

Adaptive dating and faster proposals: revisiting the phylogenetic relaxed clock model

Jordan Douglas^{1,2*}, Rong Zhang^{1,2}, Alexei J. Drummond^{1,2,3}, Remco Bouckaert^{1,2}

1 Centre for Computational Evolution, University of Auckland, Auckland, New Zealand

2 School of Computer Science, University of Auckland, Auckland, New Zealand

3 School of Biological Sciences, University of Auckland, Auckland, New Zealand

* jordan.douglas@auckland.ac.nz

Abstract

Author summary

Introduction

The molecular clock hypothesis states that the evolutionary rates of biological sequences are approximately constant throughout time [1]. This assumption forms the basis of phylogenetics, under which the evolutionary trees and divergence dates of life forms are inferred from biological sequences, such as nucleotides and amino acids [2, 3]. In Bayesian phylogenetics, these trees and their associated parameters are estimated as probability distributions [4–6]. Statistical inference is typically performed by sampling using the Markov chain Monte Carlo (MCMC) algorithm [7, 8]. Bayesian phylogenetic clock models have been implemented in many platforms including BEAST [9], BEAST2 [10], MrBayes [11], and RevBayes [12].

The simplest phylogenetic clock model – the strict clock – makes the convenient assumption that the evolutionary rate is constant across all lineages [4, 5, 13]. However, molecular substitution rates are known to vary over time, over population sizes, over evolutionary pressures, and over nucleic acid replicative machineries [14–16]. Moreover, any given dataset may be clock-like (where the substitution rate has a small variance across lineages) or non clock-like (a large variance). In the latter case, a strict clock is probably not suitable.

This led to the development of relaxed (uncorrelated) clock models, under which each branch in the phylogenetic tree has its own molecular substitution rate [3]. Branch rates can be drawn from a range of probability distributions including Log-Normal, Exponential, Gamma, and Inverse-Gamma distributions [3, 17, 18]. This class of models is widely used, and has aided insight into many recent biological problems, including the 2016 Zika virus outbreak [19] and the COVID-19 pandemic [20].

Finally, although not the focus of this article, the class of correlated clock models assumes some form of auto-correlation between rates over time. The correlation itself can invoke a range of stochastic models, including compound Poisson [21] and CIR processes [17], or it can exist as a series of local relaxed clocks [22]. However, due to the correlated and discrete nature of such models, the time until MCMC convergence may be cumbersome, particularly for larger datasets [22].

With the increasing availability of biological sequence data, the development of efficient Bayesian phylogenetic methods is more important than ever. The performance

of MCMC is dependent not only on computational runtime but also the efficacy of an MCMC setup to achieve its convergence. A critical task therein lies the further advancement of MCMC operators. Recent developments in this area include the advancement of guided tree proposals [23–25], coupled MCMC [26, 27], adaptive multivariate transition kernels [28], and other explorative proposal kernels such as the Bactrian and mirror kernels [29, 30]. In the case of clock models, informed tree proposals can account for correlations between substitution rates and divergence times [31]. The rate parameterisation itself can also affect the ability to “mix” during MCMC [3, 18, 31].

While a range of advanced operators and other MCMC optimisation methods have arisen over the years, there has yet to be a widescale performance benchmarking of such methods as applied to the relaxed clock model. In this article, we systematically evaluate how the relaxed clock model can benefit from i) adaptive operator weighting, ii) different substitution rate parameterisations, iii) the use of Bactrian proposal kernels [29], iv) tree operators which account for correlations between substitution rates and times, and v) adaptive multivariate operators [28]. The discussed methods are implemented in and compared using BEAST2 [10].

Models

Preliminaries

Let \mathcal{T} be a binary rooted time tree with N taxa. Let L be the number of sites within the multiple sequence alignment D , and let L_{eff} be the effective number of sites in the alignment (i.e. the number of unique patterns across all sites in the alignment). The posterior density of a phylogenetic model is described by

$$p(\mathcal{T}, \vec{\mathcal{R}}, \sigma, \theta | D) \propto p(D | \mathcal{T}, r(\vec{\mathcal{R}}), \theta) p(\mathcal{T} | \theta) p(\vec{\mathcal{R}} | \sigma) p(\sigma) p(\theta). \quad (1)$$

σ represents clock model related parameters, and $p(\mathcal{T} | \theta)$ is the tree prior where θ describes further unspecified parameters. The tree likelihood $p(D | \mathcal{T}, r(\vec{\mathcal{R}}), \theta)$ is computed using the tree-peeling algorithm [32], where $\vec{\mathcal{R}}$ is a vector of abstracted substitution rates and transformed into real rates by function $r(\vec{\mathcal{R}})$. Three methods of representing rates as $\vec{\mathcal{R}}$ are presented in **Substitution rate parameterisations**.

Let t_i be the height (time) of node i . Each node i in \mathcal{T} , except for the root, is associated with a parental branch length τ_i (the height difference between i and its parent) and a parental branch substitution rate $r_i = r(\mathcal{R}_i)$. In a relaxed clock model, each of the $2N - 2$ elements in $\vec{\mathcal{R}}$ are independently distributed under the prior $p(\vec{\mathcal{R}} | \sigma)$.

The posterior distribution is sampled by the Metropolis-Hastings-Green MCMC algorithm [7, 8, 33], under which the probability of accepting proposed state x' from state x is equal to:

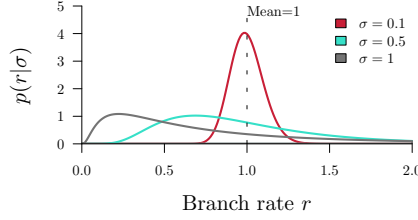
$$\alpha(x' | x) = \min \left(1, \frac{p(x' | D)}{p(x | D)} \frac{q(x | x')}{q(x' | x)} |J| \right). \quad (2)$$

$q(a | b)$ is the transition kernel: the probability of proposing state b from state a . The ratio between the two $\frac{q(x | x')}{q(x' | x)}$ is also known as the Hastings ratio [8]. The determinant of the Jacobian matrix $|J|$ solves the dimension-matching problem for proposals which operate on multiple terms across one or more spaces [33, 34]. This term is known as the Green ratio.

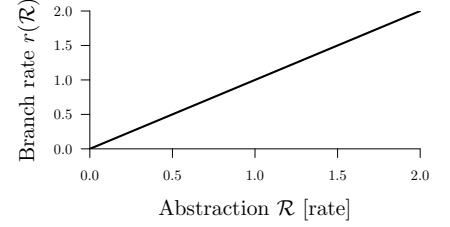
Substitution rate parameterisations

In Bayesian inference, the way parameters are represented in the model can affect the mixing ability of the model and the meaning of the model itself [35]. Three methods for parameterising substitution rates are described below. Each parameterisation is associated with i) an abstraction of the branch rate vector $\vec{\mathcal{R}}$, ii) some function for transforming this parameter into unabridged branch rates $r(\vec{\mathcal{R}})$, and iii) a prior density function of the abstraction $p(\vec{\mathcal{R}}|\sigma)$. The three methods are summarised in **Fig 2**.

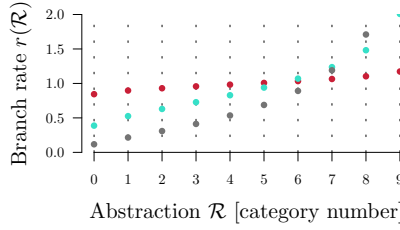
Probability density function (Log-Normal)



real



cat with 10 bins



quant

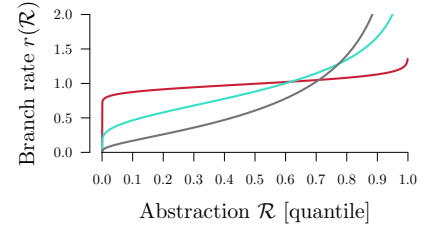


Fig 1. Branch rate parameterisations. Top left: the prior density of a branch rate r under a $\text{Log-Normal}(-0.5\sigma^2, \sigma)$ distribution (with its mean fixed at 1). The function for transforming \mathcal{R} into branch rates $r(\mathcal{R})$ is depicted for *real* (top right), *cat* (bottom left), and *quant* (bottom right). For simplicity there are 10 *cat* bins displayed, however in practice there are $2N - 2$ bins.

1. Real rates

The natural (and unabstracted) parameterisation of a substitution rate is a real number $\mathcal{R}_i \in \mathbb{R}, \mathcal{R}_i > 0$ which is equal to the rate itself. Thus, under the *real* parameterisation:

$$r(\vec{\mathcal{R}}) = \vec{\mathcal{R}}. \quad (3)$$

Under the prior distribution $p(\vec{\mathcal{R}}|\sigma)$, rates are Log-Normally distributed with a mean of 1:

$$p(\mathcal{R}_i|\sigma) = \frac{1}{\mathcal{R}_i\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln \mathcal{R}_i - \mu)^2}{2\sigma^2}\right) \quad (4)$$

where $\mu = -0.5\sigma^2$ is set such that the expected value of the Log-Normal distribution is 1. In this article we only consider Log-Normal clock priors, however the methods discussed are general.

