Bayesian Analysis (0000) **00**, Number 0, pp. 1

Bayesian analysis of continuous time Markov chains with application to phylogenetic modelling

Tingting Zhao[*], Ziyu Wang[†], Alexander Cumberworth[‡], Joerg Gsponer[‡], Nando de Freitas[†], and Alexandre Bouchard-Côté[*]

1 Bayesian optimization

1.1 Bayesian optimization for Hamiltonian Monte Carlo (HMC)

To provide additional background information, we provide here a detailed description of the Bayesian optimization method we used to adaptively tune the tuning parameters of the HMC algorithm.

Following the standard Bayesian optimization methodology, we set Γ to be a box constraint such that

$$\Gamma = \{(\epsilon, L) : \epsilon \in [b_l^\epsilon, b_u^\epsilon], L \in [b_l^L, b_u^L]\}$$

for some interval boundaries $b_l^\epsilon \leq b_u^\epsilon$ and $b_l^L \leq b_u^L$. The parameter L is discrete. The parameter ϵ is continuous, but since it is one-dimensional, we can discretize it using a very fine grid.

Since the true objective function is unknown, we specify a zero-mean Gaussian prior over it:

$$f(\cdot) \sim GP(0, k(\cdot, \cdot))$$

where $k(\cdot, \cdot)$ is the covariance function. Given a vector of noisy evaluations of the objective function $\bar{\boldsymbol{r}}_{\boldsymbol{i}} = \{r_k\}_{k=1}^i$ evaluated at points $\{\gamma_k\}_{k=1}^i$, we form the dataset $\mathcal{D}_i = \left(\{\gamma_k\}_{k=1}^i, \{r_k\}_{k=1}^i\right)$. Using Bayes rule, we arrive at the posterior predictive distribution over the unknown objective function (see Rasmussen and Williams (2006) for more details):

$$\begin{aligned} f|\mathcal{D}_i, \gamma &\sim \mathcal{N}(\mu_i(\gamma), \sigma_i^2(\gamma)) \\ \mu_i(\gamma) &= \bar{\boldsymbol{k}}^T (K + \sigma_\eta^2 I)^{-1} \bar{\boldsymbol{r}}_{\boldsymbol{i}} \\ \sigma_i^2(\gamma) &= k(\gamma, \gamma) - \bar{\boldsymbol{k}}^T (K + \sigma_\eta^2 I)^{-1} \bar{\boldsymbol{k}} \end{aligned}$$

where

$$K = \begin{bmatrix} k(\gamma_1, \gamma_1) & \dots & k(\gamma_1, \gamma_i) \\ \vdots & \ddots & \vdots \\ k(\gamma_i, \gamma_1) & \dots & k(\gamma_i, \gamma_i) \end{bmatrix},$$

 DOI: 0000

Algorithm 1 HMC Algorithm

Given: L, ϵ, and $\boldsymbol{w}^{(1)}$.
for $i = 1, 2, \cdots$ **do**
  Sample $\boldsymbol{m}^{(i)} \sim \mathcal{N}(0, I)$ and $L^{(i)} \sim \text{Uni}(\{1, \ldots, L\})$
  Let $\boldsymbol{w}^{(i,0)} := \boldsymbol{w}^{(i)}$ and $\boldsymbol{m}^{(i,0)} := \boldsymbol{m}^{(i)} - \frac{\epsilon}{2}\nabla_{\boldsymbol{w}} U(\boldsymbol{w}^{(i,0)})$
  for $j = 1, 2, \cdots, L^{(i)}$ **do**
    $\boldsymbol{w}^{(i,j)} := \boldsymbol{w}^{(i,j-1)} + \epsilon \boldsymbol{m}^{(i,j-1)}$
    if $j < L^{(i)}$**:** $\boldsymbol{m}^{(i,j)} := \boldsymbol{m}^{(i,j-1)} - \epsilon \nabla_{\boldsymbol{w}} U(\boldsymbol{w}^{(i,j)})$
  end for
  $\boldsymbol{m}^{(i,L^{(i)})} := \boldsymbol{m}^{(i,L^{(i)}-1)} - \frac{\epsilon}{2}\nabla_{\boldsymbol{w}} U(\boldsymbol{w}^{(i,L^{(i)})})$
  Draw $u \sim \text{Uni}(0, 1)$
  if $u < \min[1, \exp\{U(\boldsymbol{w}^{(i)}) + K(\boldsymbol{m}^{(i)}) - U(\boldsymbol{w}^{(i,L^{(i)})}) - K(\boldsymbol{m}^{(i,L^{(i)})})\}]$ **then**
    Let $(\boldsymbol{w}^{(i+1)}, \boldsymbol{m}^{(i+1)}) := (\boldsymbol{w}^{(i,L^{(i)})}, \boldsymbol{m}^{(i,L^{(i)})})$
  else
    Let $(\boldsymbol{w}^{(i+1)}, \boldsymbol{m}^{(i+1)}) := (\boldsymbol{w}^{(i,0)}, \boldsymbol{m}^{(i,0)})$
  end if
end for

$\bar{\boldsymbol{k}} = [k(\gamma, \gamma_1) \ \ldots \ k(\gamma, \gamma_i)]^T$, and $\bar{\boldsymbol{r}}_{\boldsymbol{i}} = [r_1 \ \ldots \ r_i]^T$.

We adopt an automatic relevance determination covariance function with $k(\gamma_i, \gamma_j) = \exp(-\frac{1}{2}\gamma_i^T \Sigma^{-1} \gamma_j)$, where Σ is a positive definite matrix. We set $\Sigma = \text{diag}\Big([\alpha(b_u^\epsilon - b_l^\epsilon)]^2 ; \big[\alpha(b_u^L - b_l^L)\big]^2 \Big)$, where $\alpha = 0.2$.

The Gaussian process simply provides a surrogate model for the true objective. The surrogate can be used to efficiently search for the maximum of the objective function. In particular, it enables us to construct an acquisition function $u(\cdot)$ that tells us which parameters γ to try next. The acquisition function uses the Gaussian process posterior mean to predict regions of potentially higher objective values (exploitation). It also uses the posterior variance to detect regions of high uncertainty (exploration). Moreover, it effectively trades-off exploration and exploitation. Different acquisition functions have been proposed in the literature, including Mockus (1982); Srinivas et al. (2010); Hoffman et al. (2011). We adopt a variant of the Upper Confidence Bound (UCB) (Srinivas et al., 2010), modified to suit our application:

$$u(\gamma, s|\mathcal{D}_i) = \mu_i(\gamma, s) + p_i \beta_{i+1}^{\frac{1}{2}} \sigma_i(\gamma).$$

As in standard UCB, we set $\beta_{i+1} = 2\log\left(\frac{(i+1)^{\frac{d}{2}+2}\pi^2}{3\delta}\right)$, where d is the dimension of Γ and δ is set to 0.1. The parameter p_i ensures that the diminishing adaptation condition of Roberts and Rosenthal (2007) is satisfied. Specifically, we set $p_i = (\max\{i-k+1, 1\})^{-0.5}$ for some $k \in \mathbb{N}^+$. As p_i goes to 0, the probability of Bayesian optimization adapting γ vanishes. For efficiency, we also impose a hard limit N_{adapt} to the number of adaptation steps. A detailed convergence analysis is presented in Wang et al. (2013).

The acquisition function also includes a scalar scale-invariance parameter s, such that $\mu_i(\gamma, s) = \bar{\boldsymbol{k}}^T (K + \sigma_\eta^2 I)^{-1} \bar{\boldsymbol{r}}_{\boldsymbol{i}} s$. This parameter is estimated automatically so as to

rescale the rewards to the same range each time we encounter a new maximal reward.

1.2 Computational cost of Adaptive Hamiltonian Monte Carlo (AHMC)

Gaussian processes require the inversion of the covariance matrix and, hence, have complexity $\mathcal{O}(i^3)$, where i is the number of iterations. Fortunately, thanks to our annealing schedule, the number of unique points in our Gaussian process grows sub-linearly with the number of iterations. This slow growth makes it possible to adopt kernel specification techniques, as proposed by Engel (2005), to drastically reduce the computational cost without suffering any loss in accuracy.

Following Wang et al. (2013), in all our experiments, we set $\alpha = 4$, $k = 100$, and $m = \frac{B}{k}$, where B is the number of burn-in samples. In our experience, the algorithm is robust with respect to these settings and we used the same set of parameters throughout our experiments.

2 Running time of previous CTMC gradient computation methods

We review here the analysis of the running time of previous methods for computing the gradient of individual entries of a rate matrix from a partially observed Continuous Time Markov Chains (CTMC). A good review of the state-of-the-art method can be found in Kenney and Gu (2012).[1] Kenney and Gu (2012) summarize the running time on page 25 as $\mathcal{O}(K(P + |E|)|\mathcal{X}|^2)$ (translating their notation to ours).[2]

Close inspection of the analysis on page 40, step 2, reveals that a term of $\mathcal{O}(P|\mathcal{X}|^3)$ is absorbed in this overall running time. This term arises from the multiplication of the matrix of derivative of the rate matrix with respect to each parameter with a fixed dense matrix coming from the eigen-decomposition (see the definition of N_β in Theorem 1 of Kenney and Gu (2012)). It takes into account the caching of the eigen-decomposition.

