Conditioned sampling of Birth-Death reconstructed trees

Lab Rotation supervised by Dr. Tim Vaughan

Jasmine Gamblin

September 16th - October 25th 2019

1 Motivation

1.1 MCMC for phylogenetic inference

Phylogenetics is the study of the evolutionary history and relationships between individuals or group of organisms. Phylogenetic inference is the fact to infer these relationships from observable traits like molecular or morphological data. In order to do this kind of analysis, a popular solution is to use Markov Chains Monte Carlo (MCMC) based on phylogenetic likelihood to sample over the tree space [1]. MCMC is a method allowing to sample from a target probability distribution for which direct sampling is difficult. It proceeds by creating a Markov chain which equilibrium distribution is the target one.
This is what the software BEAST2 performs, allowing to infer phylogenies and parameters for several evolutionary models [2]. When implementing a new model for this software or for phylogenetic inference in general, we may want to compare the tree distribution obtained by MCMC with the one we should obtain in theory.

The distribution we want to obtain is $P(\tau|\theta, s)$ where τ is a tree, θ represents the parameters of the model, and s contains the sampling times of the samples for which we want to infer evolutionary parameters. Then it would be useful to have a method allowing to sample directly from this distribution, in order to check the correctness of the MCMC one. Although it is easy to sample from $P(\tau|\theta)$, we also want to condition on the sampling times. For some models like the coalescent it is simple to sample trees conditioned on sampling times, because coalescent trees are reconstructed backward in time, but for other models like birth-death processes it requires more thought. This the class of models that we will study here.

1.2 Birth-Death process

Birth-death processes are continuous time Markov chains used to model the evolution of the population size. In the following we focus on a constant-rate processes. We consider a population which size is represented by the random variable N. Each individual of the population can give birth to an other individual with rate λ, die with rate μ or be sampled with rate ψ. We will consider two kinds of birth-death processes, as represented Figure 1:

- in the first model, the sampling is without removal: N is not modified by a sampling event
- in the second model, we introduce a parameter r which is the removal probability upon sampling: for any sampling event, there is a probability r for the individual to be removed, and thus for N to be decreased by one. This can correspond to a situation where individuals are infected people: when they go to hospital and have their virus sequenced, there is some probability that they get cured or quarantined and therefore be removed from the infected population

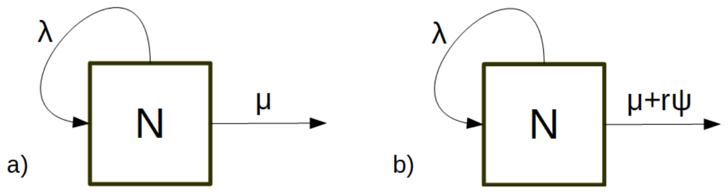


Figure 1: a) constant-rate birth-death process b) constant-rate birth-death process with removal upon sampling

