The Journal of Open Source Software

Kigali Sim: Open source simulation toolkit for modeling substances and policies related to the Montreal Protocol

A Samuel Pottinger [1¶], **Balaji Natarajan** [2], **Magali de Bruyn** [1], **and Ciera Martinez** [1]

1 Schmidt Center for Data Science and Environment, University of California, Berkeley, California, United States of America **2** Secretariat of the Multilateral Fund for the Implementation of the Montreal Protocol, United Nations Environment Programme ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: William Gearty

Reviewers:

- @fredshone
- @lionel42

Submitted: 04 November 2025
Published: unpublished

Summary

Actively used internationally, Kigali Sim offers stock and flow modeling of Montreal Protocol-controlled substances. Focusing on the Kigali Amendment which reduces hydrofluorocarbons that contribute to climate change through warming potentials up to thousands of times greater than carbon dioxide, this open source platform aids both atmospheric science and policy development. Supporting a diverse community with heterogeneous programming expertise, this parallelized toolkit interoperates visual no-code rapid model development and a domain-specific language (DSL), democratizing advanced computational tools. Through WebAssembly (WASM) or Java Virtual Machine (JVM), Kigali Sim offers portable, private, and rigorous simulation in support of history's most successful international environmental treaty.

Statement of Need

Signed by all UN member states, the Montreal Protocol phased out 99% of ozone-depleting substances (World Meteorological Organization, 2022). Already the most successful international environmental treaty (McKenzie et al., 2019), its ambitious 2016 Kigali Amendment extends this multilateral framework to hydrofluorocarbons which contribute significantly to climate change (Birmpili, 2018; Velders et al., 2009). However, research and policy analysis for these HFCs involves modeling complex economic, technological, energy, and policy interactions (MLF Staff, 2024).

State of field

National Ozone Units (NOUs) and others who possess deep domain knowledge may work with limited resources (UNEP OzonAction, n.d.). Currently requiring powerful but closed source (Gluckman Consulting, 2023) or ad-hoc models (MLF Staff, 2024) that serve a valued but distinct role, we contribute new access through open domain-oriented tools for holistic stock and flow simulation (Augspurger et al., 2023).

Research impact statement

Serving Article 5 nations (United Nations, 1987), Implementing Agencies, analysts, researchers, and related organizations, Kigali Sim provides the first open source reusable lifecycle modeling toolkit focused on Montreal Protocol-controlled substances. Simulating emissions, energy, consumption, equipment populations, trade, and policy while following treaty conventions

(Multilateral Fund Secretariat, 2024), more than a dozen nations and supporting organizations co-designed Kigali Sim over more than a year, spanning scientists, analysts, and policy-makers. Participating in many international meetings (Executive Committee of the Multilateral Fund, 2025; Natarajan et al., 2025; Pacific Island Countries Ozone Officers Network, 2025; South East Asia and the Pacific Ozone Officers Network, 2025), multiple governments publicly acknowledge contributing to this community project (Kigali Sim Community, 2026) which also received media coverage (Shaban, 2025).

Software Design

Kigali Sim's engine supports domain experts in atmospheric science and environmental policy with varied programming expertise through a dual-interface design.

Flexible engine

Countries and supporting organizations work with varied information from trade records to industry census data to observed emissions. Given diverse methodologies, Kigali Sim pushes information from known to unknown stocks, providing automated unit conversions dependent on equipment properties. As opposed to a unidirectional structure with a single entry-point, this propagates limited user-provided values through substance flows and lifecycles to estimate unmeasured quantities. It then layers complex policy interventions such as permitting and recycling on top of the triangulated "business as usual" scenario.