Absorbing the $\mathcal{O}(P|\mathcal{X}|^3)$ term into $\mathcal{O}((P + |E|)|\mathcal{X}|^2 K)$ is reasonable given that the number of sites K is often larger than the number of CTMC states $|\mathcal{X}|$ in practice. However, it highlights the fact that when the number of parameters is of order $\mathcal{O}(|\mathcal{X}|^2)$, previous methods will scale as $\mathcal{O}(|\mathcal{X}|^5)$. Moreover, the entire algorithm needs to be started from scratch after each of the L leapfrog step, giving an overall running time of $\mathcal{O}(L|\mathcal{X}|^5)$.

[1] Although Kenney and Gu (2012) focuses on Hessian computation, it also includes a detailed running time analysis of gradient computation, found in Appendix 2.

[2] We present the algorithm in the context of a phylogenetic tree with K sites. We recover the case of a time series as a special case by setting $K = 1$.

3 Cached-Uniformization method

We use uniformization for exact sampling from a CTMC conditional on the initial and ending states. The key idea of uniformization is to transform a continuous time Markov process to a discrete one subordinated to a Poisson process. The general procedure of uniformization is described in Hobolth and Stone (2009).

Uniformization is based on a transition probability constructed from the rate matrix by adding self-transitions (virtual jumps):

$$B = I + \frac{1}{\bar{q}} Q \tag{3.1}$$

$$\bar{q} = \max_x |q_{x,x}|. \tag{3.2}$$

Given two states at the two end-points of a single branch the posterior distribution over the number of jumps (which can be either between distinct states, or self-transitions) J is given by:

$$\mathbb{P}(J = j | X_0 = x_1, X_t = x_2) = e^{-\bar{q}t} \frac{(\bar{q}t)^j}{j!} \frac{\left(B^j\right)_{x_1,x_2}}{(\exp(tQ)_{x_1,x_2}}. \tag{3.3}$$

Given the number of transitions $J = j$ and the end-points, the list of states visited is obtained by simulating a discrete Markov chain of lengths $j + 2$, where the first state conditioned to be x_1, the last state, x_2, and the transitions probabilities are given by B.

Finally, the location of the jumps (between distinct states, or self-transitions) are obtained by simulating J independent uniform numbers and sorting them. The self-transitions can be discarded as a post-processing step.

The computational bottleneck of this process is the evaluation of $\left(B^j\right)_{x_1,x_2}$ for various powers of j. If one is concerned with a single branch, it may be attractive to do this computation in the following way:

$$\left(B^j\right)_{x_1,x_2} = (\cdots((\boldsymbol{e}_{x_1} B)B \ldots B)_{x_2}, \tag{3.4}$$

where $\boldsymbol{e}_x$ is the standard basis vector with a one at entry x, and equal to zero otherwise. This yields a running time of $J|\mathcal{X}|^2$.

However, since the matrix B is shared among all sites and all branches, it becomes attractive to store the powers of B in a phylogenetic context. This requires a more expensive compute time of $|\mathcal{X}|^3$ per power, but this pays off in a phylogenetic context, since most branches have moderate lengths, which implies that the number of transitions per branch is also moderate.

We use a cache taking the form of a list, cache, initialized to contain a single element: cache[1] $\leftarrow B$. We retrieve from the cache using the procedure get_and_cache(j), defined as follows:

1. If the length of the list cache is smaller than j, store in cache$[j]$ the product of the matrices B and get_and_cache$(j-1)$.

2. Return cache$[j]$.

We denote by M_{e} the expectation of the number of times Step 1 requires the computation of a matrix multiplication, over the full process of resampling once the auxiliary variable $\boldsymbol{Z}$.

Using this cache, we sample $\boldsymbol{Z}$ as follows (writing in brackets the total running time of each step over the entire process of resampling once the auxiliary variable $\boldsymbol{Z}$:[3]

1. Form the matrix in Equation (3.1). $\{|\mathcal{X}|^2\}$

2. Compute the sum-product algorithm, and use its output to sample reconstructed states for every end-point of every branch and site (see the Background section of the main paper for details). {Reversible case: $|\mathcal{X}|^3 + 2K|E| \cdot |\mathcal{X}|^2 + K|E| \cdot |\mathcal{X}|$, non-reversible case: $|E| \cdot |\mathcal{X}|^3 + K|E| \cdot |\mathcal{X}|^2 + K|E| \cdot |\mathcal{X}|$ (both in the reversible and non-reversible cases, the first two terms account for the computation of the sum-product algorithm, and the last term is the cost of sampling internal reconstructions)}[4]

3. Initialize transition counts $\boldsymbol{c}$ and total sojourn times $\boldsymbol{h}$ to zero. $\{|\mathcal{X}|^2\}$

4. In the reversible case, perform the eigen-decomposition of Q. In the non-reversible case, for each branch of length Δ, cache the matrix exponential, $\exp(\Delta Q)$. {Reversible case: $|\mathcal{X}|^3$, non-reversible case: $|E| \cdot |\mathcal{X}|^3$}

5. For each branch and each site with end points x_1 and x_2 (obtained from Step 2 above):

   (a) Simulate $J = j$ using the following method:

      i. Simulate a uniform random number $u \in [0, 1]$. $\{K|E|\}$

      ii. Set $s \leftarrow 0$ $\{K|E|\}$

      iii. Loop, for $j = 0, 1, \ldots$

         A. $s \leftarrow s + e^{-\bar{q}\Delta} \frac{(\bar{q}\Delta)^j}{j!} \frac{(\text{get\_and\_cache}(j))_{x_1,x_2}}{(\exp(\Delta Q)_{x_1,x_2}}$ {Reversible case: $\mathcal{O}(KM_{\mathrm{e}}|\mathcal{X}| \cdot |E| + M_{\mathrm{e}}|\mathcal{X}|^3)$, non-reversible case: $\mathcal{O}(KM_{\mathrm{e}} \cdot |E| + M_{\mathrm{e}}|\mathcal{X}|^3)$ (using the cache of the exponentiated transition matrix, and the cached matrix exponential/diagonalization (step 4))}

         B. If $s > u$, break the inner loop. $\{K|E|M_{\mathrm{e}}\}$

[3] As in the previous section, we present the algorithm in the context of a phylogenetic tree with K sites. We recover the case of a time series as a special case by setting $K = 1$.

[4] In the reversible case, the running time assumes that matrix exponentiation is performed via diagonalization (at the cost of $|\mathcal{X}|^3$) (Moler and Van Loan, 1978). In the non-reversible case, the matrix exponentials can be computed using the Pade method at a cost of $|E| \cdot |\mathcal{X}|^3$ (Moler and Van Loan, 1978).

(b) Simulate J uniform numbers in the interval $[0, \Delta]$, and sort them to obtain a list $u_1 < u_2 < \cdots < u_J$. {This can be done in time $K|E|M_e$ by simulating spacings via exponential random variables and normalizing them (Carpenter et al., 1999)}

(c) To sample the path, initialize a current state x to the first end point, x_1, and loop over $i = 1, 2, \ldots, J$:

i. Form a vector of posterior transition probabilities $\boldsymbol{p} = (p_1, \ldots, p_{|\mathcal{X}|})$, with entry x' given by:

$$p_{x'} = B_{x,x'} \left(\mathsf{get\_and\_cache}(J - i)\right)_{x',x_2} . \tag{3.5}$$

{$KM_e|E| \cdot |\mathcal{X}|$}

ii. Normalize the entries in $\boldsymbol{p}$ to form a probability distribution, and sample from it to obtain a new state $\tilde{x}$. {$KM_e|E| \cdot |\mathcal{X}|$}

iii. Update the sufficient statistics $\boldsymbol{c}$ and $\boldsymbol{h}$ using the transition $(x, \tilde{x})$ and corresponding time $u_{i+1} - u_i$. {$KM_e|E|$}

iv. Update the current state, $x \leftarrow \tilde{x}$. {$KM_e|E|$}

In the reversible case, the running time is $\mathcal{O}(K|E| \cdot |\mathcal{X}|^2 + M_e|\mathcal{X}|^3 + KM_e|E| \cdot |\mathcal{X}|)$, and in the non-reversible case, $\mathcal{O}(K|E| \cdot |\mathcal{X}|^2 + M_e|\mathcal{X}|^3 + |E| \cdot |\mathcal{X}|^3 + KM_e|E| \cdot |\mathcal{X}|)$.

We also investigate the relationship between the branch lengths and the cache size using cached uniformization to sample $\boldsymbol{Z}$. We generate pairs of DNA sequences of 5000 sites under GTR with branch lengths from 0.2 to 5 with step size 0.2. See Figure 7. The figure indicates that the cache size increases linearly as the branch length increases.

4 Running time for the gradient computation via auxiliary variables

The running time of the precomputation step is analyzed in the previous section. In general, the dominant term will be $|E|K|\mathcal{X}|^2$, namely the cost of a sum-product execution. Importantly, this factor is *not* multiplied by the number leapfrog steps, as we are not required to refresh the auxiliary variable at each leapfrog step.

