Received: 17 October 2023 | Revised: 21 December 2023 | Accepted: 9 January 2024

DOI: 10.1111/1755-0998.13930

MOLECULAR ECOLOGY RESOURCES WILEY

RESOURCE ARTICLE

Population genetic simulation: Benchmarking frameworks for non-standard models of natural selection

Olivia L. Johnson[1] | Raymond Tobler[2] | Joshua M. Schmidt[3] | Christian D. Huber[1,4]

[1]School of Biological Sciences, University of Adelaide, Adelaide, South Australia, Australia

[2]Evolution of Cultural Diversity Initiative, The Australian National University, Canberra, Australian Capital Territory, Australia

[3]Department of Ophthalmology, College of Medicine and Public Health, Flinders University, Adelaide, South Australia, Australia

[4]Department of Biology, Pennsylvania State University, University Park, Pennsylvania, USA

Correspondence
Christian D. Huber, Department of Biology, Pennsylvania State University, University Park, Pennsylvania, USA.
Email: cdh5313@psu.edu

Funding information
Australian Research Council, Grant/Award Number: DE190101069 and DP190103606; Westpac Foundation; Australian Government; National Institute of Health, Grant/Award Number: R35GM146886

Handling Editor: Frederic Austerlitz

Abstract

Population genetic simulation has emerged as a common tool for investigating increasingly complex evolutionary and demographic models. Software capable of handling high-level model complexity has recently been developed, and the advancement of tree sequence recording now allows simulations to merge the efficiency and genealogical insight of coalescent simulations with the flexibility of forward simulations. However, frameworks utilizing these features have not yet been compared and benchmarked. Here, we evaluate various simulation workflows using the coalescent simulator *msprime* and the forward simulator *SLiM*, to assess resource efficiency and determine an optimal simulation framework. Three aspects were evaluated: (1) the burn-in, to establish an equilibrium level of neutral diversity in the population; (2) the forward simulation, in which temporally fluctuating selection is acting; and (3) the final computation of summary statistics. We provide typical memory and computation time requirements for each step. We find that the fastest framework, a combination of coalescent and forward simulation with tree sequence recording, increases simulation speed by over twenty times compared to classical forward simulations without tree sequence recording, although it does require six times more memory. Overall, using efficient simulation workflows can lead to a substantial improvement when modelling complex evolutionary scenarios—although the optimal framework ultimately depends on the available computational resources.

KEYWORDS
fluctuating selection, msprime, population genetic simulation, SLiM, tree sequence recording

1 | INTRODUCTION

Evolutionary and demographic processes have long been explored using theoretical models (Fisher, 1930; Haldane, 1926; Wright, 1929). While these models have been predominantly examined using analytical methods, simulation approaches have recently become a common companion to such methods and have facilitated the exploration of evolutionary genetic contexts that are not tractable using conventional analytic approaches (Isildak et al., 2021; Korfmann et al., 2023; Matheson & Masel, 2023; Wittmann et al., 2023). Over the last few decades, simulation software has been developed and enhanced to the point that they are now, in principle, easily adaptable to arbitrarily complex population and evolutionary genetic scenarios (Adrion et al., 2020; Baumdicker et al., 2022; Haller & Messer, 2017, 2019, 2023; Hoban et al., 2012). Although realism is still restricted by the time and memory (random-access memory, RAM) required to simulate

a given model, recent advances have increased the capability of simulation software and enabled more efficient use of resources. In the following sections, we provide a brief overview of the two principled means of simulating population genetic models—namely, coalescent simulations and forward simulations (Haller & Messer, 2017; Kelleher et al., 2016; Wakeley, 2009)—then introduce the tree sequence data structure (Haller et al., 2019; Kelleher et al., 2018), a recent advance in data recording that allows for the combination of coalescent and forward simulation in a single framework.

1.1 | Simulation types

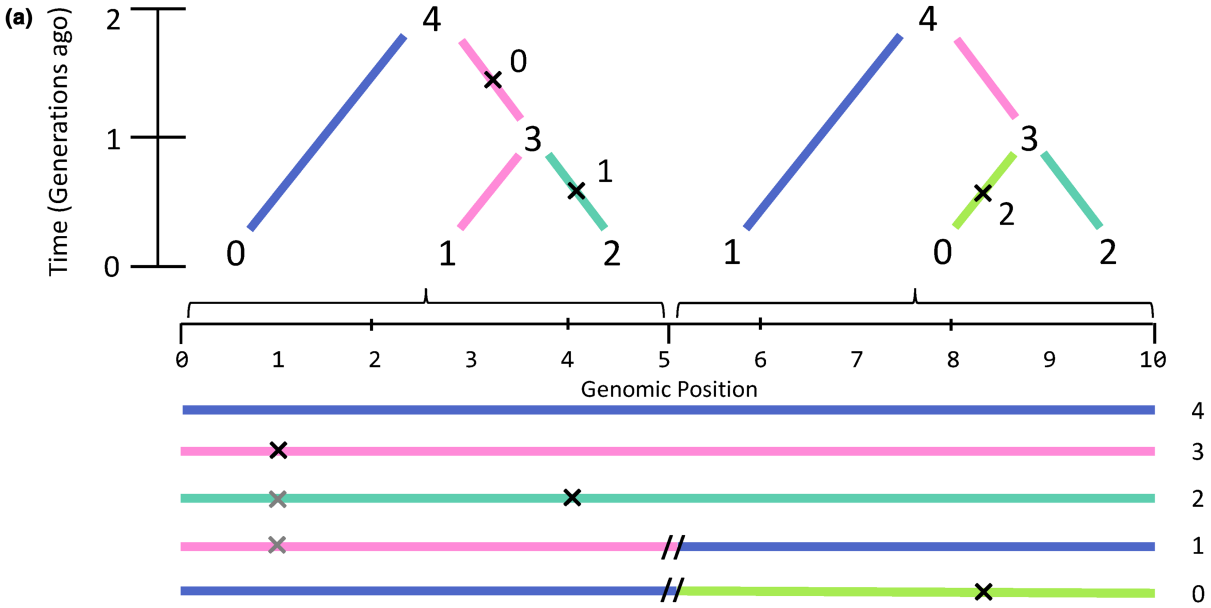
There are three main types of numerical approaches available for population genetic studies: forward, coalescent and resampling (Yuan et al., 2012). A description of resampling simulations is omitted as it requires empirical data from existing samples, likely to contain non-neutral regions and an unknown selection regime, from which the simulated data is generated. Given that the general aim of simulation in evolutionary genomics is to model specific modes of selection and their effect on the linked neutral background (Barton, 2000; Charlesworth, 2006; Charlesworth & Jensen, 2021; Kaushik, 2023; Smith & Haigh, 1974), this work will instead focus on coalescent and forward simulations.

Forward simulations model population genetic change across successive generations as it advances forward in time. This allows for the exploration of complex events with high levels of ecological and evolutionary realism throughout the simulation, as demographic and selective events can vary across time and space and act heterogeneously across the genome (Bank et al., 2014). As forward simulations simulate all individuals across successive generations, they can be both time and memory-intensive, particularly because not all individuals contribute to the subsequent generation and, consequently, most simulated individuals/genomes are not ancestral to those sampled in the present (Bank et al., 2014; Hoban et al., 2012). In contrast, coalescent simulations actively reduce computational burden by working backwards in time and only model the genetic ancestry of individuals directly ancestral to a fixed set of individuals sampled in the present. Thus, coalescent simulations start from the sampled individuals and build genealogical trees backwards in time, connecting individual DNA sequences with ancestral lineages until all branches have coalesced to form a single root sequence, which signifies the most recent common genetic ancestor of all sampled individuals (Wakeley, 2009). While coalescent models did not accommodate recombination initially, the ancestral recombination graph (ARG) is a form of the coalescent model that not only includes recombination by allowing for standard ancestry coalescence (i.e. two homologous sequences meeting at a common ancestor) but also recombination-based coalescence (i.e. two contiguous genetic sequences annealing onto a common ancestral background) (Griffiths, 1991; Griffiths & Marjoram, 1997). This results in multiple different genealogies across large DNA sequences, such as chromosomes, each of which represents the ancestry of a distinct recombined segment (Griffiths, 1991; Griffiths & Marjoram, 1997; Kelleher et al., 2016; Wakeley, 2009). This method of genealogical back-tracking means that coalescent simulations are faster and often more efficient than those that move forward in time, as they use an idealized population model to generate the genetic ancestry of individuals that does not require simulating the genetic history of whole populations across all generations (Kelleher et al., 2016; Yuan et al., 2012). Coalescent models also generally assume that new mutations arise independently of the underlying genealogies, which allows mutations to be overlaid once the genealogy has been generated (Kelleher et al., 2018; Wakeley, 2009). However, while this replicates the expected behaviour of neutral mutations, it is not the case for those that are under selection (Braverman et al., 1995; Hudson & Kaplan, 1988; Kaplan et al., 1988). Accordingly, while it is possible to simulate selection in the coalescent framework, the available models are limited in complexity, with most coalescent simulators only considering selection at a single locus (e.g. (Baumdicker et al., 2022; Ewing & Hermisson, 2010; Kern & Schrider, 2016; Shlyakhter et al., 2014)).

