Deep Generative Models for Phylogenetic Inference with Complex Evolutionary Processes

Ethan Baron [*1] **Alan N. Amin** [*1] **Andrew Gordon Wilson** [1]

Abstract

Phylogenetic inference plays a central role in understanding evolutionary relationships, with applications ranging from tracking pathogen spread to reconstructing the history of life. Conventionally, practitioners obtain posteriors over the phylogenetic tree of species using MCMC methods. However, such likelihood-based methods are only tractable for simple evolutionary models with restrictive assumptions. For more complex and realistic evolutionary models, conventional methods are prohibitively expensive, inaccurate, or even impossible. We instead advocate for a simulation-based inference approach, by using simulated data from an evolutionary model to train a neural network that predicts tree topologies conditioned on sequences. To accurately represent the complex posterior distributions over tree topologies that can arise, we present flexible models that iteratively generate trees using three natural paradigms: top-down, middle-out, and bottom-up. We use a discrete diffusion framework to train these models efficiently on large-scale simulated datasets of phylogenetic trees. For all three generative paradigms, our models fit the data substantially better than the previous state-of-the-art simulation-based method, Phyloformer 2, and obtain more accurate posteriors on real datasets. Finally, our models outperform misspecified conventional methods on data following complex evolutionary processes.

1. Introduction

To understand ancient evolution (Whelan et al., 2017), the origins of pathogens (Bloom, 2025), and the function of the immune system (DeWitt et al., 2018), biologists collect DNA or protein sequences and try to infer a phylogenetic tree (Felsenstein, 2004). In addition to the topology of the tree, biologists wish to infer parameters that describe, for example, when a most recent common ancestor has occurred (Tamura et al., 2012), whether there is selective pressure (Goldman & Yang, 1994), or how fast a pathogen is growing (Drummond et al., 2005). Just as important as inferring the tree is representing our uncertainty (Huelsenbeck et al., 2000). Bayesian inference provides a natural framework: the posterior represents our belief in the tree, including our uncertainty (Felsenstein, 2004, Ch. 18).

To sample from the posterior, in principle we can use MCMC (Huelsenbeck & Ronquist, 2001; Suchard et al., 2018). However, Bayesian inference tools like MCMC require access to a likelihood, which is only easy to calculate for the simplest models of evolution. We can divide more complex models of evolution into those where the likelihood is tractable but expensive, and those where the likelihood is completely intractable.

In models with expensive likelihoods, inference with MCMC is prone to errors (Whelan & Halanych, 2017) and time consuming, in some cases taking weeks to infer a tree with a hundred species (Susko et al., 2018). These models include those accounting for different mutation processes at different sites (Banos et al., 2024; Lartillot & Philippe, 2004), changes of mutation processes over time (Wang et al., 2007; Crotty et al., 2020), and different mutation rates at different sites (Yang, 1994) and their correlation between nearby sites (Felsenstein & Churchill, 1996). In practice, practitioners speed up inference with strategies like taking the values of evolutionary parameters from previously fit models (Le & Gascuel, 2008; Le et al., 2012), alternate fitting evolutionary parameters and the tree (Prillo et al., 2023), or relying on simpler models. Unfortunately, these strategies are the sources of many errors in the literature (Pisani et al., 2015; Susko et al., 2018; Youssef et al., 2020).

Other models have completely intractable likelihoods despite describing ubiquitous biological processes. These include models of sequence insertions and deletions (Fletcher & Yang, 2009), selection-dependence between sites (Youssef et al., 2020), and context-dependent mutation rates (Seplyarskiy et al., 2021). Such models are often dealt with by making strong assumptions (Holmes

[1]New York University. Correspondence to: Ethan Baron <eb4727@nyu.edu>.

Accepted at the 2026 Workshop on Generative and Agentic AI for Biology (ICML 2026)

& Bruno, 2001; Matsen & Ralph, 2022), discarding data (Siepel & Haussler, 2004; Tan et al., 2015), or, more often, ignoring these phenomena entirely.

Unlike likelihood calculation, simulation is simple in nearly all models of evolution, including those with intractable likelihoods. Ideally we could learn to perform inference by training a model on data simulated from the prior (Papamakarios & Murray, 2016), a framework known as simulation-based inference. Indeed, this framework has been extremely successful for Bayesian inference in astronomy (Dax et al., 2021), neuroscience (Gonçalves et al., 2020), and on tabular data (Hollmann et al., 2025). The main challenge in applying simulation-based inference to this problem is exploring the combinatorial space of tree topologies. Previous models map sequences to parameters of a parametric distribution over trees, limiting expressivity (Blassel et al., 2025).

To build flexible models that can support rich posterior distributions, we use iterative generative models that perform multiple forward passes to sample a tree topology from the posterior. We describe three natural paradigms for parameterizing iterative generative models for trees: **bottom-up**, **middle-out**, and **top-down**. We train flexible attention-based generative models for each paradigm using a discrete diffusion objective which allows us to train efficiently on large simulated data.

We show that our generative models learn to accurately represent the posterior in cases where Bayesian inference is tractable, outperforming previous simulation-based models. Moreover, we show that our models successfully learn posteriors under complex evolutionary models where Bayesian inference is intractable, significantly improving the fit on simulated data. Overall, our work unlocks a new frontier in Bayesian phylogenetic inference by enabling accurate and efficient training of deep learning architectures to learn the posteriors under realistic evolutionary models.

In Section 2, we introduce the phylogenetic inference problem, discuss limitations of conventional methods, and motivate our simulation-based approach. In Section 3, we discuss how to construct flexible generative models over trees using the three paradigms and iterative generation. In Section 4, we describe how to efficiently train these models using a discrete diffusion approach. In Section 6, we present our empirical results. We conclude in Section 7. Our code is available at `https://anonymous.4open.science/r/PhyloPFN-3DEE`.

2. Background and motivation

In this section, we introduce Bayesian phylogenetic inference, explain why conventional methods cannot be used under realistic evolutionary models, and advocate for a simulation-based inference approach instead.

2.1. Overview of phylogenetic inference

Let $X_1, \ldots, X_N \in V^L$ be a set of DNA sequences of length L from the genomes of N species (also known as "taxa") descended from a common ancestor. We represent how these sequences evolved with a phylogenetic tree T with N leaves, one representing each species. In a phylogenetic tree, each internal node represents the common ancestor of its two children. Phylogenetic trees are bifurcating, meaning each internal node has exactly two children, representing the fact that speciation events give rise to two new species.

In Bayesian phylogenetic inference, we wish to sample from the *posterior* $p(T \mid X_{1:N})$ under some *prior* base distribution over tree topologies $p(T)$ and *likelihood* corresponding to an evolutionary model $p(X_{1:N} \mid T)$. In this paper, we consider inferring the topology of the tree T without predicting each branch length, though we believe our methods can be easily adapted to accommodate branch length prediction.

2.2. Priors over tree topologies

There are several canonical choices for the prior over tree topologies $p(T)$. The PDA distribution (also known sometimes as simply the uniform model) assigns a uniform probability to each topology, typically producing less balanced trees. In contrast, the Yule model assumes a uniform rate of speciation at all nodes, producing relatively balanced trees. In practice, published trees are generally less balanced than Yule trees but more balanced than PDA trees. Ford's alpha model (Ford, 2005) and Aldous' beta model (Aldous, 1996) offer two ways to interpolate between these extremes.

2.3. Evolutionary models

The simplest evolutionary model is known as the Jukes-Cantor (JC) model (Jukes & Cantor, 1969). The JC model simulates an independent Markov substitution process at each position of the sequence for each species.

Formally, JC parameterizes the evolution of species along a tree T with given branch lengths $r_{u\to v}$ for all parents and children u, v in T. We start with a single species Y at the root, whose sequence is sampled uniformly at random $Y^{(\ell)}_{\text{root}} \sim \text{Uniform}\{A, T, C, G\}$ for all $\ell = 1, 2, \ldots, L$. For each branch, we then simulate an independent uniform Markov substitution process for time $r_{\text{root}\to v}$, mutating $Y^{(\ell)}$ into a different random letter every $\Delta t \sim \text{Exp}(1)$. We continue similarly, simulating the Markov process down the tree T until we have simulated the sequences at all of the leaves $X_1, \ldots, X_N$.

Despite the simplicity of the JC model, it is not obvious how to calculate $p(X_{1:N} \mid T)$, since it requires marginalizing out all ancestral sequences. For this calculation, Felsenstein (1973) developed the pruning algorithm, which continues

to be the backbone for likelihood calculations in phylogenetic inference. Felsenstein's pruning algorithm can also be applied to calculate likelihoods for some more elaborate models, such as ones that allow different sites to mutate with different rates. However, it strongly relies on the assumption that evolution occurs independently at each position.

Imagine if, when simulating the JC model, the rate of mutation of a letter $Y^{(l)}$ was not independent, but rather depended on the identity of its neighbors $Y^{(l-1)}$ and $Y^{(l+1)}$. Alternatively, imagine if there was a chance for a random chunk of five contiguous letters to be inserted or deleted. These sorts of phenomena are ubiquitous in real data, and while it is straightforward to simulate $X_{1:N} \mid T$ with these effects, computing the likelihood becomes intractable, making typical strategies for posterior inference impossible.

2.4. Simulation-based posterior inference

When we use simple evolutionary models like JC, we can compute likelihoods with Felsenstein's pruning algorithm and perform Bayesian inference with MCMC. However, when likelihood evaluation is expensive or impossible, we cannot perform Bayesian inference with these methods. How can we perform posterior inference when we cannot compute the likelihood $p(X_{1:N} \mid T)$, but we can only sample from this distribution?

We suggest *simulation-based inference*, in which we sample many synthetic examples of $(X_{1:N}, T)$ pairs following a given prior and likelihood, and train a generative model to predict T conditioned on $X_{1:N}$. The trained model can then be used to sample from the posterior for T given real sequences $X_{1:N}$! Unfortunately, most past works on machine learning methods for phylogenetic inference are not applicable to the problem of simulation-based inference. We cover these related works in Appendix A.

3. Flexible generative models over trees

To perform simulation-based inference for phylogenetic trees, we need a flexible model capable of representing complex posterior distributions over tree topologies. In this section, we introduce three natural paradigms for constructing generative models over trees, describe why previous methods are insufficiently flexible for our purposes, and argue that we should instead use iterative generative models for this problem.

3.1. Three generative paradigms for trees

We consider three paradigms for parameterizing generative models over trees: **bottom-up**, **middle-out**, and **top-down**. We provide an illustration of the three paradigms in Figure 1.

The **bottom-up** paradigm generates trees by starting with each taxon as its own subtree containing one node, and iteratively merging two subtrees together until a complete bifurcating tree is formed.

The **middle-out** paradigm generates trees by adding one taxon to the tree at a time. Specifically, we start by randomly selecting two species and placing them in an unrooted tree, along with a "pseudo-taxon" representing the root of the original tree. We then graft one taxon at a time, attaching it to the current unrooted tree, until all taxa have been added.

The **top-down** paradigm generates trees by percolating taxa down from the root one layer at a time. Specifically, we start with all taxa at the root node, at a depth of 0. At each step, we select one taxon with minimal depth and predict whether to percolate it down to the left or right. We stop percolating when a full bifurcating tree is formed.

3.2. Limitations of previous work

While all three generative paradigms have been considered in the context of phylogenetic inference, past methods are insufficiently flexible to parameterize complex posterior distributions over tree topologies. In particular, most generative models were used for summarizing MCMC samples or variational inference for a fixed set of sequences.

Of the three generative paradigms, the **bottom-up** paradigm is the only one that has previously been applied to simulation-based phylogenetic inference, in Phyloformer 2 (Blassel et al., 2025). However, this method parameterizes the posterior distribution with only one forward pass through the neural network, producing embeddings for each taxon and then predicting a tree by running a neighbor-joining algorithm on the fixed predicted embeddings. As a result, it is unable to represent all possible distributions over tree topologies. We summarize further limitations of these methods in Section A.1.

The **middle-out** paradigm has also been used to summarize MCMC posteriors for a single phylogenetic tree, in ARTree (Xie & Zhang, 2023) and ARTreeFormer (Xie et al., 2025a). These methods have only been used to summarize MCMC samples for a single set of sequences, and have a noisy objective since they impose an arbitrary order over the species (see Section 4.2). It is unclear how to efficiently scale these methods up to full simulation-based inference.

The **top-down** paradigm has been used to parameterize distributions over trees since at least 2013 (Larget, 2013). In particular Zhang & Matsen (2018) and Zhang & Matsen (2024) start with all taxa at the root and write a probability for each splitting into children of the root; the tree is then built up by iteratively splitting. However, to represent all possible splits this method would require an exponential number of parameters: at the root alone, there are $2^{N-1} - 1$ possible splits. In practice, they instead restrict support to

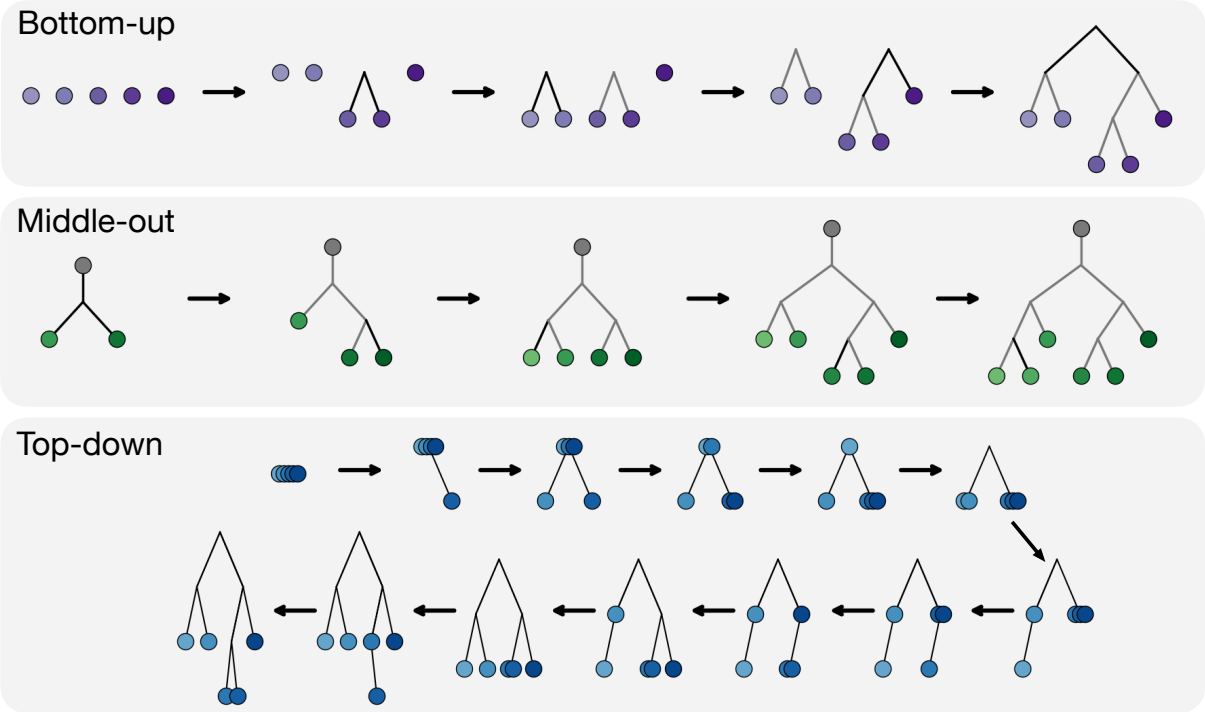


Figure 1. Illustration of three generative paradigms for trees. Each colored circle represents one taxon. For **bottom-up**, we start with each taxon as a separate subtree and repeatedly merge two subtrees until a single bifurcating tree is formed. Dark edges indicate which subtrees were merged in each step. For **middle-out**, we start with an unrooted tree containing two taxa and add one taxon at a time. The gray circle depicts the "pseudo-taxon" representing the root of the tree, and dark edges show which taxon was added in each step. For **top-down**, we start with all taxa at the root and percolate them downwards one step at a time until we reach a full bifurcating tree.

splits that were observed in MCMC posteriors (Xie & Zhang, 2023), and therefore cannot support the full combinatorial space of tree topologies.

We discuss alternative paradigms for generative models over trees and their shortcomings in Appendix B.

3.3. Achieving flexibility with iterative generative models

How can we develop generative models that are sufficiently flexible to represent arbitrary posterior distributions over tree topologies? We advocate for using *iterative generative models* that perform multiple forward passes through the model to gradually construct a tree one step at a time. Unlike past methods that use a single forward pass to parameterize the full posterior, performing multiple forward passes allows the model to represent arbitrarily complex distributions by conditioning on the intermediate state of the tree before predicting the next step.

Formally, we start the generative process with a structure T_M that depends only on the sequences, without any information about the real tree. We then use a model q_θ to build up the tree one step at a time by repeatedly sampling from the distribution $q_\theta(T_{m-1} \mid T_m, X_{1:N})$. We continue this process for M steps until we have generated the full tree T_0.