We now compute the running time of the operations that need to be computed at each leapfrog step j, which boils down to computing $\nabla U_{\boldsymbol{z}^{(i)}}(\boldsymbol{w}^{(i,j)})$. Naively, this would take $\mathcal{O}(|\mathcal{X}|^2 P)$. However, we can exploit the fact that $\boldsymbol{\varphi}(x, x')$ is often sparse. If $s := \sum_{(x,x') \in \mathcal{X}^{\text{distinct}}} \sum_{m=1}^{P} \mathbf{1}[\varphi_m(x, x') \neq 0]$ denotes the total number of non-zero entries across all possible features, then $\nabla U_{\boldsymbol{z}^{(i)}}(\boldsymbol{w}^{(i,j)})$ can be computed in time $\mathcal{O}(s)$, therefore, computation of an HMC iteration has a running time of $\mathcal{O}(sL)$. In all examples presented in the paper, $s \in \mathcal{O}(|\mathcal{X}|^2)$.

Note that this type of sparsity is distinct from the sparsity in the matrix Q exploited in previous work (Rao and Teh, 2013). In our examples, Q is not sparse.

5 Gradient computation for the reversible, normalized parameterization

In this section, we derive the gradient expression for the reversible, normalized model introduced the main paper. The derivation is lengthy, but note that the auxiliary variable $\boldsymbol{Z}$ makes it possible to generate gradient calculation code directly from the augmented joint distribution, for example using Stan. In contrast, as discussed in the main manuscript, this is not practical without conditioning on the auxiliary variable $\boldsymbol{Z}$. We show the expression of the gradient here for completeness. The partial derivatives have been verified numerically using numerical differentiation. The log-likelihood with the normalized rate matrix is:

$$\begin{aligned}\log f^{(\mathrm{norm})}_{\mathrm{w|z,y}}(\boldsymbol{w}|\boldsymbol{z},\boldsymbol{y}) \quad &:= \quad \sum_{x\in\mathcal{X}} n_x \log \pi_x(\boldsymbol{w}) + \sum_{x\in\mathcal{X}}\sum_{x'\in\mathcal{X}:x\neq x'} c_{x,x'} \log q^{(\mathrm{norm})}_{x,x'}(\boldsymbol{w}) \\ &\quad - \sum_{x\in\mathcal{X}} h_x \sum_{x'\in\mathcal{X}:x\neq x'} q^{(\mathrm{norm})}_{x,x'}(\boldsymbol{w}) + \text{constant}. \end{aligned} \tag{5.1}$$

Computing the gradient is now reduced to a routine derivation yielding:

$$\begin{aligned}\nabla\left[\log f^{(\mathrm{norm})}_{\mathrm{w|z,y}}(\boldsymbol{w}|\boldsymbol{z},\boldsymbol{y})\right] \quad &= \quad \sum_{x\in\mathcal{X}} n_x \left(\boldsymbol{\psi}(x) - \nabla A(\boldsymbol{w})\right) \\ &\quad + \sum_{x\in\mathcal{X}}\sum_{x'\in\mathcal{X}:x\neq x'} \left(c_{x,x'} - h_x q^{(\mathrm{norm})}_{x,x'}(\boldsymbol{w})\right) \\ &\quad \times\Big(\beta^{-1}\nabla\beta(\boldsymbol{w}) + \boldsymbol{\phi}(\{x,x'\}) + \boldsymbol{\psi}(x') - \nabla A(\boldsymbol{w})\Big), \end{aligned} \tag{5.2}$$

where:

$$\nabla A(\boldsymbol{w}) \quad = \quad \sum_{x\in\mathcal{X}} \boldsymbol{\psi}(x)\pi_x(\boldsymbol{w}), \tag{5.3}$$

$$\nabla\beta(\boldsymbol{w}) \quad = \quad \beta\sum_{x\in\mathcal{X}} \pi_x(\boldsymbol{w}) \left(2\boldsymbol{\psi}(x) - \left(\sum_{x'\in\mathcal{X}:x\neq x'} \Big(\boldsymbol{\psi}(x) + \boldsymbol{\psi}(x') + \boldsymbol{\phi}(\{x,x'\})\Big) q^{(\mathrm{norm})}_{x,x'}(\boldsymbol{w})\right)\right).$$

6 Modelling language

Our framework can incorporate as special cases most existing DNA, amino acid and codon evolution models. For simplicity, we take the HKY85 model (Hasegawa et al., 1985) as an example to show how to represent a previously constructed rate matrix using our framework. We start by showing mathematically how the classical HKY85 model is translated into features and weights. We then show concretely how it is input into our software implementation using a small, JSON-based modelling language.

The classic representation of HKY85 model is

$$Q = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array} \begin{array}{c} \begin{array}{cccc} A & C & G & T \end{array} \\ \begin{pmatrix} * & \pi_C & \kappa\pi_G & \pi_T \\ \pi_A & * & \pi_G & \kappa\pi_T \\ \kappa\pi_A & \pi_C & * & \pi_T \\ \pi_A & \kappa\pi_C & \pi_G & * \end{pmatrix} \end{array} \cdot \beta,$$

where $\beta = 1/\left(2(\pi_A + \pi_G)(\pi_C + \pi_T) + 2\kappa(\pi_A\pi_G + \pi_C\pi_T)\right)$ to ensure that the expected base change per unit time is one, and the diagonal elements, "*", enforce that each row sums to zero. The "A" and "G" nucleotides contain bases that belong to a chemical group known as purines, while the C and T nucleotides contain bases that belong to a chemical group known as pyrimidines. Importantly, substitutions are more frequent between members of the same chemical group, motivating the addition of an extra parameter, κ, to differentiate these intragroup substitutions (transitions) from intergroup substitutions (transversions).

To encode this under our Bayesian rate matrix GLMs, we set $\underline{\theta}(\{x, x'\}) = 1$ and $\underline{\pi}(x) = 1$, allowing us to simplify our model into:

$$\begin{aligned}
\theta_{\{x,x'\}} &= \theta_{\{x,x'\}}(\boldsymbol{w}) = \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\phi}(\{x, x'\})\rangle\right\}, \\
\pi_x &= \pi_x(\boldsymbol{w}) = \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\psi}(x)\rangle - A(\boldsymbol{w})\right\}, \\
A(\boldsymbol{w}) &= \log \sum_{x\in\mathcal{X}} \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\psi}(x)\rangle\right\}, \\
q^{(\mathrm{rev})}_{x,x'}(\boldsymbol{w}) &= \begin{cases} \theta_{\{x,x'\}}(\boldsymbol{w})\pi_{x'}(\boldsymbol{w}) & \text{if } x \neq x', \\ -\sum_{z:z\neq x} q_{x,z}(\boldsymbol{w}) & \text{otherwise.} \end{cases} \\
\beta(\boldsymbol{w}) &= -\left(\sum_{x\in\mathcal{X}} \pi_x(\boldsymbol{w}) q^{(\mathrm{rev})}_{x,x}(\boldsymbol{w})\right)^{-1},
\end{aligned}$$

$$q^{(\mathrm{norm})}_{x,x'}(\boldsymbol{w}) = \beta(\boldsymbol{w}) q^{(\mathrm{rev})}_{x,x'}(\boldsymbol{w}).$$

We have $\boldsymbol{w} \in \mathbb{R}^5$, as there are four univariate feature used to calculate the stationary distribution for the four states "A", "C", "G", "T" and one bivariate feature to differentiate transitions from transversions. Univariate features are determined by only one state in the state space $\mathcal{X}$, while bivariate features are characterized by an unordered pair of states. We represent the weights as a vector $\boldsymbol{w} = (w_{\mathrm{A}}, w_{\mathrm{C}}, w_{\mathrm{G}}, w_{\mathrm{T}}, w_\kappa)^T$. The sufficient statistic of the π-exponential family for $x = \mathrm{A}$ is $\boldsymbol{\psi}(\mathrm{A}) = (1, 0, 0, 0, 0)^T$, while for $x = \mathrm{C}$, it is $\boldsymbol{\psi}(\mathrm{C}) = (0, 1, 0, 0, 0)^T$. The normalization used to calculate the stationary

distribution is given by:

$$\begin{aligned}\exp(A(\boldsymbol{w})) &= \sum_{x\in\{\mathrm{A,C,G,T}\}} \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\psi}(x)\rangle\right\} \\ &= \exp\left\{(w_{\mathrm{A}}, w_{\mathrm{C}}, w_{\mathrm{G}}, w_{\mathrm{T}}, w_{\kappa})^T \cdot \boldsymbol{\psi}(\mathrm{A})\right\} + \\ &\exp\left\{(w_{\mathrm{A}}, w_{\mathrm{C}}, w_{\mathrm{G}}, w_{\mathrm{T}}, w_{\kappa})^T \cdot \boldsymbol{\psi}(\mathrm{C})\right\} + \\ &\exp\left\{(w_{\mathrm{A}}, w_{\mathrm{C}}, w_{\mathrm{G}}, w_{\mathrm{T}}, w_{\kappa})^T \cdot \boldsymbol{\psi}(\mathrm{G})\right\} + \\ &\exp\left\{(w_{\mathrm{A}}, w_{\mathrm{C}}, w_{\mathrm{G}}, w_{\mathrm{T}}, w_{\kappa})^T \cdot \boldsymbol{\psi}(\mathrm{T})\right\} \\ &= \exp(w_{\mathrm{A}}) + \exp(w_{\mathrm{C}}) + \exp(w_{\mathrm{G}}) + \exp(w_{\mathrm{T}})\end{aligned}$$