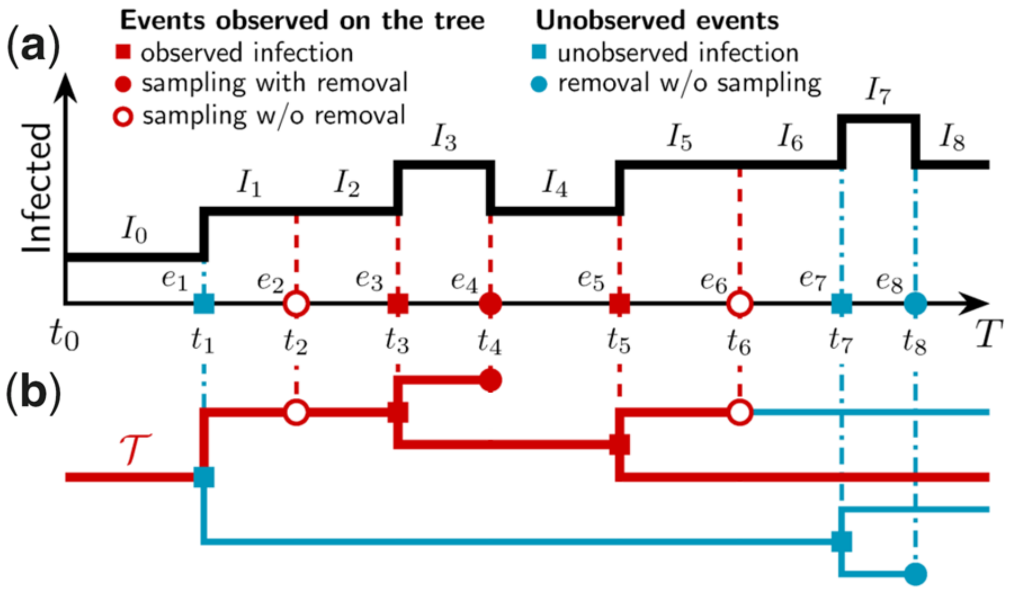

(a)
Events observed on the tree
observed infection
sampling with removal
sampling w/o removal
Unobserved events
unobserved infection
removal w/o sampling
Infected
I_0 I_1 I_2 I_3 I_4 I_5 I_6 I_7 I_8
e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 T
(b)
$\mathcal{T}$

Figure 2: Number of lineages through time plot (a) and corresponding tree (b), in the case of an infectious disease study (adapted from [3])

To simulate a tree under these models, we proceed forward in time. Beginning with a single individual at time 0, we can simulate the number of lineages through time (LTT) until a chosen time t_{max}. Then we can reconstruct a tree from the sampled individuals, this time proceeding backward in time. However, this alone does not allow to sample trees conditioned on the number of samples and their sample times. The mapping between the LTT plot and the reconstructed tree is represented Figure 2.

2 Theoretical background: sampling and weighting

In this part we will show which methods allow us to sample from the wanted distribution, and what are the theoretical justification for these methods. As we will see the method differs slightly whether $r = 0$ or not.

2.1 Sampling without removal

We wish to sample from the distribution $P(\tau|\lambda, \mu, \psi, s)$, which we can rewrite in the following way:

$$
\begin{aligned}
P(\tau|\lambda, \mu, \psi, s) &= \frac{P(\tau, s|\lambda, \mu, \psi, T)}{P(s|\lambda, \mu, \psi, T)} \\
&= \frac{\sum_\eta P(\tau, s|\eta, \psi)P(\eta|\lambda, \mu, T)}{\sum_\eta P(s|\eta, \psi)P(\eta|\lambda, \mu, T)} \\
&= \frac{\sum_\eta P(\tau|s, \eta)P(s|\eta, \psi)P(\eta|\lambda, \mu, T)}{\sum_\eta P(s|\eta, \psi)P(\eta|\lambda, \mu, T)}
\end{aligned}
$$

Here T is the stopping time of the birth-death process, that can be chosen as the last sampling time $s_{|s|}$, and η is a trajectory, that is a sequence of times and N values defining an LTT plot. We can further write:

$$
P(\tau|\lambda, \mu, \psi, s) = \frac{\sum_\eta \sum_{\tau'} \delta_{\tau,\tau'} P(\tau'|\eta, s)P(\eta|\lambda, \mu, T)P(s|\eta, \psi)}{\sum_\eta P(s|\eta, \psi)P(\eta|\lambda, \mu, T)}
$$

If we jointly sample trajectories $\eta^{(d)} \sim P(\eta|\lambda, \mu, T)$ and trees $\tau^{(d)} \sim P(\tau|\eta^{(d)}, s)$, we can apply the law of large numbers to this expression to obtain:

$$
\begin{aligned}
P(\tau|\lambda, \mu, \psi, s) &= \frac{\lim_{n\to\infty} \frac{1}{n} \sum_{d=1}^{n} \delta_{\tau,\tau^{(d)}} P(s|\eta^{(d)}, \psi)}{\lim_{n\to\infty} \frac{1}{n} \sum_{d=1}^{n} P(s|\eta^{(d)}, \psi)} \\
&= \lim_{n\to\infty} \frac{1}{n} \sum_{d=1}^{n} \delta_{\tau,\tau^{(d)}} w^{(d)}
\end{aligned}
$$

with

$$
w^{(d)} = \frac{P(s|\eta^{(d)}, \psi)}{\lim_{n\to\infty} \frac{1}{n} \sum_{d'=1}^{n} P(s|\eta^{(d')}, \psi)}
$$

This means that we can approximate this tree distribution by sampling trajectories $\eta^{(d)}$ from $P(\eta|\lambda, \mu, T)$, which we can easily do using Gillespie algorithm [4], then reconstruct trees $\tau^{(d)}$ from them and weight them with $w^{(d)}$.

