SLiM-Gym: Reinforcement Learning for Population Genetics

Niko Zuppas[1,*], Bryan C. Carstens[1]

[1]Department of Evolution, Ecology, and Organismal Biology and Museum of
Biological Diversity, 1315 Kinnear Rd., Columbus OH 43212

[*]Corresponding author: zuppas.3@osu.edu

22 April 2025

Summary

Wright-Fisher evolutionary dynamics provide a mathematical framework for modeling populations over discrete time steps [Fisher, 1930]. Deep reinforcement learning (RL) has proven highly effective in optimizing complex sequential decisions, achieving expert-level performance on tasks such as Poker and Go [Heinrich and Silver, 2016, Silver et al., 2016]. However, applying RL to evolutionary problems requires suitable training environments. We present `SLiM-Gym`, a Python package that bridges this gap by connecting the `Gymnasium` RL framework with `SLiM`, a forward-time population genetics simulator, enabling researchers to apply RL methods to study evolutionary processes and generate novel hypotheses.

Statement of Need

Evolutionary trajectories are shaped by the interaction of genetic forces acting sequentially across generations. While theoretical models like Wright-Fisher capture these dynamics with well-defined transition probabilities, unified tools for studying and controlling evolutionary processes through reinforcement learning have been limited. Recent work demonstrated the potential of reinforcement learning for controlling ecological and physiological aspects of evolution [Bagley et al., 2025], optimizing breeding programs [Younis et al., 2024] and addressing emergent drug resistance [Engelhardt, 2020]. However, these implementations often rely on custom simulation tools, potentially limiting accessibility for population geneticists. Reinforcement learning has also been applied to phylogenetic tree space [Azouri et al., 2023] and ancestral recombination graph construction [Ray-

1

mond et al., 2024], highlighting the broader potential of RL in evolutionary biology.

SLiM [Haller and Messer, 2023] has emerged as a popular forward-time population genetics simulator, with extensive adoption among population geneticists. It excels at modeling complex evolutionary scenarios with individual-level resolution and supports extensive customization through its Eidos scripting language. However, it lacks agent-based optimization or exploration. Gymnasium [Towers et al., 2024], originally released as Gym by OpenAI in 2016 [Brockman et al., 2016], provides a standardized framework for RL research, while environments like MuJoCo, an advanced physics simulator [Todorov et al., 2012], illustrate the value of coupling RL with high-fidelity simulations. SLiM-Gym combines SLiM's evolutionary modeling with Gymnasium's RL framework, allowing us to leverage RL to generate testable hypotheses about evolutionary processes and outcomes.

SLiM-Gym Wrapper

The base wrapper extends Gymnasium with modifications for SLiM integration. When initializing a new training episode, the wrapper launches SLiM as a subprocess that runs a user-defined simulation recipe as an environment an agent can interact with. Communication between Gymnasium and SLiM occurs through a file-based communication protocol, enabling real-time manipulation of simulation parameters. The wrapper provides abstract methods that environments must implement to handle state processing, action translation, and reward calculation.

Communication Protocol

The SLiM-Gym communication protocol begins by starting the SLiM subprocess and initializing a signaling file. During initialization and burn-in generations, SLiM operates independently of Python. Once this phase is complete, synchronization between both SLiM and SLiM-Gym is achieved through two mechanisms: SLiM's output logging and the signaling file. After burn-in, SLiM logs generation data and deletes the signaling file, entering a waiting state. The wrapper's step function monitors for both presence of the log file and absence of the signaling file, proceeding to execute when both conditions are met. The action generated by the learning algorithm is written to a new signaling file, lifting SLiM's waiting condition and allowing the action to be applied to the simulation before continuing. The process is visualized in Figure 1.

Figure 1: Communication protocol between SLiM-Gym and SLiM recipe. Synchronization between SLiM and SLiM-Gym is achieved through two mechanisms: SLiM's output logging and the signaling file.

Reference Environment

We provide a `SLiM` recipe as a reference environment for modeling a *K*-allele Wright–Fisher process with mutation, formulated as a partially observable Markov decision process (POMDP). While the classical Wright–Fisher model assumes a fixed population size, `SLiM-Gym` relaxes this constraint by allowing the population size N to vary over time according to a specified growth or bottleneck model. The agent interacts with the system by dynamically adjusting the mutation rate μ in response to the changing size of N. This reference environment explores parameter relationships in evolutionary dynamics. Through reinforcement learning, we aim to determine whether an agent can learn to infer and compensate for unobserved demographic changes by adjusting the mutation rate μ to maintain genetic diversity, discovering the functional relationship between these parameters.

The underlying simulation state at generation n includes the allele frequency vector $X_n^N = (p_1, \ldots, p_K)$, where p_i represents the proportion of derived allele i in the population, and a mutation parameter μ. In each generation, the agent selects an action that modifies and sets the next mutation rate μ_{n+1}. `SLiM` then applies evolutionary processes including mutation according to this rate, and draws the resulting allele counts for the next generation from a multinomial distribution with parameters N_{n+1} and the post-mutation frequencies.