Figure 2 illustrates the shape of the model's predictions for a single step of generation under each of the paradigms. For the **bottom-up** paradigm, at each step, the model takes in a set of $m+1$ subtrees, and predicts a distribution of size $\binom{m+1}{2}$ over the next two subtrees to merge. For the **middle-out** paradigm, at each step the model conditions on the current tree and predicts where to attach each missing taxon using a distribution over the tree's edges. For the **top-down** paradigm, at each step the model conditions on the current state of the tree and predicts the probability of each taxon percolating left or right.

Interestingly, a model that naively outputs uniform predictions leads to a different distribution over tree topologies for each of the three paradigms. For **bottom-up** we obtain the Yule distribution, for **middle-out** we obtain the PDA distribution, and for **top-down** we obtain the "almost completely balanced" distribution described in (Aldous, 2001). This might suggest that our choice of generative paradigm depends on our desired prior over tree topologies. However, we show in Appendix C that all three paradigms can support a full spectrum from PDA to Yule with simple adjustments to the naive uniform predictions, meaning all are sufficiently flexible for our task.

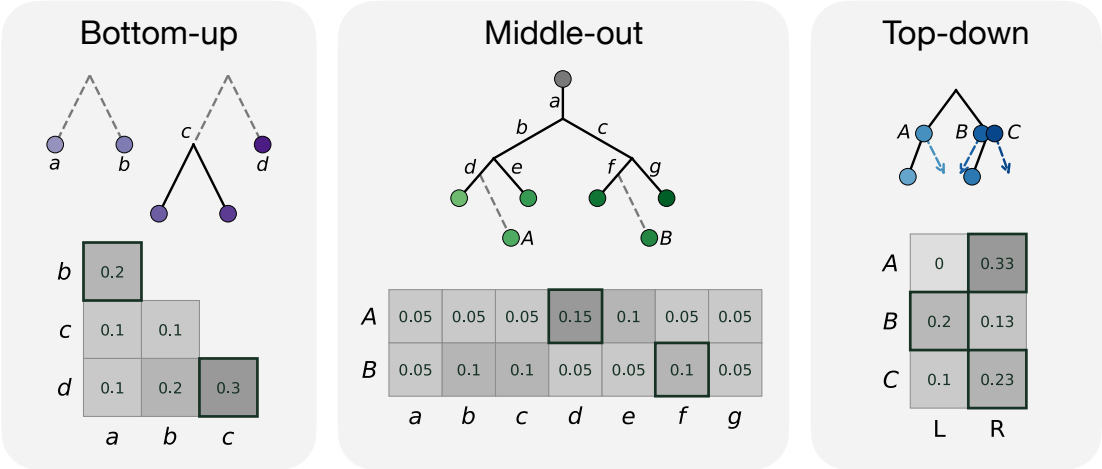


Figure 2. Illustration of sample model predictions for the three generative paradigms. For **bottom-up**, the dashed edges indicate the valid merges consistent with T_0, corresponding to the highlighted cells in the matrix of model predictions. For **middle-out**, the dashed edges indicate the correct attachment edges consistent with T_0, corresponding to the highlighted cells in the matrix of model predictions. For **top-down**, the arrows indicate the correct percolation directions consistent with T_0, corresponding to the highlighted cells in the matrix of model predictions. Taxon A is constrained to percolate right as T_0 is a bifurcating tree.

4. Efficiently training our iterative generative models

In this section, we describe how to efficiently train iterative generative models over trees using the discrete diffusion framework, explain our choices for the noising process in each paradigm, and provide details on our implementation.

4.1. Training iterative generative models with discrete diffusion

We want to train our models to maximize the likelihood of the simulated data. Letting $T_M, T_{M-1}, \ldots, T_1$ be the states of the tree while generating T_0, the likelihood is:

$$q_\theta(T_0 \mid X_{1:N}) = \sum_{T_1, T_2, \ldots, T_M} \prod_{m=1}^{M} q_\theta(T_{m-1} \mid T_m, X_{1:N}) \tag{1}$$

Unfortunately, directly optimizing such a likelihood is intractable as it requires marginalizing over all possible paths that reach the tree T_0. As the number of possible paths is typically combinatorial in N, this marginalization is prohibitively expensive.

Instead, we propose training the model q_θ using the discrete diffusion framework. Specifically, we define a noising process that generates paths $p(T_1, \ldots, T_M \mid T_0)$ by noising T_0; then we train our model to reverse these paths. We do so by optimizing an ELBO of the form $\log q_\theta(T_0 \mid X_{1:N}) \geq$

$$-\sum_{m=1}^{M} \mathbb{E}\mathrm{KL}(p(T_{m-1} \mid T_m, T_0) \,||\, q_\theta(T_{m-1} \mid T_m, X_{1:N})) \tag{2}$$

Training the model now involves simulating a training example $(T_0, X_{1:N})$, applying the noising process $p(T_m \mid T_0, m)$ for some number of noising steps m, and pushing the model's predictions on T_m closer to the true reverse process that recovers T_0. We review the derivation of the ELBO and cover related work on discrete diffusion in Appendix D.

4.2. Noising process for generative models over trees

To train a discrete diffusion model using Equation (2), we must define a noising process $p(T_m \mid T_0, m)$ and determine how to compute the target distribution $p(T_{m-1} \mid T_m, T_0)$.

For **bottom-up**, at the start of the noising process, we label the root node as "exposed". Then, each noising step involves removing an exposed internal node from the current graph, "exposing" its two children. We select the next node to remove in each step as:

$$p_{\text{BU}}(T_m = \texttt{remove}(T_{m-1}, \texttt{node}) \mid T_{m-1}) \propto \texttt{num_descendants}(\texttt{node})$$

We multiply by $\texttt{num_descendants}(\texttt{node})$ to get a tractable form for $p(T_{m-1} \mid T_m, T_0)$, described in Section E.2.

For **middle-out**, each noising step involves pruning one taxon from the tree and collapsing the edges connecting its two siblings. Formally, we have, for a taxon v:

$$p_{\text{MO}}(T_m = \texttt{prune}(T_{m-1}, v) \mid T_{m-1}) = \frac{1}{N - m + 1}$$

Here, the distribution $p(T_{m-1} \mid T_m, T_0)$ is again uniform over all valid attachments in T_m that are consistent with T_0.

Note that for a given taxon, there is only one attachment edge in T_m that is consistent with the ground truth T_0. Note also that our target has multiple non-zero entries corresponding to each of the taxa not in the current tree. ARTree (Xie & Zhang, 2023) instead imposed an arbitrary ordering over the sequences, resulting in a sparser target with higher variance.

For **top-down**, each noising step involves selecting a taxon of maximal depth and percolating it up one layer to its parent. The noising process continues until all taxa reach the root node, at T_M. Formally, for a tree T_{m-1} with maximum depth d, we have:

$$p_{\text{TD}}(T_m = \texttt{percolate_up}(T_{m-1}, \texttt{taxon}) \mid T_{m-1}) = \begin{cases} 0, & \texttt{depth}(\texttt{taxon}) < d \\ \frac{1}{\texttt{\# of taxa at depth}\, d}, & \text{otherwise} \end{cases}$$

The distribution $p(T_{m-1} \mid T_m, T_0)$ is uniform over all percolations in T_m that are consistent with T_0.

4.3. Implementation details

Equipped with the noising process $p(T_m \mid T_0, m)$ and target distribution $p(T_{m-1} \mid T_m, T_0)$ for each of the paradigms, we now describe two other implementation details to make training efficient.

Noise schedule In the ELBO from Equation (2), we use a uniform distribution over the number of noising steps m. However, in practice some noising steps may be much easier for the model to learn than others. Ideally, to train as efficiently as possible, we would like to sample training examples according to the difficulty, by adapting the noise schedule. In Section E.1, we describe how our choice of noise schedule mirrors the mutual information schedules of Austin et al. (2021).

Initializing at our prior distribution over trees As noted in Section 3.3, each generative paradigm is capable of representing a full spectrum of prior distributions over topologies $p(T_0)$, ranging from Yule to PDA, using particular weights for the predictions. We reparameterize our generative models by multiplying the neural network outputs with these weights, ensuring that the models are initialized at the desired prior distribution.

5. Experiment details

We now describe our experimental setup, including training data, architecture, and training details.

5.1. Simulated training data

We train our models on synthetic data simulated following a prior $p(T)$ and evolutionary model $p(X_{1:N} \mid T)$. Below we set $p(T)$ to be a PDA prior on trees to match the priors in MrBayes.

We consider three types of evolutionary models $p(X_{1:N} \mid T)$ for our experiments. First, we train our models on the simplest model of DNA evolution (**JC**). As described in Section 2.3, the JC model yields tractable likelihoods, allowing us to use MrBayes to obtain gold standard MCMC samples from the posterior. Next, we consider two more realistic evolutionary models: a more *complex model* accounting for natural selection in proteins (**CAT**) and a model of DNA evolution that has *intractable* likelihoods because of the presence of insertions and deletions (**InDel**).

We train our models to support datasets with between $N = 25$ and $N = 65$ species, and sequences of length $L = 300$ to $L = 2600$, based on the sizes of 8 classical evolutionary datasets (Zhang & Matsen, 2018). Our models can therefore be immediately applied to most evolutionary and immunological inference tasks on $1 - 5$ genes. We discuss how to scale to larger N and L in Section 7. Lastly, we randomly mask a subset of positions in the DNA sequences to reflect the missing information that is common in real datasets. We provide further details on the evolutionary models and our simulated data in Section G.1.

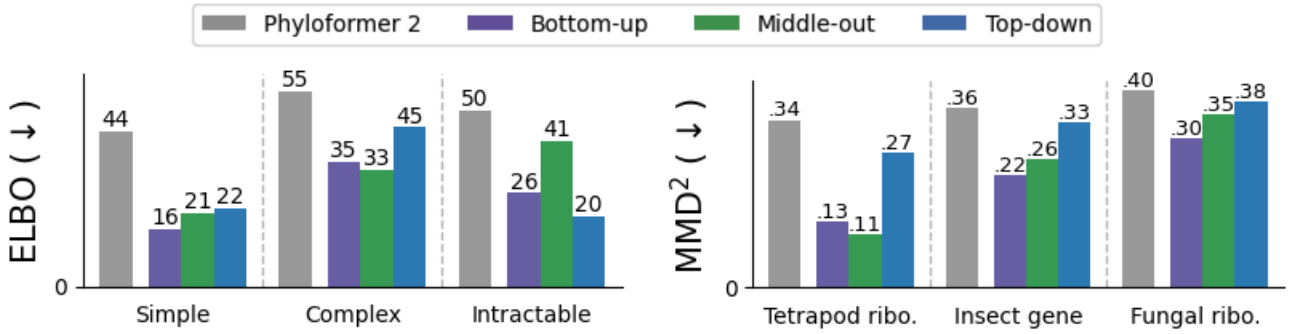
5.2. Model architecture and training details

Our model q_θ must condition on the sequences $X_{1:N}$ and current state of the tree T_m. To condition on the sequences, we adapt the EvoPF architecture from Phyloformer 2 (Blassel et al., 2025). EvoPF embeds N sequences of length L, $X_{1:N}$, into a tensor of size $[N, L, 256]$ and performs axial attention over the dimensions N and L separately. For scalability, we drop the quartic pairwise attention from EvoPF so our architecture has complexity $\mathcal{O}(NL^2)$. We provide full details in Appendix F.

We use AdamW with μP-rescaled per-layer learning rates (Yang et al., 2022) (base $\text{lr} = 0.03$), 1,000-step linear warmup, EMA 0.9998, gradient-norm clip 500, `bf16-true` precision, and an effective batch of 32 trees per step. Each model trains for 48 hours on 2 A100 GPUs, which is approximately 20,000 updates for each paradigm.

6. Results

In this section, we provide our empirical results. We compare against Phyloformer 2, the previous state-of-the-art method for simulation-based inference. First, we show that our models learn to fit simulated data much better than Phyloformer 2, as measured by ELBOs. Second, we show



(a) ELBOs *(b)* MMDs

Figure 3. **(Left)** Our models obtain substantially better ELBOs than Phyloformer 2. **(Right)** Our models reconstruct MCMC posteriors on real datasets more accurately than Phyloformer 2.

that our models match gold standard MCMC posteriors more accurately than Phyloformer 2. Lastly, we show that our models accurately infer the correct tree for the complex and intractable evolutionary models, improving accuracy over the best existing alternative.

6.1. Our models learn faster than previous state-of-the-art for simulation-based inference

In Figure 3a, we compare the ELBOs achieved by our models against Phyloformer 2. We find that across datasets and generative paradigms, our models consistently fit the data much better than Phyloformer 2. These results confirm that our flexible parameterization and efficient training via the discrete diffusion ELBO significantly improve the performance of simulation-based inference for phylogenetics. More generally, our work is the first to successfully apply the **top-down** and **middle-out** paradigms to this problem, and we show that both of these paradigms can far surpass the performance of the previous state-of-the-art.

Interestingly, different paradigms better fit different datasets. The superior performance of **bottom-up** on the simple data may reflect the bottom-up nature of the likelihood calculation on this data (Felsenstein, 1973). On the intractable data, **top-down** may perform the best because it can use InDel patterns to accurately define the deepest splits first. Finally, **middle-out** may perform the best on the complex data since it conditions on a fully resolved partial tree at each step, allowing it to best implicitly infer the hyperparameters of the evolutionary process. This suggests that the optimal paradigm for fitting a particular evolutionary process may be the one that matches its inductive biases.

6.2. Our models accurately reconstruct gold standard MCMC posteriors

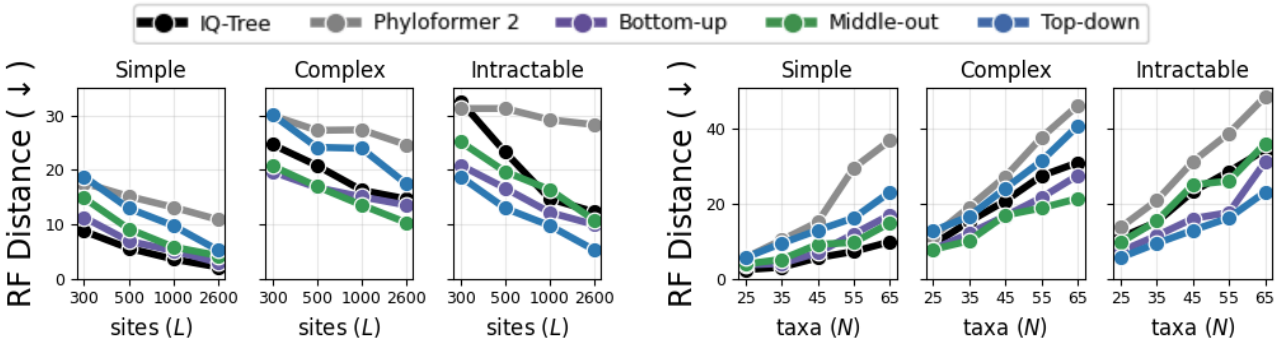
We now take three classical evolutionary datasets from Lakner et al. (2008) selecting a variety of N and L: tetrapod ribosomal DNA with $N = 27$ and $L = 1949$; insect *wingless* gene DNA with $N = 50$ and $L = 378$; and fungal ribosomal DNA with $N = 64$ and $L = 1008$. See Section G.2 for details. For each dataset, we infer 100 trees using our JC models, and compare the distribution of our inferred trees to the posterior estimated by MrBayes under the JC model. We remove constant sites in the sequence and shuffled the order of sites before inferring the tree using our models.

We compare the distributions using the MMD distance. MMD measures the average distance between trees sampled from separate distributions and subtracts the average distance between trees sampled from the same distribution, with the best MMD score of 0 achieved when the two distributions are identical. See Section G.4 for more details. In Figure 3b we report MMD scores between the posteriors of each model and the gold standard posterior samples from MrBayes. Our models accurately recover the posterior, obtaining significantly lower scores than the previous state-of-the-art Phyloformer 2.

6.3. Our models successfully infer trees for complex evolutionary models

For the CAT and InDel datasets, obtaining accurate MCMC posteriors is prohibitively expensive or impossible. To make predictions in these settings, practitioners often resort to point estimates using workhorse maximum likelihood methods such as IQ-Tree. Practitioners use these procedures despite knowing they are misspecified in hopes that the evidence still points to the correct tree.

For each dataset, we sample 30 simulated pairs of sequences and trees at various combinations of N and L. We then condition on the sequences and predict the tree topology by sampling eight trees from our models and taking a majority-rule clade consensus tree. We evaluate the Robinson-Foulds (RF) distance with respect to the true simulated tree, and



(a) $N = 45$ *(b)* $L = 500$

Figure 4. Our models often make more accurate predictions than IQ-Tree for the complex and intractable evolutionary models. **(Left)** We hold N fixed and vary the sequence length L. **(Right)** We hold L fixed and vary the number of taxa N.

compare against the maximum likelihood tree predicted by IQ-Tree with default settings. Note that IQ-Tree does not support the CAT or InDel models and is therefore misspecified in these settings.

