The Journal of Open Source Software

The Pulmonary Agent-based Infection simulator (PAI): A Multi-Scale Agent-Based Model of Pulmonary Host-Pathogen Interactions

Henrique AL Ribeiro [1¶], **Sandra A Tsiorintsoa** [1], **and Reinhard Laubenbacher** [1]

1 Department of Medicine, Division of Pulmonary, Critical Care, and Sleep Medicine, University of Florida, Gainesville 32610, FL, USA. ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 13 June 2025
Published: unpublished

Summary

Understanding the spatial and regulatory dynamics of the immune response to pulmonary infections remains a central challenge in computational immunology. While mechanistic models, such as ODEs and PDEs, offer insights into intracellular processes, they are limited in representing discrete behaviors and spatial heterogeneity. The pulmonary Agent-based Infection simulator (PAI) addresses this gap through an extensible, multi-scale agent-based modeling designed to simulate lung immune responses to pathogens such as *Aspergillus fumigatus*.

PAI operates within a three-dimensional voxelized space that approximates the alveolar microenvironment, incorporating host and fungal cells. It models inter- and intracellular signaling, immune cell recruitment and movement, and pathogen nutrient acquisition.

PAI is implemented in both Java (jPAI) and C++ (PAI++), with identical outputs and comparable performance, though PAI++ significantly reduces memory usage. With its modular architecture and biological fidelity, PAI provides a valuable tool for conducting in silico experiments where empirical approaches are limited.

Statement of Need

Invasive Pulmonary aspergillosis is a human infection with increasing incidence in immunosuppressed patients such as those receiving chemotherapy or organ transplants (Pappas et al. (2010)). It has also been observed that 10%-14% of COVID-19 patients in the ICU develop invasive aspergillosis (Mitaka et al. (2021); Chong & Neu (2021)). Despite advances in diagnostics and therapy, mortality remains as high as 30–60% in recent surveys (Neofytos et al. (2013)).

Understanding the complex interplay between immune cells, pathogens, and signaling molecules during pulmonary infections requires computational models that capture both the spatial and regulatory dynamics of the immune response (Minucci et al. (2020); Yue & Dutta (2022)). While mechanistic models such as ODEs or PDEs are effective at simulating intracellular or population-level behavior, they lack the flexibility to represent discrete cellular interactions and heterogeneous spatial environments.

