Readability Is Freedom: A Case Against Proprietary Math Empires A Philosophical Defense of

Open-Source Computation

Prologue: The Emperors Equation

In 1675, Gottfried Leibniz fought to keep calculus free from Isaac Newtons proprietary grip, knowing that monopolies on mathematics choke progress.[1] Today, we face a similar battlenot between men, but between computational languages. Proprietary systems like Wolfram Language and MATLAB promise insight but deliver control, locking knowledge behind paywalls and opaque syntax. This essay is a case against these math empires and a defense of open-source alternatives like Python, which embody skepticism, accessibility, and communal progress. Readability is not just a technical virtueits a political act of resistance.

1. Python Is a Gift, Not a Gate

When Guido van Rossum released Python in 1991, he didnt just create a languagehe gifted a tool to the world.[2] Pythons open-source ethos built a community, not a walled economy. Libraries like NumPy, SymPy, and SciPy democratized numerical and symbolic computation, offering free, transparent alternatives to proprietary giants.[3] Contrast this with Wolfram Language and MATLAB, which monetize access to their toolchains, turning science into a subscription service.

Wolframs symbolic algebra and MATLABs matrix operations are powerful, but their price is obedience. Want to know how `DSolve[]` works? Good luckyoull need a license and a prayer. Python, meanwhile, invites you to peek under the hood with `numpy.source()` or fork SymPys symbolic engine on GitHub. This isnt just a technical differenceits a philosophy of access. Python says, Heres the code. Make it yours. Wolfram and MATLAB say, Pay up and trust us.

2. Culture of Parsimony vs. Proprietary Overload

Python thrives on parsimony. Its libraries are modular, each doing one thing well: NumPy for arrays, SymPy for algebra, Matplotlib for plots. This simplicity invites participationanyone can learn, tweak, or extend a library.[4] Wolfram Language, by contrast, is a monolith, built around Stephen Wolframs singular vision of computation.[5] Its syntax, while English-like, is esoteric, with

[1] Isaac Newton and Gottfried Wilhelm Leibniz, The Calculus Wars, ed. Jason Socrates Bardi (New York: Thunders Mouth Press, 2006), 4550.

[2] Guido van Rossum, Pythons Design Philosophy, Python Documentation Archive, 1991, https://www.python.org/doc/essays/foreword/.

[3] NumPy Developers, NumPy: The Fundamental Package for Scientific Computing, NumPy Documentation, 2023, https://numpy.org/doc/stable/.

[4] Travis E. Oliphant, NumPy: A Guide to the Scientific Python Ecosystem, Computing in Science & Engineering 9, no. 3 (2007): 1017.

[5] Stephen Wolfram, A New Kind of Science (Champaign, IL: Wolfram Media, 2002), 1215.

idioms that only make sense within Wolframs paradigm. Try debugging a `CompiledFunction[...]` output, and youre met with a black box.

Pythons artificialitykeywords like `def`, `import`, `lambda`is its strength. It doesnt pretend to be natural; its a tool, legible and honest. Wolframs faux readability is a trap, masking proprietary operations as inevitable truths. Pythons clarity fosters collaboration; Wolframs opacity demands allegiance.

3. Personal Vision vs. Economic Extraction

Every language reflects its creators vision, but the difference lies in what happens next. Van Rossum shaped Pythons minimalist syntax but stepped down as its Benevolent Dictator for Life in 2018, letting the community steer its future.[6] His humility birthed a living language. Stephen Wolfram, by contrast, remains the eternal prophet of his language, embedding his ontologyand egointo its syntax.[7] MATLAB, meanwhile, evolved under corporate custodianship, not communal governance, prioritizing profit over accessibility.[8]

Wolfram didnt just build a languagehe built a kingdom with toll booths. Every function, every update, every computational essay reinforces his brand. Python built a playground with toolboxes, where anyone can tinker, teach, or transform. This is the difference between a commons and a chokepoint.

4. Ecosystem as Commons vs. Ecosystem as Brand

Pythons ecosystem is a distributed, open-source commons. Libraries are maintained by volunteers, academics, and coders worldwide, aligned with the ethos of science as a shared endeavor.[9] The Wolfram ecosystem is centralized, closed-source, and dominated by a single author. Its toolsWolframAlpha, Mathematica, the Wolfram Cloudare extensions of a brand, not a community.[10]

Python became a lingua franca because it prioritized collaboration over control. Its libraries are battle-tested by millions, from students in Nairobi to researchers in Peru, who can fork and adapt without permission. Wolframs cult dialect, by contrast, thrives on exclusivity, its logic locked behind licenses and paywalls. Science demands pluralism; Wolfram offers a monarchy.