1.2 | The tree sequence

The "succinct tree sequence" is a data structure that captures the genealogy of individuals and recombination events (shown in Figure 1), similar to an Ancestral Recombination Graph (ARG) (Haller et al., 2019; Kelleher et al., 2016). It can be used in both coalescent and forward simulations and is an efficient way to simulate, store and analyse genealogical and genetic variation data (Haller et al., 2019; Kelleher et al., 2016, 2018, 2019). Several simulators use this data structure (Baumdicker et al., 2022; Haller et al., 2019; Haller & Messer, 2023; Thornton, 2014), notably the widely used coalescent simulator *msprime* (Baumdicker et al., 2022) and forward simulator *SLiM* (Haller & Messer, 2023).

The tree sequence data structure consists of four tables (Figure 1b): a node table, which contains haploid genome information including ID and age; an edge table, that records the branches between distinct ancestor and child nodes (haploid genomes); a site table, which records the ancestral state of sites in the simulated sequence; and a mutation table, which records the position, the derived state and the first node to inherit each mutation that occurs during the simulation. The tree sequence tables are filled backwards in time in coalescent simulations (Kelleher et al., 2016) and chronologically for forward simulations, iterating over all individuals in each generation (Kelleher et al., 2018). For forward simulations, the tree sequence is simplified periodically, removing records of branches that terminate prematurely; this eliminates redundant node and edge information, improving memory efficiency and ensuring the final tree sequence only contains information relevant to the final population. The efficiency can be further enhanced by delaying the storage of neutral sequence data in the tree sequence. When filling the tree sequence tables using simulations, only non-neutral mutations that will affect the resulting genealogy need to be accounted for, as neutral mutations do not affect genealogical relationships and thus would only slow the simulation with additional data recording and storage burden. Neutral diversity can be added onto the final tree sequence



(b)

Node Table

ID	Time
0	0
1	0
2	0
3	1
4	2

Edge Table

Left	Right	Parent	Child
0	5	4	0
5	10	3	0
0	5	3	1
5	10	4	1
0	10	3	2
0	10	4	3

Site Table

ID	Position	Ancestral State
0	1	A
1	4	C
2	8	T

Mutation Table

ID	Site	Node	Derived State
0	0	3	T
1	1	2	G
2	2	0	A

(c)

```
#OUT: 3 3 SS p1 3
Mutations:
0  0  m2  1  1  0.6  p1  1  3
1  1  m1  4  0  0.5  p1  2  1
2  2  m1  8  0  0.5  p1  2  1
Individuals:
p1:i0  H  p1:0   p1:3
p1:i1  H  p1:1   p1:4
p1:i2  H  p1:2   p1:4
Genomes:
p1:0  A  2
p1:1  A  0
p1:2  A  0  2
p1:3  A  0
p1:4  A
```

FIGURE 1 Diagram of genealogy with associated output tree sequence and non-genealogical data. (a) Genealogies across a genomic sequence and through time, with time units on the *y*-axis in generations and branches coloured by the ancestry of the numbered nodes. Mutations are shown by crosses (x) and labelled with their associated ID found in the subsequent tables. Each genealogy derives from a contiguous segment of the genome, with the corresponding genomic coordinates shown on the *x*-axis. The haploid genomes of each node are shown below, with segments coloured according to their branch. Each node is labelled on the right-hand side of the segment. New mutations introduced to that node are shown in black, whereas inherited mutations are shown in grey. Recombination events are also depicted (//). (b) The tree sequence output consists of a node table, edge table, site table and mutation table. The values included in these tables correspond to the illustrated genealogy in (a). This data structure is coloured red or purple throughout the study, depending on the incorporation of neutral mutation. Panel a and b are adapted from Kelleher et al. (2018). (c) An example of non-genealogical output in the form of standard *SLiM* output. This data structure is coloured in turquoise throughout the study. The header contains information about the output, these values are (left to right): the time (tick and cycle) that the output corresponds to, whether it is an output of the full population (A) or just a sample (SS), the population sampled and the number of individuals. This is followed by mutation information. Each line is a unique mutation with the associated values being (left to right): a within-file numeric identifier, mutation ID, mutation type, position, selection coefficient, dominance coefficient, the population the mutation arose in, the time it arose and a frequency count. In this example, mutation 0 is under selection, while the others are neutral. The mutation section is followed by information on sampled individuals. This includes the individual ID, the sex if enabled (in this case H for hermaphrodite), and the identifier for the two haploid genomes of the individual. Finally, the genome section contains the genome ID, followed by the chromosome type (A for autosomal), and the identifiers of the mutations in the genome.

conditional on the local genealogy (i.e. neutral mutations drawn onto branches of the tree after it has been formed) (Haller et al., 2019; Kelleher et al., 2018). Additionally, summary statistics are efficiently calculable over the tree sequence and these calculations—which utilize the topology and branch lengths of the tree sequence—are often faster than those using allele frequencies (Baumdicker et al., 2022; Haller et al., 2019; Kelleher et al., 2016). The tree sequence approach stands in contrast to standard forward simulations, which do not retain genealogical relationship information for sampled individuals (Haller & Messer, 2022; Hudson, 2002; Kern & Schrider, 2016). An example of this non-genealogical format is shown in Figure 1c.

In this study, simulations were run using both coalescent and forwards-in-time simulators to evaluate the efficiency of these methods and their use of tree sequence recording compared to standard simulation practices. Five methods were compared in the generation of a burn-in to establish equilibrium levels and patterns of neutral diversity (see Figure 2); the burn-in was then used as the initial population data for the forward simulation of selection. The primary aim of these simulations is to examine the temporal impact of evolutionary processes on the neutral diversity within a population, necessitating the sampling of individuals at multiple timepoints throughout the forward simulation. As a test model, we implement both single- and