Zhang and Drummond 2020 introduced a series of tree operators which propose node heights and branch rates, such that the resulting genetic distances ($r_i \times \tau_i$) remain constant [31]. These operators account for correlations between branch rates and branch times. By keeping the genetic distance of each branch constant, the likelihood is unaltered by the proposal.

2. Categories

The category parameterisation (*cat*) is an abstraction of the *real* parameterisation. Each of the $2N - 2$ branches are assigned an integer from 0 to $n - 1$:

$$\vec{\mathcal{R}} \in \{0, 1, \dots, n - 1\}^{2N-2}. \quad (5)$$

These integers correspond to n rate categories (**Fig. 2**). The domain of $\vec{\mathcal{R}}$ is uniformly distributed under the prior:

$$p(\mathcal{R}_i|\sigma) = p(\mathcal{R}_i) = \frac{1}{n}. \quad (6)$$

Let $f(x|\sigma)$ be the probability density function (PDF) and let $F(x|\sigma) = \int_0^x f(t|\sigma) dt$ be the cumulative distribution function (CDF) of the prior distribution used by the underlying *real* clock model. Then, in the *cat* parameterisation, $f(x|\sigma)$ is discretised into n bins and each element within $\vec{\mathcal{R}}$ points to one such bin, where each bin has uniform prior density. The rate of each bin is equal to the median value within the bin

$$r(\mathcal{R}_i) = F^{-1}\left(\frac{\mathcal{R}_i + 0.5}{n}\right), \quad (7)$$

where F^{-1} is the inverse cumulative distribution function (i-CDF).

The key advantage of the *cat* parameterisation is the removal of a term from the posterior density (Equation 1), or more accurately the replacement of a non-trivial $p(\vec{\mathcal{R}}|\sigma)$ term with that of a uniform prior. This may facilitate efficient traversal of the parameter space by MCMC.

This parameterisation has been widely used in BEAST and BEAST2 analyses [3]. However, the recently developed constant distance operators – which are incompatible with the *cat* parameterisation – can yield an increase in mixing rate under *real* by up to an order of magnitude over that of *cat*, depending on the dataset [31].

3. Quantiles

Finally, rates can be parameterised as real numbers $0 < \mathcal{R}_i < 1$ which describe the rate's quantile with respect to some underlying clock model distribution. Under the *quant* parameterisation, each of the $2N - 2$ elements in $\vec{\mathcal{R}}$ are uniformly distributed.

$$\vec{\mathcal{R}} \in \mathbb{R}^{2N-2}, 0 < \mathcal{R}_i < 1 \quad (8)$$

$$p(\mathcal{R}_i|\sigma) = p(\mathcal{R}_i) = 1 \quad (9)$$

Transforming these quantiles into rates invokes the i-CDF of the underlying *real* clock model distribution:

$$r(\mathcal{R}_i) = F^{-1}(\mathcal{R}_i). \quad (10)$$

While this approach has clear similarities with *cat*, the domain of rates here is continuous (as opposed to being confined to a discrete number of bins). In this project we extended the family of constant distance operators [31] so they are compatible with *quant*. These operators are presented in **S1 Appendix**.

The computational requirements of i-CDF evaluation may hinder the performance of the *quant* parameterisation.

Adaptive operator weighting

The weight of an operator determines the probability of the operator being selected at each step during MCMC. In BEAST2, operators can have a tunable parameter s which determines the step size of the operator [10]. Although s is learned throughout the MCMC chain, the operator weights themselves are typically held constant. Pre-existing BEAST2 clock model operators (i.e. those which generate proposals for either $\vec{\mathcal{R}}$ or σ) are summarised in **Table 1**, and further operators are introduced throughout the paper.

Operator	Description	Parameters	Parameterisations
RandomWalk	Moves a single element by a tunable amount.	$\vec{\mathcal{R}}, \sigma$	<i>cat, real, quant</i>
Scale	Applies RandomWalk on the log-transformation (suitable for parameters with positive domains).	$\vec{\mathcal{R}}, \sigma$	<i>real, quant</i>
Interval	Applies RandomWalk on the logit-transformation (suitable for parameters with upper and lower limits).	$\vec{\mathcal{R}}$	<i>quant</i>
Swap	Swaps two random elements in the vector [3].	$\vec{\mathcal{R}}$	<i>cat, real, quant</i>
Uniform	Resamples one element in the vector from a uniform distribution.	$\vec{\mathcal{R}}$	<i>cat, quant</i>
ConstantDistance	Adjusts an internal node and recalculates all incident branch rates such that the genetic distances remain constant [31].	$\vec{\mathcal{R}}, \mathcal{T}$	<i>real, quant</i>
SimpleDistance	Applies ConstantDistance to the root node [31].	$\vec{\mathcal{R}}, \mathcal{T}$	<i>real, quant</i>
SmallPulley	Proposes new branch rates incident to the root such that their combined genetic distance is constant [31].	$\vec{\mathcal{R}}$	<i>real, quant</i>
FastClockScale	Applies Scale to σ and then recomputes all rates $\vec{\mathcal{R}}$ such that their quantiles under the prior $p(\vec{\mathcal{R}} \sigma)$ are constant [31].	\mathcal{R}	<i>real</i>

Table 1. Summary of pre-existing BEAST2 operators, which apply to either branch rates $\vec{\mathcal{R}}$ or the clock standard deviation σ , and the substitution rate parameterisation they apply to. **ConstantDistance** and **SimpleDistance** also adjust node heights in the tree \mathcal{T} .

It is not always clear which mixture of operators is best for a given dataset. In this article we introduce **AdaptiveOperatorSampler** – a meta-operator which learns operator weights during MCMC and samples operators according to these weights. The meta-operator undergoes three phases. In the first phase (burn-in),

`AdaptiveOperatorSampler` samples from its set of sub-operators uniformly at random. In the second phase (learn-in), the meta-operator continues to sample operators uniformly at random however it begins learning several terms detailed below. In its final phase, `AdaptiveOperatorSampler` samples operators (denoted by ω) using the following distribution:

$$p(\omega_i) \propto \begin{cases} 1 & \text{with probability } \Omega \\ \frac{1}{\mathbb{T}(\omega_i)} \sum_{p \in \text{parameters}} \frac{1}{\sigma_p^2} \left[\sum_{x \in \text{accepts}_i} (x_p - x'_p)^2 \right] & \text{with probability } 1 - \Omega \end{cases} \quad (11)$$

where $\Omega = 0.01$ allows any sub-operator to be sampled regardless of its performance. The remaining terms are trained during the second and third phases: the cumulative computational runtime spent on each operator $\mathbb{T}(\omega_i)$, the sample variance σ_p^2 of each parameter p , and the sum-of-squares $\sum_{x \in \text{accepts}_i} (x_p - x'_p)^2$, where x_p and x'_p are the values of p before and after each acceptance of a proposal made by ω_i .

Under Equation 11, operators which effect larger changes on the parameters of interest, in shorter runtime, are sampled with greater probabilities. Division of the sum-of-squares term by the parameter variance σ_p^2 enables comparison between different parameters.

We also introduce the `SampleFromPrior`(\vec{x}) operator. This operator resamples ψ randomly selected elements within vector \vec{x} from their prior distributions, where $\psi \sim \text{Binomial}(n = |\vec{x}|, p = \frac{s}{|\vec{x}|})$ for tunable parameter s .

We hypothesise that datasets with strong signal (or large L) will mix best when the more precise and meticulous kind of operator is employed, such as those informed by correlations within the posterior distribution e.g. the family of constant distance operators [31] (**Fig. 2**). Whereas, datasets which contain very poor signal (or small L) are likely to mix best when there is more weight placed on bold operators such as `SampleFromPrior`, `Swap`, or `Uniform` which sample each branch rate independently of its current estimate. `AdaptiveOperatorSampler` should learn the combination of weights behind these classes of operators best suited for any given dataset.

In this article we apply three instances of the `AdaptiveOperatorSampler` meta-operator to the *real* and *quant* parameterisations. These are summarised in **Table 2**.

