VECTORIZED MARKOV CHAIN MONTE CARLO PARAMETER ESTIMATION FOR THE MULTISPECIES COALESCENT MODEL

An Xuelong
Dept. of Computer Science
University College London
ucabxan@ucl.ac.uk

ABSTRACT

The multispecies coalescent (MSC) model provides a statistical framework for estimating evolutionary parameters while accounting for incomplete lineage sorting. However, exact likelihood-based approaches are computationally intractable for large genomic datasets. We optimize an existing Markov Chain Monte Carlo (MCMC) algorithm through vectorization to efficiently estimate key MSC parameters—species divergence time (τ) and population size (θ)—from multi-locus sequence alignments of humans and chimpanzees. By vectorizing the posterior computation and acceptance-rejection steps, we achieve substantial computational speedups, completing 20000 MCMC iterations for ≈ 14000 loci in under 1 minute. Our results demonstrate stable convergence and efficient sampling, with an acceptance rate of 43% and an effective sample size of ≈ 1500. We further explore the impact of window size selection on MCMC efficiency and highlight the model's limitations due to data constraints. This work contributes a practical and scalable approach for parameter estimation in comparative genomics, where we open-source our implementation at `https://github.com/awxlong/machine_learning_projects_ucl/blob/main/msc_mcmc/mcmc-msc-vectorized-submit.ipynb`

***K*eywords** Vectorization · MCMC · Metropolis-Hastings

1 Introduction

In comparative genomics, the multispecies coalescent (MSC) model is a powerful statistical framework proposed to address conflicting genealogical histories by accounting for incomplete lineage sorting (ILS) owed to polymorphism in ancestral species [1, 2]. Its most important parameters are τ and θ, which represent the species divergence time and population size, respectively. Both are assumed to be constant across the genome. We can resort to exact likelihood methods such as Bayesian inference to estimate them and accommodating their uncertainties accordingly. However, such approaches are often intractable owed to the high dimensionality of the available genomic data. Thus, in this work, we resort to an approximate method by a Monte-Carlo Markov-Chain (MCMC) algorithm for tractably estimating the parameters of the MSC model [1, 3, 4] to help us understand the intermingled evolutionary history of humans and chimpanzees.

2 Methodology

We collect genomic, multi-locus data consisting of sequence alignments of humans and chimpanzees $X = \{(x_1, n_1), \ldots (x_L, n_L)\}$ for $L = 1000$ loci where for locus i we observe x_i differences at n_i sites [see 5]. These loci are loosely linked short genomic segments, such that we ignore recombination and treat each locus as independent. The posterior distribution f of the parameters of interest is, according to Bayes Theorem:$f\left(\tau,\ \theta,\ \{t_i\} \mid X\right) = \frac{1}{Z} f\left(X|\tau,\ \theta, \{t_i\}\right) f(\tau) f(\theta)$, where $f\left(X|\tau,\ \theta, \{t_i\}\right) = \prod_{\{i=1\}}^{1000} \binom{n_i}{x_i} p_i^{x_i} \left(1-p_i\right)^{n_i - x_i}$ is the binomial likelihood of observing all differences at all sites, where $p_i = \frac{3}{4} - \frac{3}{4} e^{-\frac{8}{3}(\tau + t_i)}$ is the probability of observing a difference at any site at locus i and t_i is an exponential variable measuring the coalescent time at locus i with density $f\left(t_i|\tau, \theta\right) = \frac{2}{\theta} e^{-\frac{2}{\theta} t_i}$.