The agent does not directly observe X_n^N, N, or μ. Instead, it observes a site

3

Figure 2: Site frequency spectra across three demographic scenarios. The neutral model exhibits the expected 1/frequency distribution, with the proportion of variants declining smoothly as allele frequency increases. The bottleneck model shows characteristic depletion of rare variants and enrichment of intermediate frequency alleles, reflecting the loss of low frequency variants during population contraction. The growth model shows an excess of rare variants, a signature of recent population expansion when numerous new mutations have not had time to reach higher frequencies. These distinct patterns demonstrate how population size history can shape genetic diversity in populations.

frequency spectrum (SFS—see Figure 2 for more detail), represented as a fixed-length, 100-element vector where each entry corresponds to a 1% frequency bucket. During an initial burn-in phase, the environment establishes an expected SFS under constant N and μ.

The reward function evaluates how well the agent maintains the expected site frequency distribution by calculating the Kullback–Leibler divergence between the observed and expected SFS distributions. Population size N is consistently modified by a hidden demographic model and as such the agent must adapt μ_{n+1} based solely on the observed SFS. This provides a theoretically tractable, biologically grounded test case for RL in evolutionary dynamics, with potential extensions to more complex multi-parameter relationships and selective scenarios. For details on custom environments and additional functionality, see the project documentation: `https://github.com/nzupp/SLiM-Gym/`

Conclusion

The application of reinforcement learning algorithms to evolutionary scenarios represents an exciting direction for future research. RL offers a novel way to approach open questions in evolutionary biology by allowing agents to develop process-driven hypotheses about evolutionary dynamics. `SLiM-Gym` enables this exploratory framework, where agent-derived strategies can be scientifically evaluated, advancing our understanding of complex systems.

Acknowledgments

References

D. Azouri, O. Granit, M. Alburquerque, Y. Mansour, T. Pupko, and I. Mayrose. The tree reconstruction game: phylogenetic reconstruction using reinforcement learning. *ArXiv e-prints*, March 2023. doi: 10.48550/arXiv.2303.06695. URL `http://arxiv.org/abs/2303.06695`.

B. A. Bagley, N. Khoshnan, and C. K. Petritsch. Reinforcement Learning for Control of Evolutionary and Ecological Processes. *ArXiv e-prints*, January 2025. doi: 10.48550/arXiv.2305.03340. URL http://arxiv.org/abs/2305.03340.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI Gym. *ArXiv e-prints*, June 2016. doi: 10.48550/arXiv.1606.01540. URL http://arxiv.org/abs/1606.01540.

D. Engelhardt. Dynamic Control of Stochastic Evolution: A Deep Reinforcement Learning Approach to Adaptively Targeting Emergent Drug Resistance. *ArXiv e-prints*, October 2020. doi: 10.48550/arXiv.1903.11373. URL http://arxiv.org/abs/1903.11373.

R. A. Fisher. *The genetical theory of natural selection*. Clarendon Press, Oxford, 1930. doi: 10.5962/bhl.title.27468. URL https://www.biodiversitylibrary.org/bibliography/27468.

B. C. Haller and P. W. Messer. SLiM 4: Multispecies Eco-Evolutionary Modeling. *The American Naturalist*, 201(5):E127–E139, May 2023. doi: 10.1086/723601. URL https://www.journals.uchicago.edu/doi/10.1086/723601.

J. Heinrich and D. Silver. Deep Reinforcement Learning from Self-Play in Imperfect-Information Games. *ArXiv e-prints*, June 2016. doi: 10.48550/arXiv.1603.01121. URL http://arxiv.org/abs/1603.01121.

M. Raymond, M.-H. Descary, C. Beaulac, and F. Larribe. Constructing Ancestral Recombination Graphs through Reinforcement Learning. *ArXiv e-prints*, June 2024. doi: 10.48550/arXiv.2406.12022. URL http://arxiv.org/abs/2406.12022.

D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016. doi: 10.1038/nature16961. URL https://www.nature.com/articles/nature16961.

E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, Vilamoura-Algarve, Portugal, 2012.

IEEE. ISBN 9781467317368. doi: 10.1109/IROS.2012.6386109. URL `http://ieeexplore.ieee.org/document/6386109/`.

M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. D. Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis. Gymnasium: A Standard Interface for Reinforcement Learning Environments. *ArXiv e-prints*, November 2024. doi: 10.48550/arXiv.2407.17032. URL `http://arxiv.org/abs/2407.17032`.

O. G. Younis, L. Corinzia, I. N. Athanasiadis, A. Krause, J. M. Buhmann, and M. Turchetta. Breeding Programs Optimization with Reinforcement Learning. *ArXiv e-prints*, June 2024. doi: 10.48550/arXiv.2406.03932. URL `http://arxiv.org/abs/2406.03932`.