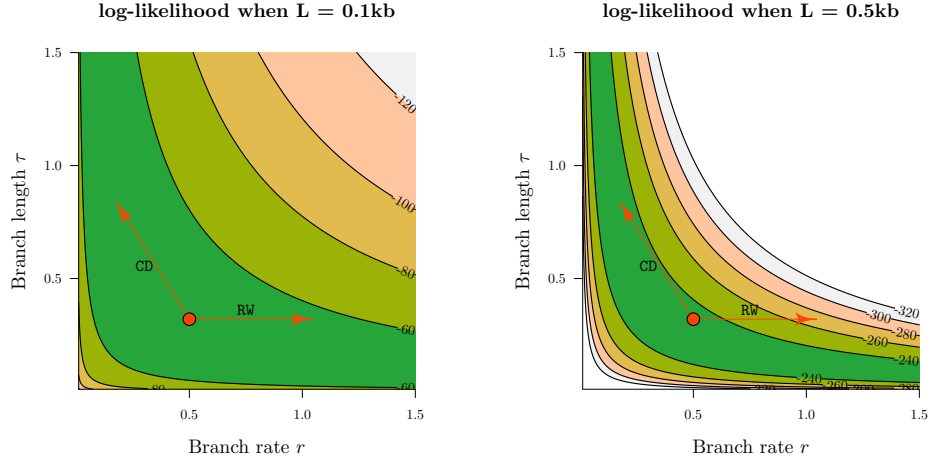


Fig 2. Traversing likelihood space. The z-axes above are the log-likelihoods of the genetic distance $r \times \tau$ between two simulated nucleic acid sequences of length L , under the Jukes-Cantor substitution model [36]. Two possible proposals from the current state (red circle) are depicted. These proposals are generated by the **RandomWalk** (RW) and **ConstantDistance** (CD) operators. In the low signal dataset ($L = 0.1\text{kb}$), both operators can traverse the likelihood space effectively. However, the same exact same proposal by **RandomWalk** incurs a much larger likelihood penalty in the $L = 0.5\text{kb}$ dataset by “falling off the ridge”, in contrast to **ConstantDistance** which “walks along the ridge”. This discrepancy is even stronger for larger datasets and thus necessitates the use of operators such as **ConstantDistance** which account for correlations between branch lengths and rates.

Meta-operator	Operators
AdaptiveOperatorSampler (σ)	FastClockScale ($\sigma, \vec{\mathcal{R}}$)*
	RandomWalk (σ)
	Scale (σ)
	SampleFromPrior (σ)
AdaptiveOperatorSampler ($\vec{\mathcal{R}}$)	ConstantDistance ($\vec{\mathcal{R}}, \mathcal{T}$)
	RandomWalk ($\vec{\mathcal{R}}$)
	Scale ($\vec{\mathcal{R}}$)*
	Interval ($\vec{\mathcal{R}}$)**
	Swap ($\vec{\mathcal{R}}$)
AdaptiveOperatorSampler (root)	SampleFromPrior ($\vec{\mathcal{R}}$)
	SimpleDistance ($\vec{\mathcal{R}}, \mathcal{T}$)
	SmallPulley ($\vec{\mathcal{R}}, t$)

Table 2. Summary of the three instances of **AdaptiveOperatorSampler** used under the *real* and *quant* parameterisations. **real* only. ***quant* only.

Bactrian proposal kernel

The step size of a proposal kernel $q(x'|x)$ should be such that the proposed state x' is sufficiently far from the current state x to explore vast areas of parameter space, but not so large that the proposal is rejected too often [37]. Yang et al. have challenged the widely used uniform proposal kernel in place of the Bactrian kernel [29, 30]. The Bactrian(m) distribution is defined as the sum of two Normal distributions:

$$\Sigma \sim \text{Bactrian}(m) \equiv \frac{1}{2}\text{Normal}(-m, 1 - m^2) + \frac{1}{2}\text{Normal}(m, 1 - m^2) \quad (12)$$

where $0 \leq m < 1$ describes the modality of the Bactrian distribution. When $m = 0$, the Bactrian distribution is equivalent to a Normal(0, 1) distribution. As $m \rightarrow 1$, the distribution becomes increasingly bimodal (Fig. 3). Yang et al. 2013 [29] suggest that Bactrian($m = 0.95$) yields a proposal kernel superior to the uniform kernel, by placing minimal probability on steps which are too small or too large.

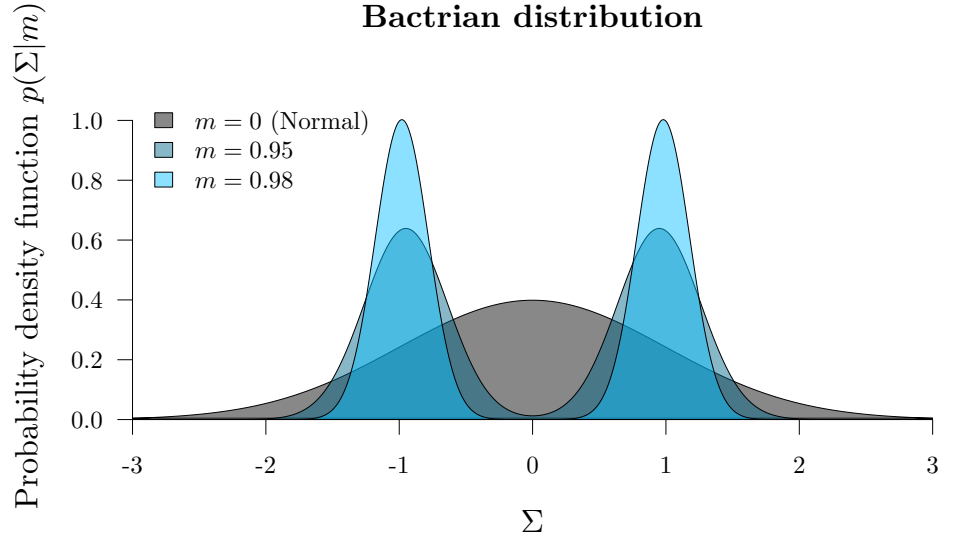


Fig 3. The Bactrian proposal kernel. The step size made under a Bactrian proposal kernel is equal to $s\Sigma$ where Σ is drawn from the above distribution and s is tunable.

In this article we compare the performance of uniform and Bactrian proposal kernels in the clock model. Two Bactrian distributions are compared ($m = 0.95$ and $m = 0.98$). The clock model operators which these proposal kernels apply to are described in **Table 3**.

	Operator(s)	Proposal	Parameter x
1	RandomWalk	$x' \leftarrow x + s\Sigma$	$\vec{\mathcal{R}}, \sigma$
2	Scale	$x' \leftarrow x \times e^{s\Sigma}$	$\vec{\mathcal{R}}, \sigma$
3	Interval	$y \leftarrow \frac{1-x}{x} \times e^{s\Sigma}$ $x' \leftarrow \frac{y}{y+1}$	$\vec{\mathcal{R}}$
4	ConstantDistance SimpleDistance	$x' \leftarrow x + s\Sigma$	t
5	SmallPulley	$x' \leftarrow x + s\Sigma$	$\vec{\mathcal{R}}$
6	FastClockScale	$x' \leftarrow x \times e^{s\Sigma}$	σ

Table 3. Proposal kernels $q(x'|x)$ of clock model operators. In each operator, Σ is drawn from either a Bactrian(m) or Uniform distribution. The scale size s is tunable. **ConstantDistance** and **SimpleDistance** propose tree heights t . The **Interval** operator applies to rate quantiles and respects its domain i.e. $0 < x, x' < 1$.

Narrow Exchange Rate

The **NarrowExchange** operator [38], used widely in BEAST [9, 39] and BEAST2 [10], is similar to nearest neighbour interchange (NNI), and works as follows (**Fig. 4**):

Step 1. Sample an internal/root node E from tree \mathcal{T} , where E has grandchildren.

Step 2. Identify the child of E with the greater height. Denote this child as D and its sibling as C (i.e. $t_D > t_C$).

Step 3. Randomly identify the two children of D as A and B .

Step 4. Relocate the $B - D$ branch onto the $C - E$ branch, so that B and C become siblings and their parent is D . All node heights are unchanged.