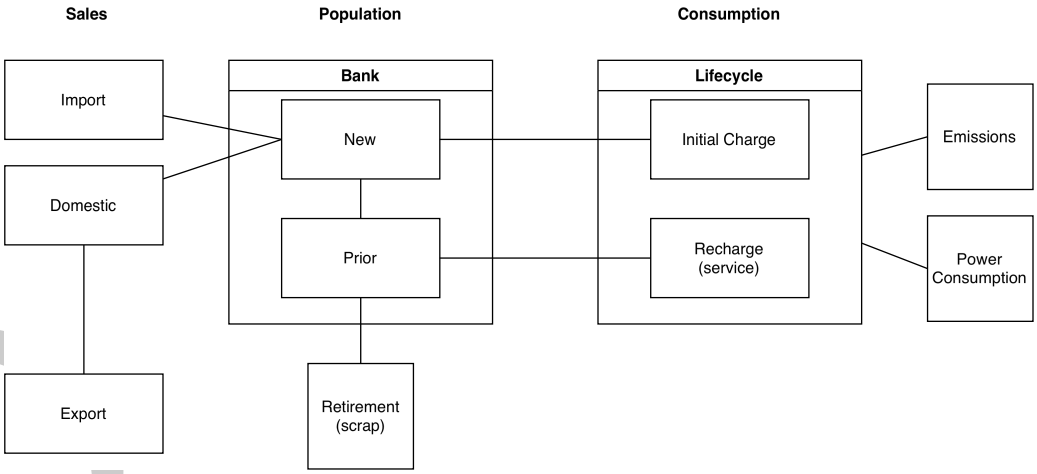

Figure 1: Diagram showing data flow through Kigali Sim simulation engine from input data (trade records, bank estimates, or equipment surveys) through stock and flow calculations to output metrics.

Dual-interface design

Many Kigali Sim users do not identify as programmers and, reflecting empirical findings that domain experts with "limited programming knowledge" benefit from DSLs (Schindler & Cabot, 2022), it forgoes general purpose languages to instead progress from a UI-based editor to DSL code-based authoring. Most simulations can be modified either by the UI-based editor or the code-based editor where changes in one reflect in the other, attempting to bridge preferences and skill sets. Indeed, many community collaborators who identify as non-programmers report starting in the UI-editor but transition to code.

UI-based authoring

To support beginning programmers, the UI-based point-and-click editor acclimates the user to Kigali Sim's concepts through loop-based design (A. Pottinger et al., 2025; Skills, 2020) where small GUI-based changes automatically translate to code run for immediate feedback. This web interface progressively exposes functionality through sequenced disclosure (Brown, 2015; Hayashida, 2012; A. S. Pottinger & Zarpellon, 2023) as an on-ramp into a more open design (Brown, 2017; A. S. Pottinger et al., 2023) that familiarizes code.

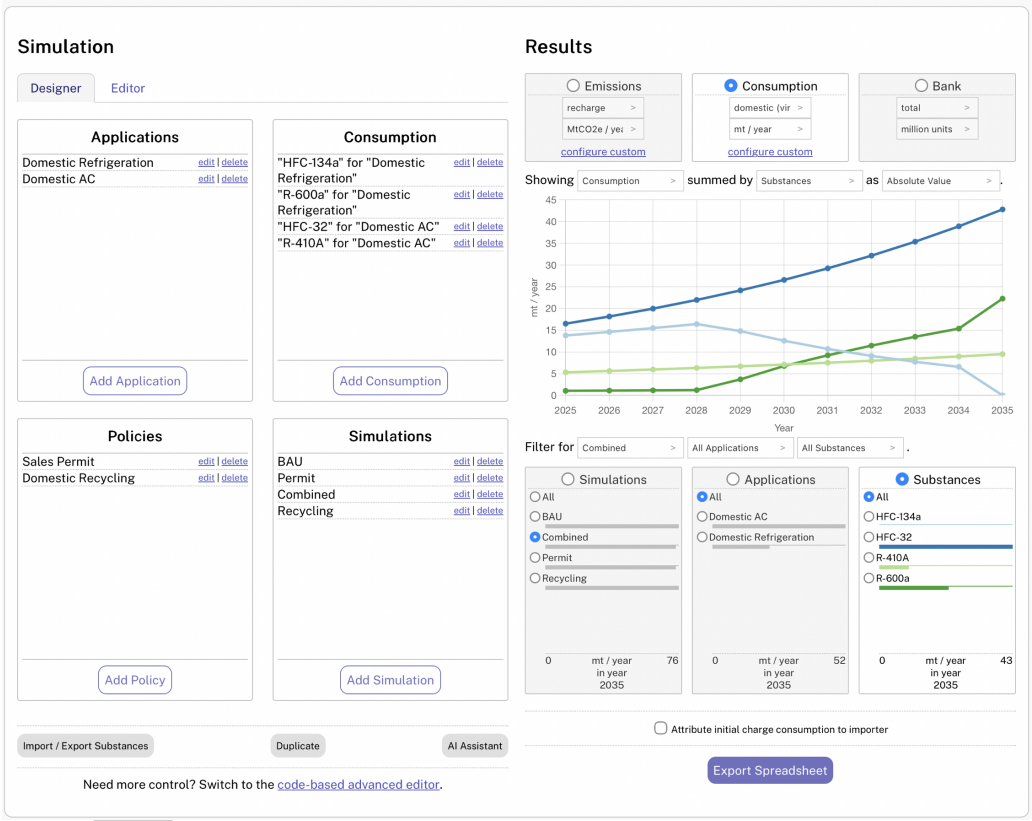

Figure 2: Screenshot of the UI-based editor modifying an example simulation.

