REIST Division: A Mathematical and Applied Framework for Negative Remainder Division and Process Optimization

Extended & Refined Version

Author Identifier

Rudolf Stepan
Independent Researcher, Vienna, Austria

ORCID: https://orcid.org/0009-0004-2842-2579
Google Scholar: https://scholar.google.com/citations?user=ER3cemsAAAAJ
Zenodo: https://zenodo.org/search?q=Rudolf%20Stepan
Academia.edu: https://independentresearcher.academia.edu/rudolfstepan

Abstract

Traditional integer division enforces a non-negative remainder, a constraint that simplifies arithmetic but introduces asymmetries and inefficiencies in systems that operate cyclically, discretely, or require feedback-based correction.

This paper introduces **REIST Division** (Remainder-Extended Inversion and Subtraction Technique), a generalized framework that permits negative remainders within bounded limits. By treating the remainder as a *signed correction factor* rather than a residual value, REIST Division provides a more flexible representation for cyclic, logistical, computational and predictive processes.

This extended edition presents a refined formal definition, mathematical properties, illustrative examples, and real-world applications in logistics, scheduling, signal processing, robotics, CPU pipeline correction, forecasting, and numerical algorithms. The REIST framework reveals connections to balanced modular arithmetic, rounding functions, error-feedback control systems, and phase-alignment models.
The framework is contextualized within the broader history of arithmetic generalizations such as negative numbers, complex numbers, and signed zero in IEEE-754 floating-point arithmetic.

TOC

1. Introduction

The classical division identity for integers

$$T = Q \cdot B + R, 0 \leq R < B$$

fixes the remainder to a non-negative value. This constraint simplifies basic arithmetic but embeds an *asymmetry* that is suboptimal in many real-world settings:

- cyclic systems
- scheduling and phase-alignment tasks
- iterative numerical algorithms
- forecasting approaches
- distributed computing
- resource-balancing systems

In such domains, the remainder is better interpreted as a **signed deviation**, not as a leftover quantity. Classical division ignores the possibility that a *negative adjustment* can be the most efficient corrective action.

REIST Division relaxes the non-negativity constraint and redefines the allowable remainder interval so that:

- remainders represent *direction* and *magnitude* of deviation
- quotient and remainder jointly encode the correction
- adjustments become locally optimal in cyclic contexts

This yields a more symmetrical, flexible representation of division.

