Assessing the universality of Bolthausen-Sznitman coalescent genealogies in models of rapid selection

Internship report

Tristan Godart
ENS Paris-Saclay
tristan.godart@ens-paris-saclay.fr

Supervised by Fabian Freund
University of Leicester
ff95@leicester.ac.uk

2025

Résumé

Comprendre l'évolution génétique des populations est un objectif central en biologie évolutive et en génétique. La théorie des coalescents constitue à ce titre un cadre théorique puissant : il s'agit d'un modèle rétrospectif de génétique des populations permettant de modéliser l'histoire évolutive d'un échantillon d'individus en remontant dans le temps jusqu'à leur dernier ancêtre commun. Elle permet de simplifier la complexité des processus évolutifs en un arbre généalogique, ce qui facilite l'étude de phénomènes complexes comme la dérive génétique ou les goulots d'étranglement. La théorie des coalescents est de plus largement utilisée pour inférer des paramètres démographiques à partir de données génétiques réelles, comme les tailles de population passées ou les événements de sélection.

L'un des principaux intérêts des coalescents réside dans le fait que de nombreux modèles discrets de génétique des populations convergent, lorsque la taille N de la population tend vers l'infini, vers des modèles de coalescents continus. C'est en particulier le cas pour le modèle de Wright-Fisher, qui converge vers le coalescent de Kingman [Kin82a], ou pour certains modèles avec sélection comme les modèles de Brunet qui convergent vers le coalescent de Bolthausen et Sznitman [Bru+06b]. De nombreux autres modèles présentent également des résultats de convergence similaires, soulignant la portée théorique du cadre des coalescents.

Toutefois, ces résultats de convergence sont presque exclusivement démontrés pour des populations haploïdes, comme les virus [TLK96; Kes+97; TL14], certaines cellules isolées [SS16], ou des pathogènes [BBS14], dans lesquels les individus héritent leur génome d'un seul parent. Cela restreint considérablement l'applicabilité directe de ces modèles à des populations plus complexes, notamment animales ou végétales [SG14], sauf à se limiter à l'étude de séquences uniparentales spécifiques comme l'ADN mitochondrial [Che+95; Wil96] ou certaines régions du chromosome Y [Ham95; JT95; WSG95].

Cette distinction entre haploïdes et diploïdes devient encore plus marquée dans le cadre de modèles avec sélection comme ceux de Brunet. Chez les diploïdes, l'information génétique est répartie sur deux chromosomes, qui peuvent porter des allèles distincts avec des effets sélectifs différents, ce qui pose des enjeux supplémentaires liés à la dominance ou la récessivité.

L'objectif de ce stage est donc d'évaluer, par des simulations numériques, si des résultats de convergence analogues à ceux de [Bru+06b] peuvent être observés dans des modèles de populations diploïdes soumis à sélection.

Enfin, afin d'exploiter pleinement la richesse du modèle diploïde, nous proposerons un nouveau modèle incorporant des mutations en tenant compte de la dominance ou de la récessivité des mutations, pour mieux modéliser la transmission et la sélection de l'information génétique lors de la reproduction.

Abstract

Understanding the genetic evolution of populations is a central goal in evolutionary biology and genetics. In this context, coalescent theory provides a powerful theoretical framework: it is a retrospective model in population genetics that describes the evolutionary history of a sample of individuals by tracing their lineages back in time to their most recent common ancestor. This approach simplifies the complexity of evolutionary processes into a genealogical tree, making it easier to study complex phenomena such as genetic drift or population bottlenecks. Moreover, coalescent theory is widely used to infer demographic parameters from real genetic data, such as past population sizes or selective events.

One of the main advantages of coalescent models lies in the fact that many discrete models of population genetics converge, as the population size N tends to infinity, towards continuous coalescent processes. This is particularly the case for the Wright-Fisher model, which converges to the Kingman coalescent [Kin82a], or for certain selection models such as the Brunet models, which converge to the Bolthausen–Sznitman coalescent [Bru+06b]. Many other models also exhibit similar convergence results, highlighting the broad theoretical relevance of the coalescent framework.

However, the convergence results towards the Bolthausen-Sznitman coalescent have been demonstrated almost exclusively for haploid populations, such as viruses [TLK96; Kes+97; TL14], certain isolated cells [SS16], or pathogens [BBS14], in which individuals inherit their genome from a single parent. This considerably limits the direct applicability of such models to more complex populations, particularly animal or plant populations [SG14], except when focusing on specific uniparental sequences such as mitochondrial DNA [Che+95; Wil96] or certain regions of the Y chromosome [Ham95; JT95; WSG95].

This distinction between haploid and diploid organisms becomes even more significant in the context of selection models such as those introduced by Brunet and colleagues. In diploids, genetic information is split between two chromosomes, which may carry different alleles with distinct selective effects, raising additional challenges related to dominance and recessiveness.

The goal of this internship is therefore to investigate, through numerical simulations, whether convergence results similar to those of [Bru+06b] can be observed in models of diploid populations under selection.

Finally, to fully leverage the complexity of the diploid model, we propose a new framework incorporating mutations and accounting for the dominance or recessiveness of mutations, in order to better capture the transmission and selection of genetic information during reproduction.

Contents

Aknowledgements I would like to deeply thank my mentor Fabian Freund for welcoming me in the best conditions. He truly was the best advisor I could have hoped for. I also want to thank Danny, Isra and Nawel, three other great friends who made me enjoy my sojourn in England.

Keywords Population genetics, Kingman coalescent, Markov process, Bolthausen-Sznitman coalescent, Simulations, Genealogy models, Diploid genealogy, Mutation models, Random partitions

MSC 60J90 Coalescent processes, 92B05 General biology and biomathematics, 92-10 Mathematical modeling or simulation for problems pertaining to biology, 92D15 Problems related to evolution.

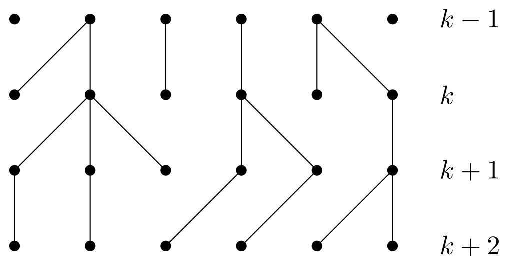


Figure 1: Example of a population of $N = 6$ individuals under the Wright-Fisher model, for 4 generations.

1 Introduction

1.1 The Wright-Fisher model

For population genetics, branching process models fail to carry conviction because, in biological reality, the total population size depends mainly on external factors (availability of food, living space area, quantity of predators, ...) rather than on variables inherent to the individuals [Kin82a]. One of the first approximations in population genetics is therefore to fix the total population size to a number given by external constraints. The most important model with this axiom was developed implicitly by Fisher [FIS22] in 1922 and explicitly by Wright [Wri31] in 1931. It is mainly used to describe the *genetic drift* of a population, i.e. the evolution of this population caused by purely random events, with no selective pressure.

The Wright-Fisher model assumes discrete, non-overlapping generations $(G_k)_{k\in\mathbb{N}}$. Each generations contains N individuals, all equally likely to reproduce. We only consider haploid reproduction for now, where each individual $i \in G_k$ has a single parent in G_{k-1}. Its number of children in G_{k+1} is however a random variable $\nu_{k,i}$, subject to the constraint :

$$\forall k \in \mathbb{N}, \quad \sum_{i=1}^{N} \nu_{k,i} = N. \tag{1}$$

Because each individual has the same *fitness* (i.e. they same likelihood to reproduce), $\nu_{k,j} \overset{\mathcal{D}}{=} \nu$ is equally distributed for all individuals. In fact, ν follows a symmetric multinomial distribution :

$$\mathbb{P}\left(\forall i \in [\![1, N]\!], \nu_i = n_i\right) = \frac{N!}{n_1! \dots n_N!} N^{-N}. \tag{2}$$

This process is particularly convenient because it can be interpreted as a "backwards" model where each individual in G_k chooses its parents uniformly at random in G_{k-1}, for each generation $k \in \mathbb{N}^*$. This way, we can easily extend the Wright-Fisher model to generations indexed by negative integers. We thus consider from now on the sequence $(G_k)_{k\in\mathbb{Z}}$ of generations.

If $i, j \in G_k$ are two distinct individuals in a given generation G_k, we denote by the random variable $\tilde{T}_2^N(i,j)$ the smallest integer m such that i and j share a common ancestor in G_{k-m}. Note that this quantity is always well-defined thanks to the last paragraph. We moreover have the independence of $\tilde{T}_2^N(i,j_0)$ and $\tilde{T}_2^N(i,j_1)$ whenever $j_0 \neq j_1$, as j_0 and j_1 choose their parent independently. It can be shown by induction that $\tilde{T}_2^N(i,j) \sim \mathcal{G}eom(\frac{1}{N})$ for each couple (i,j) of

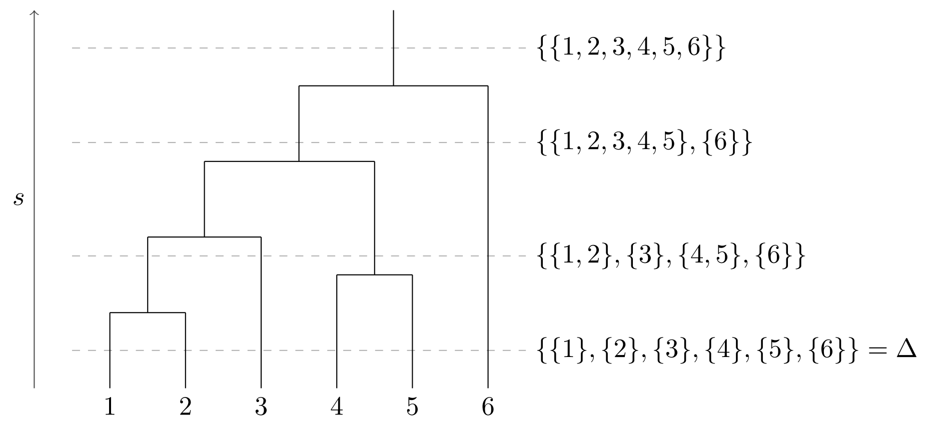


Figure 2: Example of Kingman coalescent with a sample of $n = 6$ individuals.

distinct individuals of a given generation. This gives the following result :

$$\forall k \in \mathbb{Z}, \forall i \neq j \in G_k, \quad \mathbb{E}[\tilde{T}_2^N(i,j)] = N. \tag{3}$$

This means that $\tilde{T}_2^N(i,j)$ is on average of the order of N. This defines a natural time unit of N generations that Kingman used in [Kin82a] as the cornerstone of his coalescent model.

1.2 Kingman's coalescent theory

Fix r large enough so that the whole G_r has a common ancestor in G_0, and sample $n \leqslant N$ individuals from G_r. The family tree of these n individuals and their ancestors may be described as a sequence $(C_k)_{0 \le k \le r}$ of partitions of the set $[\![1, n]\!]$, such that two individuals i and j of the generation G_r are in the same block of C_s if and only if they have a common ancestor in generation G_{r-s} (see Figure 2).