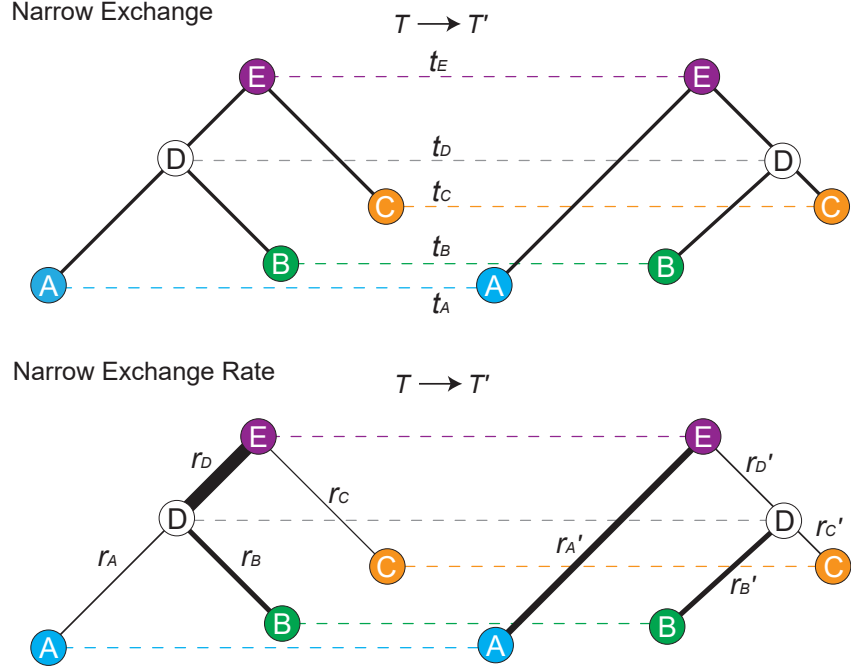


Fig 4. Depiction of NarrowExchange and NarrowExchangeRate operators.

Proposals are denoted by $\mathcal{T} \rightarrow \mathcal{T}'$. The vertical axis corresponds to node height t . In the bottom figure, branch rates r are indicated by line thickness. In this example, the \mathcal{D}_{AE} and \mathcal{D}_{CE} constraints are satisfied.

Lakner et al. 2008 [40] found that tree operators which perturb topology (such as NNI) consistently perform better than those which also change branch lengths (such as LOCAL [41] and Continuous Change [42]). If **NarrowExchange** could be adapted to the relaxed clock model by ensuring that genetic distances remain constant after the proposal, its performance may be improved even further. This may in turn permit proposing a new node height t_D and therefore changing branch (time) lengths.

Here, we present the **NarrowExchangeRate** (NER) operator. Let r_A, r_B, r_C , and r_D be the clock rates of nodes A, B, C , and D , respectively. In addition to the modest topological change applied by **NarrowExchange**, NER also proposes new clock rates r_A', r_B', r_C' , and r_D' . While NER does not alter t_D (i.e. $t_D' \leftarrow t_D$), we also consider NERw - a special case of the NER operator which embarks t_D on a random walk:

$$t_D' \leftarrow t_D + s\Sigma \quad (13)$$

for random walk step size $s\Sigma$ where s is a tunable scalar parameter and Σ is drawn from a uniform or **Bactrian proposal kernel**. NER (and NERw) are compatible with both the *real* and *quant* parameterisations. Analogous to the **ConstantDistance** operator, new rates are proposed such that genetic distances between nodes A , B , C , and E are maintained. Thus, there are $\binom{4}{2} = 6$ pairwise distance constraints.

$$\begin{aligned} \mathcal{D}_{AB} : \quad & r_A(t_D - t_A) + r_B(t_D - t_B) = \\ & r_A'(t_E - t_A) + r_D'(t_E - t_D') + r_B'(t_D' - t_B) \end{aligned} \quad (14)$$

$$\begin{aligned} \mathcal{D}_{AC} : \quad & r_A(t_D - t_A) + r_D(t_E - t_D) + r_C(t_E - t_C) = \\ & r_A'(t_E - t_A) + r_D'(t_E - t_D') + r_C'(t_D' - t_C) \end{aligned} \quad (15)$$

$$\begin{aligned} \mathcal{D}_{AE} : \quad & r_A(t_D - t_A) + r_D(t_E - t_D) = \\ & r_A'(t_E - t_A) \end{aligned} \quad (16)$$

$$\begin{aligned} \mathcal{D}_{BC} : \quad & r_B(t_D - t_B) + r_D(t_E - t_D) + r_C(t_E - t_C) = \\ & r_B'(t_D' - t_B) + r_C'(t_D' - t_C) \end{aligned} \quad (17)$$

$$\begin{aligned} \mathcal{D}_{BE} : \quad & r_B(t_D - t_B) + r_D(t_E - t_D) = \\ & r_B'(t_D' - t_B) + r_D'(t_E - t_D') \end{aligned} \quad (18)$$

$$\begin{aligned} \mathcal{D}_{CE} : \quad & r_C(t_E - t_C) = \\ & r_C'(t_D' - t_C) + r_D'(t_E - t_D') \end{aligned} \quad (19)$$

Further constraints are imposed by the model itself:

$$r_i > 0 \text{ and } r_i' > 0 \text{ for } i \in \{A, B, C, D\} \quad (20)$$

$$\max\{t_B, t_C\} < t_D' < t_E. \quad (21)$$

Unfortunately, it is not possible to solve all six \mathcal{D}_{ij} constraints without permitting non-positive rates or illegal trees. Therefore rather than conserving all six pairwise distances, NER conserves a *subset* of distances. It is not immediately clear which distances should be conserved.

Automated generation of operators and constraint satisfaction

The total space of NER operators is comprised of all possible subsets of distance constraints (i.e. $\{\}, \{\mathcal{D}_{AB}\}, \{\mathcal{D}_{AC}\}, \dots, \{\mathcal{D}_{AB}, \mathcal{D}_{AC}, \mathcal{D}_{AE}, \mathcal{D}_{BC}, \mathcal{D}_{BE}, \mathcal{D}_{CE}\}$) which are solvable. The simplest NER – the null operator denoted by $\text{NER}\{\}$ – does not satisfy any distance constraints. This is equivalent to **NarrowExchange**.

As it is unclear which NER variants would perform the best, we developed an automated pipeline for generating and testing these operators.

1. Solution finding. Using standard analytical linear-system solving libraries in MATLAB, the $2^6 = 64$ subsets of distance constraints are solved. 54 out of the 64 subsets were found to be solvable, and the unsolvables were discarded.

2. Solving Jacobian determinants. The determinant of the Jacobian matrix J is required for computing the Hastings ratio of the proposal. J is defined as

$$J = \begin{bmatrix} \frac{\partial r_A'}{\partial r_A} & \frac{\partial r_A'}{\partial r_B} & \frac{\partial r_A'}{\partial r_C} & \frac{\partial r_A'}{\partial r_D} \\ \frac{\partial r_B'}{\partial r_A} & \frac{\partial r_B'}{\partial r_B} & \frac{\partial r_B'}{\partial r_C} & \frac{\partial r_B'}{\partial r_D} \\ \frac{\partial r_C'}{\partial r_A} & \frac{\partial r_C'}{\partial r_B} & \frac{\partial r_C'}{\partial r_C} & \frac{\partial r_C'}{\partial r_D} \\ \frac{\partial r_D'}{\partial r_A} & \frac{\partial r_D'}{\partial r_B} & \frac{\partial r_D'}{\partial r_C} & \frac{\partial r_D'}{\partial r_D} \end{bmatrix}. \quad (22)$$

Computing the determinant $|J|$ invokes standard analytical differentiation and linear algebra libraries of MATLAB. 6 of the 54 solvable operators were found to have $|J| = 0$, corresponding to irreversible proposals, and were discarded.

3. Automated generation of BEAST2 operators. Java class files are generated using string processing. Each class corresponds to a single operator, extends the class of a meta-NER-operator, and is comprised of the solutions found in **1** and the Jacobian determinant found in **2**. $|J|$ is further augmented if the *quant* parameterisation is employed (**S1 Appendix**).

The 48 operators generated by this pipeline are evaluated and compared in **Results**. Each operator is considered with and without a random walk on t_D and thus there are 96 total settings.

A guided adaptive leaf rate operator

A *guided* operator incorporates knowledge about neighbouring states, while an *adaptive* operator undergoes a training process to improve its efficiency over time [43]. In previous work, parsimony scores and conditional clade probabilities of neighbouring trees have been employed by guided tree operators [23–25] and the latter has also been explored as the basis of adaptive tree operators [24, 25]. The (adaptive) mirror kernel [30] learns a target distribution which acts as a ‘mirror image’ of the current point x . The adaptable variance multivariate normal (AVMVN) kernel [28, 39] learns correlations between parameters during MCMC. Baele et al. 2017 observed a large increase ($\approx 5 - 10\times$) in sampling efficiency from using the AVMVN kernel on clock rates and substitution model parameters across partitions [28].

In this article we consider application of the AVMVN kernel to the branch rates of leaf nodes. This operator, referred to as **LeafAVMVN**, is not readily applicable to internal node branch rates due to their dependencies on tree topology.