We show our results for $N = 45$ and $L = 500$ in Figure 4. We see all of our models outperform Phyloformer 2, and that their sample quality reflects how well they fit the data (Fig. 3a). IQ-Tree performs well on the JC data where it is well-specified. However, on the CAT and InDel datasets, IQ-Tree ignores more complex effects leading to lower-quality predictions. In contrast, some of our models are able to leverage their knowledge about complex evolutionary processes and indels to make better predictions than IQ-Tree. In particular, our models more strongly outperform baselines in regimens with low data (small L) and complex posteriors (high N). Appendix H shows run-time comparisons, showing our models are faster than IQ-Tree for most N, L settings.

These results demonstrate that our models can give practitioners higher-quality point estimates of the tree under realistic evolutionary models. Moreover, in contrast with bespoke algorithms such as IQ-Tree, our models simply require the practitioner to provide simulated data.

7. Conclusion

We develop flexible deep generative models that can be trained to accurately model posteriors over phylogenetic trees. Our models are trained on simulated data only, allowing them to learn posteriors for more realistic evolutionary models where traditional Bayesian approaches are intractable. Our contributions open up promising directions for future work: building a universal default tree prior, building better models for inferring evolutionary parameters, and scaling to larger datasets.

A universal tree prior There are a plethora of evolutionary models proposed in the literature. In many cases, practitioners may not know which evolutionary model best explains their data, and resort to using a model that is a poor match for the data. For many protein and DNA evolution datasets, the `GTR+G4+I` prior in MrBayes (Huelsenbeck & Ronquist, 2001) acts as a default, and many evolutionary parameters come with common and default values used nearly universally for inference (Susko et al., 2018). Our framework opens the door for a phylogenetic inference foundation model: by training on simulated data from a variety of evolutionary models, the model automatically selects the appropriate evolutionary process as part of the posterior inference, and can be used across a large variety of applications. The foundation model can also be fine-tuned on simulated data when a particular evolutionary model is desired.

Inferring evolutionary parameters In this paper we focused on inference of the tree topology T alone. However, in many settings we are also interested in inferring other parameters, such as branch lengths. As our work addresses the combinatorially challenging part of the inference problem, it can be extended by applying modern Bayesian inference methods for Euclidean data to infer other parameters (Wang et al., 2015; Papamakarios & Murray, 2016; Fourment et al., 2024).

Larger scale inference We train our models on datasets with up to $N = 65$ sequences of length up to $L = 2600$. While these sizes are sufficient for many important phylogenetic tasks, certain applications such as pandemic data or full-genome analyses involve larger datasets. Future work could seek to extend our methods for these settings, such as supporting inference for large trees by combining our approach with super-tree methods (Warnow, 2018; Jiang et al., 2024), or combine inferred trees for individual genes into one tree for whole genomes (Mirarab et al., 2014).

References

Aldous, D. Probability distributions on cladograms. In Aldous, D. and Pemantle, R. (eds.), *Random Discrete Structures*, pp. 1–18, New York, NY, 1996. Springer New York. ISBN 978-1-4612-0719-1.

Aldous, D. J. Mixing time for a markov chain on cladograms. *Comb. Probab. Comput.*, 9(3):191–204, May 2000.

Aldous, D. J. Stochastic models and descriptive statistics for phylogenetic trees, from Yule to today. *Stat. Sci.*, 16 (1):23–34, February 2001.

Allen, B. L. and Steel, M. Subtree transfer operations and their induced metrics on evolutionary trees. *Ann. Comb.*, 5(1):1–15, June 2001.

Amin, A. N., Gruver, N., and Wilson, A. G. Why masking diffusion works: Condition on the jump schedule for improved discrete diffusion. In *Frontiers in Probabilistic Inference: Learning meets Sampling*, April 2025.

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and Van Den Berg, R. Structured denoising diffusion models in discrete state-spaces. *Adv. Neural Inf. Process. Syst.*, 34: 17981–17993, 2021.

Banos, H., Wong, T. K. F., Daneau, J., Susko, E., Minh, B. Q., Lanfear, R., Brown, M. W., Eme, L., and Roger, A. J. GTRpmix: A linked general time-reversible model for profile mixture models. *Mol. Biol. Evol.*, 41(9): msae174, September 2024.

Baron, E., Amin, A. N., Weitzman, R., Marks, D. S., and Wilson, A. G. A diffusion model to shrink proteins while maintaining their function. *ICLR*, 2026.

Bertoin, J. Exchangeable coalescents. Technical report, ETH Z¨urich, 2010.

Billera, L. J., Holmes, S. P., and Vogtmann, K. Geometry of the space of phylogenetic trees. *Adv. Appl. Math.*, 27 (4):733–767, November 2001.

Blassel, L., Boussau, B., Lartillot, N., and Jacob, L. Likelihood-free inference of phylogenetic tree posterior distributions. *arXiv [q-bio.PE]*, October 2025.

Bloom, J. D. The data are insufficient to confidently root the SARS-CoV-2 phylogenetic tree. *Mol. Biol. Evol.*, 42 (6):msaf118, June 2025.

Bohde, M., Manjrekar, M., Wang, R., Ji, S., and Coley, C. W. DiffMS: Diffusion generation of molecules conditioned on mass spectra. *arXiv [cs.LG]*, February 2025.

Bordewich, M. and Semple, C. On the computational complexity of the rooted subtree prune and regraft distance. *Ann. Comb.*, 8(4):409–423, January 2005.

Brower, A. V. Phylogenetic relationships among the nymphalidae (lepidoptera) inferred from partial sequences of the wingless gene. *Proc. Biol. Sci.*, 267(1449): 1201–1211, June 2000.

Campbell, A., Benton, J., De Bortoli, V., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, October 2022.

Chen, A., Chlenski, P., Munyuza, K., Moretti, A. K., Naesseth, C. A., and Pe'er, I. Variational combinatorial sequential Monte Carlo for Bayesian phylogenetics in hyperbolic space. *arXiv [cs.LG]*, July 2025.

Crotty, S. M., Minh, B. Q., Bean, N. G., Holland, B. R., Tuke, J., Jermiin, L. S., and Haeseler, A. V. GHOST: Recovering historical signal from heterotachously evolved sequence alignments. *Syst. Biol.*, 69(2):249–264, March 2020.

DasGupta, B., He, X., Jiang, T., Li, M., Tromp, J., and Zhang, L. On computing the nearest neighbor interchange distance. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pp. 125–143. American Mathematical Society, Providence, Rhode Island, December 2000.

Dax, M., Green, S. R., Gair, J., Macke, J. H., Buonanno, A., and Schölkopf, B. Real-time gravitational wave science with neural posterior estimation. *Phys. Rev. Lett.*, 127 (24):241103, December 2021.

DeWitt, 3rd, W. S., Mesin, L., Victora, G. D., Minin, V. N., and Matsen, 4th, F. A. Using genotype abundance to improve phylogenetic inference. *Mol. Biol. Evol.*, 35(5): 1253–1265, May 2018.

Diaconis, P. and Holmes, S. Random walks on trees and matchings. *Electron. J. Probab.*, 7(none):1–17, January 2002.

Dinh, V., Bilge, A., Zhang, C., and Matsen, IV, F. A. Probabilistic path Hamiltonian Monte Carlo. *arXiv [q-bio.PE]*, February 2017.

Drummond, A. J., Rambaut, A., Shapiro, B., and Pybus, O. G. Bayesian coalescent inference of past population dynamics from molecular sequences. *Mol. Biol. Evol.*, 22 (5):1185–1192, May 2005.

Duan, C., Zang, Z., Li, S., Xu, Y., and Li, S. Z. PhyloGen: Language model-enhanced phylogenetic inference via graph structure generation. *38th Conference on Neural Information Processing Systems*, December 2024.

Felsenstein, J. Maximum likelihood and minimum-steps methods for estimating evolutionary trees from data on discrete characters. *Syst. Biol.*, 22(3):240–249, September 1973.

Felsenstein, J. *Inferring Phylogenies*. 2004.

Felsenstein, J. and Churchill, G. A. A hidden Markov model approach to variation among sites in rate of evolution. *Mol. Biol. Evol.*, 13(1):93–104, January 1996.

Fletcher, W. and Yang, Z. INDELible: a flexible simulator of biological sequence evolution. *Mol. Biol. Evol.*, 26(8): 1879–1888, August 2009.

Ford, D. J. Probabilities on cladograms: introduction to the alpha model. *arXiv [math.PR]*, November 2005.

Fourment, M., Macaulay, M., Swanepoel, C. J., Ji, X., Suchard, M. A., and Matsen, IV, F. A. Torchtree: flexible phylogenetic model development and inference using PyTorch. *arXiv [q-bio.PE]*, June 2024.

Gavryushkin, A. and Drummond, A. J. The space of ultrametric phylogenetic trees. *J. Theor. Biol.*, 403:197–208, August 2016.

Goldman, N. and Yang, Z. A codon-based model of nucleotide substitution for protein-coding DNA sequences. *Mol. Biol. Evol.*, 11(5):725–736, September 1994.

Gonçalves, P. J., Lueckmann, J.-M., Deistler, M., Nonnenmacher, M., Öcal, K., Bassetto, G., Chintaluri, C., Podlaski, W. F., Haddad, S. A., Vogels, T. P., Greenberg, D. S., and Macke, J. H. Training deep neural density estimators to identify mechanistic models of neural dynamics. *Elife*, 9(e56261):e56261, September 2020.

Havasi, M., Karrer, B., Gat, I., and Chen, R. T. Q. Edit flows: Flow matching with edit operations. June 2025.

Hedges, S. B., Moberg, K. D., and Maxson, L. R. Tetrapod phylogeny inferred from 18S and 28S ribosomal RNA sequences and a review of the evidence for amniote relationships. *Mol. Biol. Evol.*, 7(6):607–633, November 1990.

Hollmann, N., Müller, S., Purucker, L., Krishnakumar, A., Körfer, M., Hoo, S. B., Schirrmeister, R. T., and Hutter, F. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, January 2025.

Holmes, I. and Bruno, W. J. Evolutionary HMMs: A Bayesian approach to multiple alignment. *Bioinformatics*, 17(9):803–820, September 2001.

Huelsenbeck, J. P. and Ronquist, F. MRBAYES: Bayesian inference of phylogenetic trees. *Bioinformatics*, 17(8): 754–755, August 2001.

Huelsenbeck, J. P., Rannala, B., and Masly, J. P. Accommodating phylogenetic uncertainty in evolutionary studies. *Science*, 288(5475):2349–2350, June 2000.

Jiang, Y., Balaban, M., Zhu, Q., and Mirarab, S. DEPP: Deep learning enables extending species trees using single genes. *Syst. Biol.*, 72(1):17–34, May 2023.

Jiang, Y., McDonald, D., Perry, D., Knight, R., and Mirarab, S. Scaling DEPP phylogenetic placement to ultra-large reference trees: a tree-aware ensemble approach. *Bioinformatics*, 40(6):btae361, June 2024.

Jukes, T. H. and Cantor, C. R. Evolution of protein molecules. In *Mammalian Protein Metabolism*, pp. 21–132. Elsevier, 1969.

Kingma, D. P., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. In *NeurIPS 2021*, July 2021.

Lakner, C., van der Mark, P., Huelsenbeck, J. P., Larget, B., and Ronquist, F. Efficiency of markov chain monte carlo tree proposals in bayesian phylogenetics. *Syst. Biol.*, 57 (1):86–103, February 2008.

Lambert, A. and Stadler, T. Birth-death models and coalescent point processes: the shape and probability of reconstructed phylogenies. *Theor. Popul. Biol.*, 90:113–128, December 2013.

Larget, B. The estimation of tree posterior probabilities using conditional clade probability distributions. *Syst. Biol.*, 62(4):501–511, July 2013.

Lartillot, N. and Philippe, H. A Bayesian mixture model for across-site heterogeneities in the amino-acid replacement process. *Mol. Biol. Evol.*, 21(6):1095–1109, June 2004.

Lartillot, N., Rodrigue, N., Stubbs, D., and Richer, J. PhyloBayes MPI: phylogenetic reconstruction with infinite mixtures of profiles in a parallel environment. *Syst. Biol.*, 62(4):611–615, July 2013.

Lassmann, T. Kalign 3: multiple sequence alignment of large data sets. *Bioinformatics*, 36(6):1928–1929, October 2019.

Le, S. Q. and Gascuel, O. An improved general amino acid replacement matrix. *Mol. Biol. Evol.*, 25(7):1307–1320, July 2008.

Le, S. Q., Dang, C. C., and Gascuel, O. Modeling protein evolution with several amino acid replacement matrices depending on site rates. *Mol. Biol. Evol.*, 29(10):2921–2936, October 2012.

Ly-Trong, N., Naser-Khdour, S., Lanfear, R., and Minh, B. Q. AliSim: A fast and versatile phylogenetic sequence simulator for the genomic era. *Mol. Biol. Evol.*, 39(5): msac092, May 2022.

Löhr, W., Mytnik, L., and Winter, A. The aldous chain on cladograms in the diffusion limit. *arXiv [math.PR]*, May 2018.

Lüdke, D., Raventós, E. R., Kollovieh, M., and Günnemann, S. Unlocking point processes through point set diffusion. *arXiv [cs.LG]*, October 2024.

Matsen, 4th, F. A. and Ralph, P. L. Enabling inference for context-dependent models of mutation by bounding the propagation of dependency. *J. Comput. Biol.*, 29(8): 802–824, August 2022.

Mimori, T. and Hamada, M. GeoPhy: Differentiable phylogenetic inference via geometric gradients of tree topologies. *37th Conference on Neural Information Processing Systems*, July 2023.

Mirarab, S., Reaz, R., Bayzid, M. S., Zimmermann, T., Swenson, M. S., and Warnow, T. ASTRAL: genome-scale coalescent-based species tree estimation. *Bioinformatics*, 30(17):i541–8, September 2014.

Moretti, A. K., Zhang, L., Naesseth, C. A., Venner, H., Blei, D., and Pe'er, I. Variational combinatorial sequential Monte Carlo methods for Bayesian phylogenetic inference. *arXiv [stat.ML]*, (Uai), May 2021.

Nesterenko, L., Blassel, L., Veber, P., Boussau, B., and Jacob, L. Phyloformer: Fast, accurate and versatile phylogenetic reconstruction with deep neural networks. *Bioinformatics*, (biorxiv;2024.06.17.599404v1), June 2024.

Owen, M. and Provan, J. S. A fast algorithm for computing geodesic distances in tree space. *IEEE/ACM transactions on computational biology and bioinformatics*, 8(1), January 2011.

Papamakarios, G. and Murray, I. Fast ϵ-free inference of simulation models with Bayesian conditional density estimation. *30th Conference on Neural Information Processing Systems*, 2016.

Pisani, D., Pett, W., Dohrmann, M., Feuda, R., Rota-Stabelli, O., Philippe, H., Lartillot, N., and Wörheide, G. Genomic data do not support comb jellies as the sister group to all other animals. *Proc. Natl. Acad. Sci. U. S. A.*, 112(50): 15402–15407, December 2015.

Prillo, S., Deng, Y., Boyeau, P., Li, X., Chen, P.-Y., and Song, Y. S. CherryML: scalable maximum likelihood estimation of phylogenetic models. *Nat. Methods*, 20(8): 1232–1236, August 2023.

Richman, H., Zhang, C., and Matsen, IV, F. A. Vector encoding of phylogenetic trees by ordered leaf attachment. *arXiv [q-bio.PE]*, March 2025.

Rossman, A. Y., McKemy, J. M., Pardo-Schultheiss, R. A., and Schroers, H.-J. Molecular studies of the bionectriaceae using large subunit rDNA sequences. *Mycologia*, 93(1):100–110, January 2001.

Sainudiin, R. and Veber, A. A beta-splitting model for evolutionary trees. *arXiv [math.PR]*, November 2015.

Saitou, N. and Nei, M. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4(4):406–425, July 1987.

Sarkar, R. Low distortion delaunay embedding of trees in hyperbolic plane. In *Graph Drawing*, Lecture Notes in Computer Science, pp. 355–366. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

Seplyarskiy, V. B., Soldatov, R. A., Koch, E., McGinty, R. J., Goldmann, J. M., Hernandez, R. D., Barnes, K., Correa, A., Burchard, E. G., Ellinor, P. T., McGarvey, S. T., Mitchell, B. D., Vasan, R. S., Redline, S., Silverman, E., Weiss, S. T., Arnett, D. K., Blangero, J., Boerwinkle, E., He, J., Montgomery, C., Rao, D. C., Rotter, J. I., Taylor, K. D., Brody, J. A., Chen, Y.-D. I., de las Fuentes, L., Hwu, C.-M., Rich, S. S., Manichaikul, A. W., Mychaleckyj, J. C., Palmer, N. D., Smith, J. A., Kardia, S. L. R., Peyser, P. A., Bielak, L. F., O'Connor, T. D., Emery, L. S., NHLBI Trans-Omics for Precision Medicine (TOPMed) Consortium, TOPMed Population Genetics Working Group, Gilissen, C., Wong, W. S. W., Kharchenko, P. V., and Sunyaev, S. Population sequencing data reveal a compendium of mutational processes in the human germ line. *Science*, 373(6558):1030–1035, August 2021.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. K. Simplified and generalized masked diffusion for discrete data. *arXiv [cs.LG]*, June 2024.