The stationary distribution $\pi_x(\boldsymbol{w})$, where $x \in \{\mathrm{A,C,G,T}\}$ may be calculated as (we take the case of $x = A$ without loss of generality)

$$\begin{aligned}\pi_{\mathrm{A}}(\boldsymbol{w}) &= \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\psi}(\mathrm{A})\rangle - A(\boldsymbol{w})\right\} \\ &= \frac{\exp(w_{\mathrm{A}})}{\exp(w_{\mathrm{A}}) + \exp(w_{\mathrm{C}}) + \exp(w_{\mathrm{G}}) + \exp(w_{\mathrm{T}})}.\end{aligned}$$

For the sufficient statistics of the ϕ-exponential family, if the substitution between x, x' is a transition, $\boldsymbol{\phi}(x, x') = (0, 0, 0, 0, 1)^T$, otherwise $\boldsymbol{\phi}(x, x') = (0, 0, 0, 0, 0)^T$. Using the above results, we can now calculate the rates, which for a transition between A and G would be

$$\begin{aligned}q_{\mathrm{A,G}}(\boldsymbol{w}) &= \theta_{\{\mathrm{A,G}\}}(\boldsymbol{w})\pi_{\mathrm{G}}(\boldsymbol{w}) \\ &= \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\phi}(\{\mathrm{A,G}\})\rangle\right\}\pi_{\mathrm{G}}(\boldsymbol{w}) \\ &= \exp\left\{\langle (w_{\mathrm{A}}, w_{\mathrm{C}}, w_{\mathrm{G}}, w_{\mathrm{T}}, w_{\kappa})^T, (0, 0, 0, 0, 1)\rangle\right\}\pi_{\mathrm{G}}(\boldsymbol{w}) \\ &= \exp(w_{\kappa})\pi_{\mathrm{G}}(\boldsymbol{w}) \\ &= \kappa\pi_{\mathrm{G}}(\boldsymbol{w}). \qquad (6.1)\end{aligned}$$

If the substitution is a transversion between a purine A and a pyrimidine C,

$$\begin{aligned}q_{\mathrm{A,C}}(\boldsymbol{w}) &= \theta_{\{\mathrm{A,C}\}}(\boldsymbol{w})\pi_{\mathrm{C}}(\boldsymbol{w}) \\ &= \exp\left\{\langle (w_{\mathrm{A}}, w_{\mathrm{C}}, w_{\mathrm{G}}, w_{\mathrm{T}}, w_{\kappa})^T, (0, 0, 0, 0, 0)\rangle\right\}\pi_{\mathrm{C}}(\boldsymbol{w}) \\ &= \pi_{\mathrm{C}}(\boldsymbol{w}). \qquad (6.2)\end{aligned}$$

Equation (6.1) and Equation (6.2) show the equivalence of classic HKY85 rate matrix representation and our weights, feature construction.

We now provide here the JSON file for the HKY85 read by software implementation.

```
{
  "nCategories" : 1,
  "orderedLatents" : ["A","C","G","T"],
  "fullSupport" : true,
  "unaryFeatures" :
  [
```

```
    {
      "state" : { "categoryIndex" : 0, "latent" : "A" },
      "features" : { "statio(A)" : 1.0 }
    },
    {
      "state" : { "categoryIndex" : 0, "latent" : "C" },
      "features" : { "statio(C)" : 1.0 }
    },
    {
      "state" : { "categoryIndex" : 0, "latent" : "G" },
      "features" : { "statio(G)" : 1.0 }
    },
    {
      "state" : { "categoryIndex" : 0, "latent" : "T" },
      "features" : { "statio(T)" : 1.0 }
    }
  ],
  "binaryFeatures" :
  [
    {
      "state0" : { "categoryIndex" : 0, "latent" : "A" },
      "state1" : { "categoryIndex" : 0, "latent" : "G" },
      "features" : { "isTransition" : 1.0 }
    },
    {
      "state0" : { "categoryIndex" : 0, "latent" : "C" },
      "state1" : { "categoryIndex" : 0, "latent" : "T" },
      "features" : { "isTransition" : 1.0 }
    }
  ]
}
```

Here, "unaryFeatures" represents the univariate features and "binaryFeatures" represents the bivariate features. For example,

```
  "unaryFeatures" :
  {
    "state" : { "categoryIndex" : 0, "latent" : "A" },
    "features" : { "statio(A)" : 1.0 }
    }
```

indicates that when the "latent" state is A (the terminology "latent" is used since it is possible to have the emissions different than the states in the CTMC, a feature not needed here—by default, latents and emissions are assumed to match), the elements in $\boldsymbol{\psi}(A)$ should be zero everywhere, except for the coordinate corresponding to univariate feature A, equivalently, $\boldsymbol{\psi}(\mathrm{A}) = (1, 0, 0, 0, 0)^T$. Here, "statio(A)" is an arbitrary string used to index the coordinates of the weights and sufficient statistics. The only assumption is that this string should be distinct to the other feature labels.

Next, let us look at a "binaryFeatures":

```
  "binaryFeatures" :
```

```
{
  "state0" : { "categoryIndex" : 0, "latent" : "A" },
  "state1" : { "categoryIndex" : 0, "latent" : "G" },
  "features" : { "isTransition" : 1.0 }
}
```

which means that if the substitution is between "A" and "G", the value of the "isTransition" feature coordinate should be one, indicating $\boldsymbol{\phi}(\{A, G\}) = (0, 0, 0, 0, 1)^T$.

Finally, the header gives general information on the state space:

```
{
  "nCategories" : 1,
  "orderedLatents" : ["A","C","G","T"],
  "fullSupport" : true,
  }
```

"nCategories : 1" refers to the fact that no rate variations or other latent structure is assumed. In general, the number of categories can be any positive integer. The different latent states can each have their own combination of features. Finally, "orderedLatents : ["A", "C", "G", "T"]" simply represents the state in the CTMC. In this example, our state space consists of four nucleotides ["A","C","G","T"]. "fullSupport : true" indicates that the base measure is uniform, i.e. all transitions between states are allowed (otherwise, an adjacency graph can be provided).

7 Full weights in the synthetic data experiments

```
Weights
$state
 [1] "A" "R" "N" "D" "C" "Q" "E" "G" "H" "L" "I" "K" "M" "F" "P" "S" "T" "
     W" "Y"
[20] "V"

$univariateWeight
 [1]  0.098  0.113 -0.192  0.110 -0.173  0.058  0.172  0.087  0.171 -0.086
[11]  0.022  0.019  0.033  0.033 -0.200 -0.024 -0.075  0.096 -0.145  0.149

$bivariateFeature
  [1] "AR" "AN" "AD" "AC" "AQ" "AE" "AG" "AH" "AL" "AI" "AK" "AM" "AF" "AP
      " "AS"
 [16] "AT" "AW" "AY" "AV" "RN" "RD" "RC" "RQ" "RE" "RG" "RH" "RL" "RI" "RK
      " "RM"
 [31] "RF" "RP" "RS" "RT" "RW" "RY" "RV" "ND" "NC" "NQ" "NE" "NG" "NH" "NL
      " "NI"
 [46] "NK" "NM" "NF" "NP" "NS" "NT" "NW" "NY" "NV" "DC" "DQ" "DE" "DG" "DH
      " "DL"
 [61] "DI" "DK" "DM" "DF" "DP" "DS" "DT" "DW" "DY" "DV" "CQ" "CE" "CG" "CH
      " "CL"
 [76] "CI" "CK" "CM" "CF" "CP" "CS" "CT" "CW" "CY" "CV" "QE" "QG" "QH" "QL
      " "QI"
 [91] "QK" "QM" "QF" "QP" "QS" "QT" "QW" "QY" "QV" "EG" "EH" "EL" "EI" "EK
      " "EM"
[106] "EF" "EP" "ES" "ET" "EW" "EY" "EV" "GH" "GL" "GI" "GK" "GM" "GF" "GP
      " "GS"
[121] "GT" "GW" "GY" "GV" "HL" "HI" "HK" "HM" "HF" "HP" "HS" "HT" "HW" "HY
      " "HV"
[136] "LI" "LK" "LM" "LF" "LP" "LS" "LT" "LW" "LY" "LV" "IK" "IM" "IF" "IP
      " "IS"
[151] "IT" "IW" "IY" "IV" "KM" "KF" "KP" "KS" "KT" "KW" "KY" "KV" "MF" "MP
      " "MS"
[166] "MT" "MW" "MY" "MV" "FP" "FS" "FT" "FW" "FY" "FV" "PS" "PT" "PW" "PY
      " "PV"
[181] "ST" "SW" "SY" "SV" "TW" "TY" "TV" "WY" "WV" "YV"

$bivariateWeight
  [1]  0.491  0.563 -0.960  0.552 -0.866  0.290  0.859  0.435  0.855 -0.43
       2
 [11]  0.111  0.095  0.166  0.166 -0.998 -0.118 -0.374  0.480 -0.723  0.74
       4
 [21]  0.046  0.158  0.730  0.235 -0.020 -0.250  0.393 -0.617  0.678  0.76
       5
 [31]  0.238 -0.495 -0.648  0.410  0.067  0.376  0.402 -0.693  0.013 -0.29
       1
 [41]  0.163  0.815  0.691 -0.535  0.317  0.554 -0.519  0.257  0.186 -0.74
       7
 [51]  0.270 -0.240 -0.692  0.615 -0.842 -0.711 -0.827 -0.077  0.617  0.36
       5
 [61] -0.264 -0.021 -0.490  0.729 -0.080 -0.212  0.519  0.008  0.339  0.20
       4
 [71]  0.866 -0.307  0.171 -0.375 -0.865 -0.240  0.099 -0.221 -0.701 -0.75
       6
```

```
 [81]  0.174  0.072 -0.339  0.690  0.861  0.290 -0.422  0.912 -0.337  0.64
    7
 [91]  0.707  0.710 -0.696 -0.707  0.032  0.987 -0.257 -0.255  0.378  0.02
    5
[101]  0.562  0.497 -0.476 -0.112 -0.342  0.800 -0.732 -0.465 -0.684  0.12
    7
[111]  0.458 -0.610  0.237  0.250 -0.532 -0.933 -0.662 -0.738  0.470  0.00
    2
[121]  0.492 -0.228  0.183  0.191  0.321 -0.736  0.373  0.395 -0.651 -0.69
    8
[131]  0.063 -0.278  0.438  0.877 -0.074  0.812  0.324 -0.748  0.285 -0.41
    1
[141]  0.698  0.836  0.159  0.298  0.836  0.914 -0.290 -0.297 -0.676 -0.16
    6
[151]  0.193 -0.381  0.965  0.181 -0.848 -0.563 -0.726  0.158  0.186  0.16
    3
[161]  0.267 -0.605 -0.905 -0.477  0.532 -0.774  0.512  0.762 -0.908 -0.55
    1
[171]  0.018 -0.270  0.874  0.507 -0.733 -0.662 -0.162 -0.238 -0.315 -0.35
    6
[181] -0.345 -0.447 -0.652  0.595  0.829  0.599  0.285 -0.830  0.692  0.78
    7

$polarityFeature
 [1] "AcidicAcidic" "AcidicBasic"  "AcidicNon"    "AcidicPolar"  "
     BasicBasic"
 [6] "BasicNon"     "BasicPolar"   "NonNon"       "NonPolar"     "
     PolarPolar"

$polarityWeight
 [1]  0.35  0.20 -0.35  0.20  0.35 -0.35  0.20  0.35 -0.35  0.35

$size
[1] "BigBig"     "BigMicro"   "MicroMicro"

$sizeWeight
[1]  0.25 -0.40  0.40
```