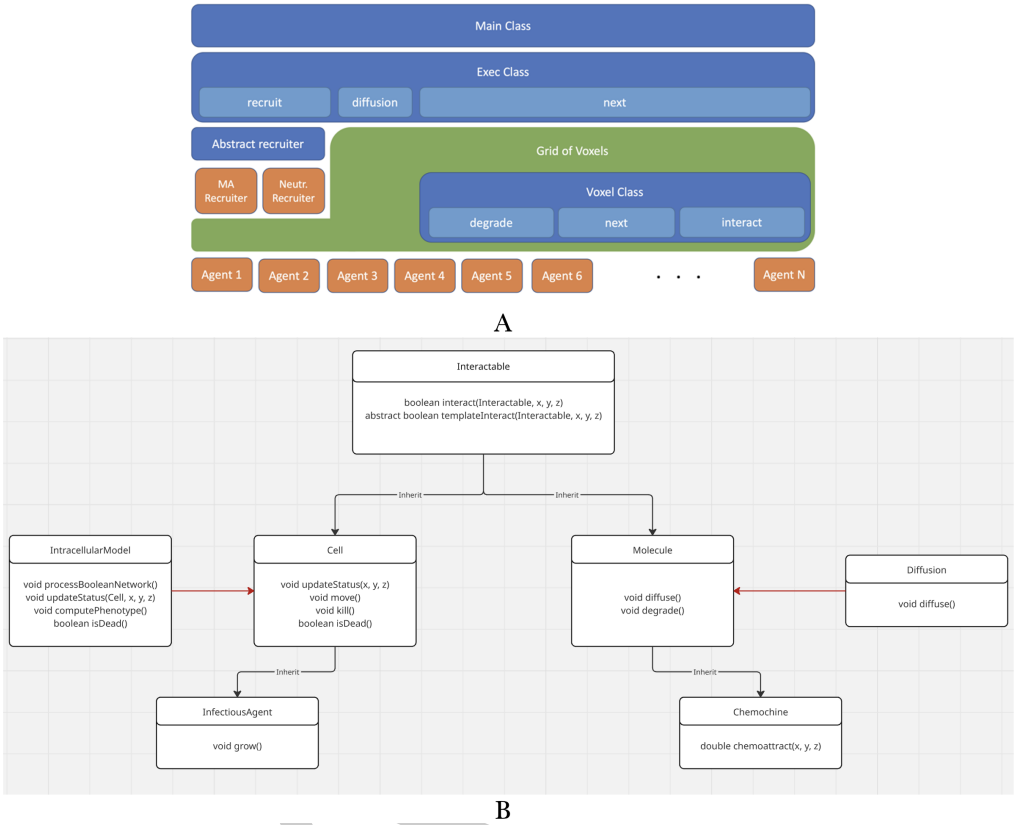

Figure 1: Figure 1: ABM Architecture. A: Overall architecture. The "Exec" iterates through the grid-of-voxels (green) and executes the public methods of Voxel (blue). The Voxels iterates through their local lists of agents (orange bottom), executing their public methods. Exec also executes molecule diffusion, "garbage collection (gc)" of dead cells, and cell recruitment. B: Diagram showing the agent's classes. Black arrows represent inheritance, while red represents composition. All agents are interactables, which means that they can potentially interact with any other agents.

PAI addresses this gap by providing a scalable, extensible agent-based modeling framework for simulating the tissue-scale immune response to lung infections, such as *Aspergillus fumigatus*. PAI has been used before, and the mathematical and biological components are better explained in Ribeiro et al. (2022). While in Ribeiro et al. (2023), we modified it to simulate and generate a hypothesis about Covid-Associated-Pulmonary-Aspergillosis. Qu et al. (2025) expanded the original model to simulate the effect of NETosis and hemorrhage on Aspergillosis. However, none of these papers provides detailed information about the software.

Model

Architecture

As shown in Figure 1A, at the top level, a Main module orchestrates the initialization and execution of simulations, including setting up initial conditions, managing output, and controlling the simulation loop. Initialization is handled by a dedicated Initializer class, which instantiates the molecular fields, populates the simulation space with agents, and assigns intracellular models where appropriate.

The simulation domain is discretized into a three-dimensional voxel grid, where each voxel represents a microenvironment (~40 μm per side) and maintains localized lists of agents. The Voxel class governs local interactions (interact method), molecular degradation (degrade

method), and cell updates (update method) and is invoked iteratively by the Exec class. The Exec class also handles molecular diffusion (via PDE solvers), cellular recruitment (based on chemokine gradients), and agent-level garbage collection.

Agents in the PAI are broadly categorized as Cells or Molecules, both inheriting from an abstract Interactable class (Figure 1B). This base class implements a symmetric interact method based on an "inductive" architecture: new agent types define how they interact with existing agents. If Agent "A" does not provide code to interact with Agent "B," nor does Agent "B" provide code to interact with "A," they learn on the fly that they do not interact (Figure 2).

