Divide and Conquer Sequential Monte Carlo for Phylogenetics

by

Sean William Jewell

B.Sc., McMaster University, 2012

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
(Statistics)

The University of British Columbia
(Vancouver)

August 2015

Abstract

Recently reconstructing evolutionary histories has become a computational issue due to the increased availability of genetic sequencing data and relaxations of classical modelling assumptions. This thesis specializes a Divide & conquer sequential Monte Carlo (DCSMC) inference algorithm to phylogenetics to address these challenges. In phylogenetics, the tree structure used to represent evolutionary histories provides a model decomposition used for DCSMC. In particular, speciation events are used to recursively decompose the model into subproblems. Each subproblem is approximated by an independent population of weighted particles, which are merged and propagated to create an ancestral population. This approach provides the flexibility to relax classical assumptions on large trees by parallelizing these recursions.

Preface

This dissertation is solely authored, unpublished work by the author, Sean Jewell. Alexandre Bouchard-Côté and Sean Jewell designed research and experiments.

Table of Contents

List of Tables

List of Figures

List of Listings

Glossary

$\bar{\gamma}$	Unnormalized target posterior distribution
$\mathbf{r}$	Vector of compound Poisson process scalings
$\mathbf{z}$	Vector of compound Poisson process event locations
γ	Target posterior distribution
λ	Poisson process rate parameter
$\mathcal{C}$	Children
$\mathcal{L}$	Loss function
$\mathcal{P}$	Parents
$\mathcal{Y}$	Data at leaf nodes
b	Branch length function
Q	Rate matrix
$\mathbb{P}$	CTMC probability
π	Stationary distribution
Σ	Character set
τ	Tree topology
ξ	Number of compound Poisson process events

B	Ambient space
d	Height increments
E	Edge set
e	Collection of compound Poisson process events
h	Height function
J	Compound Poisson process
M	Number of taxa
m	Substitution rate
N	Number of particles
n	Number of nucleotides sequenced
P	Poisson process
q	Proposal distribution
r	Root
S	Stochastic process
T	Tree length
V	Node set
X	Leaf set $X \subset V$
CPP	Compound Poisson process
CTMC	Continuous time Markov chain
DCSMC	Divide and conquer sequential Monte Carlo
ESS	Effective sample size
IS	Importance sampling

MAP	Maximum a posteriori
MCMC	Markov chain Monte Carlo
ML	Maximum likelihood
PMCMC	Particle Markov chain Monte Carlo
PP	Poisson process
SIS	Sequential importance sampling
SMC	Sequential Monte Carlo

Acknowledgments

Going to the mountains is going home.
— John Muir

The last two years in Vancouver have been a wonderful experience. I am indebted to a number of people who have helped me grow professionally at UBC, and personally through many adventures in British Columbia.

Firstly, none of this thesis would have been possible without the excellent tutelage of Alexandre Bouchard-Côté. Alex introduced me to the beautiful field of probabilistic approaches to machine learning. Within this area, I had the distinct pleasure of exploring applications spanning problems in phylogenetics (through a collaborative research project with the BC Cancer Agency) to modelling urban spatial data. Alex's tireless dedication to curate this formative part of my research career cannot be undervalued.

I am financially indebted to both the BC Cancer Agency and the Natural Sciences and Engineering Research Council of Canada. Thank you for your support.

Within the Department of Statistics I have made many friends. Thank you to Andrés Sánchez Ordoñez for always organizing creative ways for us to explore Vancouver, and to Creagh Briercliffe for injecting his witty sense of humour into our office culture. A special thank you to Neil Spencer for the many lively and engaging office conversations that often resulted in laughter, new insights, or another side project.

Vancouver's close proximity to both the mountains and sea offered many opportunities to ski powder, hike, and kayak. I am especially grateful to

Greg Shield, Kym Berenschot, Sunny Sheshgiri, Dan Ashby, Hyun-Kyoung Koo, Ariel Bultz, and Patrick Prendergast for always spurring new adventures.

Thanks to Janine Roller and her family for welcoming me to their home for the holidays and for hosting me in Whistler after our skiing escapades. You helped make this coast feel like home.

Lastly, I am fortunate to have very supportive parents and little sister. The amount of patience and love they have given me through this meandering path is remarkable.

Sungmee, thank you for always believing in us.

Chapter 1

Introduction

The evolutionary history of species, cells, or tumours is often of interest to researchers trying to understand the complex relationships between extant species and their ancestors. Knowledge of these histories allows researchers to study, for example, ancestral state reconstruction, population histories, phylogeny and alignment, and species divergence times (see Chen et al. [5, Chapter 1] and the references therein).

Attempts to encapsulate the complicated dynamics of evolution have given rise to a host of different approaches from heuristics to model based methods [13]. The introduction of likelihood based approaches led to Bayesian modelling in phylogenetics [8, 22, 25, 27, 30, 34, 43]. Interestingly, in contrast to the usual connotations associated with Bayesian modelling, the early Bayesian phylogenetic models provided a more computationally efficient framework than standard Maximum likelihood (ML) bootstrapping methods [22]. Moreover, the computational efficiency of Bayesian modelling versus traditional ML methods allowed for more biologically reasonable evolutionary models to be studied [19].

Within Bayesian phylogenetics, the demand for efficient and accurate posterior inference has driven development of sophisticated Markov chain Monte Carlo (MCMC) methods [22, 25, 27, 30, 43], and more recently particle filter methods [4]. These methods approximate the posterior distribution via a collection of weighted samples. Such weighted particle populations can

be subsequently used to approximate quantities of biological interest (which are often represented as expectations with respect to the posterior distribution).

Although MCMC and particle filters have facilitated inference for increasingly realistic evolutionary models, the recent influx of genetic sequence data and the desire for (even) more biologically reasonable models has begun to demand more efficient methods. To address potentially inefficiencies in MCMC sampling for large scale data, Bouchard-Côté et al. [4] introduced a particle filter approach for phylogenetics by using techniques similar to the agglomerative clustering methods of Teh et al. [40].

Research in areas outside of efficient MCMC and particle filter methods have also been developed for the data surge. Åkerborg et al. [1] presented a fast dynamic programming approach that can be used to speed-up MCMC methods, and also in a hill-climbing maximum a posteriori (MAP) estimate for dating inference. A difference of this work is that it provides the MAP estimate and not an estimate of the full posterior distribution as in MCMC and particle filtering methods. Our goal is to utilize the the full power of Bayesian decision theory, which couples the posterior distribution and a loss function to yield an optimal estimator known as the Bayes estimator. Hence, we focus on MCMC and particle filtering.

The work developed herein continues the trend of particle filtering methods for Bayesian phylogenetics by specializing the computationally efficient and parallelizable DCSMC algorithm of Lindsten et al. [26] to phylogenetics. We anticipate that this framework will become desirable as evolutionary biologists sequence more species while simultaneously demanding more complex evolutionary models.

This thesis is organized as follows. Chapter 2 provides the necessary biological, phylogenetics, and sampling preliminary material for the rest of this thesis. Chapter 2 concludes with an introduction to particle filtering methods which serve as the motivation and building blocks for DCSMC. DCSMC and its specialization to phylogenetics are proposed in Chapter 3. Experimental results are presented in Chapter 4. Conclusions and future directions are listed in Chapter 5.

Chapter 2

Background

This chapter is divided into three different parts. The first section discusses the biological background necessary to describe phylogenetics and briefly mentions some common biological hypotheses. The second section describes Bayesian phylogenetic models: both likelihood and prior choices are discussed. Finally, the third part reviews sampling through MCMC and sequential Monte Carlo (SMC) methods. Taken together, these sections provide the necessary notation and background to discuss the specialization of the DCSMC algorithm to phylogenetics.

2.1 Biological Background

To achieve the goal of modelling evolution several ideas need to be mathematically formalized. This section serves to formalize the mutation process, the notion of evolutionary histories, and several common biological hypotheses.

2.1.1 Phylogenetic Tools

The mutation process and evolutionary history are formalized through a stochastic process for character mutations (specifically point substitutions) and a directed graph representation for evolutionary histories. Both modelling components are discussed in this section.

We assume that the evolutionary process of point substitution between characters is governed by a Markov model [11, 12]. These discrete characters could be binary to indicate the presence/absence of traits, or letters to denote DNA or protein nucleotides. The running example of a character set Σ in this work is DNA–whose four possible characters are the nucleotides $\{A, C, G, T\}$. Character substitution among these nucleotides is commonly modelled with a continuous time Markov chain (CTMC) S_t whose realizations are values in Σ. Such CTMCs are parameterized by a rate matrix Q that encodes the rate of transition between characters in Σ.

The topology of a phylogenetic tree is represented by a directed graph $\tau = (V, E)$. The set of nodes V represents extant and extinct species, and can be partitioned into sets based on biological meaning. The root of a phylogenetic tree $r \in V$ represents the most recent common ancestor of all extant species $X \subset V$. The internal nodes $V \setminus (r \cup X)$ represent speciation events, and the leaves X are extant species. The edges E represent the evolutionary relationship between an ancestor $v \in V$ and its descendants $\mathcal{C}(v) \subset V$ such that $\mathcal{C}(v) = \{c_1, c_2\}$ if $(v, c_1), (v, c_2) \in E$. Associated with the edge set E is a branch length function that maps from E to the positive real numbers $b : E \to \mathbb{R}^+$. These branch lengths $b(v_i, v_j)$ represent the genetic distance between two species $(v_i, v_j) \in E$. An important quantity related to branch lengths is the total length of a tree,

$$T = \sum_{(v_i, v_j) \in E} b(v_i, v_j). \tag{2.1}$$

The data at leaf nodes $X' \subset X$ is a set $\mathcal{Y}(X')$ of $|X'|$ lists each of which contain n values in Σ. Each list is the DNA sequence for a species $x \in X$ for each of the $|X| = M$ species. These ideas are summarized in Figure 2.1.

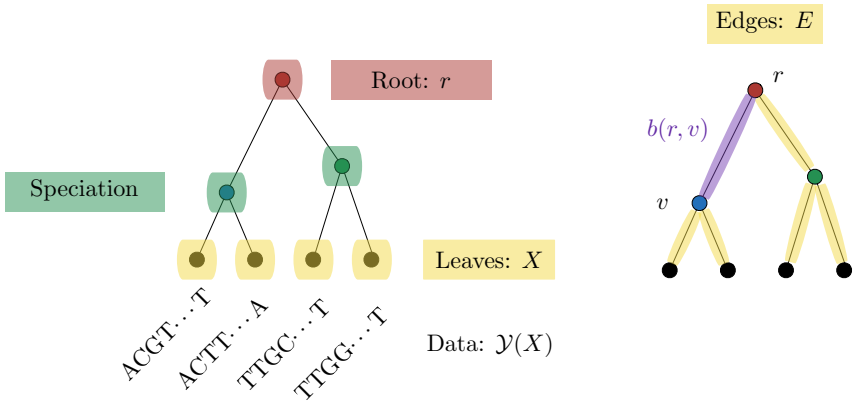


Figure 2.1: Components of Phylogenetic Trees: $\tau = (V, E)$. *(Left)* Schematic of the partition of the node set V into the root r, speciation events, and leaves X. The observed data $\mathcal{Y}(X)$ is shown as $M = |X|$ lists of n characters from a DNA character set $\Sigma = \{A, C, G, T\}$. *(Right)* Illustration of the edge set E and the branch length $b(v_i, v_j) > 0$ associated with each directed edge $(v_i, v_j) \in E$.

With the topology, branch length function, and stochastic process defined we can now discuss how these ideas are combined to form a phylogenetic tree. The stochastic process is defined on the tree through a bijective mapping from (τ, b) to an interval $[0, T]$. S_t's index set is the interval $[0, T]$. From a forward simulation perspective, the stochastic process S_t starts at the root r with a character $\sigma \in \Sigma$ drawn from the stationary distribution of Q. At speciation nodes, the stochastic process splits into two independent stochastic processes, each of which start at the state of the chain before splitting. Conditional probabilities (marginalized over all possible paths) are easily computed through the matrix exponential. For example, if $c \in \mathcal{C}(r)$ then the conditional probability at c is given as:

$$\mathbb{P}_{b(r,c)}(S_c = y | S_r = x) = [\exp(b(r, c) \cdot Q)]_{x,y}\,, \tag{2.2}$$

where $S_\bullet$ has been overloaded to denote the stochastic process S_t at nodes

r and c. The matrix exponential is defined as the infinite sum:

$$\exp(b(r,c)\cdot Q) = \sum_{i=0}^{\infty} \frac{(b(r,c)\cdot Q)^i}{i!}, \tag{2.3}$$

and can often be computed efficiently (e.g., Moler and Van Loan [28]).