5. The Fork Test: A Litmus for Freedom

Heres a simple test for any computational tool: Can a scientist in a resource-constrained lab rewrite part of the engine to fit local needswithout fees or gatekeepers? If the answer is no, its dogma

[6] Guido van Rossum, Transfer of Power, Python Mailing List, July 12, 2018, https://mail.python.org/pipermail/python-dev/2018-July/154148.html.

[7] Stephen Wolfram, The Wolfram Language: Fast Introduction for Programmers, Wolfram Documentation, 2023, https://www.wolfram.com/language/fast-introduction-for-programmers/.

[8] MathWorks, MATLAB Licensing Terms, MathWorks Documentation, 2023, https://www.mathworks.com/products/matlab/licensing.html.

[9] Python Software Foundation, The Zen of Python, Python Documentation, 2004, https://peps.python.org/pep-0020/.

[10] Wolfram Research, Wolfram Ecosystem Overview, Wolfram Documentation, 2023, https://www.wolfram.com/ecosystem/.

dressed as software.[11] Python passes with flying colorsits source code is open, its libraries forkable. Wolfram and MATLAB fail spectacularly, their internals sealed like royal vaults.

The right to fork is the right to dissent. If you cant audit the code, you cant trust the method. If you cant debug, youre not free. Proprietary systems dont just limit accessthey limit skepticism, the heartbeat of scientific progress.

6. Readability Is a Political Virtue

Readable code is more than maintainableits liberating. It invites critique, sharing, and reinvention. Opaque systems, by contrast, are epistemic weapons, extracting rent from ignorance and treating trust as a commodity.[12] Pythons transparencyits open source files, clear syntax, and modular designmakes knowledge a commons, not a kingdom. Wolframs opacity turns computation into a priesthood, where only the anointed understand the rituals.

If science is a collective pursuit, its tools must be inspectable. If truth is the goal, the code must be as open as the questions it answers.

Epilogue: A Skeptics Creed

We reject: - Brilliance that hides its logic. - Founders who play prophet. - Knowledge sold without scrutiny.

We demand: - Auditable code. - Transparent syntax. - The freedom to fork.

Math is a path to truth, not a product to be leased. Resist the empires of opacity. Fork something. Teach with open tools. Demand show me over trust me. The future of computationand sciencedepends on it.

Draw your own map. Start now.

References

[1] Bardi, Jason Socrates, ed. The Calculus Wars. New York: Thunders Mouth Press, 2006.

[2] Van Rossum, Guido. Pythons Design Philosophy. Python Documentation Archive, 1991. https://www.python.org/doc/essays/foreword/.

[3] NumPy Developers. NumPy: The Fundamental Package for Scientific Computing. NumPy Documentation, 2023. https://numpy.org/doc/stable/.

[4] Oliphant, Travis E. NumPy: A Guide to the Scientific Python Ecosystem. Computing in Science & Engineering 9, no. 3 (2007): 1017.

[5] Van Rossum, Guido. Transfer of Power. Python Mailing List, July 12, 2018. https://mail.python.org/pipermail/python-dev/2018-July/154148.html.

[6] Wolfram, Stephen. A New Kind of Science. Champaign, IL: Wolfram Media, 2002.

[11] Adapted from Richard Stallman, Free Software, Free Society, Philosophy of the GNU Project, 2002, https://www.gnu.org/philosophy/free-sw.html.

[12] Lawrence Lessig, Code: Version 2.0 (New York: Basic Books, 2006), 138140.

[7] Wolfram, Stephen. The Wolfram Language: Fast Introduction for Programmers. Wolfram Documentation, 2023. https://www.wolfram.com/language/fast-introduction-for-programmers/.

[8] MathWorks. MATLAB Licensing Terms. MathWorks Documentation, 2023. https://www.mathworks.com/products/matlab/licensing.html.

[9] Python Software Foundation. The Zen of Python. Python Documentation, 2004. https://peps.python.org/pep-0020/.

[10] Wolfram Research. Wolfram Ecosystem Overview. Wolfram Documentation, 2023. https://www.wolfram.com/ecosystem/.

[11] Stallman, Richard. Free Software, Free Society. Philosophy of the GNU Project, 2002. https://www.gnu.org/philosophy/free-sw.html.

[12] Lessig, Lawrence. Code: Version 2.0. New York: Basic Books, 2006.