8 Connection Between Bayesian Generalized Linear Models (GLMs) and General Time Reversible model (GTR)

In this section, we build the connection between our Bayesian GLMs framework and the GTR model. GTR is a special case within our framework under a particular configuration. Our main conclusion is that a Normal prior $f_{\text{nor}}(\boldsymbol{w})$ on weights $\boldsymbol{w}$ under GTR is equivalent to a log-Normal prior on exchangeable coefficients $\theta_{\{x,x'\}}$, for any $\{x, x'\} \in \mathcal{X}^{\text{unordered,dist.}}$, the set of unordered distinct pairs of states and a logistic-Normal prior on the stationary distribution π_x if the GTR rate matrix Q is parameterized by $\theta_{\{x,x'\}}$ and π_x.

8.1 Configuration of GTR using Bayesian GLMs

For any $x, x' \in \mathcal{X}$, $x \neq x'$, Q is defined as

$$q_{x,x'}^{(\text{rev})} = \theta_{\{x,x'\}}\pi_{x'},$$

where $\theta_{\{x,x'\}} = \theta_{\{x',x\}}$ under the reversible assumption.

The dimension of $\boldsymbol{w}$ denoted as P is equal to the total number of *univariate features* M_{statio} and *bivariate features* M_{bi} for any unordered pair of states under GTR. The total number of states is $|\mathcal{X}|$. The univariate features for the π-exponential family are used to define the stationary distribution, where we have the constraint that $\sum_{x\in\mathcal{X}} \pi_x = 1$. Thus,

$$M_{\text{statio}} = |\mathcal{X}| - 1,\ M_{\text{bi}} = |\mathcal{X}^{\text{unordered,dist.}}| = \frac{|\mathcal{X}|(|\mathcal{X}| - 1)}{2},\ P = M_{\text{statio}} + M_{\text{bi}}.$$

Under our Bayesian reversible matrix GLMs configuration,

$$\theta_{\{x,x'\}}(\boldsymbol{w}) \coloneqq \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\phi}(\{x, x'\})\rangle\right\}. \tag{8.1}$$

We pick an arbitrary state x^* as a reference such that

$$\pi_{x^*}(\boldsymbol{w}) = \frac{1}{1 + \sum_{x'\neq x^*: x'\in\mathcal{X}} \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\psi}(x')\rangle\right\}}, \tag{8.2}$$

hence, for any other state x',

$$\pi_{x'}(\boldsymbol{w}) = \exp\left\{\langle \boldsymbol{w}, \boldsymbol{\psi}(x')\rangle\right\}\pi_{x^*}. \tag{8.3}$$

To simplify notations, we order the first M_{statio} elements of $\boldsymbol{w}$ to be the weights for univariate features. Thus, we have

$$\pi_{x^*}(\boldsymbol{w}) = \frac{1}{1 + \sum_{i=1}^{|\mathcal{X}|-1} \exp(\boldsymbol{w}_i)}. \tag{8.4}$$

For any $x' \neq x$, define

$$k_{x'} = \arg_{1 \leqslant j \leqslant M_{\text{statio}}} [\mathbf{1}\{\boldsymbol{\psi}(x') = e_j\} = 1],$$

$$\pi_{x'}(\boldsymbol{w}) = \pi_{x^*}(\boldsymbol{w}) \exp(\boldsymbol{w}_{k_{x'}}), \tag{8.5}$$

where $(\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_p)$ forms a basis of $\mathbb{R}^P$. Given any state $x' \in \mathcal{X}$, $k_{x'}$ is the index such that the $k_{x'}$th element of $\boldsymbol{w}$ represents the weight for univariate feature x'.

We use the $(M_{\text{statio}} + 1)$th to the $(M_{\text{statio}} + M_{\text{bi}})$th elements of $\boldsymbol{w}$ to calculate the exchangeable coefficients with

$$k_{x,x'} = \arg_{M_{\text{statio}}+1 \leqslant j \leqslant M_{\text{statio}}+M_{\text{bi}}} [\mathbf{1}\{\boldsymbol{\phi}(x, x') = e_j\} = 1],$$

$$\theta_{\{x,x'\}}(\boldsymbol{w}) = \exp(\boldsymbol{w}_{k_{x,x'}}). \tag{8.6}$$

Similarly, $k_{x,x'}$ is the index such that the $k_{x,x'}$th element of $\boldsymbol{w}$ corresponds to the bivariate feature $\{x, x'\}$ used to define $\theta_{\{x,x'\}}(\boldsymbol{w})$, where $x \neq x'$.

8.2 Proof of equivalence under GTR configuration

Our goal is to show a Normal prior $f_{\text{nor}}(\boldsymbol{w})$ on $\boldsymbol{w}$ under GTR configuration within our framework is equivalent to a log-Normal on the exchangeable coefficients $\theta_{\{x,x'\}}$ and a logistic-Normal prior on the stationary distribution π_x if Q is parameterized by $\theta_{\{x,x'\}}$ and π_x instead of $\boldsymbol{w}$. To achieve this, we only need to show after a variable transformation from $\boldsymbol{w}$ under a Normal prior to $\theta_{\{x,x'\}}$ and π_x parameterization under GTR, the prior for $\theta_{\{x,x'\}}$ and π_x is log-Normal and logistic-Normal, respectively. We first present the definition of multivariate log-Normal and logistic-Normal distribution. Then we summarize our result in Theorem 8.1.

Definition 8.1 (Section 2, Tarmast (2001)). *Let $\boldsymbol{X} = (X_1, X_2, \ldots, X_p)$ be a p-component random vector having a multivariate Normal distribution with mean $\boldsymbol{v}$ and covariance matrix $B = (b_{ij})$, denoted as $mlnN(\boldsymbol{v}, B)$. We use the transformation $Y_i = \exp(X_i)$ and define $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_p)$, $\boldsymbol{y} = (y_1, y_2, \ldots, y_p)$ is a realization of $\boldsymbol{Y}$. The density of $\boldsymbol{Y}$ is multivariate log-Normal distribution and has the following form:*

$$f_{\boldsymbol{Y}}(\boldsymbol{y}) = (2\pi)^{-\frac{p}{2}} |B|^{-\frac{1}{2}} \boldsymbol{y}^{-1} \cdot \exp\left(-(\log \boldsymbol{y} - \boldsymbol{v})^T B^{-1} (\log \boldsymbol{y} - \boldsymbol{v})/2\right),$$

where $0 < y_i < \infty$, $\log \boldsymbol{y} = (\log y_1, \log y_2, \ldots, \log y_p)$ and $y_i = \exp(x_i)$.