In order to address time and rate confounding, Q is constructed such that the expected number of transitions in unit time is one:

$$\sum_{\sigma\in\Sigma} [Q]_{\sigma,\sigma}\,\pi_\sigma = 1, \tag{2.4}$$

where π is the stationary distribution of the CTMC,

$$\pi Q = 0. \tag{2.5}$$

Consequently, the branch lengths are in terms of the expected number of substitutions. For example, the normalized Jukes-Cantor rate matrix is:

$$Q_{JC} = \begin{bmatrix} -1 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & -1 & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & -1 & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & -1 \end{bmatrix}, \tag{2.6}$$

and has stationary distribution $\pi_{JC} = \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$. This form of the rate matrix assumes that the rate of substitution is equal for all possible character pairs; however, the family of matrices given as Generalized time-reversible matrices of Tavaré [39] relax this assumption.

2.1.2 Biological Hypotheses

Molecular Clock: A simple biological hypothesis is that the evolutionary rate across all lineages is *constant*. This hypothesis, known as the molecular clock hypothesis, was first introduced in Zuckerkandl and Pauling [45]. A consequence of this conjecture is that the distance between species is ultrametric: the distances between a species v and leaf nodes $v_i, v_j \in X$ connected by a (directed) path from v are equivalent. This implies that the distances from the root to any leaf node are also all equivalent. Ultrametric trees invite us to define a notion of tree height. The height of a node $v \in V$ is the sum of branch lengths along a path from itself to a leaf, or more precisely through the following recursive definition:

$$h(v) \quad = \quad \begin{cases} b(v,c) + h(c) & \mathcal{C}(v) \neq \emptyset \\ 0 & \mathcal{C}(v) = \emptyset \end{cases}, \tag{2.7}$$

where $c \in \mathcal{C}(v)$ can be chosen arbitrarily.

To prepare for the specialization of the DCSMC algorithm these definitions are used to define a data structure that encapsulates both tree topology and character sequences (for each node) under the clock hypothesis. As shown in Listing 2.1, the topology is represented through a list of lists, branch lengths are a function of both the list of children and their associated heights, and observed sequence data is stored as a character string. This listing complements Figure 2.1.

Although the molecular clock hypothesis is simple, empirical evidence suggests that rates of evolution *vary* across different lineages (e.g., Gillespie [17]); for example due to differing evolutionary pressures. To account for these empirical observations, relaxations of the molecular clock are the focus of our discussion in Section 3.3 and below.

Despite the empirical evidence that the molecular clock is not perfect, it remains an important conceptual tool. It provides a timescale of evolution that can be used in estimating divergence times from molecular data. Moreover, there is a rich set of unsolved inference problems under these assumptions. These challenges are addressed in Chapter 3. As a result, we

```
public class Tree<T>{
  // Properties
  private final List<Tree<T>> children;
  private final double height;
  private final T nodeLabel;
  private final String characterSequence;

  // Methods
  public Tree<T> getChildren(){
    // Return list of children
  }
  public branchLength(T node1, T node2){
    /**
    *   Recurse children to obtain node1.height and node2.height
    *   Return | node1.height - node2.height | if there is a
    *   path between node1 and node2, otherwise return RuntimeError()
    */
  }
  public getCharacterSequence(){
    // Return character sequence for current node
  }
  public boolean isLeaf(){
        // Return true if children = ∅
  }
}
```

Listing 2.1: Ultrametric Tree Class: The topology is represented as a list of lists, branch lengths are a function of both the list of children and their associated heights, and the data is stored as a character string.

first develop methods for this simple model then consider expanded models that capture rate variation across different lineages. The expanded model is detailed next.

Relaxed Clock: Relaxed clock models aim to retain some of the properties of the molecular clock, such as rooting, while simultaneously relaxing the assumption that all lineages evolve with the same rate. As described in Lepage et al. [24], these relaxations have been modelled through non-parametric [37, 38], local clock [44], and Bayesian parametric [3, 9, 18, 21, 41] methods. In this thesis the focus is on the compound Poisson process (CPP) model of Huelsenbeck et al. [18]. It is an example of a Bayesian parametric model

where the rate is modelled as a piece-wise constant function.

A CPP is a jump process J_t where events occur according to a Poisson process (PP), and the jumps U_i are each independent identically distributed random variables. That is,

$$J_t = \sum_{i=1}^{P_t} U_i, \tag{2.8}$$

where P_t is a Poisson process with rate λ. In Huelsenbeck et al. [18], the CPP construction is used to generate points of substitution rate change (according to the underlying PP) and substitution rate scalings through gamma distributed random variables $U_i \sim \text{Gamma}(\alpha, \beta)$. The illustration of the CPP given in Figure 2.2 demonstrates that events of substitution rate change give rise to differing rates of evolution. Since branch lengths are viewed as the expected number of substitutions along a branch the resulting tree is scaled. A more detailed description of the CPP is given in Section 3.3 under the context of the DCSMC algorithm.

2.2 Bayesian Phylogenetics

Bayesian phylogenetic inference is summarized through the posterior distribution for the evolutionary history of species (represented through a tree structure τ), the associated branch lengths b, and additional parameters such as substitution rates θ, given observed data $\mathcal{Y}(X)$ at the leaves. Inference of the tree topology, branch lengths, and parameters is facilitated through the posterior distribution:

$$\gamma(\tau, b, \theta|\mathcal{Y}(X)) = \frac{l(\mathcal{Y}(X)|\tau, b, \theta)\pi(\tau, b, \theta)}{\int l(\mathcal{Y}(X)|\tau, b, \theta)\pi(\,\mathrm{d}\tau, \,\mathrm{d}b, \,\mathrm{d}\theta)}, \tag{2.9}$$

where $\pi(\tau, b, \theta)$ is the joint prior density over (τ, b, θ) and $l(\mathcal{Y}(X)|\tau, b, \theta)$ is the likelihood of the data. Posterior inference is challenging computationally due to the analytically intractable multidimensional integral:

$$Z = \int l(\mathcal{Y}(X)|\tau, b, \theta)\pi(\,\mathrm{d}\tau, \,\mathrm{d}b, \,\mathrm{d}\theta).$$

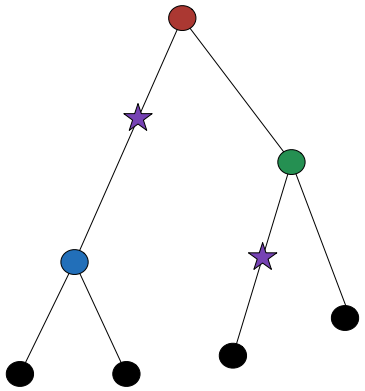


Figure 2.2: Relaxed Molecular Clock via Compound Poisson Process: Illustration of the relaxed molecular clock via the compound Poisson process. Stars represent realizations from a Poisson process for the location of substitution rate change. Substitution rate change occurs just prior (in evolutionary time) to the event of rate change. The rate is modified by scaling the branch length immediately before the event by a gamma random variable. Since the tree is drawn such that distances are in terms of the expected number of substitutions the scalings are illustrated by stretching the tree on lineages where a scaling event occurred.

In this work we assume that the topology τ and rate matrix parameters θ are fixed. Thus Equation 2.9 simplifies to:

$$\gamma(b|\mathcal{Y}(X)) \quad = \quad \frac{l(\mathcal{Y}(X)|b)\pi(b)}{\int l(\mathcal{Y}(X)|b)\pi(\,\mathrm{d}b)} = \frac{\bar{\gamma}(b|\mathcal{Y}(X))}{Z}, \tag{2.10}$$

where $Z = \int l(\mathcal{Y}(X)|b)\pi(\,\mathrm{d}b)$ is still analytically intractable.

Probabilistic phylogenetic models can be formed by defining different prior densities $\pi(\cdot)$ or likelihood models $l(\cdot|\cdot)$. A common model choice is independent CTMCs for each character in a DNA sequence. Under these assumptions the likelihood of the leaf data given branch lengths is $l(\mathcal{Y}(X)|b) =$

$$\prod_{i=1}^{n}\sum_{\sigma_r}\sum_{\sigma_{v_{i_1}}}\cdots\sum_{\sigma_{v_{i_{M-2}}}}\left[\prod_{j=1}^{M-2}\mathbb{P}_{b(\mathcal{P}(v_j),v_j)}\left(S^i_{v_j}=\sigma_{v_j}|S^i_{\mathcal{P}(v_j)}=\sigma_{\mathcal{P}(v_j)}\right)\right]\times \left[\prod_{j=1}^{M}\mathbb{P}_{b(\mathcal{P}(x_j),x_j)}\left(S^i_{x_j}=\mathcal{Y}(x^i_j)|S^i_{\mathcal{P}(x_j)}=\sigma_{\mathcal{P}(x_j)}\right)\right]\times\pi_{\sigma_r}, \tag{2.11}$$

where $\{\sigma_{v_{i_1}},\ldots,\sigma_{v_{i_{M-2}}}\}$ denote the values of the internal nodes of τ, $X=\{x_1,\ldots,x_M\}$, S^i_v is the value of the stochastic process at node v for the i^{th} character, and $\mathcal{Y}(x^i_j)$ denotes the i^{th} observed character for the j^{th} taxa. For example, consider the labelled tree given in Figure 2.3. In that case, Equation 2.11, is:

$$\begin{aligned}\sum_{\sigma_r}\sum_{\sigma_{v_1}}\sum_{\sigma_{v_2}}&\pi_{\sigma_r}\mathbb{P}_{b(r,v_1)}\left(S_{v_1}=\sigma_{v_1}|S_r=\sigma_r\right)\mathbb{P}_{b(r,v_2)}\left(S_{v_2}=\sigma_{v_2}|S_r=\sigma_r\right)\times\\ &\mathbb{P}_{b(v_1,x_1)}\left(S_{x_1}=\mathrm{C}|S_{v_1}=\sigma_{v_1}\right)\mathbb{P}_{b(v_1,x_2)}\left(S_{x_2}=\mathrm{A}|S_{v_1}=\sigma_{v_1}\right)\times\\ &\mathbb{P}_{b(v_2,x_3)}\left(S_{x_3}=\mathrm{C}|S_{v_2}=\sigma_{v_2}\right)\mathbb{P}_{b(v_2,x_4)}\left(S_{x_4}=\mathrm{T}|S_{v_2}=\sigma_{v_2}\right).\end{aligned} \tag{2.12}$$

In Chapter 3, methods to calculate Equation 2.11 efficiently via the so-called peeling recursions introduced by Felsenstein [11, 12] are discussed.

This likelihood can be conceptualized as follows. Assume that character locations are independent and first consider only one site. The likelihood of observing the leaf data at one site is obtained by marginalizing over all possible internal node and root values. Finally, since independence across sites is assumed, the full likelihood is the product over all n sites.

With the likelihood model defined, we turn attention to the prior distribution over branch lengths. The methods developed in this thesis are designed for clock and relaxed clock trees [18] which imply that the tree topology is rooted. Therefore, the branch length prior is the only remaining component required to complete the Bayesian model specification. For a detailed list of branch length priors for rooted trees see Chen et al. [5, Chapters 2.3-2.4]. The one considered here is the Yule prior.

The Yule process is a convenient form of the birth-death process where the death rate is set to zero. This process induces a distribution over branch lengths by defining a stochastic process of cladogenesis from ancestors to extant species. In particular, the Yule process induces branch lengths through

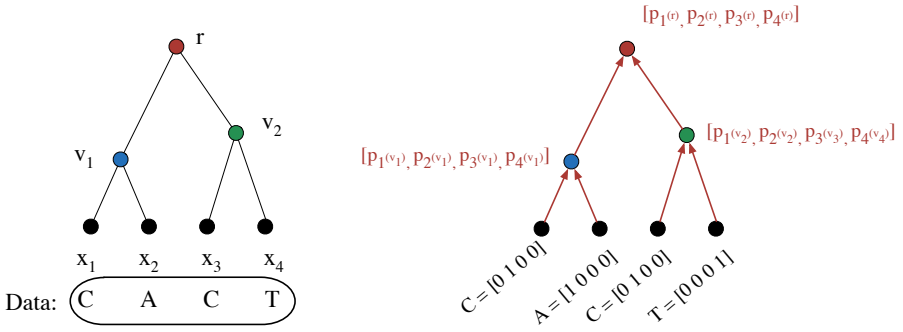


Figure 2.3: Labelled Tree with Data and Message Passing Algorithm: (Left) Labelled phylogenetic tree with leaf data. (Right) Schematic for message passing algorithm. Leaf characters are coded as row matrices, messages propagate from the leaves through internal nodes to the root. Messages represent the conditional probability of each character for internal nodes given the descended observed data. The likelihood of the tree is given as the sum $\sum_{i=1}^{4} p_i(r)\pi(i)$.

speciation times whose increments are exponentially distributed with rate parameter proportional to the number of speciation events from the root to the current node. This process is depicted in panel I of Figure 3.3. Mathematically, this prior is defined as follows. Let $\mathbf{h}$ be a list of M sorted heights for (τ, b) and let $\mathbf{d}$ a list of height increments obtained from $\mathbf{h}$ via differencing the elements of $\mathbf{h}$: $d_i = h_i - h_{i-1}$ for $i = 1, \ldots, M-1$. The density of the Yule prior is defined as:

$$\pi(b) = \prod_{i=1}^{M-1} (i+1) \exp\left((i+1) g_i(b)\right) \tag{2.13}$$

$$= \prod_{i=1}^{M-1} (i+1) \exp\left((i+1) d_i\right), \tag{2.14}$$

where $g_i(b) = d_i$.