AVMVN kernel

The AVMVN kernel assumes its parameters live in $x \in \mathbb{R}^N$ and that these parameters follow a Multivariate Normal distribution with covariance matrix Σ_N . Hence, the kernel operates on the logarithmic or logistic transformation of the N leaf branch rates, depending on the rate parameterisation:

$$x_i = \begin{cases} \log r_i & \text{for } \textit{real} \\ \log \frac{q_i}{1-q_i} & \text{for } \textit{quant} \end{cases} \quad (23)$$

where r_i is a real rate and q_i is a rate quantile. The AVMVN probability density is defined by

$$\mathcal{AVMVN}(x) = \mathcal{MVN}(x, (1 - \beta) \frac{\Sigma_N}{N} + \beta \frac{\mathbb{I}_N}{N}), \quad (24)$$

where \mathcal{MVN} is the Multivariate Normal probability density. β ($= 0.05$) is a constant which determines the fraction of the proposal determined by the identity matrix \mathbb{I}_N , as opposed to the covariance matrix Σ_D which is trained during MCMC.

The AVMVN proposal kernel is computed as

$$x' \leftarrow x + \sum_{i=1}^N \sum_{j=i}^N c_{i,j} \times s \Sigma \quad (25)$$

$$\text{where } c = \text{cholesky} \left((1 - \beta) \frac{\Sigma_N}{N} + \beta \frac{\mathbb{I}_N}{N} \right). \quad (26)$$

The $\text{cholesky}(Y)$ decomposition returns a lower diagonal matrix L , with positive real diagonal entries, such that $Y = LL'$ [44, 45]. s is a tunable step size parameter and Σ is a random variable drawn from a proposal kernel (uniform or Bactrian for instance). Our BEAST2 implementation of the AVMVN kernel is adapted from that of BEAST [39].

In Results, we evaluate the **LeafAVMVN** operator for its ability to estimate leaf rates. As the size of the covariance matrix Σ_N grows with the number of taxa N , AVMVN is hypothesised to work well on small trees but become less efficient with larger taxon sets.

Results

In this article we have introduced four new clock model operators. These operators are summarised in **Table 4** and are applied in conjunction with the pre-existing clock model operators presented in **Table 1**.

Operator	Description	Parameters
AdaptiveOperatorSampler	Samples sub-operators proportionally to their weights, which are learned (see Adaptive operator weighting).	$\vec{\mathcal{R}}, \sigma, \mathcal{T}$
SampleFromPrior	Resamples a random number of elements from their prior (see Adaptive operator weighting).	$\vec{\mathcal{R}}, \sigma$
NarrowExchangeRate	Moves a branch and re-computes branch rates so that genetic distances are constant (see Narrow Exchange Rate).	$\vec{\mathcal{R}}, \mathcal{T}$
LeafAVMVN	Proposals new rates for all leaves in one move (see A guided adaptive leaf rate operator) [28].	$\vec{\mathcal{R}}$

Table 4. Summary of clock model operators introduced throughout this article.

Assessment criteria and datasets

To avoid a cross-product explosion, the five targets for clock model improvement are evaluated sequentially in the following order: **Adaptive operator weighting**, **Substitution rate parameterisations**, **Bactrian proposal kernel**, **Narrow Exchange Rate**, and **A guided adaptive leaf rate operator**.

The setting which is considered to be the best in each step is then incorporated into the following step. This protocol and its outcomes are summarised in **Fig. 5**.

Methodologies are assessed according to the following criteria.

1. Validation. This is assessed by measuring the coverage of all estimated parameters in a well-calibrated simulation study, using 100 simulated datasets (with $N = 100$ taxa and $L = 5000$ nucleotide alignments). These are presented in **S2 Appendix**.

2. Mixing of parameters. Key parameters are evaluated for the number of effective samples generated per hour (ESS/hr). This calculation is performed by BEAST2 [10].

Methodologies are benchmarked using one simulated and nine empirical datasets – the latter were compiled [46] and partitioned [47] by Lanfear et al. as ‘benchmark alignments’ (**Table 5**). To facilitate convergence on complex datasets, each posterior distribution is sampled using coupled MCMC (MC³) with 4 chains [27]. Each methodology is benchmarked across 5 replicates on each dataset, using an the Intel Xeon Gold 6138 CPU (2.00 GHz) and all methodologies use identical models and operator configurations, except where a difference is specified.

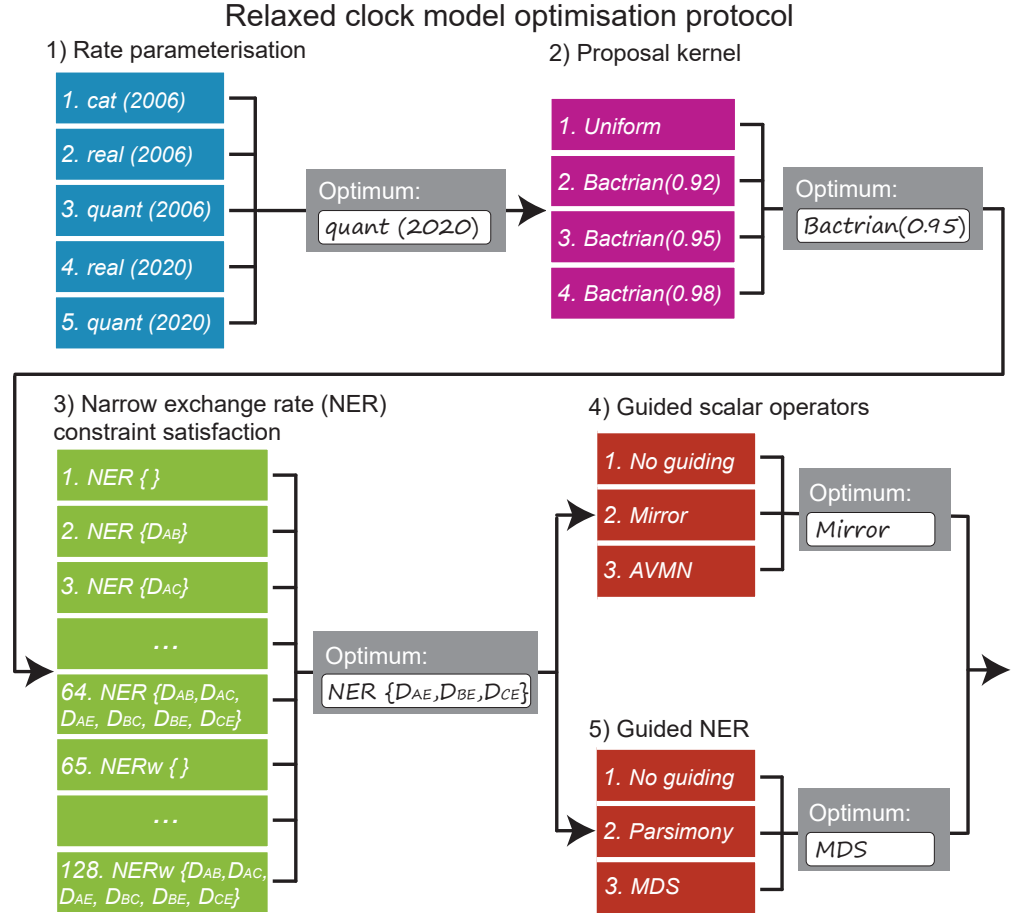


Fig 5. Protocol for optimising methodology settings. The three areas (detailed in **Models**) are optimised sequentially, where the best setting from each step is used when optimising the following step.

	N	P	L (kb)	L_{eff} (kb)	Description
1	38	8	9.1	6.45	Seed plants (Ran 2018 [48])
2	44	7	5.9	1.8	Squirrel Fishes (Dornburg 2012 [49])
3	44	3	1.9	0.8	Bark beetles (Cognato 2001 [50])
4	51	6	5.4	1.8	Southern beeches (Sauquet 2011 [51])
5	61	8	6.9	4.3	Bony fishes (Broughton 2013 [52])
6	70	3	2.2	0.9	Caterpillars (Kawahara 2013 [53])
7	78	8	3.4	3.1	Animals (Cannon 2016 [54])
8	80	1	10.0	4.2	<i>Simulated data</i>
9	94	4	2.2	1	Bees (Rightmyer 2013 [55])
10	106	1	0.8	0.5	Songbirds (Moyle 2016 [56])

Table 5. Datasets used during benchmarking, sorted in increasing order of taxa count N . Number of partitions P , total alignment length L , and number of patterns L_{eff} are also specified.

Round 1: A simple operator-weight learning algorithm can improve performance

We compared the cons, nocons, and adapt operator configurations for the *real* and *quant* parameterisations (**Table 6**). The nocons configuration contained the standard BEAST2 operator mix, while cons contained the operator mix used by Zhang and Drummond 2020 [31]. The adapt mix combines the above operators, as well as the rudimentary `SampleFromPrior` operator, and learns the weights of each operator using `AdaptiveOperatorSampler`.

