Neural Backward Filtering Forward Guiding

Gefan Yang
Department of Computer Science
University of Copenhagen
Universitesparken 1, 2100 København, Denmark
gy@di.ku.dk

Frank van der Meulen
Department of Mathematics
Vrije Universiteit Amsterdam
De Boelelaan 1111, 1081HV Amsterdam, The Netherlands
f.h.van.der.meulen@vu.nl

Stefan Sommer
Department of Computer Science
University of Copenhagen
Universitesparken 1, 2100 København, Denmark
sommer@di.ku.dk

Abstract

Inference in non-linear continuous stochastic processes on trees is challenging, particularly when observations are sparse and the topology is complex. Exact smoothing via Doob's h-transform is intractable for general nonlinear dynamics. We propose Neural Backward Filtering Forward Guiding (NBFFG), a unified framework for both discrete transitions and continuous diffusions. Our method constructs a variational posterior by leveraging a proxy linear-Gaussian process. This proxy process yields a closed-form backward filter that serves as a "guide", steering the generative path toward high-likelihood regions. We then learn a neural residual to capture the non-linear discrepancies. This formulation allows for an unbiased path-wise subsampling scheme, reducing the training complexity from tree-size dependent to path-length dependent. Empirical results show that NBFFG outperforms baselines on synthetic benchmarks. We also evaluate how neural correction handles proxy misspecification, and study nonlinear multimodal smoothing. On a high-dimensional butterfly phylogeny, we reconstruct ancestral shapes conditioned on observed species.

1 Introduction

Tree-structured stochastic processes are fundamental for modeling hierarchical dependencies in complex systems, from phylogenetic evolution in biology [Felsenstein, 1981, Huelsenbeck et al., 2003] and spin-glass state spaces in statistical physics [Bovier, 2017] to cascading diffusion in social networks [Kempe et al., 2003]. In these settings, a central inferential challenge is smoothing: reconstructing latent internal states and trajectories from partially available observations. Exact Bayesian smoothing is characterized by Doob's h-transform, where future observations enter through a harmonic h-function computed by collecting evidence from leaves to the root, equivalently by belief propagation or sum-product message passing. While this backward pass is analytically tractable for

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

simple Linear-Gaussian (LG) models, it becomes computationally prohibitive for general nonlinear dynamics.

Current approaches often rely on Sequential Monte Carlo (SMC) or Markov Chain Monte Carlo (MCMC). However, particle methods can suffer severe weight degeneracy as the hierarchy deepens, and MCMC can mix slowly when distant observations provide weak global information to internal states. We introduce Neural Backward Filtering Forward Guiding (NBFFG), a variational framework that approximates the intractable smoothing posterior by correcting a tractable guided proposal. The proposal is built from a cheap proxy backward filter, which replaces the exact Doob h-function with a closed-form approximation and steers samples toward regions compatible with descendant observations. This proxy need not be exact: it only guides the training trajectories towards the observations, but not necessarily following the exact posterior paths.. A neural residual then corrects the induced proposal over the tree, with parameterizations adapted to discrete transitions and continuous diffusion edges.

Our contributions are:

- We formulate tree smoothing as neural correction of a guided proposal: a cheap proxy backward filter provides global descendant information, while a neural residual learns to correct proxy mismatch. We further study the capability of correction through a series of controlled experiments.
- We give a unified variational construction for discrete tree transitions and continuous diffusion edges, instantiated as Gaussian-mixture corrections and controlled SDEs.
- We introduce path-subsampled training with amortized correction over root-to-node contexts,reducing each update from the full tree to sampled root-to-leaf paths.

We evaluate these claims on LG benchmarks with Gaussian smoothing references, nonlinear multimodal systems, and high-dimensional phylogenetic shape inference.

2 Problem Setup

2.1 Generative process on the tree

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed tree rooted at a fixed vertex $\mathbf{0}$. We write $\mathcal{V}^+ = \mathcal{V} \setminus \{\mathbf{0}\}$ for the non-root vertices, $\mathcal{L} \subset \mathcal{V}$ for the leaves, and $\mathcal{S} = \mathcal{V} \setminus (\{\mathbf{0}\} \cup \mathcal{L})$ for the non-root internal vertices. The root is fixed at $X_{\mathbf{0}} = x_{\mathbf{0}}$, and each non-root vertex $v \in \mathcal{V}^+$ carries a random variable X_v taking values in a measurable space $\mathcal{X}_v$. We denote by $\mathrm{pa}(v)$ and $\mathrm{ch}(v)$ the parent and children of v, respectively. Let $\mathcal{L}_v \subseteq \mathcal{L}$ denote the leaves of the subtree rooted at v.

All leaves are observed terminal variables. Observations of an internal state are represented by attaching observed leaf children to that state, with multiple observations corresponding to multiple leaves. Thus latent dynamics and observation models are both treated as edge transitions, with leaf edges clamped to their observed child values.

The prior is a Markov generative process on the tree: conditional on a parent state, the subtrees below its children evolve independently. Hence the joint law over non-root variables factorizes edge-wise as

$$\mathbb{P}(\mathrm{d}x_{\mathcal{V}^+} \mid x_{\mathbf{0}}) = \prod_{v \in \mathcal{V}^+} P_v(x_{\mathrm{pa}(v)}, \mathrm{d}x_v), \tag{1}$$

where $P_v : \mathcal{X}_{\mathrm{pa}(v)} \rightharpoonup \mathcal{X}_v$ is the Markov kernel along $(\mathrm{pa}(v), v)$. This notation covers both transition types—discrete and continuous transitions—considered in the paper. For discrete edges, we use Gaussian kernels on $\mathbb{R}^d$,

$$P_v(x, \mathrm{d}y) = \mathcal{N}(\mathrm{d}y; \mu_v(x), \Sigma_v(x)) \,. \tag{2}$$

For continuous edges, we distinguish the vertex state X_v on $\mathbb{R}^d$ from the within-edge path $Z_v(t)$ on the continuous path space $\mathcal{C}([0, T_v], \mathbb{R}^d)$. Conditional on $X_{\mathrm{pa}(v)} = x$, the path starts at the parent value and ends at the child state:

$$\mathrm{d}Z_v(t) = b_v(Z_v(t))\mathrm{d}t + \sigma_v(Z_v(t))\mathrm{d}W(t), \qquad Z_v(0) = x, \quad X_v = Z_v(T_v). \tag{3}$$

The corresponding vertex-to-vertex kernel is the endpoint law of this diffusion. A measure-theoretic formulation of these kernels is given in Appendix A.

2.2 Smoothing on the tree

We now turn from the generative description to the smoothing problem. For each leaf $l \in \mathcal{L}$, assume the leaf-edge kernel is dominated by a reference measure ν_l, so $P_l(x, \mathrm{d}y) = p_l(y \mid x)\nu_l(\mathrm{d}y)$. In the smoothing problem, the child value on this edge is observed as $X_l = x_l$, and the density above is evaluated at the observed value. Given observed leaf values $x_{\mathcal{L}}$ and fixed root value $X_{\mathbf{0}} = x_{\mathbf{0}}$, the smoothing posterior is the probability measure on unobserved internal realizations $x_{\mathcal{S}}$:

$$\mathbb{P}^\star(\mathrm{d}x_{\mathcal{S}} \mid x_{\mathcal{L}}, x_{\mathbf{0}}) = \frac{1}{Z(x_{\mathcal{L}}, x_{\mathbf{0}})} \prod_{l \in \mathcal{L}} p_l(x_l \mid x_{\mathrm{pa}(l)}) \prod_{s \in \mathcal{S}} P_s(x_{\mathrm{pa}(s)}, \mathrm{d}x_s), \tag{4}$$

where $Z(x_{\mathcal{L}}, x_{\mathbf{0}})$ is the marginal likelihood. The inferential task is to characterize this posterior, especially its marginals at unobserved internal vertices. For example, given a phylogenetic tree with known topology and shape observations from fossils or extant species, smoothing infers ancestral shapes and their uncertainty at unobserved vertices. Except in special cases such as linear models, the normalizing constant and posterior marginals in Equation (4) are not available in closed form.

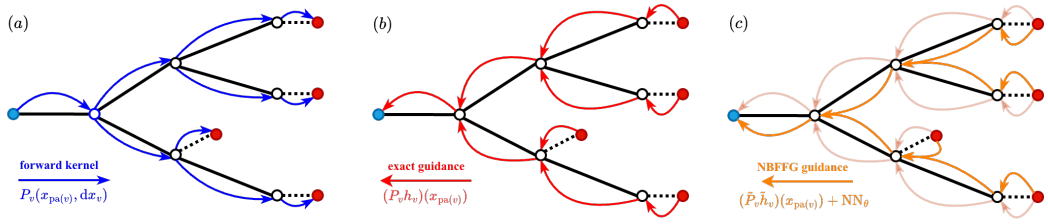


Figure 1: Smoothing on a directed tree with a fixed root (blue solid dot). Leaves (red solid dots) are observed, internal vertices (black hollow dots) are latent, dashed lines indicate the observations/emissions, and arrows indicate computation direction: (a) generative sampling from root to leaves; (b) exact backward information filtering from leaves to root; (c) NBFFG framework, where a tractable proxy guide is corrected by a neural residual.

3 Backward Filtering Forward Guiding

Under the setup above, exact Bayesian smoothing can be written as message passing on the tree. A backward pass propagates observed leaf information toward the root, producing at each non-leaf vertex the conditional density of its descendant observations. This is the tree analogue of the backward information filter and phylogenetic pruning recursion [Felsenstein, 1981, Briers et al., 2010], and is equivalently a sum-product or belief-propagation pass on a tree [Pearl, 1988, Kschischang et al., 2001]. A subsequent forward pass samples from the root toward the leaves using transitions conditioned on these backward messages.

Let $\mathcal{S}_0 := \mathcal{S} \cup \{\mathbf{0}\}$ denote the vertices that are not observed leaves. For an edge entering an unobserved internal vertex $s \in \mathcal{S}$, the conditioned transition $P^\star$ is obtained by a Doob h-transform of the prior transition P:

$$P_s^\star(x, \mathrm{d}y) = \frac{h_s(y)}{m_s(x)} P_s(x, \mathrm{d}y), \qquad m_s(x) := (P_s h_s)(x), \tag{5a}$$

$$h_u(x) = \prod_{v \in \mathrm{ch}(u)} m_v(x), u \in \mathcal{S}_0, \qquad m_v(x) := \begin{cases} (P_v h_v)(x), & v \in \mathcal{S}, \\ p_v(x_v \mid x), & v \in \mathcal{L}. \end{cases} \tag{5b}$$

where we use the notation of pullback $(P_s h_s)$ defined by Equation (18), and denote $h_u(x)$ as the density contribution of the observed leaves below u conditioned on $X_u = x$, and m_v is the message sent from child v to its parent. This representation is exact, but evaluating the message $m_s = (P_s h_s)$ is generally intractable. For continuous-time transitions, the same pullback is characterized by the Kolmogorov backward equation:

$$\left(\frac{\partial}{\partial t} + \mathcal{A}_s\right) \psi_s(t, z) = 0, \qquad \psi_s(T_s, z) = h_s(z), \tag{6}$$

so that $(P_s h_s)(x) = \psi_s(0, x)$ by Feynman-Kac formula. Here $\mathcal{A}_s$ is the infinitesimal generator of Equation (3), defined by $(\mathcal{A}_s f)(\cdot, z) := \sum_i [b_s(z)]_i \partial_i f(\cdot, z) + \frac{1}{2} \sum_{i,j} [(\sigma_s \sigma_s^\top)(z)]_{ij} \partial_{i,j} f(\cdot, z)$.

To address this, van der Meulen and Sommer [2025] construct tractable guiding messages $\tilde{m}_s$ by replacing the true transition kernels in the backward pass with proxy kernels $\tilde{P}_s$. These kernels are chosen so that their corresponding pullbacks can be evaluated in closed form:

$$\tilde{h}_u(x) = \prod_{v \in \mathrm{ch}(u)} \tilde{m}_v(x), u \in \mathcal{S}_0, \qquad \tilde{m}_v(x) := \begin{cases} (\tilde{P}_v \tilde{h}_v)(x), & v \in \mathcal{S}, \\ p_v(x_v \mid x), & v \in \mathcal{L}. \end{cases} \tag{7}$$

Once $\tilde{h}_s(x)$ is specified for all $s \in \mathcal{S}$, define

$$P_s^\circ(x, \mathrm{d}y) := \frac{\tilde{h}_s(y)}{(P_s \tilde{h}_s)(x)} P_s(x, \mathrm{d}y) \tag{8}$$

as the *guided proposal* of the exact smoothing kernel $P_s^\star(x, \mathrm{d}y)$. The proxy kernels $\tilde{P}_s$ are used only to build the guide $\tilde{h}_s$; while the guided proposal is normalized under the original kernel P_s. The ratio between the exact smoothing distribution of the full tree $\mathbb{P}^\star$ and the proposal distribution $\mathbb{P}^\circ$ induced by P_s° can then be evaluated explicitly [van der Meulen and Sommer, 2025, Theorem 8]:

$$\frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}^\circ}(x_\mathcal{S}) = \frac{\tilde{h}_\mathbf{0}(x_\mathbf{0})}{h_\mathbf{0}(x_\mathbf{0})} \prod_{v \in \mathcal{V}^+} \omega_v(x_{\mathrm{pa}(v)}), \qquad \omega_v(x) = \begin{cases} (P_v \tilde{h}_v)(x) / (\tilde{P}_v \tilde{h}_v)(x), & v \in \mathcal{S}, \\ 1, & v \in \mathcal{L}. \end{cases} \tag{9}$$

This ratio can be used as an importance weight in SMC or MCMC-based sampling frameworks. Such a filtering-guiding procedure is referred to as Backward Filtering Forward Guiding (BFFG). The filtering procedure is illustrated in Figure 1 (b). Directly learning the exact h from prior samples is difficult: descendant evidence may enter at different tree depths and through multiple states, while uninformed prior trajectories rarely visit high-likelihood regions under informative leaves. This motivates using a guided proposal as the base distribution, a point we examine in Section 5.1.

4 Neural Backward Filtering Forward Guiding

4.1 A canonical choice of proxy

The choice of proxy kernel $\tilde{P}_s$ in BFFG can be nontrivial in general nonlinear models. There is an inherent trade-off between computational efficiency and sampling quality. A simple proxy may lead to closed-form backward messages and an easily sampled guided proposal, but it can also produce high-variance importance weights when $\mathbb{P}^\circ$ is far from $\mathbb{P}^\star$. While model-specific strategies for proxy selection exist [Mider et al., 2021], NBFFG holds the proxy guide fixed during neural training and uses its messages to convey information from descendants. The residual is learned under the true-model ELBO. Appendix F.1 examines how well it corrects proxy mismatch.

For the models considered in this work, we therefore use a canonical proxy that ignores the nonlinear mean μ_v in Equation (2) or the drift b_v in Equation (3), while retaining a state-independent covariance or diffusion scale:

$$\text{discrete proxy:} \quad \tilde{P}_v(x, \mathrm{d}y) = \mathcal{N}(\mathrm{d}y; x, \tilde{\Sigma}_v), \tag{10a}$$

$$\text{continuous proxy:} \quad \mathrm{d}\tilde{Z}_v(t) = \tilde{\sigma}_v(t) \mathrm{d}\tilde{W}(t), \qquad \tilde{Z}_v(0) = x. \tag{10b}$$

For the continuous proxy, we write $\tilde{P}_{v, t \to T_v}, 0 \le t \le T_v$ for the proxy transition semigroup from time t to the endpoint T_v: $(\tilde{P}_{v, t \to T_v} f)(z) = \mathbb{E}_{\tilde{P}_v}\Big[f(\tilde{Z}_v(T_v)) \mid \tilde{Z}_v(t) = z \Big]$. When the leaf-edge observation models are also Gaussian, this proxy yields closed-form guiding messages and closed-form scores for continuous edges.

Proposition 4.1 (Gaussian proxy guide)**.** *Assume Gaussian leaf likelihoods $p_l(x_l \mid x) = \mathcal{N}(x_l; L_l x + \beta_l, R_l)$ and the proxy kernels in Equation* (10)*, with nonsingular information matrices. Then the backward proxy guides remain quadratic:*

$$\log \tilde{h}_u(x) = -\frac{1}{2} x^\top \tilde{H}_u x + \tilde{\eta}_u^\top x + \mathrm{const.}$$

For a continuous edge (u, v), the within-edge guide $\tilde{h}_{v,t}(z) := (\tilde{P}_{v,t\to T_v}\tilde{h}_v)(z)$ is also quadratic, so its score is affine:

$$\nabla_z \log \tilde{h}_{v,t}(z) = -\tilde{H}_{v,t}z + \tilde{\eta}_{v,t}.$$

The information parameters are given by the closed-form backward recursion in Appendix E.

This gives exactly the quantities needed below: in discrete time, the proxy guide yields the base guided proposal through Equation (8); in continuous time, its affine score serves as part of the SDE control. We do not require $\tilde{h}_v$ or $\tilde{h}_{v,t}$ to be exact. Their role is to conduct the message-preconditioned learning by directing samples towards the conditioned states, while the neural component introduced next learns the residual correction to the intractable h_v or $h_{v,t}$.

4.2 Variational posterior

We now define the neural correction at the level of Markov kernels. For each unobserved internal vertex $s \in \mathcal{S}$, let $Q_s^\theta(x_{\mathrm{pa}(s)}, \mathrm{d}x_s)$ be a variational transition kernel parameterized by θ, absolutely continuous with respect to the prior transition $P_s(x_{\mathrm{pa}(s)}, \mathrm{d}x_s)$. We define the joint variational law as the tree-structured measure

$$\mathbb{Q}^\theta(\mathrm{d}x_{\mathcal{S}} \mid x_{\mathbf{0}}) = \prod_{s\in\mathcal{S}} Q_s^\theta(x_{\mathrm{pa}(s)}, \mathrm{d}x_s). \tag{11}$$

The leaves are not sampled under $\mathbb{Q}^\theta$; they are observed boundary values and enter the objective through the leaf-edge densities $p_l(x_l \mid x_{\mathrm{pa}(l)})$.

The concrete parameterization of Q_s^θ will differ between discrete and continuous edges. The common principle is that Q_s^θ is built by correcting the proxy-induced guide rather than by approximating the posterior from scratch. The following objective, however, does not depend on how the correction is parameterized.

Proposition 4.2 (Tree ELBO)**.** *Assume $Q_s^\theta(x, \cdot) \ll P_s(x, \cdot)$ for all $s \in \mathcal{S}$. Define*

$$\mathcal{J}(\theta) = \mathbb{E}_{\mathbb{Q}^\theta}\left[\sum_{s\in\mathcal{S}} \log \frac{\mathrm{d}Q_s^\theta(X_{\mathrm{pa}(s)}, \cdot)}{\mathrm{d}P_s(X_{\mathrm{pa}(s)}, \cdot)}(X_s) - \sum_{l\in\mathcal{L}} \log p_l(x_l \mid X_{\mathrm{pa}(l)})\right]. \tag{12}$$

Then

$$\mathcal{J}(\theta) = D_{\mathrm{KL}}\big(\mathbb{Q}^\theta(\cdot \mid x_{\mathbf{0}}) \,\big\|\, \mathbb{P}^\star(\cdot \mid x_{\mathcal{L}}, x_{\mathbf{0}})\big) - \log Z(x_{\mathcal{L}}, x_{\mathbf{0}}).$$

Thus minimizing $\mathcal{J}(\theta)$ is equivalent to minimizing the KL divergence from $\mathbb{Q}^\theta$ to the exact smoothing posterior $\mathbb{P}^\star$.