Tree reconstruction In order to reconstruct a tree $\tau^{(d)}$ from a trajectory $\eta^{(d)}$, we begin with one lineage at the time of the last sample. Then we go backward in time: at time t, we denote by l_t the number of lineages in the tree, and by N_t the number of individuals in the population as defined by the trajectory.

- If there is a sampling event at time t, we place this sample on an existing lineage with probability $\frac{l_t}{N_t}$, and create a new lineage with probability $\frac{N_t - l_t}{N_t}$.
- If there is a birth event at time t, we merge two lineages of the tree with probability $\frac{\binom{l_t}{2}}{\binom{n_t}{2}}$.

We continue until $t = 0$ and we have only one lineage left.

2.2 Sampling with removal

Now we consider that an individual can be removed from the population upon sampling, with probability $r > 0$. It means that the sampling process has now an influence over the trajectory η. We introduce a binary vector y such that $y_i = 1$ if sample i has been removed and $y_i = 0$ if not. We can write:

$$
P(\tau|\lambda, \mu, \psi, s) = \frac{\sum_\eta \sum_y P(\tau|\eta, y, s)P(\eta, y, s|\lambda, \mu, \psi, r, T)}{\sum_\eta \sum_y P(\eta, y, s|\lambda, \mu, \psi, r, T)}
$$

But here we cannot factorise by the probability of a trajectory η. Instead, we will use $P(\eta_i|\eta_{i-1}, y_{i-1}, \lambda, \mu)$ the probability distribution governing the portion i of a trajectory η, that is the portion between sample times s_{i-1} and s_i. Then we can factorise the following way:

$$P(\eta, y, s|\lambda, \mu, \psi, r, T) = \left[\prod_{i=1}^{|s|} P(\eta_i|\eta_{i-1}, y_{i-1}, \lambda, \mu) P(s_i|\eta_i, \psi) P(y_i|r)\right]$$

where $P(s_i|\eta_i, \psi)$ is the probability of there being no sampling in the time interval (s_{i-1}, s_i) followed by a sample at time s_i, and $P(y_i|r)$ is the probability for the sample type y_i. To sample trees from this distribution, we can iteratively draw sample types $y_i^{(d)} \sim P(y_i|r)$ and partial trajectories $\eta_i^{(d)} \sim P(\eta_i|\eta_{i-1}, y_{i-1}, \lambda, \mu)$, then draw $\tau^{(d)} \sim P(\tau|\eta^{(d)}, y^{(d)}, s)$ and weight the triple $(\tau^{(d)}, \eta^{(d)}, y^{(d)})$ according to the weight $w^{(d)}$ having the following expression:

$$w^{(d)} = \frac{\prod_{i=1}^{|s|} P(s_i^{(d)}|\eta_i^{(d)}, \psi)}{\lim_{n\to\infty} \sum_{d'=1}^{n} \prod_{i=1}^{|s|} P(s_i^{(d')}|\eta_i^{(d')}, \psi)}$$

Tree reconstruction The tree reconstruction is very similar to the previous case, except that if at time t there is a sampling event with removal we create a new lineage in the tree with probability 1.

3 Implementation

3.1 Code structure

In order to implement the method described above, I developed an R package called bdTreeSim (available on my GitHub [5]) with the following functions and classes.

3.1.1 Functions

trajectory(lambda, mu, r, sampling_times) draw a vector $y \sim P(y|r)$ and generate partial trajectories $\eta_i \sim P(\eta_i|\eta_{i-1}, y_{i-1}, \lambda, \mu)$ using Gillespie algorithm. Return the resulting complete trajectory η as well as the vector y.

weight(trajectory, sampling_times, psi) compute the log of the unnormalised weight $\log(\prod_{i=1}^{|s|} P(s_i^{(d)}|\eta_i^{(d)}, \psi))$ for a trajectory $\eta^{(d)}$.

tree_from_trajectory(trajectory, sampling_times) reconstruct a tree $\tau^{(d)}$ from a trajectory $\eta^{(d)}$.

ess_value(weights, is.log) compute the estimated number of independent samples, called ESS or estimated sample size, based on the Rényi entropy. **is.log** indicates if the **weights** are in their log form or not, and the function returns $\frac{1}{\sum_{d=1}^{n}(w^{(d)})^2}$.

sample_trees(e, n1, n2, sampling_times, lambda, mu, psi, r) sample $n2$ trees from $P(\tau|\lambda, \mu, \psi, r, s)$. The function can proceeds in two different ways: if **n1** is specified it simulates $n1$ trajectories with the function **trajectory**, whereas if **e** is specified it sample trajectories until the ESS value exceeds **e**. Then in the two cases we subsample $n2$ trajectories from them using the weights w, and reconstruct $n2$ trees from these using **tree_from_trajectory**.

height_distribution(trees, t_max) return the heights of **trees**, ie. the time elapsed between the most recent common ancestor to all samples and the last sample.

newick(trees, file) create a text file containing the trees of the list **trees** in Newick format, to export or visualise them with a tool like IcyTree [6].