Configuration	Operator	Weight	Parameterisations
nocons	<code>RandomWalk(\mathcal{R})</code>	10	<i>real</i>
	<code>Scale($\vec{\mathcal{R}}$)</code>	10	<i>real</i>
	<code>Uniform($\vec{\mathcal{R}}$)</code>	10	<i>quant</i>
	<code>Interval($\vec{\mathcal{R}}$)</code>	10	<i>quant</i>
	<code>Swap($\vec{\mathcal{R}}$)</code>	10	<i>real, quant</i>
	<code>Scale(σ)</code>	10	<i>real, quant</i>
cons	<code>ConstantDistance($\vec{\mathcal{R}}, \mathcal{T}$)</code>	$20 \times \frac{2N-2}{2N-1}$	<i>real, quant</i>
	<code>SimpleDistance($\vec{\mathcal{R}}, \mathcal{T}$)</code>	$10 \times \frac{2N-2}{2N-1}$	<i>real, quant</i>
	<code>SmallPulley($\vec{\mathcal{R}}$)</code>	$10 \times \frac{2N-2}{2N-1}$	<i>real, quant</i>
	<code>RandomWalk($\vec{\mathcal{R}}$)</code>	5	<i>real</i>
	<code>Scale($\vec{\mathcal{R}}$)</code>	2.5	<i>real</i>
	<code>Uniform($\vec{\mathcal{R}}$)</code>	5	<i>quant</i>
	<code>Interval($\vec{\mathcal{R}}$)</code>	2.5	<i>quant</i>
	<code>Swap($\vec{\mathcal{R}}$)</code>	2.5	<i>real, quant</i>
	<code>FastClockScale($\sigma, \vec{\mathcal{R}}$)</code>	10	<i>real</i>
	<code>Scale(σ)</code>	10	<i>quant</i>
adapt	<code>AdaptiveOperatorSampler(σ)</code>	10	<i>real, quant</i>
	<code>AdaptiveOperatorSampler($\vec{\mathcal{R}}$)</code>	$30 \times \frac{2N-2}{2N-1}$	<i>real, quant</i>
	<code>AdaptiveOperatorSampler(root)</code>	$30 \times \frac{1}{2N-1}$	<i>real, quant</i>

Table 6. Operator configurations for Round 1. In each configuration, the weight behind \mathcal{R} sums to 30 and for σ is equal to 10 within any one rate parameterisation. Operators which apply to either internal nodes or the root (but not both) are weighted according to leaf count N . The adapt operators are further broken down in **Table 2**. The total weight of all operators across all parameters was 116 in each MCMC setup. Therefore a weight of 10 corresponds to sampling probability of 0.086, for example.

These results show that, while nocons gives better performance than cons on smaller datasets (corresponding to low signal), and cons performs better on larger datasets (high signal), the adapt configuration gives the best performance overall. This trend is observed for both the *real* and the *quant* parameterisations. Therefore, the `AdaptiveOperatorSampler` operator will be included in all subsequent rounds in the tournament.

Round 2: The three rate parameterisations yield similar mixing ability

We compared the three rate parameterisations described in **Substitution rate parameterisations**. The *cat* operator configuration is described in **Table 7**. *real* and *quant* make use of constant distance tree operators [31] and both use the **AdaptiveOperatorSampler** operator to learn clock model operator weights. The three settings are validated in **S2 Appendix**.

Configuration	Operator	Weight
<i>cat</i>	RandomWalk ($\vec{\mathcal{R}}$)	10
	Uniform ($\vec{\mathcal{R}}$)	10
	Swap ($\vec{\mathcal{R}}$)	10
	Scale (σ)	10

Table 7. Operator configurations for *cat* in Round 2. The configurations of *real* (adapt) and *quant* (adapt) are shown in **Table 6**.

Round 3: The Bactrian proposal kernel is the best for the clock model

Round 4: Finding the best NER operator variant

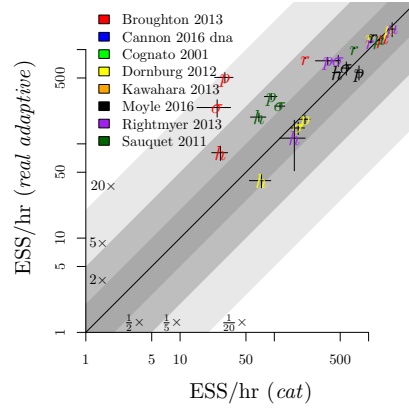
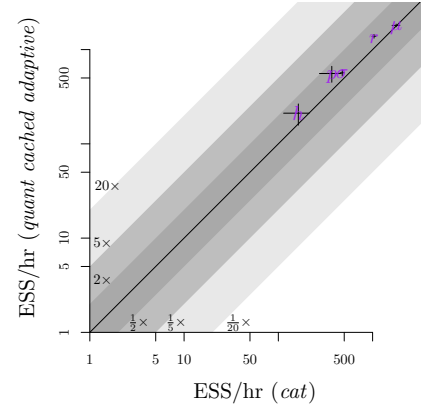
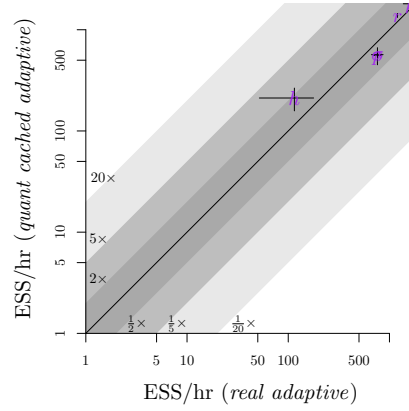
The **Narrow Exchange Rate** (NER) operators are evaluated. This protocol selects the best among 48 NER (no random walk) and 48 NERw (Bactrian(0.95) random walk) operators, and has two phases. First, the best of the 96 is selected by comparing operator acceptance rates on simulated data. Second, the selected operator is benchmarked with respect to convergence time and sampling rate on real data (**Table 5**). The analyses in this section invoke the *quant* parameterisation and Bactrian(0.95) proposal kernels on clock model parameters.

Initial screening by acceptance rate

We selected the best operator variant by performing MCMC on 300 simulated datasets, where each MCMC employed all 96 NER/NERw variants. Simulated datasets have $N = 30$ taxa and an alignment with $L \sim \text{Uniform}(10^2, 10^4)$ sites. The acceptance rate of each operator is compared to that of the null operator $\text{NER}\{\}$ (i.e. **Narrow Exchange**).

Fig. 7 shows that NER variants which satisfy the genetic distances between nodes B and A (i.e. \mathcal{D}_{AB}) or between B and C (i.e. \mathcal{D}_{BC}) usually perform worse than the standard **Narrow Exchange** operator, where B is the node being interchanged from the A branch to the C branch (**Fig. 4**). This is an intuitive result. If the posterior distribution is relatively flat, and the data presents high uncertainty in the positioning of B , with respect to A and C , then the topological rearrangement performed by **Narrow Exchange** will be favoured. However, this uncertainty in the *topology* is likely coupled with uncertainty in the *distance* between B and A or between B and C . Thus, in this case, respecting the \mathcal{D}_{AB} and \mathcal{D}_{BC} constraints (by proposing branch rates) makes too many unnecessary changes to the state and the operator performs worse.

Fig. 7 also reveals a cluster of NER variants which – under the conditions of the simulation – performed better than the null operator $\text{NER}\{\}$ around 25% of the time and performed worse around 10% of the time. One such operator is $\text{NER}\{\mathcal{D}_{AE}, \mathcal{D}_{BE}, \mathcal{D}_{CE}\}$. This variant conserves the genetic distance between all child nodes A , B , and C , and the grandparent node E . This is performed by proposing rates

Comparison of *real adaptive* and *cat*Comparison of *quant cached adaptive* and *cat*Comparison of *quant cached adaptive* and *real adaptive*

Speed comparison

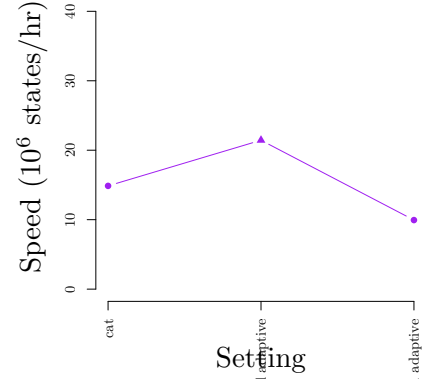


Fig 6. Rate parameterisation performance evaluation. Comparison of ESS/hr (averaged across five independent MC3 analyses) with respect to relevant terms – P : posterior density; L : likelihood, p : prior density, r : clock rate ESS averaged across all leaves, \hat{r} : branch rate mean, v : branch rate variance, σ : clock standard deviation, κ : HKY model transition-transversion ratio, λ : Yule model birth rate. h : tree height. Datasets are displayed in **Table 5**.

for r_A , r_B , and r_C while obeying the distance constraints imposed by the operator. Exploring this operator further, we can see that $\text{NER}\{\mathcal{D}_{AE}, \mathcal{D}_{BE}, \mathcal{D}_{CE}\}$ is at its best when there is a large variance in branch rate i.e. when clock standard deviation σ is high ($\sigma \gtrsim 0.5$ for $N = 30$), corresponding to data which is not clock-like. On the other hand, $\text{NER}\{\}$ is much preferred when the operator's acceptance rate is high ($\gtrsim 0.15$) – corresponding with datasets with a small number of site patterns ($L_{\text{eff}} \lesssim 500$ for $N = 30$) and thus poor signal. Overall, $\text{NER}\{\mathcal{D}_{AE}, \mathcal{D}_{BE}, \mathcal{D}_{CE}\}$ outperforms the standard **Narrow Exchange** operator when the data is not clock-like and contains enough signal.