Siepel, A. and Haussler, D. Phylogenetic estimation of context-dependent substitution rates by maximum likelihood. *Mol. Biol. Evol.*, 21(3):468–488, March 2004.

Strimmer, K. and von Haeseler, A. Quartet puzzling: A quartet maximum-likelihood method for reconstructing tree topologies. *Mol. Biol. Evol.*, 13(7):964–969, September 1996.

Su, J. and Jiao, X. Quartformer: An accurate deep learning framework for phylogenetic tree construction. *Bioinformatics*, (biorxiv;2025.04.05.646867v1), April 2025.

Suchard, M. A., Lemey, P., Baele, G., Ayres, D. L., Drummond, A. J., and Rambaut, A. Bayesian phylogenetic and phylodynamic data integration using BEAST 1.10. *Virus Evol.*, 4(1):vey016, January 2018.

Susko, E., Lincker, L., and Roger, A. J. Accelerated estimation of frequency classes in site-heterogeneous profile mixture models. *Mol. Biol. Evol.*, 35(5):1266–1283, May 2018.

Suvorov, A., Hochuli, J., and Schrider, D. R. Accurate inference of tree topologies from multiple sequence alignments using deep learning. *Syst. Biol.*, 69(2):221–233, March 2020.

Tamura, K., Battistuzzi, F. U., Billing-Ross, P., Murillo, O., Filipski, A., and Kumar, S. Estimating divergence times in large molecular phylogenies. *Proc. Natl. Acad. Sci. U. S. A.*, 109(47):19333–19338, November 2012.

Tan, G., Muffato, M., Ledergerber, C., Herrero, J., Goldman, N., Gil, M., and Dessimoz, C. Current methods for automated filtering of multiple sequence alignments frequently worsen single-gene phylogenetic inference. *Syst. Biol.*, 64(5):778–791, September 2015.

Vignac, C., Krawczuk, I., Siraudin, A., Wang, B., Cevher, V., and Frossard, P. DiGress: Discrete denoising diffusion for graph generation. *arXiv [cs.LG]*, September 2022.

Wang, H.-C., Spencer, M., Susko, E., and Roger, A. J. Testing for covarion-like evolution in protein sequences. *Mol. Biol. Evol.*, 24(1):294–305, January 2007.

Wang, L., Bouchard-Côté, A., and Doucet, A. Bayesian phylogenetic inference using a combinatorial sequential Monte Carlo method. *J. Am. Stat. Assoc.*, 110(512):1362–1374, October 2015.

Warnow, T. Supertree construction: Opportunities and challenges. *arXiv [q-bio.PE]*, May 2018.

Whelan, N. V. and Halanych, K. M. Who let the CAT out of the bag? accurately dealing with substitutional heterogeneity in phylogenomic analyses. *Syst. Biol.*, 66 (2):232–255, March 2017.

Whelan, N. V., Kocot, K. M., Moroz, T. P., Mukherjee, K., Williams, P., Paulay, G., Moroz, L. L., and Halanych, K. M. Ctenophore relationships and their placement as the sister group to all other animals. *Nat. Ecol. Evol.*, 1 (11):1737–1746, November 2017.

Wong, T. K. F., Ly-Trong, N., Ren, H., Demotte, P., Baños, H., Roger, A. J., Susko, E., Bielow, C., De Maio, N., Goldman, N., Hahn, M. W., dos Reis, M., Sy Vinh, L., Huttley, G., Lanfear, R., and Minh, B. Q. IQ-TREE 3: Phylogenomic inference software using complex evolutionary models. *Mol. Biol. Evol.*, (msag117):msag117, May 2026.

Woodman, W. M. and Nye, T. M. W. Brownian motion, bridges and bayesian inference in phylogenetic tree space. *arXiv [stat.ME]*, June 2025.

Xie, T. and Zhang, C. ARTree: A deep autoregressive model for phylogenetic inference. *arXiv [q-bio.PE]*, October 2023.

Xie, T., Mao, Y., and Zhang, C. Artreeformer: A faster attention-based autoregressive model for phylogenetic inference. *PLOS Computational Biology*, 21(12):e1013768, 2025a. doi: 10.1371/journal.pcbi.1013768. URL https://doi.org/10.1371/journal.pcbi.1013768.

Xie, T., Richman, H., Gao, J., Matsen, IV, F. A., and Zhang, C. PhyloVAE: Unsupervised learning of phylogenetic trees via variational autoencoders. *arXiv [stat.ML]*, February 2025b.

Yang, G., Hu, E. J., Babuschkin, I., Sidor, S., Liu, X., Farhi, D., Ryder, N., Pachocki, J., Chen, W., and Gao, J. Tensor programs V: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv [cs.LG]*, March 2022.

Yang, Z. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: approximate methods. *J. Mol. Evol.*, 39(3):306–314, September 1994.

Youssef, N., Susko, E., and Bielawski, J. P. Consequences of stability-induced epistasis for substitution rates. *Mol. Biol. Evol.*, 37(11):3131–3148, November 2020.

Zhang, C. Learnable topological features for phylogenetic inference via graph neural networks. *arXiv [stat.ML]*, February 2023.

Zhang, C. and Matsen, IV, F. A. Generalizing tree probability estimation via Bayesian networks. *arXiv [stat.AP]*, (NeurIPS):1444–1453, May 2018.

Zhang, C. and Matsen, IV, F. A. A variational approach to Bayesian phylogenetic inference. *J. Mach. Learn. Res.*, 25:1–56, 2024.

Zhang, X., Ding, S., Yu, C., Zhao, J., and Bu, D. Accurate and efficient phylogenetic inference through end-to-end deep learning. *Bioinformatics*, (biorxiv;2025.09.30.679045v1), October 2025.

Zhang, Y., Yang, D., and Liao, R. SymmetricDiffusers: Learning discrete diffusion on finite symmetric groups. *arXiv [cs.LG]*, October 2024.

Zhou, M., Yan, Z., Layne, E., Malkin, N., Zhang, D., Jain, M., Blanchette, M., and Bengio, Y. PhyloGFN: Phylogenetic inference with generative flow networks. *International Conference of Learning Representations*, 2024.

A. Related works on machine learning methods for phylogenetic inference

There are many relevant works that apply machine learning methods for Bayesian phylogenetic inference. Most of these methods seek to improve upon MCMC methods for Bayesian inference, which are applicable when likelihoods are tractable. These include Hamiltonian Monte Carlo (Dinh et al., 2017), sequential Monte Carlo (Wang et al., 2015; Moretti et al., 2021; Chen et al., 2025), variational inference (Zhang & Matsen, 2024), and pseudo-likelihood methods (Prillo et al., 2023). These methods are particularly powerful for inferring the evolutionary parameters, since they live in Euclidean space and therefore are amenable to standard Bayesian inference techniques (Fourment et al., 2024).

When likelihood evaluation is expensive or impossible, we cannot perform Bayesian inference with the above methods. Instead we can simulate data from our prior and then attempt to learn the tree. This was framed as a regression problem by Suvorov et al. (2020) and re-framed as learning the posterior distribution $p(T \mid X_{1:N})$ by Phyloformer 2 (Blassel et al., 2025).

To model the posterior accurately, we require a flexible generative model over trees. Although many generative models over trees have been proposed for variational likelihood-based inference (VI) (Zhang & Matsen, 2024), summarizing distributions over trees (Xie et al., 2025b), or likelihood-free inference (Suvorov et al., 2020), most existing methods are problematic for the task of simulation-based inference for several reasons:

- They do not learn a full posterior. Methods that predict quartets (Suvorov et al., 2020; Su & Jiao, 2025) or distance matrices (Jiang et al., 2023; Nesterenko et al., 2024); or use a heuristic loss (Zhang et al., 2025) only learn to make accurate point predictions.
- They are not scalable. It is not clear how to efficiently learn energy-based models (Zhang, 2023) for datasets larger than a few short sequences.
- They require a tractable likelihood to train. Integrating the continuous encodings of Mimori & Hamada (2023) and Duan et al. (2024) and training subsplit networks with restricted support (Zhang & Matsen, 2018) is only possible when optimizing a reverse KL. Also, the GFlowNet framework of Zhou et al. (2024) requires a reward model.

A.1. Improvements upon Phyloformer 2

Phyloformer 2 (Blassel et al., 2025) is the previous state-of-the-art method for simulation-based inference for Bayesian phylogenetic inference. Our work improves upon Phyloformer 2 in several significant ways:

1. **Flexibility:** Phyloformer 2 uses a neural network backbone they call EvoPF that predicts embeddings for each species, followed by a module they call BayesNJ which parameterizes the distribution over topologies conditioned the embeddings. BayesNJ uses a symmetric bilinear layer to convert the embeddings into a probability distribution over each merge, using the same embeddings for the full generation process. However, as noted in Section 3.2 of (Blassel et al., 2025), in general the posterior merge probabilities at a given step might depend on the previous merges. Our models unlock this expressivity by using iterative forward passes through the network, each conditioning on the current state of the tree.
2. **Reliance on canonical merge order**: Another challenge faced by Phyloformer 2 is that it is intractable to sum over the likelihoods of all possible merge orders consistent with a given topology T_0. To solve this, they introduce a canonical merge order that depends on the branch lengths of the tree. As a result, the training objective is sparse, as there is only one valid merge at each generation step, similar to the main limitation of ARTree (Xie & Zhang, 2023).
3. **Initialization at Yule**: As discussed in Section 3.3, predicting uniform probabilities over merges induces a Yule distribution over tree topologies. Therefore, Phyloformer 2 is biased towards the Yule distribution at random initialization. However, in practice, we often wish to set a different prior over tree topologies, such as PDA. In Section 4.3, we explain how we re-paramaterize our neural network predictions to achieve the desired prior distribution over topologies at random initialization.
4. **Computationally expensive architecture**: The EvoPF architecture used in Phyloformer 2 includes attention over pairs of species, inducing time and memory costs of $\mathcal{O}(N^4)$. We remove the quartic pairwise attention, allowing for $O(N^2)$ runtime and memory instead.
5. **Restricted generative paradigm**: Phyloformer 2 only considers the **bottom-up** paradigm for generating trees. Our work also achieves state-of-the-art performance with the **middle-out** and **top-down** paradigms.

B. Other potential paradigms for generative models over trees

In our work, we consider three natural paradigms for generative models over trees: **bottom-up**, **middle-out**, and **top-down**. Here, we list other possible paradigms for defining generative models over trees and discuss the obstacles of using them for flexible simulation-based inference.

Table A1. Summary of alternative generative modeling approaches for trees and their limitations.

Approach	Status or challenge
Splits and edge contractions	analogous to **top-down**
Leaf delete-regraft	analogous to **middle-out**
Add one leaf at a time and quarter puzzling	analogous to **middle-out**
Coalescents and neighbor joining	analogous to **bottom-up**
Dyck paths and permutations	models planar, unlabeled trees
Hyperbolic embeddings	models planar trees
NNI, SPR, and TBR moves	tractability of conditionals
Tree matchings	tractability of conditionals
BHV space	tractability of conditionals

B.1. Paradigms that recover our models

The following paradigms are practical for simulation-based inference, and are analogous to the three paradigms which we consider.

Splits and edge contractions Many distributions of trees start with all species starting at the root, and then splitting left or right at a certain rate (Aldous, 2001; Ford, 2005); these can include birth-death models. Indeed this was explored for building generative models in Larget (2013) and Zhang & Matsen (2018). However, the number of possible splits of n species is 2^{n-1}; it is therefore a challenge to parameterize a distribution over splits in a flexible way. Indeed Larget (2013) and Zhang & Matsen (2018) had to restrict the possible splits to those seen in an MCMC run. Similarly, one could imagine corrupting trees by contracting random edges rather than just leaf edges; the de-corruption process would need to deal with the same challenge of parameterizing a distribution of splits of objects. One could parameterize such a distribution by gradually having nodes split one way or another using the diffusion method of Lüdke et al. (2024). This recovers the **top-down** paradigm we explore.

Leaf delete-regraft The "Aldous chain" involves tree corruptions via pruning and regrafting individual leaves (Aldous, 2000). Extensive theory on the process relies on the theory of random Dyck paths (Löhr et al., 2018), which we discuss below. We are not sure if $p(T_m \mid T_0)$ is tractable, however we note that schedule-conditioning this model as described in Amin et al. (2025) leads to something analogous to ARTree. In particular, Amin et al. (2025) advocates improving discrete diffusion by keeping track of how many times each token in a sequence has been corrupted during diffusion. But once a leaf has been corrupted, it no longer contains any information about its original position. Therefore, the task of predicting $q_\theta(T_{m-1} \mid T_m, \vec{s}_M)$, where $\vec{s}_M$ is an indicator of how many times each leaf has been corrupted, simply involves taking the subtree of un-noised leaves and predicting which edge any noised leaf falls in that tree, recovering the exact prediction problem from **middle-out**.

Add one leaf at a time and quartet puzzling Sainudiin & Veber (2015) and Ford (2005) describe distributions of trees in which one leaf is added at a time; these differ by introducing different hyperparameters and latent variables, and possibly disallowing addition to edges not connected to a leaf. Integer vector encodings of trees describe the order in which individual leaves are attached (Richman et al., 2025). Quartet puzzling is also an algorithm that adds one leaf at a time to a tree based on likelihoods of "quartet" splits (Strimmer & von Haeseler, 1996). These approaches are analogous to the **middle-out** paradigm we explore.

Coalescents and neighbor joining One of the most popular models for for phylogenetic trees is the coalescent (Bertoin, 2010). And one of the most popular algorithms for inferring a tree form a distance matrix is the neighbor-joining algorithm (Saitou & Nei, 1987). These involve joining subtrees at some rate, which is equivalent to the **bottom-up** paradigm we explore.

B.2. Paradigms that model the wrong types of trees

Alternatively, some paradigms are not applicable for our problem since they model the wrong types of trees. We are interested in modeling *non-planar* trees (which do not distinguish between left and right branches) and *labeled* trees (which distinguish the leaves).

Dyck paths and permutations Every labeled planar tree – that is, a tree distinguishing left and right children – can be uniquely described by a Dyck path and a permutation of $\{1, \ldots, N\}$, suggesting we could model trees over these two objects (Aldous, 2000). The coalescent point process (Lambert & Stadler, 2013) is another clever approach to efficiently generating trees; every labeled planar tree with branch lengths is characterized by a vector in $(0, 1)^{N-1}$ and a permutation. One issue with this approach is that it is not clear that modeling permutations or Dyck paths is simpler than modeling trees. Another issue is that we wish to model non-planar trees, and each non-planar tree admits 2^{N-1} planar representations. Stochastically choosing a planar representation for each tree would significantly increase variance when training, but canonically choosing a planar representation would introduce discontinuities or artificial structure.

Hyperbolic embeddings Sarkar (2012) shows how to embed trees into hyperbolic space with arbitrarily small distortion. Putting the distortion aside, the embedding is not unique: rotations, translations, and the ordering of cones emanating from each internal node do not affect the tree distance matrix. In particular, the ordering of cones can be used to define a planarity of the tree, showing that the symmetries of this embedding cause problems at least as bad as those of the Dyck paths and permutations above.

B.3. Paradigms that do not admit tractable conditionals for discrete diffusion training

The following paradigms are poor choices for simulation-based inference since they do not allow for efficient training with a discrete diffusion objective as explained in Section 4.1. Specifically, to train a diffusion model as in Campbell et al. (2022), we need the conditionals $p(T(t) \mid T(0))$ to be tractable for any time t, or to train a diffusion model as in Amin et al. (2025), we need $p(T_m \mid T_0)$ to be tractable for any number M, where T_m is the tree after m corruptions[1]. Many stochastic processes on trees are not suitable for a generative model because these conditionals are intractable.

NNI, SPR, and TBR moves Many MCMC methods explore tree-space with proposals based on nearest neighbor interchange (NNI), subtree prune-and-regraft (SPR), and tree bisection and reconnection (TBR) moves. However, calculating whether two trees T_1, T_2 have NNI, SPR, or TBR distance less than or equal to M is NP-Hard (DasGupta et al., 2000; Bordewich & Semple, 2005; Allen & Steel, 2001). Therefore, if we corrupt our trees using these moves, simply determining if $p(T_m = T' \mid T_0 = T) > 0$ is NP-Hard, making it likely intractable. We conjecture that $p(T_m = T' \mid T_0 = T)$ is similarly challenging to calculate. Meanwhile, other tree distances which are easier to calculate, like RF distance, are not associated with any particular sequence of tree operations.