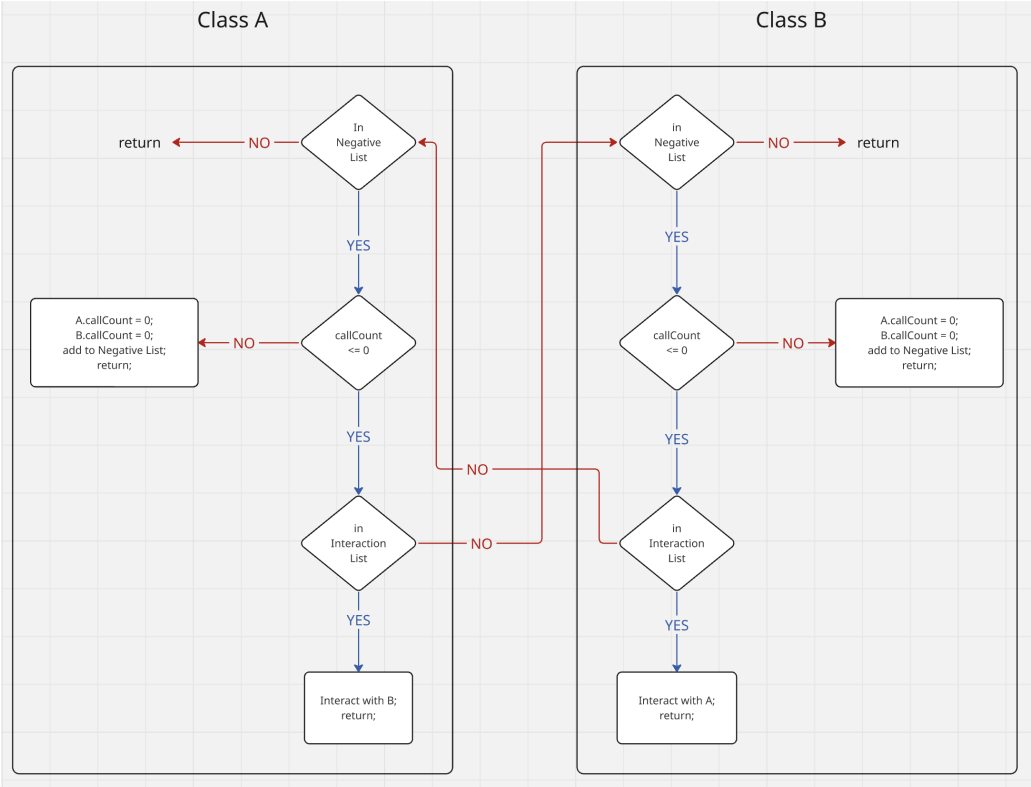

Figure 2: Figure 2: Dynamics of interaction between two agents. Decision diagram made by the methods "interact" and "templateInteract" from the two classes trying to interact ("A" and "B").

Instantiation

A typical instantiation of the PAI model, as used in Ribeiro et al. (2022), involves a 10×10×10 voxel grid populated with both cellular and molecular agents. The simulation initializes molecular fields, Transferrin, Lactoferrin, TAFC, Iron, IL-6, TNF-α, IL-10, TGF-β, CXCL2, CCL4, and Hepcidin. It is also initialized with 640 Type II pneumocytes, 15 macrophages, 15 neutrophils, and 1920 *Aspergillus fumigatus* (in the resting conidia state) randomly distributed across the grid. The simulation proceeds via the Exec class, which iterates over all voxels and over the lists of agents they contain. Simulations typically run for 2160 iterations (approximately 72 hours), as in Ribeiro et al. (2022), where results were quantitatively compared with *in vivo* data and demonstrated close agreement.