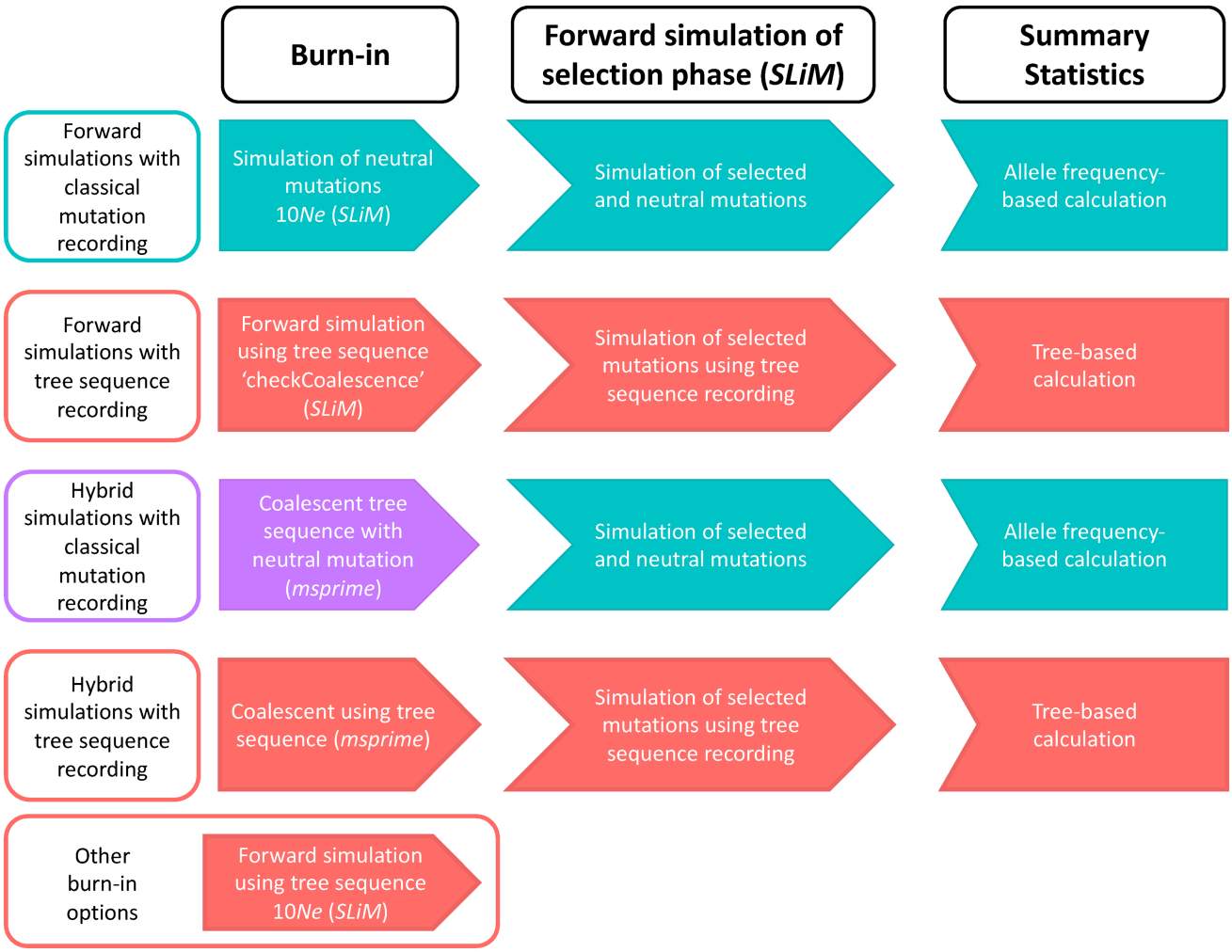


FIGURE 2 Workflow of the benchmarking tests. Four primary workflows were examined: The first workflow entailed a classical forward simulation, encompassing a burn-in phase followed by a selection simulation where both neutral and non-neutral mutations were generated. Summary statistics were subsequently derived from the resulting allele frequency data. The second workflow, akin to the first, employed forward simulation for both the burn-in and selection simulation phases; however, tree sequence recording was utilized and only non-neutral mutations were simulated. This allowed the use of the 'checkCoalescence' option in *SLiM* for the burn-in, which guarantees coalescence of all local genealogies across the genome. All simulations in these two workflows were executed solely in *SLiM*. The third and fourth workflows adopted a hybrid approach, commencing with a coalescent burn-in conducted in *msprime*. In the third workflow, neutral mutations were overlaid on the resulting trees and then used to initiate a classical forward simulation. The fourth approach followed the coalescent burn-in with a forward simulation in *SLiM* utilizing tree sequence recording. Summary statistics were calculated using both tree sequence and allele frequency-based calculations. Additionally, a burn-in comprising a forward simulation with tree sequence recording for $10N_e$ generations, where N_e is the effective population size, was also conducted to directly compare the classical and tree sequence approach under the widely used $10N_e$ criterion. The delineated boxes are colour-coded based on the data type employed: turquoise for non-genealogical data format, red for the tree sequence data structure and purple for the tree sequence accompanied by neutral mutation information.

multi-locus models of fluctuating selection—a dynamic selection model wherein selection pressures vary over time. This model is challenging to execute with standard simulation software and is introduced as the evolutionary process of interest in these forward simulations (Figure 2). Each aspect of the simulation framework was compared using tree sequence recording and classical non-genealogical (i.e. sequence-based) methods of simulating data, which simulate neutral mutations in real time rather than overlaying them onto the genealogy after the simulation is complete. Furthermore, the calculation of population genetic statistics using functions that utilize the tree sequence data structure was compared to those that require allele frequency information. Finally, the complete simulation framework was evaluated by benchmarking the time and memory usage across various combinations of simulation programs, data types and summary statistic calculations. While the classical and tree sequence approaches have been benchmarked previously (Haller et al., 2019), hybrid approaches using a coalescent burn-in and forward simulation with selection have not yet been compared.

2 | METHODS

All simulations were conducted on a MacBook Pro (2.4 GHz 8-Core Intel Core i9 with 64 GB of RAM), using python 3.8.12.

2.1 | Simulated population context

For all simulations, a 1 megabase (Mb) chromosomal region was simulated, with evolutionary and population genetic parameters reflecting those of a wild *Drosophila melanogaster* population. *D. melanogaster* was chosen because *Drosophilids* have been the focal taxon of research into the population genetics of many different types of selection (Sella et al., 2009), including temporally fluctuating selection (Behrman & Schmidt, 2022; Machado et al., 2021; Nunez et al., 2022; Rudman et al., 2022; Wittmann et al., 2017), a form of selection that displays complex dynamics not easily implemented using standard population genetic simulation software. As *D. melanogaster* has a very large estimated effective population size (N_e), on the order of 1 million individuals (Sprengelmeyer et al., 2020), population parameters were downscaled in the simulations, and recombination rate (r) and mutation rate (μ) parameters proportionately upscaled. This conserves key compound parameters (e.g. $2N_e r$ and $4N_e \mu$) while expediting run times by avoiding simulating and storing genetic data of 1 million individuals each generation. Accordingly, rescaled population sizes of 10,000 individuals, r of 10^{-6} and μ of 10^{-7} were simulated to model a *Drosophila* population with a N_e equal to 1 million, r of 10^{-8} (Comeron et al., 2012) and μ of 10^{-9} (Keightley et al., 2014; Schrider et al., 2013) respectively. When scaling population parameters, the selection coefficient must also be appropriately scaled. Hence, we use a scaled selection coefficient of 1 that is equivalent to a selection pressure two orders of magnitude weaker in a natural population (i.e. unscaled $s = 0.01$).

2.2 | Simulation workflows

Three different aspects of a typical population genetic simulation project were evaluated (Figure 2): (1) the burn-in, to establish an equilibrium level of neutral diversity in the population; (2) the forward simulation, in which selection is acting; and (3) the computation of summary statistics. This study utilizes the coalescent simulator *msprime* (v. 1.2.0; (Baumdicker et al., 2022)) and the forward simulator *SLiM* (v. 4.0.1; (Haller & Messer, 2023)) in conjunction with the python programming libraries: *pySLiM* (v. 1.0), to process the tree sequences; and *tskit* (v. 0.5.2), to analyse the tree sequences (Baumdicker et al., 2022; Kelleher et al., 2018; Ralph et al., 2020). *SLiM* was chosen due to its flexibility, which allows users to customize the simulation by scripting specific evolutionary and demographic events using a range of functions that target these aspects. Similarly, *msprime*, *PySLiM* and *tskit* are used as they also work with tree sequences and have been integrated with *SLiM*.

To conduct benchmarking, the memory and time requirements of *msprime* and *SLiM* are compared, as well as the efficiency of using the tree sequence data structure compared to the classical forward approach. Memory usage was measured using the python package *tracemalloc*. As this package cannot measure the memory usage of *SLiM*, the peak RAM usage of *SLiM* simulations was measured using an internal function, 'usage (peak = T)', in the software. Following the *SLiM* tree sequence recording benchmarks presented in Haller et al. (Haller et al., 2019), we conducted 10 replicates of each simulation. Examination of the results revealed that the inter-replicate variance was sufficiently small relative to the absolute values to provide adequately robust estimates (Tables S1–S3).

2.3 | Burn-in

Burn-in simulations are commonly employed in population genetic models to establish equilibrium levels of neutral population genetic diversity prior to the commencement of the phase of empirical interest (e.g. the fluctuating selection phase in the present study). The difference in resource usage was compared for five different burn-in scenarios that combined the two types of simulations, that is, the *msprime* coalescent simulator or *SLiM* forward simulations, and two recording scenarios, that is, with neutral mutations being directly recorded throughout the simulations or tree sequence recording used instead (see Figure 2).

When running burn-ins in *SLiM*, classical forward simulations (i.e. those utilizing in-time neutral mutation recording) were run for 100,000 generations, since the number of generations required to achieve sequence-wide coalescence and stable levels of neutral diversity is approximately 10 times the effective population size (N_e) (Messer, 2013; Wakeley, 2009). However, recent work suggests that simulating for $10N_e$ generations is not sufficient to ensure coalescence (Haller et al., 2019; Haller & Messer, 2022) and, accordingly, *SLiM* has introduced an option (*checkCoalescence* = *T*) to check if all

lineages have coalesced across the full length of the simulated sequence. Hence, *SLiM* simulated burn-ins that used tree sequence recording were run by either simulating for $10N_e$ generations or until all lineages had confirmed coalescence. Each burn-in approach was replicated 10 times.

Diversity was measured for all burn-in simulations to evaluate if the levels of neutral population genetic variation match theoretical expectations (Wakeley, 2009); that is, $\frac{\theta}{1+\theta}$ where θ is $4N_e\mu$, and μ is the per-generation mutation rate. For *msprime* simulations and *SLiM* tree sequence output, *tskit's* diversity function was used to calculate nucleotide diversity. For the classical simulation approach, diversity was calculated directly in *SLiM* based on allele frequency data using the 'calcHeterozygosity' function. The normality of the diversity distributions for each simulation type was examined using the Shapiro–Wilk test. Because the diversity distribution of some simulation types was not strictly Gaussian, we conducted both the t-test and the nonparametric Mann–Whitney U-test to examine if the mean significantly deviates from the neutral expectation. Relative diversity, calculated by dividing the diversity value by the neutral theoretical expectation ($\frac{\theta}{1+\theta}$), was then compared between the different simulation approaches.

2.4 | Forward simulation selection models

Next, we evaluated the resource usage of *SLiM* simulations of a complex selection regime—namely, a stable population of 10,000 individuals experiencing a fluctuating selection pressure that oscillates across two seasons, with 10 generations per season—comparing the same two recording options that were used in the burn-ins (i.e. either tree sequence recording or the classical in-time mutation recording).