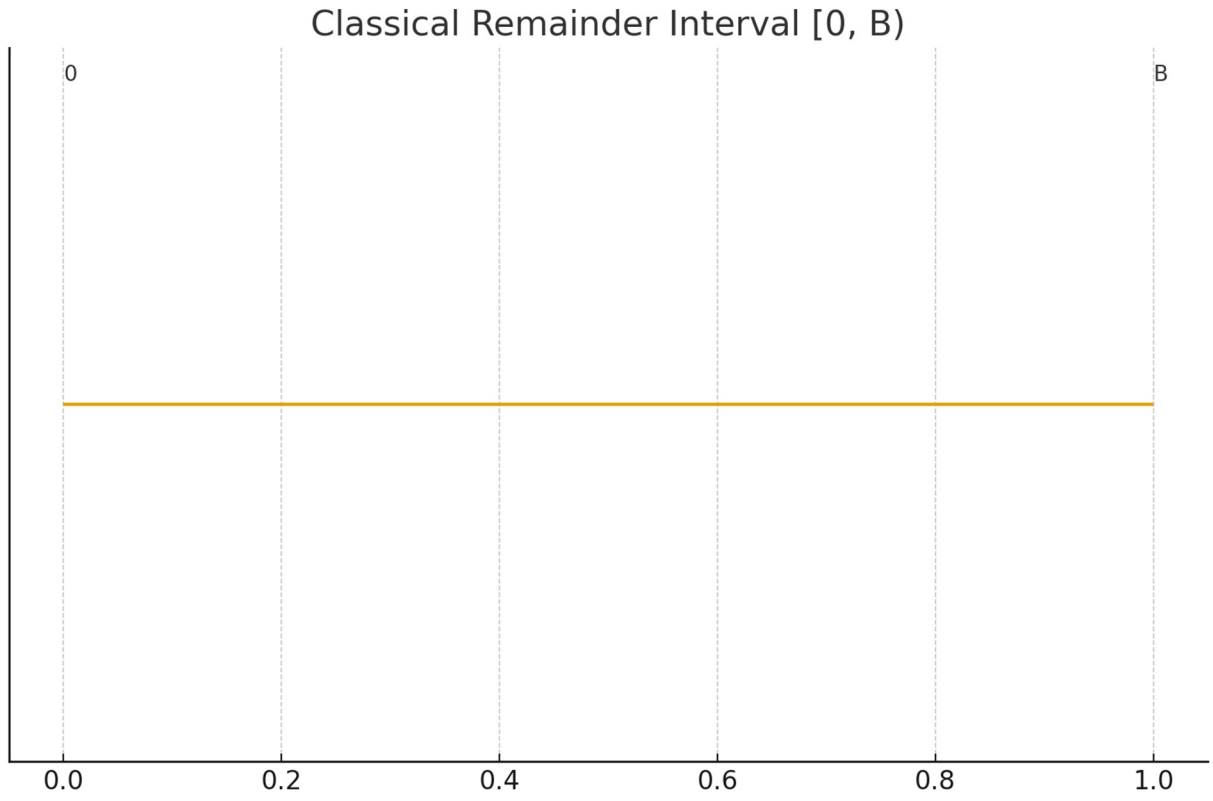

Figure 1. Classical remainder interval [0, B).
This visualization illustrates that classical division does not allow negative corrections, producing an inherent structural asymmetry.

2. Theoretical Framework

2.1 Classical Division (for comparison)

Given integers T and $B > 0$, classical division provides unique integers Q, R satisfying:

$$T = QB + R, 0 \leq R < B.$$

This forces all deviations upward. If a value is "just below" a boundary, classical division overshoots rather than undershoots.