This objective makes the role of the proxy explicit. The target posterior and the ELBO are defined using the true transitions P_s and the true leaf-edge densities. The proxy guide $\tilde{h}$ enters only through the parameterization of Q_s^θ. We then specialize in two different edge dynamics—discrete and continuous—leading to different variational implementations.

Discrete transitions For a discrete internal edge $s \in \mathcal{S}$, the proxy guide gives an explicit Gaussian base. Suppose $P_s(x, \mathrm{d}y) = \mathcal{N}(\mathrm{d}y; \mu_s(x), \Sigma_s(x))$ and $\log \tilde{h}_s(y) = \kappa_s + \tilde{\eta}_s^\top y - \frac{1}{2}y^\top \tilde{H}_s y$. Then Equation (8) gives $P_s^\circ(x, \mathrm{d}y) = \mathcal{N}(\mathrm{d}y; \mu_s^\circ(x), \Sigma_s^\circ(x))$, with

$$\Sigma_s^\circ(x) = \left(\Sigma_s(x)^{-1} + \tilde{H}_s\right)^{-1}, \qquad \mu_s^\circ(x) = \Sigma_s^\circ(x)\left(\Sigma_s(x)^{-1}\mu_s(x) + \tilde{\eta}_s\right). \tag{13}$$

We parameterize $Q_s^\theta(\cdot \mid x)$ as a neural Gaussian mixture that is initialized at, and corrected around, this guided base. Increasing the number of mixture components lets the variational transition represent skewed or multimodal smoothing structure, while the ELBO only requires sampling from Q_s^θ and evaluating $\log q_s^\theta - \log p_s$. The exact network parameterization is given in Appendix B.1.

Continuous transitions For a continuous edge, the variational object is a path measure on $\mathcal{C}([0, T_s], \mathbb{R}^d)$. We use the same notation Q_s^θ for this path law; its endpoint pushforward gives the vertex-level transition appearing in Equation (11). Let $a_s(z) = \sigma_s(z)\sigma_s(z)^\top$. We define a controlled SDE

$$\mathrm{d}Z_s(t) = \big[b_s(Z_s(t)) + a_s(Z_s(t))g_s^\theta(t, Z_s(t))\big]\,\mathrm{d}t + \sigma_s(Z_s(t))\mathrm{d}W_t^{\mathbb{Q}^\theta}, \qquad Z_s(0) = x_{\mathrm{pa}(s)}, \tag{14}$$

where the control decomposes into the fixed proxy score and a neural residual,

$$g_s^\theta(t,z) = \nabla_z \log \tilde{h}_{s,t}(z) + r_s^\theta(t,z) = -\tilde{H}_{s,t} z + \tilde{\eta}_{s,t} + r_s^\theta(t,z). \tag{15}$$

Under standard Novikov-type conditions, Girsanov's theorem gives the edge-wise KL contribution

$$D_{\mathrm{KL}}(Q_s^\theta \| P_s) = \frac{1}{2}\mathbb{E}_{Q_s^\theta}\left[\int_0^{T_s} g_s^\theta(t, Z_s(t))^\top a_s(Z_s(t)) g_s^\theta(t, Z_s(t)) \mathrm{d}t\right]. \tag{16}$$

The affine proxy score steers paths toward descendant observations, and the residual field r_s^θ learns the nonlinear correction needed by the exact posterior; implementation details are given in Appendix B.1. Thus, in both discrete and continuous time, the target posterior and ELBO use the true generative model, while the proxy supplies only a structured base for neural correction.

4.3 Tree-path subsampling

The tree ELBO is a sum of local transition and leaf-edge costs. Evaluating every term at each optimization step can be expensive on large trees, especially for continuous edges where each transition requires simulating a controlled path. NBFFG therefore uses topology-level path subsampling: sample a small set of leaves and simulate only the union $A(B)$ of their root-to-leaf paths, with $A_0(B) = A(B) \cup \{\mathbf{0}\}$. This is distinct from the time discretization of a continuous SDE edge.

Let C_s^θ denote the local variational cost on an internal edge $s \in \mathcal{S}$ and let $C_l = -\log p_l(x_l \mid X_{\mathrm{pa}(l)})$ be the observed leaf-edge cost. With inclusion probabilities $\pi_s = \mathbb{P}\{s \in A(B)\}$ and $\rho_l = \mathbb{P}\{\mathrm{pa}(l) \in A(B) \cup \{\mathbf{0}\}\}$, we optimize the path loss

$$\widehat{\mathcal{J}}_B(\theta) = \sum_{s \in A(B)} \frac{C_s^\theta}{\pi_s} + \sum_{\substack{l \in \mathcal{L} \\ \mathrm{pa}(l) \in A_0(B)}} \frac{C_l}{\rho_l} \tag{17}$$

which is an unbiased estimator of the full-tree objective under the sampled path-union law. The first sum is the standard Horvitz–Thompson correction for sampled internal edges. The second sum uses parent-inclusion weighting: it includes every observed child of each sampled parent and weights its cost by that parent's inclusion probability. This preserves unbiasedness; its effect on variance depends on the sampling design. Appendix B.2 gives the formal statement, implementation details, and proof.

4.4 Amortized correction

The neural correction can be parameterized independently on every edge, but this scales poorly and prevents information sharing across related branches. We instead use shared parameters with edge-specific context:

$$Q_s^\theta(\cdot \mid x) = Q^\theta(\cdot \mid x; \psi_s), \qquad s \in \mathcal{S},$$

where ψ_s summarizes the root-to-node structural context of edge s. In discrete time, the shared network predicts residual mixture parameters around the guided Gaussian base in Equation (13); in continuous time, it predicts the residual control r_s^θ in Equation (15). The proxy supplies coarse descendant information, while the amortized neural component learns corrections reused across sampled and unsampled branches. Appendix B.1 gives the correction networks, and Appendix B.3 gives the path encoder.

Algorithms 1 and 2 summarize training and objective estimation. Here $\mathcal{S}_{\mathrm{d}} \subseteq \mathcal{S}$ indexes discrete internal edges; a full-tree training without path subsampling uses $B = \mathcal{L}$ and $\pi_s = \rho_l = 1$.

Algorithm 1 NBFFG training

Require: $\mathcal{G}, x_{\mathbf{0}}, x_{\mathcal{L}}, P = (P_s)_{s\in\mathcal{S}}, p = (p_l)_{l\in\mathcal{L}}, \tilde{P}$
Require: $\theta_0, \Gamma, \pi = (\pi_s)_{s\in\mathcal{S}}, \rho = (\rho_l)_{l\in\mathcal{L}}, N, K, (\eta_k)_{k=1}^K$
Ensure: $\mathbb{Q}^\theta$

```
h̃ ← BackwardFilter(P̃, p, x_L); θ ← θ_0                    ▷ solve Eq. (7)
for k = 1, ..., K do
    Ĵ ← TreePathSubsampling(G, x_0, x_L, P, p, Q^θ, Γ, π, ρ, N)    ▷ see Alg. 2
    θ ← OptimizerStep(θ, ∇_θ Ĵ, η_k)
end for
return Q^θ
```

Table 1: Core Gaussian tree benchmark results. Values are mean $\pm$ std. over five runs. The proxy column uses $(\tilde{A}, \tilde{b})$ for the discrete LG model and $(\tilde{B}, \tilde{\theta})$ for the OU model. We show only Δ_{rel} and $\mathrm{KL}_{\mathrm{avg}}$ from Equation (29); complete diagnostics are reported in Tables 2 and 3. Lower is better.

1A: Discrete Linear Gaussian

	$\Delta_{\mathrm{rel}} \downarrow$		$\mathrm{KL}_{\mathrm{avg}} \downarrow$	
Proxy	Uncorr.	Corr.	Uncorr.	Corr.
No guidance	$1.70\times10^{4} \pm 7.34\times10^{3}$	96.659 ± 15.237	9.670 ± 2.548	75.495 ± 24.010
(A, b)	0.000 ± 0.000	0.002 ± 0.001	0.056 ± 0.004	0.060 ± 0.010
$(I, 0)$(10a)	0.913 ± 0.230	0.035 ± 0.011	1.861 ± 0.096	0.118 ± 0.004
$(-A, b)$	1.227 ± 0.322	0.016 ± 0.005	1.677 ± 0.602	0.080 ± 0.017
$(A, -b)$	3.352 ± 0.800	0.046 ± 0.015	6.559 ± 0.051	0.096 ± 0.022
$(-A, -b)$	4.482 ± 0.743	0.078 ± 0.040	6.690 ± 1.073	0.111 ± 0.020

1B: Continuous Heterogeneous OU

	$\Delta_{\mathrm{rel}} \downarrow$		$\mathrm{KL}_{\mathrm{avg}} \downarrow$	
Proxy	Uncorr.	Corr.	Uncorr.	Corr.
No guidance	29.245 ± 17.626	0.605 ± 0.226	11.904 ± 1.922	0.144 ± 0.022
(B, θ)	0.892 ± 0.583	0.213 ± 0.075	0.261 ± 0.051	0.048 ± 0.011
$(0, 0)$(10b)	1.273 ± 0.724	0.262 ± 0.104	0.387 ± 0.121	0.062 ± 0.008
(DBD, θ)	1.294 ± 0.648	0.214 ± 0.073	0.431 ± 0.051	0.045 ± 0.004
$(B, -\theta)$	5.122 ± 1.722	0.244 ± 0.092	1.381 ± 0.208	0.056 ± 0.008
$(DBD, -\theta)$	3.955 ± 1.470	0.264 ± 0.111	1.105 ± 0.105	0.062 ± 0.009

Algorithm 2 Tree-path subsampling

Require: $\mathcal{G}, x_{\mathbf{0}}, x_{\mathcal{L}}, P, p, \mathbb{Q}^{\theta}, \Gamma, \pi, \rho, N; \pi_s > 0,\ \rho_l > 0$
$B \sim \Gamma$; $A \leftarrow \bigcup_{l \in B} \{s \in \mathcal{S} : s \preceq l\}$
$A_0 \leftarrow A \cup \{\mathbf{0}\}$; $\mathcal{L}_A \leftarrow \{l \in \mathcal{L} : \mathrm{pa}(l) \in A_0\}$
for $n = 1, \ldots, N$ **do**
  $X_{\mathbf{0}} \leftarrow x_{\mathbf{0}}$
  for $s \in \textsc{TopSort}(A)$ **do** ▷ ensure parent before child
    if $s \in \mathcal{S}_{\mathrm{d}}$ **then** ▷ discrete edges
      $X_s \sim Q_s^{\theta}(\cdot \mid X_{\mathrm{pa}(s)})$; $C_s \leftarrow \log \frac{q_s^{\theta}(X_s | X_{\mathrm{pa}(s)})}{p_s(X_s | X_{\mathrm{pa}(s)})}$
    else ▷ continuous edges
      $Z_s \sim Q_s^{\theta}(\cdot \mid X_{\mathrm{pa}(s)})$; $X_s \leftarrow Z_s(T_s)$ ▷ solve Eq. (14)
      $C_s \leftarrow \frac{1}{2} \int_0^{T_s} \|\sigma_s(Z_s : (t))^{\top} g_s^{\theta}(t, Z_s(t))\|_2^2 \, \mathrm{d}t$
    end if
  end for
  $\widehat{\mathcal{J}}^{(n)} \leftarrow \sum_{s \in A} \frac{C_s}{\pi_s} - \sum_{l \in \mathcal{L}_A} \frac{\log p_l(x_l | X_{\mathrm{pa}(l)})}{\rho_l}$
end for
return $N^{-1} \sum_{n=1}^{N} \widehat{\mathcal{J}}^{(n)}$

5 Experiments

5.1 Gaussian tree benchmarks with smoothing references

We first use Gaussian tree models with analytically computable smoothing references. Benchmarks 1A and 1C use heterogeneous linear-Gaussian edges $X_v = A_v X_{\mathrm{pa}(v)} + b_v + \epsilon_v$, $\epsilon_v \sim \mathcal{N}(0, Q)$. Benchmarks 1B and 1D use heterogeneous Ornstein–Uhlenbeck edges $dZ_v(t) = B_v(\theta_v - Z_v(t))dt + \sigma dW_t$, whose closed-form endpoint transitions give Gaussian tree smoothing references. Benchmarks 1A–1B test proxy misspecification, while 1C–1D isolate path subsampling under the canonical proxy. We use five independent observation instances for the proxy benchmarks and report metrics from Equation (29); full model, topology, and training details are in Appendices D.1–D.4.

We compare several proxy coefficient choices with a no-guidance ablation. Table 1 shows that neural correction sharply reduces the relative NELBO gap in the discrete benchmark and gives a more variable but consistent reduction in the continuous benchmark. Guided correction also achieves a better final objective than no guidance in these settings. We also perform the analyse on the sensitivity to different choices of proxy, see Appendix F.1 for details.

Benchmarks 1C–1D compare direct leaf subsampling with terminal-group subsampling, which selects terminal latent vertices together with all their attached observation edges. Both retain the full latent prefix and subsample auxiliary observation edges. Small observation budgets can increase accuracy variability, while moderate budgets offer useful speedups. Savings are larger in the OU benchmark because each omitted observation edge avoids many Euler–Maruyama steps. Appendices D.3 and D.4 report the accuracy–speed trade-offs; Appendix F.6 measures continuous backpropagation cost separately.

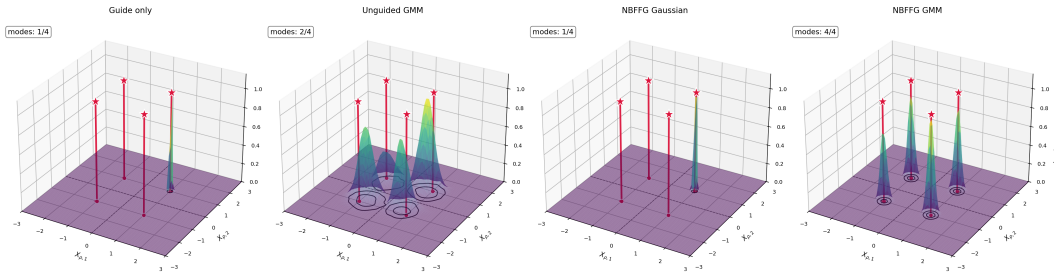


Figure 2: Root posterior samples for Benchmark 2A. The folded nonlinear transition creates four equiprobable root modes under the grid reference. NBFFG with a Gaussian-mixture correction recovers the multimodal posterior, while the raw guide and single-Gaussian correction collapse to one quadrant.

5.2 Nonlinear multimodal smoothing

We next test whether the discrete Gaussian-mixture correction can represent posterior multimodality induced by nonlinear hidden dynamics. Benchmark 2A uses a folded nonlinear Gaussian transition $X_v = f(X_\rho) + \epsilon_v$, $f(x) = (x_1^2, x_2^2)$, with linear Gaussian observation leaves fixed at $(1, 1)$. The four sign choices of the stochastic root are therefore all plausible, and a two-dimensional grid quadrature gives a reference posterior with four equiprobable quadrant modes. Details are given in Appendix D.5.

Figure 2 shows that the raw guide and the single-Gaussian correction collapse to one quadrant, while NBFFG with four mixture components recovers the four-mode root posterior in the representative run shown here. An unguided Gaussian mixture is less stable across seeds, and one NBFFG seed covers only three modes; full diagnostics are reported in Table 4. Overall, this isolates the role of the guided Gaussian-mixture correction in representing multimodal discrete smoothing posteriors. We also evaluate the sensitivity to the number of mixtures K and the choice of proxy covariance quantitatively. See Appendices F.2 and F.3, respectively.

Benchmark 2B moves to continuous nonlinear smoothing. Each hidden edge follows the double-well diffusion $\mathrm{d}Z_v(t) = -4\alpha Z_v(t)(Z_v(t)^2 - 1)\mathrm{d}t + \sigma \mathrm{d}W_t$, with Gaussian observation leaves attached to terminal latent vertices. We use two observation regimes. The early-commitment regime requires the two main branches to separate near the root, while the bimodal regime leaves competing explanations for the branch assignments. We compare the prior, uncorrected guide, NBFFG, guided SMC, and guided pCN against numerical marginal references for the discretized target (Appendix D.6).

Figure 3 compares sampled paths from all five methods. Against numerical references for the 100-step Euler–Maruyama target, neural correction reduces mean root-child marginal W_1 from 0.759 to 0.0046 in early commitment and from 0.639 to 0.110 under competing assignments. Table 5 and Figure 10 report marginal errors and distributions, with sampling budgets specified separately. NBFFG has the lowest mean marginal error in these configurations, but bias remains under competing assignments. Appendix F.5 examines how the target changes with temporal resolution.

5.3 Phylogenetic butterfly shape inference

As a real-world application, we apply NBFFG to phylogenetic shape inference on butterfly wing outlines. The latent state is a high-dimensional landmark representation of wing shape, and observed extant species are attached as leaf observations on the phylogeny. Unlike the previous benchmarks, no exact posterior is available; the goal is therefore to assess whether NBFFG can produce observation-conditioned ancestral reconstructions on a realistic nonlinear tree-structured problem. Details are given in Appendix D.7.

Figure 4 compares NBFFG ancestral mean shapes (blue) with forward-prior means conditional on the fixed root (dashed gray). The differences show how descendant observations affect the reconstructions. A separate evaluation with 512 paths gives NELBO -3565.86 for NBFFG and -3546.60 for a trained no-guidance model. Both models use one training seed and differ in training precision; Appendix D.7 reports these settings and Monte Carlo uncertainty. This qualitative result demonstrates that NBFFG scales beyond synthetic benchmarks to high-dimensional nonlinear phylogenetic smoothing.

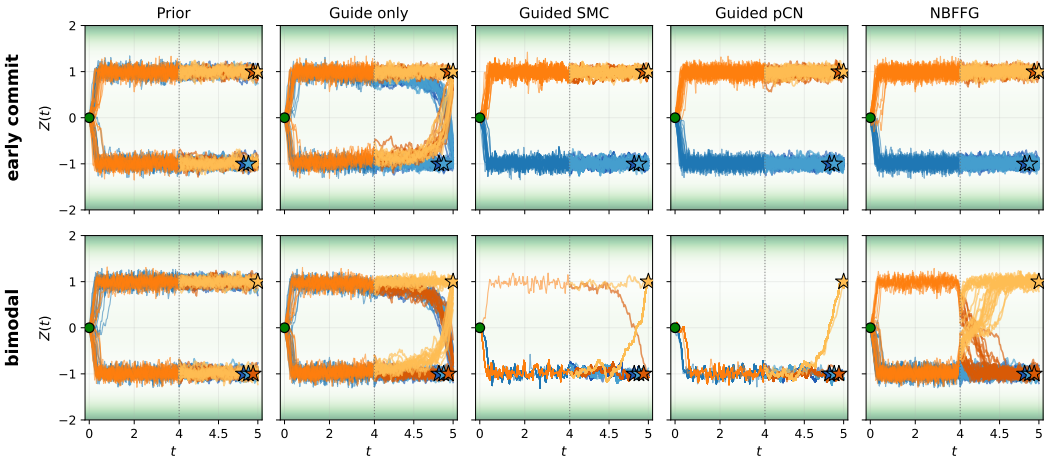


Figure 3: Sampled latent paths for Benchmark 2B under early-commitment (top) and competing-assignment (bottom) observations. Columns show the prior, uncorrected guide, guided SMC, guided pCN, and NBFFG; stars mark observed leaves. Each panel displays 30 paths. SMC resampling duplicates and pCN chain dependence are retained. In the bimodal regime, guided pCN collapses to one mode and guided SMC suffers path degeneracy.