This way, each block of C_s corresponds to a member of G_{r-s}. Two such members may have the same parent in G_{r-s-1}, in which case their blocks are combined in C_{s+1}. If their parents are not the same individual, then the blocks are not combined.

It follows that the sequence $(C_s)_{s\in\mathbb{N}}$ forms a Markov chain. We denote by $p_{\xi\eta} = \mathbb{P}(C_{s+1} = \eta \mid C_s = \xi)$ its transition probabilities. Note that we always have

$$C_0 = \Delta := \{\{i\} \mid i \in [\![1, n]\!]\}. \tag{4}$$

We write $\xi \prec \eta$ if η can be derived from ξ by merging two of its blocks. For the purpose of this report, there is no need to compute $p_{\xi\eta}$ in detail, since it is enough to remember that $p_{\xi\eta}$ is of order N^{-2} unless $\xi = \eta$ or $\xi \prec \eta$. In fact, Kingman computed in [Kin82a] that

$$p_{\xi\eta} = \delta_{\xi\eta} + \frac{q_{\xi\eta}}{N} + O(N^{-2}), \tag{5}$$

where $\delta_{\xi\eta}$ is the Kronecker delta, and

$$q_{\xi\eta} = \begin{cases} -\frac{1}{2}|\xi|(|\xi|-1) & \text{if } \xi = \eta, \\ 1 & \text{if } \xi \prec \eta, \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

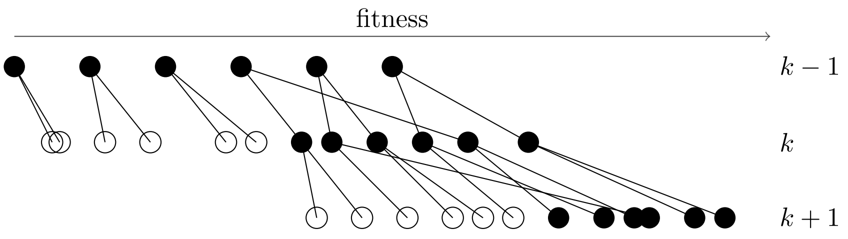


Figure 3: Example of Brunet's model A with a population of $N = 6$ individuals.

where $|\xi|$ is the number of blocks in ξ.

Kingman has shown in [Kin82a] that, as $N \to +\infty$, the rescaled process $(C_s)_{s\in\mathbb{N}}$ converges in distribution to the Markov process whose starting state is Δ and whose infinitesimal generator is given by (6). We call this process the *Kingman coalescent.*

In fact, the Kingman coalescent is central in the theory of neutral evolution because of its universality [Gri80; Tav84; DT95; DS05; BBC08]: the rules of the Wright-Fisher model can be changed in many ways and still recover the Kingman coalescent in the limit of a large population. For example, this applies if the parents are chosen with a non uniform probability, as long as these probabilities decay fast enough when the size N of the population grows large [MS01].

It is important to note that, because of the $\frac{1}{N}$ factor in (5), one unit of time in the Kingman coalescent asymptotically corresponds to N generations in the Wright-Fisher model. This is a callback to the natural time unit of N generations defined earlier in the Wright-Fisher model.

1.3 Brunet models

Similarly to the Wright-Fisher model, Brunet models also consider a population of fixed size N over discrete generations, but this time with selective pressure. The simplest way of implementing selection is to associate to each individual a value - its *fitness* - which dictates how likely it is to have children [Sny03; KT04; Klo05; Bru+06a]. Contrary to its biological definition, the fitness of population genetics models often intervene in selection instead of in reproduction [TLK96; Kes+97].

All of the models of this subsection were introduced by Brunet et al. in [Bru+06b].

Model A In *Model A* (see Figure 3), two steps occur at each generation.

1. Reproduction : each individual gives birth to k children, and the fitness x_j of the offspring j of the individual i is given by :
$$x_j = x_i + \varepsilon_{ij}, \tag{7}$$
where x_i is the fitness of the parent and $(\varepsilon_{ij})_{i,j}$ is a family of i.i.d random variables with distribution $\rho(\varepsilon)$. This way, the fitness is inherited and $\varepsilon_{i,j}$ accounts for the effects of random mutations.

2. Selection : kill every individual out of the kN, except the N offspring with the highest fitness.

In [Bru+06b], Brunet uses $k = 2$ and $\rho(\varepsilon)$ uniform between 0 and 1, so we will use these values from now on. This choice of $\rho(\varepsilon)$ makes it so that the average fitness of the population almost surely grows at each generation, which is not important as its variance stays relatively constant, as we checked during our simulations.

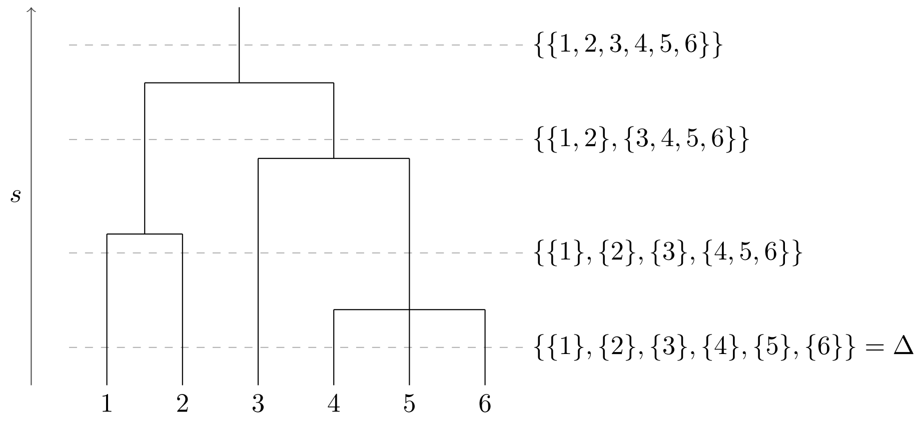


Figure 4: Example of Bolthausen-Sznitman coalescent with a sample of $n = 6$ individuals.

In this model, it is possible that a *superfit* individual (i.e. an individual whose fitness is much higher than any other individuals) appears. In this case, its children, which inherit its fitness, will not die during the selection process. The superfit lineage then escapes completely from the selective pressure for a few generations.

In [Bru+06b], Brunet et al. also investigate *Model A'*, where instead of keeping the N best children at each generation, we keep N children randomly chosen among the $\frac{3}{2}N$ best ones. The results were the same as Model A, meaning that, once rescaled, the fitness of Brunet models can also be interpreted as a *probability* to produce offspring, which is closer to its biological definition.

Model B The last model Brunet introduced in [Bru+06b] is *Model B*, where each individual i has infinitely many potential offspring, whose fitness are distributed according to a Poisson point process of density $\psi(x - x_i)$. As in model A, we only keep the N offspring whose fitness are the highest. We are especially interested in the *exponential model* where $\psi : x \mapsto \exp(-x)$, as it showed a high convergence speed towards the Bolthausen-Sznitman coalescent in [Bru+06b].

1.4 The Bolthausen-Sznitman coalescent

Similarly to the Kingman coalescent, the Λ-*coalescent* is also a Markov process with values in the set of all partitions of $[\![1, n]\!]$. Its transition rates, which describe the rate at which any merger between l branches out of m may occur for any $l, m \in [\![2, n]\!]$, are given by :

$$\lambda_{m,l} := \binom{m}{l} \int_0^1 x^{l-2}(1-x)^{m-l}\Lambda(dx). \tag{8}$$

Note that the rate of the merger that changes ξ into η can also be obtained, and only depends on the number of blocks $|\xi|$ and $|\eta|$:

$$\lambda_{\xi,\eta} := \int_0^1 x^{|\xi|-|\eta|-1}(1-x)^{|\eta|-1}\Lambda(dx). \tag{9}$$

The *Bolthausen-Sznitman coalescent* [BS98] is a particular case of the Λ-coalescent, where Λ is the uniform measure between 0 and 1. Its transition rates can be computed from (8) by integrating by parts, which yields :

$$\lambda_{m,l} = \binom{m}{l} \frac{(l-2)!(m-l)!}{(m-1)!} = \frac{m}{l(l-1)} \qquad \text{and} \qquad \lambda_m = m-1 \tag{10}$$

The main point of relevance of the Bolthausen-Sznitman coalescent in this report is that, in the haploid case, it approximated well after time rescaling the genealogies in Brunet's models for N large [Bru+06b]. It can then be seen as an supplement of the Kingman coalescent for populations subject to selection. The main difference between both models is that in the Bolthausen-Sznitman coalescent, a merger between more than two branches can occur in case a superfit individual appears (see Figure 4), which almost surely never happens in the Kingman coalescent (6).

1.5 Characterization of the genealogical trees

This internship focuses on interpreting data from the simulations of given models of evolution to infer whether the genealogical trees of the populations can be described by a particular coalescent process. These processes can be characterized in many ways [Pit99], but here we will only focus on the *average coalescence time* $\langle T_p \rangle$, which is the average age T_p of the most recent common ancestor of p individuals of age 0 chosen randomly among the population in one generation, with $p \in [\![2, N]\!]$.

One advantage of the coalescence times is that there is no loss of information compared to the genealogical tree. Indeed, the trees store nothing more than the ages $T_2(i,j)$ of the most recent common ancestor of any two distinct individuals i, j of a given generation. We can then easily compute $\langle T_p \rangle$ for any $p \geqslant 3$ with (11), as Brunet et al. did in [Bru+06b].

$$T_p(i_1, \dots, i_p) = \max_{2 \leqslant l \leqslant p} T_2(i_1, i_l). \tag{11}$$

However, one of the major drawbacks of this approach is that, for $T_2(i,j)$ to be properly defined for each couple (i,j) of individuals of a given generation, the whole population needs to have a common ancestor in G_0 (we say that the tree has *coalesced*). This means that the simulations need to run for $\tilde{T}_N^N$ generations before we can extract data from the tree. Most of the models we have simulated have no explicit bound for $\tilde{T}_N^N$, so when necessary we used the heuristic [HM16, p.43] of $10N$ generations to make sure that the trees have coalesced, which is sometimes not enough.

Another issue with this approach is that the results will be neither a full proof nor a full evidence for the convergence. This is because we only look at a feature of the models and we are not really considering the case $N \to +\infty$, as N is finite for every one of our simulations. Finally, we are not able to assess the full distribution of the trees as we are just simulating a few realizations for each model.

In this report, we will only look $\langle T_3 \rangle / \langle T_2 \rangle$. This is because we realized that the values of $\langle T_p \rangle / \langle T_2 \rangle$ for $p \geqslant 4$ always lead to the same conclusions as $\langle T_3 \rangle / \langle T_2 \rangle$ for each model and for each population size N, except that they are more irregular and seem to converge slower towards the theoretical coalescent values. Furthermore the computation of $\langle T_p \rangle$ is computationally slower when p is larger.