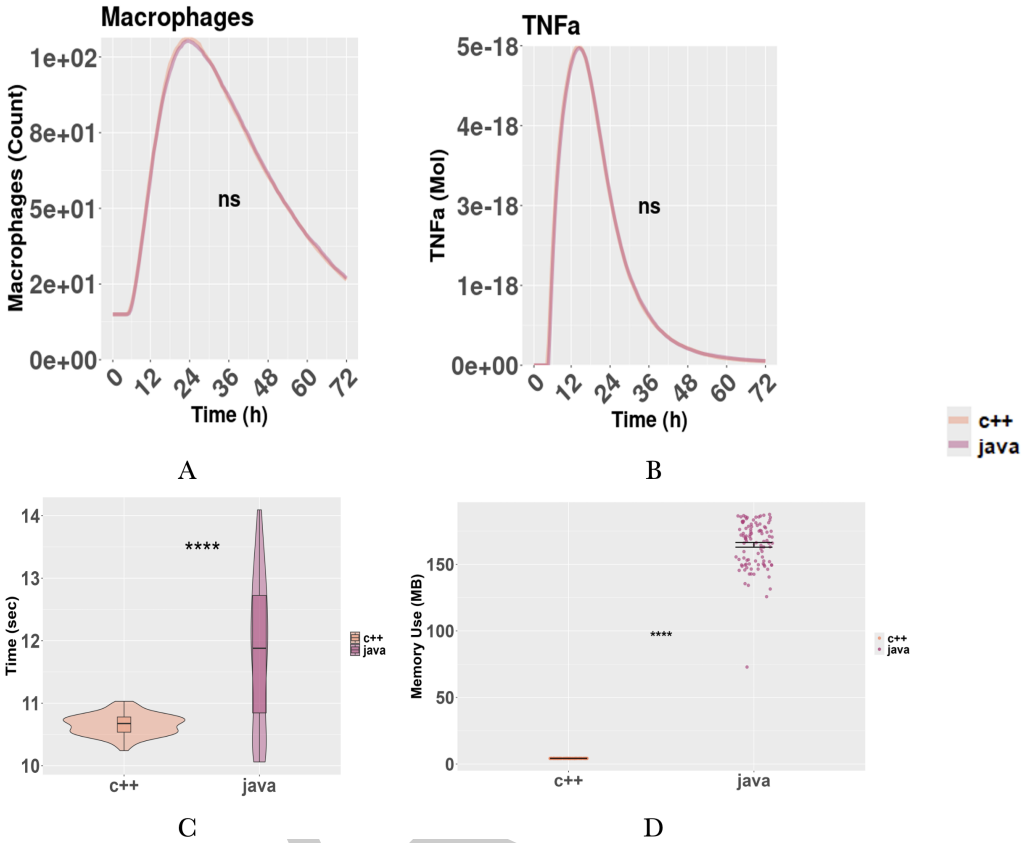

Figure 3: Figure 3: jPAI and PAI++ comparison. A: Macrophage time series. B: TNF-a time series. C: Running time. D: Memory usage. Memory recorded with Valgrind (C++) and JProfiler (Java). All figures represent the average of 100 simulations. Simulations initialized with 10X10X10 voxels, 640 pneumocytes, 1920 conidia, 15 macrophages, and 2160 iterations. PAI++ compiled with g++ 12.4.0 using the MinGW toolchain with level three optimization; jPAI run with Java 23.0.1. Execution environment: Windows 10 Education 64-Bit Dell laptop, Intel® Core™ Ultra 7 155U (14CPUs), ~2.1GHz, 16384MB RAM. A-B: paired t-test. C-D: Wilcoxon rank-sum test. (ns not significant $p > 0.05$; **** $p < 0.0001$).

Benchmark

PAI++ contains only the code of the Ribeiro et al. (2022) model, which is the focus of this paper, while jPAI contains other pieces of code not discussed here. The code used for this benchmark is a simplification of Ribeiro et al. (2022). Within this context, PAI++ and jPAI are nearly identical (Figure 3A-B). The similarity between the two implementations serves as an important benchmark for model reproducibility, a key target in the modeling community (Donkin et al. (2017); Masison et al. (2021)). Figure 3C shows that the running time (wall time) of both implementations is similar (~11 sec), with C++ being slightly faster. On the other hand, there were major differences in the memory usage (Figure 3D). A more thorough benchmark is provided in Benchmark.pdf.

We provided documentation only for the Java version of the code. However, a C++ code follows the same structure.