Code-based authoring

We find some simulations' complexity cumbersome in UI-based authoring. Therefore, our QubecTalk domain-specific language facilitates expression of complex models in human-readable syntax inspired by but distinct to HyperTalk (Wheeler, 2004). Mirrored by the UI-editor, QubecTalk speaks in treaty terminology, translating terms of art into simulations. This also supports uncertainty quantification, conditional logic, and policy stacking. With optional AI assistant compatibility via the llms.txt specification (Answer.AI, 2024), users may author scripts in a web-based programming portal (Appleton, 2022) or outside with direct JVM invocation.

Limitations and future work

Kigali Sim can model energy consumption but only estimates direct emissions[1]. Additionally, it applies treaty trade attribution but will only attribute charge prior to equipment sale entirely to either the importer or exporter[2]. However, it will receive updates as official guidance changes in the future (Multilateral Fund Secretariat, 2024).

[1] Users may calculate indirect emissions using outside energy mix data (Ang et al., 2016).
[2] Local assembly can be modeled as domestic production.

Implementation

Migrated from an original JavaScript implementation for performance and portability, Kigali Sim runs via WASM[3] or JVM (Konshin, 2024). Without transmitting simulations to external servers, both enable privacy-preserving parallel local computation with BigDecimal (Friesen, 2024) after ANTLR (Parr, 2013) interpretation.

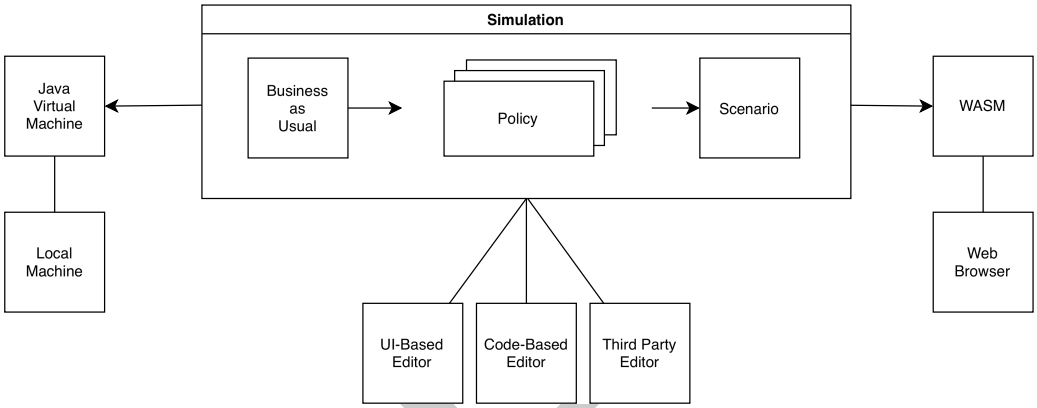

Figure 3: Diagram describing multi-modal execution in which simulations run across different platforms.

Acknowledgments

BSD-licensed.

Thanks

Amanda Anderson-You, Tina Birmpili, Matt Fisher, Ava Hu, Kevin Koy, Douglas McCauley, Alejandro Ramirez-Pabon, Frederico San Martini, Annie Snyder, Suzanne Spencer, Elina Yuen, runtime dependencies (Ajax.org, 2010; Apache Software Foundation, 2024; Bostock, 2024; Brewer et al., 2013; Chart.js Contributors, 2024; Ferdinandi, 2024; Holt, 2024; Konshin, 2024; OpenJS Foundation, 2024; Parr, 2013; Verou, 2024; Webpack Contributors, 2024), drawio (JGraph & draw.io, 2023), and valued community members (Kigali Sim Community, 2026).

Funding

Eric and Wendy Schmidt Center for Data Science and the Environment at UC Berkeley. No conflicts.

AI usage disclosure

As described in-repo, AI used with constrained tasks and strict human review (Anthropic, 2025; GitHub et al., 2025; JetBrains, 2025; Replit, 2025). No substantial LLM use in original JavaScript implementation.