Our interpretation will mainly be focused around the two coalescent models introduced above, the Kingman coalescent (no apparent selection) and the Bolthausen-Sznitman coalescent (apparent selection).

Average coalescence time of the Kingman coalescent The expected time $\mathbb{E}[T_p]$ of the most recent common ancestor of a sample of p individuals of the population in a Kingman coalescent can be computed by iterating on mergers [Tav+97]. This can be done by introducing i.i.d random variables $(W_k)_{2\leqslant k\leqslant p}$, where W_k depicts the time for any two branches among k to merge in the Kingman coalescent. This way, since at most two branches can merge at a time (5), we have :

$$\mathbb{E}[T_p] = \sum_{k=2}^{p} \mathbb{E}[W_k] \tag{12}$$

Moreover, it can be inferred from (6) that $\forall k \in [\![2,p]\!], W_k \sim \mathcal{E}xp(\frac{k(k-1)}{2})$, which gives us :

$$\mathbb{E}[W_k] = \frac{2}{k(k-1)}. \tag{13}$$

Finally, we get from (12) and (13) that :

$$\mathbb{E}[T_p] = \sum_{k=2}^{p} \frac{2}{k(k-1)} = \sum_{k=2}^{p} \left(\frac{2}{k-1} - \frac{2}{k}\right) = 2\left(\frac{p-1}{p}\right) \tag{14}$$

With a time unit of N generations, and considering the case $p = 2$, this confirms the result obtained in (3).

The first values of p are the most interesting for this report. For them, (14) yields :

$$\mathbb{E}[T_2] = 1, \quad \mathbb{E}[T_3] = \frac{4}{3}, \quad \mathbb{E}[T_4] = \frac{3}{2}, \quad \mathbb{E}[T_5] = \frac{8}{5}, \quad \dots \tag{15}$$

However, we need to keep in mind that we are not directly concerned about the coalescence times in and of themselves, but instead in their rescaling by $\mathbb{E}[T_2]$. Indeed, the relevance of the Kingman coalescent comes from the fact that a lot of discrete models converge to it when properly rescaled. To check if our discrete models converge towards it, we then need to rescale them properly. This rescaling can also be interpreted as a way to deal with the lack of dimension of the coalescent, which cannot be properly measured in generations, or as a way to stop the average coalescence times $\langle T_p\rangle$ from growing with the population size N. To characterize the Kingman coalescent, we will then check if we have these results on the average coalescence times :

$$\frac{\langle T_3\rangle}{\langle T_2\rangle} \underset{N\to+\infty}{\longrightarrow} \frac{4}{3}, \quad \frac{\langle T_4\rangle}{\langle T_2\rangle} \underset{N\to+\infty}{\longrightarrow} \frac{3}{2}, \quad \frac{\langle T_5\rangle}{\langle T_2\rangle} \underset{N\to+\infty}{\longrightarrow} \frac{8}{5}, \quad \dots \tag{16}$$

The average coalescence times therefore yield two main interpretations: $\langle T_2\rangle$ contains informations about the timescale of the sampled tree, which allows us to rescale and to get the values $\langle T_p\rangle/\langle T_2\rangle$, which themselves provide information on the underlying coalescent model.

Average colascence time of the Bolthausen-Sznitman coalescent The expected time $\langle T_p\rangle$ of the most recent common ancestor of a sample of the whole population in a Bolthausen-Sznitman coalescent can once again be computed by iterating on mergers. This time however, we need to take into account that mergers with more than two branches are possible (8).

We introduce the family of random variables $(\mathcal{T}_k)_{0\leqslant k\leqslant N}$, where $\mathcal{T}_k \sim \mathcal{E}xp(\lambda_k)$ depicts the time for any subset of branches among k to merge for the Bolthausen-Sznitman coalescent. We have the induction formula [FM09] :

$$\forall n \in [\![2,N]\!], \quad \mathbb{E}[T_n] = \mathbb{E}[\mathcal{T}_n] + \sum_{k=2}^{n} \frac{\lambda_{nk}}{\lambda_n} \mathbb{E}[T_{n-k+1}] \tag{17}$$

For the Bolthausen-Sznitman coalescent, we can plug in the values computed in (10), which yields :

$$\forall n \in [\![2, N]\!], \quad \mathbb{E}[T_n] = \frac{1}{n-1} + \sum_{k=2}^{n} \frac{n(n-1)}{k(k-1)} \mathbb{E}[T_{n-k+1}] \tag{18}$$

This formula allows to compute $\mathbb{E}[T_p]$ recursively, which is sufficient for the purposes of this internship.

Once again, we need to rescale by $\mathbb{E}[T_2]$ to compare empirical data with theoretical results. This yields the results of convergence needed to characterize the Bolthausen-Sznitman :

$$\frac{\langle T_3 \rangle}{\langle T_2 \rangle} \underset{N \to +\infty}{\longrightarrow} \frac{5}{4}, \quad \frac{\langle T_4 \rangle}{\langle T_2 \rangle} \underset{N \to +\infty}{\longrightarrow} \frac{25}{18}, \quad \dots \tag{19}$$

The time-scale $\langle T_2 \rangle$ of the Bolthausen-Sznitman coalescent has been shown [Bru+06b] to be of the order of $\log(N)^\alpha$, with $\alpha > 0$.

1.6 Diploid models

The models mentioned above are mainly defined and studied for *haploid* populations, where the individuals have only one chromosome in their cells, like bacteria for instance. These individuals reproduce *asexually*, which means every individual only has one parent, and its fitness is mainly based on only one individual's fitness.

Diploid individuals, like animals for example, have two chromosomes and reproduce *sexually*. They have two parents (one male and one female), and their fitness is a mix of both their parents's. For our simplest models, we chose to make the fitness of a child a function of the fitnesses of its two parents, with a random noise ε added. The implementation of diploid models from haploid ones will be explained in more details in Section 2.3.

In reality, each parent does not contribute equally to a child's fitness because of the dominance effect of some alleles. Fortunately, this difference can be handled by the following mutations models.

The main question of this internship is to inquire with simulations whether diploid models with selection converge towards a coalescent model when the size N of the population tends towards infinity, similarly to what has been done in [Bru+06b] for haploid populations. This question is far from trivial, because the fitness of a superfit individual dilutes way faster in diploid populations, making superfit lineages less abiding and completely changing the population behavior.

1.7 Mutations model

A more realistic approach to fitness-based models is to use *mutations*, random changes in the genetic code that can affect the fitness of the individual. Mutations occur randomly with a specified rate during the reproduction process, and can be used to replace the random noise ε which makes the fitness of a child different from its parent's.

One advantage of mutations is that, contrary to random noises ε, they are actually stored in the genetic code. For diploid populations, their effect can be amplified or reduced via their *dominance coefficient* h, which dictates how much the mutation is expressed when it is present in one chromosome only: if the mutation makes individuals gain a fitness of s, then an individual who has only one copy of this mutations gains a fitness of hs.

In this report we used the *infinite site models*: we assumed that every mutation occurs in a different part of the chromosome, and that a new mutation cannot overwrite an old one. We also assumed that there is **no recombination** between the two chromosomes that are passed onto the child.

Once again, how this model changes for a diploid population is quite unpredictable. This is mainly because when a individual reproduces, they pass on only one of their chromosome to the next generation. This chromosome is chosen randomly, with no regard to its contribution to fitness whatsoever. This way, the chromosome of a superfit invidual with good mutations has no guarantee to be passed onto the next generation, even if the other chromosome does not contribute to the fitness at all. This is absolutely not the case for haploid populations, where an individual's fitness is entirely based on its only chromosome, which will be passed onto the next generation for sure if this individual reproduces and if its offspring survives the selection process.

1.8 Building confidence intervals with the standard error of the mean

Our analysis consists in making a lot of simulations in order to compute averages with as many values as possible. This approach requires us to inquire how precise our averages are.

To answer this question, we used a statistical approach with the estimators $(\langle T_p \rangle_1, \ldots, \langle T_p \rangle_m)$, with $\langle T_p \rangle_l$ describing the average T_p computed with the trees of the l-th simulation and m the total number of simulations. As our statistic is the sample mean, we used the *standard error of the mean* (SEM) of the estimator $\langle T_p \rangle_l$, which is the standard deviation of its sampling distribution and is generated by repeated samplings from the same population and recording the sample mean per sample. This forms a distribution whose variance is given by :

$$\text{SEM} := \frac{\langle \langle T_p \rangle_l \rangle}{n_{\text{sims}}}, \tag{20}$$

where $\langle \langle T_p \rangle_l \rangle$ is the average of $(\langle T_p \rangle_1, \ldots, \langle T_p \rangle_m)$ and n_{sims} is the total number of distinct simulations.

The SEM is the estimated standard deviation of the sample mean in the process by which it was generated, i.e. the estimated standard deviation of the sampling distribution of the sample statistic. It estimates how far the sample mean is likely to be from the population mean. Its main purpose here is to create confidence intervals, which can be done with *Chebyshev's inequality.*

Take X a random variable with finite variance σ_X^2. Then for any $r > 0$, Chebyshev's inequality [Fel+71] states that :

$$\mathbb{P}\big(|X - \mathbb{E}[X]| < r\sigma_X\big) \geqslant 1 - \frac{1}{r^2}. \tag{21}$$

If we take $\overline{X_n} := \frac{1}{n}\sum_{i=1}^{n} X_i$ the empirical mean of the i.i.d unimodal random variables $(X_1, \ldots, X_n)$, then we have $\mathbb{E}[\overline{X_n}] = \mathbb{E}[X]$ and $\sigma_{\overline{X}_n} = \sigma_{X_1}/\sqrt{n}$. (21) then yields :

$$\mathbb{P}\left(\left|\overline{X_n} - \mathbb{E}[X_1]\right| < r\frac{\sigma_{X_1}}{\sqrt{n}}\right) \geqslant 1 - \frac{1}{r^2}. \tag{22}$$

By reformulating (22) to form a confidence interval, we get :

$$\mathbb{P}\left(\mathbb{E}[X_1] \in \left[\overline{X_n} - r \cdot \text{SEM},\ \overline{X_n} + r \cdot \text{SEM}\right]\right) \geqslant 1 - \frac{1}{r^2}. \tag{23}$$

With $r = 2\sqrt{5} \approx 4.742$, this interval corresponds to a confidence of 95%. We used this method in the following sections to create a confidence interval around our empirical means.

2 Implementation and results

The previous section introduced discrete and coalescent models, as well as the differences between haploids and diploids and our means of characterizing coalescent models. Now that all of this points are behind us, this section will focus on the implementation and convergence results of the discrete models.