3.1.2 Classes

trajectory class An object of class **trajectory** is a list with 3 fields:

- a field **t** which contains the times of birth, death and removal events in increasing order $t_1, ..., t_k$, with $t_1 = 0$ and $t_k = s_{|s|}$ (except for zero-weight trajectories that can have $t_k < s_{|s|}$)
- a field **N** containing the values for the population size $N_1, ..., N_k$, with N_i being the value of N in the time interval $[t_i, t_{i+1})$, and N_k the value at t_k
- a field **removed** containing the sampling times for removed samples

tree class An object of class **tree** has a nested structure. It represents a node from a tree by a list with 2 fields:

- a field **branch_length** containing the length of the branch incident to the node
- a field **subtrees** containing a list of 0, 1 or 2 **tree** objects which are the children of the node

A binary tree is represented by the **tree** object corresponding to its root.

3.2 Results

Height distribution Although we cannot really represent a tree distribution, we can instead compute and compare some of their characteristics. Here we used the tree height, that is the time elapsed between the most recent common ancestor to all samples and the last sample. Thus we compared the tree height distribution for the MCMC method implemented in the BEAST2 package SA, and for the direct sampling method implemented in the R package bdTreeSim.
We tested several parameters values and sample times, and simulated an increasing number of trajectories. Figures 3 and 4 were created using the following parameter values: $\lambda = 1.9$, $\mu = 1$, $\psi = 0.3$, $r = 0.5$, $s = (1, 2)$, $\mathbf{n2} = 10^5$. The length of the Markov chain for the BEAST simulation was 10^8 and every 10^3 value was stored, to match the number of trees **n2**.
Figure 3 shows the evolution of the KL divergence between the two distributions as the number of simulated trajectories **n1** increases. We can see that it tends to zero and is of the order of 10^{-4} for $\mathbf{n1} = 10^6$.
Figure 4 shows the superimposed histograms of the two distribution (a) and the decile values (b) for $\mathbf{n1} = 10^6$. We can see that there is very little difference between the two. An other value that we compared was the proportion of trees having exactly a height of 1. With these parameters, we obtained a proportion of 0.1945 ± 0.0005 for BEAST and bdTreeSim.

Running time Here we tried to study the running time of the direct sampling method as we increase one parameter. For Figures 5 and 6 we used the same model parameters as before, that is $\lambda = 1.9$, $\mu = 1$, $\psi = 0.3$, $r = 0.5$ and $s = (1, 2)$.

- for Figure 5 we increased **n1** from 10^3 to 10^6 with **n2** fixed at 10^4. Logically, we can confirm that the running time is linear in **n1**.
- for Figure 6 we increased **n2** from 10^3 to 10^6 with **n1** fixed at 10^4. Again, we verify that the running time is linear in **n2**.
- for Figure 8 we used a different set of parameters ($\lambda = 2$, $\mu = 1$, $\psi \in \{0.3, 1.5\}$, $r = 0.2$) and increased the number of sampled individuals from 2 to around 80. Here we fixed $\mathbf{n2} = 1000$ and simulated enough trajectories to achieve an ESS value of 1000. The results are more difficult to interpret because for the same number of sampled individuals we can have sampling times which are more or less likely to be generated by a birth-death model with these parameters. Then for the more likely sampling times we will have to simulate less trajectories before obtaining an ESS value of 1000. In order to work