Tree matchings Diaconis & Holmes (2002) showed that trees are in bijection with pairings of the numbers $\{1, \ldots, 2N-2\}$. They characterized the eigenspectrum of the action of the symmetric group on this representation, finding deep algebraic structure. However, the exact formulas for calculating $q_\theta(T_m \mid T_0)$, while "solved", are not easily computable in practice.

BHV space On the continuous Billera-Holmes-Vogtmann tree space (Billera et al., 2001), where topologies change via NNI moves, the remarkable theorem of Owen & Provan (2011) demonstrates that geodesic distances are still computable in polynomial time. Nevertheless, recent work has highlighted the intractability of calculating the densities of diffusion processes on this space (Woodman & Nye, 2025); tractable distributions meanwhile (GGF of Woodman & Nye (2025)) have restricted support. It is not clear if other continuous parameterizations, like that of Gavryushkin & Drummond (2016), have more tractable diffusions.

[1] SCUD (Amin et al., 2025) models can technically be trained with only empirical samples from $p(T_m \mid T_{m-1}, T_0)$. However this training method sacrifices a closed-form ELBO and has extremely high variance in its gradient estimates (Baron et al., 2026).

C. Expressivity of sampling paradigms

Here we prove small modifications to the logits can allow all three paradigms to generate trees from PDA to Yule, the spectrum of realistic trees.

C.1. Bottom-up

Consider a process that generates a tree T_0 by starting from T_{N-1}, a forest of N singleton components, and building up to a complete tree. At each step, the process merges two components of the forest T_m with a and b leaves to produce T_{m-1}, selecting this pair with probability proportional to $\gamma(a+b-2)+1$, for some parameter $\gamma \geq 0$. Each merge creates one internal node of the final tree T_0; the first merge creates the deepest internal node and the last creates the root.

Reversing the merge order, we obtain an ordering $v_1, v_2, \ldots, v_{N-1}$ of the internal nodes of T_0 in which every ancestor appears before its descendants—that is, a linear extension of $\texttt{int}(T_0)$. It will be useful below that paths are uniform over linear extensions.

Proposition C.1. *When $\gamma = 0$, $p(T_0)$ is a Yule distribution over trees; when $\gamma \to \infty$, $p(T_0)$ concentrates on combs. When $\gamma = 1$, $p(T_0) = 1/(2N-3)!!$ is the PDA (uniform) distribution over topologies, and each path is equally likely conditioned on the topology: $p(T_1, \ldots, T_{N-1} \mid T_0) = 1/|\mathrm{LE}(\texttt{int}(T_0))|$.*

Proof. The $\gamma = 0$ case is Kingman's coalescent, which is well known to produce the Yule distribution (Aldous, 2001); the $\gamma \to \infty$ case follows since the merge probability concentrates on the pair with largest $a+b$, which always includes the component containing all previously merged taxa.

For $\gamma = 1$, at step m, the forest T_m has $k = m+1$ components with leaf counts $n_1, \ldots, n_k$. The normalizing constant is

$$\sum_{i<j}(n_i + n_j - 1) = (k-1)\sum_i n_i - \binom{k}{2} = \frac{1}{2}(k-1)(2N-k),$$

since each n_i appears in $k-1$ pairs. Taking the product from $k = N$ to $k = 2$, we get

$$\prod_{k=2}^{N} \frac{1}{2}(k-1)(2N-k) = 2^{-(N-1)}(N-1)!\frac{(2N-2)!}{(N-1)!} = (N-1)!(2N-3)!!,$$

with the last equality coming by counting even and odd terms of $(2N-2)!$. Finally,

$$\begin{aligned} p(T_1, \ldots, T_{N-1} \mid T_0) &= \prod_{m=0}^{N-2} \frac{n_{\texttt{left}(v_m)} + n_{\texttt{right}(v_m)} - 1}{\frac{1}{2}(N-m)(2N-m-1)} \\ &= \frac{\prod_{v \in \texttt{int}(T_0)}(|\texttt{descendants}(v)| - 1)}{(N-1)!(2N-3)!!} \end{aligned}$$

which does not depend on the order of the merges. One can verify this equals $1/(|\mathrm{LE}(\texttt{int}(T_0))|(2N-3)!!)$ by the hook length formula $|\mathrm{LE}(\texttt{int}(T_0))| = (N-1)!/\prod_v |\texttt{descendants}(v)|$, noting each subtree at node v has $|\texttt{descendants}(v)| - 1$ internal nodes. Summing over all $|\mathrm{LE}(\texttt{int}(T_0))|$ linear extensions gives $p(T_0) = 1/(2N-3)!!$. □

C.2. Middle-out

Consider a process that generates a rooted tree by starting with a tree on two taxa and grafting one taxon at a time. At each step, the current tree has k leaves and $2k-1$ edges. The new taxon subdivides an edge e chosen with probability proportional to $w_\alpha(e)$, where *pendant* edges (those incident to a leaf) receive weight $1-\alpha$ and all other edges receive weight α, for a parameter $\alpha \in [0, 1)$.

Proposition C.2. *When $\alpha = 0$, $p(T_0)$ is a Yule distribution; when $\alpha = \frac{1}{2}$, $p(T_0)$ is PDA; and when $\alpha \to 1$, $p(T_0)$ concentrates on combs. Moreover, the distribution does not depend on the order in which taxa are inserted.*

Proof. This is Ford's alpha model (Ford, 2005). When $\alpha = 0$ only pendant edges carry weight, so each of the k lineages is equally likely to be split, recovering the Yule process. When $\alpha = \frac{1}{2}$ every edge carries equal weight, so the insertion

is uniform over all $2k-1$ edges; Ford (2005) shows this gives PDA. When $\alpha \to 1$ only non-pendant edges carry weight, producing combs. Order-independence is the exchangeability property of the alpha model (Ford, 2005). □

C.3. Top-down

The Aldous beta-splitting model (Aldous, 1996) generates a labeled rooted tree by recursively splitting sets of taxa. At a node containing $n \geq 2$ taxa, the set is partitioned into two non-empty groups; the probability of each labeled partition $\{A, B\}$ is proportional to $B(|A|+\beta+1,\ |B|+\beta+1)$ for $\beta > -2$. Each group is then split recursively until every group is a singleton. When $\beta = 0$ this gives the Yule distribution; when $\beta = -\frac{3}{2}$ it gives PDA; and as $\beta \to \infty$ it gives the "almost completely balanced" distribution (Aldous, 2001).

In our top-down paradigm, we do not split all taxa at a node simultaneously. Instead, taxa percolate one at a time: all taxa at a given depth complete their left/right assignments before any taxon proceeds to the next depth. We derive the per-step probabilities $f(a, b)$ from the Aldous model and show that every percolation order is equally likely conditioned on T_0, paralleling the analogous results for the **bottom-up** and **middle-out** paradigms.

For a node with n taxa where a have been assigned left and b right, with $M = n - a - b$ remaining, define

$$S(a,b,M) = \sum_{\substack{\{A,B\}:\ |A|\geq 1,\ |B|\geq 1 \\ \text{specified } a \text{ taxa}\in A,\ \text{specified } b \text{ taxa}\in B}} B(|A|+\beta+1,\ |B|+\beta+1), \tag{3}$$

the sum of the Aldous weights over all binary partitions consistent with the observed assignments. Since $|A| = a + j$ and $|B| = b + M - j$ for j of the remaining taxa joining A, we can write

$$S(a,b,M) = \sum_j \binom{M}{j} B(a{+}j{+}\beta{+}1,\ b{+}M{-}j{+}\beta{+}1), \tag{4}$$

where the sum runs over all j for which $a + j \geq 1$ and $b + M - j \geq 1$. Each remaining taxon joins exactly one side, so

$$S(a{+}1,\ b,\ M{-}1) + S(a,\ b{+}1,\ M{-}1) = S(a,\ b,\ M). \tag{5}$$

The conditional probability that the next taxon goes left is therefore

$$f(a,b) = \frac{S(a{+}1,\ b,\ M{-}1)}{S(a,\ b,\ M)}. \tag{6}$$

Proposition C.3. *The per-step probabilities (6) equal*

$$f(a,b) = \begin{cases} \dfrac{a+\beta+1}{a+b+2\beta+2}, & a, b > 0, \\[2ex] \dfrac{B(a{+}\beta{+}2,\ \beta{+}1) - B(n{+}\beta{+}1,\ \beta{+}1)}{B(a{+}\beta{+}1,\ \beta{+}1) - B(n{+}\beta{+}1,\ \beta{+}1)}, & b = 0,\ M > 1, \\[2ex] 0, & b = 0,\ M = 1, \end{cases} \tag{7}$$

with the $a = 0$ cases given by symmetry ($f(0,b) = 1 - f(b,0)$). The resulting marginal split distribution at each node is the Aldous beta-splitting model, every percolation order of the taxa is equally likely conditioned on T_0, and in particular: $\beta = 0$ gives Yule, $\beta = -\frac{3}{2}$ gives PDA, and $\beta \to \infty$ gives the almost completely balanced distribution.

Proof. We evaluate $S(a,b,M)$ in closed form using the integral representation $B(s,t) = \int_0^1 x^{s-1}(1{-}x)^{t-1}dx$ and the binomial theorem.

Interior case ($a, b > 0$). When $a \geq 1$ and $b \geq 1$ the constraints $a+j \geq 1$ and $b+M-j \geq 1$ hold for all $j \in \{0, \ldots, M\}$, so from (4)

$$S(a,b,M) = \int_0^1 x^{a+\beta}(1{-}x)^{b+\beta} \underbrace{\sum_{j=0}^{M} \binom{M}{j} x^j (1{-}x)^{M-j}}_{=1}\ dx = B(a{+}\beta{+}1,\ b{+}\beta{+}1).$$

Therefore $S(a{+}1, b, M{-}1) = B(a{+}\beta{+}2,\ b{+}\beta{+}1)$ and

$$f(a,b) = \frac{B(a{+}\beta{+}2,\ b{+}\beta{+}1)}{B(a{+}\beta{+}1,\ b{+}\beta{+}1)} = \frac{a+\beta+1}{a+b+2\beta+2},$$

using the identity $B(s{+}1,t)/B(s,t) = s/(s{+}t)$. Note this is a valid probability for all $\beta > -2$ when $a, b \geq 1$.

Boundary case ($b = 0$, $M > 1$). The constraint $b + M - j \geq 1$ forces $j \leq M - 1$, excluding the degenerate all-left split. Writing $n = a + M$,

$$\begin{aligned} S(a,0,M) &= \int_0^1 x^{a+\beta}(1{-}x)^{\beta}\Big[\underbrace{\sum_{j=0}^{M}\binom{M}{j}x^j(1{-}x)^{M-j}}_{=1} - x^M\Big]\,dx \\ &= B(a{+}\beta{+}1,\ \beta{+}1) - B(n{+}\beta{+}1,\ \beta{+}1). \end{aligned}$$

Similarly $S(a{+}1, 0, M{-}1) = B(a{+}\beta{+}2,\ \beta{+}1) - B(n{+}\beta{+}1,\ \beta{+}1)$. Dividing gives the $b = 0$ case of (7).

This expression can be simplified to a running product. Using $1 - x^k = (1{-}x)(1{+}x{+}\cdots{+}x^{k-1})$ in numerator and denominator,

$$f(a,0) = \frac{\sum_{j=0}^{M-2} B(a{+}\beta{+}2{+}j,\ \beta{+}2)}{\sum_{j=0}^{M-1} B(a{+}\beta{+}1{+}j,\ \beta{+}2)}.$$

Writing the denominator as $B(a{+}\beta{+}1,\ \beta{+}2) + \sum_{j=0}^{M-2} B(a{+}\beta{+}2{+}j,\ \beta{+}2)$ and applying $B(s{+}1,t)/B(s,t) = s/(s{+}t)$ to express each summand as a telescoping product gives

$$f(a,0) = 1 - \frac{1}{1 + \sum_{j=0}^{M-2}\prod_{\ell=0}^{j}\frac{a+\beta+1+\ell}{a+2\beta+3+\ell}}.$$

Terminal case ($b = 0$, $M = 1$). The last remaining taxon must go right to produce a binary split, giving $f(a, 0) = 0$. This is consistent with the formula above: $S(a{+}1, 0, 0) = B(n{+}\beta{+}1,\ \beta{+}1) - B(n{+}\beta{+}1,\ \beta{+}1) = 0$.

Order independence. Since the Aldous beta-splitting model assigns each labeled partition $\{A, B\}$ a weight depending only on $|A|$ and $|B|$, the model is exchangeable in the taxon labels. Therefore the conditional probability (6) at each step depends only on the counts (a, b) and not on which taxa were assigned, so every permutation of the n taxa yields the same path probability. Since all taxa at each depth complete before the next depth begins, the splits at distinct nodes are independent, and order independence extends to the full tree. □

D. Discrete diffusion models

D.1. Background

Discrete diffusion models have been recently highlighted for flexibly fitting complex distributions of discrete objects. However previous diffusion models have largely been built to model data of the form $\text{tokens}^{\text{length}}$ like language (Austin et al., 2021; Campbell et al., 2022; Shi et al., 2024). To apply these models to graphs therefore, one simply models their adjacency matrices $A \in \{0,1\}^{N\times N}$ (Vignac et al., 2022; Bohde et al., 2025). modeling phylogenetic trees as abstract graphs is problematic however as (1) internal nodes of the tree are latent, so there is no ground truth adjacency matrix A, and (2) these trees are highly constrained structures, and there is no clear way to incorporate these constraints during generation. Our models avoid these issues by building a diffusion model directly on the space of trees.

Discrete diffusion models have also succeeded at modeling other sorts of discrete objects, like permutation processes (Zhang et al., 2024), and sequences of different lengths (Havasi et al., 2025; Baron et al., 2026). However, these methods do not map to modeling trees.

D.2. Derivation of discrete diffusion ELBO

$$q_\theta(T_0 \mid X_{1:N}) = \sum_{T_1, T_2, \ldots, T_M} \prod_{m=1}^{M} q_\theta(T_{m-1} \mid T_m, X_{1:N})$$

Here, we provide the derivation of the discrete diffusion ELBO in Equation (2). We start by taking the logarithm of the expression for the likelihood in Equation (1). Then, we multiply and divide each term by $p\left(T_{1:M} \mid T_0\right)$, yielding

$$\begin{aligned}
\log q_\theta(T_0 \mid X_{1:N}) &= \log \sum_{T_{1:M}} p\left(T_{1:M} \mid T_0\right) \frac{\prod_{m=1}^{M} q_\theta(T_{m-1} \mid T_m, X_{1:N})}{p\left(T_{1:M} \mid T_0\right)} \\
&= \log \mathbb{E}_{T_{1:M} \sim p(\cdot \mid T_0)} \left[\prod_{m=1}^{M} \frac{q_\theta(T_{m-1} \mid T_m, X_{1:N})}{p(T_{m-1} \mid T_m, T_0)} \right] \\
&\geq \mathbb{E}_{T_{1:M} \sim p(\cdot \mid T_0)} \left[\sum_{m=1}^{M} \log \frac{q_\theta(T_{m-1} \mid T_m, X_{1:N})}{p(T_{m-1} \mid T_m, T_0)} \right] \\
&\geq \sum_{m=1}^{M} \mathbb{E}_{(T_{m-1}, T_m) \sim p(\cdot \mid T_0, m)} \left[\log \frac{q_\theta(T_{m-1} \mid T_m, X_{1:N})}{p(T_{m-1} \mid T_m, T_0)} \right] \\
&\geq - \sum_{m=1}^{M} \mathbb{E}_{T_m \sim p(\cdot \mid T_0, m)} \left[D_{\mathrm{KL}}(p(T_{m-1} \mid T_m, T_0) \,||\, q_\theta(T_{m-1} \mid T_m, X_{1:N})) \right]
\end{aligned}$$

E. Additional information on methods

For each paradigm we describe how to calculate the target $p(T_{m-1} \mid T_m, T_0)$ and sample from our mutual-information informed noising schedule.

E.1. Noise schedule details

The ELBO in Equation (2) requires computing $p(T_{m-1} \mid T_m, T_0)$, the distribution over the previous state given the current noised state and the ground truth. We achieve this by introducing a noise schedule $\pi(m, T_m \mid T_0)$ and rewriting the ELBO as $\mathbb{E}_{p(T_0, X_{1:N})} \log q_\theta(T_0 \mid X_{1:N}) \geq$

$$-\mathbb{E}_{m, T_m \sim \pi(\cdot \mid T_0)} \left[w(m, T_m) \cdot D_{\mathrm{KL}}(p(T_{m-1} \mid T_m, T_0) \,\|\, q_\theta(T_{m-1} \mid T_m, X_{1:N})) \right] \tag{8}$$

where

$$w(m, T_m) = \frac{M \cdot p(T_m \mid T_0, m)}{\pi(m, T_m \mid T_0)}. \tag{9}$$

A classic heuristic for designing discrete diffusion noise schedules is to choose π to minimize the variance of this Monte Carlo estimator, assuming q_θ has converged to the true reverse process (Kingma et al., 2021; Austin et al., 2021). To make this tractable, we further assume a simple reference posterior $p(T_0 \mid X_{1:N})$. A natural choice is any of the prior distributions described in Section 3.3 whose reverse process we can compute in closed form; we use $p_{\mathrm{PDA}}(T_0)$ as a concrete example throughout, though any such prior works equally well.