We explored two models of fluctuating selection, in which selection targeted either a single locus or multiple loci. Both models were replicated 10 times for each data recording method. To simulate single locus selection, we implemented the model proposed by Wittmann and colleagues, who derived fitness equations for genotypes at a seasonal locus (Table 1; (Wittmann et al., 2023)). For all simulations conducted using this model, we set the selection coefficient to 1 and the dominance value to 0.6 across both seasons.

For our simulations of multi-locus fluctuating selection, we used the following two-part fitness function (Wittmann et al., 2017):

$$z_x = n_x + d_x \times n_{\text{het}} \quad (1)$$

$$\omega(z) = (1+z)^y \quad (2)$$

The first equation is a score (z) that combines the effects of favoured alleles at seasonal loci in a given season (the x subscript in Equation 1 indicates that the season could be either summer or winter). This score comprises the number of loci homozygous for the seasonal allele (n_x) plus the product of the number of sites that are heterozygous for the seasonal allele (n_{het}) and the dominance of that allele (d_x). This score is then incorporated in a fitness function ($\omega(z)$; Equation 2), where y is a coefficient that accounts for epistasis between seasonal loci. For this model, 10 seasonal loci are simulated, each with a dominance value of 0.6 and a y value of 4 for both seasons as this was shown to lead to stable polymorphism (Wittmann et al., 2017).

TABLE 1 Fitness equations from Wittmann et al. 2023 used to simulate seasonally fluctuating selection at a single locus.

Season	ω_{WW}	ω_{SW}	ω_{SS}
Winter	$1+s_w$	$1+h_w s_w$	1
Summer	1	$1+h_s s_s$	$1+s_s$

Note: A fitness equation was used to calculate an individual's fitness (ω) depending on the season (either summer or winter) and the genotype at the seasonal locus (i.e. either *WW*, *SW* or *SS*, where *W* denotes the winter-adapted allele and *S* the summer-adapted allele), by assuming a selection coefficient (s) and a dominance coefficient (h) for each seasonal allele.

For simulated selection models using a specific recording type (i.e. classical mutation recording or tree sequencing recording), the final population from the burn-in step was used as the initial population for the selection phase. All forward simulations of fluctuating selection were run in *SLiM* for $4N_e$ (40,000) generations, as this provided sufficient time for allele frequencies to reach stable oscillations and is the time required for the stabilization of genome-wide patterns of neutral diversity under the influence of fluctuating selection (Wittmann et al., 2023). To observe changes in genetic variation over time as selection exerts its effect on linked diversity, 100 individuals were sampled in the first generation after the burn-in, and then every 10,000 generations throughout the simulation. In the final generation, the whole population was sampled. For simulations using tree sequence recording, resource usage calculations also included importing the final tree sequence into python and overlaying neutral mutations with *msprime*, resulting in neutral and non-neutral mutation information at the conclusion of both simulation workflows.

2.5 | Calculation of summary statistics

The resource usage involved in the calculation of summary statistics was also tested, with comparisons of functions that either utilize the tree sequence data structure or those that use allele frequency information for estimation. Calculations were performed using *tskit* (v. 0.5.2) (Baumdicker et al., 2022; Kelleher et al., 2018; Ralph et al., 2020) functions on tree sequence data and the python package *scikit-allel* (v. 1.3.5) (Miles et al., 2021) on allele frequency data from the 10 replicates of each selection model and data recording combination. Tajima's D and nucleotide diversity statistics were calculated using both *tskit* and *scikit-allel*, as both packages have functions to calculate these statistics. The peak memory usage and time required for the calculation of each

statistic were recorded as well as the total time, including necessary reformatting steps.

3 | RESULTS

Population genomic simulation is an important tool for exploring and testing the dynamics and consequences of complex evolutionary phenomena. Given the large number of simulations typically required for robust quantification and testing, efficient simulation frameworks are key. Here, a combination of modern simulation approaches are compared to benchmark workflows for population genetic studies of non-standard forms of natural selection.

3.1 | Burn-in to establish neutral diversity

When comparing the computational efficiency of *msprime* and *SLiM* to run a neutral burn-in, the former was considerably faster than when using either tree sequence recording ($10N_e$ generations or until coalescence is confirmed) or the classical mutation recording approach in *SLiM*. *msprime* burn-ins were 40 times faster than those conducted in *SLiM* when simulating for $10N_e$ generations and using

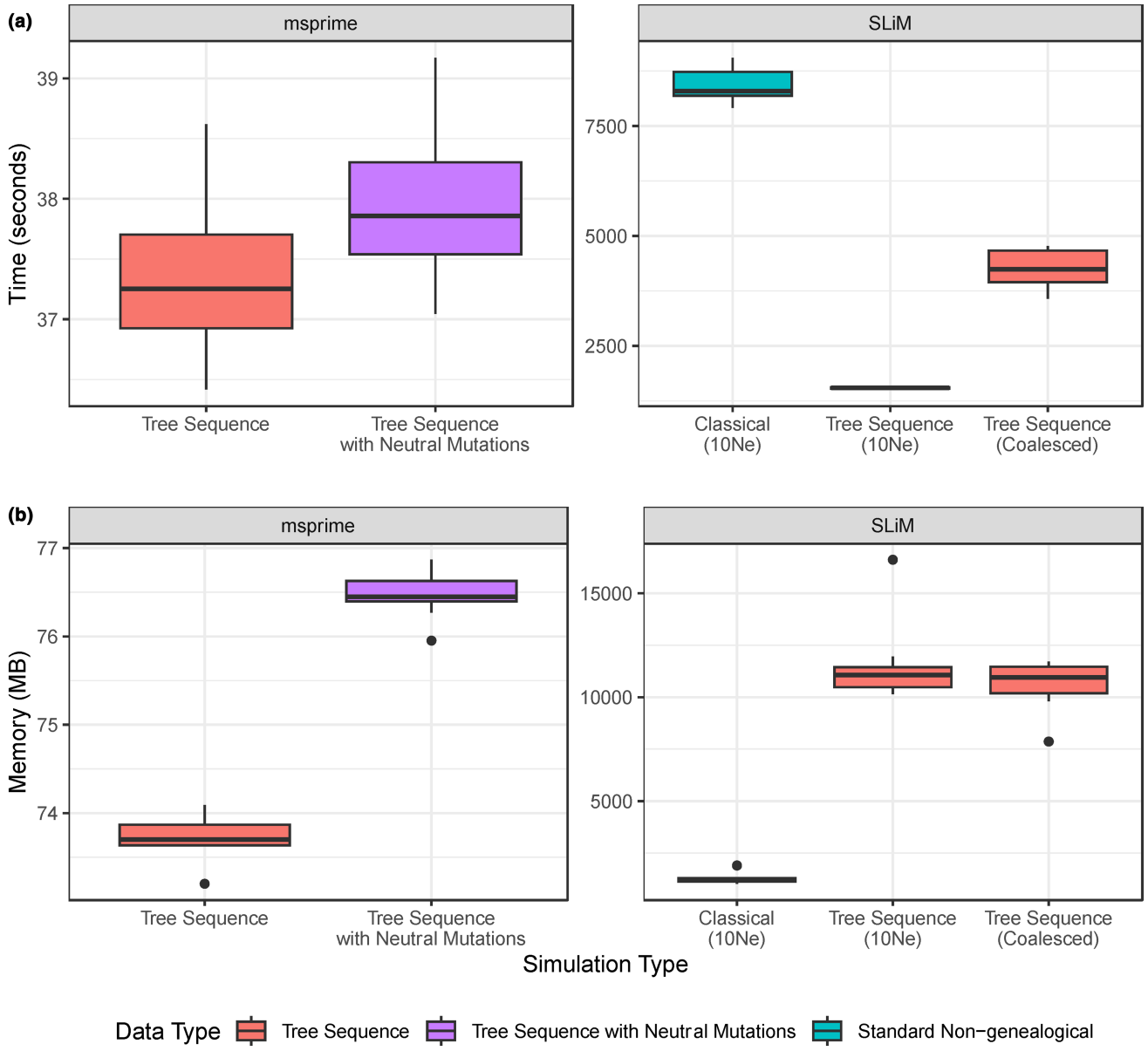


FIGURE 3 Resource usage for burn-in simulations. Simulations that use standard non-genealogical output are shown in turquoise, red signifies simulations that output a tree sequence and purple are simulations resulting in a tree sequence that is overlaid with neutral mutations. Each approach was replicated 10 times. (a) Time in seconds for simulations to complete, comparing the use of the tree sequence data structure, in *msprime* with and without overlaid neutral mutations as well as in *SLiM* run for $10N_e$ generations and until coalescence is ensured, and using the classical approach. (b) Memory usage in megabytes (MB) for these same simulations.

tree sequence recording (Figure 3a; Table S1). Overlaying neutral mutations had a negligible impact on runtime, with *msprime* being able to generate ~260,000 mutations in less than a second, including file input and output (Baumdicker et al., 2022). The runtime difference between *msprime* and *SLiM* approaches increased to almost 114 times when using the 'checkCoalescence' option for tree sequence recording in *SLiM*, which ensures that all lineages have coalesced across the simulated sequence. In this case, simulations ran for an average of 283,840.5 generations, and coalescence was reached in no fewer than 240,189 generations. Notably, this is substantially more (~180% increase) than the value of $10N_e$ that is typically recommended for reaching an equilibrium state in burn-in simulations (Haller & Messer, 2022), suggesting that the coalescence checking option should be used to ensure robust burn-ins in *SLiM*. *msprime* was over 220 times faster than *SLiM* when using the classical in-time mutation recording approach, consistent with previous benchmarking results showing that forward approaches only outcompete *msprime* when extremely long sequences (i.e. 10^{10} nucleotides) are simulated (Haller et al., 2019).