The models discussed so far implicitly assume that the evolutionary process follows the molecular clock hypothesis. Loosely speaking, this hypothesis states that the rate of character change is uniform across all branches. Although numerous empirical studies have provided evidence to the contrary, it remains difficult to conduct (large scale) inference on models that relax this hypothesis to allow for rate variation across branches. The DCSMC framework is envisaged to perform well in modelling scenarios where these types of assumptions are relaxed via a CPP construction as in Huelsenbeck et al. [18].

2.3 Sampling Background

This section provides an overview of particle filtering methods starting from its foundations in Importance Sampling (IS) and incrementally working towards SMC.[1] The discussion serves as a way to set notation and link these ideas for sampling on product spaces, to general spaces, to the development of sampling via tree decompositions in DCSMC.

Given a difficult to sample target distribution $\gamma(x) = \bar{\gamma}(x)/Z$, where $\bar{\gamma}(x)$ is easy to evaluate point-wise, and where $Z = \int \bar{\gamma}(x)\,\mathrm{d}x$ is difficult to calculate, our aim is to approximate $\gamma(x)$ via a collection of N weighted particles $(x^i, w^i)_{i=1}^N$. Our desiderata for these collections of weighted particles is that for well behaved measurable functions $f : \mathcal{X} \to \mathbb{R}$ the following result holds:

$$\frac{1}{\sum_{j=1}^N w^j} \sum_{i=1}^N w^i f(x^i) \quad \xrightarrow{\mathbb{P}} \int f(x)\gamma(x)\,\mathrm{d}x, \quad \text{as } N \to \infty. \tag{2.15}$$

Different sampling methods can be distinguished in the way that such collections of weighted particles are generated. For example, MCMC generates a collection of samples whose weights are equal (and deterministic), perfect sampling (either via coupling from the past [31, 32] or other exact methods) generates collections with equal deterministic weights, whereas particle

[1] An excellent introduction to particle filtering methods is provided in Doucet and Johansen [7].

methods often generate collections with unequal random weights.[2] An example of a sampling method with unequal random weights is presented next.

2.3.1 Importance Sampling

A common method to generate samples from a target γ is via rejection sampling. In rejection sampling a proposal distribution q whose support is a superset of the target's support is used to generate candidate samples from γ. To determine whether each candidate sample is kept as a sample from γ or rejected a randomized algorithm is used. The algorithm's accept-reject decisions are based on the probability of drawing these candidate samples from γ.

IS builds on rejection sampling by noticing that rejection can be computational expensive (especially for "bad" proposals). To ameliorate this inefficiency, samples generated from the proposal are never discarded, but instead weighted based on the evidence that they arose from γ. The weights should be viewed as a correction for the discrepancy between the proposal distribution and the target.

The rationale of this modification to rejection sampling is shown via the law of large numbers. Let $x^i \sim q(\cdot), i = 1, \ldots, N$ and $w^i = \frac{\bar{\gamma}(x^i)}{q(x^i)}$ be a weight assigned to each sample x^i. Then for well behaved measurable functions f the strong law of large numbers, and Slutsky yields:

$$\frac{1}{\sum_{j=1}^{N} w^j} \sum_{i=1}^{N} w^i f(x^i) \xrightarrow{a.s} \frac{1}{\int \frac{\bar{\gamma}(x)}{q(x)} q(\,\mathrm{d}x)} \int f(x) \frac{\bar{\gamma}(x)}{q(x)} q(\,\mathrm{d}x) \tag{2.16}$$

$$= \frac{1}{Z} \int f(x)\bar{\gamma}(x)\,\mathrm{d}x = \int f(x)\gamma(x)\,\mathrm{d}x. \tag{2.17}$$

This consistency result is encouraging; however, the asymptotic variance (see Doucet and Johansen [7, Eqn. 27]),

$$\frac{1}{N} \int \frac{\gamma^2(x)}{q(x)} (f(x) - \mathbb{E}(f(x)))^2\,\mathrm{d}x, \tag{2.18}$$

is known to be large in simple scenarios [7]. The next two modifications to IS improve on this issue by incorporating statistically efficient proposal strate-

[2] If resampling is used the weights are renormalized and each particle has equal weight.

gies and variance reduction methods via particle resampling techniques.

2.3.2 Sequential Importance Sampling

We first address the inefficiency of IS when known model decompositions exist. When the target is defined on a high dimensional space and admits a "sequential" decomposition the proposals used in importance sampling could be inefficient (both computationally and statistically). Suppose x can be written as $(x_1, x_2, \ldots, x_m) = x_{1:m}$. sequential importance sampling (SIS) modifies importance sampling by proposing component x_j conditionally on $x_{1:j-1}$ and sequentially updating weights. In particular, the proposal can be written as the product [7]:

$$\begin{aligned} q_m(x^i_{1:m}) &= q_{m-1}(x^i_{1:m-1}) q_m(x^i_m | x^i_{1:m-1}) && (2.19) \\ &= q_1(x^i_1) \textstyle\prod_{k=2}^m q_k(x^i_k | x^i_{1:k-1}), && (2.20) \end{aligned}$$

with unnormalized weights given as:

$$w^i_m = \frac{\bar{\gamma}_m(x^i_{1:m})}{q_m(x^i_{1:m})} = \frac{\bar{\gamma}_{n-1}(x^i_{1:m-1})}{q_{m-1}(x^i_{1:m-1})} \frac{\bar{\gamma}_m(x^i_{1:m})}{\bar{\gamma}_{m-1}(x^i_{1:m-1}) q_n(x^i_{1:m})}. \quad (2.21)$$

This telescoping product yields weight updates given by:

$$w^i_m = w_{m-1}(x^i_{1:m-1}) \alpha_m(x^i_{1:m}) = w^i_1 \textstyle\prod_{k=2}^m \alpha^i_k, \quad (2.22)$$

with incremental weights

$$\alpha^i_k = \frac{\gamma_m(x^i_{1:m})}{\gamma_{m-1}(x^i_{1:m-1}) q_m(x^i_m | x_{1:m-1})}. \quad (2.23)$$

Data received in an online fashion is amenable to these types of decompositions and sequential updates. Examples include target tracking and stochastic volatility models. Although proposing x via sequential proposals can be more efficient, SIS suffers from the same high variance as IS. Moreover, as particles are sequentially updated in SIS, the corresponding set of (normalized) weights often concentrates on values close to zero with a few particles close to one. This is known as particle degeneracy since only a few

particles contribute to the target approximation. SMC addresses this issue by adding a resampling step between sequential proposals.

2.3.3 Sequential Monte Carlo

As alluded to in the previous section, SMC addresses the high variance found in both IS and SIS and particle degeneracy by adding a resampling step after each weight update. The details of this scheme are illustrated in Algorithm 1, for more detail see Doucet and Johansen [7].

Algorithm 1 `smc`:

Sample $x_1^i \sim q_1(x_1)$
Compute weight $w_1^i = \frac{\bar{\gamma}_1(x_1^i)}{q_1(x_1^i)}$
Resample $(x_1^i, w_1^i)_{i=1}^N$ to obtain equally weighted particle system $(\check{x}_1^i, 1)_{i=1}^N$

for $c = 2, \ldots, m$ **do**
  Sample $x_c^i \sim q_c(x_c|\check{x}_{1:c-1})$
  Set $x_{1:c}^i \leftarrow (\check{x}_{1:c-1}^i, x_c^i)$
  Compute incremental weight $\alpha_{1:c}^i = \frac{\bar{\gamma}_c(x_{1:c}^i)}{\bar{\gamma}_{c-1}(x_{1:c-1}^i)q_c(x_c^i|x_{1:c-1}^i)}$
  Resample $(x_{1:c}^i, w_{1:c}^i)_{i=1}^N$ to obtain equally weighted particle system $(\check{x}_{1:c}^i, 1)_{i=1}^N$
end for
return Equally weighted particle system $(\check{x}_{1:n}^i, 1)_{i=1}^N$

The upshot of resampling is that the particle system is "reset" whenever resampling occurs [7]. This can be shown to significantly reduce the number of particles required to achieve reasonable performance in terms of the ratio of asymptotic variance of $\hat{Z}$ to Z.

The discussion of particle methods has thus far focused on specific assumptions of the space $(X, \mathcal{X})$. IS assumed that efficient proposals existed for the (potentially high dimensional) space, whereas SIS and SMC assumed a sequential decomposition. The sequential decomposition was originally used to create more efficient proposals by sequentially building a weighted collection of particles on $(X, \mathcal{X})$. SMC introduced resampling to alleviate the high variance in both IS and SIS. However, particle methods are not lim-

ited to these specialized domains. In fact, a general framework (confusingly) named SMC samplers has been developed to cover general spaces [6].

As mentioned in the introduction, Bouchard-Côté et al. [4] introduced particle filters for phylogenetics by using techniques similar to the agglomerative clustering methods of Teh et al. [40]. The sequential decomposition was possible by building the tree topology in a sequential manner. We investigate another model decomposition for phylogenetics when τ is fixed.

Chapter 3

Phylogenetic Divide and Conquer Sequential Monte Carlo

This chapter delineates the contributions made through this thesis work to the phylogenetic sampling literature. First an example is given to show how phylogenetic inference proceeds with the IS method introduced in Section 2.3.1. This example motivates the introduction of a specialized version of the DCSMC algorithm which follows. Throughout, specialized DCSMC data structures and pseudocode are introduced to aid exposition and facilitate easy reproduction. Next, several theoretical results for general DCSMC algorithms are described. The chapter wraps up with a discussion of how the DCSMC algorithm can be easily used for relaxations of classical biological assumptions and a notes a few extensions.

3.1 Phylogenetic Inference with IS

To frame Bayesian phylogenetic inference in terms of particle filters we first show how posterior inference of γ is completed under simple IS. The variable of interest is the list of branch lengths b. These branch lengths can be described through a list of height increments $\mathbf{d}$ and an ordering on the

internal nodes as in Figure 3.1 (Left). An obvious proposal distribution q is a product of independent exponential distributions as in the Yule prior (see Figure 3.3 and Equation 2.13) and a permutation over the internal nodes. Figure 3.1 *(Bottom-Right)* shows an example realization from this proposal. The importance weights are assigned to each particle according to:

$$w^i = \frac{\bar{\gamma}(b^i)}{q(b^i)}, \tag{3.1}$$

as in Figure 3.1 (Top-Right).

This procedure can be computationally inefficient in scenarios where inference is performed on large trees. Since scalable inference motivates this work the simple IS method is unacceptable. However, instead of proposing jointly over all height increments and orderings, a model decomposition can be employed to increase the statistical efficiency. In contrast to SIS and SMC methods, this decomposition is not sequential in nature, but rather makes use of the topology of the phylogenetic tree. A general overview of the DCSMC methodology follows.

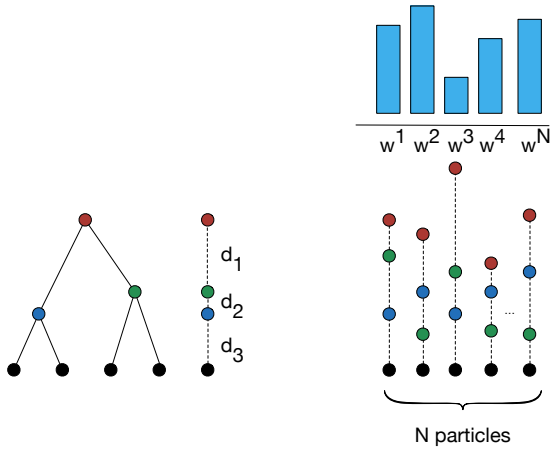


Figure 3.1: Importance Sampling in Phylogenetics: *(Left)* Phylogenetic tree and height increments **d**. *(Bottom-Right)* N-particle population of height increments. *(Top-Right)* IS particle weights for the N-particle population of height increments.

3.2 Phylogenetic Inference with DCSMC

In contrast to the sequential decompositions often found in hidden Markov models, phylogenetic models are endowed with a tree topology that allows for model decompositions based on the tree structure. The DCSMC algorithm, described in Lindsten et al. [26], is a novel variation of the typical particle filtering methodology designed for models endowed with a tree decomposition. Complementary to the sequential decompositions already in use by SIS and SMC, DCSMC uses a(n) (auxiliary) tree structure to recursively decompose the model into computationally easy submodels. In DCSMC each subproblem is endowed with its *own* population of weighted particles which are merged at intermediate steps.

In particular, DCSMC recurses through $\mathcal{C}(r)$ to the leaves, proposes according to intermediate distributions γ_v for each $v \in V$ defined on $B_v = \left(\bigotimes_{c\in\mathcal{C}(v)} B_c\right) \times \tilde{B}_v$, and propagates forward to v's parents $\mathcal{P}(v)$ by a resampling step. The incremental set $\tilde{B}_v$ can be chosen arbitrarily, and is equal to $\mathbb{R}^+$ for the molecular clock case. The only restriction on the set of intermediate distributions is that $\gamma_r(\cdot) = \gamma(\cdot)$; however, in practice, the choice of intermediate distributions can greatly affect the quality of the resulting approximations. A full description of DCSMC follows in Algorithm 2.