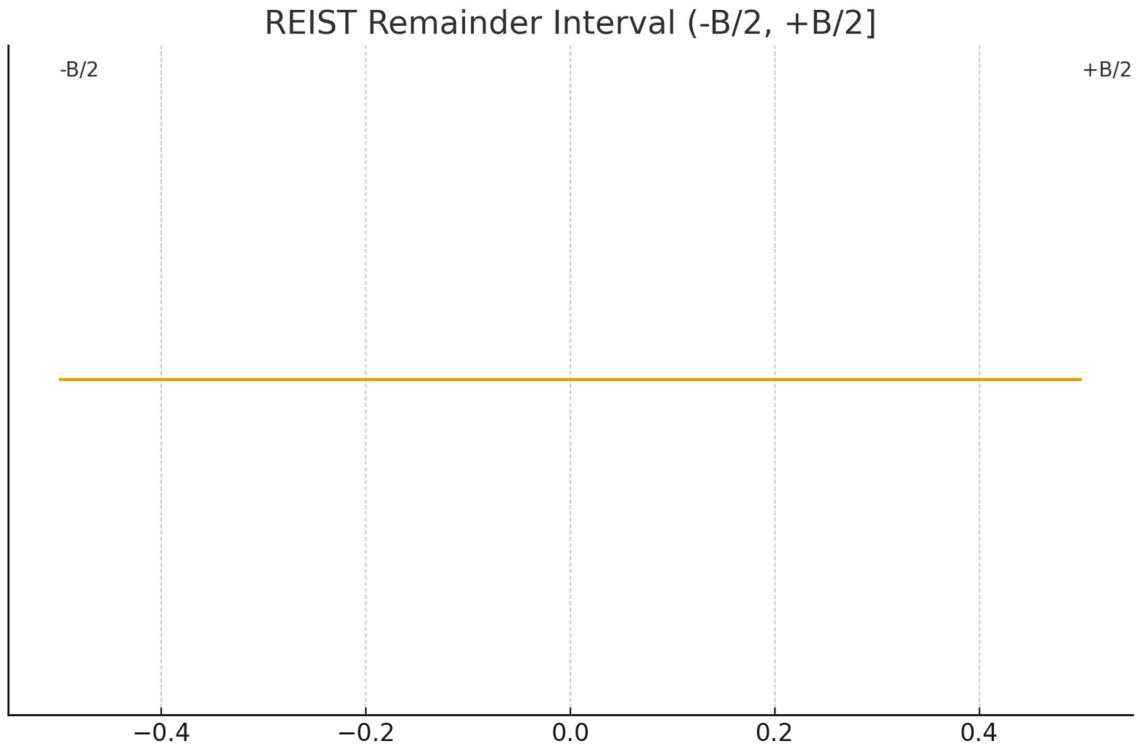

Figure 2. Classical and REIST remainder intervals.
This illustration contrasts the classical remainder interval $[0, B)$, which restricts all corrections to positive values, with the REIST interval $(-B/2, +B/2]$, which allows symmetric positive and negative adjustments and removes the inherent asymmetry of classical division.

2.2 Definition of REIST Division

REIST Division generalizes the remainder interval to:

$$T = QB + R, -\frac{B}{2} < R \leq \frac{B}{2}.$$

This symmetric interval around zero removes the asymmetry of classical arithmetic.

Interpretation:

- $\boldsymbol{R > 0}$**: the system is leading**
- $\boldsymbol{R < 0}$**: the system is lagging**
- $\boldsymbol{R = 0}$**: perfect alignment**

Unlike balanced modulo arithmetic, REIST feeds the signed remainder back into the quotient selection, ensuring the chosen quotient minimizes deviation.

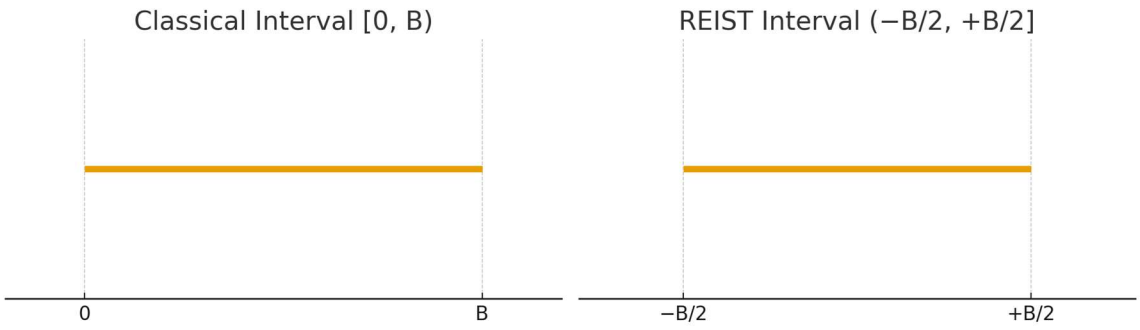

Figure 3. Comparison of classical and REIST remainder intervals.
The classical remainder interval $[0, B)$is asymmetric and permits only positive corrective deviations. REIST Division replaces this with a symmetric interval $(-B/2, +B/2]$, enabling both positive and negative adjustments and eliminating the inherent asymmetry of classical division.

2.3 Mathematical Properties

Uniqueness

- For odd B: unique remainder.
- For even B: two midpoint candidates; a deterministic tie-break rule assigns $R = +B/2$.

Relation to Balanced Modulo

Balanced modulo returns a signed remainder but does not adjust the quotient.
REIST integrates both in a single step.