The memory required for *msprime* simulations was also significantly lower compared to simulating forwards in time with *SLiM* (Figure 3b), irrespective of the recording method used. *msprime* had a peak memory consumption of about 76 MB when running a burn-in and overlaying neutral mutations. This was almost 140 times lower than the mean peak memory of the *SLiM* burn-in using tree sequence recording, which required an average of about 10.6 GB whether the 'checkCoalescence' option was activated or not. The use of in-time mutation recording resulted in significant reductions in memory burden relative to tree sequence recording, requiring around 1.3 GB, although this is still 16× more memory than that consumed by *msprime* using in-time mutation recording. This difference in memory consumption was also found when the tree sequence recording method was first proposed (Kelleher et al., 2018). However, the initial benchmarking found that tree sequence recording consumed less memory compared to the classical mutation recording approach (Haller et al., 2019). This disparity is potentially due to less frequent usage of tree simplification in the current study, as there is a trade-off between the frequency of simplification events and resource usage (i.e. increasing the frequency of simplification events leads to greater computational time but lower memory consumption (Haller et al., 2019)).

When comparing levels of diversity (Figure 4), the coalescent simulations in *msprime* resulted in levels of neutral diversity close to the theoretical expectation (relative diversity = 0.996; *p*: Shapiro–Wilk <.05, *t*-test >.05, Mann–Whitney *U*-test >.05). Forwards simulations in *SLiM* result in slightly decreased levels of diversity when 'checkCoalescence' is not implemented, such that the mean levels of neutral diversity when simulating for $10N_e$ generations with tree sequence recording (0.966; *p*: Shapiro–Wilk <.05, *t*-test <.001, Mann–Whitney *U*-test <.01) and using the classical mutational recording

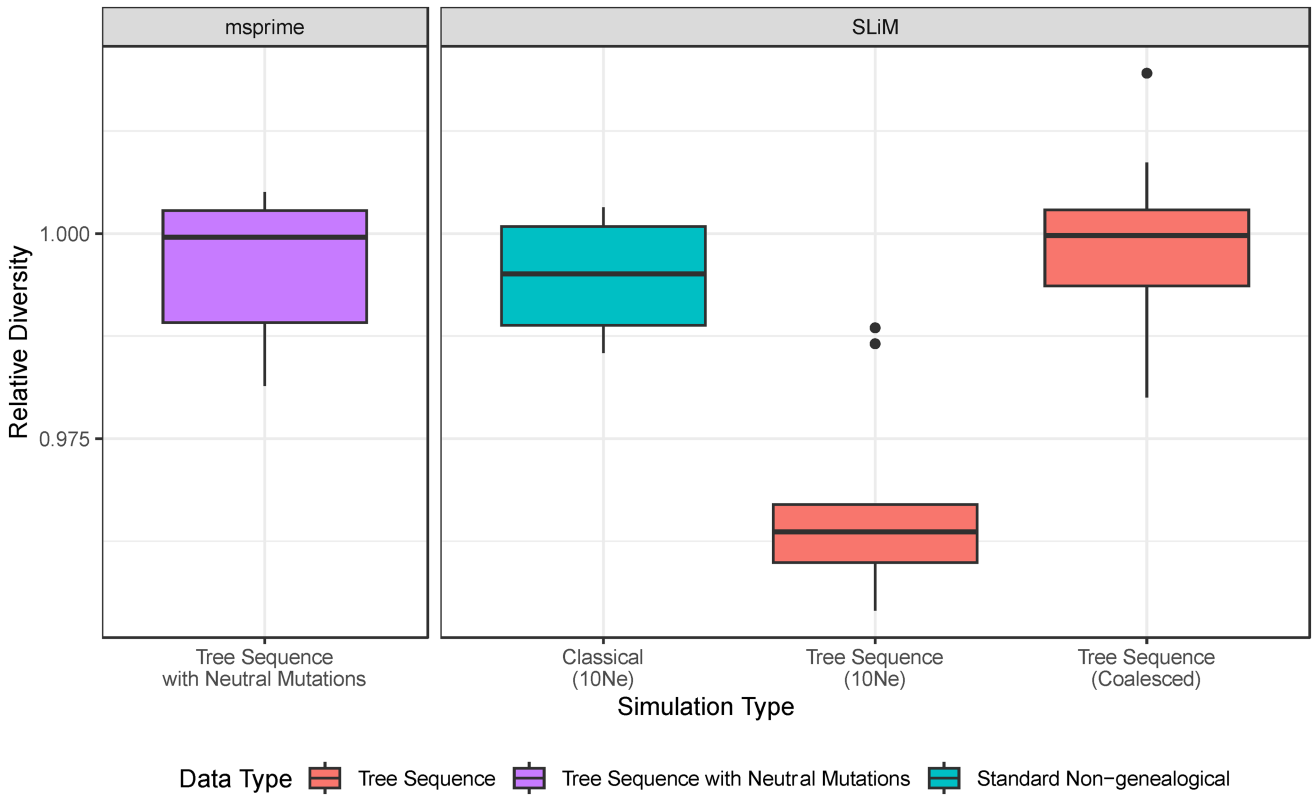


FIGURE 4 Relative diversity of burn-in simulations. Relative diversity was calculated to compare the levels of neutral diversity at the end of each burn-in simulation with expected diversity levels for the simulated population under neutral evolution (expectation is equal to $\frac{\theta}{1+\theta}$). Coalescent simulations generated in *msprime* were compared with forward simulations that were generated in *SLiM*.

approach (0.994; *p*: Shapiro–Wilk >.05, *t*-test <.05) were significantly less than the neutral expectation. However, using tree sequence recording and conditioning on coalescence results in a mean diversity that is similar to the neutral expectation (0.999; *p*: Shapiro–Wilk >.05, *t*-test >.05). This reinforces the importance of ensuring coalescence when running burn-ins with tree sequence recording in *SLiM*. Notably, coalescent approaches such as *msprime* guarantee coalescence across the full simulated sequence, which together with the minimal resource usage makes it the preferred method for producing burn-ins to establish neutral diversity. However, should a non-neutral burn-in be required, for example, to evaluate changing selection pressures or continuously introduced deleterious mutations, then forward simulations are required and users need to decide between the time-efficient but memory-heavy tree sequence recording (confirming coalescence) or the slower but more memory-efficient classical mutation reporting approach.

3.2 | Forward simulation of fluctuating selection

Following the benchmarking of the burn-in methods, the resource usage of the forward simulation of fluctuating selection in SLiM was tested. Here, classical forward simulations utilizing in-time mutation reporting and forward simulation with recently developed tree sequence recording methods are compared, and both the single locus and multi-locus models were implemented.

For the forward simulations of the single locus selection model, the classical approach took over 12 times longer (approximately 2h