Under this assumption, the variance-minimizing π sets $w(m, T_m) \cdot D_{\mathrm{KL}}(\cdots)$ to a constant across (m, T_m), giving

$$\pi(m, T_m \mid T_0) \propto M \cdot p(T_m \mid T_0, m) \cdot D_{\mathrm{KL}}(p(T_{m-1} \mid T_m, T_0) \,\|\, p_{\mathrm{PDA}}(T_{m-1} \mid T_m)). \tag{10}$$

The normalizing constant of π is

$$\sum_{m=1}^{M} \sum_{T_m} M \cdot p(T_m \mid T_0, m) \cdot D_{\mathrm{KL}}(p(T_{m-1} \mid T_m, T_0) \,\|\, p_{\mathrm{PDA}}(T_{m-1} \mid T_m)) = M \cdot \mathrm{NELBO}_{\mathrm{PDA}}(T_0). \tag{11}$$

For all three paradigms, our noising process is chosen so that pairing it with the p_{PDA} reverse model described above yields a consistent generative model, meaning the ELBO is tight and $\text{NELBO}_{\text{PDA}}(T_0) = -\log p_{\text{PDA}}(T_0)$. We verify this and show how to sample from π exactly below.

Marginalizing over $T_0 \sim p_{\text{PDA}}$, the quantity $p(T_m \mid T_0, m) \cdot D_{\text{KL}}(p(T_{m-1} \mid T_m, T_0) \,\|\, p_{\text{PDA}}(T_{m-1} \mid T_m))$ becomes the mutual information $\text{MI}(T_{m-1}; T_0 \mid T_m)$: the amount of information about T_0 that is lost at step m. This mirrors the mutual information schedules of Austin et al. (2021) and concentrates training on steps where the model must add the most information about T_0 in a single step.

E.2. Bottom-up

E.2.1. Distribution over paths

It will be convenient to derive the distribution over paths $p(T_1, \ldots, T_{N-1} \mid T_0)$. The path $(T_m)_m$ is characterized by an order of internal vertices which are removed from T_0, $v_1, \ldots, v_{N-1}$; on the other hand every ordering of the internal vertices of T_0 $v_1, \ldots, v_{N-1}$ corresponds to a unique path as long as the ancestors of every node appear before it in the ordering. An ordering of the internal vertices that satisfies this rule is known in the literature as a *linear extension* of the subtree of T_0 made of its internal nodes, $\texttt{int}(T_0)$. We call $\text{LE}(T)$ the set of linear extensions of a tree T. We now show our choice of noising process makes *every linear extension equally likely*; this is the property that will allow us to derive the quantities we need below.

Proposition E.1. *The probability of a path $(T_m)_m$ that removes vertices in order $v_1, \ldots, v_{N-1}$ is*

$$p(T_1, \ldots, T_{N-1} \mid T_0) = 1/|\text{LE}(\texttt{int}(T_0))|.$$

Proof. Recall for a root of a subtree v, $p(\texttt{remove } v \mid T_m) \propto |\text{descendants}(v)|$, excluding itself as a descendant. The denominator is the sum of the descendants of all roots, so it's simply the number of nodes in the tree, minus the roots: $2N - 2(m+1)$. Finally,

$$\begin{aligned} p(T_1, \ldots, T_{N-1} \mid T_0) &= \prod_{m=0}^{N-2} p(\texttt{remove } v \mid T_m) \\ &= \prod_{m=0}^{N-2} \frac{|\text{descendants}(v_m)|}{2N - 2(m+1)} \\ &= \frac{\prod_{v \in \texttt{int}(T_0)} |\text{descendants}(v)|}{2^{N-1}(N-1)!} \end{aligned}$$

which does not depend on the order of the vertices. One can verify this also matches the classical hook length formula by noting $|\text{descendants}(v)|$ is twice the hook length of v in $\texttt{int}(T_0)$. □

E.2.2. Calculating the target

We can now derive $p(T_{m-1} \mid T_m, T_0)$. The set of removed nodes $\{v_1, \ldots, v_m\}$ forms a rooted subtree of $\texttt{int}(T_0)$ containing the root; call it R_m. Its leaves are exactly the candidates for the last node removed. Since all linear extensions of $\texttt{int}(T_0)$ are equally likely, the probability that node v was removed last is simply the fraction of linear extensions of R_m that end in v: $p(T_{m-1} = \texttt{add}(T_m, v) \mid T_m, T_0) = \frac{|\text{LE}(\texttt{remove}(R_m, v)|}{|\text{LE}(R_m)|}$.

Proposition E.2. *For a leaf v of R_m,*

$$p(T_{m-1} = \texttt{add}(T_m, v) \mid T_m, T_0) = \frac{1}{m} \prod_{u \in \texttt{ancestors}(v, R_m)} \frac{|\texttt{descendants}(u, R_m)|}{|\texttt{descendants}(u, R_m)| - 1}, \tag{12}$$

where $\texttt{descendants}(u, R_m)$ counts the descendants of u in R_m, including u itself.

Proof. By the hook length formula for rooted trees, $|\text{LE}(R_m)| = m!/\prod_{u \in R_m} |\texttt{descendants}(u, R_m)|$. Removing a leaf v from R_m decreases the numerator from $m!$ to $(m-1)!$, removes the factor $|\texttt{descendants}(v, R_m)| = 1$ from the denominator, and decrements $|\texttt{descendants}(u, R_m)|$ by 1 for every strict ancestor u of v. Taking the ratio gives Equation (12). □

where $\texttt{descendants}(v, R_m)$ counts the descendants of v in R_m, including v itself. This can be calculated with a post-order traversal of R_m to compute $|\texttt{descendants}(v, R_m)|$ for all v, then a pre-order pass to accumulate the product over ancestors. The cost of computing $p(T_{m-1} \mid T_m, T_0)$ for every possible T_{m-1} is therefore $O(m)$.

E.2.3. Noise schedule

As described in Section E.1, we importance-sample the noise level m and noised state T_m in proportion to their contribution to the NELBO, concentrating training on the most informative steps.

Overview Our strategy proceeds in three parts. First, we show that the probability of reaching any particular noised state T_m can be written as a ratio of linear extension counts (Equation (13)). This means we can compute contributions to the NELBO by sampling a single linear extension of $\texttt{int}(T_0)$ uniformly at random. Second, we define a per-step contribution $c_m(\sigma)$ whose sum over m is the same for every linear extension σ (Theorem E.3), enabling exact categorical sampling of the noise level (Theorem E.4). Third, since individual $c_m(\sigma)$ values can be negative, we decompose them into non-negative parts and use rejection sampling (Algorithm 1).

State probabilities Recall from Section E.2 that each run of the noising process produces a linear extension $\sigma = (v_1, \ldots, v_{N-1})$ of $\texttt{int}(T_0)$, determining removed sets $R_0 = \emptyset \subset R_1 \subset \cdots \subset R_{N-1} = \texttt{int}(T_0)$ with $R_m = \{v_1, \ldots, v_m\}$. We showed above that every linear extension is equally likely under our noising process.

How many linear extensions pass through a given R_m? The orderings within R_m and within $\texttt{int}(T_0) \setminus R_m$ are independent: any linear extension of R_m can be concatenated with any linear extension of $\texttt{int}(T_0) \setminus R_m$ to form a valid linear extension of $\texttt{int}(T_0)$. Therefore the probability of reaching state T_m at step m is therefore

$$p(T_m \mid T_0, m) = \frac{|\mathrm{LE}(R_m)| \cdot |\mathrm{LE}(\texttt{int}(T_0) \setminus R_m)|}{|\mathrm{LE}(\texttt{int}(T_0))|}. \tag{13}$$

The PDA reverse model By Theorem C.1, the PDA coalescent ($\gamma = 1$) also produces uniform linear extensions conditioned on T_0. Its reverse model $q_{\mathrm{PDA}}(T_{m-1} \mid T_m)$ merges subtree roots u, w in the forest T_m with probability proportional to $n_u + n_w - 1$, where n_u, n_w are their respective leaf counts. We can therefore calculate $\mathbb{E}_{T_0|T_m} p(T_{m-1} \mid T_m, T_0) = q_{\mathrm{PDA}}(T_{m-1} \mid T_m)$.

Per-step contributions For a linear extension $\sigma = (v_1, \ldots, v_{N-1})$ and step $m \in \{1, \ldots, N-1\}$, define

$$c_m(\sigma) = \log \frac{|\mathrm{LE}(R_{m-1})|}{|\mathrm{LE}(R_m)|} - \log q_{\mathrm{PDA}}(v_m \mid T_m), \tag{14}$$

where the first term equals $\log p(T_{m-1} = \texttt{add}(T_m, v_m) \mid T_m, T_0)$ by Equation (12), and $q_{\mathrm{PDA}}(v_m \mid T_m)$ is the probability that q_{PDA} merges the two subtree roots that were children of v_m in T_0. Note that $c_m(\sigma)$ depends on σ only through the pair (R_m, v_m): the LE ratio depends only on which leaf of R_m was removed last, and q_{PDA} depends only on the forest T_m and which merge is performed.

Proposition E.3 (Constant sum)**.** *For every* $\sigma \in \mathrm{LE}(\texttt{int}(T_0))$,

$$\sum_{m=1}^{N-1} c_m(\sigma) = \log(2N-3)!!.$$

Proof. The first sum in Equation (14) telescopes:

$$\sum_{m=1}^{N-1} \log \frac{|\mathrm{LE}(R_{m-1})|}{|\mathrm{LE}(R_m)|} = \log |\mathrm{LE}(\emptyset)| - \log |\mathrm{LE}(\texttt{int}(T_0))| = -\log |\mathrm{LE}(\texttt{int}(T_0))|.$$

For the second sum, the product $\prod_{m=1}^{N-1} q_{\mathrm{PDA}}(v_m \mid T_m)$ is the probability of producing T_0 via the specific merge order given by the reversal of σ. By Theorem C.1, every linear extension is equally likely under the PDA coalescent conditioned on T_0, so this product equals $q_{\mathrm{PDA}}(T_0)/|\mathrm{LE}(\texttt{int}(T_0))|$ for every σ. Therefore

$$\sum_{m=1}^{N-1} \log q_{\mathrm{PDA}}(v_m \mid T_m) = \log q_{\mathrm{PDA}}(T_0) - \log |\mathrm{LE}(\texttt{int}(T_0))|.$$

Subtracting: $-\log|\mathrm{LE}(\mathtt{int}(T_0))| - \log q_{\mathrm{PDA}}(T_0) + \log|\mathrm{LE}(\mathtt{int}(T_0))| = -\log q_{\mathrm{PDA}}(T_0) = \log(2N-3)!!$. □

Proposition E.4 (Sampling scheme)**.** *The following procedure samples (m, T_m) from π:*

1. *Draw $\sigma = (v_1, \ldots, v_{N-1})$ uniformly from $\mathrm{LE}(\mathtt{int}(T_0))$.*
2. *Draw $m \in \{1, \ldots, N-1\}$ with probability proportional to $c_m(\sigma)$.*
3. *Output (m, T_m) determined by $R_m = \{v_1, \ldots, v_m\}$.*

Proof. Fix a step m and a valid removed set R_m. The probability of drawing a particular σ in step 1 and selecting step m in step 2 is

$$\frac{1}{|\mathrm{LE}(\mathtt{int}(T_0))|} \cdot \frac{c_m(\sigma)}{Z_\pi},$$

using Theorem E.3 for the normalizer.

We sum over all σ with $\{v_1, \ldots, v_m\} = R_m$, grouping by $v = v_m \in \mathrm{leaves}(R_m)$. Since v is a leaf of R_m (i.e. it has no descendants in R_m), it is maximal in the ancestor ordering restricted to R_m, and can therefore be placed last in any ordering of R_m. A linear extension of $\mathtt{int}(T_0)$ with $\{v_1, \ldots, v_m\} = R_m$ and $v_m = v$ is therefore determined by two independent choices: a linear extension of $R_m \setminus \{v\}$ for the first $m-1$ positions, and a linear extension of $\mathtt{int}(T_0) \setminus R_m$ for the last $N-1-m$ positions. The count of such linear extensions is $|\mathrm{LE}(R_m \setminus \{v\})| \cdot |\mathrm{LE}(\mathtt{int}(T_0) \setminus R_m)|$.

Writing $c(v, R_m) = c_m(\sigma)$ (which depends on σ only through (R_m, v_m)):

$$P(\text{output } R_m) = \frac{1}{|\mathrm{LE}(\mathtt{int}(T_0))| \cdot Z_\pi} \sum_{v \in \mathrm{leaves}(R_m)} |\mathrm{LE}(R_m \setminus \{v\})| \cdot |\mathrm{LE}(\mathtt{int}(T_0) \setminus R_m)| \cdot c(v, R_m).$$

We factor out $|\mathrm{LE}(R_m)| \cdot |\mathrm{LE}(\mathtt{int}(T_0) \setminus R_m)|$, which equals $p(T_m \mid T_0, m) \cdot |\mathrm{LE}(\mathtt{int}(T_0))|$ by Equation (13):

$$P(\text{output } R_m) = \frac{p(T_m \mid T_0, m)}{Z_\pi} \sum_{v \in \mathrm{leaves}(R_m)} \frac{|\mathrm{LE}(R_m \setminus \{v\})|}{|\mathrm{LE}(R_m)|} \cdot c(v, R_m).$$

By Equation (12), the ratio $|\mathrm{LE}(R_m \setminus \{v\})|/|\mathrm{LE}(R_m)| = p(T_{m-1} = \mathtt{add}(T_m, v) \mid T_m, T_0)$. The sum is therefore $\mathrm{KL}(p(T_{m-1} \mid T_m, T_0) \,\|\, q_{\mathrm{PDA}}(T_{m-1} \mid T_m))$, giving $P(\text{output } R_m) = \pi(m, T_m)$. □

The sampling scheme requires $c_m(\sigma) \geq 0$ for all σ, m, since we use it as a categorical weight in step 2. While $\sum_v p(v \mid R_m, T_0)\, c(v, R_m) = \mathrm{KL}(\cdots) \geq 0$, individual values $c(v, R_m)$ can be negative when $p(v \mid R_m, T_0) < q_{\mathrm{PDA}}(v \mid T_m)$, making the categorical sampling ill-defined. We resolve this below.

Rejection sampling from π We decompose $c_m(\sigma) = c_m^+(\sigma) - c_m^-(\sigma)$ where

$$c_m^+(\sigma) = -\log q_{\mathrm{PDA}}(v_m \mid T_m) \geq 0, \qquad c_m^-(\sigma) = -\log \frac{|\mathrm{LE}(R_{m-1})|}{|\mathrm{LE}(R_m)|} \geq 0,$$

corresponding to the decomposition $\mathrm{KL} = H_{\mathrm{cross}} - H$.

Proposition E.5. *For every $\sigma \in \mathrm{LE}(\mathtt{int}(T_0))$:*

$$\sum_{m=1}^{N-1} c_m^+(\sigma) = \log|\mathrm{LE}(\mathtt{int}(T_0))| + \log(2N-3)!! =: Z^+, \tag{15}$$

$$\sum_{m=1}^{N-1} c_m^-(\sigma) = \log|\mathrm{LE}(\mathtt{int}(T_0))| =: Z^-. \tag{16}$$

Proof. The sum $\sum_m c_m^+(\sigma) = -\sum_m \log q_{\mathrm{PDA}}(v_m \mid T_m) = \log|\mathrm{LE}(\mathtt{int}(T_0))| - \log q_{\mathrm{PDA}}(T_0)$ follows from Theorem C.1, as in the proof of Theorem E.3. The sum $\sum_m c_m^-(\sigma) = \sum_m \log \frac{|\mathrm{LE}(R_m)|}{|\mathrm{LE}(R_{m-1})|}$ telescopes to $\log|\mathrm{LE}(\mathtt{int}(T_0))|$. □

Since c_m^+ is non-negative with constant sum Z^+, we can define the *cross-entropy proposal*

$$\pi^+(m, T_m) \propto p(T_m \mid T_0, m) \cdot H_{\text{cross}}\big(p(T_{m-1} \mid T_m, T_0),\ q_{\text{PDA}}(T_{m-1} \mid T_m)\big),$$

and sample it exactly by replacing c_m with c_m^+ in Theorem E.4. Note $Z_\pi = Z^+ - Z^-$, consistent with $\text{KL} = H_{\text{cross}} - H$.

Since $\text{KL} \le H_{\text{cross}}$ pointwise, $\pi(m, T_m)/\pi^+(m, T_m) \le Z^+/Z_\pi$ for all (m, T_m), yielding the following exact rejection sampler.