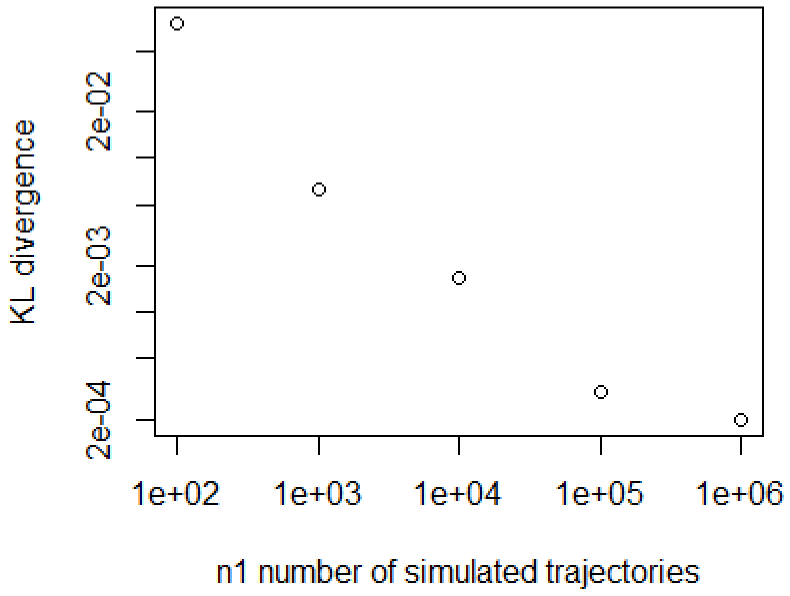
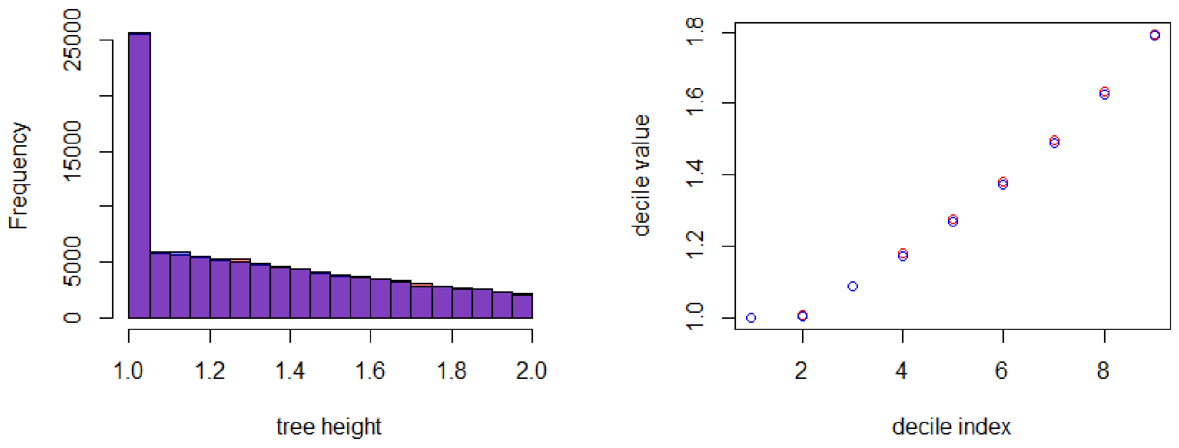


Figure 3: KL divergence between the BEAST2 and the bdTreeSim tree height distributions



(a) height distribution

(b) deciles of height distribution

Figure 4: comparison of the tree height distributions obtained by the package bdTreeSim (in red) and by BEAST (in blue) for the parameters $\lambda = 1.9$, $\mu = 1$, $\psi = 0.3$, $r = 0.5$, $s = (1, 2)$