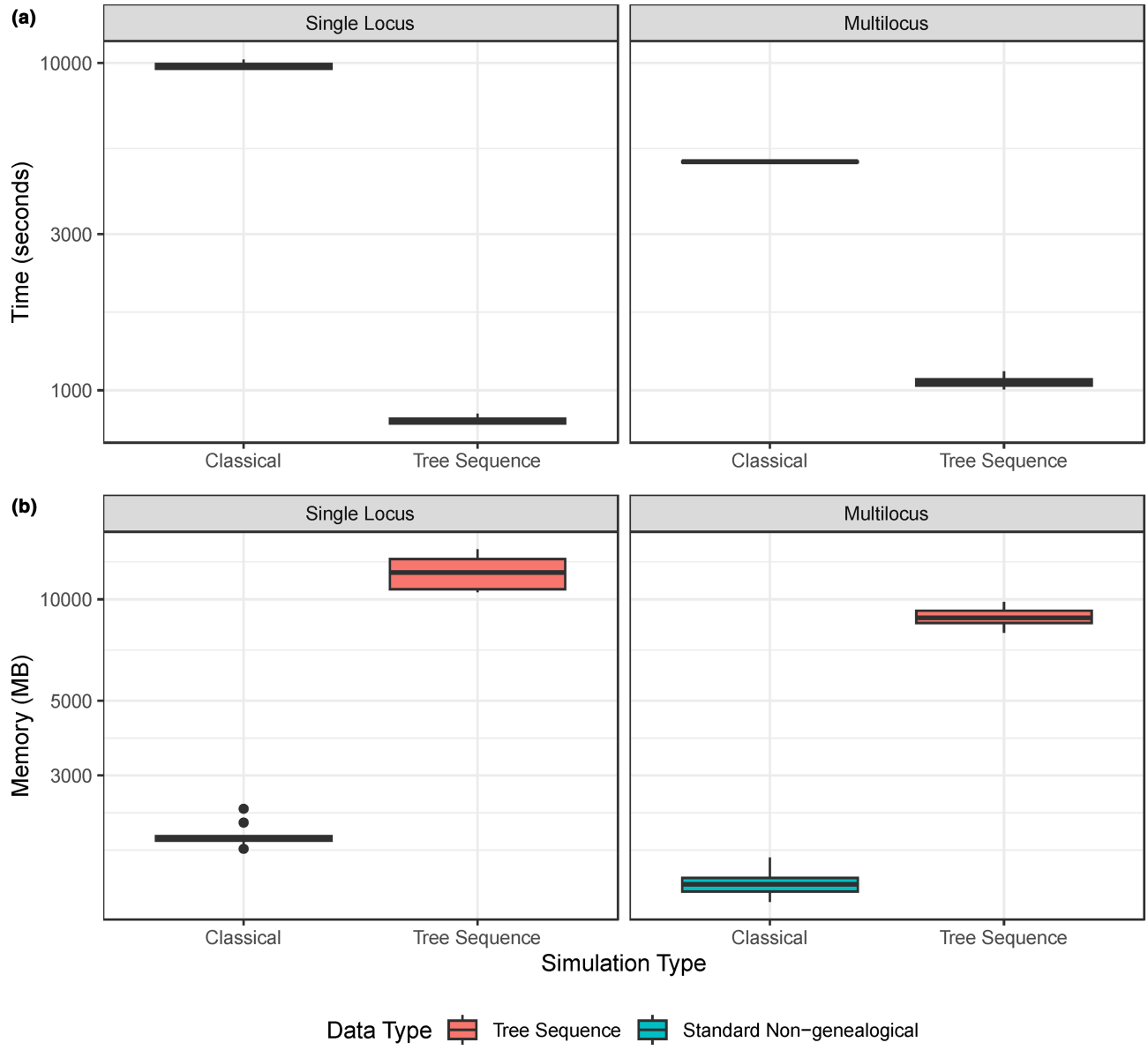


FIGURE 5 Resource usage for forward simulations modelling selection. (a) Time and (b) memory requirements for forward simulations in *SLiM* on a log scale. Two models of fluctuating selection, a single locus and a multi-locus model, are compared, along with the use of the tree sequence recording (red) compared to classical forward simulation (i.e. using in-time mutational reporting; turquoise).

and 43 min; Table S2; Figure 5a) than using tree sequence recording and overlaying neutral mutations afterwards (~13.5 min). The runtime of the classical approach decreased when simulating multi-locus selection, taking an average of 1 h and 23 min versus approximately 18 min when using tree sequence recording. This time improvement provided by tree sequence recording does come with an increase in memory burden (Figure 5b), requiring approximately six times more memory (12.1 and 8.8 GB for single locus and multi-locus models respectively) than forward simulations employing in-time mutation reporting (1.9 and 1.4 GB). The developers of tree sequence recording do note that it can be more memory intensive and thus is not always advisable (Haller et al., 2019; Kelleher et al., 2018). However, when the required RAM is available, the reduction in the time required for the simulation to run, combined with the additional data captured in the tree sequence data structure, can make tree sequence recording more desirable than classical forward simulations.

3.3 | Summary statistics

Given the benchmarking results reported in the previous sections, a simulation set-up that utilized a coalescent burn-in in *msprime* and forward simulation of selection using tree sequence recording was determined to be the fastest approach. Previous studies indicate that the use of the tree sequence data structure can also facilitate the rapid calculation of population genetic statistics (Baumdicker et al., 2022; Kelleher et al., 2018). Accordingly, we performed tree- or allele-based calculations on the relevant outputs of our simulated fluctuating selection regime and compared computation times and memory usage (Table S3). The tree-based statistics required less memory than the allele-based calculations, exhibiting an 80% average reduction in RAM usage for the single locus selection model, and a 50% average reduction for the multi-locus selection model (Figure 6a). In contrast, the tree-based calculations had the longest mean compute time, ranging from 0.3 to 1.2 s for the four combinations of statistics and selection models (Figure 6b). The allele-based calculations were significantly faster, taking less than a hundredth of a second. However, this calculation time does not take into account the time required to manipulate that data into the form required by the statistical function. The tree-based calculation can be applied directly to the tree sequence resulting from the forward simulation, whereas the allele-based statistics require a specific array of allele counts and a separate vector of variant positions. When the time taken to prepare these data formats is included, allele- and tree-based calculations have similar compute times (Figure 6c).

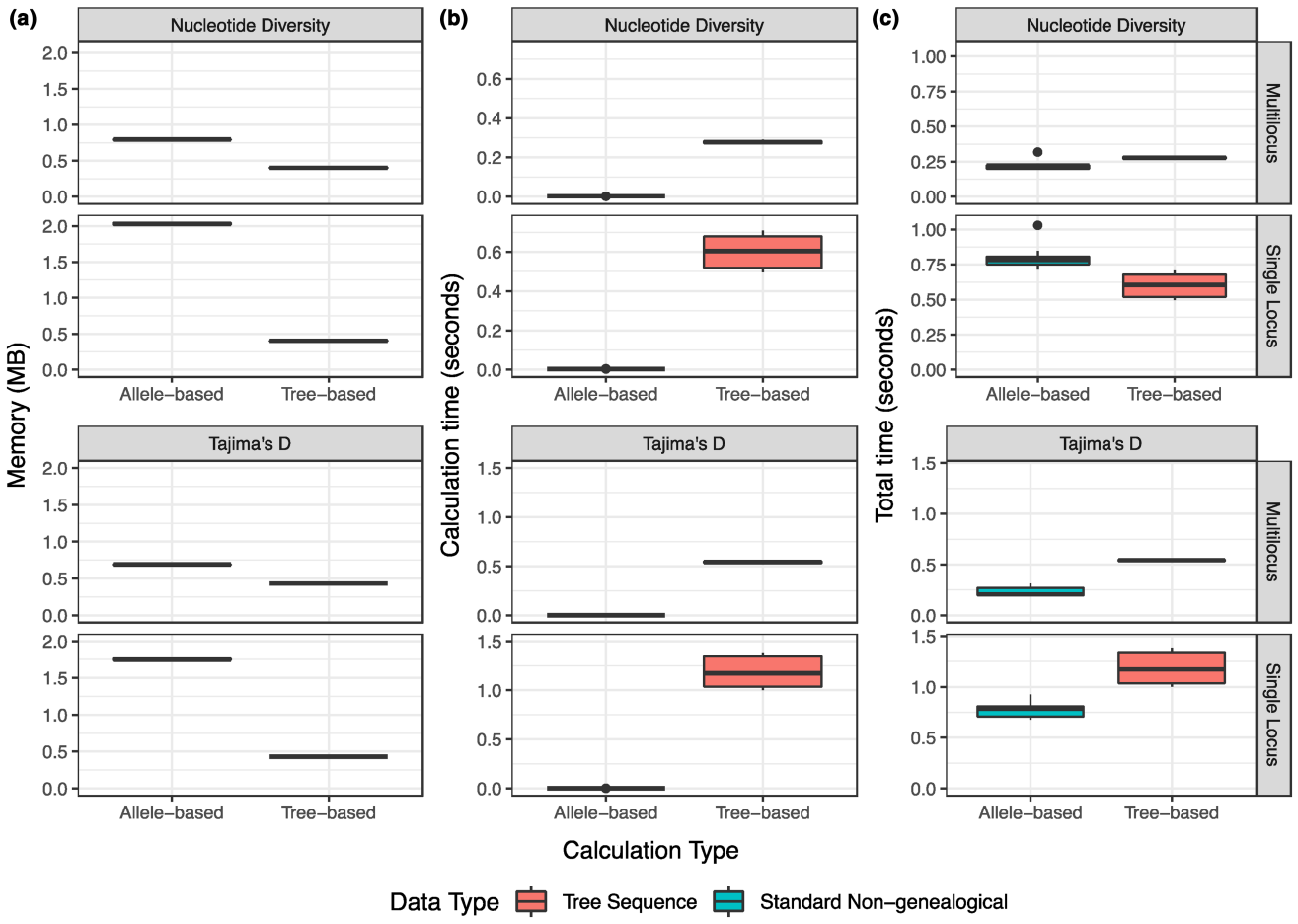


FIGURE 6 Resource requirements for analysis of simulated data. Comparisons of (a) memory, (b) calculation time and (c) total time taken including data manipulation for combinations of models of selection and calculation type for two common population genetic statistics, nucleotide diversity and Tajima's D.