2.1 A quick introduction to SLiM

All of the discrete models introduced above were implemented on Messer Lab's evolutionary simulation framework SLiM 3 [HM19], which allows the simulation of complex populations. It is built on the scripting language Eidos [Hal16] and has built-in classes which implement entities such as mutations, individuals and chromosomes. SLiM is also endowed with a C++ core which implement standard model behaviors, and the graphical modeling environment SLiMgui. The following paragraphs are focused on the small fraction of tools provided by SLiM that we used during this internship.

SLiM generations SLiM works with discrete generations in which each individual can reproduce and die. To understand this section well, it is important to note that each generation is essentially a sequence of callbacks that are called in a precise order. In this report we used *non-WF models*, i.e. models which do not follows the rules of the diploid Wright-Fisher model. Therefore in our simulations, the callbacks followed the order given by Algorithm 1 at each generation.

Algorithm 1 One SLiM generation

```
Require: Population $\{i_1, \dots, i_N\}$, Number of children per individual $k$
  first()
  for $i \in \{i_1, \dots, i_N\}$ do
    reproduction(i)                                   ▷ Explained below
  end for
  early()
  for $i \in \{i_1, \dots, i_{(k+1)N}\}$ do
    survival(i)                                       ▷ Explained below
  end for
  late()
```

The `reproduction` and `survival` callbacks are looped over each alive individual and are essentially the reproduction and selection steps of the algorithms. The purpose of the `first`, `early` and `late` callbacks is to allow the user to create and modify variables or to export tree files at precise steps of each generation.

Sexual and asexual reproduction SLiM was first created with diploid populations in mind, so it is well-suited for the purposes of this internship. It also provides the possibility of using sexual reproduction, which we used to be more realistic with actual animal populations. Furthermore, it also supports haploid populations and haploid (clonal) reproduction.

The main appeal of SLiM's non-WF models however is the `reproduction()` callback, which iterates over every alive individuals to specify if they can reproduce, their number of children and properties thereof such as fitness. This callback has been used in all diploid models in a similar way to Algorithm 2. For haploid models, we just cloned each individual k times instead of making all females chose k mates.

In Brunet models, especially in the exponential model, a lot of children are created at each generation. It is important to note that they too are subject to the `survival()` callback.

Algorithm 2 Sexual reproduction process

```
Require: Adult individual $i$, Number $k$ of children per female, Adult male population $M$
  if $i$ is female then                                   ▷ Iterate on females only.
    for $l \in [\![1, k]\!]$ do
      Take $i_l \in M$ randomly.                          ▷ No link with the male's fitness.
      Create $j_l$ the child of $i$ and $i_l$.
      State properties of $j_l$ such as its fitness.
    end for
  end if
```

Fitness-based survival and selection In SLiM, each individual i is associated with a fitness value $x_i \in [0, 1]$, which dictates how likely it is to die at each generation. The main way to change the fitness of a lineage is with mutations, which we will see later.

The basic functioning of SLiM is that at each generation, during the `survival(i)` callback, SLiM draws an random variable $X_i \sim \mathcal{U}nif([0, 1])$, making i survive if and only if $X_i < x_i$ [HM16, p.171]. The user can then change this decision by writing code in the `survival()` callback, which will decide the fate of i. This individual dies if `False` is returned, lives if `True` is returned, and if `NULL` is returned, its fate stays unchanged.

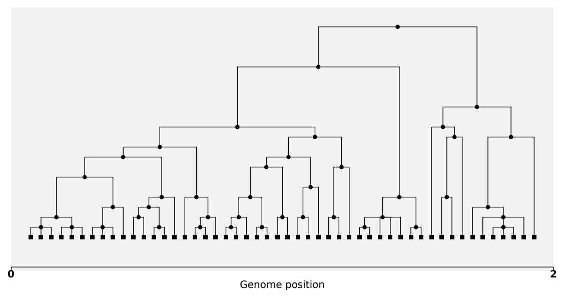
In Model A and in the Exponential model, only the N fittest children survive the selection process, which makes the survival callback similar to Algorithm 3. Note that the ordered list F of the fitness of all the children can be computed in the `early()` callback, after the reproduction step.

Algorithm 3 Survival callback in Brunet's models

```
Require: Adult individual $i$, Target size of population $N$, list $F$ of the fitness of all the children
  (sorted in descending order).
  if $i$ is of age 0 then
    if $x_i \geqslant F[N]$ then        ▷ Must be inclusive to not kill everyone in the first generation
      return True
    else
      return False
    end if
  else
    return False                                          ▷ Only the children survive.
  end if
```

Note that in practice, fitness is quite hard to handle in SLiM for models not based on mutations. We instead associated each individual with a custom floating-point tag with `individual.tagF`.

Mutation rates and effects In SLiM, mutations are initialized at the beginning of a simulations, and then randomly applied to every new nucleotide with a chosen rate. Several parameters are chosen during their initialization, like the dominance coefficient h and their effect on their carriers's fitness, which can be chosen to be fixed or randomly drawn according to a given probability distribution.



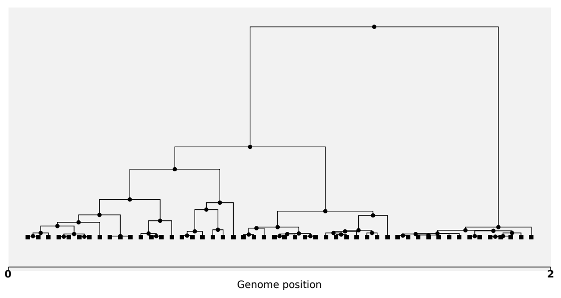


Figure 5: Example of trees obtained by sampling 50 leaves from the tree of a simulation of the Exponential model, haploid (left) and diploid (right) with $N = 100$. Both trees show multiple mergers, although the haploid one has more.

SLiM uses the infinite site model, which means that no matter where they appear on the genome, mutations cannot overwrite each other.

Trees recording and analysis in Python One other major advantage of SLiM is that it can save the simulation history into a `.trees` file, storing a tree where each merger describes the time of the last common ancestor of certain chromosomes [Hal+19] (see Figure 5).

The `.trees` files thus obtained can then be analyzed in Python with the `tskit` module. It is important to note that the `.trees` files are not actually storing a single tree, but instead a `TreeSequence` object containing a single tree. The command `TreeSequence.first()` can be used in order to get the tree from the `TreeSequence` object.

In `tskit` trees, the nodes are labeled with a specific integer, from the down top. For diploids, the labels of the *leaves* (the nodes at the bottom of the tree, at the other end from the *root*) are always between 0 and $2N - 1$. To sample the tree, `TreeSequence.simplify()` must be called with specific leaf labels.

As stated before, we are only interested in the values taken by $T_2(i_1, i_2)$ for $i_1 \neq i_2$, as $\langle T_p \rangle$ can be obtained for any $p \geqslant 3$ via (11). Therefore we only extract the values of T_2, storing it in a dictionary in which the key of $T_2(i_1, i_2)$ is the tuple $(\hat{i}_1, \hat{i}_2)$, where $\hat{i}$ is the label of the individual i in the sampled tree. This dictionary is then saved for later computations and plots in a `.json` file, which is a common file type for storing dictionaries and can easily be written or read in Python.

The extraction of T_2 from a tree file is summed up in Algorithm 4. Extra steps are needed for haploids because SLiM saves the history of haploid populations in diploid trees. This results in a tree where half of the leaves are not connected to any other node, and need to be removed in order to not raise errors in further computations.

2.2 Our simulations workflow

This section describes in detail how we conducted our simulations and extracted data from genealogical trees to generate the graphs presented in the following section.

For each model, we typically simulated populations with sizes ranging from 10^2 to 10^4 individuals. In each simulation, we extracted several genealogical trees, sampled from successive generations.

Sampling multiple trees from the same simulation is a practical way of obtaining a large number of data points, as coalescence needs to be simulated only once. However, this approach introduces

Algorithm 4 Extration of T_2 from a tree with Python

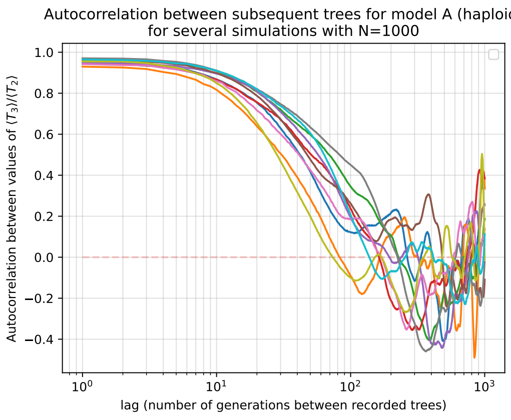
```
Require: Tree file, Number of samples per tree $s$, Sample size $n_s$.
  ts ← TreeSequence object from tree file.                              ▷ Use tskit.load().
  if Model is haploid then
      tree ← ts.first().
      r ← root of tree.                                                 ▷ Use max(tree.roots).
      ts ← Sampled ts with only the leaves that descend from this root. ▷ Use ts.simplify().
  end if
  Generate a sequence $(\hat{u}_l)_{1 \leqslant l \leqslant s \times n_s}$ of $s \times n_s$ distinct random integers of $[\![0, N-1]\!]$.
  Separate $(\hat{u}_l)_l$ in $s$ sequences $((v_l)^1_{1 \leqslant l \leqslant n_s}, \dots, (v_l)^s_{1 \leqslant l \leqslant n_s})$ of $n_s$ integers.
  for each sequence $(\hat{v}_l)^m_{1 \leqslant l \leqslant n_s}$ do
      ts2 ← Sampled ts with only leaves in $(v_l)^m_l$.
      Initialize T2 to the empty dictionary.
      for $(\hat{i}, \hat{j}) \in \{(\hat{i}, \hat{j}) \in [\![0, n_s - 1]\!]^2 \mid \hat{i} < \hat{j}\}$ do
          ts3 ← sampled ts2 with only leaves $i$ and $j$.
          T2($i, j$) ← coalescence time for ts3.   ▷ Use ts3.time(ts3.mrca(*ts3.samples())).
      end for
      Store T2 in a .json file.
  end for
```

a potential issue: trees sampled from successive generations may be statistically dependent. Fortunately, as illustrated in Figure 6, this correlation fades away if we space the recording of trees by a few hundred generations instead of doing it at every generation. In this report we lacked the time to redo all the simulations when we realized this, so we instead averaged over a lot of generations as Figure 7 showed that this should alleviate the dependency. We do not expect however that any dependency will have a huge effect on the means, as the values are repeated proportionally to how often they appear in the distribution.

In this report, every model is initialized with N individuals of the same fitness which can be haploid, male or female. It is impossible for us to simulate the ancestral history of these N initial individuals, so we need to wait for the chromosomal tree to coalesce before extracting it (i.e. we need to wait for a generation where every alive chromosome share a common ancestor among the starting ones). The only way to know if the tree has coalesced is to use `sim.treeSeqCoalesced()`, which does not work for haploid populations because of the way their chromosomes are stored. We thus proceeded in two ways for our simulations:

- For haploid models, we sampled the trees from generation $10N$ to generation $10N + 2000$. If the tree has not coalesced, an error is raided during its analysis in Python, forcing us to start over the simulation. Waiting for $10N$ generations ensure a high probability that the ancestry tree has coalesced.
- For diploid models, we waited for the ancestry tree to coalesce, and then we sampled the trees from the following 2000 generations.