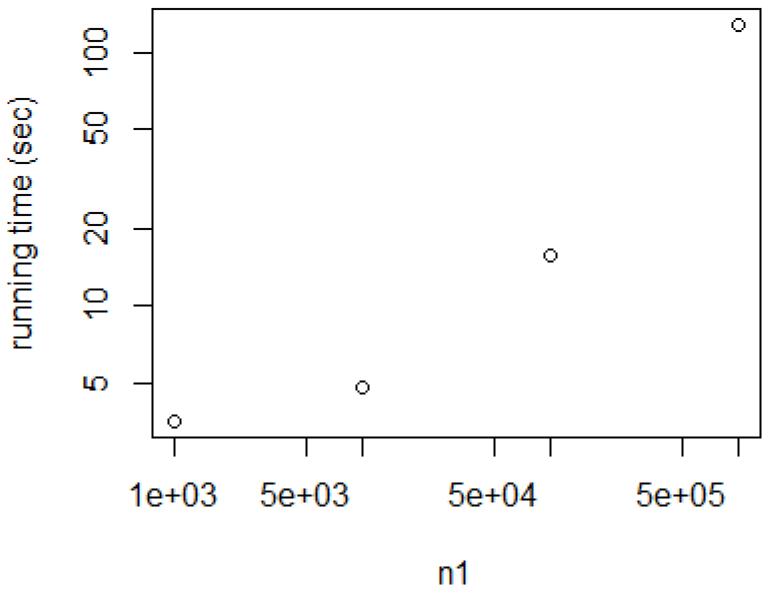


Figure 5: evolution of running time in seconds when **n1** increases

with sampling times that are not too unlikely, I generated them from a birth-death process (without removal) using Gillespie algorithm. In Figure 7, we can see that depending on the value of ψ we do not obtain the same distribution for the number of sampled individuals. Thus for $\psi = 0.3$ the running time increases faster when the number of sampled individuals increases because we rapidly attain numbers which are very unlikely. For $\psi = 1.5$ the running time increases more slowly but starts at a higher value because small numbers of samples are a bit less likely than for $\psi = 0.3$.

4 Discussion

We described here a direct sampling method to simulate birth-death trees conditioned on sampling times. We introduced the motivation behind this problem, then the theoretical considerations involving the law of large numbers which allow us to approximate this distribution by simulating trajectories under a birth-death model and weighting them. We described the implementation of this method in an R package called bdTreeSim, and presented the performance of this package in terms of correctness and running time.

It appears that this method give the expected results when we compare it to the tree distribution from BEAST, and is of course much more efficient than a method consisting in sampling trajectories following a given birth-death model and report only the ones corresponding to the wanted sampling times. Thus this kind of method seems to be promising and it would be useful to make it work for other variants of birth-death processes, like the multitype birth-death model or models with piece-wise constant parameters.

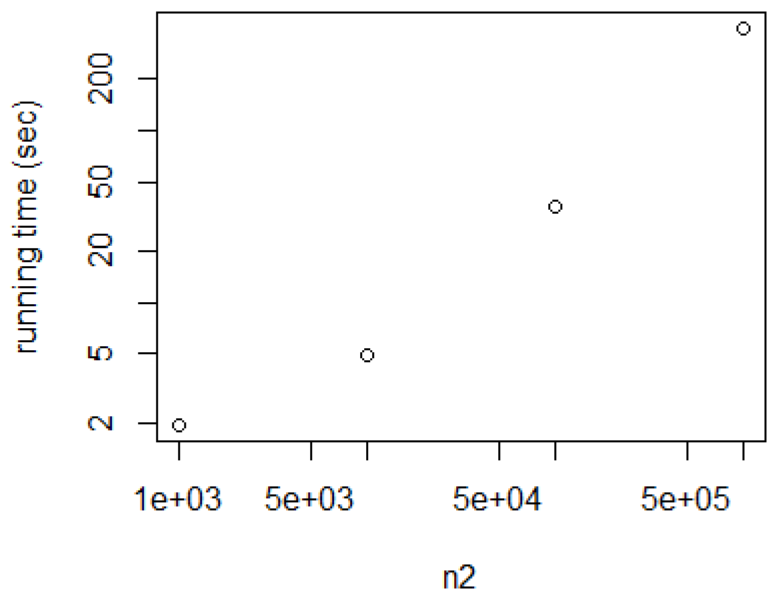
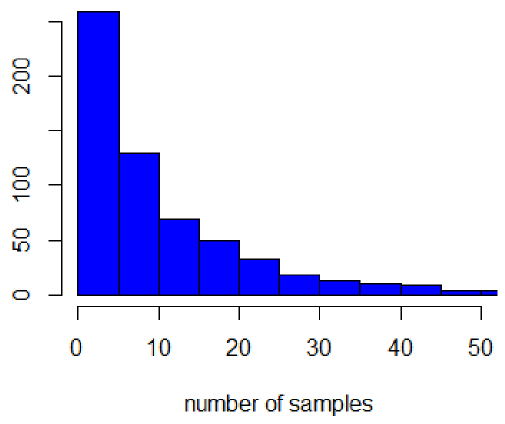
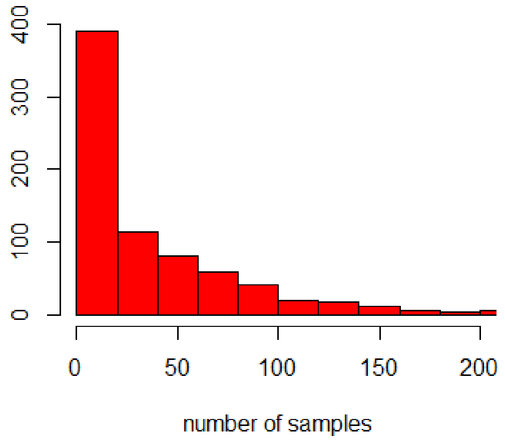