3.4 | The complete simulation framework

We combined the outcomes of the burn-in, selection phase (single allele model), and summary statistic calculations (nucleotide diversity), to obtain benchmarks for the four tested simulation frameworks (Figure 2; Table S4): 1, classical forward simulations using in-time mutation recording during the burn-in and selection phases with allele-based calculations; 2, forward simulations using tree sequence recording in the burn-in (generating a robust burn-in using the 'checkCoalescence' option to ensure coalescence of lineages) and selection phase with tree-based calculations; 3, a coalescent burn-in overlaid with neutral mutations followed by a selection phase with in-time mutation recording and allele-based calculations; and 4, a coalescent burn-in followed by a selection phase using tree sequence recording with tree-based calculations of summary statistics. Notably, the complete framework values for the multi-locus method are similar and can be found in Table S4. In contrast, single locus selection models displayed considerable differences with approaches that utilized classical methods taking the longest time to complete, having an average computation time of approximately 5 h when in-time mutation reporting was used for both burn-in and selection phase and taking 2 h and 43 min when it was only used in the selection phase. Significant gains were made by frameworks that used tree sequence recording, with run times decreasing to 85 min when this was employed in the selection phase and only 14 min when employed in both the burn-in and selection phase. These gains in computation incur a trade-off with memory consumption, with frameworks employing tree sequencing recording in at least one step having a maximum memory burden of ~12 GB versus ~2 GB for frameworks where this recording type was not used.

4 | DISCUSSION

SLiM is a powerful population genetic simulation tool that implements tree sequence recording within a forward simulation structure, which has made the evaluation of arbitrarily complex evolutionary scenarios feasible. Previous studies benchmarking *SLiM* simulations have all employed *msprime's* recapitation function, where lineages that remain uncoalesced at the end of the simulation are joined in post-simulation data processing (Haller et al., 2019). Accordingly, population-wide levels of neutral diversity are not fixed at a specific value at the onset of the simulation, making this method unsuitable for research that aims to investigate how the effect of selection on linked neutral variation builds up over time.

In our study, we have explored four options for generating appropriate population-wide patterns of neutral diversity in a pre-simulation burn-in phase, followed by *SLiM* forward simulations of a complex fluctuating selection scenario. Our benchmarks show that a hybrid workflow employing an *msprime* coalescent burn-in phase and conducting the selection phase using *SLiM's* tree reporting procedure provides appropriate levels of initial population genetic diversity as well as delivering rapid simulations. This hybrid framework is between 6 and 21 times faster than the other three options. While the tree sequence recording format does impose higher memory burdens than simulations that employ in-time mutation reporting, these RAM levels are feasible for modern computing environments (~12 GB in the present study). Similarly, summary statistic calculations on tree-based outputs are slightly slower than allele-based methods, although this comprises a small proportion of the overall run time and is not sufficient to offset the time savings created by the usage of tree recording (in place of in-time mutation recording) throughout the simulation.

Another important consideration arising from our study is that the coalescence of neutrally evolving lineages is not guaranteed to occur within $10N_e$ generations, a criterion that is often used in population genetic simulations. While this is not an issue when using coalescent simulators like *msprime*, which ensure genome-wide lineage coalescence, for users choosing to work solely in *SLiM*, we strongly encourage the use of the 'checkCoalescence' option to generate appropriate population levels of neutral diversity at the onset of the simulations.

Taken together our benchmarks demonstrate the hybrid tree sequence workflow—which combines an *msprime* coalescent burn-in with the core simulation conducted in *SLiM*—is an effective framework for the simulation of non-standard evolutionary processes due to the rich information provided by the tree sequence data structure and the potential for greatly reduced computational run times compared to workflows adopting in-time mutation recording. As shown in this study, however, the improved run times afforded by tree sequence recording also come with higher memory requirements, and this trade-off depends on the population genetic model being simulated. Accordingly, the optimal simulation framework for each evolutionary genetic study will ultimately depend on both the simulated population genetic model and resources available in the local computational environment.

AUTHOR CONTRIBUTIONS

All authors designed the simulation and benchmarking framework. O.L.J. set up the simulations. O.L.J. and C.D.H. conducted the analysis of the results. All authors contributed to writing the manuscript.

ACKNOWLEDGEMENTS

C.D.H. was funded by the National Institute of Health under award number R35GM146886, R.T. was funded by Australian Research Council's DECRA Fellowship DE190101069, J.M.S. was funded by the Australian Research Council's Discovery Project DP190103606 and O.L.J. was funded by a Westpac Future Leaders Scholarship and an Australian Government Research Training Program Scholarship. Thank you to the Australian Centre for Ancient DNA's Thesis Writing Group for providing feedback on an early version of this manuscript.

DATA AVAILABILITY STATEMENT

Code to replicate this benchmarking can be found at github.com/olivia-johnson/2023_popgen_benchmarking.git.

BENEFIT-SHARING STATEMENT

This study was conducted in accordance with the principles of the Nagoya Protocol on Access to Genetic Resources and the Fair and Equitable Sharing of Benefits Arising from their Utilization.

ORCID

Olivia L. Johnson https://orcid.org/0000-0001-8029-2397
Raymond Tobler https://orcid.org/0000-0002-4603-1473
Joshua M. Schmidt https://orcid.org/0000-0002-5862-7389
Christian D. Huber https://orcid.org/0000-0002-2267-2604

REFERENCES

Adrion, J. R., Cole, C. B., Dukler, N., Galloway, J. G., Gladstein, A. L., Gower, G., Kyriazis, C. C., Ragsdale, A. P., Tsambos, G., Baumdicker, F., Carlson, J., Cartwright, R. A., Durvasula, A., Gronau, I., Kim, B. Y., McKenzie, P., Messer, P. W., Noskova, E., Ortega-del Vecchyo, D., … Kern, A. D. (2020). A community-maintained standard library of population genetic models. *eLife*, *9*, e54967. https://doi.org/10.7554/eLife.54967

Bank, C., Ewing, G. B., Ferrer-Admettla, A., Foll, M., & Jensen, J. D. (2014). Thinking too positive? Revisiting current methods of population genetic selection inference. *Trends in Genetics*, *30*, 540–546. https://doi.org/10.1016/j.tig.2014.09.010

Barton, N. H. (2000). *Genetic hitchhiking* (Vol. *355*, pp. 1553–1562). Phil Trans R Soc Lond B.

Baumdicker, F., Bisschop, G., Goldstein, D., Gower, G., Ragsdale, A. P., Tsambos, G., Zhu, S., Eldon, B., Ellerman, E. C., Galloway, J. G., Gladstein, A. L., Gorjanc, G., Guo, B., Jeffery, B., Kretzschumar, W. W., Lohse, K., Matschiner, M., Nelson, D., Pope, N. S., … Kelleher, J. (2022). Efficient ancestry and mutation simulation with msprime 1.0. *Genetics*, *220*, 220. https://doi.org/10.1093/genetics/iyab229

Behrman, E. L., & Schmidt, P. (2022). How predictable is rapid evolution? *bioRxiv*. https://doi.org/10.1101/2022.10.27.514123

Braverman, J. M., Hudson, R. R., Kaplan, N. L., Langley, C. H., & Stephan, W. (1995). The hitchhiking effect on the site frequency spectrum of DNA polymorphisms. *Genetics*, *140*, 783–796. https://doi.org/10.1093/genetics/140.2.783

Charlesworth, B., & Jensen, J. D. (2021). Effects of selection at linked sites on patterns of genetic variability. *Annual Review of Ecology and Systematics*, *52*, 177–197. https://doi.org/10.1146/annurev-ecolsys-010621-044528

Charlesworth, D. (2006). Balancing selection and its effects on sequences in nearby genome regions. *PLoS Genetics*, *2*, e64. https://doi.org/10.1371/journal.pgen.0020064

Comeron, J. M., Ratnappan, R., & Bailin, S. (2012). The many landscapes of recombination in Drosophila melanogaster. *PLoS Genetics*, *8*, e1002905. https://doi.org/10.1371/journal.pgen.1002905

Ewing, G., & Hermisson, J. (2010). MSMS: A coalescent simulation program including recombination, demographic structure and selection at a single locus. *Bioinformatics*, *26*, 2064–2065. https://doi.org/10.1093/bioinformatics/btq322

Fisher, R. A. (1930). *The genetical theory of natural selection*. The Clarendon Press.