It is important to note that our workflow may introduce a bias for haploid populations, because we do not consider simulations where coalescence is slow enough to not happen before generation $10N$. Indeed we used to proceed for diploids the same way we do for haploids now, which resulted in a large bias towards the Kingman coalescent. Fortunately we can use [Bru+06b] to check if our results are truthful for haploid simulations.



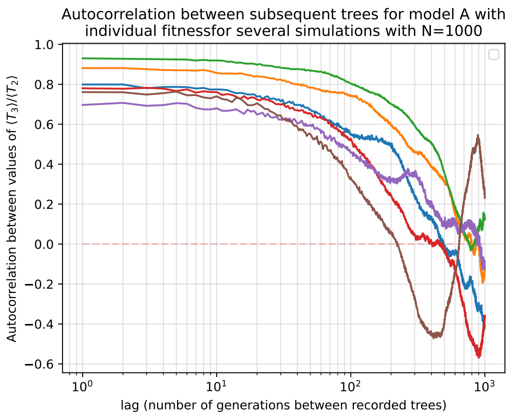
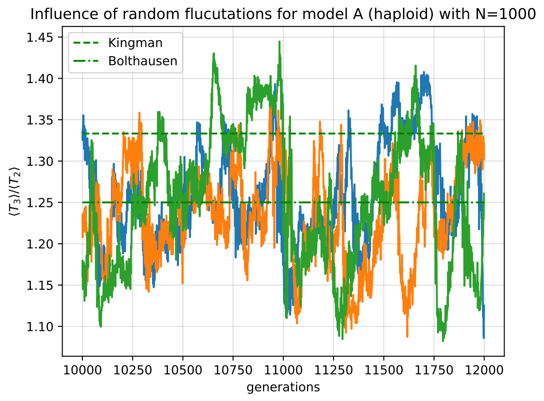


Figure 6: Autocorrelations between subsequent recorded trees for given number of generations between trees, for haploid model A (left) and diploid model A with individual-based fitness (right).



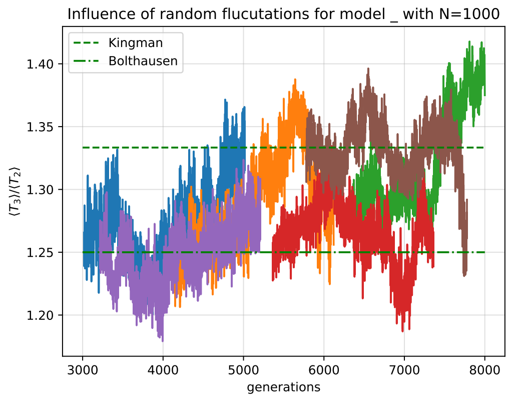


Figure 7: Graph of $\langle T_3\rangle/\langle T_2\rangle$ over several generations for independent simulations of Model A, haploid (left) and diploid with individual fitness (right) with $N = 1000$. Diploid models shows more dependency between generations, but sampling over a great number of generations reduce this effect.

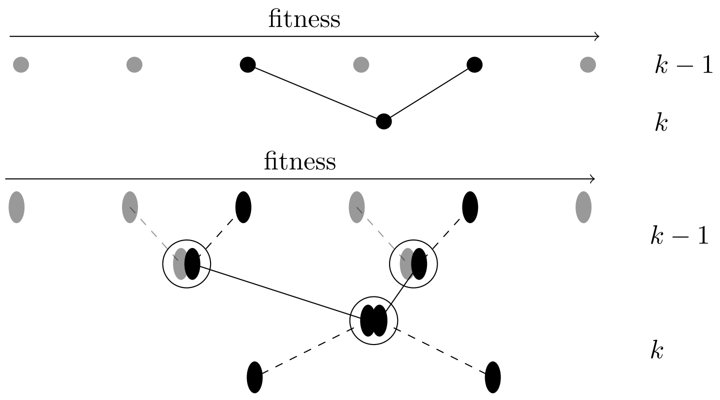


Figure 8: Example of reproduction process for Model A_1 (above) and Model A_2 (below). Both children have similar fitnesses, but in Model A_2 the fitnesses of each chromosome are not diluted for the next generation.

To characterize each model, we used the average coalescence times of the sampled trees, with equations (15) and (19) as benchmarks. Since the convergence of $\langle T_p \rangle / \langle T_2 \rangle$ to its theoretical value is slower for larger p, we primarily focused on ratios $\langle T_3 \rangle / \langle T_2 \rangle$ and $\langle T_4 \rangle / \langle T_2 \rangle$.

To compute these ratios, we extracted 10 small subtrees of 10 individuals from each tree. We computed the values of T_p for all p-tuples of each subtree with $p \in \{2, 3\}$, and averaged the results.

2.3 Brunet's model A

Implementation Model A is the most straightforward, essentially well-implemented with Algorithm 2 and Algorithm 3. The real question however consists in the way we make it diploid. We implemented this change in two different ways, labeling the underlying models A_1 and A_2 (see Figure 8).

1. *Model A_1* : individual-based fitness. This is the most straightforward approach. The fitness of a child is simply the average fitness of its parents, with a random noise $\varepsilon_{i,j}$ added. The genome of a child is assumed to be an even mix of the genomes of its parents, with no dominance effects.

2. *Model A_2* : chromosome-based fitness. This is a more realistic approach in which the fitness of an individual is the sum of its chromosomes's fitnesses, and the fitness of a chromosome is inherited from its parent chromosome in a haploid way, with a random noise $\varepsilon_{i,j}$ added. The genome of a child is still assumed to be an even mix of the genome of its parents, but this time we take into account that only half of both genomes is inherited. Once again, this reproduction model assumes that there is **no recombination** between the chromosomes passed onto the child. It can be seen as a simpler implementation of the Mutations model.

Model A_2 is actually quite hard to implement in SLiM, because the `.tagF` tag is only available for `individual` objects. We bypassed this restriction by using the spatial properties `individual.x` and `individual.y` instead, dictating which one should be inherited by taking Bernoulli random variables (see Algorithm 5).

In [Bru+06b], Brunet et al. took $k = 2$ children per individual in the reproduction process, and $\rho(\varepsilon) = \mathcal{U}nif([0, 1])$ in (7) for the fitness of the offspring. We took the same parameters for both models for reproduction purposes.

Algorithm 5 Reproduction callback in Model A_2

```
Require: Adult individual $i$, Adult male population $M$
  if $i$ is female then
    for $j \in [\![1, k]\!]$ do
      Take $i_j \in M$ randomly.
      if $B_1 = 1$, with $B_1 \sim \mathcal{B}er(\frac{1}{2})$ then
        $H_F \leftarrow$ first chromosome of $i$
        $x \leftarrow x(i)$
      else
        $H_F \leftarrow$ second chromosome of $i$
        $x \leftarrow y(i)$
      end if
      if $B_2 = 1$, with $B_2 \sim \mathcal{B}er(\frac{1}{2})$ then
        $H_M \leftarrow$ first chromosome of $i_l$
        $y \leftarrow x(i_l)$
      else
        $H_M \leftarrow$ second chromosome of $i_l$
        $y \leftarrow y(i_l)$
      end if
      Create $j_l$ the child of $i$ and $i_l$.          ▷ Use p1.addRecombinant().
      Set $j_l$'s chromosomes to $H_F$ and $H_M$.
      $x(j_l) \leftarrow x + \varepsilon_1$, with $\varepsilon_1 \sim \rho(\varepsilon)$
      $y(j_l) \leftarrow y + \varepsilon_2$, with $\varepsilon_2 \sim \rho(\varepsilon)$
    end for
  end if
```

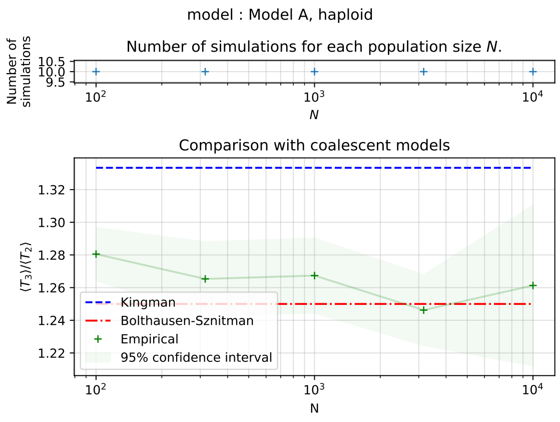
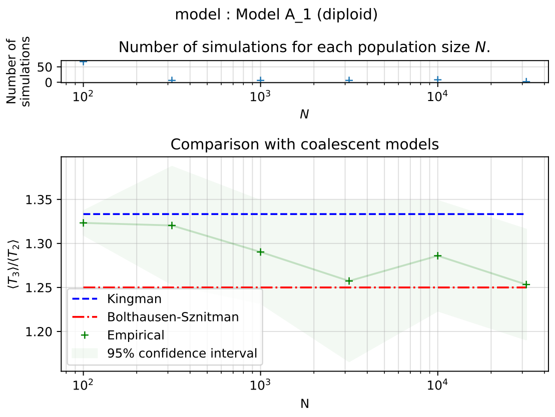


Figure 9: Simulation results for Model A (haploid).



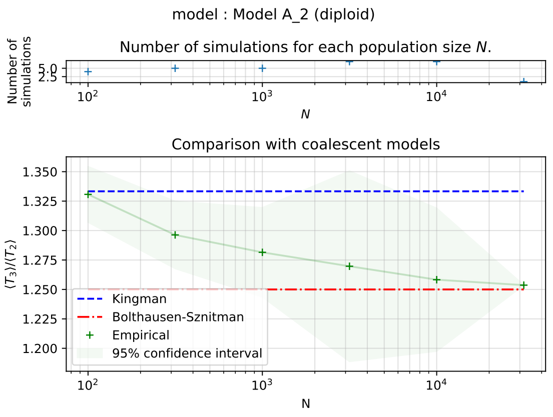


Figure 10: Simulation results for Models A_1 (left) and A_2 (right), both diploid.

Results We began by testing the haploid model to be sure that we could replicate the results of [Bru+06b] and that our implementation was not faulty. As it can be seen of Figure 9, the Bolthausen-Sznitman coalescent approximates the genealogy in this model very well.

The next step is to simulate the diploid extensions of this model : Model A_1 and Model A_2. Figure 10 shows that these models still converge towards the Bolthausen-Sznitman coalescent, but with a slower speed.

This result hints that diploidy slows down the effect of selection, especially for small populations. There are quick explanations for this phenomenon for both models A_1 and A_2.