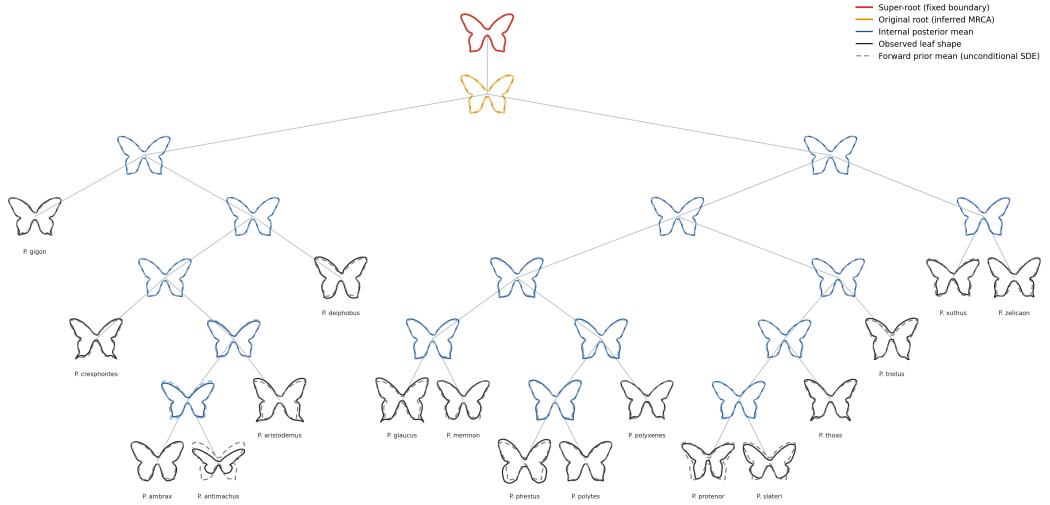


Figure 4: Phylogenetic butterfly shape inference. Observed extant species are shown in black, NBFFG posterior means at internal nodes in blue, and forward-prior means conditional on the fixed super root in dashed gray. NBFFG produces ancestral shape reconstructions conditioned on the observed leaves, while the forward prior omits the leaf likelihood.

6 Related Work

State-space smoothing and message passing For linear-Gaussian state-space models, the Rauch–Tung–Striebel smoother gives exact smoothing recursions [Rauch et al., 1965]; see Särkkä and Svensson [2023, Chapter 12] for a modern treatment. Forward filtering–backward sampling was introduced for Bayesian state-space inference by Carter and Kohn [1994] and Frühwirth-Schnatter [1994]. On trees, the same backward-then-forward structure appears as Pearl's belief propagation [Pearl, 1988], the sum-product algorithm on factor graphs [Kschischang et al., 2001], and Felsenstein's pruning recursion in phylogenetics [Felsenstein, 1981]. Linear-Gaussian and pairwise Markov tree variants have also been studied in multiscale estimation and signal processing [Chou et al., 1994, Desbouvries et al., 2006].

Particle smoothing, twisting, and variational objectives For nonlinear or non-Gaussian state-space models, particle smoothers provide a general simulation-based alternative. Two-filter smoothing [Briers et al., 2010] and twisted particle filters [Whiteley and Lee, 2014] make explicit the role of future information in reducing weight degeneracy. Subsequent work approximates optimal twists iteratively [Guarniero et al., 2017], casts twisting as stochastic control [Heng et al., 2020], or learns proposals and Doob-type transformations [Paige and Wood, 2016, Chopin et al., 2023]. Related variational SMC objectives include FIVO and VSMC [Maddison et al., 2017, Naesseth et al., 2018], while SIχO and NAS-χ learn smoothing-aware twists within reweighted wake-sleep objectives [Lawson et al., 2022, 2023].

Diffusion bridges and guided smoothing Diffusion smoothing is often formulated as a bridge problem. Classical guided proposals add an auxiliary drift that steers paths toward future observations and then correct the mismatch by a change of measure [Delyon and Hu, 2006, Schauer et al., 2017]; related continuous-discrete smoothing and control perspectives are developed by Mider et al. [2021] and Eklund et al. [2026]. BFFG extends this backward-filtering forward-guiding principle to directed trees and DAGs [van der Meulen and Sommer, 2025]. Recent learning-based bridge samplers use variational diffusion objectives [Archambeau et al., 2007], score matching [Heng et al., 2025, Baker et al., 2025], control-consistency losses [Howard et al., 2025], or neural guided controls [Yang et al., 2025]. NBFFG builds on BFFG and neural guided diffusion bridges, but differs by using a deliberately simple proxy guide and learning a neural correction over tree-structured discrete and continuous transitions.

7 Conclusion

We introduced Neural Backward Filtering Forward Guiding, a variational framework for smoothing on tree-structured stochastic processes. The central idea is to separate global information propagation from local posterior correction: a tractable proxy model provides closed-form backward messages and a guided proposal, while a neural correction repairs the mismatch between this guide and the true posterior. This gives a unified construction for discrete tree transitions and continuous diffusion edges, with Gaussian-mixture corrections in discrete time and controlled SDEs in continuous time.

Our experiments show that neural correction can substantially improve guided approximations on Gaussian and nonlinear diffusion benchmarks. Guided mixtures also capture multiple posterior modes in discrete models, and the butterfly application illustrates the use of this construction for high-dimensional ancestral shape reconstruction. The observed sensitivity to proxy choice and numerical resolution informs how the framework can be applied effectively. Together, these results support the central premise that a tractable model can provide useful global structure for learning a richer smoothing distribution. Combining backward information propagation with learned local corrections thus offers a flexible approach to reconstructing latent states and trajectories in tree-structured stochastic processes.

Future work includes using the path-union estimator to skip latent branches and developing Bellman-style actor-critic objectives that train local corrections with propagated value information. More inductive correction networks could support transfer to new observations and unseen topologies.

Acknowledgments and Disclosure of Funding

We thank the anonymous reviewers for their comments, which helped clarify the presentation and improve the empirical evaluation. The work presented in this paper was supported by the Villum Foundation Grant 40582, and the Novo Nordisk Foundation grants NNF24OC0093490 and NNF24OC0089608. The authors declare no conflicts of interest.

References

Joseph Felsenstein. Evolutionary trees from dna sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, 1981. doi: 10.1007/BF01734359.

John P. Huelsenbeck, Rasmus Nielsen, and Jonathan P. Bollback. Stochastic mapping of morphological characters. *Systematic Biology*, 52(2):131–158, 2003. doi: 10.1080/10635150390192780.

Anton Bovier. *Gaussian Processes on Trees: From Spin Glasses to Branching Brownian Motion*. Cambridge University Press, 2017. doi: 10.1017/9781316675779.

David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146. ACM, 2003. doi: 10.1145/956750.956769.

Mark Briers, Arnaud Doucet, and Simon Maskell. Smoothing algorithms for state-space models. *Annals of the Institute of Statistical Mathematics*, 62(1):61–89, 2010. doi: 10.1007/s10463-009-0236-2.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001. doi: 10.1109/18.910572.

Frank H. van der Meulen and Stefan Sommer. Backward filtering forward guiding. *Journal of Machine Learning Research*, 26(281):1–51, 2025. URL `https://jmlr.org/papers/v26/25-1130.html`.

Marcin Mider, Moritz Schauer, and Frank H. van der Meulen. Continuous-discrete smoothing of diffusions. *Electronic Journal of Statistics*, 15(2):4294–4342, 2021. doi: 10.1214/21-EJS1894.

H. E. Rauch, F. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965. doi: 10.2514/3.3166.

Simo Särkkä and Lennart Svensson. *Bayesian Filtering and Smoothing*. Cambridge University Press, 2 edition, 2023. doi: 10.1017/9781108917407.

Chris K. Carter and Robert Kohn. On gibbs sampling for state space models. *Biometrika*, 81(3):541–553, 1994. doi: 10.1093/biomet/81.3.541.

Sylvia Frühwirth-Schnatter. Data augmentation and dynamic linear models. *Journal of Time Series Analysis*, 15(2):183–202, 1994. doi: 10.1111/j.1467-9892.1994.tb00184.x.

K. C. Chou, A. S. Willsky, and A. Benveniste. Multiscale recursive estimation, data fusion, and regularization. *IEEE Transactions on Automatic Control*, 39(3):464–478, 1994. doi: 10.1109/9.280746.

François Desbouvries, Jean Lecomte, and Wojciech Pieczynski. Kalman filtering in pairwise markov trees. *Signal Processing*, 86(5):1049–1054, 2006. doi: 10.1016/j.sigpro.2005.07.026.

Nick Whiteley and Anthony Lee. Twisted particle filters. *The Annals of Statistics*, 42(1):115–141, 2014. doi: 10.1214/13-AOS1167.

Pieralberto Guarniero, Adam M. Johansen, and Anthony Lee. The iterated auxiliary particle filter. *Journal of the American Statistical Association*, 112(520):1636–1647, 2017. doi: 10.1080/01621459.2016.1222291.

Jeremy Heng, Adrian N. Bishop, George Deligiannidis, and Arnaud Doucet. Controlled sequential monte carlo. *The Annals of Statistics*, 48(5):2904–2929, 2020. doi: 10.1214/19-AOS1914.

Brooks Paige and Frank Wood. Inference networks for sequential monte carlo in graphical models. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 3040–3049. PMLR, 2016. URL `https://proceedings.mlr.press/v48/paige16.html`.

Nicolas Chopin, Andras Fulop, Jeremy Heng, and Alexandre H. Thiery. Computational doob h-transforms for online filtering of discretely observed diffusions. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 5904–5923. PMLR, 2023. URL https://proceedings.mlr.press/v202/chopin23a.html.

Chris J. Maddison, Dieterich Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Whye Teh. Filtering variational objectives. In *Advances in Neural Information Processing Systems 30*, 2017. URL https://papers.neurips.cc/paper/7235-filtering-variational-objectives.

Christian A. Naesseth, Scott W. Linderman, Rajesh Ranganath, and David M. Blei. Variational sequential monte carlo. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 968–977. PMLR, 2018. URL https://proceedings.mlr.press/v84/naesseth18a.html.

Dieterich Lawson, Allan Raventós, Andrew Warrington, and Scott W. Linderman. SIχO: Smoothing inference with twisted objectives. In *Advances in Neural Information Processing Systems 35*, pages 38844–38858, 2022. URL https://papers.neurips.cc/paper_files/paper/2022/hash/fddc79681b2df2734c01444f9bc2a17e-Abstract-Conference.html.

Dieterich Lawson, Michael Y. Li, and Scott W. Linderman. NAS-χ: Neural adaptive smoothing via twisting. In *Advances in Neural Information Processing Systems 36*, pages 8602–8633, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b3d005a2cb0e71e698e0b13ac657473-Abstract-Conference.html.

Bernard Delyon and Ying Hu. Simulation of conditioned diffusion and application to parameter estimation. *Stochastic Processes and their Applications*, 116(11):1660–1675, 2006. doi: 10.1016/j.spa.2006.04.004.

Moritz Schauer, Frank H. van der Meulen, and Harry van Zanten. Guided proposals for simulating multi-dimensional diffusion bridges. *Bernoulli*, 23(4A):2917–2950, 2017. doi: 10.3150/16-BEJ833.

Oskar Eklund, Annika Lang, and Moritz Schauer. Guided smoothing and control for diffusion processes. *Stochastic Processes and their Applications*, 192:104806, 2026. doi: 10.1016/j.spa.2025.104806.

Cédric Archambeau, Manfred Opper, Yuan Shen, Dan Cornford, and John Shawe-Taylor. Variational inference for diffusion processes. In *Advances in Neural Information Processing Systems 20*, 2007. URL https://papers.nips.cc/paper/3282-variational-inference-for-diffusion-processes.

Jeremy Heng, Valentin De Bortoli, Arnaud Doucet, and James Thornton. Simulating diffusion bridges with score matching. *Biometrika*, 112(4):asaf048, 2025. doi: 10.1093/biomet/asaf048.

Elizabeth L. Baker, Moritz Schauer, and Stefan Sommer. Score matching for bridges without learning time-reversals. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pages 775–783. PMLR, 2025. doi: 10.48550/arXiv.2407.15455. URL https://proceedings.mlr.press/v258/baker25a.html.

Samuel Howard, Nikolas Nüsken, and Jakiw Pidstrigach. Control consistency losses for diffusion bridges. In *Frontiers in Probabilistic Inference: Sampling Meets Learning, NeurIPS 2025 Workshop*, 2025. doi: 10.48550/arXiv.2512.05070.

Gefan Yang, Frank H. van der Meulen, and Stefan Sommer. Neural guided diffusion bridges. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 71210–71230. PMLR, 2025. URL https://openreview.net/forum?id=4LClOWTAth.

Marguerite A. Butler and Aaron A. King. Phylogenetic comparative analysis: A modeling approach for adaptive evolution. *The American Naturalist*, 164(6):683–695, 2004. doi: 10.1086/426002.

Stefan Sommer, Gefan Yang, and Elizabeth Louise Baker. Stochastics of shapes and kunita flows, 2025. URL https://arxiv.org/abs/2512.11676.

Sofia Stroustrup, Morten Akhøj Pedersen, Frank van der Meulen, Stefan Sommer, and Rasmus Nielsen. Stochastic phylogenetic models of shape. *bioRxiv*, 2025. doi: 10.1101/2025.04.03.646616. URL https://www.biorxiv.org/content/10.1101/2025.04.03.646616v1.

Michael Lind Severinsen. *Stochastic Morphometry: Applying stochastic processes to model infinite dimensional shape evolution*. Phd thesis, University of Copenhagen, 2026.

A Additional Problem Setup Details

This section gives the more explicit version of the setup underlying Equation (1).

A.1 Markov Kernels as Edge Transitions

We start with a brief recap of Markov kernels, focusing on their action on functions.

Let $(\mathcal{X}, \mathfrak{B}), (\mathcal{X}', \mathfrak{B}')$ be Borel measurable spaces. A Markov kernel $P : \mathcal{X} \times \mathfrak{B}' \to [0, 1]$ is a map such that:

- Given $B \in \mathfrak{B}'$, the map $x \mapsto P(x, B)$ is $\mathfrak{B}$-measurable for $x \in \mathcal{X}$;
- Given $x \in \mathcal{X}$, the map $B \mapsto P(x, B)$ is a probability measure on $(\mathcal{X}', \mathfrak{B}')$.

We call $(\mathcal{X}, \mathfrak{B})$ and $(\mathcal{X}', \mathfrak{B}')$ the "source" and "target" of P, and denote $P : (\mathcal{X}, \mathfrak{B}) \rightharpoonup (\mathcal{X}', \mathfrak{B}')$. For a bounded measurable function h on $(\mathcal{X}', \mathfrak{B}')$, the pullback of h through P is

$$(Ph)(x) := \int_{\mathcal{X}'} h(y) P(x, \mathrm{d}y). \tag{18}$$

For an edge in the tree, we use an edge-wise Markov kernel $P_v : (\mathcal{X}_{\mathrm{pa}(v)}, \mathfrak{B}_{\mathrm{pa}(v)}) \rightharpoonup (\mathcal{X}_v, \mathfrak{B}_v)$ to describe the conditional law of X_v given $X_{\mathrm{pa}(v)} = x$:

$$P_v(x, \mathrm{d}y) \equiv \mathbb{P}(X_v \in \mathrm{d}y \mid X_{\mathrm{pa}(v)} = x).$$

When a density p_v of P_v exists with respect to a reference measure λ_v, we write

$$P_v(x, \mathrm{d}y) = p_v(y \mid x) \lambda_v(\mathrm{d}y).$$

For a leaf $l \in \mathcal{L}$, the edge kernel is still the generative conditional law of the child given its parent. If the leaf-edge kernel admits a density p_l with respect to a reference measure ν_l, then

$$P_l(x, \mathrm{d}y) = p_l(y \mid x) \nu_l(\mathrm{d}y),$$

and the smoothing problem conditions on the realized child value $X_l = x_l$. This contributes the factor $p_l(x_l \mid x_{\mathrm{pa}(l)})$ to the posterior over the unobserved internal vertices.

A.2 Diffusion Paths as Continuous Transitions

The same Markov-kernel notation applies to continuous-time edges, but there are two useful levels of description. At the tree level, an edge still defines a vertex-to-vertex kernel $P_v(x, \mathrm{d}y)$ from the parent state to the child state. At the path level, this kernel is induced by a conditional diffusion path law.

For a continuous edge, conditional on $X_{\mathrm{pa}(v)} = x$, let $\mathbb{P}_v^x$ denote the law on the path space $\mathcal{C}_v := \mathcal{C}([0, T_v], \mathbb{R}^d)$ of the solution Z_v to the SDE

$$\mathrm{d}Z_v(t) = b_v(Z_v(t))\mathrm{d}t + \sigma_v(Z_v(t))\mathrm{d}W(t), \qquad Z_v(0) = x. \tag{19}$$

We assume b_v and σ_v satisfy standard regularity conditions ensuring existence and uniqueness of Z_v. The endpoint evaluation map $e_{T_v} : \mathcal{C}_v \to \mathbb{R}^d$ is given by $e_{T_v}(z) = z(T_v)$. With this evaluation map, we can explicitly define the vertex-level Markov kernel P_v as the pushforward of the path law $\mathbb{P}_v^x$:

$$P_v(x, A) := \mathbb{P}_v^x(e_{T_v}^{-1}(A)) = \mathbb{P}_v^x\{z \in \mathcal{C}_v : z(T_v) \in A\}, \qquad A \in \mathfrak{B}(\mathbb{R}^d). \tag{20}$$

Equivalently, with $X_v = Z_v(T_v)$,

$$P_v(x, A) = \mathbb{P}_v^x(Z_v(T_v) \in A).$$

Thus, we can fit continuous edges within the same vertex-level Markov-kernel framework as discrete edges, while retaining an underlying path-space law when pathwise quantities are needed.

The pullback of an endpoint-valued function $h : \mathbb{R}^d \to \mathbb{R}$ through the continuous edge is therefore

$$(P_v h)(x) := \mathbb{E}_{\mathbb{P}_v^x}[h(Z_v(T_v))].$$

It is also useful to pull endpoint functions back to intermediate times. For $0 \le t \le T_v$, define the time-inhomogeneous backward semigroup

$$(P_{v, t \to T_v} h)(z) := \mathbb{E}[h(Z_v(T_v)) \mid Z_v(t) = z]. \tag{21}$$

At $t = 0$, this recovers the vertex-level pullback, $(P_{v, 0 \to T_v} h)(x) = (P_v h)(x)$. This is the continuous-time analogue of the discrete pullback in Equation (18).