In the phylogenetic setup, the target distribution γ is the posterior distribution of branch lengths $\gamma(b|\mathcal{Y}(X))$. DCSMC methods approximate this posterior with a weighted collection of branch lengths and an estimate of the marginal likelihood $\left((b_v^i, w_v^i)_{i=1}^N, \hat{Z}_v^N\right)$. The phylogenetic notation introduced earlier is overloaded so that, b_v denotes the restriction of the branch length function to edges that are connected to the subtree with pseudo-root v and the superscript i indexes each particle.

To develop the specialization of the DCSMC method to phylogenetics, we consider an object oriented representation of particle data structures and the main algorithmic components of DCSMC. These are described through explicit data structures and pseudocode. In the object oriented representations both object properties and methods are presented for each class.

Algorithm 2 `dc_smc(v)`: // by convention $\prod_{c\in\emptyset}(\cdot) = 1$

for $c \in \mathcal{C}(v)$ **do**
  $((b_c^i, w_c^i)_{i=1}^N, \hat{Z}_c^N) \leftarrow$ `dc_smc(c)`
  Resample $(b_c^i, w_c^i)_{i=1}^N$ to obtain equally weighted particle system $(\check{b}_c^i, 1)_{i=1}^N$
end for
for particle $i = 1, 2, \ldots, N$ **do**
  Simulate $\tilde{b}_v^i \sim q_t(\cdot|\check{b}_{c_1}^i, \check{b}_{c_2}^i)$ from proposal kernel defined on $\mathbb{R}^+$
  Set $b_v^i = (\check{b}_{c_1}^i, \check{b}_{c_2}^i, \tilde{b}_v^i)$
  Compute $w_v^i = \frac{\bar{\gamma}_v(b_v^i)}{\prod_{c\in\mathcal{C}(v)} \bar{\gamma}_c(\check{b}_c^i)} \frac{1}{q_v(\tilde{b}_v^i|\check{b}_{c_1}^i, \check{b}_{c_2}^i)}$
end for
Compute $\hat{Z}_t^N = \left[\frac{1}{N}\sum_{i=1}^N w_v^i\right] \prod_{c\in\mathcal{C}(t)} \hat{Z}_c^N$
return $((b_v^i, w_v^i)_{i=1}^N, \hat{Z}_v^N)$

First we formalize a computational strategy to calculate the data likelihood, introduced in Equation 2.11. To this end, we revisit Equation 2.11 to discuss its implementation through an efficient message passing framework. The key observation by Felsenstein [11, 12] in efficiently calculating the likelihood of the data is that if Equation 2.11 is calculated naïvely the (full) sum contains 4^{M-1} terms, whereas if the summation signs are moved inwards the number of computations is reduced. For example, in Equation 2.12, the total number of terms is 64 ($= 4^3$). However, by "moving the summations inwards" one can obtain the computationally efficient peeling-recursion of Felsenstein [12].

The right panel of Figure 2.3 depicts the message passing view of the peeling algorithm. Leaf data is viewed as a row vector denoting the observed character. The messages from descendant nodes propagate to the parent node via the conditional likelihood described above. The parent message from each character is represented as a row vector containing the conditional likelihood for each character. The total likelihood is given as the dot product between the stationary distribution and the message at the root. This is equivalent to the peeling recursion.

The data structure of phylogenetic particles is visited next. In general, these particles need to be able to determine their height, calculate the evolutionary distance between descendant particles, create new particles based on the merging of descendant particles, and compute the unnormalized posterior density of itself. These requirements are summarized in the `PhyloParticle<P>` interface given in Listing 3.1.

This interface design allows for efficient code reuse for the two different envisaged models of evolutionary change. Each model, the molecular clock and relaxed clock, is associated with a particle class that implements the `PhyloParticle<P>` interface. The implementation for the molecular clock `ClockParticle` is given in Listing 3.2.

```
public interface PhyloParticle<P>{
  public P create(P particle1, P particle2, HeightScaling heightScaling);
  public double getHeight();
  public double getEvolutionaryDistance(P particle1, P particle2);
  public double getLogPosterior();
}
```

Listing 3.1: Particle Interface

In addition to the methods required by the `PhyloParticle` interface, clock particles require four fields to complete their specification. Listing 3.2 develops this implementation of the `PhyloParticle` interface. Specifically, a list of children provides a set of pointers to descendant particles, a list of sorted heights serves as node speciation times, propagated messages from descendant particles are represented as a matrix where each row corresponds to a character (within the character sequence) and columns correspond to the propagated message. The log posterior is also stored for each particle.

Intermediate steps within the DCSMC algorithm call for new particle creation based on existing particles. Within the `ClockParticle` setup this is particularly straightforward. The new particle's children are the two merged particles, speciation times are a union over the speciation times within each particle to be merged and the speciation time of the merged particle, and the new root message is calculated by propagating the messages of each

descendant particle. The message is calculated via the evolutionary distance between each particle and their parent according to the CTMC S_t.

```
public class ClockParticle implements PhyloParticle<ClockParticle>{
  // Properties
  public final List<ClockParticle> children;
  public final List<Double> sortedHeights;
  public final double[][] rootMessage;
  public final double logPosterior;

  // Methods
  public double getHeight(){
  // Return first element of sortedHeights
  }
  public ClockParticle
    create(ClockParticle particle1, ClockParticle particle2,
           HeightScaling heightScaling){
    /**
    * children = (particle1, particle2)
    * sortedHeights = sort(particle1.sortedHeights ∪
    *                      particle2.sortedHeights ∪ heightScaling.height)
    * rootMessage = propogateMessage(particle1, particle2)
    */
  }
  public double
    getEvolutionaryDistance(ClockParticle parent, ClockParticle descendant){
    // Return parent.getHeight() - descendant.getHeight()
  }
  public double getLogPosterior(){
  // Return logPosterior
  }
}
```

Listing 3.2: Clock Particle

With the particle and the underlying tree structure defined as objects, we now formulate the DCSMC algorithm in Listing 3.3. Listing 3.3's fields are a proposal distribution (to be defined precisely shortly), and a random seed that governs all randomness in the algorithm. The principal method of this class, `dcSMC`, generates a population of particles `ParticlePopulation` based on the DCSMC algorithm. Given a fixed tree topology, the algorithm recurses to the leaves of the tree through the tree's list of children. At

the leaves, independent `ParticlePopulation`s are created, resampled, and then propagated forward through the `merge` function. Merging involves creating new particles based on the previous two populations of particles and a proposal for the height of the merged particle. The weight of each particle in log-scale is the difference between the log posterior of the particle and the sum of the children's log posterior and the log proposal density.

```
public class PhyloDCSMC<P extends PhyloParticle<P>, T>{
  // Properties
  private final Proposal<P> proposal;
  private final Random random;

  // Methods
  public ParticlePopulation<P> dcSMC(Tree<T> tree){
    if (tree.isLeaf())
      return new ParticlePopulation<P>();
    // Recurse tree
    ParticlePopulation<P> leftParticles =  dcSMC(tree.getChildren.get(0))
    leftParticles.resample(random);  // Multinomial resampling
    ParticlePopulation<P> rightParticles =  dcSMC(tree.getChildren.get(1))
    rightParticles.resample(random);  // Multinomial resampling
    return merge(leftParticles, rightParticles);
    }
  public ParticlePopulation<P> {
      merge(ParticlePopulation leftParticles, ParticlePopulation rightParticles){
    P[] newParticles = new P[N]; // Array of new particles
    double[] logWeights = new double[N]; // Array of particle weights
    /**
    * New particles are formed from leftParticles and rightParticles
    * Proposed heights given by Proposal. For each i = 1,...,N
    */
    newParticles[i] = proposal(random, leftParticles[i], rightParticles[i]);
    logweights[i] = calculateLogWeight(newParticles[i])
        return new ParticlePopulation<P>(newParticles, logWeights);
  }
  private calculateLogWeight(P particle){
    // Calculate sum of particle's children logPosterior (kidsLogPosterior)
    // Calculate logDensity of proposed increment by Proposal (proposalLogDensity)
    return particle.logPosterior - kidsLogPosterior - proposalLogDensity;
  }
}
```

Listing 3.3: DCSMC Pseudocode

3.2.1 Intermediate Distributions

To complete the DCSMC specialization to phylogenetics the intermediate distributions γ_v and the proposals q_v must be defined for all $v \in V$. Although phylogenetic models considered here admit natural decompositions through tree structures, a challenge arises in specifying high quality intermediate target γ_v and corresponding proposal q_v distributions. Both intermediate targets and proposals are easy to specify since DCSMC only requires that the root "intermediate" distribution is equal to the target distribution $\gamma_r(\cdot) = \gamma(\cdot)$. Nevertheless, since the particle population's approximation of the posterior is often determined by the intermediate and proposal distributions judicious choices must be made. These choices are difficult as a priori knowledge of the posterior shape is generally unknown.

For the most part, in SMC algorithms the accuracy of the posterior approximation is driven by the correspondence between target and proposal distributions. In Bayesian statistics the prior distribution is meant to encapsulate our beliefs of the parameter of interest a priori and it thus is a natural candidate for the proposal. While this choice is convenient it may lead to a poor approximation if the posterior does not resemble the prior. We examine two different intermediate and proposal distributions in the following discussion.

Since the prior over the full tree is Yule based we attempt to extend this birth process for subtrees. The main challenge in extending the Yule process for subtrees arises due to independent particle populations inherent to DCSMC. In particular, independence implies that there is no information sharing across speciation events. Therefore, the ordering of speciation events of a subtree is unknown with respect to the merged tree of v_1 and v_2.

An example of a simple strategy to assign Yule-like priors to subtrees is taken up in Figure 3.2. This naïve strategy ignores the prior tree structure knowledge (in addition to its ignorance of speciation times from other populations), and considers each subtree in isolation. Precisely, each subtree is assumed to be its own tree and the usual Yule prior is employed. The simple nature of this prior distribution is appealing; however, it may be desirable

to explicitly incorporate a priori knowledge of the tree structure since this information could decrease the discrepancy between weight updates. This is described in detail in the following paragraphs.

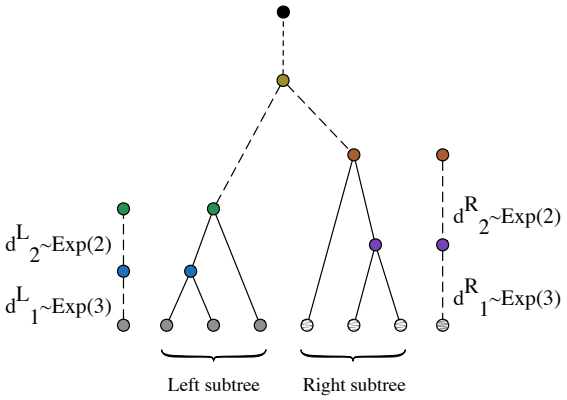


Figure 3.2: Naïve Yule Prior: Each subtree is viewed in isolation to the whole tree structure.

The following intermediate specification improves upon the simple strategy by incorporating knowledge of the tree structure through the number of taxa M. Let $X_v \subset X$ be the set of leaves descendant from species $v \in V$, and $\pi_v(b_v)$ be a modified form of the Yule prior chosen so that (minimally) $\pi_r(b_r) = \pi(b)$. In particular, let the intermediate prior have density,

$$\pi_v(b_v) = \prod_{i=1}^{|X_v|-1} (M - |X_v| + i + 1) \exp\left((M - |X_v| + i + 1) g_i(b_v)\right) \tag{3.2}$$

$$= \prod_{i=1}^{|X_v|-1} (M - |X_v| + i + 1) \exp\left((M - |X_v| + i + 1) d_i\right), \tag{3.3}$$

where $g_i(b_v) = d_i$. The corresponding intermediate proposal $q_v(\cdot)$ has the exponential density with rate parameter $M - |X_v| + 2$. This prior is shown in panel II of Figure 3.3.

Comparing Figure 3.2 and panel II of Figure 3.3 to the Yule prior over the full tree in panel I of Figure 3.3 illustrates potential disagreement in the distribution of the joint Yule process prior for height increments and the distribution on increments within each subtree. The flexibility of DCSMC

allows for disagreement between the intermediate priors and proposal distributions. These differences are incorporated (along with the likelihood) in the particle system weights and are subsequently corrected in merged samples; however, the quality of the posterior distribution via weighted particles depends upon the intermediate priors and proposals. Since the modified Yule prior decreases the discrepancy versus the naïve prior the modified setup is expected to provide a higher quality approximation to the posterior distribution, especially if the prior dominates the likelihood in the posterior.