References

Ajax.org. (2010). *Ace: The high performance code editor for the web*. https://ace.c9.io/

Ang, B. W., Goh, T., & Su, B. (2016). Carbon emission intensity in electricity production: A global analysis. *Energy Policy*, *94*, 56–63. https://doi.org/10.1016/j.enpol.2016.03.038

[3] Zero-install browser-based.

Answer.AI. (2024). *The /llms.txt file*. https://llmstxt.org/

Anthropic. (2025). *Claude*. https://www.anthropic.com/claude

Apache Software Foundation. (2024). *Apache commons CSV*. https://commons.apache.org/proper/commons-csv/

Appleton, M. (2022). *Programming portals*. https://maggieappleton.com/programming-portals

Augspurger, T., Malliaraki, E., & Hopkins, J. (2023). *Open source in environmental sustainability*.

Birmpili, T. (2018). Montreal protocol at 30: The governance structure, the evolution, and the kigali amendment. *C. R. Geosci.*, *350*(7), 425–431.

Bostock, M. (2024). *D3.js: Data-driven documents*. https://d3js.org/

Brewer, C., Harrower, M., Sheesley, B., Woodruff, A., & Heyman, D. (2013). Colorbrewer 2.0. In *ColorBrewer*. The Pennsylvania State University. https://colorbrewer2.org

Brown, M. (2015). *Super mario 3D world's 4 step level design*. Game Maker's Toolkit. https://youtu.be/dBmIkEvEBtA

Brown, M. (2017). How nintendo solved zelda's open world problem. In *Game Maker's Toolkit*. https://gmtk.substack.com/p/how-nintendo-solved-zeldas-open-world

Chart.js Contributors. (2024). *Chart.js: Simple yet flexible JavaScript charting library*. https://www.chartjs.org/

Executive Committee of the Multilateral Fund. (2025). *Report of the ninety-seventh meeting of the executive committee*. United Nations Environment Programme. https://www.multilateralfund.org/api/drupal-documents/download/file/8f132cb8-e84c-418e-88c2-e55d9798a33d?filename=9702.pdf

Ferdinandi, C. (2024). *Tabby: Lightweight JavaScript plugin for managing tabs*. https://github.com/cferdinandi/tabby

Friesen, J. (2024). Math, BigDecimal, and BigInteger. In *Learn java fundamentals: A primer for java development and programming* (pp. 311–324). Apress. https://doi.org/10.1007/979-8-8688-0351-2_12

GitHub, OpenAI, & Microsoft. (2025). *GitHub copilot*. https://github.com/features/copilot

Gluckman Consulting. (2023). *HFC outlook: Description and specification*. https://www.gluckmanconsulting.com/wp-content/uploads/2023/11/HFC-Outlook-Description-and-Specification.pdf

Hayashida, K. (2012). *The structure of fun: Learning from super mario 3D land's director*. https://www.gamedeveloper.com/design/the-structure-of-fun-learning-from-i-super-mario-3d-land-i-s-director

Holt, M. (2024). *Papa parse: Powerful CSV parser for JavaScript*. https://www.papaparse.com/

JetBrains. (2025). *JetBrains AI assistant*. https://www.jetbrains.com/ai/

JGraph, & draw.io. (2023). Jgraph/drawio: Draw.io is a JavaScript, client-side editor for general diagramming and whiteboarding. In *GitHub*. GitHub, Inc. https://github.com/jgraph/drawio

Kigali Sim Community. (2026). *Kigali sim credits*. https://kigalisim.org/guide/credits.html

Konshin, A. (2024). *TeaVM: Build fast, modern web applications in java*. https://teavm.org/

McKenzie, R., Bernhard, G., Liley, B., Disterhoft, P., Rhodes, S., Bais, A., Morgenstern,

O., Newman, P., Oman, L., Brogniez, C., & Simic, S. (2019). Success of montreal protocol demonstrated by comparing high-quality UV measurements with "world avoided" calculations from two chemistry-climate models. *Sci. Rep.*, *9*(1), 12332.