The terms $f(\tau) = \frac{1}{\mu_\tau} e^{-\frac{1}{\mu_\tau}\tau}$, $f(\theta) = \frac{1}{\mu_\theta} e^{-\frac{1}{\mu_\theta}\theta}$ are exponential priors with $\mu_\tau = 0.005$, $\mu_\theta = 0.001$. $Z = \int f(X|\tau,\ \theta, \{t_i\})\, f(\tau) f(\theta) d\tau, \theta$ is known as the *normalization constant*, and it is the main computational bottleneck for obtaining the posterior. Because it's a multidimensional integral, can't be derived analytically and expensive to tractably compute numerically, we resort to the classical Metropolis-Hasting algorithm [5] to draw probabilistic samples of $f(\tau, \theta,\ \{t_i\}|X)$[1] whilst bypassing the normalization constant by canceling it out through the log ratio of the unnormalized posterior:

$$\log f\,(\tau,\ \theta, \{t_i\}|X) = C - \frac{1}{\mu_t}\tau - \frac{1}{\mu_\theta}\theta + \sum_{i=1}^{1000} [log\frac{2}{\theta} - \frac{2}{\theta}t_i + x_i \log p_i + (n_i - x_i)\log(1 - p_i)] \tag{1}$$

Equation 1: log unnormalized posterior (later simplified as $\pi(\phi)$), where C is a constant absorbing terms we ignore in the MCMC algorithm. Taking the log helps avoid numerical over/underflow during MCMC.

This MCMC algorithm, adapted for our purposes, is as follows:

1. Initialize parameters $\Phi = \{\tau = 0.01,\ \theta = 0.001,\ t_{1:1000} = 0.001\}$ and respective window sizes w_ϕ[2] (see Table 1 for candidate window sizes)
2. For each MCMC iteration, and for each parameter ϕ in Φ :
   a. Sample ϕ^* from a uniform proposal distribution $U\left(\phi - \frac{w_\phi}{2}, \phi + \frac{w_\phi}{2}\right)$, where if $\phi < 0,\ \phi = -\phi$
   b. Compute $\alpha = min(1, \frac{\pi(\phi^*)}{\pi(\phi)} \times \frac{U(\phi|\phi^*)}{U(\phi^*|\phi)})$, where $\pi(\phi^*)$ is the unnormalized posterior (Equation 1), thus the ratio effectively cancels out the normalization constant. The log ratio $\frac{\pi(\phi^*)}{\pi(\phi)}$ is computed by keeping the remaining parameters in Φ fixed to the previous state, thus treating this multidimensional MCMC into separate unidimensional updates. As such, we adjust window sizes for an acceptance rate of 43% for each ϕ.
   c. Accept the proposed sample if $u < \alpha,\ u \sim U(0,\ 1)$, otherwise reject.

We optimize runtime of the algorithm as follows: 1) we vectorize the computation of $\pi(\phi)$ for summing out the coalescent times, 2) we also vectorize the proposal and subsequent acceptance/rejection of the 1000 coalescent times, noting that their logratios are directly calculated as $-\frac{2}{\theta}\ (\mathbf{t}^* - \mathbf{t}\ ) + \mathbf{x} * \log\left(\frac{\mathbf{p}^*}{\mathbf{p}}\right) + (\mathbf{n} - \mathbf{x}) \log\frac{1-\mathbf{p}^*}{1-\mathbf{p}}$ where in bold we highlight the vectorization and 3) we cache the current $\pi\ (\phi^*)$ to avoid duplicate computations for α. Altogether, our MCMC implementation finishes under ~ 30 seconds for $L = 1000$ loci and 20000 iterations.

Step 2b) is done by exploring over a grid of window sizes for τ and θ, with τ : [5.3e-06, 0.001, 0.053] and θ: [9.e-06, 0.001, 0.090], to study how it influences acceptance rate, efficiency and final posterior sample values.

3 Results

The results of our experiments are recorded in Table 1, where we achieve an acceptance rate of 43% for both τ and θ through the same window size of 0.001. For τ, we obtain a posterior mean of 0.004012 with $2.5\%,\ 97.5\%$ credibility interval of $[0.003728,\ 0.004294]$. For θ, we obtain a posterior mean of 0.003984 and credibility interval of $[0.003416,\ 0.004609]$. We further run a more comprehensive set of experiments with more varying window sizes at Table 2.

In Figure 1, the trace plots of the posterior samples after a burn-in of 5000 also show a stable convergence for both parameters. Both parameters have an efficiency measured via a ratio of the variance based on the independent sample to the variance based on the MCMC sample of ~ 0.075, yielding a reasonable[3] effective sample size (ESS) of around $20000 * .075 \cong 1500$. This means a sample of size 20000 from the MCMC is as good (in terms of variance) as an independent sample of size 1500.