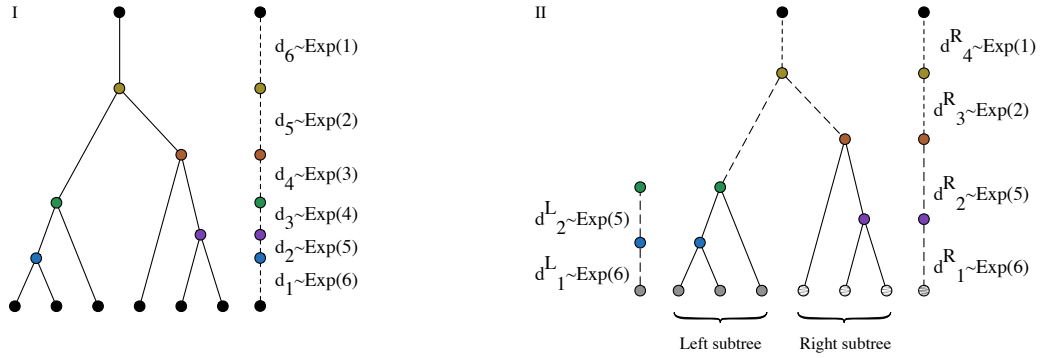


Figure 3.3: Yule Prior: *(I)* Branch length prior induced by the Yule process for cladogenesis over speciation times (represented as height increments **d**). The time between speciation events is exponentially distributed with rate parameter proportional to the total number of speciation events between the node of interest and the root. *(II)* An adapted branch length prior based on the Yule process for DCSMC. In DCSMC setups intermediate distributions are independent and thus each subtree lacks full knowledge of other subtrees. A consequence of independence is a disagreement between the Yule prior on the full tree versus subtrees. One solution is illustrated in *(II)* where the prior over branch lengths is exponentially distributed with rate proportional to the number of total species less the total number of leaves connected to the root of the subtree.

The intermediate target density for subpopulation $v \in V$ is defined as $\gamma_v \propto l_v(\mathcal{Y}(X_v)|b_v)\pi_v(b)$. This yields intermediate weights:

$$w_v^i = \frac{l_v(\mathcal{Y}(X_v)|b_v^i)}{l_{c_1}(\mathcal{Y}(X_{c_1})|b_{c_1}^i)l_{c_2}(\mathcal{Y}(X_{c_2})|b_{c_2}^i)} \frac{1}{q(b^i|b_{c_1}^i, b_{c_2}^i)}, \tag{3.4}$$

where $\{c_1, c_2\} = \mathcal{C}(v)$. Since the root target distribution $\gamma_r \equiv \gamma$ this choice of intermediate distributions satisfies the requirements of DCSMC.

The proposal distribution is implemented as a class containing the number of taxa (as the prior tree knowledge), and a `propose` method that returns a new particle whose height increment is exponentially distributed. Pseduocode is presented in Listing 3.4. Future work is aimed at examining the posterior approximation with different proposals.

```
public class Proposal<P extends PhyloParticle<P>>{
  // Properties
  private final int M;

  // Methods
  public P propose(Random random, P particle1, P particle2){
  /**
  * Determine number of leaves descendant from particle1 and particle2 |X_v|
  * Propose increment ~ Exponential(rate = M - |X_v| + 2)
  * Height of subtree is max(particle1.height, particle2.height) + increment
  * Return a new P with this height
  */
  }
}
```

Listing 3.4: Yule-like Proposals

In summary, a schematic showing the differences between the IS algorithm and DCSMC algorithm applied to phylogenetics is shown in Figure 3.4. As opposed to IS, this algorithm can be roughly divided into four different steps for each recursion level: (I) Independent particle populations are proposed and weighted for each (of the two) subtrees; (II) Multinomial resampling is carried out for each particle population to create two new equally weighted particle populations; (III) These particle populations are merged based on the multinomial induced permutation on particle indices;

(IV) New height increments are proposed and weighted (if at the root then the algorithm terminates; otherwise the algorithm continues to step II). The gains of the divide and conquer approach are from a sparse representation of the parameter space. Specifically, the binary speciation events used to recursively decompose the model yield two independent populations of N particles. Each population covers N^2 points, but only requires $O(2N)$ storage.[1]

Intuitively, the efficiency of SMC methods over MCMC methods can be divided into statistical and computational reasons. From a statistical view, the proposal distribution used in SMC methods are often designed to create a set of particles covering a large part of the sample space. In contrast, MCMC methods rely on a kernel distribution to sequentially explore the target space through accept-reject steps. Serial exploration of the sample space via MCMC methods are prone to finding local optima, especially if exploration requires visiting areas of lower probability before reaching areas of higher probability (since these paths are often rejected). These exploration and coverage issues are inherent to MCMC, but are not necessarily present in SMC algorithms. From a computational perspective, the serial nature of MCMC prohibits easy parallelization of these algorithms across multiple computer cores or servers. On the other hand, the tree decomposition of DCSMC into independent particle populations is natural to parallelize. These two reasons provide justification for our belief that DCSMC could show efficiency gains over standard MCMC techniques for large trees and more complex models of evolution.

[1] As noted in Lindsten et al. [26], the most basic DCSMC procedure is essentially a sequential importance resampling method. Such methods may perform (significantly) better if in each intermediate step the intermediate target distribution is slowly reached through annealing in the intermediate likelihood.

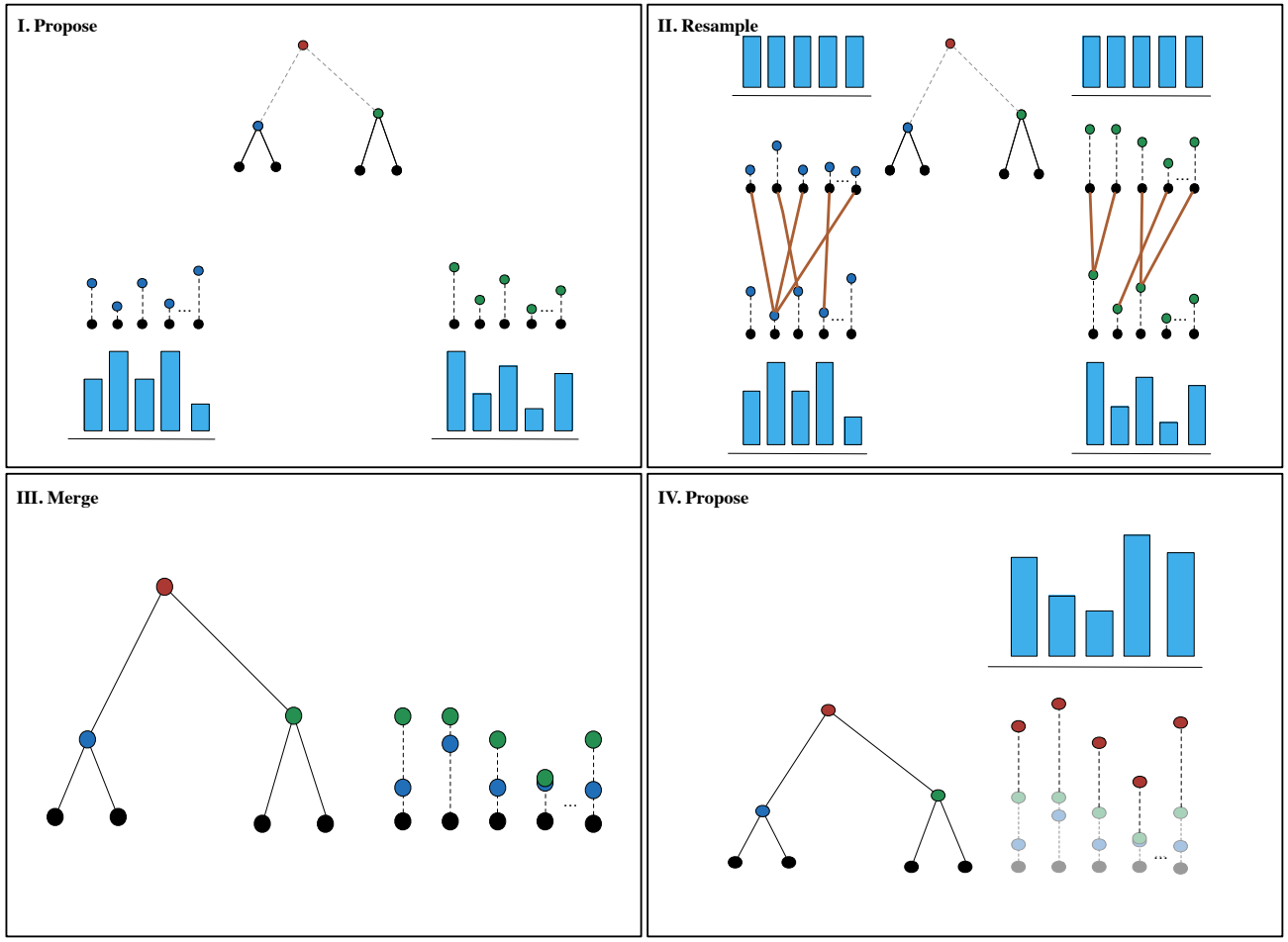


Figure 3.4: Phylogenetic DCSMC: *(I. Propose)* For each child, $\{c_1, c_2\} \in \mathcal{C}(r)$, propose independent particle populations according to proposals q_{c_1}, q_{c_2} and weight according to DCSMC intermediate weights. *(II. Resample)* Conduct multinomial resampling independently according to the particle weights to each particle populations. *(III. Merge)* Merge the particle populations according to the induced permutation from resampling. *(IV. Propose)* Propose an additional height increment and weight according to $\bar{\gamma}_r$.

3.2.2 Theoretical Implications

The DCSMC algorithm has two theoretically important guarantees: unbiasedness of the marginal likelihood estimator and consistency of the particle system. These results are captured in the following two propositions from Lindsten et al. [26].

Proposition 1 (Proposition 1 [26])**.** *Assume that the auxiliary tree used for decomposition is a balanced binary tree, that appropriate absolute continuity requirements are met, and that multinomial sampling is carried out for every population during every iteration. Then* $\mathbb{E}\left(\hat{Z}_r^N\right) = Z_r$ *for any* $N \geq 1$.

Proposition 2 (Proposition 2 [26])**.** *Under regularity conditions (see Lindsten et al. [26, Appendix A.2]), the weighted particle system* $(x_r^{N,i}, w_r^{N,i})_{i=1}^N$ *generated by* `dc_smc(r)` *is consistent in that for all non-negative, bounded, measurable functions* $f : X \to \mathbb{R}$*:*

$$\sum_{i=1}^{N} \frac{w_t^{N,i}}{\sum_{j=1}^{N} w_t^{N,j}} f\left(x_r^{N,i}\right) \quad \xrightarrow{\mathbb{P}} \int f(x)\gamma(x)\,\mathrm{d}x, \quad \text{as } N \to \infty. \tag{3.5}$$

These results exceed our desiderata from Section 2.3 which only requested consistency for well behaved test functions f. The additional theoretical result for the unbiasedness of the DCSMC estimate of the probability of the data $\hat{Z}$ allows for two downstream analyses. First, Proposition 1 provides a method to compare different models based on the Bayes factor [20], that is the ratio of the marginal likelihood of the two models under consideration:

$$\text{Bayes factor} \quad = \quad \frac{\hat{Z}_{\text{Model 1}}}{\hat{Z}_{\text{Model 2}}}. \tag{3.6}$$

Values of this ratio that are greater than one favour Model 1, whereas values that are less than one favour Model 2. Second, as mentioned in Lindsten et al. [26], the unbiased estimate can be used in particle Markov chain Monte Carlo (PMCMC)[2]. In a PMCMC setup DCSMC is embedded inside of a MCMC chain, so that inference can be performed on additional quantities

of interest. For example, inference could be performed on branch lengths, rate matrix parameters, and the topology via the full posterior $\gamma(\tau, b, \theta)$ displayed in Equation 2.9.

In addition to applications provided for Proposition 1, Proposition 2 can be used in debugging situations. Two immediate possibilities exist for utilizing this theoretical result in debugging: (i) comparing functions of samples (to real-values) from DCSMC and MCMC through frequentist hypothesis testing; (ii) extending the MCMC correctness tests of Geweke [16]. Possibility (i) is presented in Chapter 4.