Relation to Rounding

For odd B:

$$Q = \lfloor \frac{T}{B} + \frac{1}{2} \rfloor$$

so REIST quotient selection is equivalent to nearest integer rounding.

Error-Minimizing Quotient

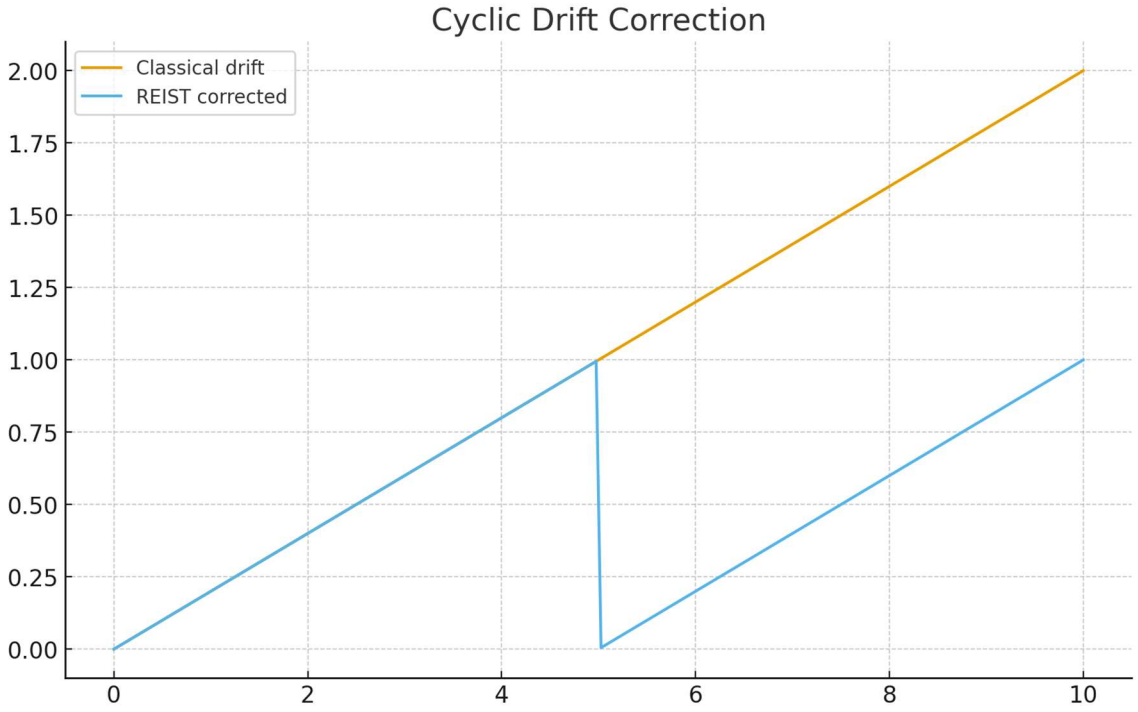

Figure 4. Cyclic drift behavior under classical division versus REIST correction. This diagram illustrates how classical division accumulates drift linearly over time, while REIST Division resets the deviation using a signed correction. The REIST approach minimizes the remainder magnitude and prevents drift growth, leading to more stable behavior in cyclic or periodic systems.

REIST chooses the quotient that minimizes $|R|$.

This is optimal for drift reduction in:

- signal synchronization
- scheduling
- feedback control
- iterative computations

2.4 Examples

Example 1 – Error Minimization

$$T = 17, B = 10$$

Classical:

$$17 = 1 \cdot 10 + 7$$

Remainder = 7 (large deviation) - Large positive deviation.

REIST:

$$17 = 2 \cdot 10 - 3$$

Remainder = −3 (much smaller deviation) - Smaller signed deviation: $R = -3$.