B Additional Method Details

B.1 Edge-Wise Neural Corrections

The theoretical BFFG construction combines the true local prior with the proxy message. If $P_s(x, \mathrm{d}y) = \mathcal{N}(\mathrm{d}y; \mu_s(x), \Sigma_s(x))$ and $\log \tilde{h}_s(y) = \kappa_s + \tilde{\eta}_s^\top y - \frac{1}{2} y^\top \tilde{H}_s y$, define

$$\begin{aligned} H_s^\circ(x) &= \Sigma_s(x)^{-1} + \tilde{H}_s, \quad & \eta_s^\circ(x) &= \Sigma_s(x)^{-1}\mu_s(x) + \tilde{\eta}_s, \\ \Sigma_s^\circ(x) &= H_s^\circ(x)^{-1}, & \mu_s^\circ(x) &= \Sigma_s^\circ(x)\eta_s^\circ(x). \end{aligned} \tag{22}$$

The discrete implementation also allows a base defined by chosen Gaussian moments $(\mu_s^{\mathrm{base}}, \Sigma_s^{\mathrm{base}})$:

$$\Sigma_s^{\mathrm{g}} = \left((\Sigma_s^{\mathrm{base}})^{-1} + \tilde{H}_s\right)^{-1}, \qquad \mu_s^{\mathrm{g}} = \Sigma_s^{\mathrm{g}}\left((\Sigma_s^{\mathrm{base}})^{-1}\mu_s^{\mathrm{base}} + \tilde{\eta}_s\right). \tag{23}$$

Choosing the true moments recovers Equation (22). Benchmarks 1A/1C and the LG proxy sweeps use the proxy's local affine mean and covariance together with its backward messages. This choice can give a different base from the BFFG transition formed with the true local kernel. The folded model keeps its nonlinear local mean, and its covariance sweep changes the local base covariance. In all cases, the $\log p_s$ term uses the true-model parameters, so changes to the base and messages leave the inference target fixed.

Let ψ_s denote the path context from Appendix B.3. The Gaussian-mixture network takes the chosen base moments as inputs:

$$\zeta_s(x) = (x, \psi_s, \mu_s^{\mathrm{g}}(x), \operatorname{vec} \Sigma_s^{\mathrm{g}}(x)). \tag{24}$$

It outputs logits $a_{s,k}^\theta(x)$, residual means $\Delta\mu_{s,k}^\theta(x)$, and lower-triangular factors $M_{s,k}^\theta(x)$ with positive diagonal. Write $L_s^{\mathrm{g}}(L_s^{\mathrm{g}})^\top = \Sigma_s^{\mathrm{g}}$, and let $o_{s,k}$ be a specified fixed component offset. The transition is

$$\begin{aligned} q_s^\theta(y \mid x) = \sum_{k=1}^{K} &\operatorname{softmax}(a_s^\theta(x))_k \\ &\times \mathcal{N}\big(y; \mu_s^{\mathrm{g}}(x) + o_{s,k} + \Delta\mu_{s,k}^\theta(x), L_s^{\mathrm{g}}(x) M_{s,k}^\theta(x) M_{s,k}^\theta(x)^\top L_s^{\mathrm{g}}(x)^\top\big). \end{aligned} \tag{25}$$

Each component has Cholesky factor $L_s^{\mathrm{g}} M_{s,k}^\theta$. The softplus parameterization initializes the diagonal of $M_{s,k}^\theta$ near one and the off-diagonal entries at zero. With the final layer initialized at zero, $a_{s,k}^\theta = 0$, $\Delta\mu_{s,k}^\theta = 0$, and $M_{s,k}^\theta \approx I$. Component means therefore start at $\mu_s^{\mathrm{g}} + o_{s,k}$, giving an initial mixture when offsets differ. Offsets are zero except at the stochastic root in Benchmark 2A, which also uses fixed uniform mixture weights (Appendix D.5). Without guidance, the base is the true local prior.

For continuous transitions, the affine proxy score is kept outside the network. Let $v_s = \tilde{H}_{s,T_s}^{-1}\tilde{\eta}_{s,T_s}$ be the proxy endpoint summary and let γ_θ denote Fourier embeddings. The residual control used in Equation (15) is

$$r_s^\theta(t, z) = R_\theta(z, \gamma_\theta(t/T_s), \gamma_\theta(T_s), v_s, \psi_s), \qquad g_s^\theta(t, z) = \nabla_z \log \tilde{h}_{s,t}(z) + r_s^\theta(t, z). \tag{26}$$

The score network first embeds $(z, \gamma_\theta(t/T_s))$ through a linear layer. Each hidden layer is then modulated by a FiLM map whose conditioning vector is $(v_s, \gamma_\theta(T_s), \gamma_\theta(t/T_s), \psi_s)$: if $c_s(t)$ denotes this conditioning vector, the layer applies $h \mapsto (1+\Gamma_\theta(c_s(t)))\odot h + B_\theta(c_s(t))$ before the nonlinearity. The FiLM projections and final output layer are initialized at zero, so the residual control starts at zero and the initial controlled SDE is exactly the proxy-guided SDE. In the no-guidance ablation, both $\nabla_z \log \tilde{h}_{s,t}(z)$ and v_s are set to zero, so the network must learn the observation-conditioning control without backward-filtering messages. We use JAX[1] for numerical computation and Flax[2] for the neural networks. The code is available at `https://github.com/gefanyang/nbffg`.

B.2 Path-Subsampled Training

Proposition B.1 (Union-of-paths subsampling)**.** *Let Γ be a distribution over nonempty subsets of $\mathcal{L}$. Draw $B \sim \Gamma$ and write $A(B) = \{s \in \mathcal{S} : \exists l \in B \text{ such that } s \preceq l\}$ and $A_0(B) = A(B) \cup \{\mathbf{0}\}$, where*

[1] https://github.com/jax-ml/jax

[2] https://github.com/google/flax

$s \preceq l$ means that s lies on the unique path from the root $\mathbf{0}$ to the leaf l. Define $\pi_s = \mathbb{P}_{B\sim\Gamma}\{s \in A(B)\}$ and $\rho_l = \mathbb{P}_{B\sim\Gamma}\{\mathrm{pa}(l) \in A_0(B)\}$, and assume $\pi_s > 0$ for all $s \in \mathcal{S}$ and $\rho_l > 0$ for all $l \in \mathcal{L}$. If only the variational transitions on $A(B)$ are simulated, then the following path loss

$$\widehat{\mathcal{J}}_B(\theta) = \sum_{s\in A(B)} \frac{C_s^\theta}{\pi_s} + \sum_{\substack{l\in\mathcal{L}\\ \mathrm{pa}(l)\in A_0(B)}} \frac{C_l}{\rho_l}$$

satisfies

$$\mathbb{E}_{B\sim\Gamma}\mathbb{E}_{\mathbb{Q}^\theta_{A(B)}}\left[\widehat{\mathcal{J}}_B(\theta)\right] = \mathcal{J}(\theta),$$

where $\mathbb{Q}^\theta_{A(B)}$ is the marginal variational law on the sampled path union.

For continuous edges, the proposition applies to the ELBO on edge paths, with $\mathbb{Q}^\theta_{A(B)}$ denoting their marginal law on the selected union. Endpoint maps determine child initial states and observation likelihoods; local variational costs depend on the simulated paths.

Proposition B.1 uses Horvitz–Thompson inclusion weights on the sampled path union. A leaf-selection estimator evaluates leaves in B and weights their costs by leaf inclusion probabilities. Equation (17) uses parent-inclusion weighting to include all observed-child costs whose parent has been sampled. Both estimators are unbiased, but there is no general variance ordering between them. Their variance depends on the sampling design.

For uniform sampling without replacement with $|B| = m$ and $N = |\mathcal{L}|$, let $\mathcal{L}_s$ be the leaves below internal vertex s. Then $\pi_s = 1 - \binom{N-|\mathcal{L}_s|}{m}/\binom{N}{m}$. The parent-inclusion probability is $\rho_l = \pi_{\mathrm{pa}(l)}$ for $\mathrm{pa}(l) \in \mathcal{S}$ and $\rho_l = 1$ for $\mathrm{pa}(l) = \mathbf{0}$. The single-path case uses $m = 1$. Shared ancestors are simulated once and reused.

Variance and sampling design. Consider two terminal parents with two observed children each, all leaf costs equal to one, and a uniformly selected set of two of the four leaves. The leaf-selection estimator is identically 4. Each parent is included with probability $5/6$, so evaluating all its children with parent-inclusion weighting gives $12/5$ with probability $1/3$ and $24/5$ with probability $2/3$. The expectation is 4 and the variance is $32/25$, compared with zero variance for leaf selection. With one selected leaf, the estimator using parent-inclusion weighting is also identically 4. Thus parent-inclusion weighting can have higher variance than leaf selection, and variance need not decrease monotonically with batch size.

B.3 Amortized Correction and Path Encoding

For each vertex s, let $u_{s,j}$ be the structural feature vector at the jth step of the padded root-to-s path, and let $a_{s,j} \in \{0, 1\}$ be its mask. In our implementation, $u_{s,j}$ contains normalized sibling rank, normalized branching factor, subtree leaf fraction, normalized depth, and optionally log edge length. The path embedding is computed by a masked GRU,

$$\bar{h}_{s,j} = \mathrm{GRU}_\theta(\ell_\theta(u_{s,j}), h_{s,j-1}), \qquad h_{s,j} = a_{s,j}\bar{h}_{s,j} + (1 - a_{s,j})h_{s,j-1}, \qquad \psi_s = h_{s,D}, \quad (27)$$

where ℓ_θ is a learned linear projection and D is the maximum padded depth. The shared correction networks use ψ_s as the structural edge context, so every sampled path updates parameters reused by unsampled branches.

C Full-path weights for evaluation

Importance sampling and Metropolis corrections require complete proposal draws and their realized density ratios. Let z collect all generated states or paths and let $L(x_\mathcal{L} \mid z)$ be their observation likelihood. A discrete edge contributes $\ell_s = \log q_s^\theta - \log p_s$. For an EM-simulated continuous edge with the same true diffusion under prior and proposal, retain its driving increments and use

$$\begin{aligned} \ell_s(z) &= \frac{1}{2}\sum_i g_{s,i}^\top a_{s,i} g_{s,i} \Delta t_s + \sum_i g_{s,i}^\top \sigma_{s,i} \Delta W_{s,i}^Q, \\ \log w(z) &= \log L(x_\mathcal{L} \mid z) - \sum_s \ell_s(z). \end{aligned} \quad (28)$$

The sums cover every simulated edge of the implemented finite-step EM target. The stochastic term has zero expectation under the proposal and contributes to each realized path weight. For a proposal reversible with respect to $\mathbb{Q}^\theta$, the Metropolis correction uses the difference of full log weights; other proposals require their corresponding Hastings factors. Computing the full-tree ratio requires these complete path contributions, including the stochastic term. Marginal diagnostics use unweighted proposal draws.

D Additional Experimental Details

D.1 Benchmark 1A: Discrete Gaussian Tree

The true hidden transition on each non-leaf edge is $X_v = A_v X_{\mathrm{pa}(v)} + b_v + \epsilon_v$, where $\epsilon_v \sim \mathcal{N}(0, Q)$, and each observed leaf is generated from $X_l \mid X_{\mathrm{pa}(l)} \sim \mathcal{N}(X_{\mathrm{pa}(l)}, r_{\mathrm{obs}}^2 I)$ with $r_{\mathrm{obs}} = 0.05$. The root is fixed at $x_0 = 0$. All runs use the same tree topology and transition parameters; each run changes only the single forward sample used to generate the observed leaves and the downstream training randomness.

The latent tree is an irregular binary backbone generated by repeated splits. Starting from the root, we split uniformly chosen active terminal latent vertices until $K = 7$ splits have been made, rejecting and resampling if the latent-backbone depth before attaching observations exceeds $D_{\max} = 5$. This gives $N_{\mathrm{lat}} = 1 + 2K = 15$ latent vertices and 8 terminal latent vertices. We then attach one observed leaf to each terminal latent vertex, giving an augmented tree with 23 total vertices. Figure 5 shows this augmented tree, so the displayed topology includes the additional observation leaves and can be one level deeper than the latent backbone controlled by $D_{\max}$.

For $d = 4$, let U_d be a fixed random orthogonal matrix and define $A_0 = U_d \,\mathrm{diag}(\lambda_1, \ldots, \lambda_d) U_d^\top$, where $\lambda_i = 0.35 + 0.50(i-1)/(d-1)$. Each hidden edge uses $A_v = \rho_v A_0$ with $\rho_v \sim \mathrm{Unif}(0.85, 1.05)$, and $b_v = 0.15 g_v/\sqrt{d}$ with $g_v \sim \mathcal{N}(0, I_d)$. The shared process covariance is $Q = U_d \,\mathrm{diag}(q_1^2, \ldots, q_d^2) U_d^\top$, where $q_i = 0.05 + 0.07(i-1)/(d-1)$.

All guided proxy models keep the true covariance and true leaf observation model. We evaluate $(\tilde{A}_v, \tilde{b}_v) \in \{(A_v, b_v), (I, 0), (-A_v, b_v), (A_v, -b_v), (-A_v, -b_v)\}$. For each proxy, "uncorrected" denotes the guided proposal built from the proxy messages, and "corrected" denotes NBFFG using the same proxy guide after neural correction. We additionally evaluate a no-guidance ablation, where the neural correction is trained from the prior without proxy messages. For this proxy comparison, the guided Gaussian base uses the selected proxy's local coefficients as well as its backward messages; the ELBO density ratio is evaluated against the true transition coefficients.

Gaussian tree smoothing provides the reference log evidence and posterior marginals. Let $\mathcal{J}^\star = -\log p(x_{\mathcal{L}})$ denote the reference negative log evidence, let $p_s^\star = \mathcal{N}(m_s^\star, C_s^\star)$ be the reference posterior marginal at internal latent vertex $s \in \mathcal{S}$, and let $q_s = \mathcal{N}(\hat{m}_s, \hat{C}_s)$ be the empirical Gaussian marginal fitted from posterior samples. We report

$$\begin{aligned} \Delta_{\mathrm{rel}} &= \frac{\mathcal{J} - \mathcal{J}^\star}{|\mathcal{J}^\star|}, & \mathrm{KL}_{\mathrm{avg}} &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathrm{KL}(q_s \| p_s^\star), \\ E_{\mathrm{mean}} &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \|\hat{m}_s - m_s^\star\|_2, & E_{\mathrm{cov}} &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{\|\hat{C}_s - C_s^\star\|_F}{\|C_s^\star\|_F}. \end{aligned} \tag{29}$$

Gains are computed as $1 - M_{\mathrm{corrected}}/M_{\mathrm{uncorrected}}$.

The corrected model uses a three-layer neural correction network with embedding dimension 8, trained for 10,000 iterations with 32 particles per iteration, peak learning rate 10^{-3}, 500 warmup steps, cosine decay to 10% of the peak learning rate, and global gradient clipping at 1.0. Evaluation uses 16 batches of 128 particles for NELBO estimation and 128 posterior samples for marginal diagnostics.

D.2 Benchmark 1B: Continuous OU Tree

The true hidden dynamics on each non-leaf edge are $dZ_v(t) = B_v(\theta_v - Z_v(t))dt + \sigma dW_t$, with $Z_v(0) = X_{\mathrm{pa}(v)}$ and $X_v = Z_v(T_v)$. Each observed leaf is generated from $X_l \mid X_{\mathrm{pa}(l)} \sim$

Table 2: Benchmark 1A: discrete heterogeneous linear Gaussian tree under proxy misspecification. Values are mean ± std. over five runs. The proxy column gives $(\tilde{A}, \tilde{b})$, with "No guidance" denoting an unguided prior-based ablation. "Uncorr." denotes the metrics before neural correction, "Corr." denotes metrics after neural correction.

Proxy	Uncorr. ↓	Corr. ↓
(a) Relative NELBO gap		
No guidance	$1.70\times10^4 \pm 7.34\times10^3$	96.659 ± 15.237
(A, b)	0.000 ± 0.000	0.002 ± 0.001
$(I, 0)$	0.913 ± 0.230	0.035 ± 0.011
$(-A, b)$	1.227 ± 0.322	0.016 ± 0.005
$(A, -b)$	3.352 ± 0.800	0.046 ± 0.015
$(-A, -b)$	4.482 ± 0.743	0.078 ± 0.040
(b) Average marginal KL		
No guidance	9.670 ± 2.548	75.495 ± 24.010
(A, b)	0.056 ± 0.004	0.060 ± 0.010
$(I, 0)$	1.861 ± 0.096	0.118 ± 0.004
$(-A, b)$	1.677 ± 0.602	0.080 ± 0.017
$(A, -b)$	6.559 ± 0.051	0.096 ± 0.022
$(-A, -b)$	6.690 ± 1.073	0.111 ± 0.020
(c) Marginal mean error		
No guidance	0.133 ± 0.041	0.635 ± 0.138
(A, b)	0.009 ± 0.001	0.009 ± 0.001
$(I, 0)$	0.092 ± 0.003	0.015 ± 0.001
$(-A, b)$	0.104 ± 0.019	0.013 ± 0.002
$(A, -b)$	0.181 ± 0.001	0.016 ± 0.002
$(-A, -b)$	0.192 ± 0.025	0.016 ± 0.002
(d) Marginal covariance error		
No guidance	5.475 ± 0.165	0.820 ± 0.077
(A, b)	0.185 ± 0.013	0.198 ± 0.011
$(I, 0)$	0.273 ± 0.010	0.234 ± 0.007
$(-A, b)$	0.180 ± 0.022	0.197 ± 0.013
$(A, -b)$	0.185 ± 0.013	0.203 ± 0.012
$(-A, -b)$	0.180 ± 0.022	0.216 ± 0.016

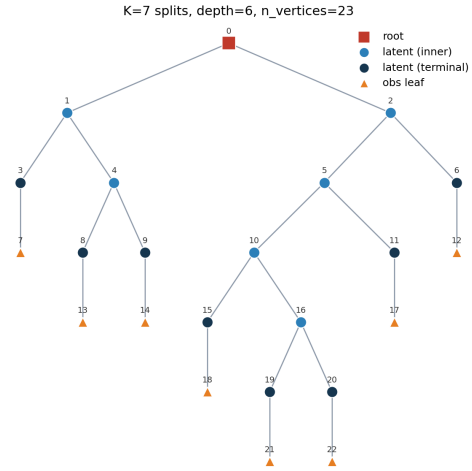
$\mathcal{N}(X_{\mathrm{pa}(l)}, r_{\mathrm{obs}}^2 I)$ with $r_{\mathrm{obs}} = 0.05$, and the root is fixed at $x_0 = 0$. As in Benchmark 1A, all runs share one topology and one prior model, and each run produces one observation instance.

The latent tree is generated by the same random-split procedure as Benchmark 1A, using $K = 7$ splits and latent-backbone depth limit $D_{\max} = 5$ before observations are attached. This gives 15 latent vertices and 8 observed leaves after augmentation. The state dimension is $d = 2$. Edge lengths are sampled as $T_v = 0.4 + 0.6u_v$, with $u_v \sim \mathrm{Unif}(0, 1)$. Figure 5 again shows the augmented tree including the observed leaves.

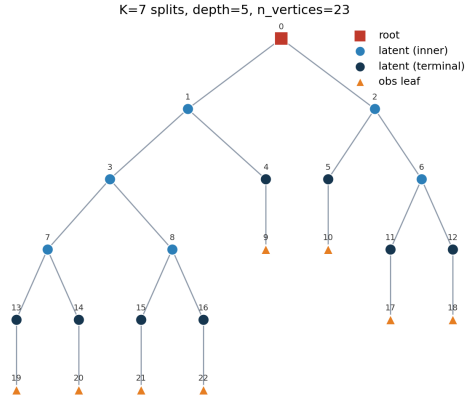
Let $U = 2^{-1/2}\left(\begin{smallmatrix} 1 & -1 \\ 1 & 1 \end{smallmatrix}\right)$ and $B_0 = U\,\mathrm{diag}(0.6, 1.2)U^\top$. Each hidden edge uses $B_v = \rho_v B_0$, with $\rho_v \sim \mathrm{Unif}(0.85, 1.15)$, and $\theta_v = 0.50 g_v/\sqrt{d}$, with $g_v \sim \mathcal{N}(0, I_d)$. The shared diffusion covariance is $a = \sigma\sigma^\top = U\,\mathrm{diag}(0.15^2, 0.30^2)U^\top$, and the same σ is used by all proxy models and by the variational SDE.