3.3 Relaxing the Molecular Clock with DCSMC

As discussed in Section 2.1.2, the CPP is a parametric model to account for rate inhomogeneity across lineages. The following generative process can be used to describe a CPP on phylogenetic trees. Simulate locations of substitution rate change according to a PP with rate λ over the tree topology and branch lengths. The rate of substitution change m just before an event of rate change is scaled by a gamma random variable $r \sim \text{Gamma}(\alpha, \beta)$:

$$m' = r \cdot m, \tag{3.7}$$

where α is the shape parameter, and β is the scale parameter.[2]

To formalize this process, consider the notation of Huelsenbeck et al. [18]. Let the collection of events from a CPP be denoted $e = (\xi, \mathbf{z}, \mathbf{r})$, where ξ is the number of events, $\mathbf{z}$ is a vector of event locations, and $\mathbf{r}$ is a vector of scalings. If $\xi = 0$, then $\mathbf{z}$ and $\mathbf{r}$ are empty, otherwise if $\xi > 0$, then $\mathbf{z} \in [0, T]^{\xi}$ where T is the total tree length, and $r \in (\mathbb{R}^{+})^{\xi}$. Furthermore, the bijective map from $[0, T] \to (\tau, b)$ allows each point in $\mathbf{z}$ to be identified with a point on the tree. The prior on e given the tree topology, branch lengths, shape

[2]For technical reasons, see Huelsenbeck et al. [18], β is set to $\exp(\psi(\alpha))$ where:

$$\psi(\alpha) = \frac{\mathrm{d}}{\mathrm{d}\alpha} \log \Gamma(\alpha).$$

parameter α, and Poisson process rate λ is given as:

$$f(e|\tau, b, \lambda, \alpha) = \begin{cases} e^{-\lambda T} & \xi = 0 \\ \frac{e^{-\lambda T}(\lambda T)^{\xi}}{\xi!} \left(\frac{1}{T}\right)^{\xi} \prod_{i=1}^{\xi} g(r_i|\alpha) & \xi > 0 \end{cases}, \tag{3.8}$$

where $g(\cdot|\alpha)$ is the gamma density with shape parameter α and scale parameter β set as above. It has been shown in Rannala [33] that the rate of change and rate multiplier parameters are non-identifiable; nonetheless, inference is possible for the mean substitution rate of a lineage.

This CPP construction achieves differing evolutionary rates across lineages and its functional form makes it an excellent candidate for DCSMC algorithms. In particular, the proposals shown in Figure 3.4 are easy to adjust for the CPP. Figure 3.5 demonstrates that the earlier proposals can be adjusted to include a PP for locations of rate substitution change over the single proposed branch and gamma random variables to scale the rate. The joint prior is calculated as:

$$\pi(b, e) = \pi(e|b)\pi(b), \tag{3.9}$$

and intermediate distributions can be constructed in a similar way as in the molecular clock example.

Listing 3.5 illustrates the implementations of `PhyloParticle` for relaxed clock models. The main difference between molecular clock particles and relaxed clock particles is the extra field to store the scaling factors, and a new implementation of the evolutionary distance between parents and descendants. This extra information is sufficient to describe the relaxed clock model and allows for evolutionary distances to depend directly on the scaled branches and not the original height differences between parent and descendent. Also note that since the CPP employs a homogeneous PP, the locations of the events of substitution rate change need not be recorded. The prior density only depends on the number of events and their corresponding values.

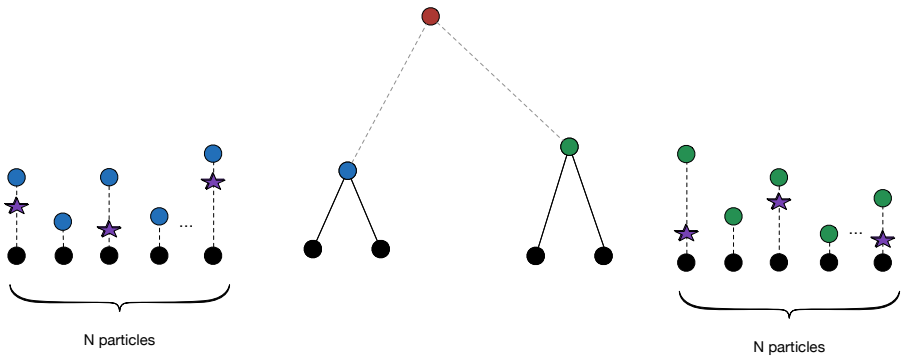


Figure 3.5: DCSMC for the Compound Poisson Process: Changes to the simple phylogenetic DCSMC algorithm to accommodate the CPP are limited to adding a Poisson point process with gamma random variable scalings to the proposal. Purple stars indicate realizations of the PP over the proposed branch lengths for each particle population.

```
public class RelaxedParticle implements PhyloParticle<RelaxedParticle>{
  // Properties
  public final ClockParticle baseParticle;
  public final Map<List<Double>,List<Double>> scalingFactors;

  // Methods
  public double getHeight(){
    /**
    * Return first element of baseParticle's sortedHeights
    */
  }
  public RelaxedParticle
    create(RelaxedParticle particle1, RelaxedParticle particle2,
           HeightScaling heightScaling){
    /**
    * Create a new baseParticle via baseParticle's create method
    * scalingFactors = particle1.scalingFactors ∪
    *                  particle2.scalingFactors ∪ heightScaling.scalings
    */
  }
  public double
    getEvolutionaryDistance(RelaxedParticle parent, RelaxedParticle descendant){
    /**
    * Determine path between parent & descendant (via baseParticle children) p
    * Return ∑_p ∑_i scalingFactors.get(p).get(i)
    */
  }
}
```

Listing 3.5: Relaxed Clock Particle

3.4 Extensions

As discussed in Section 2.2, the CPP is one of many techniques used to relax the molecular clock. The DCSMC framework can also be utilized in approaches where the rate is modelled as a Brownian motion. In particular, DCSMC only requires efficient simulation from a proposal distribution and point-wise evaluation of the prior density over perturbed branch lengths. For example, in the CPP, the piece-wise constant rate can be viewed as a special case of a Brownian motion. Future work will explicitly integrate these techniques into the DCSMC framework.

We also take this opportunity to build on the earlier efficiency discussion of this chapter by introducing a specific example for relaxed clock models. Consider the height inference problem for relaxed clock models when the tree topology is unknown. Inference proceeds under MCMC methods by sampling the full state space (τ, b, relaxed clock parameters) via a collection of MCMC kernels. Conversely, in a PMCMC setup DCSMC is embedded into a MCMC chain where the MCMC chain samples only the topology τ and the branch lengths and relaxed clock parameters are inferred inside of DCSMC routines.

In MCMC implementations topological moves may require proposing several relaxed clock parameters for each branch, whereas the PMCMC implementation provides the flexibility to initialize a particle population from scratch for each newly proposed topology. Figure 3.6 demonstrates a MCMC move on tree topologies for relaxed clock models. Branch colours represent different values of the evolutionary rate. As illustrated in this figure, relaxed clock models may require more complicated MCMC proposals to accommodate the Brownian motion governing the evolutionary rate.

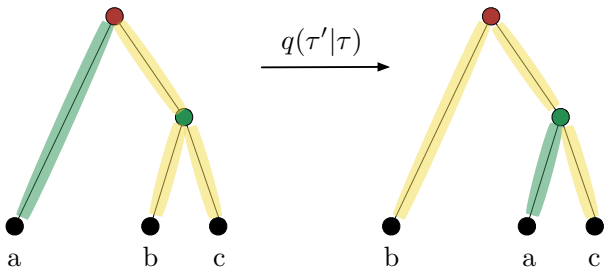


Figure 3.6: MCMC Topology Move for Relaxed Clock Models: (Left) Current state space of MCMC sampler. (Right) Possible proposal from a kernel restricted to topological moves. Branch colours represent a stochastic process for the rate.

Chapter 4

Experiments

This section covers several experiments that (i) show the correctness of the DCSMC implementation, and (ii) demonstrate areas where we predict DCSMC will outperform traditional approaches. Implementation correctness is assessed via several unit tests of the peeling recursions and numerical marginal likelihood calculations. The validity of the DCSMC method in phylogenetics is compared versus traditional MCMC methods. These comparisons are focused on two main metrics: marginal likelihood estimation, and estimated tree height $\hat{h}$. For each metric, we aim to show comparable performance; future work is planned to show comparable performance with significantly reduced computational budgets on large trees. The baseline to which all marginal likelihood comparisons are made against is the phylogenetic-specialized MCMC software called Mr. Bayes [19].

Correctness Tests: The DCSMC implementation can be tested with a variety of different unit tests. Two simple tests are described below. The first test is to ensure the correctness of the peeling-recursions. Since the likelihood function has the property that,

$$\sum_{\mathcal{Y}(X)} l(\mathcal{Y}(X)|b) = 1, \tag{4.1}$$

the peeling-recursion, which calculates the likelihood, can be explicitly checked by enumerating the support of $\mathcal{Y}(X)$. Figure 4.1 depicts a full enumeration

of leaf observations for one character DNA sequences in a two taxa tree. Since there are two leaves and four possible characters this space has 16 ($= 4^2$) elements whose likelihoods must sum to one.

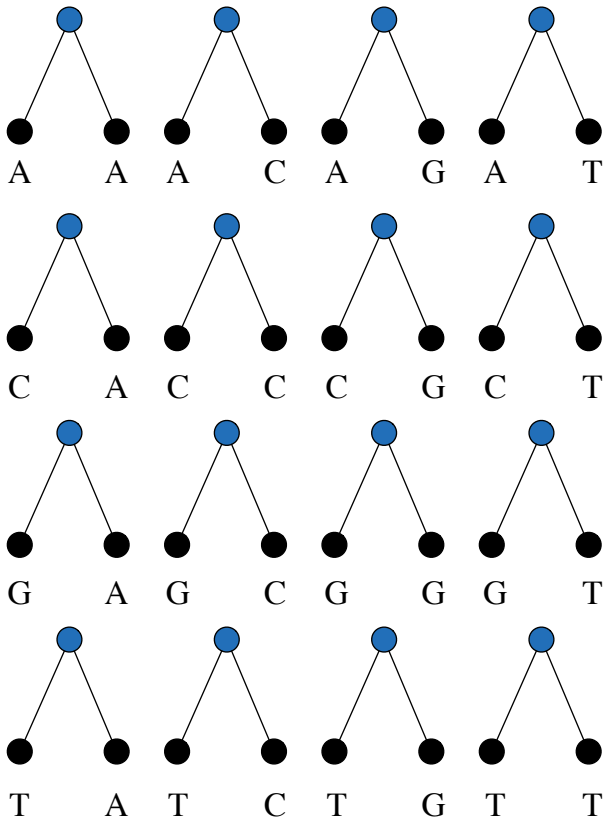


Figure 4.1: Correctness Unit Test: Enumerate all possible data sequences, determine the likelihood of each, and ensure that the sum of likelihoods is unity.

The second analytical correctness test is to compare the Z estimate from DCSMC to numerical methods. Although intractable for any tree of practical interest, for small enough trees and bounded support, the integral:

$$Z = \int l(\mathcal{Y}(X)|b)\pi(\,\mathrm{d}b),$$

can be numerically calculated via multivariate trapezoidal integration. For instance, suppose $M = 3$ and the prior over the branch lengths (b_1, b_2) is $\text{Unif}(0,1) \times \text{Unif}(0,1)$. Then the integral is simply:

$$Z = \int_0^1 \int_0^1 l(\mathcal{Y}(X)|b_1, b_2)\, \mathrm{d}b_1\, \mathrm{d}b_2,$$

which can be calculated via evaluating $l(\mathcal{Y}(X)|b_1, b_2)$ at points on the unit square and employing trapezoidal integration iteratively.

The implementation of the DCSMC algorithm for phylogenetics passes both peeling recursion and marginal likelihood tests.

Marginal Likelihood: The marginal likelihood estimate is returned directly from the DCSMC algorithm; however, its estimation from other Monte Carlo frameworks like MCMC is not straightforward. Common methods of estimating this quantity from MCMC use different importance sampling distributions to create estimators of $\hat{Z}$. One such estimator is the harmonic mean introduced by Newton and Raftery [29] via a particular choice of importance distribution. While Mr. Bayes [19] reports the harmonic mean by default, it has been shown to be inefficient since it often has infinite variance.

In light of this limitation, other statistically efficient methods adapted from bridge sampling and path sampling of Gelman and Meng [15] have been introduced into Bayesian phylogenetics. A computationally intensive alternative, thermodynamic integration, was developed first in Friel and Pettitt [14], Lartillot and Philippe [23]. Subsequently, a more computationally feasible method to estimate the marginal likelihood, called the stepping stone method, was developed by Xie et al. [42]. The experiments presented in this paper compare the marginal likelihood estimate of DCSMC versus the estimate provided by a stepping stone method implemented in Mr. Bayes [19].

Height estimates $\hat{h}$: Within a Bayesian phylogenetic setup, parameter estimation or estimation of functions of parameters can be cast as minimizing the posterior expected loss $\mathbb{E}\left(\mathcal{L}(\bullet, \delta)|\mathcal{Y}(X)\right)$ given the observed data and with respect to the posterior. The loss $\mathcal{L}(\bullet, \delta)$ encapsulates the error associated with evaluating $\bullet$ by δ. The loss function provides a principled way

to summarize trees and to estimate the tree height. Under the quadratic loss function, $\mathcal{L}(\mathbf{v}, \delta) = ||\mathbf{v} - \delta||$ for $\mathbf{v}, \delta \in \mathbb{R}^{\mathbf{P}}$, the Bayes estimator can be shown to be the posterior mean, see Robert [35, Chapter 2.5]. As a result, trees are summarized by the posterior mean of b; the height of the tree h is similarly summarized as the posterior mean of $h(r)$.