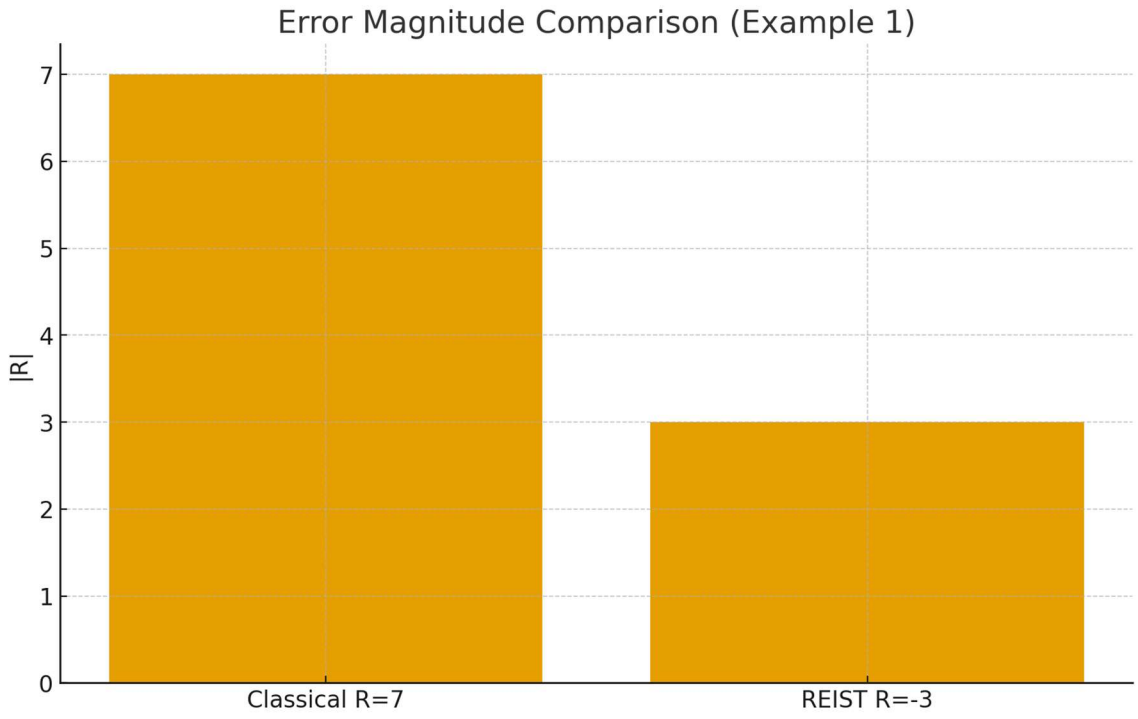

Figure 5. Error magnitude comparison for Example 1 (T = 17, B = 10).
The classical division produces a remainder of $R = 7$, representing a large positive deviation. In contrast, REIST Division yields a signed remainder of $R = -3$, which has a significantly smaller magnitude. This illustrates REIST's core principle: selecting the quotient that minimizes $|R|$ to reduce deviation within a single computational step.

Example 2 – Periodic Alignment

$$T = 28, B = 12$$

Classical: remainder = 4
REIST: remainder = −2 (smaller corrective term)