Algorithm 1 Importance Sampling the Bottom-Up Noise Schedule

Require: Tree T_0 with N leaves
Ensure: Sample $(m, T_m) \sim \pi$
  repeat
    Draw $\sigma = (v_1, \ldots, v_{N-1})$ uniformly from $\text{LE}(\texttt{int}(T_0))$
    Draw $m \in \{1, \ldots, N-1\}$ with probability $\propto c_m^+(\sigma) = -\log q_{\text{PDA}}(v_m \mid T_m)$
    Set $R_m \leftarrow \{v_1, \ldots, v_m\}$
    Compute acceptance ratio $\rho \leftarrow \frac{\text{KL}(p(T_{m-1}|T_m,T_0) \,\|\, q_{\text{PDA}}(T_{m-1}|T_m))}{H_{\text{cross}}(p(T_{m-1}|T_m,T_0),\, q_{\text{PDA}}(T_{m-1}|T_m))}$
    Accept with probability ρ
  until accepted
  return (m, T_m)

Proposition E.6 (Acceptance rate)**.** *The acceptance rate of Algorithm 1 is*

$$\frac{Z_\pi}{Z^+} = \frac{\log(2N-3)!!}{\log|\text{LE}(\texttt{int}(T_0))| + \log(2N-3)!!}.$$

Proof. The acceptance rate of a rejection sampler with proposal π^+ and envelope constant Z^+/Z_π is Z_π/Z^+. Substituting $Z_\pi = \log(2N-3)!!$ and $Z^+ = \log|\text{LE}(\texttt{int}(T_0))| + \log(2N-3)!!$ from Equation (15) yields the result. □

Both $\log(2N-3)!!$ and $\log|\text{LE}(\texttt{int}(T_0))|$ grow as $O(N \log N)$, so the acceptance rate is bounded away from 0 for all tree shapes. It equals 1 for caterpillar trees (where $|\text{LE}(\texttt{int}(T_0))| = 1$) and is minimized for maximally balanced trees (where $|\text{LE}(\texttt{int}(T_0))|$ is largest by the hook length formula).

Importance weight for training When training a model q_θ rather than q_{PDA}, we sample $(m, T_m) \sim \pi$ and reweight. Recall from Equation (10) that $\pi(m, T_m \mid T_0) \propto p(T_m \mid T_0, m) \cdot D_{\text{KL}}(p(T_{m-1} \mid T_m, T_0) \,\|\, q_{\text{PDA}}(T_{m-1} \mid T_m))$, with normalizing constant Z_π from Equation (11). The unbiased single-sample estimator of the NELBO is

$$\widehat{\text{NELBO}}_\theta = \frac{p(T_m \mid T_0, m) \cdot D_{\text{KL}}\big(p(T_{m-1} \mid T_m, T_0) \,\big\|\, q_\theta(T_{m-1} \mid T_m, X_{1:N})\big)}{\pi(m, T_m \mid T_0)}. \tag{17}$$

Since $p(T_m \mid T_0, m)$ appears in both the numerator and the definition of π, it cancels, giving

$$\widehat{\text{NELBO}}_\theta = \text{NELBO}_{\text{PDA}}(T_0) \cdot \frac{D_{\text{KL}}\big(p(T_{m-1} \mid T_m, T_0) \,\big\|\, q_\theta(T_{m-1} \mid T_m, X_{1:N})\big)}{D_{\text{KL}}\big(p(T_{m-1} \mid T_m, T_0) \,\big\|\, q_{\text{PDA}}(T_{m-1} \mid T_m)\big)}, \tag{18}$$

where $\text{NELBO}_{\text{PDA}}(T_0) = -\log p_{\text{PDA}}(T_0) = \log(2N-3)!!$ for the bottom-up paradigm.

E.3. Middle-out

E.3.1. Distribution over paths

The noising process removes one taxon at each step, selecting uniformly among the $N - m + 1$ taxa present in T_{m-1}. A path from T_0 to T_{N-2} is a sequence $(v_1, \ldots, v_{N-2})$ of distinct taxa, where v_m is the taxon removed at step m. The fully noised state T_{N-2} is an unrooted tree containing two taxa and the pseudo-taxon. Clearly we have

Proposition E.7. *Every removal sequence has the same probability:* $p(v_1, \ldots, v_{N-2} \mid T_0) = 2/N!$.

E.3.2. CALCULATING THE TARGET

At state T_m, the tree contains $N - m$ taxa and the pseudo-taxon. To reverse one noising step, we reattach one of the m pruned taxa to an edge of T_m. For each pruned taxon there is exactly one edge in T_m such that grafting produces a tree consistent with T_0.

Proposition E.8. *The reverse distribution is uniform over the m valid re-attachments:*

$$p(T_{m-1} \mid T_m, T_0) = \frac{1}{m}.$$

Proof. Since all removal sequences are equally likely (Theorem E.7), the last-removed taxon is uniform over the m taxa absent from T_m. Each such taxon determines a unique reattachment consistent with T_0. □

E.3.3. NOISE SCHEDULE

We importance-sample the noise level m in proportion to its contribution to the NELBO under a PDA reference model.

The PDA reverse model. Under the PDA distribution, all $(2N-3)!!$ unrooted binary topologies on N taxa plus the pseudo-taxon are equally likely. The corresponding reverse model $q_{\mathrm{PDA}}(T_{m-1} \mid T_m)$ attaches a uniformly random missing taxon to a uniformly random edge of T_m. Since T_m has $N - m$ taxa and the pseudo-taxon, it is an unrooted binary tree with $N - m + 1$ leaves and therefore $2(N-m) - 1$ edges. Each of the $m\big(2(N-m)-1\big)$ possible re-attachments receives probability

$$q_{\mathrm{PDA}}(T_{m-1} \mid T_m) = \frac{1}{m\big(2(N-m)-1\big)}.$$

Per-step contributions. For a removal sequence $\sigma = (v_1, \ldots, v_{N-2})$ and step $m \in \{1, \ldots, N-2\}$, define

$$\begin{aligned} c_m(\sigma) &= \log p(T_{m-1} \mid T_m, T_0) - \log q_{\mathrm{PDA}}(T_{m-1} \mid T_m) \\ &= \log \frac{1}{m} - \log \frac{1}{m\big(2(N-m)-1\big)} \\ &= \log\big(2(N-m)-1\big). \end{aligned}$$

Proposition E.9 (Constant sum)**.** *For every removal sequence σ,*

$$\sum_{m=1}^{N-2} c_m(\sigma) = \log(2N-3)!!.$$

Proof. Since $c_m(\sigma) = \log(2(N-m)-1)$ does not depend on σ, the sum is

$$\sum_{m=1}^{N-2} \log\big(2(N-m)-1\big) = \log\big(3 \cdot 5 \cdots (2N-3)\big) = \log(2N-3)!!.$$ □

As a consistency check, $\mathrm{NELBO}_{\mathrm{PDA}}(T_0) = \log(2N-3)!! = -\log q_{\mathrm{PDA}}(T_0)$, confirming that the PDA ELBO is tight.

Sampling scheme. Because $c_m(\sigma) = \log(2(N-m)-1) > 0$ for all m and does not depend on σ or on the particular noised state T_m, the importance distribution takes a simple closed form.

Proposition E.10. *The following procedure samples (m, T_m) from π:*

1. *Draw $m \in \{1, \ldots, N-2\}$ with probability $\pi(m) = \dfrac{\log\big(2(N-m)-1\big)}{\log(2N-3)!!}$.*
2. *Draw a uniformly random subset $S \subset \{1, \ldots, N\}$ of size m and set $T_m = \texttt{prune}(T_0, S)$.*

No rejection sampling is needed since $c_m > 0$ for all valid m.

Proof. By Theorem E.7, conditioning on any removal count m, the set of removed taxa is a uniformly random m-element subset. Since the per-step KL c_m depends only on m, the target importance distribution $\pi(m, T_m \mid T_0) \propto p(T_m \mid T_0, m) \cdot c_m$ factors as $\pi(m) \cdot p(T_m \mid T_0, m)$, which is exactly the distribution sampled by the two-step procedure. □

Importance weight for training. When training a model q_θ rather than q_{PDA}, we sample $(m, T_m) \sim \pi$ and reweight. The unbiased single-sample estimator of the NELBO is

$$\widehat{\text{NELBO}}_\theta = \log(2N-3)!! \cdot \frac{D_{\text{KL}}\big(p(T_{m-1} \mid T_m, T_0) \,\big\|\, q_\theta(T_{m-1} \mid T_m, X_{1:N})\big)}{\log\big(2(N-m)-1\big)}. \tag{19}$$

E.4. Top-down

E.4.1. Distribution over paths

The noising process percolates taxa upward from depth $d_{T_0}(v)$ to the root, one level at a time. Since only taxa at maximal depth are eligible, the process decomposes into *rounds*: round d (for $d = d_{\max}, d_{\max}-1, \ldots, 1$, where $d_{\max}$ is the maximum leaf depth in T_0) moves every taxon currently at depth d up to depth $d-1$.

Definition E.11 (Round structure)**.** Let $N_d = |\{v : d_{T_0}(v) \geq d\}|$ be the number of taxa that participate in round d. At the start of round d, all N_d of these taxa are at depth d: those originally at depth exactly d have remained there, while those originally deeper have been brought to depth d by earlier rounds. Within round d, the N_d taxa are moved from depth d to $d-1$ one at a time, each selected uniformly from the taxa still at depth d.

A *percolation sequence* is the full list $(v_1, \ldots, v_M)$ of taxa moved across all rounds, where $M = \sum_{v=1}^{N} d_{T_0}(v)$.

Proposition E.12. *Every percolation sequence has the same probability:* $p(v_1, \ldots, v_M \mid T_0) = 1/\prod_{d=1}^{d_{\max}} N_d!$.

Proof. Within round d, each of the $N_d!$ orderings of the N_d taxa is equally likely (uniform selection without replacement). Since the rounds are processed sequentially and each round's set of taxa is determined by T_0, the orderings across rounds are independent. □

E.4.2. Calculating the target

At state T_m, some rounds have been completed and one is partially processed. Let d^* be the depth of the current round (the maximal depth of any taxon in T_m). Within round d^*, let k be the number of taxa already moved from depth d^* to d^*-1.

Let $\mathcal{V}_m = \{v : d_{T_m}(v) = d^*-1 \text{ and } d_{T_0}(v) \geq d^*\}$ be the set of taxa moved so far in round d^*. Equivalently, $\mathcal{V}_m$ consists of those taxa at minimal depth among all taxa satisfying $d_{T_m}(v) < d_{T_0}(v)$.

Proposition E.13. *The reverse distribution is uniform over* $|\mathcal{V}_m|$ *valid downward moves:*

$$p(T_{m-1} \mid T_m, T_0) = \frac{1}{|\mathcal{V}_m|}.$$

Proof. By Theorem E.12, the ordering within round d^* is a uniform random permutation. Conditioned on the set $\mathcal{V}_m$ of taxa already moved, the last one moved is uniform over $\mathcal{V}_m$. Each such taxon determines a unique downward move consistent with T_0 (it returns to the child of its current node that is an ancestor of its ground-truth position). □

E.4.3. Noise schedule

Unlike the middle-out case where the per-step KL depends only on m, the top-down per-step KL depends on the state at each internal node. However, the *total* NELBO still admits a clean factorization over the internal nodes of T_0.

The PDA reverse model. At state T_m within round d^*, each internal node u at depth d^*-1 has some taxa that have been percolated up to u (with their child assignments unknown) and some taxa still observed at their children. Write a_u and b_u for the number of taxa still observed at u's left and right children respectively. The PDA reverse model predicts the direction of the next taxon at node u using the Aldous beta-splitting probabilities $f(a_u, b_u)$ from Theorem C.3 with $\beta = -\frac{3}{2}$.

Per-step contributions. For a percolation sequence σ and step j, let u_j be the parent node of the taxon v_j in T_0. Define

$$c_j(\sigma) = -\log f_{u_j}(\text{correct direction for } v_j),$$

where f_{u_j} denotes $f(a_{u_j}, b_{u_j})$ or $1 - f(a_{u_j}, b_{u_j})$ according to whether v_j belongs to the left or right subtree of u_j, evaluated at the observed counts in state T_j. Since $f \leq 1$, we have $c_j(\sigma) \geq 0$ for all j and σ.

Proposition E.14 (Constant sum)**.** *For every percolation sequence σ,*

$$\sum_{j=1}^{M} c_j(\sigma) = (N-1)\log 2 + \log(2N-3)!!.$$

Proof. The sum decomposes over internal nodes: each node v at depth $d-1$ contributes those steps j where v_j is percolated through v during round d. By the order independence established in Theorem C.3, the product of per-step probabilities at node v is the same for every ordering of v's taxa, and equals the Aldous beta-splitting probability of the specific labeled partition $\{n_L(v), n_R(v)\}$:

$$\prod_{j \text{ at } v} f_v(\text{correct direction}) = \frac{S(n_L(v),\, n_R(v),\, 0)}{S(0,\, 0,\, n(v))} = \frac{B(n_L(v)+\beta+1,\; n_R(v)+\beta+1)}{S(0,\, 0,\, n(v))}. \tag{20}$$

Setting $\beta = -\frac{3}{2}$ and using the identity $B(L-\frac{1}{2},\, R-\frac{1}{2}) = \pi\,(2L-2)!\,(2R-2)!\,/\,(4^{L+R-1}\,(L-1)!\,(R-1)!)$ together with the evaluation $S(0,0,n) = 2B(\beta+1,\, \beta+1) - 2B(n+\beta+1,\, \beta+1)$ (the total Aldous weight at a node with n taxa), this product simplifies to

$$\prod_{j \text{ at } v} f_v(\text{correct direction}) = \frac{(2n_L(v)-3)!!\,(2n_R(v)-3)!!}{2\,(2n(v)-3)!!}. \tag{21}$$

Taking negative logarithms and summing over all $N-1$ internal nodes:

$$\sum_j c_j = \sum_{v \,\in\, \texttt{int}(T_0)} \log \frac{2\,(2n(v)-3)!!}{(2n_L(v)-3)!!\,(2n_R(v)-3)!!}.$$

The double-factorial ratio telescopes across the tree: the root contributes $(2N-3)!!$ in the numerator, and at every other internal node the $(2n(v)-3)!!$ in the numerator cancels against the matching factor in its parent's denominator; the leaves contribute $(-1)!! = 1$. Therefore

$$\prod_{v \,\in\, \texttt{int}(T_0)} \frac{(2n(v)-3)!!}{(2n_L(v)-3)!!\,(2n_R(v)-3)!!} = (2N-3)!!,$$

and together with the factor of 2 per internal node we obtain $(N-1)\log 2 + \log(2N-3)!!$. □

The constant $(N-1)\log 2 + \log(2N-3)!! = \log(2^{N-1}(2N-3)!!)$ is the negative log-probability of a specific *oriented* labeled tree under PDA, where the $\log 2$ per node accounts for distinguishing left from right children. Since our model is symmetric in its left/right predictions (Section 4.3), the orientation cost of $(N-1)\log 2$ is achieved automatically and does not provide training signal. The topology-relevant part of the NELBO is $\log(2N-3)!! = -\log q_{\text{PDA}}(T_0)$, as in the bottom-up and middle-out cases.

Sampling scheme. Since $c_j \geq 0$ for all j and the sum is constant across percolation sequences, we can sample from the importance distribution π without rejection. Define

$$p_{\text{PDA}}(L, R) = \frac{(2L-3)!!\,(2R-3)!!}{(2(L+R)-3)!!}$$

to be the PDA probability that a specific set of L taxa forms the left subtree at a node with $L + R$ taxa; the product $\prod_v p_{\text{PDA}}(n_L(v), n_R(v)) = 1/(2N-3)!!$ recovers $q_{\text{PDA}}(T_0)$.

Proposition E.15. *The following procedure samples (m, T_m) from π:*

1. ***Sample a focus node.*** *Draw $v^* \in \texttt{int}(T_0)$ with probability*

$$P(v^*) = \frac{\log 2 - \log p_{\mathrm{PDA}}(n_L(v^*),\, n_R(v^*))}{(N-1)\log 2 + \log(2N-3)!!}.$$

2. ***Sample the noise state at v^*.*** *Build the $O(n(v^*)^2)$ table of importance weights*

$$\pi_{v^*}(a,b) \propto h(a,b \mid n_L(v^*), n_R(v^*)) \cdot I(a,b),$$

where $h(a,b \mid L,R) = \binom{L}{a}\binom{R}{b}/\binom{L+R}{a+b}$ is the hypergeometric probability that a of the observed taxa are on the left and b on the right given $a+b$ total observed, and $I(a,b) = \frac{a}{a+b}\big(-\log f(a,b)\big) + \frac{b}{a+b}\big(-\log(1-f(a,b))\big)$ is the per-element KL contribution. Draw (a_{v^*}, b_{v^*}) from this table.