The consistency result of Proposition 2 provides a method to check the correctness of the algorithm. For instance, the weighted average of the particle heights from a DCSMC population tend in probability to the Bayes estimator of tree height for the posterior distribution as the number of particles approach infinity. A similar consistency result holds for samples generated from MCMC.

There are thus two different correctness tests to evaluate an implementation of the phylogenetic DCSMC. In a simulated setting, where the true height is known, the DCSMC and MCMC estimates can be directly compared to the true known height (which is not equal to the synthetic held-out height in general). Additionally, given a correct MCMC implementation such as Mr. Bayes, the DCSMC estimates can be compared to those of MCMC using frequentist hypothesis testing techniques.

In all instances that follow, Mr. Bayes is executed with one run (without tempering) with a varying number of MCMC iterations. The model is specified to match the description in this thesis (Jukes-Cantor rate matrix with Yule process prior)[1]. The tree topology is fixed for all MCMC iterations to a user-supplied tree that corresponds to the fixed tree used in DCSMC.

The remainder of this section covers synthetic experiments that validate the DCSMC implementation, and a real world example on green plant DNA sequences. It concludes with a note on future directions for this work.

[1]There is a technical difference between the Yule prior described herein and the implementation in Mr. Bayes. A recompiled version of Mr. Bayes can be used to correct for this discrepancy; however, for simplicity of exposition of the parameter settings this is not discussed outside of this footnote.

```
#nexus
begin mrbayes;
  set autoclose = yes nowarn = yes;
  set seed = 1;
  set swapseed = 1;
  execute tree.nexus;
  lset nst = 1;
  prset brlenspr = clock:birthdeath;
  prset speciationpr = fixed(1);
  prset extinctionpr = fixed(0);
  prset statefreqpr = fixed(equal);
  prset topologypr = fixed(true);
  mcmcp nruns = 1;
  mcmcp nchains = 1;
  // Height reconstructions:
  mcmc ngen = 1000000;
  // Log(Z) estimate:
  ss ngen = 1000000 Alpha = 0.3 NSteps = 10 Burninss = -1;
  quit;
end;
```

Listing 4.1: Mr. Bayes Parameter Specification

4.1 Synthetic Results

The first synthetic result validates the implementation of DCSMC by comparing the log marginal estimate $\log(\hat{Z}_{DCSMC})$ to the log marginal estimate $\log(\hat{Z}_{MCMC})$ from Mr. Bayes, and the estimated height from DCSMC versus the true known tree. The true synthetic tree is used to initialize Mr. Bayes' MCMC chain for the log marginal estimate. This helps ensure that the Markov chain explores the posterior distribution near the true value.

The setup consisted of simulating in silico three rooted ultrametric trees containing 3, 5, 10 species with 100 character DNA sequences. The in silico trees are produced via forward sampling: (1) simulate the tree topology and speciation times via the Yule process prior; (2) simulate the stochastic process over DNA characters from the root to the leaves for each site. At the root, DNA characters are drawn from the stationary distribution π of Q.

The marginal log likelihood for each tree is estimated with any increasing number of particles from 2 to 10,000 particles via DCSMC and 1,000,000 MCMC iterations. This process is repeated for 100 Monte Carlo replicates, and performance is measured by aggregating across replicates. In total, this represents 3 taxa $\times$ 6 particle configurations $\times$ 100 replicates $= 1,800$ experiments.

Figure 4.2 depicts the results of this validation experiment. In panel (a) the percent difference between DCSMC and Mr. Bayes is plotted for each of the three different simulated trees and for an increasing number of particles. Panel (b) tells a similar story for the height estimate provided by DCSMC versus the true known tree. (In Section 4.3 height reconstructions from both Mr. Bayes and DCSMC are compared to the true synthetic tree.)

In Figure 4.2 the differences between the $\log(\hat{Z})$ estimate from DCSMC and Mr. Bayes, and the DCSMC reconstructed height versus the known true tree height, increase as the number of leaves increase. One explanation of this experimental behaviour is that the unbiasedness and consistency of the DCSMC algorithm are only asymptotic guarantees, yet these experiments use a finite number of particles. Below, we attempt to develop a more satisfying resolution to the performance of the algorithm under finite particle populations.

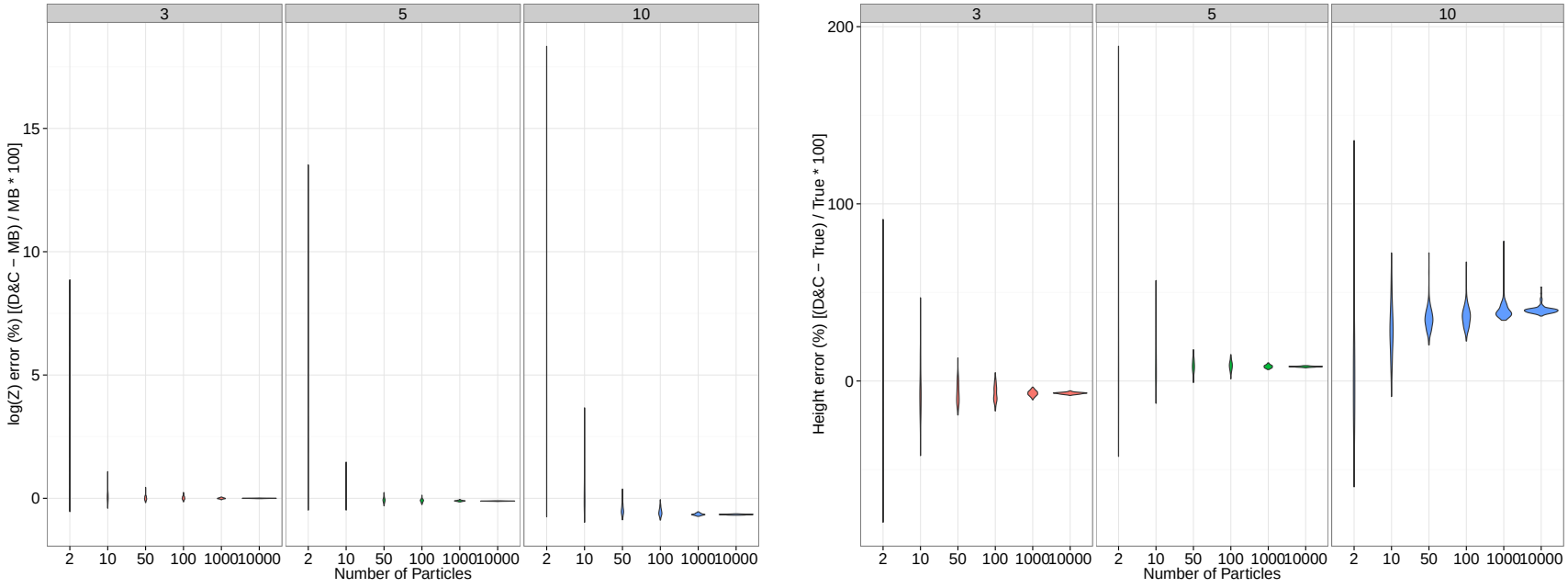
We endeavour to characterize the discrepancy between the two log evidence estimates. Since the MCMC runs of Mr. Bayes are initialized to the true synthetic tree we believe that Mr. Bayes is not stuck in a local maximum and that the value provided for $\log(\hat{Z}_{MCMC})$ represents Mr. Bayes best estimate. To supplement Mr. Bayes' estimate another $\log(\hat{Z}_{MCMC})$ estimate is planned via the phylogenetic software package Beast [10]. Since DCSMC cannot be initialized to the true tree (it explores the sample space by generating samples from a proposal as opposed to the random MCMC walks) this additional comparison could provide insight into the discrepancy between the original estimates.

A complementary explanation is that the proposals used in DCSMC are not "good enough". As discussed earlier, the quality of the approximation to the posterior distribution is dependent on the proposal distribution. To

assess and correct proposal quality the effective sample size (ESS) at each node can be evaluated:

$$ESS_v = \frac{1}{\sum_{i=1}^{N}(w_v^i)^2}. \tag{4.2}$$

The ESS is a measure of the particle degeneracy within a population of particles. If all particles are equally-weighted the ESS is maximized as N; if all particles except one have weight 0 the ESS is minimized to 1. Larger ESS values imply less particle degeneracy and higher quality approximations. Low ESS values are symptoms of poor proposal distributions and can often be alleviated by slowly annealing the intermediate likelihoods. Future work is aimed at adding an annealing step to address these potential issues. See Lindsten et al. [26] for a discussion of annealing in DCSMC methods.



(a) (b)

Figure 4.2: Synthetic Validation Comparisons: Synthetic Trees Validation for 100 Monte Carlo replicates of DCSMC over different number of taxa and differing N particles. (a) $\log(\hat{Z})$ comparisons versus Mr.Bayes. (b) $\hat{h}$ comparisons versus the true synthetic tree.

4.2 Green Plant rbcL

In addition to the in silico experiments, the DCSMC methodology is also evaluated on a published real dataset. This dataset consists of 10 taxa from the green plant family. Specifically, DNA sequences are from the chloroplast-encoded large subunit of the RuBisCO gene (rbcL) for different green plants. This gene is involved in an enzyme used when plants convert carbon dioxide to glucose (i.e., photosynthesis). Understanding RuBisCO's evolutionary relationships could be of interest in areas such as its role in global climate change, see for example Sage et al. [36]. More information on this dataset can be found in Xie et al. [42].

Similarly to the synthetic experiments, the $\log(\hat{Z})$ estimates of Mr.Bayes and DCSMC are compared for the RuBisCO dataset. However, in contrast to the synthetic datasets, the true tree topology is unknown for the RuBisCO data. Hence, we have assumed that the tree topology is fixed to the one presented in Figure 4.3. To check that both DCSMC and Mr. Bayes estimates of $\log(\hat{Z})$ are similar 50 Monte Carlo replicates of DCSMC with $1,000$ particles and $500,000$ iterations of Mr.Bayes are generated. Each Monte Carlo replicate differs due to the differing initial random seeds.

Table 4.1 tabulates 95% confidence intervals for these Monte Carlo replicates. In addition to the fact that these confidence intervals overlap, frequentist tests, such as the t-Test, for a difference in means between DCSMC and Mr. Bayes are unable to reject the null hypothesis that the true difference between the means is zero. In conclusion, the real dataset provides evidence that the DCSMC method is valid, in the sense that its estimates of the marginal likelihood are not significantly different than Mr. Bayes' estimates (Mr. Bayes is viewed as the gold standard).

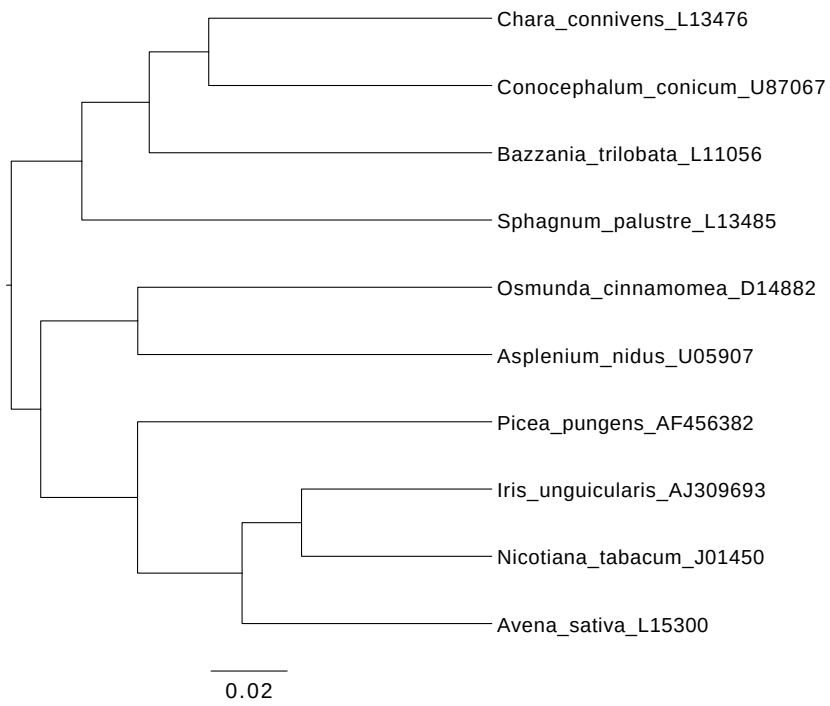


Figure 4.3: Green Plant Phylogeny: 10-taxon green plant data set with DNA sequences of chloroplast-encoded large subunit of RuBisCO gene (rbcL) [42].

Table 4.1: Green Plant Comparisons: 95% confidence intervals from 50 Monte Carlo replicates of the DCSMC and MCMC algorithms initialized with different random seeds.

	DCSMC $1,000$ particles	Mr. Bayes $500,000$ iterations
$\log(\hat{Z})$	$(-7298, -7295)$	$(-7297.6, -7297.1)$

4.3 Future Work

There is a rich set of problems and application areas for this framework to be applied to. In this section, we outline future experimental work on two main areas. The first set of planned experiments is testing the scaleability of this framework on large trees (i.e., $M > 1000$). Scalability experiments are expected to show that DCSMC requires smaller computational budgets than Mr. Bayes for posterior estimation.

