous wave mechanics. Evaluating functions like $\sin(n)$ for $n \in \mathbb{Z}$ generates apparently chaotic behavior due to the incommensurability between the integer lattice and the transcendental period 2π.

The present work posits that this chaos is apparent and can be reordered via a filter based on the modulus $m = 6$, expanding preliminary findings on prime channel structures presented in earlier works [1]. The choice of this modulus is not arbitrary: it is the product of the first primes (2×3) and defines the hexagonal lattice, allowing it to capture rotational symmetries that other moduli lose.

2. Theoretical Foundations: The $\mathbb{Z}/6\mathbb{Z}$ Filter

2.1. Definition of the Modular Decomposition

Every angle $\theta \in \mathbb{Z}$ admits a unique decomposition based on the division algorithm:

$$\theta = 6k + r, \quad \text{where } r \in \{0,1,2,3,4,5\} \tag{1}$$

This transformation converts the number line into a cylindrical structure $\mathbb{Z} \times \mathbb{Z}/6\mathbb{Z}$. The residues are classified into three physical categories:

- **Null Channels $(0,3)$:** Anchor points ($\gcd(r,6) \neq 1$).
- **Prime Channels $(1,5)$:** High-energy generators ($\gcd(r,6) = 1$).
- **Composite Channels $(2,4)$:** Even harmonics.

2.2. Euler's Identity and Diophantine Approximations

The numerical stability of $e^{i\pi} + 1 = 0$ is analyzed through the modular lens. Given that $\pi \approx 3{,}14159$, the nearest integer is 3. The unit vector in channel $r = 3$ is:

$$e^{i3} = \cos(3) + i\sin(3) \approx -0{,}989 + 0{,}141i \tag{2}$$

This is the integer point closest to -1 in the complex plane, suggesting that Euler's identity leverages a natural "stability attractorïn channel 3.

3. Isomorphism with Digital Signal Processing (DSP)

One of the central contributions of this study is the formalization of the link with *multirate* system theory, as described in the classical filter bank literature [2].

Teorema 3.1 (Polyphase Isomorphism)**.** *The modular decomposition of a hypergeometric series $S = \sum a_k$ is mathematically equivalent to the Polyphase Decomposition of a discrete signal $x[n]$ with decimation factor $M = 6$.*

In the Z-domain, this is expressed as:

$$X(z) = \sum_{r=0}^{5} z^{-r} E_r(z^6) \tag{3}$$

Where $E_r(z)$ are the polyphase components (the modular channels). This validates the robustness of the proposed architecture, aligning it with proven algorithms like the Composite-radix FFT and the Prime Factor Algorithm (PFA).

4. Algorithmic Architecture: "Stride-6" Hybridization

To overcome memory bottlenecks in calculating π via the Chudnovsky series [3], we developed the **Hybrid Stride-6 Binary Splitting** architecture.

4.1. The Monolithic Problem

Traditional algorithms (standard Binary Splitting) build a single, gigantic recursion tree. This saturates memory bandwidth and causes race conditions in multi-core systems when attempting to access the same large data blocks.

4.2. The Modular Solution (Shared-Nothing)

Our implementation decomposes the series into 6 fully independent execution threads (Channels). Each channel processes a sparse sub-series:

$$S_r = \sum_{j=0}^{\infty} t_{6j+r} \tag{4}$$

4.2.1. The "Stride" Transition Leaf

To make this efficient, the fundamental computational unit (the leaf of the recursive tree) does not calculate a single term, but synthesizes the transition matrix for a full block of 6 terms. This allows the algorithm to "stride" through the series, compressing local complexity and reducing tree depth by a factor of $\log_2 6$.

Algorithm 1 Modular Worker Logic (Simplified)

```
1: **Input:** Residue $r$, Range $N$
2: Initialize accumulators $P = 1, Q = 1$
3: **for** $j$ in Local Range **do**
4:     $k_{start} \leftarrow 6j + r$
5:     Calculate Compressed Transition $(P_{leaf}, Q_{leaf}, B_{val})$ for $k \to k+6$
6:     Accumulate in Local Binary Tree
7: **end for**
8: Apply Prefix Correction (Phase)
9: **return** Partial Rational Sum $S_r$
```

5. Experimental Validation: The 100 Million Barrier

The architecture was subjected to an extreme stress test: calculating **100,000,000 digits** of π in a memory-constrained environment (Google Colab, 2 vCPU, 12GB RAM).

5.1. Benchmark Results

The calculation completed successfully, generating a 95 MB file and demonstrating the thermal and memory stability of the design.