1. For models with individual fitness, an individual with low fitness can reproduce with a superfit one and produce a viable child. This has two consequences : an individual with low fitness has a higher chance to reproduce in diploid models than for haploid ones, and a superfit individual can produce children which are not superfit. The relevance of fitness over tens or hundreds of generations is reduced because it is diluted when two individuals reproduce together.

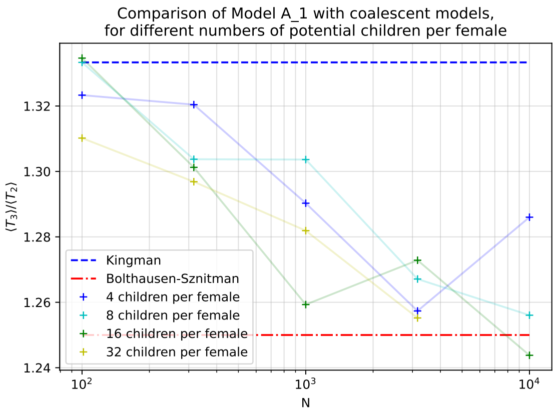


Figure 11: Simulation results for Model A_1, with varying number k of children per female.

2. For models with chromosome-based fitness, the same phenomenon occurs, but its consequences are amplified by the way fitness is tracked. During reproduction, the choice of the chromosome which is passed onto the child is random and irrespective of its fitness. Therefore a superfit chromosome which contributes greatly to the individual's fitness is not guaranteed to be passed onto the next generation.

The bottom line is that in both cases, diploidy makes it harder for superfit individuals to birth a superfit lineage. The addition of sexuality probably makes it even harder, as two superfit individuals of the same sex cannot reproduce together. These effects are however less visible for large populations, where superfit lineages seem to be more likely to appear.

One could ponder if raising the number of potential children per female could make selection more visible, as giving individuals more opportunities to reproduce would raise the probability of superfits to reproduce together, henceforth making superfit lineages more abiding. Figure 11 hints that this could be true.

The exponential model also tackle this question, with a number of potential children which can be raised as far as infinity.

2.4 Brunet's exponential model

Implementation For this model we didn't try to implement chromosome-based fitness as it would be redundant with model A_2. The main challenge however was to build a Poisson point process according to a certain distribution. Fortunately, it can be done by creating an uniform Poisson point process on $[0, 1]$ and by applying the inverse of the cumulative distribution function of the distribution to it.

Because the lowest values of the Poisson point process will yield children that will die immediately, it is in practice useless to consider them. Therefore we capped the number of children per female to N_0, which is a parameter that can be changed. Those steps are summed up in Algorithm 6.

In our simulations, we took $N_0 = 50$ to make sure that superfit individuals are well-represented in the reproduction process. This creates approximately $25N$ children at each generation, so approximately $24N$ of them will die during the `survival()` callback.

Algorithm 6 Reproduction callback in the exponential model

```
Require: Individual $i$, Maximal number of potential children per female $N_0$, male population $M$
  if $i$ is female then
    for $l \in [\![1, N_0]\!]$ do
      Evaluate $U_l \sim \mathcal{U}nif([0,1])$
      $\varepsilon_l \leftarrow \log(1 - U_l) - \log(N_0)$        ▷ Inverse of cumulative distribution function
      Take $i_l \in M$ randomly.
      Create $j_l$ the child of $i$ and $i_l$.
      $x_{j_l} \leftarrow \frac{x_i + x_{i_l}}{2} + \varepsilon_l$
    end for
  end if
```

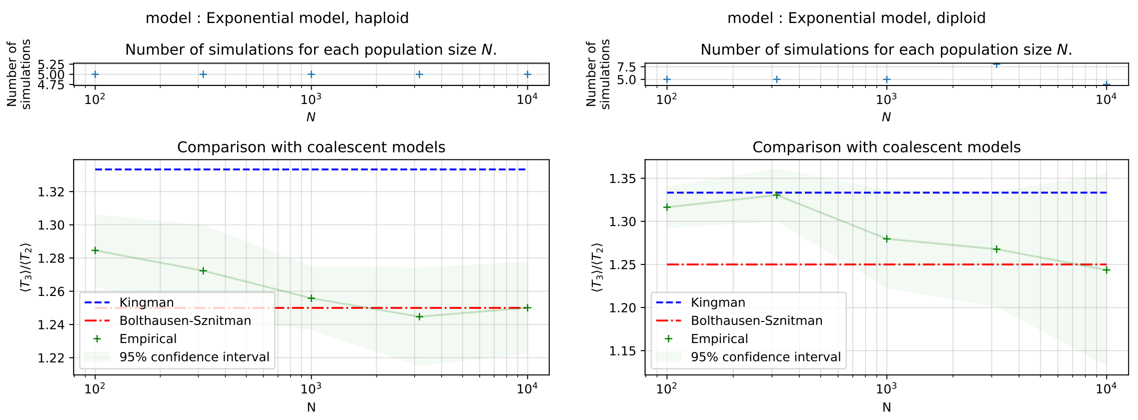


Figure 12: Simulation results for the Exponential model, haploid (left) and diploid (right).

Results Figure 12 shows that the haploid exponential model converges very quickly towards the Bolthausen-Sznitman coalescent, as showed in [Bru+06b]. The SEM is smaller than for Model A, which indicates less discrepancy.

The diploid model still seems to converge towards the Bolthausen-Sznitman coalescent, but to a much slower pace, similar to Models A_1 and A_2. This hints that diploidy still reduces the effects of selection on the population, even when the number of potential children per female is larger.

2.5 Mutations model

Implementation The Mutations model lets SLiM uses its full potential by letting it handle fitness via mutations on the genetic code of individuals. These mutations are implemented by the `initializeMutationType()` function, which takes several parameters such as its dominance coefficient or the way it changes the fitness of the individual who carry mutations.

For this model, the selection process still follows Algorithm 3, but is based on the actual fitness of individuals instead of on the `.tagF` tag. The fitness of an individual can be obtained by summing all values of `m.selectionCoeff` over each mutation `m` of each chromosome.

Several mutation types can be used in a single simulation, but for our project we chose to use only one. We chose to give it a dominance coefficient of $h = 0.5$, which means that the mutations's

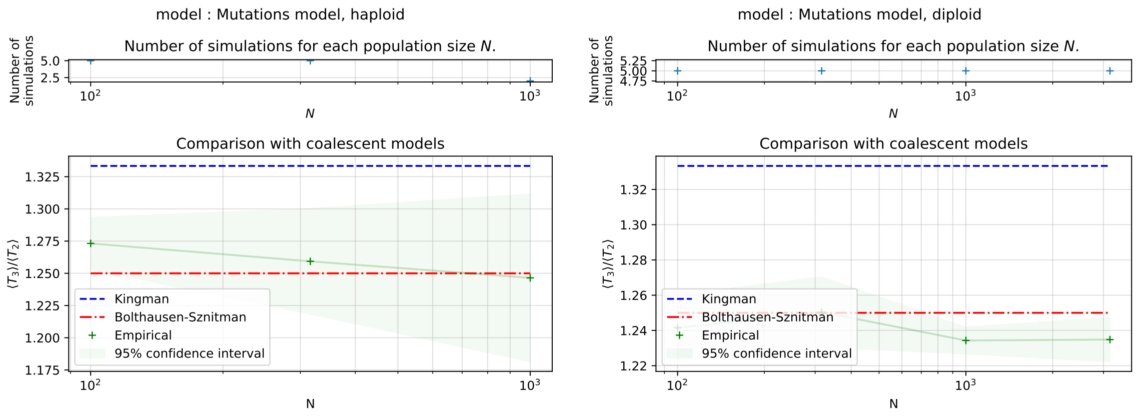


Figure 13: Simulation results for the Mutations model, haploid (left) and diploid (right).

effect on fitness are doubled when they are present on both chromosomes.

The mutations's effect on fitness can be chosen to be fixed, or drawn randomly with respect to a given probability distribution. For our simulations we chose to take a Gaussian distribution of mean 0 and of standard deviation 0.5, which means that, contrary to previous models, mutations can be deleterious.

Finally, in order to mimic Brunet models, we chose to make it so that approximately one mutations occurs every time a new chromosome is formed. We took a mutation rate of 10^{-4} for chromosomes which contain 10^4 nucleotides (remember that because of the infinite site model, mutations cannot overwrite each other).

Results Figure 13 shows the results of the Mutation model, in both the haploid and the diploid case. Because of the very high mutation rate, the simulations took a lot more time than other models, which restricted our study. However, both models seem to be better approximated by the Bolthausen-Sznitman coalescent than by the Kingman coalescent, with $\langle T_3\rangle/\langle T_2\rangle$ decreasing at a tremendous speed below the theoretical value of the Bolthausen-Sznitman coalescent. This lowness can be explained by a huge number of multiple mergers, where a chromosome has more that two child chromosomes in the next generation (see Figure 5 for examples of multiple mergers).

This time, the diploid model is the one which seems to converge the fastest. This can be explained by the fact that twice as many mutations occur for diploid individuals than for haploids.

However, our mutations model fails to capture the effects of the dominance coefficient h on selection. Another approach will be used in the following subsection to test its influence.

2.6 Influence of the model on selection

The previous results showed that the effects of selection seem to be reduced for diploids, especially in small populations. This part focuses on small tweaks in the models and their effect on the populations.

Beneficial mutations are dominant Our first idea was that the convergence towards the Bolthausen-Sznitman coalescent is slower for diploid populations because the dilution of fitness

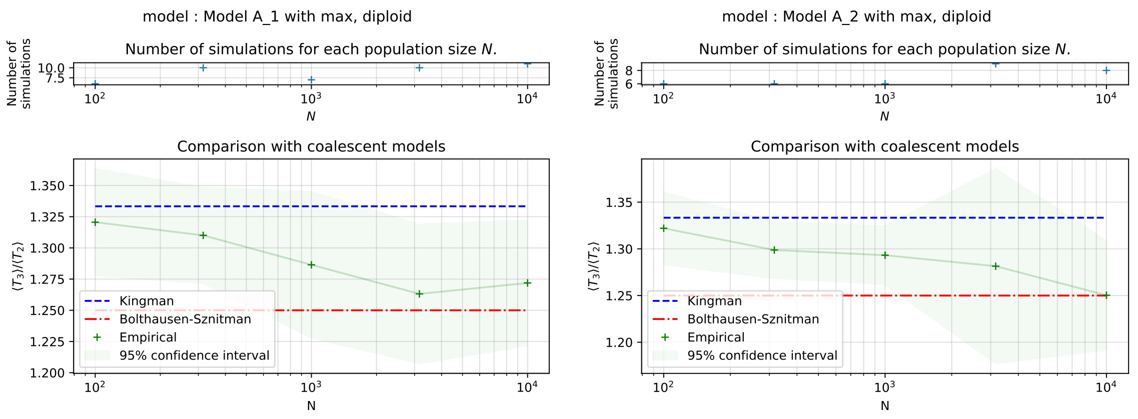


Figure 14: Simulation results for Models A_1 and A_2, with the averages in the reproduction step replaced with a maximum.

makes it harder for superfit lineages to appear. To remedy this dilution, we tried to change the averages in Algorithms 2 and 5 to a maximum, making a child's fitness the maximal fitness of its two parents in Model A_1 and the maximal fitness of its chromosomes in Model A_2.