Definition 8.2 (Definition, Atchison and Shen (1980)). *Let $\mathbb{R}^d$ denote d-dimensional real space, $\mathbb{P}^d$ be the positive orthant of $\mathbb{R}^d$ and $\mathbb{G}^d$ be the d-dimensional positive simplex defined by*

$$\mathbb{G}^d = \{\boldsymbol{\mu} \in \mathbb{P}^d : u_1 + \ldots + \mu_d < 1\}$$

The logistic transformation from $\mathbb{R}^d$ to $\mathbb{G}^d$ or its inverse log ratio transformation:

$$\boldsymbol{u} = e^{\boldsymbol{r}} / \left(1 + \sum_{j=1}^{d} e^{r_j}\right) \qquad \text{and} \qquad \boldsymbol{r} = \log(\boldsymbol{u}/u_{d+1}),$$

where $u_{d+1} = 1 - \sum_{j=1}^{d} u_j$*, can be used to define a logistic-Normal distribution over* $\mathbb{G}^d$ *and we can say that* $\boldsymbol{u}$ *is* $L_d(\mu, \Sigma)$*, assuming* $\boldsymbol{r}$ *follows the multivariate Normal distribution* $N_d(\boldsymbol{\mu}, \Sigma)$*. The density function of* $L_d(\boldsymbol{\mu}, \Sigma)$ *is*

$$|2\pi\Sigma|^{-\frac{1}{2}} \left(\prod_{j=1}^{d+1} u_j\right)^{-1} \exp\left(-\frac{1}{2}[\log(\boldsymbol{u}/u_{d+1}) - \boldsymbol{\mu}]^T \Sigma^{-1} [\log(\boldsymbol{u}/u_{d+1}) - \boldsymbol{\mu}]\right).$$

Theorem 8.1. *Following the configuration of Section 8.1, if we assign* $f_{nor}(\boldsymbol{w}) = N(\mathbf{0}, \Sigma)$ *on* $\boldsymbol{w} = (w_1, \ldots, w_P)$*, via the transformation defined in Equation (8.4), (8.5) and (8.6), the prior of* $\theta_{\{x,x'\}}$ *and* π_x *is a multivariate log-Normal distribution and a logistic-Normal distribution, where* $\Sigma_{i,i} = \sigma_i^2$*,* $\Sigma_{i,j} = 0$ *for* $j \neq i$*, for any* $\{x, x'\} \in \mathcal{X}^{unordered,dist.}$*,* $x \in \mathcal{X}$*,* $i = 1, 2, \ldots, P$*, and* $P = M_{statio} + M_{bi}$*. We assume elements in* $\boldsymbol{w}$ *are ordered such that the first* M_{statio} *elements correspond to univariate features and the rest correspond to bivariate features.*

Proof. Denote the likelihood function given the observations $\boldsymbol{y}$ and the weights $\boldsymbol{w}$ as $f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(\boldsymbol{w}))$, the posterior distribution of $\boldsymbol{w}$ given $\boldsymbol{y}$ as $f_{\mathrm{w|y}}(\boldsymbol{w}|\boldsymbol{y})$. The value of the likelihood function does not change with different parameterizations, hence,

$$f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(\boldsymbol{w})) = f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(\theta_{\{x,x'\}}, \pi_x))$$

via the transformation defined in Equation (8.4), (8.5) and (8.6), where $Q^{(\mathrm{rev})}(\theta_{\{x,x'\}}, \pi_x)$ represents Q is parameterized by $\theta_{\{x,x'\}}$ and π instead of $\boldsymbol{w}$.

Based on the independence of w_i and w_j, for any $i \neq j$, denote $f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(w_j))$ as the marginal likelihood which has integrated over all other elements of $\boldsymbol{w}$ except the jth element.

Consider the bivariate features, that is for any j, if $M_{\mathrm{statio}} + 1 \leqslant j \leqslant M_{\mathrm{statio}} + M_{\mathrm{bi}}$,

$$f_{\mathrm{w_j|y}}(w_j|\boldsymbol{y}) = \frac{\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{w_j^2}{2\sigma_j^2}\right) f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(w_j))}{\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{w_j^2}{2\sigma_j^2}\right) f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(w_j)) dw_j} \tag{8.7}$$

According to Equation (8.6), without loss of generality, assuming $k_{x,x'} = j$, we have

$$\theta_{\{x,x'\}}(\boldsymbol{w}) = \exp(w_{k_{x,x'}}) = \exp(w_j). \tag{8.8}$$

Note that $\theta_{\{x,x'\}} = \exp(w_j)$ is a variable transformation of w_j. Hence,

$$f_{\theta_{\{x,x'\}}|y}(\theta_{\{x,x'\}}|\boldsymbol{y}) = \frac{\frac{1}{\sqrt{2\pi}\sigma_j\theta_{\{x,x'\}}} \exp\left(-\frac{\log\left(\theta_{\{x,x'\}}\right)^2}{2\sigma_j^2}\right) f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(\theta_{\{x,x'\}}))}{\int_0^{\infty} \frac{1}{\sqrt{2\pi}\sigma_j\theta_{\{x,x'\}}} \exp\left(-\frac{\log\left(\theta_{\{x,x'\}}\right)^2}{2\sigma_j^2}\right) f_{\mathrm{y|Q}}(\boldsymbol{y}|Q^{(\mathrm{rev})}(\theta_{\{x,x'\}})) d\theta_{\{x,x'\}}}. \tag{8.9}$$

Equation (8.7) and (8.9) show that if the prior on w_j is $N(0, \sigma_j^2)$, the prior of $\theta_{\{x,x'\}} = \exp(w_j)$ follows $lnN(0, \sigma_j^2)$ according to Definition 8.1 of log-normal distribution in one dimension.

Regarding the bivariate features, we conclude that $\theta_{\{x,x'\}}$, for any $x, x' \in \mathcal{X}^{\text{unordered,dist.}}$, is independent with each other and they follow a multivariate log-Normal distribution $mlnN(\mathbf{0}, B)$ based on Definition 8.1, where B is a diagonal matrix with $B_{i,i} = (\sigma_i^2)$, σ_i^2s are the $(M_{\text{statio}} + 1)$th to the $(M_{\text{statio}} + M_{\text{bi}})$th diagonal elements from Σ of $f_{\text{nor}}(\boldsymbol{w})$.

If w_{statio} follows a Normal distribution, where $w_{\text{statio}} = (w_1, w_2, \ldots, w_{|\mathcal{X}|-1})$ corresponds to the univariate features defining the stationary distribution, Σ^* is a diagonal matrix with $\Sigma^*_{ii} = \sigma_i^2$, where σ_i^2s are the first M_{statio} diagonal elements from Σ of $f_{\text{nor}}(\boldsymbol{w})$, following Equation (8.4), (8.5), and Definition 8.2, the prior of π_x is $L_d(\mathbf{0}, \Sigma^*)$ since the transformation from w_i to π_x is a logistic transformation and the original weights $\boldsymbol{w}$ are normally distributed. □

9 Supplementary figures

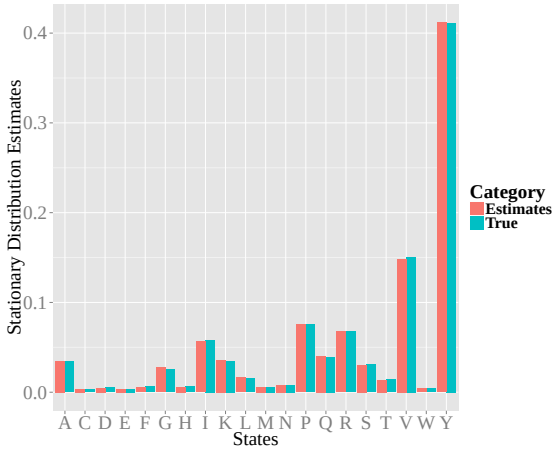

Figure 1: Actual stationary distributions are compared with estimates from a 415 sites and 641 sequences synthetic dataset generated using the PolaritySizeGtr model. We generate unbalanced stationary distributions to increase the difficulty of reconstruction.

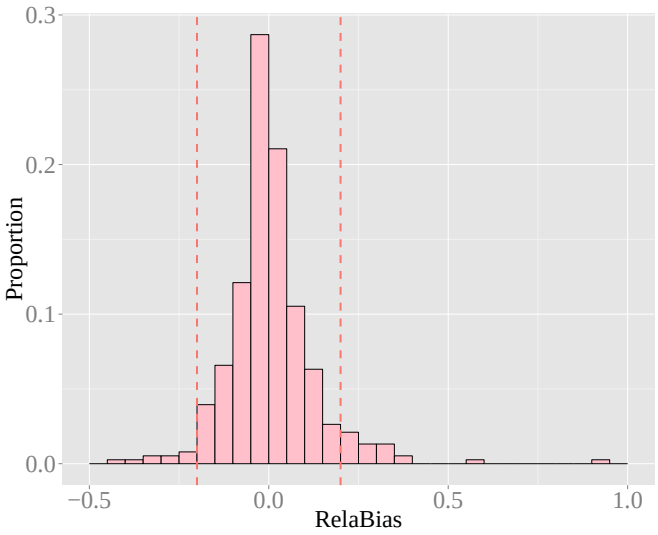

Figure 2: Histogram of the relative biases (described in Section 7.3) of the 400 transition probability matrix (excluding the redundant diagonal elements) under $t = 2$. The two dotted lines label -0.2 and 0.2.