All guided proxy models keep the true diffusion coefficient and Gaussian observation likelihood. We evaluate the coefficient choice $(\tilde{B}_v, \tilde{\theta}_v) = (B_v, \theta_v)$, the Brownian proxy $d\tilde{Z}_v(t) = \sigma d\tilde{W}_t$, and $(\tilde{B}_v, \tilde{\theta}_v) \in \{(DB_vD, \theta_v), (B_v, -\theta_v), (DB_vD, -\theta_v)\}$, where $D = \mathrm{diag}(1, -1)$. We use DB_vD so that the proxy remains mean-reverting while reversing the cross-coordinate coupling. The no-guidance ablation is trained from the true prior SDE without proxy messages.

For the reference calculation, each OU edge is converted to its closed-form endpoint transition $X_v \mid X_{\mathrm{pa}(v)} \sim \mathcal{N}(M_v X_{\mathrm{pa}(v)} + c_v, Q_v^{\mathrm{OU}})$, where $M_v = \exp(-B_v T_v)$, $c_v = (I - M_v)\theta_v$, and



(a) Benchmark 1A



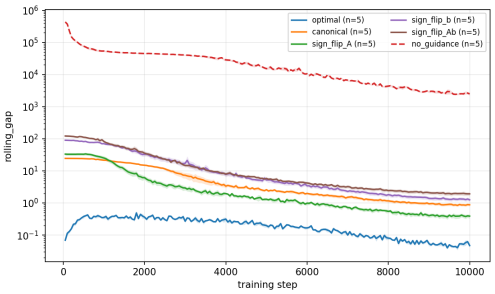
(b) Benchmark 1B

Figure 5: Tree topologies used in the Gaussian tree benchmarks. Blue vertices are latent states and orange leaves are observed children attached to terminal latent vertices.

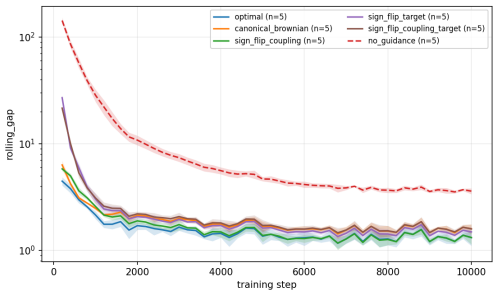
$Q_v^{\mathrm{OU}} = \int_0^{T_v} \exp(-B_v r) a \exp(-B_v^\top r) dr$. Gaussian tree smoothing on these endpoint transitions gives the reference log evidence and posterior marginals. We use the metrics in Equation (29) and the same gain definition as in Appendix D.1.

The variational SDE is simulated with 50 Euler–Maruyama steps per edge. The corrected model uses a three-layer score network with embedding dimension 8, trained for $10{,}000$ iterations with 16 particles per iteration, peak learning rate 10^{-3}, 500 warmup steps, cosine decay to 5% of the peak learning rate, and global gradient clipping at 1.0. Evaluation uses 16 batches of 128 particles for NELBO estimation and 128 endpoint samples for marginal diagnostics.

Figure 6 reports the training loss curves. The legends use implementation identifiers rather than the proxy notation used in the tables: in Benchmark 1A, `canonical` denotes $(I, 0)$, `sign_flip_A` denotes $(-A, b)$, `sign_flip_b` denotes $(A, -b)$, and `sign_flip_Ab` denotes $(-A, -b)$; in Benchmark 1B, `canonical_brownian` denotes the Brownian proxy, `sign_flip_coupling` denotes (DBD, θ), `sign_flip_target` denotes $(B, -\theta)$, and `sign_flip_coupling_target` denotes $(DBD, -\theta)$. In both panels, `optimal` labels the matching-transition proxy, and `no_guidance` denotes the prior-based ablation without proxy messages.



(a) Benchmark 1A



(b) Benchmark 1B

Figure 6: Training loss curves for the Gaussian tree benchmarks. Guided proxies converge rapidly to low NELBO values, while the no-guidance ablations remain substantially worse, illustrating the role of BFFG guidance as a useful base distribution.

Table 3: Benchmark 1B: continuous OU tree under proxy misspecification. Values are mean $\pm$ std. over five runs. The proxy column gives choices of $(\tilde{B}, \tilde{\theta})$, with "No guidance" denoting an unguided prior-based ablation. "Uncorr." denotes the metrics before neural correction, "Corr." denotes metrics after neural correction.

Proxy	Uncorr. $\downarrow$	Corr. $\downarrow$
(a) Relative NELBO gap		
No guidance	29.245 ± 17.626	0.605 ± 0.226
(B, θ)	0.892 ± 0.583	0.213 ± 0.075
$(0, 0)$	1.273 ± 0.724	0.262 ± 0.104
(DBD, θ)	1.294 ± 0.648	0.214 ± 0.073
$(B, -\theta)$	5.122 ± 1.722	0.244 ± 0.092
$(DBD, -\theta)$	3.955 ± 1.470	0.264 ± 0.111
(b) Average marginal KL		
No guidance	11.904 ± 1.922	0.144 ± 0.022
(B, θ)	0.261 ± 0.051	0.048 ± 0.011
$(0, 0)$	0.387 ± 0.121	0.062 ± 0.008
(DBD, θ)	0.431 ± 0.051	0.045 ± 0.004
$(B, -\theta)$	1.381 ± 0.208	0.056 ± 0.008
$(DBD, -\theta)$	1.105 ± 0.105	0.062 ± 0.009
(c) Marginal mean error		
No guidance	0.173 ± 0.034	0.011 ± 0.001
(B, θ)	0.032 ± 0.004	0.010 ± 0.001
$(0, 0)$	0.054 ± 0.012	0.009 ± 0.001
(DBD, θ)	0.041 ± 0.008	0.009 ± 0.001
$(B, -\theta)$	0.093 ± 0.005	0.010 ± 0.002
$(DBD, -\theta)$	0.083 ± 0.005	0.010 ± 0.002
(d) Marginal covariance error		
No guidance	7.159 ± 0.330	0.514 ± 0.058
(B, θ)	0.587 ± 0.040	0.264 ± 0.023
$(0, 0)$	0.404 ± 0.021	0.313 ± 0.033
(DBD, θ)	0.603 ± 0.041	0.257 ± 0.017
$(B, -\theta)$	0.587 ± 0.040	0.291 ± 0.028
$(DBD, -\theta)$	0.603 ± 0.041	0.317 ± 0.026

D.3 Benchmark 1C: Linear Gaussian Path Subsampling

Benchmark 1C uses a wider discrete linear-Gaussian tree to test the accuracy–speed trade-off of path-subsampled training. The hidden transition is the same family as Benchmark 1A,

$$X_v = A_v X_{\mathrm{pa}(v)} + b_v + \epsilon_v, \qquad \epsilon_v \sim \mathcal{N}(0, Q), \tag{30}$$

with $d = 4$ and a fixed root $x_0 = 0$. The canonical random-walk proxy $(\tilde{A}_v, \tilde{b}_v) = (I, 0)$ is used in all runs, with the true covariance Q. Unlike Benchmark 1A, each observed leaf uses an edge-specific orthogonal observation operator,

$$X_l \mid X_{\mathrm{pa}(l)} \sim \mathcal{N}(G_l X_{\mathrm{pa}(l)}, r_{\mathrm{obs}}^2 I), \qquad G_l \in O(d), \qquad r_{\mathrm{obs}} = 0.5. \tag{31}$$

The reference is again computed by Gaussian tree smoothing.

The latent topology is a balanced 4-ary tree with three latent layers: one root, four intermediate latent vertices, and sixteen terminal latent vertices. Four observed leaves are attached to each terminal latent vertex, giving 21 latent vertices, 64 observed leaves, and 85 total vertices in the augmented tree. Figure 7 shows this topology and highlights the multiple observation leaves attached to each terminal latent state.

We compare three training strategies. Full-tree training uses all 64 observations per iteration and serves as the time and accuracy reference. Direct leaf-path subsampling samples $m \in \{1, 2, 4, 8, 16, 32\}$ observed leaves per iteration with replacement and reweights their contributions. Terminal-path

subsampling samples $m \in \{1, 2, 4, 8\}$ terminal latent vertices per iteration without replacement and includes all four observed leaves attached to each sampled terminal, corresponding to $4m$ effective observations per step. Evaluation is always performed on the full tree against its Gaussian smoothing reference.

All models use the same five-layer correction network with hidden dimension 32, embedding dimension 16, 32 training particles per iteration, 30,000 iterations, peak learning rate 10^{-3}, 5000 warmup steps, cosine decay to 10% of the peak learning rate, and gradient clipping at 5.0. Evaluation uses 16 batches of 128 particles for NELBO estimation and 128 posterior samples for marginal diagnostics.

Figure 8 plots median KL against median speedup, with the KL interquartile range across five runs. Numerical summaries in the text use mean $\pm$ sample standard deviation. Full-tree training achieves $\mathrm{KL}_{\mathrm{avg}} = 0.066 \pm 0.008$. The settings with one or two effective observations per step are faster but show large accuracy variability. At eight effective observations per step, direct leaf-path subsampling gives mean KL 0.101 ± 0.024, while terminal-path subsampling gives 0.091 ± 0.017; their median KL values are 0.0865 and 0.0867, respectively. Their mean speedups are $1.73\times$ and $1.72\times$. Increasing the observation budget improves accuracy while reducing the speedup. At matched effective observations per step, median performance is similar, and the ordering of mean KL varies between the two schemes.

D.4 Benchmark 1D: OU Path Subsampling

Benchmark 1D repeats the path-subsampling study for continuous OU edges. The hidden dynamics are the same family as Benchmark 1B,

$$dZ_v(t) = B_v(\theta_v - Z_v(t))dt + \sigma dW_t, \qquad Z_v(0) = X_{\mathrm{pa}(v)}, \qquad X_v = Z_v(T_v), \tag{32}$$

with $d = 2$, fixed root $x_0 = 0$, and the canonical Brownian proxy $d\tilde{Z}_v(t) = \sigma d\tilde{W}_t$. Each observed leaf uses $X_l \mid X_{\mathrm{pa}(l)} \sim \mathcal{N}(X_{\mathrm{pa}(l)}, r_{\mathrm{obs}}^2 I)$ with $r_{\mathrm{obs}} = 0.1$. Reference marginals are computed from the closed-form OU endpoint transitions as in Benchmark 1B. All training strategies use the same Brownian guide.

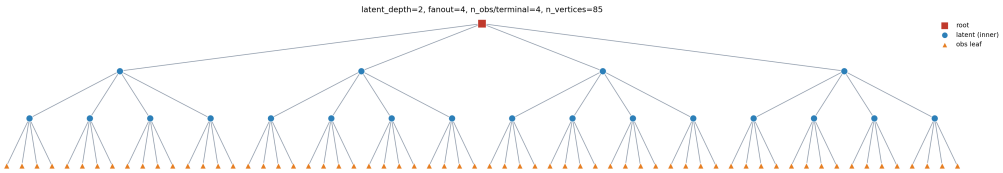
The latent tree is intentionally small and shallow: the root has four terminal latent children, and each terminal latent vertex has four observed leaves. Thus the augmented tree contains 5 latent vertices, 16 observed leaves, and 21 total vertices. Figure 7 shows this topology.

The training strategies mirror Benchmark 1C. Full-tree training uses all 16 observations per iteration. Direct leaf-path subsampling uses $m \in \{1, 2, 4, 8, 16\}$ observed leaves per iteration with replacement. Terminal-path subsampling uses $m \in \{1, 2, 4\}$ terminal latent vertices per iteration without replacement, corresponding to $4m$ effective observations. The case $m = 4$ covers all terminal groups and is therefore a sanity check against full-tree training, up to subsampler overhead. The statistical and fixed-iteration runtime summaries follow the conventions in Benchmark 1C.

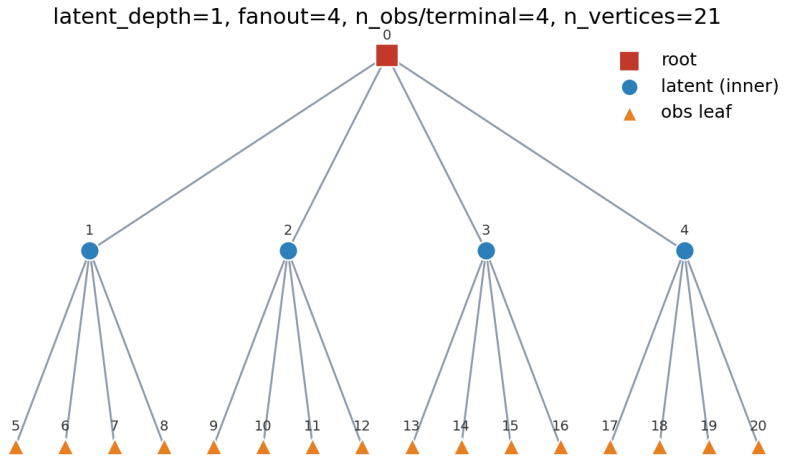
All variational SDEs use 50 Euler–Maruyama steps per hidden edge. The correction network has five layers, hidden dimension 32, node embedding dimension 16, time embedding dimension 16, 16 training particles per iteration, 10,000 iterations, peak learning rate 10^{-3}, 1000 warmup steps, cosine decay to 10% of the peak learning rate, and no gradient clipping. Evaluation uses 16 batches of 128 particles for NELBO estimation and 128 endpoint samples for marginal diagnostics.

Full-tree training gives $\mathrm{KL}_{\mathrm{avg}} = 0.061 \pm 0.016$. Direct leaf-path subsampling with one effective observation per step gives $\mathrm{KL}_{\mathrm{avg}} = 0.110 \pm 0.025$ and mean speedup $2.57\times$. The smaller latent tree and the moderately accurate uncorrected guide, with KL 0.255 ± 0.078, provide a comparatively favorable setting for aggressive path subsampling. In the continuous setting, reducing trajectory evaluations also saves Euler–Maruyama and network computations.

Terminal-path subsampling gives slightly lower mean KL at matched effective observations in this benchmark. At eight effective observations per step, direct leaf-path subsampling gives $\mathrm{KL}_{\mathrm{avg}} = 0.073 \pm 0.017$ and mean speedup $1.39\times$, while terminal-path subsampling gives 0.069 ± 0.016 and $1.38\times$. The full-coverage terminal-path case gives 0.061 ± 0.016 and $0.99\times$, matching the full-tree KL.



(a) Benchmark 1C



(b) Benchmark 1D

Figure 7: Tree topologies for the path-subsampling benchmarks. Benchmark 1C uses a wider discrete linear-Gaussian tree, while Benchmark 1D uses a smaller shallow OU tree. In both cases, multiple observed leaves are attached to each terminal latent vertex.

D.5 Benchmark 2A: Folded Nonlinear Gaussian Tree

Benchmark 2A uses a small nonlinear discrete tree designed to create a four-mode root posterior while keeping the reference calculation lightweight. The stochastic root has prior $X_\rho \sim \mathcal{N}(0, \tau_0^2 I_2)$ with $\tau_0 = 1.5$. In the implementation, this is represented by a fixed dummy parent with a Gaussian edge into the stochastic root. The root has four terminal latent children, each with transition

$$X_v = f(X_\rho) + \epsilon_v, \qquad f(x) = (x_1^2, x_2^2), \qquad \epsilon_v \sim \mathcal{N}(0, q^2 I_2), \tag{33}$$

where $q = 0.08$. Each terminal latent vertex has one observed child with $X_l \mid X_{\mathrm{pa}(l)} \sim \mathcal{N}(X_{\mathrm{pa}(l)}, r_{\mathrm{obs}}^2 I_2)$, $r_{\mathrm{obs}} = 0.05$, and all observed leaves are fixed at $x_l = (1, 1)$. The augmented implementation tree has 10 vertices: one fixed dummy parent, one stochastic root, four terminal latent vertices, and four observed leaves. Figure 9 shows the topology.

The reference root posterior is computed by analytically integrating out the terminal latent vertices. Since

$$x_l \mid X_\rho = x \sim \mathcal{N}(f(x), (q^2 + r_{\mathrm{obs}}^2) I_2), \tag{34}$$

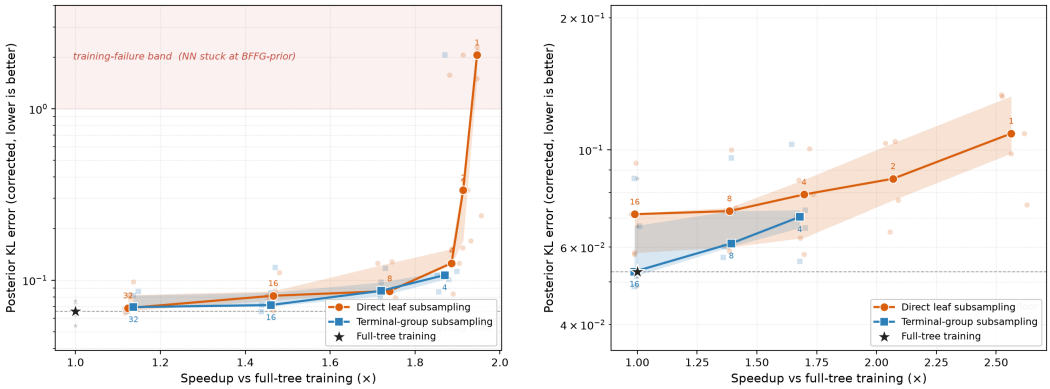
we evaluate

$$p(x \mid x_{\mathcal{L}}) \propto \mathcal{N}(x; 0, \tau_0^2 I_2) \prod_{l \in \mathcal{L}} \mathcal{N}(x_l; f(x), (q^2 + r_{\mathrm{obs}}^2) I_2) \tag{35}$$

on a 201×201 grid over $[-3, 3]^2$, normalize numerically, and use the resulting grid distribution to estimate root-quadrant probabilities and draw reference samples. The four reference quadrant probabilities are 0.25 up to numerical precision.

Let $\mathcal{Q} = \{Q_{++}, Q_{+-}, Q_{-+}, Q_{--}\}$, where each quadrant is defined by the signs of (x_1, x_2). Given root samples $\{x_\rho^{(i)}\}_{i=1}^n$ from a method, define the empirical quadrant probability

$$\hat{q}(Q) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}\{x_\rho^{(i)} \in Q\}, \qquad Q \in \mathcal{Q}. \tag{36}$$



(a) Benchmark 1C **(b) Benchmark 1D**

Figure 8: Accuracy and runtime of direct leaf-path subsampling and terminal-path subsampling. Lines connect median marginal KL and median speedup at a fixed iteration budget across five runs; bands show the KL interquartile range, faint points show individual runs, and marker annotations give effective observations per iteration. Direct leaf-path subsampling draws observed leaves with replacement; terminal-path subsampling draws terminal latent vertices without replacement and includes all their attached observed leaves. Stars denote full-tree training. Benchmark 1C has greater accuracy variability at small observation budgets, while Benchmark 1D supports more aggressive path subsampling.