Finally, **Fig. 7** shows that by applying a (Bactrian) random walk to t_D – the height of internal node D – the acceptance rate of NER plummets dramatically. This effect is most dominant for the NER variants which satisfy distance constraints (i.e. the operators which are not $\text{NER}\{\}$). This result is unfortunate however not unexpected,

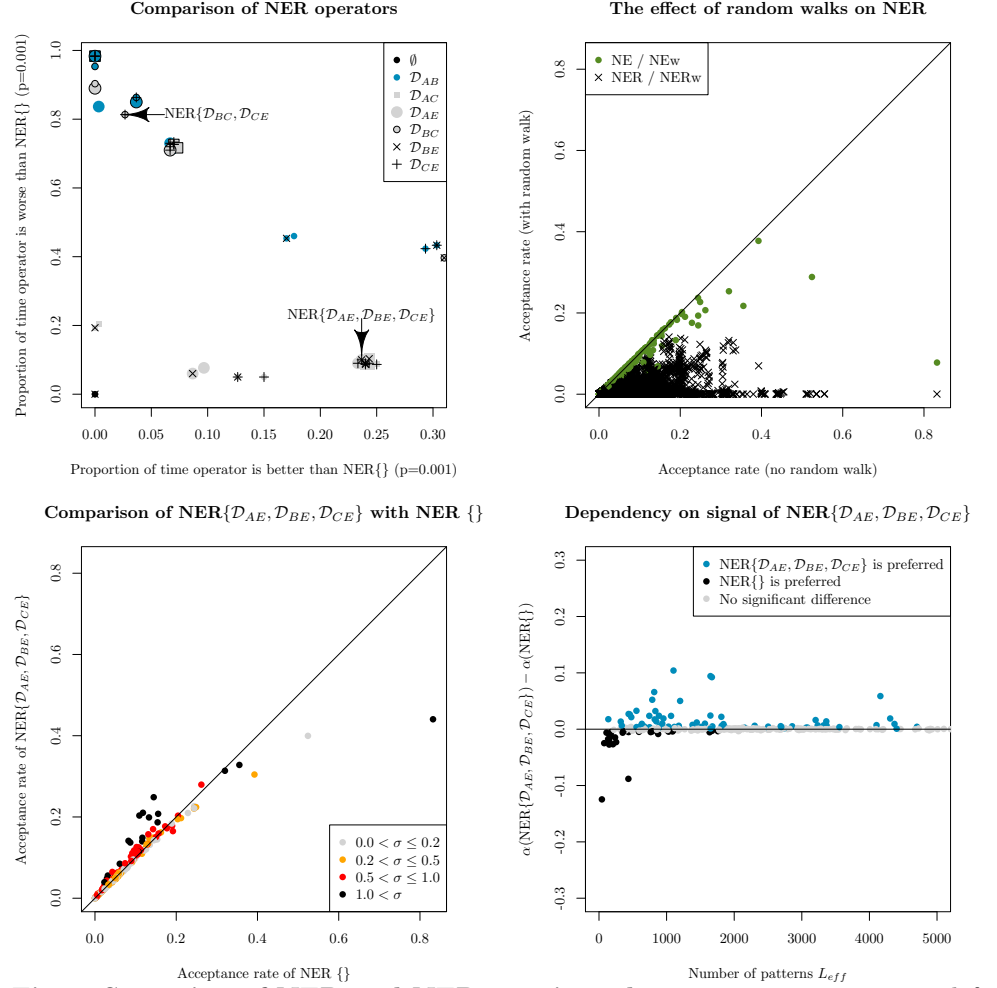


Fig 7. Screening of NER and NERw variants by acceptance rate. Top left: comparison of NER variants with the null operator $\text{NER}\{\}$ (i.e. **N**arrow **E**xchange). Each of the 48 operators are represented by a single point, uniquely encoded by the point stylings. The number of times each operator is proposed and accepted is compared with that of $\text{NER}\{\}$, and one-sided z-tests are performed to assess the statistical significance between the two acceptance rates ($p = 0.001$). This process is repeated for each of 300 simulated datasets. The axes of each plot are the proportion of these 300 simulations for which there is evidence that the operator is better than $\text{NER}\{\}$ (x-axis) or worse than $\text{NER}\{\}$ (y-axis). Top right: comparison of NER and NERw acceptance rates. Each point is one NER/NERw variant from a single simulation. Bottom: relationship between the acceptance rates α of $\text{NER}\{\mathcal{D}_{AE}, \mathcal{D}_{BE}, \mathcal{D}_{CE}\}$ and $\text{NER}\{\}$ with the clock model standard deviation σ and the number of patterns L_{eff} . Each point is a single simulation.

and is consistent with Lakner et al. 2008 [40], who observed that tree operators perform best when they change either topology, or branch lengths, but not both.

Although there are several operators tying for first place, we selected the $\text{NER}\{\mathcal{D}_{AE}, \mathcal{D}_{BE}, \mathcal{D}_{CE}\}$ operator to proceed to the next round of optimisation.

Benchmarking convergence time	356
Evaluating the AVMVN leaf rate operator	357
Clock model averaging	358
Discussion	359
Conclusion	360

Supporting information

361

S1 Appendix. Rate quantiles. The linear piecewise approximation used in the
quant parameterisation is described. **Constant distance** tree operators [31] are
extended to the *quant* parameterisation.

362

363

364

S2 Appendix. Well-calibrated simulation studies. Methodologies are validated
using well-calibrated simulation studies.

365

366

References

1. Zuckerkandl E. Molecular disease, evolution, and genetic heterogeneity. *Horizons in biochemistry*. 1962; p. 189–225.
2. Douzery EJ, Delsuc F, Stanhope MJ, Huchon D. Local molecular clocks in three nuclear genes: divergence times for rodents and other mammals and incompatibility among fossil calibrations. *Journal of Molecular Evolution*. 2003;57(1):S201–S213.
3. Drummond AJ, Ho SY, Phillips MJ, Rambaut A. Relaxed phylogenetics and dating with confidence. *PLoS biology*. 2006;4(5):e88.
4. Kuhner MK, Yamato J, Felsenstein J. Estimating effective population size and mutation rate from sequence data using Metropolis-Hastings sampling. *Genetics*. 1995;140(4):1421–1430.
5. Larget B, Simon DL. Markov chain Monte Carlo algorithms for the Bayesian analysis of phylogenetic trees. *Molecular biology and evolution*. 1999;16(6):750–759.
6. Mau B, Newton MA, Larget B. Bayesian phylogenetic inference via Markov chain Monte Carlo methods. *Biometrics*. 1999;55(1):1–12.
7. Metropolis N. Equation of state calculations by fast computing machines. *J Chem Phys*. 1953;21:1087–1092.
8. Hastings W. Monte-Carlo sampling methods using Markov chains and their applications. *Biometrika*. 1970;57:97–109.
9. Drummond AJ, Suchard MA, Xie D, Rambaut A. Bayesian phylogenetics with BEAUti and the BEAST 1.7. *Molecular biology and evolution*. 2012;29(8):1969–1973.
10. Bouckaert R, Vaughan TG, Barido-Sottani J, Duchêne S, Fourment M, Gavryushkina A, et al. BEAST 2.5: An advanced software platform for Bayesian evolutionary analysis. *PLoS computational biology*. 2019;15(4):e1006650.
11. Ronquist F, Teslenko M, Van Der Mark P, Ayres DL, Darling A, Höhna S, et al. MrBayes 3.2: efficient Bayesian phylogenetic inference and model choice across a large model space. *Systematic biology*. 2012;61(3):539–542.
12. Höhna S, Landis MJ, Heath TA, Boussau B, Lartillot N, Moore BR, et al. RevBayes: Bayesian phylogenetic inference using graphical models and an interactive model-specification language. *Systematic biology*. 2016;65(4):726–736.
13. Zuckerkandl E, Pauling L. Evolutionary divergence and convergence in proteins. In: *Evolving genes and proteins*. Elsevier; 1965. p. 97–166.
14. Gillespie JH. *The causes of molecular evolution*. vol. 2. Oxford University Press On Demand; 1994.
15. Woolfit M. Effective population size and the rate and pattern of nucleotide substitutions. *Biology letters*. 2009;5(3):417–420.
16. Loh E, Salk JJ, Loeb LA. Optimization of DNA polymerase mutation rates during bacterial evolution. *Proceedings of the National Academy of Sciences*. 2010;107(3):1154–1159.