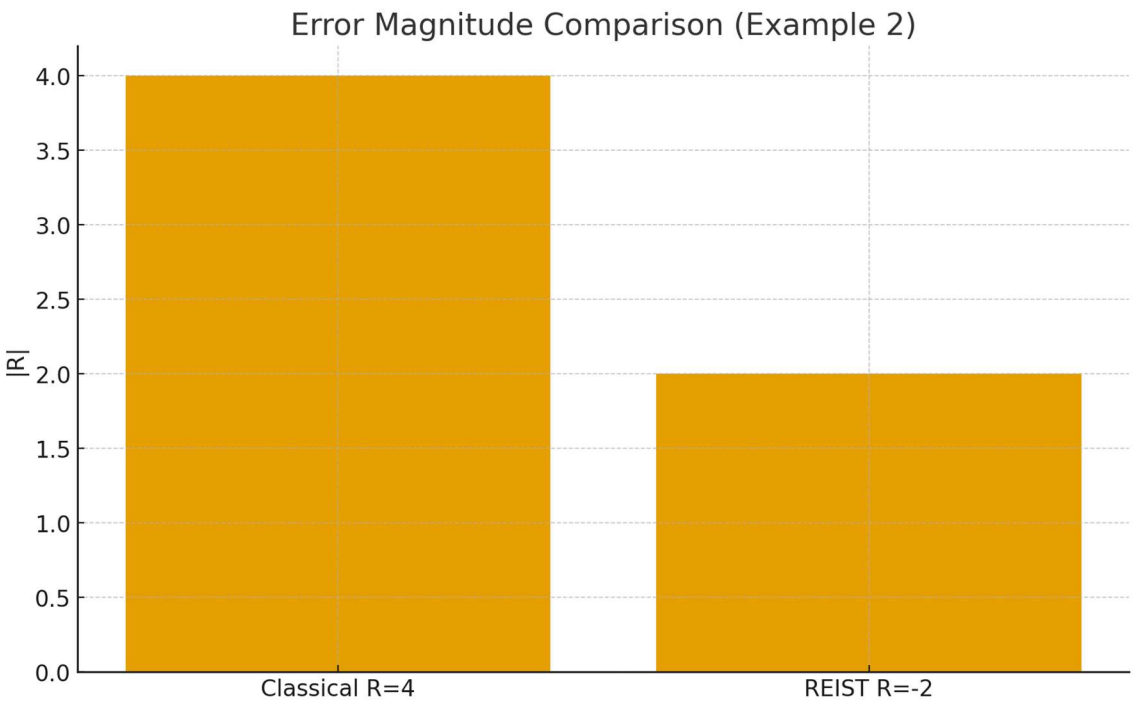

Figure 6. Error magnitude comparison for Example 2 (T = 28, B = 12).
The classical division produces a remainder of $R = 4$, while REIST Division yields a signed remainder of $R = -2$. Although the sign differs, the key point is the reduced magnitude: REIST provides a smaller corrective term, which improves alignment and minimizes deviation in periodic or cyclic systems.

3. Applications (Expanded)

3.1 Supply Chain & Logistics

Negative remainder = negative adjustment in batch size.

Signed corrections enable:

- dynamic batch-size adjustment
- reduced overproduction
- smoother JIT alignment
- symmetric safety stock behavior

3.2 Cyclic Systems & Scheduling

Any periodic activity benefits from direction-aware corrections:

- production cycles
- robotic task loops
- routing and circular queues
- simulation time steps

Negative remainders allow backward adjustments.

3.3 Computational Resource Allocation

In distributed computing:

- positive remainder = over-allocation
- negative remainder = under-allocation

REIST improves symmetric load balancing.

3.4 Forecasting and Predictive Modeling

The remainder acts as an embedded error-correction term, reducing:

- bias accumulation
- drift across iterations
- need for external correction factors

3.5 Digital Signal Processing (New Extension)

Phase errors in DSP are inherently signed. REIST matches:

- PLL alignment
- FFT window synchronization
- carrier phase recovery
- resampling drift correction

3.6 Robotics and Control Engineering (New Extension)

Feedback systems operate on signed error:

- motor control
- joint alignment
- trajectory correction
- PID-based cyclic behavior

REIST offers minimal-error correction in periodic control loops.

3.7 CPU Architecture and Numerical Hardware (New Extension)

Modern CPUs employ signed feedback in:

- division approximation loops
- MAC pipelines
- floating-point correction
- rounding and normalization routines

REIST provides a clean theoretical model for such operations.