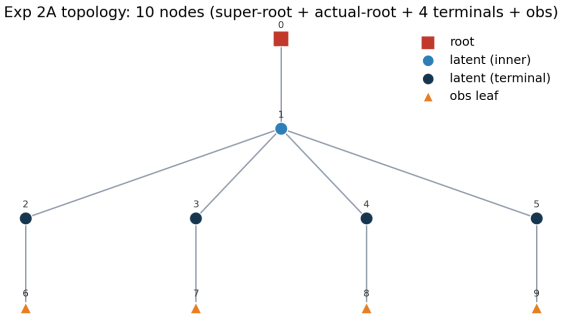


Figure 9: Tree topology for Benchmark 2A. The fixed dummy parent implements the Gaussian prior on the stochastic root.

The quadrant-probability error is

$$E_{\text{quad}} = \sum_{Q \in \mathcal{Q}} |\hat{q}(Q) - p_{\text{grid}}(Q)|, \tag{37}$$

which measures whether the method assigns the correct probability mass to the four sign modes. Because the grid reference is symmetric in this benchmark, $p_{\text{grid}}(Q) = 0.25$ for all four quadrants up to numerical precision. The maximum possible value is 1.5, attained by a method that puts all mass in one quadrant.

The recovered-mode count is

$$M_{\text{rec}} = \sum_{Q \in \mathcal{Q}} \mathbb{1}\{\hat{q}(Q) \geq 0.01\}. \tag{38}$$

It is a coarse mode-coverage diagnostic: $M_{\text{rec}} = 4$ means that all four sign modes are represented by the sample set, but it does not by itself imply the correct mode weights. This is why Table 4 reports both recovered modes and E_{quad}.

For density-level comparison, let $\hat{p}^{\text{hist}}$ and $p^{\text{hist}}_{\text{grid}}$ be the method and grid-reference root distributions binned on the same two-dimensional histogram. The reported JS divergence is

$$\text{JS}(\hat{p}^{\text{hist}}, p^{\text{hist}}_{\text{grid}}) = \frac{1}{2}\text{KL}(\hat{p}^{\text{hist}} \| m) + \frac{1}{2}\text{KL}(p^{\text{hist}}_{\text{grid}} \| m), \qquad m = \frac{1}{2}(\hat{p}^{\text{hist}} + p^{\text{hist}}_{\text{grid}}). \tag{39}$$

Table 4: Benchmark 2A: folded nonlinear Gaussian tree. Values are mean ± std. over five runs. The grid reference has four equiprobable root-quadrant modes. Lower is better for quadrant error, JS, and sliced W_2; higher is better for recovered modes.

Method	K_{mix}	Modes ↑	Quad. err. ↓	JS ↓	SW_2 ↓
Prior	–	4.0 ± 0.0	0.013 ± 0.004	0.669 ± 0.002	0.725 ± 0.007
Guide only	–	1.0 ± 0.0	1.500 ± 0.000	0.406 ± 0.001	1.441 ± 0.000
Unguided GMM	4	2.2 ± 0.4	0.903 ± 0.209	0.626 ± 0.012	0.736 ± 0.083
NBFFG Gaussian	1	1.0 ± 0.0	1.500 ± 0.000	0.406 ± 0.001	1.439 ± 0.001
NBFFG GMM	4	3.8 ± 0.4	0.118 ± 0.219	0.075 ± 0.043	0.216 ± 0.223

It measures global discrepancy of the root marginal density, not only quadrant mass. For geometric comparison, let $\hat{\mu}_r$ and $\hat{\nu}_r$ be the empirical distributions of the method and grid-reference samples projected onto unit direction $u_r \in \mathbb{S}^1$. The sliced Wasserstein distance is

$$\mathrm{SW}_2 = \left\{ \frac{1}{R} \sum_{r=1}^{R} W_2^2(\hat{\mu}_r, \hat{\nu}_r) \right\}^{1/2}. \tag{40}$$

We use $R = 32$ normalized Gaussian directions, fixed across methods and runs with random seed 0. Projected empirical quantiles are linearly interpolated onto a common midpoint grid of size equal to the larger sample set; squared differences are averaged over quantiles and directions before taking a square root. Table 4 reports this quantile-interpolation approximation to SW_2 on the saved root samples, using 8192 samples from the evaluated models and $20{,}000$ grid-reference samples per run. It is sensitive to displacement of mass within and across quadrants. Lower values of JS and SW_2 indicate closer agreement with the grid-reference root posterior.

For NBFFG GMM, four of the five independent training seeds recover all four modes. The remaining run, seed 3, recovers three modes, producing the nonzero standard deviation in the recovered-mode count and the larger standard deviation in E_{quad}, JS, and SW_2. We report this directly because it reflects residual optimization sensitivity of the mixture correction to network initialization.

The canonical guide uses the random-walk proxy $\tilde{X}_v = X_\rho + \tilde{\epsilon}_v$, $\tilde{\epsilon}_v \sim \mathcal{N}(0, q^2 I_2)$, while keeping the true observation model. We compare the prior, the uncorrected guide, an unguided Gaussian mixture, NBFFG with one Gaussian component, and NBFFG with four mixture components. Both four-component methods keep their logits at zero throughout training, giving each component a fixed weight of $1/4$. At the stochastic root, component means start at their local base means plus offsets $2(1,1)$, $2(1,-1)$, $2(-1,1)$, and $2(-1,-1)$. Since the guided and unguided bases differ, these offsets give different absolute initial component locations. Other edges have zero fixed offsets. Mode recovery is evaluated with these initially separated components. All trained models use a three-layer correction network with embedding dimension 8, 32 particles per iteration, $5{,}000$ training iterations, peak learning rate 10^{-3}, 200 warmup steps, cosine decay to 30% of the peak learning rate, and gradient clipping at 5.0. Evaluation uses 16 batches of 128 particles for NELBO estimation and $8{,}192$ root samples for posterior diagnostics.

D.6 Benchmark 2B: Double-Well Diffusion Tree

Benchmark 2B uses a one-dimensional double-well diffusion on each hidden edge,

$$dZ_v(t) = -4\alpha Z_v(t)(Z_v(t)^2 - 1)dt + \sigma dW_t, \qquad Z_v(0) = X_{\mathrm{pa}(v)}, \qquad X_v = Z_v(T_v). \tag{41}$$

The potential wells are centered near -1 and 1. Each observed leaf is attached to a terminal latent vertex with $X_l \mid X_{\mathrm{pa}(l)} \sim \mathcal{N}(X_{\mathrm{pa}(l)}, r_{\mathrm{obs}}^2)$. The root is fixed at $X_\rho = 0$. The latent tree has two root children and four terminal latent vertices; after attaching one observed leaf to each terminal latent vertex, the augmented tree has 11 vertices. The two root-child edges have length $T_{\mathrm{root}} = 4.0$, and the terminal latent edges have length $T_{\mathrm{leaf}} = 1.0$.

The numerical parameters are

$$\alpha = 3.0, \qquad \sigma = 0.5, \qquad r_{\mathrm{obs}} = 0.1. \tag{42}$$

The main comparison uses Euler–Maruyama simulation with 100 steps per edge and a numerical reference for this finite-step EM target. Appendix F.5 varies the temporal resolution. The canonical

proxy is Brownian motion with the true diffusion coefficient, $d\tilde{Z}_v(t) = \sigma d\tilde{W}_t$, and the Gaussian observation likelihood. We combine its affine score with the true double-well drift, then train an additional residual score for NBFFG. This long-horizon nonlinear model tests marginal approximation, proposal efficiency, and path diversity.

We use two deterministic observation regimes. In the early-commitment regime, the four observed leaves are fixed at

$$x_{\mathcal{L}} = (-1, -1, 1, 1), \tag{43}$$

so the two main branches should commit early to opposite wells. In the bimodal regime,

$$x_{\mathcal{L}} = (-1, -1, -1, 1), \tag{44}$$

which places opposing terminal observations within one major branch. We use this regime to examine the diversity of generated trajectories.

NBFFG uses a five-layer residual control network with hidden dimension 32, structural embedding dimension 8, 32 training particles per iteration, 10,000 iterations, peak learning rate 10^{-3}, 500 warmup steps, cosine decay to 50% of the peak learning rate, and gradient clipping at 5.0. Training uses five independent seeds.

For each seed, we restore the trained 100-step NBFFG checkpoint and draw new samples from the prior, uncorrected guide, and learned proposal, using $N = 1024$ independent paths per method and seed. We evaluate the realized density ratio of the implemented EM path law. With $g_{v,i}$ denoting the combined guide and residual score at step i of an edge and $a_{v,i} = \sigma_{v,i}\sigma_{v,i}^{\top}$, the ratio and importance weight are

$$\begin{aligned} \ell_{q/p}(z) &= \frac{1}{2}\sum_{v,i} g_{v,i}^{\top} a_{v,i} g_{v,i}\, \Delta t_v + \sum_{v,i} g_{v,i}^{\top}\sigma_{v,i}\, \Delta W_{v,i}^{Q}, \\ \log w(z) &= \log p(x_{\mathcal{L}} \mid z) - \ell_{q/p}(z). \end{aligned} \tag{45}$$

We evaluate the stochastic term with the noise used to generate each proposal path. Its expectation under the proposal is zero, so training can estimate the expected KL from the energy term alone. Individual importance and Metropolis weights require both terms, which we use in the diagnostics below.

Numerical reference for the EM target. Let v_1, v_2 denote the two children of the fixed root. Their subtrees are conditionally independent, so their evidences and posterior factors can be computed separately. Write $P_T = K_{T/100}^{100}$, where the one-step EM kernel is $K_\delta(x, \mathrm{d}y) = \mathcal{N}(\mathrm{d}y; x - 12x(x^2 - 1)\delta, 0.25\delta)$. Let $\rho(x)$ denote the density of $\delta_0 P_4$, define $\varphi_y(x) = \mathcal{N}(y; x, 0.1^2)$, and set $h_y = P_1 P_{10^{-6}}\varphi_y$. The small auxiliary observation-edge transition $P_{10^{-6}}$ is included because the runtime sampler also simulates this edge before evaluating the Gaussian likelihood. The marginal of the first root child is proportional to $\rho(x)h_{-1}(x)^2$. The second is proportional to $\rho(x)h_{+1}(x)^2$ in early commitment and $\rho(x)h_{-1}(x)h_{+1}(x)$ under competing assignments. The joint evidence is the product of the two branch normalizers.

We integrate the hidden-edge Gaussian kernels on $[-3, 3]$ with spacing $\Delta x = 0.0025$, using trapezoidal quadrature. On the shorter auxiliary edges, we use 12-point Gauss–Hermite quadrature with cubic interpolation. Halving the spacing from 0.005 to 0.0025 on $[-2.5, 2.5]$ changes evidence by less than 9×10^{-9} relatively and the interpolated marginal densities by less than 1.1×10^{-8} in total variation. Expanding the domain to $[-3, 3]$ changes evidence by less than 2×10^{-14} relatively; increasing the auxiliary quadrature order from 12 to 24 changes its likelihood values by less than 3×10^{-13} relative to their maximum. The same tree recursion reproduces the closed-form zero-drift Gaussian special case with log-evidence error below 2×10^{-10}. These empirical checks assess numerical stability. The resulting log evidences are 3.1911 and -17.9197 for the two regimes.

We evaluate all five methods against the same numerical reference. For the prior, guide, and NBFFG, each empirical distribution contains 1024 independent, unweighted paths per seed. For guided SMC, it contains the 1024 final resampled particles, including duplicates; for guided pCN, it contains the 50 retained states from one chain per seed. We give each particle or retained state equal mass without further importance weighting. The SMC particles and pCN states are dependent, and these budgets do not represent equal computational cost. Their sampler settings are specified below.

For each root child, W_1 is the integral of the absolute difference between its empirical CDF and the reference CDF. From normalized grid masses m_i, we set the reference CDF at x_i to $\sum_{k \le i} m_k - m_i/2$,

Table 5: Benchmark 2B marginal error against the 100-step EM numerical reference. We average the one-dimensional W_1 distances of the two root-child endpoints, then report mean $\pm$ sample SD over five seeds. Panel (a) uses 1024 independent, unweighted paths per seed. Panel (b) uses 1024 final resampled SMC particles or 50 retained pCN states per seed, with dependence within each run. Sampling and computational budgets differ across methods. The reference CDF is interpolated linearly between grid points. Lower values indicate smaller error.

Method	Early commitment $W_1 \downarrow$	Competing assignments $W_1 \downarrow$
(a) Unweighted independent proposals		
Prior	1.0093 ± 0.0176	0.8025 ± 0.0146
Guide only	0.7589 ± 0.0117	0.6385 ± 0.0068
NBFFG	0.0046 ± 0.0016	0.1101 ± 0.0341
(b) Particle and chain output		
Guided SMC	0.0231 ± 0.0038	0.4785 ± 0.0890
Guided pCN	0.0190 ± 0.0066	0.4508 ± 0.0950

set the domain-endpoint CDF values to 0 and 1, and interpolate linearly. We integrate the CDF discrepancy exactly on the union of sample and grid breakpoints. This avoids treating the grid as a distribution of point atoms, whose natural representation scale is $\Delta x/2 = 0.00125$. For every method, we average the two root-child distances within each run before computing the mean and sample standard deviation across five seeds. Repeating the raw-proposal comparisons with the coarser grid changes every run's two-node average by less than 10^{-5}.

NBFFG reduces root-child marginal error in both regimes (Table 5). In early commitment, its average W_1 is 0.0046 ± 0.0016, compared with 0.7589 ± 0.0117 for the guide, 0.0231 ± 0.0038 for guided SMC, and 0.0190 ± 0.0066 for guided pCN. Under competing assignments, the corresponding errors are 0.1101 ± 0.0341, 0.6385 ± 0.0068, 0.4785 ± 0.0890, and 0.4508 ± 0.0950. These results describe the stated sampler configurations and budgets; they do not establish a ranking at equal computational cost.

Figure 10 compares all five empirical marginal distributions with the reference. Under competing assignments, the numerical second-child posterior is symmetric, has local density peaks near -0.9675, 0, and 0.9675, and assigns approximately 0.674 probability to $|X_{v_2}| < 0.5$. This central peak makes the endpoint density more complex than the regime's "bimodal" name suggests. Equal sign probabilities alone would miss this structure, and NBFFG retains bias in the marginal shape. Histograms averaged across seeds can also conceal poor mixing within an individual chain; the pCN diagnostics below examine chain movement separately. The comparison measures the two endpoint marginals of the stated EM target; Appendix F.5 examines how that target changes with temporal resolution.

For normalized independent-proposal weights $\bar{w}_i$, we report $\mathrm{ESS} = 1/\sum_i \bar{w}_i^2$ and ESS/N in percent to measure weight concentration at a fixed sample budget. Posterior error is assessed separately against the marginal reference. Guided SMC uses 1024 initial paths, a linear schedule of 16 tempering/resampling stages, and no rejuvenation moves. It starts from the same Brownian-guided distribution as the uncorrected guide. For SMC, we report the number of distinct initial particle ancestors surviving final resampling. This measures particle ancestry, while independent-proposal ESS measures weight concentration.

Guided pCN uses the full path ratio and traverses every non-root tree level. We use one chain with proposal scale $\beta = 0.96$, 1000 burn-in transitions followed by 5000 transitions, and thinning by 100, giving 50 retained states per seed. Table 6 reports means and sample standard deviations over five seeds. The methods have separate diagnostics: importance ESS for independent proposals, surviving ancestry for SMC, and acceptance for pCN. Chain movement is examined below.

In early commitment, NBFFG reaches $48.71 \pm 11.18\%$ normalized importance ESS, compared with $6.12 \pm 1.33\%$ for the raw guide and $17.08 \pm 0.55\%$ for the prior. Learning improves proposal efficiency by this measure, although the uncorrected guide performs worse than the prior. Under competing assignments, NBFFG gives $2.780 \pm 2.928\%$, ranging from 0.133% to 7.536% across seeds. In the lowest-ESS run, one path carries 85.6% of the normalized weight. Weight concentration

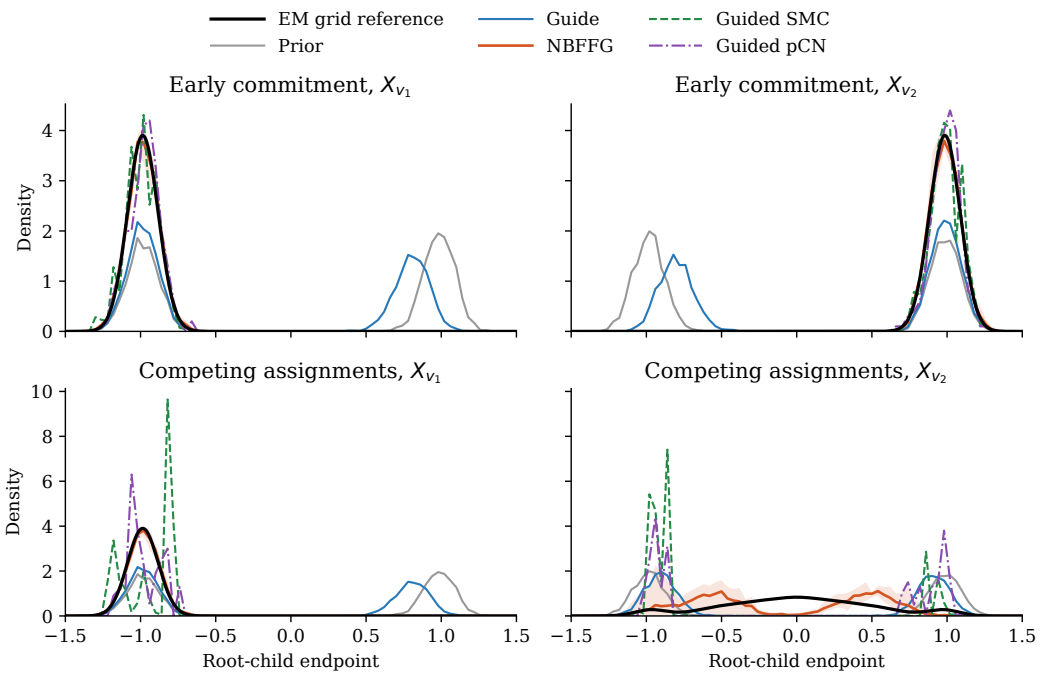


Figure 10: Root-child endpoint marginals in Benchmark 2B against the 100-step EM numerical reference (black). Colored curves are mean histograms over five seeds. Each seed contributes 1024 independent, unweighted paths for the prior, guide, and NBFFG; 1024 final resampled particles for guided SMC; or 50 retained states for guided pCN at $\beta = 0.96$. SMC particles and pCN states are dependent. NBFFG shading spans the binwise minimum and maximum across seeds. All histograms use the same bins and normalize by the full sample count.

Table 6: Benchmark 2B evaluation using realized EM path density ratios. Values are mean $\pm$ sample SD over five seeds. Panel (a) reports importance ESS as a percentage of $N = 1024$ independent proposals. Panel (b) reports surviving particle ancestry for SMC and acceptance for pCN, each in its own units. NBFFG uses the original trained checkpoints. Acceptance is descriptive and has no monotone preference; particle and chain diagnostics are not ranked against one another.