Our experiments also shed insight how varying window sizes (at the log-scale) deeply influences the quality of the samples in the MCMC (see Appendix-Figure 2 for an example of an inefficient MCMC). Both very high and low window sizes lead to strongly autocorrelated, inefficient MCMCs. Although we treat parameters as separate unidimensional candidates, a bad window size for one parameter may sometimes negatively impact the sampling quality of the other

[1] We are updating 1000 coalescent times for each locus in addition to τ and θ, however through marginalisation we ignore $t_{1:1000}$

[2] We explore different window sizes in the **Results** section to achieve an acceptance rate of $\sim 43\%$ for each ϕ

[3] An ESS greater than 1000 as per [7]

τ **window**	θ **window**	**Acceptance rate** (τ, θ)	**Efficiency** (τ, θ)	**Posterior mean with credibility interval**
5.3e-06	0.001	0.89, 0.021	0.059, 0.059	τ: 0.0070, [0.006164, 0.008422] θ: 0.0039, [0.003416, 0.004609]
0.001	9.e-06	0.44, 0.97	0.066, 0.059	τ:0.0049, [0.004494, 0.005270] θ: 0.002, [0.001415, 0.002608]
0.001	0.001	0.43, 0.43	0.08, 0.070	τ:0.0040, [0.003728, 0.004294] θ: 0.0039, [0.003416, 0.004609]
0.053	0.001	0.010, 0.43	0.059, 0.070	τ:0.0040, [0.003684, 0.004370] θ: 0.0040,[0.003287, 0.004624]

Table 1: Effect of varying window sizes on MCMC posterior samples, acceptance rates and efficiencies. In bold are the window sizes for which we manage to obtain the desired acceptance rate for both parameters.