To illustrate the idea behind this experiment, we present a prototype on a small synthetic dataset. This prototype compares height estimates provided by DCSMC, Mr. Bayes, and Mr. Bayes initialized from DCSMC, to the true height of the tree. Each of these three inference strategies are given a pre-specified computational budget; number of iterations or number of particles. Runtime is used to standardize across different computational budget metrics. Ten Monte Carlo replicates are simulated for each budget and strategy.

This prototype is illustrated in Figure 4.4 for a small ten taxa tree. The left panel of this figure shows the synthetic tree with true height 1.588. The right panel shows the reconstructed height, given as the Bayes estimator described earlier, for each of the three different strategies. The true height of the tree is listed as a black horizontal line. The estimates provided by all three methods are generally in agreement. The key domain to showcase DCSMC performance is on large trees. In these cases, we expect DCSMC to converge to the correct true tree height with smaller wall-times than MCMC based methods.

The second set of planned experiments is to evaluate the performance of the DCSMC framework on the relaxed clock models. As discussed in Section 3.3, the molecular clock models are easy to extend to the CPP relaxed clock models.

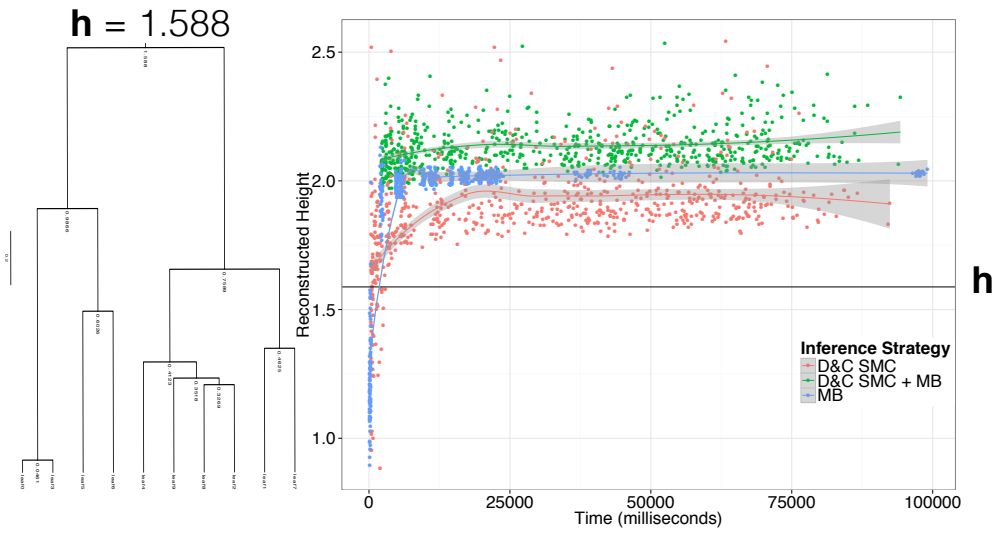


Figure 4.4: Synthetic Tree Runtime Comparisons: Reconstructed $\hat{h}$ from DCSMC, DCSMC+Mr. Bayes, and Mr. Bayes versus computational budget. True synthetic tree height denoted as black horizontal line as **h**.

Chapter 5

Conclusion

This thesis introduced a specialized DCSMC method for Bayesian phylogenetic inference. In contrast to traditional particle filtering methods, DCSMC uses a tree based decomposition of the probabilistic model. Tree based decompositions can be applied naturally to models, as in phylogenetics, that directly use a tree structure. In this thesis, two phylogenetic specializations of the DCSMC algorithm are presented. The first method addresses the standard molecular clock models, while the second specialization attends to a class of relaxed class models constructed through a CPP. Experimental results are presented to validate the correctness of the implementation on both synthetic and real data. Directions for future work are suggested to focus on demonstrating the computational performance of both large trees and relaxed models.

Bibliography

[1] Ö. Åkerborg, B. Sennblad, and J. Lagergren. Birth-Death Prior on Phylogeny and Speed Dating. *BMC Evolutionary Biology*, 8(1):77, 2008. → page(s) 2

[2] C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov Chain Monte Carlo Methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010. → page(s) 31

[3] S. Aris-Brosou and Z. Yang. Effects of Models of Rate Evolution on Estimation of Divergence Dates with Special Reference to the Metazoan 18S Ribosomal RNA Phylogeny. *Systematic Biology*, 51(5): 703–714, 2002. → page(s) 8

[4] A. Bouchard-Côté, S. Sankararaman, and M. I. Jordan. Phylogenetic Inference via Sequential Monte Carlo. *Systematic Biology*, 61:579–593, 2012. → page(s) 1, 2, 17

[5] M.-H. Chen, L. Kuo, and P. O. Lewis. *Bayesian Phylogenetics: Methods, Algorithms, and Applications*. CRC Press, 2014. → page(s) 1, 11

[6] P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo Samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006. → page(s) 17

[7] A. Doucet and A. M. Johansen. A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later. *Handbook of Nonlinear Filtering*, 12: 656–704, 2009. → page(s) 13, 14, 15, 16

[8] A. J. Drummond, G. K. Nicholls, A. G. Rodrigo, and W. Solomon. Estimating Mutation Parameters, Population History and Genealogy Simultaneously From Temporally Spaced Sequence Data. *Genetics*, 161(3):1307–1320, 2002. → page(s) 1

[9] A. J. Drummond, S. Y. Ho, M. J. Phillips, A. Rambaut, et al. Relaxed Phylogenetics and Dating with Confidence. *PLoS Biology*, 4 (5):699, 2006. → page(s) 8

[10] A. J. Drummond, M. A. Suchard, D. Xie, and A. Rambaut. Bayesian phylogenetics with BEAUti and the BEAST 1.7. *Molecular Biology and Evolution*, 29(8):1969–1973, 2012. → page(s) 43

[11] J. Felsenstein. Maximum Likelihood and Minimum-Steps Methods for Estimating Evolutionary Trees from Data on Discrete Characters. *Systematic Biology*, 22(3):240–249, 1973. → page(s) 4, 11, 21

[12] J. Felsenstein. Evolutionary Trees from DNA sequences: A Maximum Likelihood Approach. *Journal of molecular evolution*, 17(6):368–376, 1981. → page(s) 4, 11, 21

[13] J. Felsenstein. *Inferring Phylogenies*. Palgrave Macmillan, 2004. ISBN 9780878931774. → page(s) 1

[14] N. Friel and A. N. Pettitt. Marginal Likelihood Estimation via Power Posteriors. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(3):589–607, 2008. → page(s) 40

[15] A. Gelman and X.-L. Meng. Simulating Normalizing Constants: From Importance Sampling to Bridge Sampling to Path Sampling. *Statistical Science*, pages 163–185, 1998. → page(s) 40

[16] J. Geweke. Getting It Right: Joint Distribution Tests of Posterior Simulators. *Journal of the American Statistical Association*, 99(467): 799–804, 2004. → page(s) 32

[17] J. H. Gillespie. *The Causes of Molecular Evolution*. Oxford University Press, 1991. → page(s) 7

[18] J. P. Huelsenbeck, B. Larget, and D. Swofford. A Compound Poisson Process for Relaxing the Molecular Clock. *Genetics*, 154(4): 1879–1892, 2000. → page(s) 8, 9, 11, 13, 32

[19] J. P. Huelsenbeck, F. Ronquist, et al. MRBAYES: Bayesian Inference of Phylogenetic Trees. *Bioinformatics*, 17(8):754–755, 2001. → page(s) 1, 38, 40

[20] R. E. Kass and A. E. Raftery. Bayes Factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995. → page(s) 31

[21] H. Kishino, J. L. Thorne, and W. J. Bruno. Performance of a Divergence Time Estimation Method Under a Probabilistic Model of Rate Evolution. *Molecular Biology and Evolution*, 18(3):352–361, 2001. → page(s) 8

[22] B. Larget and D. L. Simon. Markov Chain Monte Carlo Algorithms for the Bayesian Analysis of Phylogenetic Trees. *Molecular Biology and Evolution*, 16:750–759, 1999. → page(s) 1

[23] N. Lartillot and H. Philippe. Computing Bayes Factors using Thermodynamic Integration. *Systematic Biology*, 55(2):195–207, 2006. → page(s) 40

[24] T. Lepage, D. Bryant, H. Philippe, and N. Lartillot. A General Comparison of Relaxed Molecular Clock Models. *Molecular Biology and Evolution*, 24(12):2669–2680, 2007. → page(s) 8

[25] S. Li, D. K. Pearl, and H. Doss. Phylogenetic Tree Construction using Markov Chain Monte Carlo. *Journal of the American Statistical Association*, 95(450):493–508, 2000. → page(s) 1

[26] F. Lindsten, A. M. Johansen, C. A. Naesseth, B. Kirkpatrick, T. B. Schön, J. Aston, and A. Bouchard-Côté. Divide-and-Conquer with Sequential Monte Carlo. *ArXiv e-prints*, June 2014. → page(s) 2, 20, 29, 31, 44

[27] B. Mau and M. A. Newton. Phylogenetic Inference for Binary Data on Dendograms Using Markov chain Monte Carlo. *Journal of Computational and Graphical Statistics*, 6(1):122–131, 1997. → page(s) 1

[28] C. Moler and C. Van Loan. Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later. *SIAM Review*, 45 (1):3–49, 2003. → page(s) 6

[29] M. A. Newton and A. E. Raftery. Approximate Bayesian Inference with the Weighted Likelihood Bootstrap. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 3–48, 1994. → page(s) 40

[30] M. A. Newton, B. Mau, and B. Larget. Markov Chain Monte Carlo for the Bayesian Analysis of Evolutionary Trees from Aligned Molecular Sequences. *Lecture Notes-Monograph Series*, pages 143–162, 1999. → page(s) 1

[31] J. G. Propp and D. B. Wilson. Exact Sampling with Coupled Markov Chains and Applications to Statistical Mechanics. *Random Structures and Algorithms*, 9(1-2):223–252, 1996. → page(s) 13

[32] J. G. Propp and D. B. Wilson. How to Get a Perfectly Random Sample from a Generic Markov Chain and Generate a Random Spanning Tree of a Directed Graph. *Journal of Algorithms*, 27(2): 170–217, 1998. → page(s) 13

[33] B. Rannala. Identifiability of Parameters in MCMC Bayesian Inference of Phylogeny. *Systematic Biology*, 51(5):754–760, 2002. → page(s) 33

[34] B. Rannala and Z. Yang. Probability Distribution of Molecular Evolutionary Trees: A New Method of Phylogenetic Inference. *Journal of Molecular Evolution*, 43(3):304–311, 1996. → page(s) 1

[35] C. Robert. *The Bayesian Choice: From Decision-Theoretic Foundations To Computational implementation*. Springer Science & Business Media, 2007. → page(s) 41

[36] R. F. Sage, D. A. Way, and D. S. Kubien. Rubisco, Rubisco Activase, and Global Climate Change. *Journal of Experimental Botany*, 59(7): 1581–1595, 2008. → page(s) 46

[37] M. J. Sanderson. A Nonparametric Approach to Estimating Divergence Times in the Absence of Rate Constancy. *Molecular Biology and Evolution*, 14(12):1218–1231, 1997. → page(s) 8

[38] M. J. Sanderson. Estimating Absolute Rates of Molecular Evolution and Divergence Times: A Penalized Likelihood Approach. *Molecular Biology and Evolution*, 19(1):101–109, 2002. → page(s) 8

[39] S. Tavaré. Some Probabilistic and Statistical Problems in the Analysis of DNA Sequences. *Lectures on Mathematics in the Life Sciences*, 17: 57–86, 1986. → page(s) 6

[40] Y. Teh, H. Daumé III, and D. Roy. Bayesian Agglomerative Clustering with Coalescents. In *Advances in Neural Information Processing Systems 20-Proceedings of the 2007 Conference*, 2009. → page(s) 2, 17

[41] J. L. Thorne, H. Kishino, and I. S. Painter. Estimating the Rate of Evolution of The Rate of Molecular Evolution. *Molecular Biology and Evolution*, 15(12):1647–1657, 1998. → page(s) 8

[42] W. Xie, P. O. Lewis, Y. Fan, L. Kuo, and M.-H. Chen. Improving Marginal Likelihood Estimation for Bayesian Phylogenetic Model Selection. *Systematic Biology*, page syq085, 2010. → page(s) 40, 46, 47

[43] Z. Yang and B. Rannala. Bayesian Phylogenetic Inference Using DNA sequences: A Markov Chain Monte Carlo Method. *Molecular Biology and Evolution*, 14(7):717–724, 1997. → page(s) 1

[44] A. D. Yoder and Z. Yang. Estimation of Primate Speciation Dates Using Local Molecular Clocks. *Molecular Biology and Evolution*, 17 (7):1081–1090, 2000. → page(s) 8

[45] E. Zuckerkandl and L. Pauling. Molecular Disease, Evolution and Genetic Heterogeneity. 1962. → page(s) 7