MLF Staff. (2024). *Preparation of Kigali HFC implementation plans*. Secretariat of the Multilateral Fund for the Implementation of the Montreal Protocol; Presentation at 46th Meeting of the Open-Ended Working Group of the Parties to the Montreal Protocol. https://ozone.unep.org/system/files/documents/OEWG/%20KIP/%20Side/%20Event/%20MLFS_070824.pdf

Multilateral Fund Secretariat. (2024). *Policies, procedures, guidelines and criteria of the multilateral fund for the implementation of the montreal protocol*. https://www.multilateralfund.org/api/drupal-documents/download/file/4286a8ec-a765-493c-bc0a-2155717410b8?filename=policies_combined.pdf

Natarajan, B., Pottinger, S., Bruyn, M. de, & Birmpili, T. (2025). *HFC modelling agenda for the 46th meeting of the open-ended working group*. Multilateral Fund Secretariat. https://ozone.unep.org/system/files/documents/MLF_HFC_ModellingAgenda.pdf

OpenJS Foundation. (2024). *QUnit: A JavaScript unit testing framework*. https://qunitjs.com/

Pacific Island Countries Ozone Officers Network. (2025). *Final agenda: Pacific island countries ozone officers network meeting*. UNEP OzonAction. https://admin.ozonactionmeetings.org/system/files/2025-09/Final%20Agenda_Network%20Meeting%20Fiji%20Sep%2025.pdf?token=-4AgUCZDrrxipNhhPHCSk_yjRE3xPsrZkDHS6vQOPZo&timestamp=1767648637

Parr, T. (2013). *ANTLR (ANother tool for language recognition)*. https://www.antlr.org/

Pottinger, A. S., Biyani, N., Geyer, R., McCauley, D. J., Bruyn, M. de, Morse, M. R., Nathan, N., Koy, K., & Martinez, C. (2023). *Combining game design and data visualization to inform plastics policy: Fostering collaboration between science, decision-makers, and artificial intelligence*. https://arxiv.org/abs/2312.11359

Pottinger, A. S., & Zarpellon, G. (2023). Pyafscgap.org: Open source multi-modal python-based tools for NOAA AFSC RACE GAP. *Journal of Open Source Software*, *8*(86), 5593. https://doi.org/10.21105/joss.05593

Pottinger, A., Connor, L., Guzder-Williams, B., Weltman-Fahs, M., Gondek, N., & Bowles, T. (2025). Climate-driven doubling of u.s. Maize loss probability: Interactive simulation with neural network monte carlo. *Journal of Data Science, Statistics, and Visualisation*, *5*(3). https://doi.org/10.52933/jdssv.v5i3.134

Replit. (2025). *Replit AI agent*. https://replit.com/ai

Schindler, J., & Cabot, J. (2022). An empirical evaluation of a novel domain-specific language—modelling vehicle routing problems with athos. *Empirical Software Engineering*, *27*(7), 1–44. https://doi.org/10.1007/s10664-022-10210-w

Shaban, B. (2025). *UC berkeley offers tech to united nations to fight global pollution*. NBC Bay Area. https://www.nbcbayarea.com/news/local/uc-berkeley-tech-united-nations-pollution/3994242/

Skills, G. D. (2020). *Core loops in gameplay*. https://gamedesignskills.com/game-design/core-loops-in-gameplay/

South East Asia and the Pacific Ozone Officers Network. (2025). *List of participants: South east asia and the pacific ozone officers network meeting*. UNEP OzonAction. https://admin.ozonactionmeetings.org/system/files/2025-12/List%20of%20participants_0.pdf?token=EPR0bhi2ZJMA6r5XChspmPU174-eovkf7b6lKbdJHHg&timestamp=1766625365

UNEP OzonAction. (n.d.). *National ozone officers' capacity building*. United Nations Environment Programme; Web page. https://www.unep.org/ozonaction/national-ozone-officers-capacity-building

United Nations. (1987). *Montreal protocol on substances that deplete the ozone layer, article 5: Special situation of developing countries*. https://ozone.unep.org/treaties/montreal-protocol/articles/article-5-special-situation-developing-countries

Velders, G. J. M., Fahey, D. W., Daniel, J. S., McFarland, M., & Andersen, S. O. (2009). The large contribution of projected HFC emissions to future climate forcing. *Proc. Natl. Acad. Sci. U. S. A.*, *106*(27), 10949–10954.

Verou, L. (2024). *Prism: Lightweight, extensible syntax highlighter*. https://prismjs.com/

Webpack Contributors. (2024). *Webpack: A static module bundler for modern JavaScript applications*. https://webpack.js.org/

Wheeler, K. (2004). *HyperTalk: The language for the rest of us*. https://www.memoryhole.net/~kyle/papers/hypertalk.pdf

World Meteorological Organization. (2022). *Scientific assessment of ozone depletion: 2022* (p. 509). GAW Report No. 278. ISBN: 978-9914-733-97-6