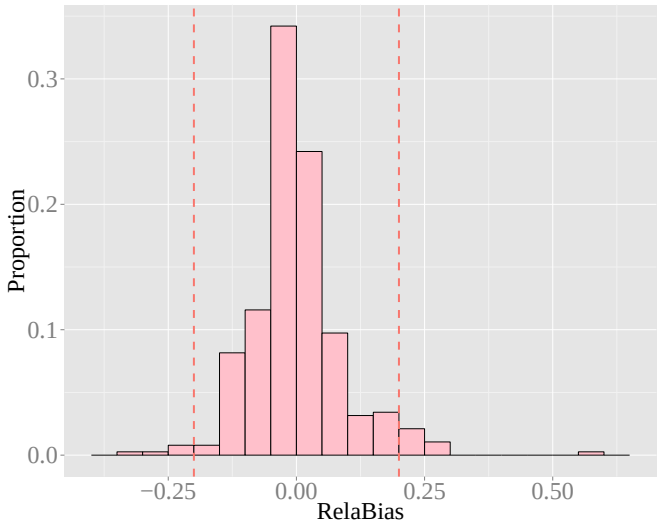

Figure 3: Histogram of the relative biases (described in Section 7.3) of the 400 transition probability matrix (excluding the redundant diagonal elements) under $t = 3$. The two dotted lines label -0.2 and 0.2.

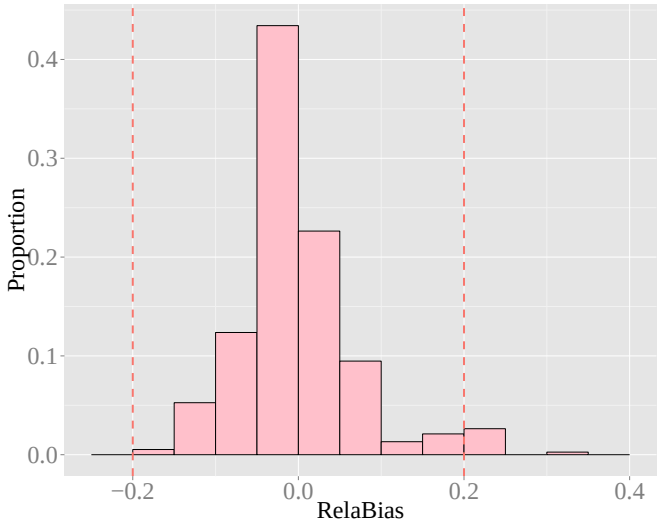

Figure 4: Histogram of the relative biases (described in Section 7.3) of the 400 transition probability matrix (excluding the redundant diagonal elements) under $t = 5$. The two dotted lines label -0.2 and 0.2.

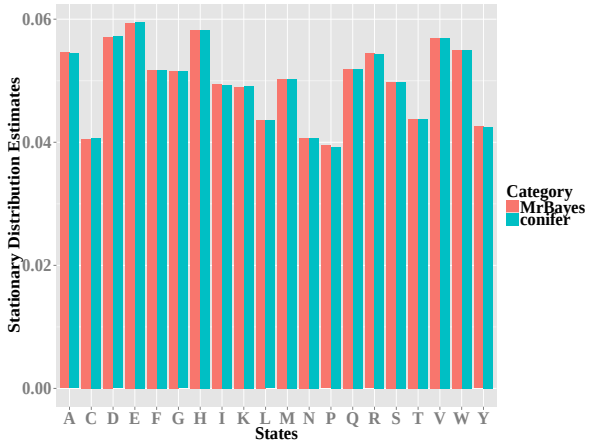

Figure 5: Estimates of the stationary distributions from MrBayes and our model, both set to their respective GTR settings.

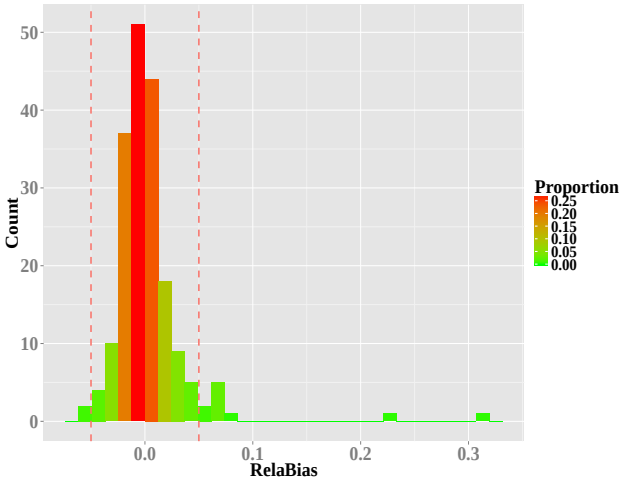

Figure 6: Histogram of the relative differences of the estimates of the 190 exchangeable coefficients between MrBayes and our model, both set to their respective GTR settings. The differences are generally within 0.05; the two outliers greater than 0.1 are between A to R and Q to P.

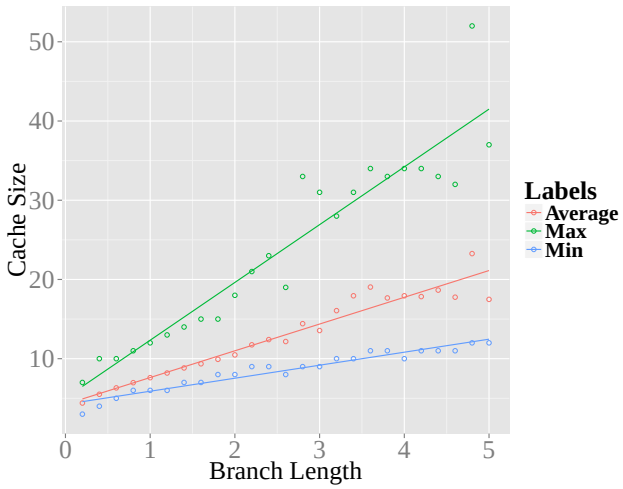

Figure 7: The relationship between cache sizes and branch lengths using pairs of DNA sequences with 5000 sites under GTR with branch lengths from 0.2 to 5 with step size 0.2. "Average" represents the averaged cache size for each dataset across 100,000 MCMC iterations. "Min" and "Max" represent the minimum and maximum of the cache size.

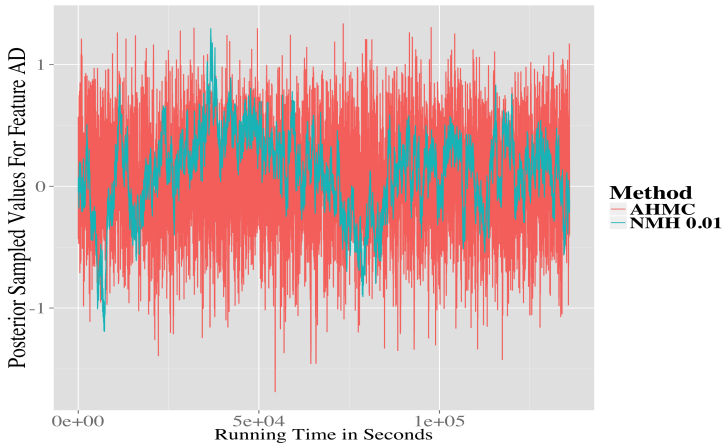

Figure 8: Trace plots for NMH with bandwidth 0.01 and our AHMC sampler, both ran within the same wall-clock time limit of 2267 minutes. After discarding the burn-in period of the first 50,000 iterations, there are 450,000 iterations for the NMH method and 229,520 iterations for our AHMC sampler. The ordinate axis represents the sampled values for the feature "AD." Other features exhibit the same qualitative behaviour.

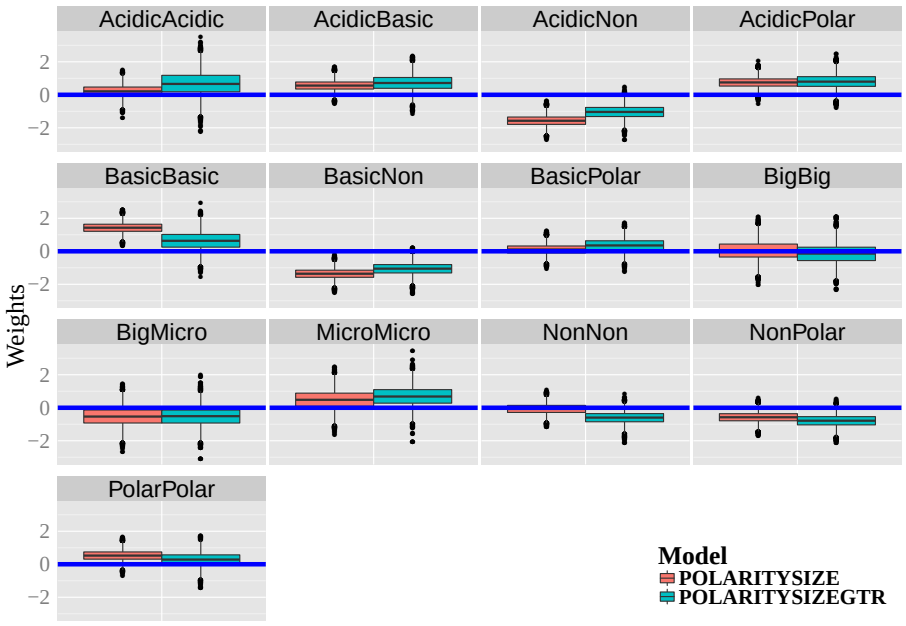

Figure 9: Estimates of the weights for polarity/charge, and size features. The legend specifies the underlying model following the naming convention defined in Section 7 of the main text.