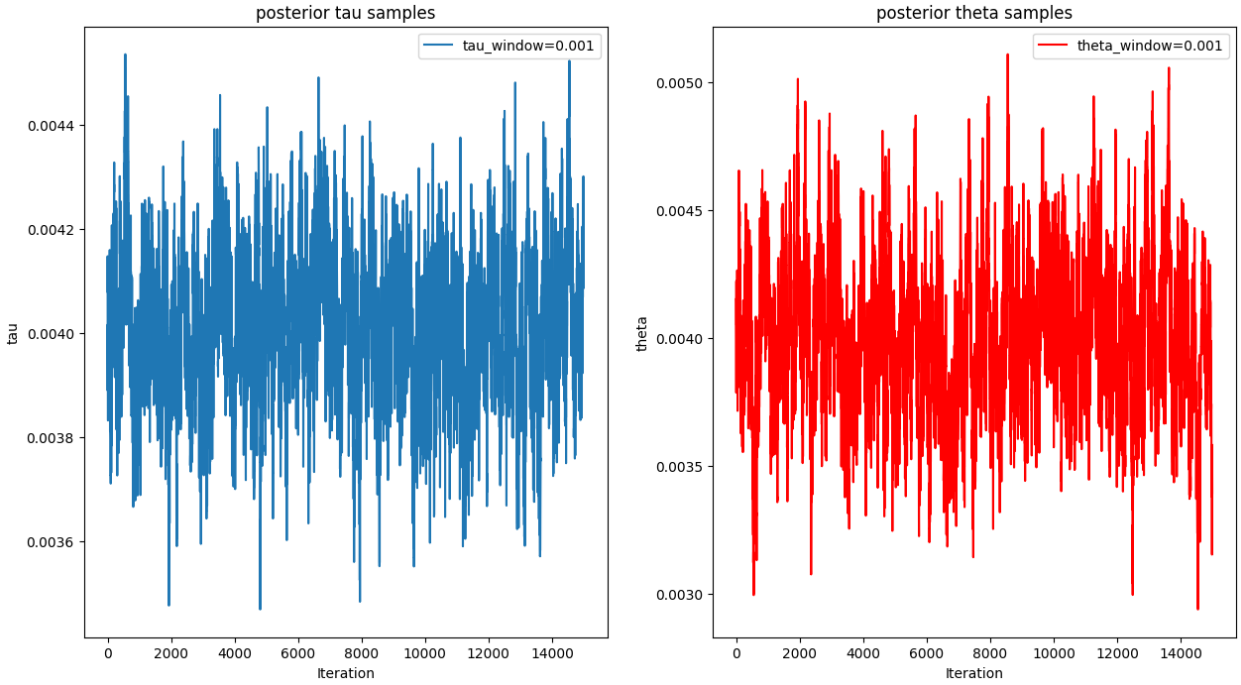

Figure 1: Trace plot for MCMC posterior τ and θ samples after burn-in.

parameter (see window sizes for τ : 5.3e-06 and θ: 0.001). This behavior can be explained by the computation of $\pi(\phi)$ which jointly depends on the current states of both parameters. In general, the choices of window sizes are performed through trial and error, as they depend on the characteristics of the specific problem, as well as the desired trade-off between acceptance rate, convergence time, and efficiency.

We also note a limitation of our work, which is that estimates are first affected by the assumptions of the MSC model, as well as the available data. We only worked with 1000 loci, while it's well established that more data yield more reliable estimates [3]. A preliminary run of our MCMC algorithm (using the same hyperparameters above: window size 0.001, 20000 iterations and 5000 burn-in) on all 14663 loci yield for τ a posterior mean of 0.004174 with credibility interval of $[0.004090,\ 0.004259]$. For θ, we obtain a posterior mean of 0.004370 and credibility interval of $[0.004203,\ 0.004551]$.

4 Conclusion

In this work, we use Bayesian MCMC to tractably estimate the parameters of a MSC model analyzing genomic data of humans and chimpanzees. Our vectorized implementation greatly optimizes the runtime of the MCMC. We release our implementation at `https://github.com/awxlong/machine_learning_projects_ucl/blob/main/msc_mcmc/mcmc-msc-vectorized-submit.ipynb`

References

[1] Bo Xu and Ziheng Yang. Challenges in species tree estimation under the multispecies coalescent model. *Genetics*, 204:1353–1368, 12 2016.

[2] James H. Degnan and Noah A. Rosenberg. Gene tree discordance, phylogenetic inference and the multispecies coalescent. *Trends in Ecology & Evolution*, 24:332–340, 06 2009.

[3] Bruce Rannala and Ziheng Yang. Bayes estimation of species divergence times and ancestral population sizes using dna sequences from multiple loci. *Genetics*, 164:1645–1656, 08 2003.

[4] Tomáš Flouri, Xiyun Jiao, Jun Huang, Bruce Rannala, and Ziheng Yang. Efficient bayesian inference under the multispecies coalescent with migration. *Proceedings of the National Academy of Sciences of the United States of America*, 120, 10 2023.

[5] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21:1087–1092, 06 1953.