(a) Independent-proposal IS ESS/N (%) $\uparrow$		
Proposal	Early commitment	Competing assignments
Prior	17.08 ± 0.55	0.154 ± 0.063
Guide only	6.12 ± 1.33	0.200 ± 0.155
NBFFG	48.71 ± 11.18	2.78 ± 2.93
(b) Separate particle/chain diagnostics		
Diagnostic (unit)	Early commitment	Competing assignments
SMC unique initial ancestors (of 1024) $\uparrow$	71.6 ± 4.3	5.6 ± 3.8
pCN acceptance (%)	17.92 ± 1.41	0.136 ± 0.083

and seed sensitivity thus remain substantial. In this regime, SMC retains 5.6 ± 3.8 distinct initial ancestors, and pCN at the original proposal scale accepts $0.136 \pm 0.083\%$ of post-burn-in proposals.

pCN proposal-scale diagnostic. We vary the pCN proposal scale over $\beta \in \{0.10, 0.30, 0.60, 0.96\}$, using four chains per setting and one evaluation seed. Each chain has 1000 burn-in transitions followed by 5000 transitions; thinning by 10 leaves 500 states per chain. Table 7 reports acceptance averaged across the four chains. Let v_2 denote the second child of the fixed root. In the bimodal regime, $\beta = 0.10$ raises mean acceptance to 69.01%, but none of the four chains changes the sign of X_{v_2} among its retained states. At $\beta = 0.96$, there are three such sign changes across all four chains.

Table 7: pCN proposal-scale diagnostic for Benchmark 2B with one evaluation seed and four chains per setting. Acceptance percentages are averaged across the four chains. Each chain uses 1000 burn-in transitions followed by 5000 transitions, with thinning by 10. The last column counts sign changes of the second root-child endpoint X_{v_2} between retained states, summed over all four bimodal chains. More retained-state sign changes indicate more observed movement between assignments at this budget; acceptance alone has no monotone preference.

β	EC acceptance (%)	CA acceptance (%)	CA X_{v_2} switches (count) ↑
0.10	88.040	69.010	0
0.30	67.540	25.535	0
0.60	42.785	2.560	0
0.96	18.740	0.105	3

Higher acceptance therefore coexists with limited movement between the competing assignments in this single-seed experiment. Sign changes between retained states can miss excursions between thinning times.

Figure 3 shows latent trajectories from the same evaluation runs: independent, unweighted prior, guide, and NBFFG paths, final resampled SMC paths, and retained pCN paths. The early-commitment row uses seed 1 and the competing-assignment row uses seed 0. Each panel displays a fixed random subset of 30 of the 32 saved paths, without removing duplicate states or paths. The saved subset contains the first 32 output paths for the proposals and SMC, and the last 32 retained states for pCN. The terminal-edge segment is stretched horizontally by a factor of 3.5 for visibility; tick labels give physical time. These single-run plots illustrate path diversity alongside the five-seed marginal errors and sampler diagnostics.

D.7 Experiment 3: Phylogenetic Butterfly Shape Inference

Experiment 3 evaluates stochastic phylogenetic shape inference through ancestral reconstructions, objective values, and proposal-weight diagnostics at fixed checkpoints. An exact posterior reference is unavailable. Classical phylogenetic comparative models often use Brownian or Ornstein–Uhlenbeck dynamics for low-dimensional continuous traits [Felsenstein, 1981, Butler and King, 2004]. Here the state is instead a two-dimensional wing outline, represented by ordered landmarks. We use a Kunita-flow shape diffusion, following recent stochastic shape-process models [Sommer et al., 2025, Stroustrup et al., 2025], because the induced noise is spatially correlated across nearby landmarks and therefore preserves coherent deformations better than independent Euclidean noise on each coordinate.

Let $z = (z^1, \dots, z^n)$, with $z^i \in \mathbb{R}^2$, denote the landmark representation of a wing outline. Let $C(z)$ be its landmark covariance matrix and $L(z)$ its lower-triangular Cholesky factor, so $L(z)L(z)^\top = C(z)$. Along each phylogenetic edge, the implementation uses the drift-free shape SDE

$$dZ_v(t) = \big(L(Z_v(t)) \otimes I_2\big) dW_v(t), \qquad Z_v(0) = X_{\mathrm{pa}(v)}, \qquad X_v = Z_v(T_v), \tag{46}$$

where the covariance kernel is

$$C(z)_{ij} = \alpha^2 \left(1 + r_{ij} + \frac{45}{105} r_{ij}^2 + \frac{10}{105} r_{ij}^3 + \frac{1}{105} r_{ij}^4\right) e^{-r_{ij}}, \qquad r_{ij} = \sqrt{\frac{\|z^i - z^j\|_2^2}{\sigma^2} + 10^{-8}}. \tag{47}$$

The SDE has covariance $C(z) \otimes I_2$ and diffusion factor $L(z) \otimes I_2$, with $\alpha = 0.05$ and $\sigma = 0.2$. For each extant species, the observed configuration x_l has Gaussian likelihood $\mathcal{N}(x_l; X_l, 0.05^2 I)$ given the simulated endpoint X_l.

The Papilio dataset is from Severinsen [2026]. The data consist of a Papilio phylogeny with 17 observed extant species. After adding a fixed super-root, the augmented tree has 34 vertices and depth 8. The fixed super-root state is the Euclidean mean of all 33 supplied CSV configurations, including internal-node rows; only the 17 extant-species configurations enter the observation likelihood. Inference conditions on this fixed boundary, and the original root is treated as the inferred most recent common ancestor. The raw outlines have 118 landmarks; we remove one overlapping landmark, keep every second landmark, rescale coordinates by 0.002, and obtain $n = 59$ landmarks, hence state

Table 8: Papilio evaluation with 512 independent full EM paths per proposal and shared batch random keys. NELBO uncertainties are Monte Carlo standard errors for the fixed networks, each trained with one seed. Path ESS/N measures full-path weight concentration; leaf RMS error measures the fit of unweighted extant endpoints in rescaled coordinates. Both networks use 5000 logged updates plus one optimizer warmup update. NBFFG was trained with single-precision kernel arithmetic, and NN only with local double precision. Evaluation uses the same 50-step EM target and local double-precision kernel arithmetic for all proposals, with single-precision states and networks.

Method	NELBO $\pm$ MCSE $\downarrow$	Path ESS/N (%) $\uparrow$	Leaf RMS error $\downarrow$
Prior	-2069.03 ± 14.15	0.20	0.0723
Guide only	-3561.63 ± 1.02	0.30	0.0295
NN only	-3546.60 ± 1.09	0.20	0.0306
NBFFG	-3565.86 ± 1.00	2.90	0.0295

dimension $d = 118$. Branch lengths are normalized by the maximum original edge length, and the synthetic super-root edge is assigned the mean normalized edge length.

The proxy has zero drift and the same edge lengths as the phylogeny. A single bottom-up pass constructs an edge-specific covariance: after merging the child messages, we evaluate $C(z) \otimes I_2$ at their Gaussian message mode and hold it constant along the edge. The guide is therefore Gaussian, with covariance determined by the shape summarized in the child message. We train the neural residual control with this guide fixed.

The variational SDE uses 50 Euler–Maruyama steps per edge. The residual control network has 4 layers, hidden dimension 512, node embedding dimension 16, time embedding dimension 16, and SiLU activations. The training budget is 5000 logged iterations after one optimizer warmup update (5001 updates in total), with Adam at learning rate 10^{-3}, cosine decay to 10% of the peak learning rate, 8 particles per iteration, and gradient clipping at 5.0. For visualization, we draw 200 posterior endpoint samples and plot posterior means for the internal vertices. The figure uses samples from the original guided run. For the quantitative comparison below, we evaluate that checkpoint and a separately trained no-guidance model.

The highly correlated landmark covariances can be ill-conditioned, causing single-precision Cholesky factorization to fail even when states and controls are bounded. The option `kernel_dtype="float64"` evaluates the covariance and its factorizations locally in double precision, retaining single-precision states and neural parameters. The qualitative run uses single precision.

Trained no-guidance comparison. We train a no-guidance model from the prior, setting both the analytic guiding score and proxy endpoint summary to zero. It uses the same network, physical parameters, fixed root, observations, and optimization budget as NBFFG. Both neural runs use training seed.

We evaluate the prior, untrained analytic guide, trained no-guidance proposal, and NBFFG with 512 independent full-tree paths per method and 50 EM steps per edge, sharing batch random keys and particle identifiers. All methods use the same physical target and local double-precision kernel evaluation, with single-precision states and neural parameters. NELBO is the mean control energy minus the summed extant-leaf log likelihood. Importance weights use the complete realized EM path density ratio in Equation (28). Uncertainties are Monte Carlo standard errors conditional on each fixed trained network. The evaluation uses one training seed.

Table 8 gives NELBO -3565.86 ± 1.00 for NBFFG, -3546.60 ± 1.09 for the trained no-guidance proposal, and -3561.63 ± 1.02 for the untrained guide. Pairing samples by their shared random keys gives NBFFG-minus-no-guidance and NBFFG-minus-guide differences of -19.27 ± 0.32 and -4.23 ± 0.16, respectively. The analytic guide accounts for much of the objective improvement, with a smaller gain from the learned correction in this run. NBFFG's full-path ESS/N is 2.90%, compared with 0.30% for the guide. This amounts to 14.8 effective paths out of 512, so substantial weight concentration remains. Leaf RMS error measures the fit of unweighted endpoints to extant observations.

E Proofs

Full recursion for Proposition 4.1. The proxy messages in Proposition 4.1 have information forms

$$\log \tilde{h}_u(x) = -\frac{1}{2}x^\top \tilde{H}_u x + \tilde{\eta}_u^\top x + \kappa_u, \qquad \log \tilde{m}_v(x) = -\frac{1}{2}x^\top G_v x + \xi_v^\top x + \kappa_v',$$

with $\tilde{H}_u = \sum_{v\in \mathrm{ch}(u)} G_v$ and $\tilde{\eta}_u = \sum_{v\in \mathrm{ch}(u)} \xi_v$. The child-message parameters satisfy

$$(G_v, \xi_v) = \begin{cases} (L_v^\top R_v^{-1} L_v,\ L_v^\top R_v^{-1}(x_v - \beta_v)), & v \in \mathcal{L}, \\ ((\tilde{H}_v^{-1} + \tilde{\Sigma}_v)^{-1},\ (\tilde{H}_v^{-1} + \tilde{\Sigma}_v)^{-1}\tilde{H}_v^{-1}\tilde{\eta}_v), & v \in \mathcal{S}. \end{cases} \tag{48}$$

For a continuous edge, with $\tilde{\Sigma}_{v,t} = \int_t^{T_v} \tilde{\sigma}_v(\tau)\tilde{\sigma}_v(\tau)^\top \mathrm{d}\tau$,

$$\log \tilde{h}_{v,t}(z) = -\frac{1}{2}z^\top \tilde{H}_{v,t} z + \tilde{\eta}_{v,t}^\top z + \kappa_{v,t}, \qquad \tilde{H}_{v,t} = (\tilde{H}_v^{-1} + \tilde{\Sigma}_{v,t})^{-1}, \qquad \tilde{\eta}_{v,t} = \tilde{H}_{v,t}\tilde{H}_v^{-1}\tilde{\eta}_v.$$

Proof of Proposition 4.1. The endpoint law of the continuous proxy in Equation (10b) can be written as $\mathcal{N}(\mathrm{d}y; x, \tilde{\Sigma}_{T_v})$, where

$$\tilde{\Sigma}_{T_v} := \int_0^{T_v} \tilde{\sigma}_v(t)\tilde{\sigma}_v(t)^\top \,\mathrm{d}t$$

so the discrete and continuous proxies have the same endpoint-kernel form, we use a unified notation $\tilde{\Sigma}_v$ to represent the covariance of both endpoint transitions unless specified. The vertex recursion follows by induction from the leaves. For a leaf $l \in \mathcal{L}$, define the message directly as the observation likelihood

$$\tilde{m}_l(x) = p_l(x_l \mid x) = \mathcal{N}(x_l; L_l x + \beta_l, R_l),$$

which, as a function of x, has information parameters $(G_l, \xi_l) = (L_l^\top R_l^{-1} L_l, L_l^\top R_l^{-1}(x_l - \beta_l))$. If

$$\log \tilde{h}_v(y) = -\frac{1}{2}y^\top \tilde{H}_v y + \tilde{\eta}_v^\top y + \kappa_v$$

for an internal child $v \in \mathcal{S}$, then $\tilde{h}_v(y) \propto \mathcal{N}(y; \tilde{H}_v^{-1}\tilde{\eta}_v, \tilde{H}_v^{-1})$ and

$$\tilde{m}_v(x) = (\tilde{P}_v \tilde{h}_v)(x) \propto \int \mathcal{N}(y; \tilde{H}_v^{-1}\tilde{\eta}_v, \tilde{H}_v^{-1})\mathcal{N}(\mathrm{d}y; x, \tilde{\Sigma}_v) = \mathcal{N}(\tilde{H}_v^{-1}\tilde{\eta}_v; x, \tilde{H}_v^{-1} + \tilde{\Sigma}_v),$$

which has information parameters given by (G_v, ξ_v) in Equation (48). Multiplying child messages adds information parameters, giving $\tilde{H}_u = \sum_{v\in \mathrm{ch}(u)} G_v$ and $\tilde{\eta}_u = \sum_{v\in \mathrm{ch}(u)} \xi_v$.

For the within-edge guide, the proxy transition from time t to T_v is $\mathcal{N}(\mathrm{d}y; z, \tilde{\Sigma}_{v,t})$. Therefore

$$\tilde{h}_{v,t}(z) = (\tilde{P}_{v,t\to T_v}\tilde{h}_v)(z) \propto \int \mathcal{N}(y; \tilde{H}_v^{-1}\tilde{\eta}_v, \tilde{H}_v^{-1})\mathcal{N}(\mathrm{d}y; z, \tilde{\Sigma}_{v,t}) = \mathcal{N}(\tilde{H}_v^{-1}\tilde{\eta}_v; z, \tilde{H}_v^{-1} + \tilde{\Sigma}_{v,t}).$$

Its information parameters are $(\tilde{H}_{v,t}, \tilde{\eta}_{v,t})$. □

Proof of Proposition 4.2. Let

$$\mathbb{P}(\mathrm{d}x_{\mathcal{S}} \mid x_{\mathbf{0}}) = \prod_{s\in\mathcal{S}} P_s(x_{\mathrm{pa}(s)}, \mathrm{d}x_s)$$

be the prior law on the unobserved internal vertices. By Equation (4),

$$\frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}}(x_{\mathcal{S}}) = \frac{1}{Z(x_{\mathcal{L}}, x_{\mathbf{0}})}\prod_{l\in\mathcal{L}} p_l(x_l \mid x_{\mathrm{pa}(l)}).$$

By the factorization of $\mathbb{Q}^\theta$ and absolute continuity of each edge kernel,

$$\frac{\mathrm{d}\mathbb{Q}^\theta}{\mathrm{d}\mathbb{P}}(x_{\mathcal{S}}) = \prod_{s\in\mathcal{S}} \frac{\mathrm{d}Q_s^\theta(X_{\mathrm{pa}(s)}, \cdot)}{\mathrm{d}P_s(X_{\mathrm{pa}(s)}, \cdot)}(X_s).$$

Therefore

$$\begin{aligned} D_{\mathrm{KL}}(\mathbb{Q}^\theta \| \mathbb{P}^\star) &= \mathbb{E}_{\mathbb{Q}^\theta}\left[\log\left(\frac{\mathrm{d}\mathbb{Q}^\theta}{\mathrm{d}\mathbb{P}^\star}(X_{\mathcal{S}})\right)\right] \\ &= \mathbb{E}_{\mathbb{Q}^\theta}\left[\log\left(\frac{\mathrm{d}\mathbb{Q}^\theta}{\mathrm{d}\mathbb{P}}(X_{\mathcal{S}}) \cdot \left(\frac{\mathrm{d}\mathbb{P}^\star}{\mathrm{d}\mathbb{P}}\right)^{-1}(X_{\mathcal{S}})\right)\right] \\ &= \mathbb{E}_{\mathbb{Q}^\theta}\left[\sum_{s\in\mathcal{S}} \log \frac{\mathrm{d}Q_s^\theta(X_{\mathrm{pa}(s)},\cdot)}{\mathrm{d}P_s(X_{\mathrm{pa}(s)},\cdot)}(X_s) - \sum_{l\in\mathcal{L}} \log p_l(x_l \mid X_{\mathrm{pa}(l)})\right] + \log Z(x_{\mathcal{L}}, x_{\mathbf{0}}). \end{aligned}$$

The bracketed expectation is $\mathcal{J}(\theta)$, proving the claim. □

Proof of Proposition B.1. For any internal vertex $s \in \mathcal{S}$, the factorization of $\mathbb{Q}^\theta$ implies that the marginal law of the variables on the path from $\mathbf{0}$ to s is the same whether we sample the full tree or only a sampled path union containing s. Hence, whenever $s \in A(B)$,

$$\mathbb{E}_{\mathbb{Q}^\theta_{A(B)}}[C_s^\theta] = \mathbb{E}_{\mathbb{Q}^\theta}[C_s^\theta].$$

Therefore the expected internal-edge part of the path loss is

$$\begin{aligned} &\mathbb{E}_{B\sim\Gamma}\mathbb{E}_{\mathbb{Q}^\theta_{A(B)}}\left[\sum_{s\in A(B)} \frac{C_s^\theta}{\pi_s}\right] \\ &\quad= \sum_{s\in\mathcal{S}} \frac{1}{\pi_s}\mathbb{P}_{B\sim\Gamma}\{s \in A(B)\}\mathbb{E}_{\mathbb{Q}^\theta}[C_s^\theta] = \sum_{s\in\mathcal{S}} \mathbb{E}_{\mathbb{Q}^\theta}[C_s^\theta]. \end{aligned}$$

For a leaf $l \in \mathcal{L}$, the cost C_l depends only on $X_{\mathrm{pa}(l)}$ and the fixed observed value x_l. Whenever $\mathrm{pa}(l) \in A_0(B)$, the marginal law of $X_{\mathrm{pa}(l)}$ under $\mathbb{Q}^\theta_{A(B)}$ agrees with its full-tree marginal; if $\mathrm{pa}(l) = \mathbf{0}$, this state is fixed. Thus the leaf term using parent-inclusion weighting satisfies

$$\begin{aligned} &\mathbb{E}_{B\sim\Gamma}\mathbb{E}_{\mathbb{Q}^\theta_{A(B)}}\left[\sum_{\substack{l\in\mathcal{L} \\ \mathrm{pa}(l)\in A_0(B)}} \frac{C_l}{\rho_l}\right] \\ &\quad= \sum_{l\in\mathcal{L}} \frac{1}{\rho_l}\mathbb{P}_{B\sim\Gamma}\{\mathrm{pa}(l) \in A_0(B)\}\mathbb{E}_{\mathbb{Q}^\theta}[C_l] = \sum_{l\in\mathcal{L}} \mathbb{E}_{\mathbb{Q}^\theta}[C_l]. \end{aligned}$$

Adding the two identities gives the stated unbiasedness. □

F Sensitivity and computational scaling

We study sensitivity to the proxy, mixture capacity, and numerical resolution, and measure training cost. Stochastic results are summarized from per-seed records as means and sample standard deviations over completed runs, with numerical failures reported separately. The double-well temporal references are deterministic quadrature results. All comparisons use the models and optimization budgets specified below.