4. Discussion (Extended)

Allowing negative remainders is not a minor modification but an extension comparable to the historical acceptance of:

- negative integers
- complex numbers
- balanced ternary notation
- signed zero in IEEE-754

Each generalization expanded arithmetic to model phenomena that classical constructs could not represent symmetrically.
Likewise, REIST Division removes unnecessary asymmetry and aligns integer division with real-world feedback-driven systems.

Open research directions:

- formalization within algebraic ring structures
- algorithmic O(1) implementations
- REIST-based numerical stabilization methods
- applications in AI planning and constraint solvers
- hybrid REIST-modulo operators

5. Conclusion

REIST Division reinterprets the remainder as a bidirectional correction factor, transforming division from a unidirectional residue extraction into a balanced optimization step.

Key advantages:

- symmetric remainder domain
- improved numerical stability
- natural alignment with cyclic and feedback systems
- broad applicability in engineering, computation, and modeling

REIST represents a conceptual and practical generalization of division, comparable to earlier mathematical expansions that improved expressiveness and real-world applicability.

6. References

Knuth, D. E. (1997). *The art of computer programming: Vol. 2. Seminumerical algorithms* (3rd ed.). Addison-Wesley.

Niven, I., Zuckerman, H. S., & Montgomery, H. L. (1991). *An introduction to the theory of numbers* (5th ed.). Wiley.

Graham, R. L., Knuth, D. E., & Patashnik, O. (1994). *Concrete mathematics* (2nd ed.). Addison-Wesley.

Zhang, Y., & Qi, L. (2014). Balanced modular arithmetic and its applications. *Journal of Number Theory, 142*, 1–12. https://doi.org/10.1016/j.jnt.2014.03.012

Oppenheim, A. V., Schafer, R. W., & Buck, J. R. (1999). *Discrete-time signal processing* (2nd ed.). Prentice Hall.

Proakis, J. G., & Manolakis, D. G. (2007). *Digital signal processing* (4th ed.). Prentice Hall.
Gardner, F. M. (2005). *Phaselock techniques* (3rd ed.). Wiley.

Åström, K. J., & Murray, R. M. (2012). *Feedback systems: An introduction for scientists and engineers*. Princeton University Press.

Siciliano, B., Sciavicco, L., Villani, L., & Oriolo, G. (2009). *Robotics: Modelling, planning and control*. Springer.

Ogata, K. (2010). *Modern control engineering* (5th ed.). Prentice Hall.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes* (3rd ed.). Cambridge University Press.

IEEE Standards Association. (2019). *IEEE standard for floating-point arithmetic (IEEE 754-2019).*

Kahan, W. (1997). *Lecture notes on the status of IEEE 754*. University of California, Berkeley.

Stepan, R. (2025). *REIST division*. Zenodo. https://doi.org/10.5281/zenodo.17612788
Stepan, R. (2025). *The Petra Principle*. Zenodo. https://doi.org/10.5281/zenodo.17621787
Stepan, R. (2025). *Black holes without singularities*. Zenodo. https://doi.org/10.5281/zenodo.17634650

As discussed in Stepan (2025), cognitive saturation often emerges when systems exhibit structural asymmetries. REIST Division provides a more symmetric arithmetic representation, reducing complexity in cyclic reasoning tasks.

Comparable to the reinterpretation of gravitational infinities proposed in Stepan (2025), REIST Division aims to eliminate unnecessary asymmetries within classical arithmetic structures.

About the Author

Rudolf Stepan is an independent researcher from Vienna with a long background in engineering, software development, and problem-solving across technical domains. Coming from outside the traditional academic system, he approaches scientific questions with a practical mindset — exploring mathematics, physics, and cognitive science through conceptual clarity, engineering intuition, and self-directed study.

His work focuses on rethinking established models when they show structural limitations: the role of infinities in gravitational theory, the cognitive boundaries of expert reasoning, and asymmetries in classical arithmetic.
Rather than following disciplinary boundaries, he develops ideas where different fields naturally overlap, aiming for simple, usable concepts that make complex systems easier to understand.