17. Lepage T, Bryant D, Philippe H, Lartillot N. A general comparison of relaxed molecular clock models. *Molecular biology and evolution*. 2007;24(12):2669–2680.
18. Li WLS, Drummond AJ. Model averaging and Bayes factor calculation of relaxed molecular clocks in Bayesian phylogenetics. *Molecular biology and evolution*. 2012;29(2):751–761.
19. Faria NR, Quick J, Claro I, Theze J, de Jesus JG, Giovanetti M, et al. Establishment and cryptic transmission of Zika virus in Brazil and the Americas. *Nature*. 2017;546(7658):406–410.
20. Giovanetti M, Benvenuto D, Angeletti S, Ciccozzi M. The first two cases of 2019-nCoV in Italy: Where they come from? *Journal of medical virology*. 2020;92(5):518–521.
21. Huelsenbeck JP, Larget B, Swofford D. A compound Poisson process for relaxing the molecular clock. *Genetics*. 2000;154(4):1879–1892.
22. Drummond AJ, Suchard MA. Bayesian random local clocks, or one rate to rule them all. *BMC biology*. 2010;8(1):1–12.
23. Zhang C, Huelsenbeck JP, Ronquist F. Using parsimony-guided tree proposals to accelerate convergence in Bayesian phylogenetic inference. *Systematic Biology*. 2020;.
24. Meyer X. Adaptive Tree Proposals for Bayesian Phylogenetic Inference. *BioRxiv*. 2019; p. 783597.
25. Höhna S, Drummond AJ. Guided tree topology proposals for Bayesian phylogenetic inference. *Systematic biology*. 2012;61(1):1–11.
26. Altekar G, Dwarkadas S, Huelsenbeck JP, Ronquist F. Parallel metropolis coupled Markov chain Monte Carlo for Bayesian phylogenetic inference. *Bioinformatics*. 2004;20(3):407–415.
27. Müller NF, Bouckaert R. Coupled MCMC in Beast 2. *bioRxiv*. 2019;.
28. Baele G, Lemey P, Rambaut A, Suchard MA. Adaptive MCMC in Bayesian phylogenetics: an application to analyzing partitioned data in BEAST. *Bioinformatics*. 2017;33(12):1798–1805.
29. Yang Z, Rodríguez CE. Searching for efficient Markov chain Monte Carlo proposal kernels. *Proceedings of the National Academy of Sciences*. 2013;110(48):19307–19312.
30. Thawornwattana Y, Dalquen D, Yang Z, et al. Designing simple and efficient Markov chain Monte Carlo proposal kernels. *Bayesian Analysis*. 2018;13(4):1037–1063.
31. Zhang R, Drummond A. Improving the performance of Bayesian phylogenetic inference under relaxed clock models. *BMC Evolutionary Biology*. 2020;20:1–28.
32. Felsenstein J. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of molecular evolution*. 1981;17(6):368–376.
33. Green PJ. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*. 1995;82(4):711–732.
34. Geyer CJ. The metropolis-hastings-green algorithm; 2003.

35. Gelman A. Parameterization and Bayesian modeling. *Journal of the American Statistical Association*. 2004;99(466):537–545.
36. Jukes TH, Cantor CR, et al. Evolution of protein molecules. *Mammalian protein metabolism*. 1969;3:21–132.
37. Roberts GO, Gelman A, Gilks WR, et al. Weak convergence and optimal scaling of random walk Metropolis algorithms. *The annals of applied probability*. 1997;7(1):110–120.
38. Drummond AJ, Nicholls GK, Rodrigo AG, Solomon W. Estimating mutation parameters, population history and genealogy simultaneously from temporally spaced sequence data. *Genetics*. 2002;161(3):1307–1320.
39. Suchard MA, Lemey P, Baele G, Ayres DL, Drummond AJ, Rambaut A. Bayesian phylogenetic and phylodynamic data integration using BEAST 1.10. *Virus evolution*. 2018;4(1):vey016.
40. Lakner C, Van Der Mark P, Huelsenbeck JP, Larget B, Ronquist F. Efficiency of Markov chain Monte Carlo tree proposals in Bayesian phylogenetics. *Systematic biology*. 2008;57(1):86–103.
41. Simon D, Larget B. Bayesian analysis in molecular biology and evolution (BAMBE) <http://www.mathcs.duq.edu/larget/bambe.html>. Pittsburgh, Pennsylvania. 1998;.
42. Jow H, Hudelot C, Rattray M, Higgs P. Bayesian phylogenetics using an RNA substitution model applied to early mammalian evolution. *Molecular Biology and Evolution*. 2002;19(9):1591–1601.
43. Roberts GO, Rosenthal JS. Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *Journal of applied probability*. 2007;44(2):458–475.
44. Lindstrom MJ, Bates DM. Newton—Raphson and EM algorithms for linear mixed-effects models for repeated-measures data. *Journal of the American Statistical Association*. 1988;83(404):1014–1022.
45. Pourahmadi M. Cholesky decompositions and estimation of a covariance matrix: orthogonality of variance–correlation parameters. *Biometrika*. 2007;94(4):1006–1013.
46. Lanfear R. BenchmarkAlignments <https://github.com/roblanf/BenchmarkAlignments>. GitHub. 2019;.
47. Lanfear R, Frandsen PB, Wright AM, Senfeld T, Calcott B. PartitionFinder 2: new methods for selecting partitioned models of evolution for molecular and morphological phylogenetic analyses. *Molecular biology and evolution*. 2016;34(3):772–773.
48. Ran JH, Shen TT, Wang MM, Wang XQ. Phylogenomics resolves the deep phylogeny of seed plants and indicates partial convergent or homoplastic evolution between Gnetales and angiosperms. *Proceedings of the Royal Society B: Biological Sciences*. 2018;285(1881):20181012. doi:10.1098/rspb.2018.1012.
49. Dornburg A, Moore JA, Webster R, Warren DL, Brandley MC, Iglesias TL, et al. Molecular phylogenetics of squirrelfishes and soldierfishes (Teleostei: Beryciformes: Holocentridae): Reconciling more than 100 years of taxonomic confusion. *Molecular Phylogenetics and Evolution*. 2012;65(2):727–738. doi:10.1016/j.ympev.2012.07.020.

50. Cognato AI, Vogler AP. Exploring Data Interaction and Nucleotide Alignment in a Multiple Gene Analysis of *Ips* (Coleoptera: Scolytinae). *Systematic Biology*. 2001;50(6):758–780. doi:10.1080/106351501753462803.
51. Sauquet H, Ho SYW, Gandolfo MA, Jordan GJ, Wilf P, Cantrill DJ, et al. Testing the Impact of Calibration on Molecular Divergence Times Using a Fossil-Rich Group: The Case of *Nothofagus* (Fagales). *Systematic Biology*. 2011;61(2):289–313. doi:10.1093/sysbio/syr116.
52. Broughton RE, Betancur-R R, Li C, Arratia G, Ortí G. Multi-locus phylogenetic analysis reveals the pattern and tempo of bony fish evolution. *PLoS Currents*. 2013;doi:10.1371/currents.tol.2ca8041495ffafd0c92756e75247483e.
53. Kawahara AY, Rubinoff D. Convergent evolution of morphology and habitat use in the explosive Hawaiian fancy case caterpillar radiation. *Journal of Evolutionary Biology*. 2013;26(8):1763–1773. doi:10.1111/jeb.12176.
54. Cannon JT, Vellutini BC, Smith J, Ronquist F, Jondelius U, Hejnol A. Xenacoelomorpha is the sister group to Nephrozoa. *Nature*. 2016;530(7588):89–93. doi:10.1038/nature16520.
55. RIGHTMYER MG, GRISWOLD T, BRADY SG. Phylogeny and systematics of the bee genus *Osmia* (Hymenoptera: Megachilidae) with emphasis on North American *Melanosmia*: subgenera, synonymies and nesting biology revisited. *Systematic Entomology*. 2013;38(3):561–576. doi:10.1111/syen.12013.
56. Moyle RG, Oliveros CH, Andersen MJ, Hosner PA, Benz BW, Manthey JD, et al. Tectonic collision and uplift of Wallacea triggered the global songbird radiation. *Nature Communications*. 2016;7(1). doi:10.1038/ncomms12709.