References

Chong, W. H., & Neu, K. P. (2021). Incidence, diagnosis and outcomes of COVID-19-associated pulmonary aspergillosis (CAPA): A systematic review. *Journal of Hospital Infection*, *113*, 115–129. https://doi.org/https://doi.org/10.1016/j.jhin.2021.04.012

Donkin, E., Dennis, P., Ustalakov, A., Warren, J., & Clare, A. (2017). Replicating complex agent based models, a formidable task. *Environmental Modelling & Software*, *92*, 142–151. https://doi.org/https://doi.org/10.1016/j.envsoft.2017.01.020

Masison, J., Beezley, J., Mei, Y., Ribeiro, H. A. L., Knapp, A. C., Sordo Vieira, L., Adhikari, B., Scindia, Y., Grauer, M., Helba, B., & others. (2021). A modular computational framework for medical digital twins. *Proceedings of the National Academy of Sciences*, *118*(20), e2024287118. https://doi.org/https://doi.org/10.1073/pnas.2024287118

Minucci, S. B., Heise, R. L., & Reynolds, A. M. (2020). Review of mathematical modeling of the inflammatory response in lung infections and injuries. *Frontiers in Applied Mathematics and Statistics*, *6*, 36. https://doi.org/https://doi.org/10.3389/fams.2020.00036

Mitaka, H., Kuno, T., Takagi, H., & Patrawalla, P. (2021). Incidence and mortality of COVID-19-associated pulmonary aspergillosis: A systematic review and meta-analysis. *Mycoses*, *64*(9), 993–1001. https://doi.org/https://doi.org/10.1111/myc.13292

Neofytos, D., Treadway, S., Ostrander, D., Alonso, C., Dierberg, K., Nussenblatt, V., Durand, C., Thompson, C., & Marr, K. (2013). Epidemiology, outcomes, and mortality predictors of invasive mold infections among transplant recipients: A 10-year, single-center experience. *Transplant Infectious Disease*, *15*(3), 233–242. https://doi.org/https://doi.org/10.1111/tid.12060

Pappas, P. G., Alexander, B. D., Andes, D. R., Hadley, S., Kauffman, C. A., Freifeld, A., Anaissie, E. J., Brumble, L. M., Herwaldt, L., Ito, J., & others. (2010). Invasive fungal infections among organ transplant recipients: Results of the transplant-associated infection surveillance network (TRANSNET). *Clinical Infectious Diseases*, *50*(8), 1101–1111. https://doi.org/https://doi.org/10.1086/651262

Qu, G., Ribeiro, H. A., Solomon, A. L., Vieira, L. S., Goddard, Y., Diodati, N. G., Lazarte, A. V., Wheeler, M., Laubenbacher, R., & Mehrad, B. (2025). The heme scavenger hemopexin protects against lung injury during aspergillosis by mitigating release of neutrophil extracellular traps. *JCI Insight*, *10*(10), e189151. https://doi.org/10.1172/jci.insight.189151

Ribeiro, H. A., Scindia, Y., Mehrad, B., & Laubenbacher, R. (2023). COVID-19-associated pulmonary aspergillosis in immunocompetent patients: A virtual patient cohort study. *Journal of Mathematical Biology*, *87*(1), 6. https://doi.org/https://doi.org/10.1007/s00285-023-01940-6

Ribeiro, H. A., Vieira, L. S., Scindia, Y., Adhikari, B., Wheeler, M., Knapp, A., Schroeder, W., Mehrad, B., & Laubenbacher, R. (2022). Multi-scale mechanistic modelling of the host defence in invasive aspergillosis reveals leucocyte activation and iron acquisition as drivers of infection outcome. *Journal of The Royal Society Interface*, *19*(189), 20210806. https://doi.org/https://doi.org/10.1098/rsif.2021.0806

Yue, R., & Dutta, A. (2022). Computational systems biology in disease modeling and control, review and perspectives. *Npj Systems Biology and Applications*, *8*(1), 37. https://doi.org/https://doi.org/10.1038/s41540-022-00247-4