Griffiths, R. C. (1991). The two-locus ancestral graph. *Lecture Notes-Monograph Series*, *18*, 100–117. http://www.jstor.org/stable/4355649

Griffiths, R. C., & Marjoram, P. (1997). An ancestral recombination graph. In *Progress in population genetics and human evolution* (pp. 257–270). Springer https://ui.adsabs.harvard.edu/abs/1997IMA....87..257G

Haldane, J. B. S. (1926). A mathematical theory of natural and artificial selection. *Mathematical Proceedings of the Cambridge Philosophical Society*, *23*, 363–372. https://doi.org/10.1017/S0305004100015176

Haller, B. C., Galloway, J., Kelleher, J., Messer, P. W., & Ralph, P. L. (2019). Tree-sequence recording in SLiM opens new horizons for forward-time simulation of whole genomes. *Molecular Ecology Resources*, *19*, 552–566. https://doi.org/10.1111/1755-0998.12968

Haller, B. C., & Messer, P. W. (2017). SLiM 2: Flexible, interactive forward genetic simulations. *Molecular Biology and Evolution*, *34*, 230–240. https://doi.org/10.1093/molbev/msw211

Haller, B. C., & Messer, P. W. (2019). SLiM 3: Forward genetic simulations beyond the Wright–Fisher model. Hernandez R, editor. *Molecular Biology and Evolution*, *36*, 632–637. https://doi.org/10.1093/molbev/msy228

Haller, B. C., & Messer, P. W. (2022). *SLiM: An evolutionary simulation framework*. Cornell University https://github.com/MesserLab/SLiM/releases/download/v4.0.1/SLiM_Manual.pdf

Haller, B. C., & Messer, P. W. (2023). SLiM 4: Multispecies eco-evolutionary modeling. *The American Naturalist*, *201*, E127–E139. https://doi.org/10.1086/723601

Hoban, S., Bertorelle, G., & Gaggiotti, O. E. (2012). Computer simulations: Tools for population and evolutionary genetics. *Nature Reviews. Genetics*, *13*, 110–122. https://doi.org/10.1038/nrg3130

Hudson, R. R. (2002). Generating samples under a Wright-Fisher neutral model of genetic variation. *Bioinformatics*, *18*, 337–338. https://doi.org/10.1093/bioinformatics/18.2.337

Hudson, R. R., & Kaplan, N. L. (1988). The coalescent process in models with selection and recombination. *Genetics*, *120*, 831–840. https://doi.org/10.1093/genetics/120.3.831

Isildak, U., Stella, A., & Fumagalli, M. (2021). Distinguishing between recent balancing selection and incomplete sweep using deep neural networks. *Molecular Ecology Resources*, *21*, 2706–2718. https://doi.org/10.1111/1755-0998.13379

Kaplan, N. L., Darden, T., & Hudson, R. R. (1988). The coalescent process in models with selection. *Genetics*, *120*, 819–829. https://doi.org/10.1093/genetics/120.3.819

Kaushik, S. (2023). Effect of beneficial sweeps and background selection on genetic diversity in changing environments. *Journal of Theoretical Biology*, *562*, 111431. https://doi.org/10.1016/j.jtbi.2023.111431

Keightley, P. D., Ness, R. W., Halligan, D. L., & Haddrill, P. R. (2014). Estimation of the spontaneous mutation rate per nucleotide site in a Drosophila melanogaster full-sib family. *Genetics*, *196*, 313–320. https://doi.org/10.1534/genetics.113.158758

Kelleher, J., Etheridge, A. M., & McVean, G. (2016). Efficient coalescent simulation and genealogical analysis for large sample sizes. *PLoS Computational Biology*, *12*, e1004842. https://doi.org/10.1371/journal.pcbi.1004842

Kelleher, J., Thornton, K. R., Ashander, J., & Ralph, P. L. (2018). Efficient pedigree recording for fast population genetics simulation. *PLoS Computational Biology*, *14*, e1006581. https://doi.org/10.1371/journal.pcbi.1006581

Kelleher, J., Wong, Y., Wohns, A. W., Fadil, C., Albers, P. K., & McVean, G. (2019). Inferring whole-genome histories in large population datasets. *Nature Genetics*, *51*, 1330–1338. https://doi.org/10.1038/s41588-019-0483-y

Kern, A. D., & Schrider, D. R. (2016). Discoal: Flexible coalescent simulations with selection. *Bioinformatics*, *32*, 3839–3841. https://doi.org/10.1093/bioinformatics/btw556

Korfmann, K., Temple-Boyer, M., Sellinger, T., & Tellier, A. (2023). Determinants of rapid adaptation in species with large variance in offspring production. *Molecular Ecology*. https://doi.org/10.1111/mec.16982

Machado, H. E., Bergland, A. O., Taylor, R., Tilk, S., Behrman, E., Dyer, K., Fabian, D. K., Flatt, T., González, J., Karasov, T. L., Kim, B., Kozeretska, I., Lazzaro, B. P., Merritt, T. J. S., Pool, J. E., O'Brien, K., Rajpurohit, S., Roy, P. R., Schaeffer, S. W., … Petrov, D. A. (2021). Broad geographic sampling reveals the shared basis and environmental correlates of seasonal adaptation in drosophila. *eLife*, *10*, e67577. https://doi.org/10.7554/eLife.67577

Matheson, J., & Masel, J. (2023). Background selection from unlinked sites causes non-independent evolution of deleterious mutations. *bioRxiv*. https://doi.org/10.1101/2022.01.11.475913

Messer, P. W. (2013). SLiM: Simulating evolution with selection and linkage. *Genetics*, *194*, 1037–1039. https://doi.org/10.1534/genetics.113.152181

Miles, A., Bot P io, R. M., Ralph, P., Harding, N., Pisupati, R., et al. (2021). cggh/scikit-allel: v1.3.3. https://doi.org/10.5281/zenodo.4759368

Nunez, J. C. B., Lenhart, B. A., Bangerter, A., Murray, C. S., Yu, Y., Nystrom, T. L., et al. (2022). A cosmopolitan inversion drives seasonal adaptation in overwintering drosophila. *bioRxiv*. https://doi.org/10.1101/2022.12.09.519676

Ralph, P., Thornton, K., & Kelleher, J. (2020). Efficiently summarizing relationships in large samples: A general duality between statistics of genealogies and genomes. *Genetics*, *215*, 779–797. https://doi.org/10.1534/genetics.120.303253

Rudman, S. M., Greenblum, S. I., Rajpurohit, S., Betancourt, N. J., Hanna, J., Tilk, S., Yokoyama, T., Petrov, D. A., & Schmidt, P. (2022). Direct observation of adaptive tracking on ecological time scales in *drosophila*. *Science*, *375*, eabj7484. https://doi.org/10.1126/science.abj7484

Schrider, D. R., Houle, D., Lynch, M., & Hahn, M. W. (2013). Rates and genomic consequences of spontaneous mutational events in Drosophila melanogaster. *Genetics*, *194*, 937–954. https://doi.org/10.1534/genetics.113.151670

Sella, G., Petrov, D. A., Przeworski, M., & Andolfatto, P. (2009). Pervasive natural selection in the drosophila genome? *PLoS Genetics*, *5*, e1000495. https://doi.org/10.1371/journal.pgen.1000495

Shlyakhter, I., Sabeti, P. C., & Schaffner, S. F. (2014). Cosi2: An efficient simulator of exact and approximate coalescent with selection. *Bioinformatics*, *30*, 3427–3429. https://doi.org/10.1093/bioinformatics/btu562

Smith, J. M., & Haigh, J. (1974). The hitch-hiking effect of a favourable gene. *Genetical Research*, *23*, 23–35. https://doi.org/10.1017/S0016672300014634

Sprengelmeyer, Q. D., Mansourian, S., Lange, J. D., Matute, D. R., Cooper, B. S., Jirle, E. V., Stensmyr, M. C., & Pool, J. E. (2020). Recurrent collection of Drosophila melanogaster from wild African environments and genomic insights into species history. *Molecular Biology and Evolution*, *37*, 627–638. https://doi.org/10.1093/molbev/msz271

Thornton, K. R. (2014). A C++ template library for efficient forward-time population genetic simulation of large populations. *Genetics*, *198*, 157–166. https://doi.org/10.1534/genetics.114.165019

Wakeley, J. (2009). *Coalescent theory: An introduction*. Roberts & Company.

Wittmann, M. J., Bergland, A. O., Feldman, M. W., Schmidt, P. S., & Petrov, D. A. (2017). Seasonally fluctuating selection can maintain polymorphism at many loci via segregation lift. *Proceedings of the National Academy of Sciences*, *114*, E9932–E9941. https://doi.org/10.1073/pnas.1702994114

Wittmann, M. J., Mousset, S., & Hermisson, J. (2023). Modeling the genetic footprint of fluctuating balancing selection: From the local to the genomic scale. *Genetics*, *223*, iyad022. https://doi.org/10.1093/genetics/iyad022

Wright, S. (1929). The evolution of dominance. *The American Naturalist*, *63*, 556–561. https://doi.org/10.1086/280290

Yuan, X., Miller, D. J., Zhang, J., Herrington, D., & Wang, Y. (2012). An overview of population genetic data simulation. *Journal of Computational Biology*, *19*, 42–54. https://doi.org/10.1089/cmb.2010.0188

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

How to cite this article: Johnson, O. L., Tobler, R., Schmidt, J. M., & Huber, C. D. (2024). Population genetic simulation: Benchmarking frameworks for non-standard models of natural selection. *Molecular Ecology Resources*, *00*, e13930. https://doi.org/10.1111/1755-0998.13930