Figure 6: evolution of running time in seconds when **n2** increases

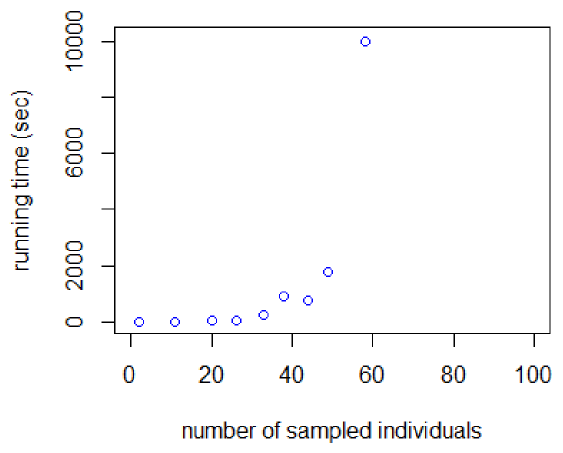


(a) $\psi = 0.3$

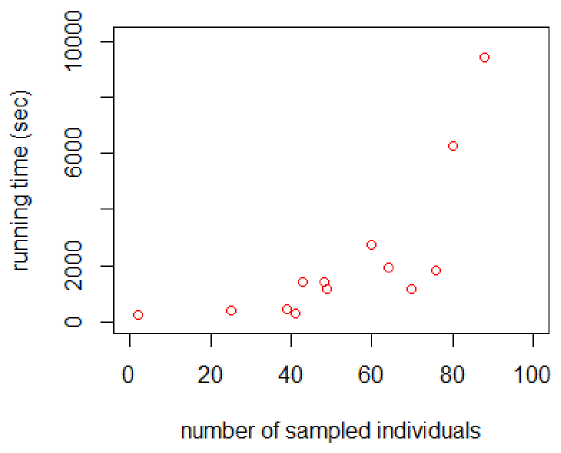


(b) $\psi = 1.5$

Figure 7: distribution of the number of sampling events for a birth-death model with parameters $\lambda = 2$, $\mu = 1$, $r = 0$, $T = 3$



(a) $\psi = 0.3$



(b) $\psi = 1.5$

Figure 8: running time for different parameters and number of samples

However the current implementation is not fast enough when using more than a hundred of sampled individuals. Several modifications could be made in order to improved the running time:

- do importance resampling during the simulation of a trajectory. For example we simulate several trajectories for the first portion (between $t = 0$ and $t = s_1$), weight them and subsample only one trajectory from them, then do the same for each portion. It should reduce the proportion of simulated trajectories that have a weight of zero because the population goes extinct before the last sampling event, thus we would have to simulate less trajectories to obtain a given ESS value
- use τ-leaping approximation [7] when simulating a trajectory. By updating less often the rates, it would allow to make Gillespie algorithm faster.

References

[1] AJ Drummond, GK Nicholls, AG Rodrigo, and W Solomon. Estimating mutation parameters, population history and genealogy simultaneously from temporally spaced sequence data. *Genetics*, 161(3):1307–1320, 2002.

[2] R. Bouckaert, T.G. Vaughan, J. Barido-Sottani, S. Duchêne, M. Fourment, and A. Gavryushkina et al. BEAST 2.5: An advanced software platform for bayesian evolutionary analysis. *PLoS computational biology*, 15(4), 2019.

[3] Timothy G Vaughan, Gabriel E Leventhal, David A Rasmussen, Alexei J Drummond, David Welch, and Tanja Stadler. Estimating Epidemic Incidence and Prevalence from Genomic Data. *Molecular Biology and Evolution*, 36(8):1804–1816, 2019.

[4] Daniel T Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22(4):403–434, 1976.

[5] Jasmine Gamblin, 2019. https://github.com/JasmineGamblin/bdTreeSim.

[6] Timothy G Vaughan. Icytree: Rapid browser-based visualization for phlogenetic trees and networks. *Bioinformatics*, 33(15):2392–2394, 2017.

[7] D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733, 2001.