3. ***Sample the global noise level.*** *Given $k_{v^*} = n(v^*) - a_{v^*} - b_{v^*}$ taxa unobserved at v^*, the round is $d^* = \texttt{depth}(v^*) + 1$. The total number of taxa moved in round d^* is $k = k_{v^*} + k_{rest}$, where k_{rest} is drawn from* $\mathrm{BetaBin}(N_{d^*} - n(v^*),\ k_{v^*},\ n(v^*) - k_{v^*} + 1)$.

4. ***Fill in remaining nodes.*** *Distribute k_{rest} among the other nodes at depth d^*-1 via the multivariate hypergeometric, and draw per-node observed counts (a_u, b_u) from the corresponding hypergeometrics.*

Proof. The factorization of π into a mixture over focus nodes follows from Theorem E.14: each node v contributes $\log 2 - \log p_{\mathrm{PDA}}(n_L(v), n_R(v))$ to the total. Conditioned on the focus node v^*, the noise state at v^* is sampled proportional to its KL contribution (step 2). The global noise level k and the states at other nodes are sampled from their conditional distributions under the uniform noising process (steps 3–4), which factor by the same Vandermonde identity used in the middle-out case. □

Importance weight for training. When training a model q_θ in place of q_{PDA}, the unbiased single-sample estimator of the NELBO is

$$\widehat{\mathrm{NELBO}}_\theta = \big((N-1)\log 2 + \log(2N-3)!!\big) \cdot \frac{D_{\mathrm{KL}}\big(p(T_{m-1} \mid T_m, T_0) \,\big\|\, q_\theta(T_{m-1} \mid T_m, X_{1:N})\big)}{\sum_{u \text{ at depth } d^*-1} (k_u/k)\, I(a_u, b_u)}, \tag{22}$$

where $k_u = n(u) - a_u - b_u$ is the number of unobserved taxa at node u, $k = \sum_u k_u$, and the denominator is the KL contribution of the sampled state under the PDA reference model.

F. Architecture details

Our model q_θ must condition on the sequences $X_{1:N}$ and current state of the tree T_m. To condition on the sequences, we adapt the EvoPF architecture from Phyloformer 2 (Blassel et al., 2025). EvoPF embeds N sequences of length L, $X_{1:N}$, into a tensor of size $[N, L, 256]$ and performs axial attention over the dimensions N and L separately. For scalability, we drop the quartic pairwise attention from EvoPF so our architecture has complexity $\mathcal{O}(NL^2)$. After a trunk made of 12 layers, we turn embeddings from each node into a set of logits.

Embedding trunk Our architecture starts with four EvoPF layers. We then pass this tensor through eight EvoPF layers modified to condition on the tree structure T_m. We make three modifications for these last 8 layers that allow us to condition on T_m:

- We include internal nodes of T_m in our calculation, so our tensor is of size $[N + \#\texttt{internal_nodes}(T_m), L, 256]$. Say $h_v^{(k)} \in \mathbb{R}^{(L+1)\times 256}$ is the embedding of node or leaf v at layer k. The $+1$ is from an extra position we also add to the beginning of each sequence which will allow us to read out a single vector.
- After each layer, each internal node v is refreshed as

$$h_v^{(k)} \leftarrow h_v^{(k)} + \mathrm{MLP}_{k,\theta}\left(\texttt{sinusoidal}(\texttt{depth}(v),\, |\mathrm{leaves}(v)|),\, \frac{1}{|\mathrm{leaves}(v)|}\sum_{\ell \in \mathrm{leaves}(v)} h_\ell^{(k)}\right)$$

  where $\mathrm{leaves}(v)$ denotes the leaf descendants of v in T_m.

- During attention over N we bias the attention using the graph distance in T_m (which can be ∞): for nodes v, u, position ℓ, and head h,
$$\text{Attention}_{v,u,\ell,h} \leftarrow \text{Attention}_{v,u,\ell,h} + \text{embed}_{\theta,h}(\texttt{dist}(u, v \mid T_m)).$$
If u and v are leaves then we also bias the attention using their Hamming distance:
$$\text{Attention}_{v,u,\ell,h} \leftarrow \text{Attention}_{v,u,\ell,h} + \text{MLP}_{\theta,h}(\texttt{sinusoidal}(\texttt{hamming}(X_u, X_v))).$$

Readout After the trunk we arrive at a vector $\texttt{embed}_\theta(v) = h_{v,0}^{(12)} \in \mathbb{R}^{256}$ for every vertex or leaf v in T_m which depends on both T_m and $X_{1:N}$. Now we must turn this into a set of scores for each possible T_{m-1}, which we then normalize to get a set of logits. To do so, we use a network which combines two vectors symmtrically:
$$\texttt{symlayer}_\theta(a, b) = \text{MLP}_\theta\left(a + b, a \circ b\right).$$

- **bottom-up**: for each pair of subtree roots u, v in the current forest,
$$\text{logit}_\theta(u, v) = \texttt{symlayer}_\theta\left(\texttt{embed}_\theta(u), \texttt{embed}_\theta(v)\right).$$
- **middle-out**: for each unattached taxon t and parent-to-child edge (u, v) in T_m,
$$\text{logit}_\theta(t, (u, v)) = \texttt{symlayer}_\theta\left(\texttt{embed}_\theta(t), e_{uv}\right),$$
where $e_{uv} = W_\theta\left[\texttt{embed}_\theta(u) \circ \texttt{embed}_\theta(v), \texttt{embed}_\theta(u) - \texttt{embed}_\theta(v)\right]$ for a matrix W_θ.
- **top-down**: each taxon t at the deepest layer of T_m sits at a host internal node with left and right children $\texttt{left}(t), \texttt{right}(t)$; the percolation logits are
$$\text{logit}_{\text{left}}(t) = \texttt{symlayer}_\theta\left(\texttt{embed}_\theta(t),\ \texttt{embed}_\theta(\texttt{left}(t))\right),$$
$$\text{logit}_{\text{right}}(t) = \texttt{symlayer}_\theta\left(\texttt{embed}_\theta(t),\ \texttt{embed}_\theta(\texttt{right}(t))\right).$$
if t is missing a child then the embedding of that child is set to be a fixed vector. Note for top-down there can be multiple taxa at a node; we therefore include all nodes in our trunk, even external ones, and $\texttt{embed}_\theta(\texttt{right}(t))$ is defined to be the embedding of the node on the right, regardless of how many taxa are at that node (and similar for the left).

G. Experiment details

G.1. Additional information on training data

We simulate data from three priors by first simulating a tree, and then simulating sequences conditional on the tree.

G.1.1. Dataset size

For each set of data point, we sample
$$N \sim \text{Uniform}(25, 65), L \sim \text{Uniform}(300, 2600).$$
Then we generate a tree with N species and sequences with length L before any insertions or deletions.

G.1.2. Trees

MrBayes (Huelsenbeck & Ronquist, 2001) and CAT (Lartillot & Philippe, 2004) both have uniform (proportional to distinguishable arrangements (PDA)) priors on tree topologies. We simulate PDA trees using the Ford-α process with $\alpha = 1/2$ (Ford, 2005).

G.1.3. Sequences and branch lengths

All simulations were done using AliSim (Ly-Trong et al., 2022), part of the IQ-TREE software used under the GPL-2.0 license. We use L to denote the length of the sequences.

Simple model To simulate from the Jukes-Cantor model, we passed `JC` to the model flag in AliSim. We simulate branch lengths independently from $\text{Exp}(1)/10$ to match the default MrBayes prior.

Complex model We simulate from a modified CAT model (Lartillot & Philippe, 2004) which models site-wise biases in mutation rates in amino acid sequences. This model has branch lengths $(r_{u\to v})_{u,v}$, class diversity parameter $\alpha \in (0, \infty)$, and a doubly stochastic matrix Q representing relative mutation rates as global parameters. Then each site ℓ mutates according to its rate g_ℓ and its stationary distribution $\vec{\pi}_\ell$. In particular, we (1) modify Q to get a stationary rate $\vec{\pi}_\ell$, $\tilde{R}_\ell = Q\text{diag}(\vec{\pi}_\ell) - \text{diag}(Q\vec{\pi}_\ell)$; (2) normalize the rates $R_\ell = \tilde{R}_\ell / \vec{\pi}_\ell^T \text{diag}(\tilde{R}_\ell)$; and finally (3) multiply by the rate $Q_\ell = g_\ell \times R_i$.

We simulate parameters from a CAT prior as used in PhyloBayes MPI (Lartillot et al., 2013). See the manual at `https://github.com/bayesiancook/phylobayes/tree/master`.

- $Q_{ij} \sim \text{Exp}(1)$ for $i < j$. This matches the `-gtr` flags.
- $\alpha \sim \text{Exp}(1) \times 10$.
- $r_{u\to v} \sim \text{Exp}(1)/\mu$ independently where $\mu \sim \text{Exp}(1) \times 10$.
- $(g_\ell)_{\ell=1}^L \sim \text{Gamma}(\theta, 1)/\theta$, discretized to 4 components, which has mean 1 and variance $1/\theta$, where $\theta \sim \text{Exp}(1)$.
- $(\vec{\pi}_\ell)_\ell^L \sim \text{DirichletProcess}(\alpha, \text{Dirichlet}(\eta_1, \ldots, \eta_{20}))$ where each $\eta_b \sim \text{Exp}(1)$. This matches the `-cat` flag. We simulate class probabilities using the standard stick-breaking procedure up to 200 classes.

Finally we can simulate this prior using AliSim by passing `GTR{...}+FMIX{...}+G4{...}` filling in the sampled parameters for Q, the probabilities and classes from the sampled stick breaking process, and θ.

Intractable model To simulate from a realistic prior including indels, we first simulate from a standard model of DNA mutation, `GTR{q_matrix}+GC{g_std}+I{invar_rate}`, where `q_matrix` is sampled from a Dirichlet as in the CAT model, `g_std` $\sim \text{Exp}(1)$, and `invar_rate` $\sim \text{Uniform}(0, 0.4)$. AliSim will simulate the stationary DNA distribution, what we called $\vec{\pi}$ in the CAT model, from an empirical distribution. We also simulate branch lengths from $\text{Exp}(1)/10$.

To add insertions and deletions, we pass `--indel ins_rate,del_rate` where we simulate `ins_rate` $\sim \text{Uniform}(0, 0.01)$ and `del_rate` $\sim \text{Uniform}(0, 0.05)$. This simulates insertions up to 1% and deletions up to 5% as frequently as substitutions; AliSim then samples insertion and deletion sizes from an empirically fit Zipfian distribution (Fletcher & Yang, 2009).

After simulating, we align sequences using `kalign3` (Lassmann, 2019) with default parameters before performing inference. Note models do not see the ground truth insertion and deletion labels generated during simulation, only those inferred by `kalign3`.

Missing-ness To simulate missing-ness, we draw a $p \sim \text{Beta}(1.25, 8.75)$ and give each non-gap position an independent $\text{Bernoulli}(p)$ chance of being masked.

G.2. Real sequence data

We downloaded sequence datasets from Lakner et al. (2008). We use DS1 originally taken from Hedges et al. (1990), DS6 original taken from Brower (2000), and DS9 originally taken from Rossman et al. (2001).

G.3. Inference with likelihood-based methods

We compared our models to MCMC sampling with MrBayes and a maximum likelihood based method, IQ-Tree.

G.3.1. MrBayes

We used code from `https://nbisweden.github.io/MrBayes/download.html` under a GPL-3.0 license. We ran MrBayes with a uniform stationary distribution `prset statefreqpr=fixed(equal)` and exponential branch lengths `prset brlenspr=unconstrained:exponential(10.0)`. Our gold standard runs were run for 5×10^8

steps, sampling every 1000 steps; after this, the standard deviation in subsplit probabilities across two runs was $\approx 10^{-4}$. We estimated marginal likelihoods using stepping stone sampling using the `ss` command.

G.3.2. IQ-TREE

We use IQ-Tree3 (Wong et al., 2026) code from `https://github.com/Cibiv/IQ-TREE` under a GPL-2.0 license. We ran IQ-tree with 16 CPUs with the flag `-nt AUTO` and otherwise used default settings. We ran IQ-Tree with `-m JC` for both DNA datasets and `-m Poisson` for the protein dataset. We could in principle run more flexible IQ-Tree models however the goal of our experiment was to demonstrate the drawbacks of using a misspecified model, a common choice in practice.

G.4. Evaluation metrics

We describe how we compare trees and their distributions using metrics derived from the Robinson-Foulds distance.

G.4.1. ROBINSON-FOULDS DISTANCE

The Robinson-Foulds (RF) metric is often used for comparing trees due to its ease of computation (Felsenstein, 2004, Ch. 30). The metric can be used to compare rooted and unrooted trees. Since JC and CAT are reversible, the position of the root is not identifiable, so we use the unrooted RF distance to compare all trees.

To compute the RF distance, we compute the partition features of each tree: for a binary partition $P_1, P_2 \subset \{1, \ldots, N\}$ a tree has that feature if there is an edge in the unrooted tree separating all nodes P_1 from all nodes P_2. Then the distance between two trees is simply the symmetric difference of their features.

G.4.2. COMPARING TREE DISTRIBUTIONS USING RF MMD

To compare two distributions p, q, using a kernel k, we can calculate

$$\mathrm{MMD}_k^2(p,q) = \mathbb{E}_{T_1\sim p, T_2\sim p} k(T_1,T_2) + \mathbb{E}_{T_1\sim q, T_2\sim q} k(T_1,T_2) - 2\mathbb{E}_{T_1\sim p, T_2\sim q} k(T_1,T_2).$$

We can build kernels using the RF distance: we can think of RF as embedding a tree T into $v(T) \in \mathbb{R}^{2^{N-1}}$ with dimensions indexed by binary partitions of $\{1, \ldots, N\}$, where, for a partition $P_1, P_2 \subset \{1, \ldots, N\}$ $v(T)_{\{P_1,P_2\}} = 1$ if there is an edge separating all nodes P_1 from all nodes P_2. Then the RF distance between trees T, T', is simply $\|v(T) - v(T')\|$.

We could compare distributions with the linear kernel $k(T,T') = v(T)^T v(T')$; the MMD in this case is simply the distance between the vectors of subsplit probabilities $\|\mathbb{E}_{T\sim p} v(T) - \mathbb{E}_{T\sim q} v(T)\|$. However, two distinct distributions can have zero MMD simply by matching the average subsplit probabilities. To more flexibly detect distributional differences we look at a more flexible kernel $k(T,T') = \exp(-\|v(T) - v(T')\|^2/2\sigma^2)$ where σ is chosen as the median RF distance in the dataset.

G.5. Phyloformer 2

We trained Phyloformer2 (Blassel et al., 2025) for 48 hours on two A100s — the same as our models. We used the training architecture and parameters from the supplementary information of `https://openreview.net/forum?id=18pFYYqFiG`. However we removed the branch length prediction head.

H. Model sampling timing comparisons

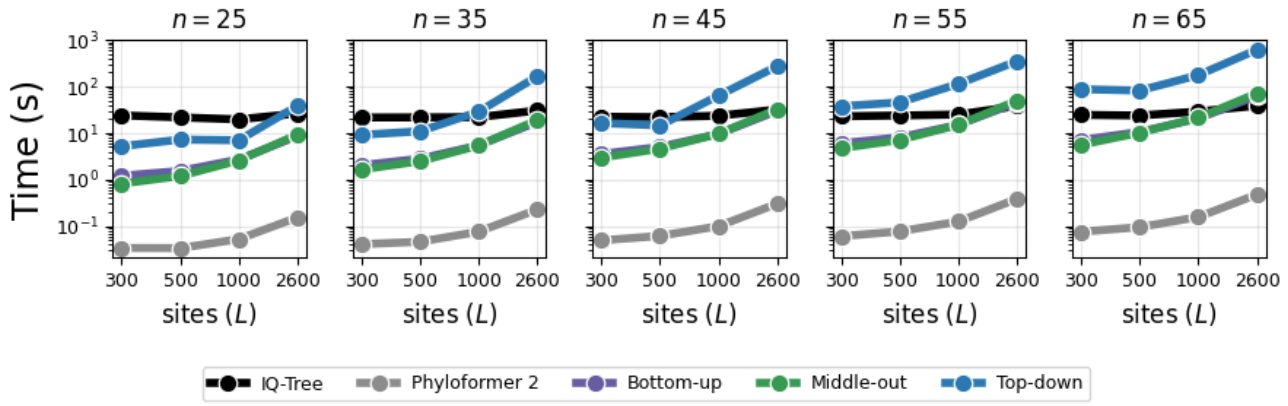


Figure 5. Comparison of sampling runtimes for our models and baselines.

Our model samples scale as N steps $\times O(NL^2)$ per step. Phyloformer 2 in principle scales as $O(N^4 + NL^2)$ but for these sizes of trees is relatively cheap at inference time as it only needs one forward pass to sample a tree. Despite the extra cost of our model, it is cheaper than the cheapest default settings of IQ-Tree for most parameter settings, making it appropriate for use in standard phylogenetics workflows. The exception is our top-down model, which needs many more iterative steps to generate a tree. The extra compute our models spend during sampling allow them to represent much more flexible distributions, bringing these resources to bear on this problem.