A Inefficient MCMC

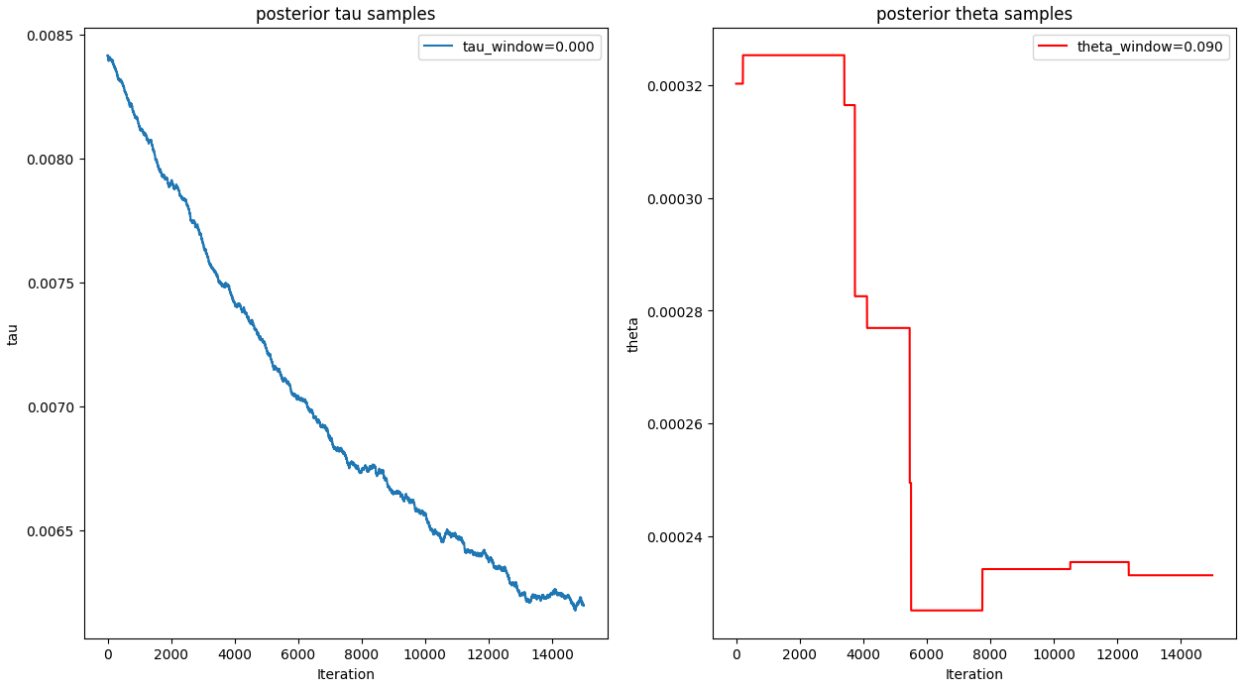

Figure 2: Example of an inefficient MCMC caused by ill-selected window sizes, where we observe no convergence after several iterations.

B Further experiments with window size

τ **Window**	θ **Window**	**Acceptance Rate** (τ, θ)	**Efficiency** (τ, θ)	**Posterior Mean with Credibility Interval**
5.3×10^{-6}	9.0×10^{-6}	0.89, 0.73	(0.059, 0.059)	τ : 0.0070, [0.006164, 0.008422] θ : 0.000094, [0.000061, 0.000140]
5.3×10^{-6}	0.001	0.89, 0.021	(0.059, 0.059)	τ : 0.0070, [0.006164, 0.008422] θ : 0.0039, [0.003416, 0.004609]
5.3×10^{-6}	0.090	0.89, 0.01	(0.059, 0.058)	τ : 0.0070, [0.006215, 0.008317] θ : 0.00026, [0.000227, 0.000325]
0.001	9.0×10^{-6}	0.44, 0.97	(0.066, 0.059)	τ : 0.0049, [0.004494, 0.005270] θ : 0.002, [0.001415, 0.002608]
0.001	**0.001**	**0.43, 0.43**	**(0.08, 0.07)**	τ : **0.0040, [0.003728, 0.004294]** θ : **0.0039, [0.003416, 0.004609]**
0.001	0.090	0.43, 0.001	(0.077, 0.059)	τ : 0.004, [0.003726, 0.004322] θ : 0.0039, [0.003394, 0.004633]
0.053	9.0×10^{-6}	0.009, 0.96	(0.0597, 0.059)	τ : 0.0049, [0.004573, 0.005115] θ : 0.002, [0.001625, 0.002453]
0.053	0.001	0.010, 0.43	(0.059, 0.070)	τ : 0.004, [0.003684, 0.004370] θ : 0.0040, [0.003287, 0.004624]
0.053	0.090	0.0096, 0.0092	(0.0603, 0.059)	τ : 0.00402, [0.003793, 0.004243] θ : 0.0040, [0.003314, 0.004662]

Table 2: Effect of varying window sizes on MCMC posterior samples, acceptance rates, and efficiencies. The highlighted row indicates the window sizes that achieve the desired acceptance rate for both parameters.