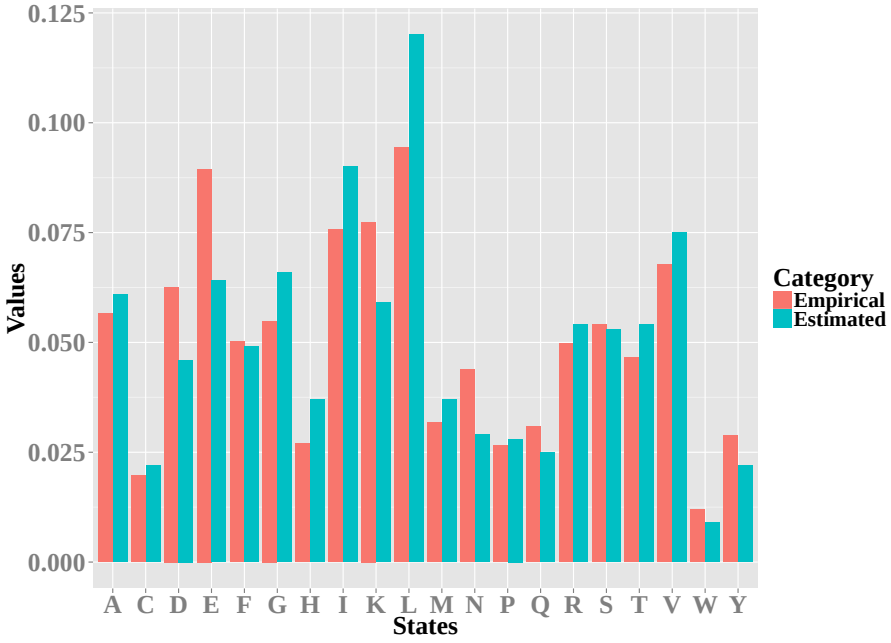

Figure 10: Stationary distributions on the protein kinase domain family data. "Estimated" refers to the stationary distribution based on the posterior mean of the univariate weights in our Bayesian model. "Empirical" refers to the empirical frequency counts, which is sometimes used in lieu of estimates derived from CTMC models (e.g., Cao et al. (1994)). Note that even though the two estimates are broadly similar, they also quantitatively differ in important aspects.

10 Supplementary tables

AcidicAcidic	AcidicBasic	AcidicPolar	BasicBasic	BasicPolar	PolarPolar
0.226	-0.711	0.440	-0.246	0.496	1.332
NonNon	AcidicNon	BasicNon	NonPolar	BigBig	BigMicro
-2.366	0.065	-0.010	-1.388	-1.215	-0.677
MicroMicro	statio(A)	statio(C)	statio(D)	statio(E)	statio(F)
-0.902	-0.685	-0.577	-0.863	-0.048	0.012
statio(G)	statio(H)	statio(I)	statio(K)	statio(L)	statio(M)
-0.455	-1.305	-0.269	-0.199	-0.136	-0.932
statio(N)	statio(P)	statio(Q)	statio(R)	statio(S)	statio(T)
-1.431	0.194	-0.629	-0.901	-0.526	-0.645
statio(V)	statio(W)	statio(Y)			
0.360	0.049	0.137			

Table 1: Geweke diagnostic test statistic values under the PolaritySizeGtr model for datasets where each sequence has 100 sites. Under the 5% significance level, an absolute value larger than 2 supports a lack of convergence for that parameter.

AcidicAcidic	AcidicBasic	AcidicPolar	BasicBasic	BasicPolar	PolarPolar
-0.365	-0.400	1.414	1.565	-0.438	1.049
NonNon	AcidicNon	BasicNon	NonPolar	BigBig	BigMicro
1.613	0.041	0.783	0.868	-0.559	-0.751
MicroMicro	statio(A)	statio(C)	statio(D)	statio(E)	statio(F)
-1.270	-0.644	-0.321	-0.090	-0.219	-0.204
statio(G)	statio(H)	statio(I)	statio(K)	statio(L)	statio(M)
-0.270	-0.300	-0.450	-0.161	-0.207	-0.525
statio(N)	statio(P)	statio(Q)	statio(R)	statio(S)	statio(T)
-0.473	-0.413	-0.008	-0.406	-0.315	-0.274
statio(V)	statio(W)	statio(Y)			
-0.587	-0.067	-0.377			

Table 2: Geweke diagnostic test statistic values under the PolaritySizeGtr model for datasets where each sequence has 1500 sites. Under the 5% significance level, an absolute value larger than 2 supports a lack of convergence for that parameter.

Method	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
AHMC	**24.56**	**41.42**	**50.32**	**86.01**	**67.90**	**432.20**
Adaptive NMH	6.32	30.91	40.07	41.84	52.45	93.34
NMH 0.002	0.17	0.53	0.97	1.54	1.53	17.77
NMH 0.005	0.51	1.70	2.47	4.89	3.99	74.39
NMH 0.01	1.44	6.97	10.88	20.25	18.62	244.20
NMH 0.0125	0.71	3.53	5.65	9.86	9.25	117.70
NMH 0.015	0.80	4.01	6.49	11.24	10.78	128.10
NMH 0.018	1.23	6.77	11.05	19.03	18.29	226.30
NMH 0.02	1.60	7.54	11.82	20.98	20.80	245.50
NMH 0.03	1.32	3.97	6.16	11.69	10.28	162.90
NMH 0.04	0.40	1.44	2.18	3.61	3.73	38.49
NMH 0.05	0.20	0.54	0.78	0.97	1.07	6.02
NMH 0.06	0.36	0.81	1.08	1.25	1.44	6.56
NMH 0.07	0.28	0.65	0.83	0.89	1.06	2.47
NMH 0.08	0.31	0.73	0.95	1.04	1.25	3.39
NMH 0.09	0.25	0.53	0.67	0.77	0.93	2.78
NMH 0.10	0.55	0.79	0.90	0.97	1.08	3.03
NMH 0.15	0.20	0.26	0.30	0.36	0.41	0.81
NMH 0.20	0.27	0.31	0.35	0.38	0.43	0.63

Table 3: ESS per 10^4 seconds. NMH x stands for a Normal proposal Metropolis Hastings algorithms with proposal bandwidth x.

Method	Min	1st Quantile	Median	Mean	3rd Quantile	Max
Aux	**20**	**165**	**295**	**551**	**484**	**8698**
No Aux	3.27	8.62	12.1	56.5	18.2	681

Table 4: Summary of the ESS per 10^6 seconds for the stationary distribution and exchangeable coefficients with auxiliary variable and without, denoted as "Aux" and "No Aux" respectively, with estimates inferred under a PolaritySizeGtr model with fixed $\epsilon = 5 \times 10^{-7}$ and $L = 30$, while fixing the topology and branch lengths to the true one with a simulated amino acid dataset of 10 leaves and 5000 sites generated under a PolaritySizeGtr model.

References

Atchison, J. and Shen, S. M. (1980). "Logistic-normal distributions: Some properties and uses." *Biometrika*, 67(2): 261–272.

Cao, Y., Adachi, J., Janke, A., Pääbo, S., and Hasegawa, M. (1994). "Phylogenetic relationships among eutherian orders estimated from inferred sequences of mitochondrial proteins: instability of a tree based on a single gene." *Journal of Molecular Evolution*, 39(5): 519–527.

Carpenter, J., Clifford, P., and Fearnhead, P. (1999). "An Improved Particle Filter for Non-linear Problems." In *IEEE Proceedings: Radar, Sonar and Navigation*, volume 146, 2–7.

Engel, Y. (2005). "Algorithms and representations for reinforcement learning." *Doktorarbeit, The Hebrew University of Jerusalem*.

Hasegawa, M., Kishino, H., and Yano, T. (1985). "Dating of the human-ape splitting

by a molecular clock of mitochondrial DNA." *Journal of Molecular Evolution*, 22(2): 160–174.

Hobolth, A. and Stone, E. A. (2009). "Simulation from endpoint-conditioned, continuous-time Markov chains on a finite state space, with applications to molecular evolution." *The Annals of Applied Statistics*, 3(3): 1204.

Hoffman, M., Brochu, E., and de Freitas, N. (2011). "Portfolio Allocation for Bayesian Optimization." In *Uncertainty in Artificial Intelligence*, 327–336.

Kenney, T. and Gu, H. (2012). "Hessian Calculation for Phylogenetic Likelihood based on the Pruning Algorithm and its Applications." *Statistical Applications in Genetics and Molecular Biology*, 11: 1544–6115.

Mockus, J. (1982). "The Bayesian Approach to Global Optimization." In *System Modeling and Optimization*, volume 38, 473–481. Springer.

Moler, C. and Van Loan, C. (1978). "Nineteen dubious ways to compute the exponential of a matrix." *SIAM review*, 20(4): 801–836.

Rao, V. and Teh, Y. W. (2013). "Fast MCMC sampling for Markov jump processes and extensions." *The Journal of Machine Learning Research*, 14(1): 3295–3320.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts: MIT Press.

Roberts, G. O. and Rosenthal, J. S. (2007). "Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms." *Journal of applied probability*, 44(2): 458–475.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. (2010). "Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design." In *International Conference on Machine Learning*.

Tarmast, G. (2001). "Multivariate Log–Normal Distribution." *ISI Proceedings: Seoul 53rd Session*.

Wang, Z., Mohamed, S., and de Freitas, N. (2013). "Adaptive Hamiltonian and Riemann Manifold Monte Carlo Samplers." In *International Conference on Machine Learning (ICML)*, 1462–1470. JMLR W&CP 28 (3): 14621470, 2013.