Because mutations are implied to never be deleterious, this change can be viewed as making them dominant. In Model A_1, this change is equivalent to making the children inherit the mutations of their fittest parent, and in Model A_2, only the mutations of the fittest chromosome are expressed.

Figure 14 show that this change did not actually alter the results. Indeed, we actually made the fitness of the less fit parent or chromosome irrelevant during reproduction, thus made it easier for unfit individuals to reproduce, which made fitness less important for reproduction.

This effect is amplified when taking the maximum instead of the average in the Exponential model (Algorithm 6). Figure 15 shows an even slower convergence, which can be explained by the fact that unfit individuals have more opportunities to reproduce in the exponential model.

Beneficial mutations are recessive Another insight was that diploidy reduces the effect of selection because the dilution of fitness during reproduction makes the appearance of superfit lineages less likely. This dilution mainly appears because unfit individuals can still birth a viable offspring, as long as they mate with a superfit one. By replacing the average in Algorithm 2 and Algorithm 5 by a minimum, we make sure that both parents must be superfit in order to birth a superfit child.

Similarly to the maximum case, taking the minimum fitness can be viewed as making mutations recessive: both parents or chromosomes need to have a mutation in order for it to be passed onto the child.

As it can be seen on Figure 16, taking the min do not increase significantly the effect of selection on the population. This can be explained by the fact that making mutations recessive makes it harder for superfit individuals to birth superfit children, as the fitness of the fittest parent is irrelevant during reproduction.

In the Exponential model however, individuals have more opportunities to reproduce, and therefore superfit children are more likely to be born. As showed in Figure 15, taking a minimum instead of an average in Algorithm 6 makes the model converge faster towards the Bolthausen-Sznitman coalescent.

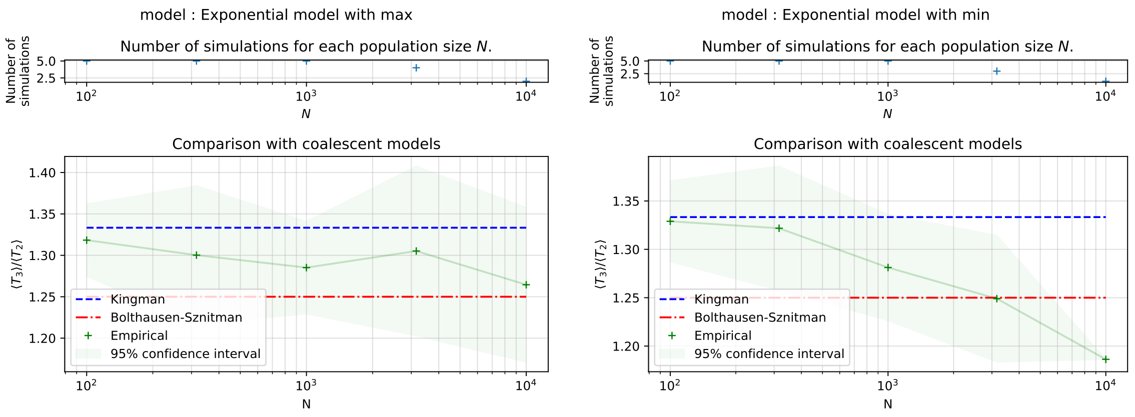


Figure 15: Simulation results for the diploid Exponential model, with the average in the reproduction step replaced with a maximum (left) or a minimum (right).

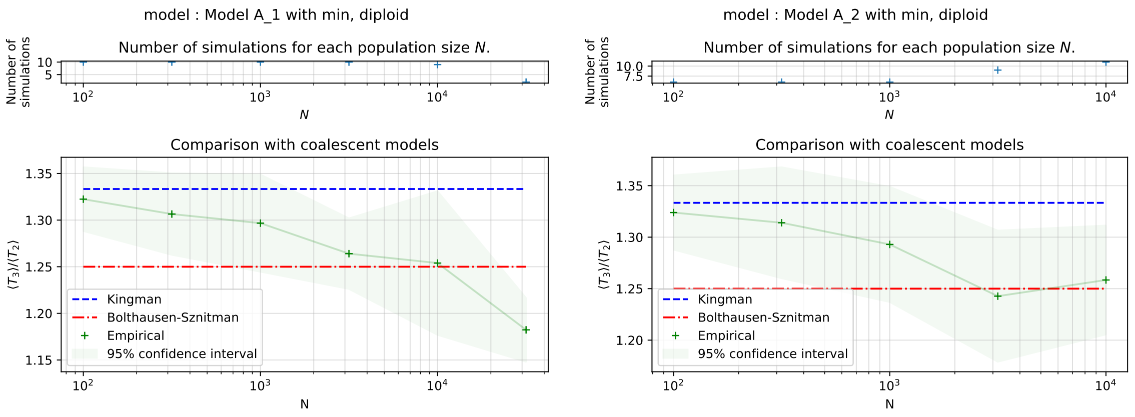


Figure 16: Simulation results for Models A_1 and A_2, with the averages in the reproduction step replaced with a minimum.

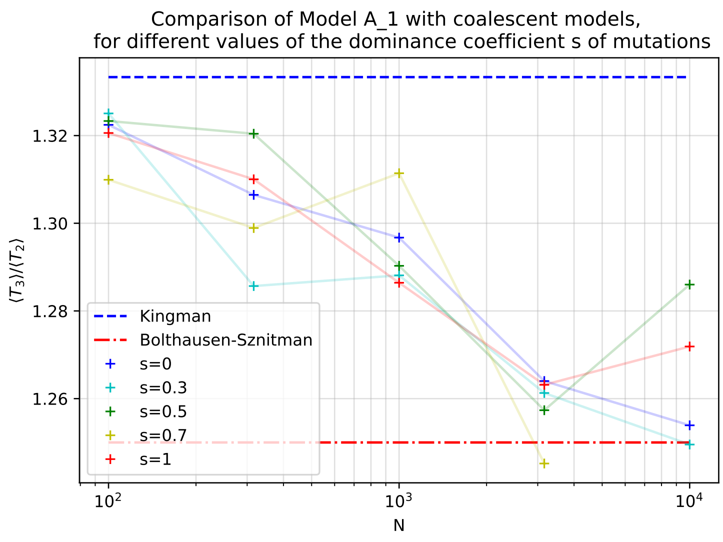


Figure 17: Effect of the dominance coefficient on Model A_1.

Instead of making the beneficial mutations fully dominant or recessive, a dominance coefficient $h \in [0,1]$ can be used in order to specify how much the fitness of the fittest parent is taken into account during reproduction, similarly to how it operates in the mutations model. If j is the child of i_1 and i_2, then its fitness can be given by :

$$x_j = h \max(x_{i_1}, x_{i_2}) + (1-h)\min(x_{i_1}, x_{i_2}). \tag{24}$$

This way, taking $h = 1$ is the same as making the positive mutations dominant, $h = 0$ corresponds to making them recessive and taking $h = \frac{1}{2}$ is the same as making the child's fitness the average of its parents', which is what we did in the beginning.

Figure 17 sums up the effect of the dominance coefficient. It seems to confirm that a lower value of h indeed makes the effects of selection stronger, though a higher number of simulations would be necessary once again in order to draw significant conclusions.

Hermaphroditic population Both previous attempts at making the effects of selection stronger highlighted that selection is the strongest when superfit lineage are the most likely to appear and thrive.

One important difference between our implementations of haploid and diploid models that we overlooked this far was the addition of sexuality: in diploid models, only a male and a female can reproduce together. Two individuals of the same sex cannot reproduce together, even if both of them are superfit. This means that the probability that two superfit individuals reproduce together is reduced by half for diploids.

We chose to add sexuality to our models because we had animal and plant populations in mind, and most of them reproduce sexually. However, some animals like worms or snails are *hermaphroditic*: they can produce viable gametes of both sexes and can reproduce with every other member of their species. Note that these animals can also self-fertilize, which will not be taken into account in our models.

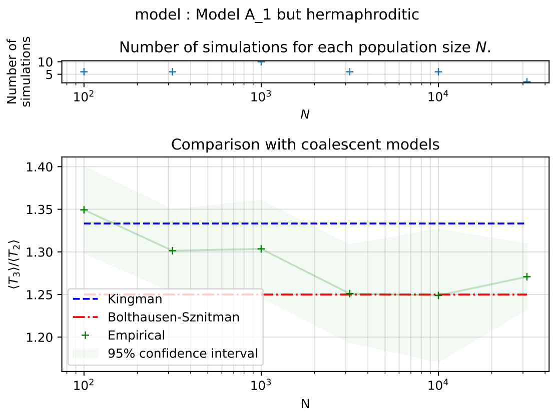


Figure 18: Simulation results for the hermaphroditic versions of Model A_1.

We then implemented hermaphroditic populations with Models A_1 to see if it visibly made the effects of selection stronger. Figure 18 shows that shows that selections indeed has more effect when the population is hermaphroditic.

Only the fittest can reproduce Our last attempt at making selection stronger was to make females chose males based on their fitness. We implemented this choice by changing Algorithm 2, randomly taking the male mate among the 10 fittest ones instead of taking him randomly among the whole adult male population.

Note that this model is not made to be realistic. It can depict a farmer artificially choosing its best animals or plants for the next generation at best, but does not reflect any natural population.

Figure 19 shows that this model is better approximated by the Kingman coalescent than by the Bolthausen-Sznitman coalescent. It seems that artificially increasing selection to this degree is counterproductive and actually drastically reduces the effects of selection. This is still the case if the population is hermaphroditic, even though hermaphroditism showed to increase the effects of selection in the previous results.

3 Conclusion

Scientific conclusion In this internship, we designed, implemented and simulated diploid extensions of population genetics models studied in [Bru+06b] in order to check if the convergence results stay true for diploid populations. We also tampered with the models to check how some modifications can change the effects of selection.

Most of the studied models showed a convergence towards the Bolthausen-Sznitman coalescent, at a speed summed up in Figure 20. Overall, it seems that diploid models converge slower than haploid ones, and raising the number of potential children, making beneficial mutations recessive or making individuals hermaphroditic can make this convergence faster. Other changes such as making the mutations dominant actually seems to make the convergence slower instead.