Tabla 1: Metrics of the "100M Barrier Run"

Metric	**Recorded Value**
Parallel Phase Time	1085.03 s (18.08 min)
Total Time (with I/O)	1194.32 s (19.90 min)
Average Speed	**83,729 digits/s**
Speedup (vs Sequential)	1,90× (Efficiency 95 %)
Peak RAM Usage	≈ 6,8 GB (Controlled)

5.2. Forensic Validation

The integrity of the final sequence was verified by comparing it with reference databases (y-cruncher records).

Final Sequence (10^8)**:** `...9757220317520748981611683139375159`

The exact match confirms that the modular decomposition introduces no numerical drift even after trillions of arithmetic operations.

6. Implications for the Riemann Zeta Function

The modular filter was applied to the non-trivial zeros of the Riemann Zeta function (γ_n), correcting preliminary hypotheses of bias.

6.1. Confirmed Spectral Rigidity

The expanded analysis revealed a highly uniform distribution ($p \approx 0{,}98$ in χ^2 test) of the zeros modulo 6. This negative result is significant: it confirms the property of **Spectral Rigidity** predicted by the connection with Random Matrices (GUE), extensively studied by Odlyzko [4]. The modular structure serves as a high-sensitivity sieve to verify the ”perfect randomness.^{o}f the zeros.

7. Conclusions and Future: The “Hexa-Core Engine”

The research validates the ”Modular Spectrum of π.^{a}s a mathematically sound and computationally superior framework for parallel architectures.

1. **Isomorphism:** The equivalence between the modular arithmetic of Chudnovsky and polyphase filter banks in DSP has been demonstrated.
2. **Performance:** The *Shared-Nothing* architecture eliminates memory lockouts, enabling massive-scale calculations (10^8 digits) on commodity hardware.

Future Vision: The results suggest that a dedicated hardware implementation (FPGA/ASIC) with 6 physically isolated cores and distributed HBM memory could scale linearly, offering energy efficiency (digits/watt) unattainable by current monolithic architectures.

Acknowledgments

The author wishes to express gratitude to the open-source community, whose collective effort enables the democratization of scientific research outside traditional academic environments.

Infrastructure and Software

This work was made possible thanks to the cloud computing infrastructure provided by **Google Colab**, which facilitated access to the necessary computational resources for running the experiments.

The computational implementation was developed using the **Python** programming language. We thank the developers of the following fundamental libraries for this study:

- **gmpy2** and **decimal**: For arbitrary-precision arithmetic and high-performance operations with large integers (`mpz`) and floats (`Decimal`), essential for the required numerical precision.
- **SciPy** and **NumPy**: For advanced scientific computing, including signal processing tools (`scipy.signal`), special functions (`scipy.special.gamma`), and efficient matrix manipulation.
- **Pandas**: For capabilities in structuring, manipulating, and analyzing tabular data.
- **Matplotlib** and **IPython**: For data visualization, generation of plots, and interactive presentation of mathematical results.
- **Python Standard Library**: Specifically the concurrency modules (`multiprocessing`, `concurrent.futures`) that enabled parallel process execution, as well as the native mathematical functions (`math`, `cmath`) for operations in the real and complex domains.

Artificial Intelligence Assistance

In line with principles of transparency in research, the use of Large Language Model (LLM)-based assistants during the development of this manuscript is declared. These tools were used for bibliographic assistance, style review, and code debugging. The theoretical conceptualization, the mathematical formulation of the modular isomorphism, and the final interpretation of the results are the exclusive responsibility of the human author.

Data and Code Availability

Aiming to foster reproducibility and the advancement of collective knowledge, the complete source code, training scripts, and model weights generated in this research are publicly available in the following repository:

`https://github.com/NachoPeinador/Arquitectura-de-Hibridacion-Algoritmica-en-Z-6Z`

Licensing

The software is distributed under a **dual licensing** model designed to protect the sustainability of independent research while promoting open science:

1. **Academic and Non-Commercial Use**: The source code is available under the **PolyForm Noncommercial License 1.0.0**. This permits its use, modification, and distribution free of charge exclusively for research, education, and non-profit personal projects.

Declaration of Interests

The author declares that this research was conducted independently, without receiving external funding, corporate grants, or institutional sponsorships. The development of the Hex-Engine architecture and the theoretical framework of modular isomorphism presents no financial or commercial conflicts of interest.

Referencias

[1] Peinador Sala, J. I. (2025). The Modular Spectrum of π: From Prime Channel Structure to Elliptic Supercongruences (Version 1). Zenodo. https://doi.org/10.5281/zenodo.17680024

[2] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*. Prentice Hall, 1993.

[3] D. Chudnovsky, G. Chudnovsky, "Approximations and Complex Multiplication", *Ramanujan Revisited*, 1988.

[4] A. M. Odlyzko, "On the distribution of spacings between zeros of the zeta function", *Mathematics of Computation*, vol. 48, no. 177, pp. 273–308, 1987.