However, this project needed several improvements in the simulation in order to be more computationally efficient and to reduce bias. In the end we only have a small number of simulations to

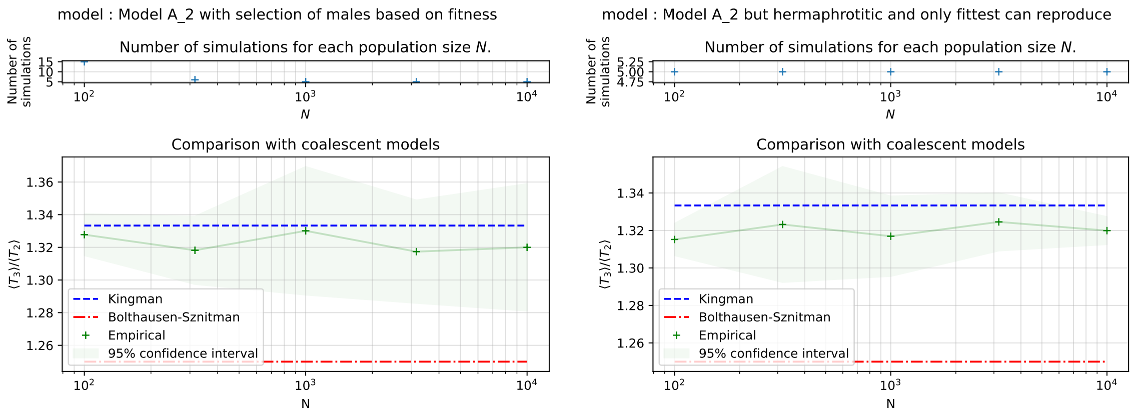


Figure 19: Simulation results for model A_2, but only the 10 fittest males can reproduce (left). On the right, the population is hermaphroditic, and only the 10 fittest individuals can reproduce.

draw results from, with uncomfortably large confidence intervals. More time to run simulations and more computation power would be necessary in order to make 20 show definitive results.

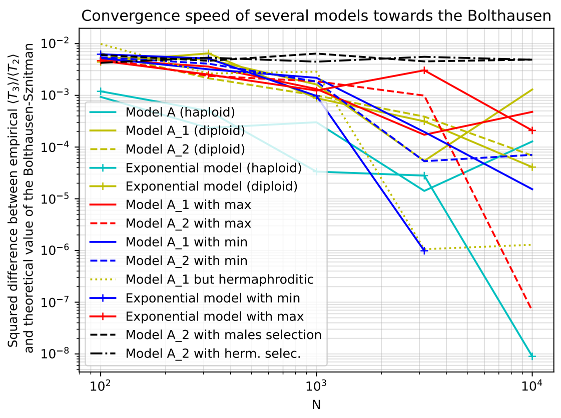


Figure 20: Empirical convergence speed of multiple models towards the Bolthausen-Sznitman coalescent.

The models can still be tweaked to simulate animal populations more realistically. So far recombination or selfing are not taken into account, and SLiM offers a lot of ways of perfecting our models to make them closer to a given existing population.

Our tree analysis can also be improved. The rescaled average coalescence times $\langle T_p\rangle/\langle T_2\rangle$ offer a practical way of comparing trees with theoretical coalescent models, but they do not provide information about the underlying causes of these results. Inquiring about the number of multiple mergers or keeping track of the fitness to detect superfit lineages for example would make the dynamics of the models much more apparent.

Personal conclusion I think I had the best possible conditions for this internship. It really was a enriching experience to use mathematics in order to solve problems from another field. I have learnt a lot, especially in statistics, bioinformatics and in the interface of both.

The way I was integrated to the lab in Leicester was really nice. I loved to be invited to the meetings, as I learned a lot on how research is performed in population genetics or in general in a PhD. I also presented the results of my internship online during a meeting with another lab, and it was really nice to be integrated among PhD students, even if my results were not as impressive as theirs.

Finally, I could visit Leicester, which is a nice city to spend weekends in. I am glad that I could also visit London with my girlfriend or with my parents.

4 Code availability

The code we used for our simulation is available at this GitHub repository. The main file is `simulate3.py`, which performs the SLiM simulations, create the tree files and extract their data into `.json` files. The file `plots3.py` can be used to make plots similar to the ones presented in this report with the data from your simulations. The data from the simulations performed during this internship is not included in the repository because of its heavy weight.

To use `simulate3.py`, three folder paths will need to be precised in the `main` function.

- `slim_path` is the folder where the SLiM files will be stored.
- `cwd` is the folder where the tree files are stored before being analyzed. This folder is separated from the rest in case an error arises during the analysis.
- `T2_dest` is the folder where the `.json` files will be stored after the trees are analyzed. The `.json` files are sorted by model and by population size.

For some models, some arguments can be changed before starting the simulations. They can be used to change the dominance coefficient s, the number k of children per female (for sexual diploids) or per individual, and the gap in generations between the recording of two trees. As shown in Figure 6, this parameter should be at least a few hundreds, even if it was only 1 during this internship.

See the README file of the repository for more information.

References

[FIS22] RA FISHER. "On the dominance ratio. Proc. Roy. Soc. Edinb. 42: 321-341." In: (1922).

[Wri31] Sewall Wright. "Evolution in Mendelian populations". In: *Genetics* 16.2 (1931), p. 97.

[Fel+71] William Feller et al. *An introduction to probability theory and its applications.* Vol. 963. Wiley New York, 1971.

[Gri80] Robert C Griffiths. "Lines of descent in the diffusion approximation of neutral Wright-Fisher models". In: *Theoretical population biology* 17.1 (1980), pp. 37–50.

[Kin82a] John FC Kingman. "On the genealogy of large populations". In: *Journal of applied probability* 19.A (1982), pp. 27–43.

[Tav84] Simon Tavaré. "Line-of-descent and genealogical processes, and their applications in population genetics models". In: *Theoretical population biology* 26.2 (1984), pp. 119–164.

[Che+95] Yu-Sheng Chen et al. "Analysis of mtDNA variation in African populations reveals the most ancient of all human continent-specific haplogroups". In: *American journal of human genetics* 57.1 (1995), p. 133.

[DT95] Peter Donnelly and Simon Tavare. "Coalescents and genealogical structure under neutrality". In: *Annual review of genetics* 29.1 (1995), pp. 401–421.

[Ham95] Michael F Hammer. "A recent common ancestry for human Y chromosomes". In: *Nature* 378.6555 (1995), pp. 376–378.

[JT95] Mark A Jobling and Chris Tyler-Smith. "Fathers and sons: the Y chromosome and human evolution". In: *Trends in Genetics* 11.11 (1995), pp. 449–456.

[WSG95] L Simon Whitfield, John E Sulston, and Peter N Goodfellow. "Sequence variation of the human Y chromosome". In: *Nature* 378.6555 (1995), pp. 379–380.

[TLK96] Lev S Tsimring, Herbert Levine, and David A Kessler. "RNA virus evolution via a fitness-space model". In: *Physical review letters* 76.23 (1996), p. 4440.

[Wil96] Christopher Wills. "Topiary pruning and weighting reinforce an African origin for the human mitochondrial DNA tree". In: *Evolution* 50.3 (1996), pp. 977–989.

[Kes+97] David A Kessler et al. "Evolution on a smooth landscape". In: *Journal of statistical physics* 87 (1997), pp. 519–544.

[Tav+97] Simon Tavaré et al. "Inferring coalescence times from DNA sequence data". In: *Genetics* 145.2 (1997), pp. 505–518.

[BS98] Erwin Bolthausen and A-S Sznitman. "On Ruelle's probability cascades and an abstract cavity method". In: *Communications in mathematical physics* 197 (1998), pp. 247–276.

[Pit99] Jim Pitman. "Coalescents with multiple collisions". In: *Annals of Probability* (1999), pp. 1870–1902.

[MS01] Martin Möhle and Serik Sagitov. "A classification of coalescent processes for haploid exchangeable population models". In: *Annals of Probability* (2001), pp. 1547–1562.

[Sny03] Robin E Snyder. "How demographic stochasticity can slow biological invasions". In: *Ecology* 84.5 (2003), pp. 1333–1339.

[KT04] Morten Kloster and Chao Tang. "Simulation and analysis of in vitro DNA evolution". In: *Physical review letters* 92.3 (2004), p. 038101.

[DS05] Rick Durrett and Jason Schweinsberg. "A coalescent model for the effect of advantageous mutations on the genealogy of a population". In: *Stochastic processes and their applications* 115.10 (2005), pp. 1628–1657.

[Klo05] Morten Kloster. "Analysis of evolution through competitive selection". In: *Physical review letters* 95.16 (2005), p. 168701.

[Bru+06a] E Brunet et al. "Phenomenological theory giving the full statistics of the position of fluctuating pulled fronts". In: *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics* 73.5 (2006), p. 056126.

[Bru+06b] Éric Brunet et al. "Noisy traveling waves: effect of selection on genealogies". In: *Europhysics Letters* 76.1 (2006), p. 1.

[BBC08] David J Balding, Martin Bishop, and Chris Cannings. *Handbook of statistical genetics.* John Wiley & Sons, 2008.

[FM09] Fabian Freund and Martin Möhle. “On the time back to the most recent common ancestor and the external branch length of the Bolthausen-Sznitman coalescent”. In: *Markov Process. Related Fields* 15.3 (2009), pp. 387–416.

[BBS14] Veronika Boskova, Sebastian Bonhoeffer, and Tanja Stadler. “Inference of epidemiological dynamics based on simulated phylogenies using birth-death and coalescent models”. In: *PLoS computational biology* 10.11 (2014), e1003913.

[SG14] Mark S Springer and John Gatesy. “Land plant origins and coalescence confusion”. In: *Trends Plant Sci* 19.5 (2014), pp. 267–269.

[TL14] Aurelien Tellier and Christophe Lemaire. “Coalescence 2.0: a multiple branching of recent theoretical developments and their applications”. In: *Molecular ecology* 23.11 (2014), pp. 2637–2652.

[Hal16] B.C. Haller. “Eidos: A Simple Scripting Language.” In: (2016). DOI: `http://benhaller.com/slim/Eidos_Manual.pdf`.

[HM16] B.C. Haller and P.W. Messer. “SLiM: An Evolutionary Simulation Framework.” In: (2016). DOI: `http://benhaller.com/slim/SLiM_Manual.pdf`.

[SS16] Patrick Smadbeck and Michael PH Stumpf. “Coalescent models for developmental biology and the spatio-temporal dynamics of growing tissues”. In: *Journal of The Royal Society Interface* 13.117 (2016), p. 20160112.

[Hal+19] B.C. Haller et al. “Tree-sequence recording in SLiM opens new horizons for forward-time simulation of whole genomes”. In: *T. Molecular Ecology* 19.2 (2019), pp. 552–566. DOI: `https://doi.org/10.1111/1755-0998.1296`.

[HM19] Benjamin C Haller and Philipp W Messer. “SLiM 3: forward genetic simulations beyond the Wright–Fisher model”. In: *Molecular biology and evolution* 36.3 (2019), pp. 632–637.