F.1 Proxy quality and correction accuracy

We extend Benchmarks 1A–1B from sign perturbations to random proxy parameters. On each internal edge, we construct a fixed random endpoint for parameter interpolation. In the discrete model, its affine matrix and offset match the Frobenius and Euclidean norms of the true parameters, respectively. In the OU model, an independent random rotation changes each mean-reversion matrix's eigenspace while preserving its positive eigenvalues; the target is randomized with its norm fixed. We interpolate from the true parameters toward this endpoint at strengths $s \in \{0.25, 0.50, 0.75, 1.00\}$. The endpoint is shared across the five observation/training seeds. Root and observation edges keep their original specifications.

We also vary proxy covariance. The overconfident series fixes the randomized parameters at $s = 1$ and scales covariance by $\tau \in \{0.30, 0.10, 0.03\}$. The diffuse series uses the canonical random-walk/Brownian proxy with $\tau \in \{10, 100\}$. In the discrete implementation, the proxy determines both backward messages and the initial Gaussian base. In the continuous implementation, covariance changes affect the backward filter, and the variational SDE retains the true drift and diffusion. Both objectives use the true model, so the inference targets remain fixed.

The topology, true model, observations, and training hyperparameters follow the corresponding proxy benchmarks. Here we use 2048 endpoint samples for marginal diagnostics, compared with 128 in the original benchmarks. The OU study uses a different random-number stream. These tables report separate runs under their respective proxy and evaluation settings.

Correction reduces both diagnostics for every misspecified guided setting tested in the discrete model. With the most overconfident guide, average marginal KL falls from 15.674 ± 2.237 to 0.943 ± 0.930. The remaining error is below the no-guidance value, 75.919 ± 28.428, but well above the corrected canonical-guide value, 0.050 ± 0.011. The unperturbed-coefficient row provides a reference.

In the OU model, corrected marginal KL stays between 0.014 and 0.017 along the randomized-drift interpolation. Reducing filter covariance at the fixed randomized endpoint degrades correction: at $\tau = 0.10$, the corrected relative NELBO gap is worse than no guidance in all five paired seeds. At $\tau = 0.03$, corrected marginal KL is 0.147 ± 0.026, compared with 0.126 ± 0.037 without guidance, and is worse in three of five paired seeds. The diffuse OU proxies fail earlier: all ten attempts at $\tau = 10$ and 100 produce non-finite backward messages before training. The overconfident runs complete training but lose accuracy.

F.2 Mixture capacity in the folded nonlinear model

We retrain Benchmark 2A with $K \in \{1, 2, 4, 8\}$ mixture components under the canonical guide, using five seeds and 8192 root samples per run. Training budget and network depth are fixed, as are uniform mixture weights for every K. The stochastic-root edge uses fixed component mean offsets: zero for $K = 1$, opposing offsets $(2, 2)$ and $(-2, -2)$ for $K = 2$, and all four sign combinations of $(2, 2)$ for $K = 4$. For $K = 8$, each of the four offsets is split into a pair with opposite tangential perturbations of magnitude 0.2. Other edges have zero fixed offsets. The comparison measures capacity with this initialization informed by the mode locations.

One and two components recover one and two quadrant modes, respectively, in every run. Four components recover all modes in four runs and three modes in seed 3; eight components recover all four modes in all five runs. Mode probabilities can remain inaccurate even with full quadrant coverage: for $K = 8$, seed 3 has quadrant error 0.255. The guide is fixed throughout this capacity comparison.

Table 9: Discrete linear-Gaussian proxy sensitivity. Values are mean ± sample SD over completed runs; attempts and numerical failures are shown separately. τ multiplies the proxy covariance used by the filter and Gaussian base. The $\tau < 1$ series fixes the randomized drift/mean at $s = 1$; the $\tau > 1$ series fixes the canonical proxy. The target model is unchanged. Lower is better.

Proxy	Attempts	Failed	$\mathrm{KL}_{\mathrm{avg}}$, before ↓	$\mathrm{KL}_{\mathrm{avg}}$, after ↓	Δ_{rel}, after ↓
Unperturbed coefficients	5	0	0.0032 ± 0.0002	0.0035 ± 0.0004	0.0041 ± 0.0035
Canonical	5	0	1.807 ± 0.085	0.050 ± 0.011	0.035 ± 0.010
No guidance	5	0	9.495 ± 2.461	75.919 ± 28.428	94.848 ± 35.616
Random, $s = 0.25$	5	0	0.338 ± 0.039	0.029 ± 0.004	0.033 ± 0.004
Random, $s = 0.50$	5	0	1.225 ± 0.099	0.034 ± 0.009	0.043 ± 0.010
Random, $s = 0.75$	5	0	2.414 ± 0.186	0.046 ± 0.013	0.066 ± 0.025
Random, $s = 1.00$	5	0	3.725 ± 0.362	0.075 ± 0.019	0.085 ± 0.025
Random $s = 1$, $\tau = 0.30$	5	0	6.380 ± 0.602	0.185 ± 0.062	0.179 ± 0.080
Random $s = 1$, $\tau = 0.10$	5	0	10.784 ± 1.341	0.433 ± 0.238	0.339 ± 0.187
Random $s = 1$, $\tau = 0.03$	5	0	15.674 ± 2.237	0.943 ± 0.930	0.654 ± 0.543
Canonical, $\tau = 10$	5	0	4.042 ± 0.228	0.061 ± 0.015	0.038 ± 0.012
Canonical, $\tau = 100$	5	0	42.121 ± 0.486	0.110 ± 0.027	0.064 ± 0.017

Table 10: Continuous OU proxy sensitivity. Values are mean ± sample SD over completed runs; attempts and numerical failures are shown separately. τ multiplies only the backward-filter covariance. The $\tau < 1$ series fixes the randomized drift/mean at $s = 1$; the $\tau > 1$ series fixes the canonical proxy. The target model is unchanged. The OU variational SDE retains the true diffusion. Lower is better.

Proxy	Attempts	Failed	$\mathrm{KL}_{\mathrm{avg}}$, before ↓	$\mathrm{KL}_{\mathrm{avg}}$, after ↓	Δ_{rel}, after ↓
Unperturbed coefficients	5	0	0.014 ± 0.001	0.014 ± 0.001	0.184 ± 0.085
Canonical	5	0	0.283 ± 0.084	0.019 ± 0.001	0.208 ± 0.103
No guidance	5	0	12.081 ± 1.766	0.126 ± 0.037	0.634 ± 0.331
Random, $s = 0.25$	5	0	0.031 ± 0.002	0.014 ± 0.001	0.186 ± 0.087
Random, $s = 0.50$	5	0	0.093 ± 0.006	0.015 ± 0.001	0.190 ± 0.088
Random, $s = 0.75$	5	0	0.216 ± 0.015	0.016 ± 0.001	0.198 ± 0.094
Random, $s = 1.00$	5	0	0.420 ± 0.031	0.017 ± 0.001	0.206 ± 0.099
Random $s = 1$, $\tau = 0.30$	5	0	0.912 ± 0.185	0.025 ± 0.007	0.536 ± 0.179
Random $s = 1$, $\tau = 0.10$	5	0	1.761 ± 0.418	0.077 ± 0.020	1.074 ± 0.536
Random $s = 1$, $\tau = 0.03$	5	0	2.336 ± 0.551	0.147 ± 0.026	1.571 ± 0.732
Canonical, $\tau = 10$	5	5	–	–	–
Canonical, $\tau = 100$	5	5	–	–	–

F.3 Proxy covariance in the folded nonlinear model

For the eight-component model in Benchmark 2A, we scale the four stochastic-root-to-terminal proxy transition covariances by $\tau \in \{0.3, 1, 3\}$. The multiplier τ acts on covariance. The Gaussian prior on the dummy-parent-to-stochastic-root edge, observation edges, observations, and true nonlinear transition density remain fixed. The covariance change affects both the local Gaussian proposal base and the backward messages. The untrained guide uses neither the neural correction nor its fixed component offsets.

Each covariance setting uses five seeds, a three-layer network, 32 training particles, and 5000 logged updates plus one optimizer warmup update, matching the capacity study. NBFFG keeps the eight weights at $1/8$ and initializes component means relative to their local guided base, using the root offsets split into tangential pairs described above and zero fixed offsets on other edges. Changing covariance therefore also changes absolute initial component locations. We retrain the $\tau = 1$ setting with this study's Monte Carlo stream. Evaluation uses 8192 independent, unweighted root samples per run and the same 20,000 grid-reference samples, matching the preceding capacity sweep. JS and SW_2 follow Appendix D.5.

All 30 method/seed evaluations complete without numerical failure. The guide occupies one quadrant at every covariance scale. NBFFG covers all four quadrants in five, four, and one of the five runs

Table 11: Mixture capacity in the folded nonlinear Gaussian tree. Each setting is retrained under the canonical guide with fixed uniform mixture weights and the initialization based on mode locations described in the text. Values are mean ± sample SD over completed seeds. Higher mode count and lower quadrant error and JS indicate better agreement with the reference.

K	Attempts	Failed	Recovered modes ↑	E_{quad} ↓	JS ↓
1	5	0	1.0 ± 0.0	1.500 ± 0.000	0.406 ± 0.001
2	5	0	2.0 ± 0.0	1.000 ± 0.000	0.252 ± 0.011
4	5	0	3.8 ± 0.4	0.118 ± 0.219	0.080 ± 0.044
8	5	0	4.0 ± 0.0	0.060 ± 0.109	0.071 ± 0.014

Table 12: Proxy covariance in the folded nonlinear model. We scale the four folded root-to-terminal proxy covariances by τ, keeping the true target and stochastic-root prior fixed. The untrained guide has no neural correction; NBFFG uses eight components with fixed uniform weights and the relative offsets based on mode locations described in the text. Values are mean ± sample SD over completed seeds, using 8192 unweighted root samples per run and the same $20{,}000$ grid-reference samples. Attempts and numerical failures are reported separately. Higher mode count and lower errors indicate better agreement with the reference.

τ	Attempts	Failed	Modes ↑	E_{quad} ↓	JS ↓	SW_2 ↓
Analytic guide (neural correction disabled)						
0.3	5	0	1.0 ± 0.0	1.500 ± 0.000	0.3848 ± 0.0003	1.4496 ± 0.0002
1	5	0	1.0 ± 0.0	1.500 ± 0.000	0.4054 ± 0.0006	1.4408 ± 0.0003
3	5	0	1.0 ± 0.0	1.500 ± 0.000	0.4539 ± 0.0009	1.4239 ± 0.0005
NBFFG, $K = 8$ (frozen uniform mixture weights)						
0.3	5	0	4.0 ± 0.0	0.013 ± 0.004	0.0181 ± 0.0061	0.0892 ± 0.0260
1	5	0	3.8 ± 0.4	0.110 ± 0.220	0.0863 ± 0.0499	0.1866 ± 0.2366
3	5	0	2.8 ± 0.8	0.650 ± 0.335	0.2672 ± 0.0553	0.6743 ± 0.2197

at $\tau = 0.3, 1, 3$, respectively. Average quadrant error rises from 0.013 ± 0.004 at $\tau = 0.3$ to 0.650 ± 0.335 at $\tau = 3$, and SW_2 rises from 0.089 ± 0.026 to 0.674 ± 0.220. Correction improves these diagnostics on average over the corresponding untrained guide. The diffuse proxy nevertheless leaves substantial mass and shape errors under the fixed budget, despite separated initial components and fixed uniform weights.

F.4 Backward ODE filter and SDE resolution

We vary numerical resolution on a smaller OU tree with seven latent vertices, including the fixed root, and four observed leaves. The model uses the heterogeneous OU family of Benchmark 1B and a Brownian proxy. The fixed-step backward-filter ODE solver and Euler–Maruyama variational sampler use the same number of steps.

We retrain each setting for 10000 iterations with 16 particles on the same five observation instances, then evaluate with 2048 endpoint samples. The 10- and 25-step settings fail in the backward filter before sampling or training. The 50-, 100-, and 200-step settings complete for every seed. Increasing resolution reduces marginal error and increases training time in this OU model.

F.5 Temporal resolution in the double-well model

For Benchmark 2B, we construct independent numerical references with $M \in \{50, 100, 200, 400, 800, 1600\}$ Euler–Maruyama steps per edge, keeping the physical model, observations, and auxiliary observation edges fixed. The root-to-child, terminal, and auxiliary edge durations are 4, 1, and 10^{-6}, giving time steps $4/M$, $1/M$, and $10^{-6}/M$, respectively. For each M, we check spatial-grid refinement from 0.005 to 0.0025, domain expansion from $[-2.5, 2.5]$ to $[-3, 3]$, auxiliary quadrature order, and conservative estimates of omitted evidence, as described in Appendix D.6. These checks assess numerical integration for each finite-step EM target.

Table 13: Joint backward-filter and EM resolution sensitivity on an 11-node OU tree. Values are mean $\pm$ sample SD over completed runs. The 10- and 25-step settings fail in the backward filter before training. Timings exclude JIT warmup.

Steps	Attempts	Failed	$\mathrm{KL}_{\mathrm{avg}}$, after $\downarrow$	Δ_{rel}, after $\downarrow$	Training time (s) $\downarrow$
10	5	5	–	–	–
25	5	5	–	–	–
50	5	0	0.027 ± 0.005	0.220 ± 0.130	91.188 ± 0.279
100	5	0	0.015 ± 0.002	0.132 ± 0.060	174.284 ± 0.726
200	5	0	0.0088 ± 0.0023	0.081 ± 0.092	329.047 ± 1.617

Table 14: Temporal sensitivity of the Benchmark 2B numerical target. Each row uses a separately checked finite-step EM posterior with M steps per edge and root-edge duration 4. EC and CA denote early commitment and competing assignments. The last two columns give the second root child's probability of $|X_{v_2}| < 0.5$ under CA, obtained from the continuous reference CDF, and its marginal W_1 to the 1600-step numerical reference. Results use deterministic quadrature. The 1600-step target provides the comparison reference. The zero distance of the reference row is by construction; evidence and central mass are descriptive quantities.

M	Δt_{root}	$\log Z_M$, EC	$\log Z_M$, CA	CA central mass	CA W_1 to $M = 1600$ $\downarrow$
50	0.08	No accepted full-target reference			
100	0.04	3.1911	-17.9197	0.6742	0.3532
200	0.02	3.2405	-19.1310	0.3155	0.0733
400	0.01	3.2633	-19.2660	0.2657	0.0265
800	0.005	3.2742	-19.3119	0.2468	0.0081
1600	0.0025	3.2796	-19.3308	0.2386	0.0000

References with $M \geq 100$ pass these checks. At $M = 50$, the root-edge step 0.08 is close to the local deterministic Euler stability limit $1/12$ at the wells. The estimated prior probability of exiting the numerical domain on one root edge is 1.19×10^{-5}. Conservative relative omitted-evidence estimates are 2.80×10^{-4} in early commitment and 0.073 under competing assignments, exceeding the 10^{-4} acceptance threshold. We therefore omit reference accuracy results at this resolution. A density normalized within the domain would condition on staying inside it and give a different target.

The competing-assignment posterior changes substantially with temporal resolution (Table 14). The second-child probability of $|X_{v_2}| < 0.5$ falls from 0.674 at $M = 100$ to 0.316 at $M = 200$ and 0.239 at $M = 1600$. Its marginal W_1 between consecutive references is 0.2805 from 100 to 200 steps and 0.0081 from 800 to 1600 steps. A central density peak persists through 1600 steps, while its relative mass changes markedly. The remaining temporal change leaves its continuous-time behavior unresolved. We use the finest resolution as a numerical comparison reference. The 100-step errors in Table 5 apply to that resolution's target. Since Z_M changes with resolution, raw NELBO values alone cannot measure discretization error across different M.

Training and evaluation at matched resolutions. We compare learned proposals at $M \in \{50, 100, 200\}$ in both observation regimes, with five seeds per setting. The 100-step models use archived checkpoints. We train the 50- and 200-step models from zero residual with the architecture and hyperparameters in Appendix D.6, 32 training particles, and 10,000 logged updates after one optimizer warmup update. Within each run, the prior simulator, backward filter, and variational sampler use the same M, including on auxiliary observation edges. Every completed run is evaluated with 1024 fresh independent paths. The evaluation stream differs from that in Table 5, giving different Monte Carlo estimates for the restored 100-step models.

All ten 50-step training attempts stop with non-finite computations between logged updates 176 and 672. The ten restored 100-step models and ten newly trained 200-step models complete evaluation. We report failed seeds as failures and retain the original seeds and all particles. Table 15 reports marginal error for raw proposals and self-normalized importance sampling (SNIS). Each metric averages the two root-child marginal W_1 distances, measured against both the reference at its own M and the common 1600-step numerical reference.

Table 15: Benchmark 2B training-resolution comparison with five seeds per regime and resolution. EC and CA denote early commitment and competing assignments; C/F gives completed/failed runs. The 100-step rows use archived models. The 50- and 200-step models are trained from zero residual for 10000 logged updates plus one optimizer warmup update. Each completed run is evaluated with 1024 fresh independent paths. Both panels average the two root-child marginal W_1 distances within each run, then report mean $\pm$ sample SD across five seeds. Own-M uses the matching numerical target; $M = 1600$ uses the finer EM reference. Panel (b) applies full-path SNIS weights to the same draws. ESS/N is computed before resampling. Failed runs have no accuracy metrics.

(a) Unweighted proposal draws

Regime	M	C/F	Own-M $W_1 \downarrow$	$M = 1600$ $W_1 \downarrow$	ESS/N (%) $\uparrow$
EC	50	0/5	–	–	–
EC	100	5/0	0.0045 ± 0.0026	0.0237 ± 0.0014	48.66 ± 11.30
EC	200	5/0	0.0042 ± 0.0011	0.0078 ± 0.0013	55.66 ± 20.60
CA	50	0/5	–	–	–
CA	100	5/0	0.1092 ± 0.0276	0.1558 ± 0.0095	0.58 ± 0.41
CA	200	5/0	0.1494 ± 0.0611	0.1610 ± 0.0847	3.36 ± 1.88

(b) The same draws after SNIS reweighting

Regime	M	Own-M $W_1 \downarrow$	$M = 1600$ $W_1 \downarrow$
EC	100	0.0054 ± 0.0016	0.0234 ± 0.0024
EC	200	0.0048 ± 0.0017	0.0086 ± 0.0020
CA	100	0.1499 ± 0.1202	0.2889 ± 0.0929
CA	200	0.0735 ± 0.0376	0.0978 ± 0.0385

In early commitment, raw proposals have similar error against their own targets at 100 and 200 steps: 0.0045 ± 0.0026 and 0.0042 ± 0.0011. Error against the 1600-step reference falls from 0.0237 ± 0.0014 to 0.0078 ± 0.0013. Under competing assignments, error against the own-resolution target rises from 0.1092 ± 0.0276 to 0.1494 ± 0.0611, and error against the common reference changes from 0.1558 ± 0.0095 to 0.1610 ± 0.0847. Finer simulation thus gives no consistent improvement in raw proposals under the fixed optimization budget. For SNIS under competing assignments, error against the own-resolution target falls from 0.1499 ± 0.1202 to 0.0735 ± 0.0376, while mean importance ESS is 0.58% and 3.36% of the 1024 paths, respectively. The comparison separates changes in the numerical target from variational error and the effect of importance correction at a finite sample budget.

F.6 Continuous-tree backpropagation cost

We measure training-step cost on binary OU trees with one observed child per terminal latent vertex. At 23 total vertices, we vary solver resolution from 10 to 200 steps per edge; at 50 steps, we vary the augmented tree size from 11 to 191 vertices. Each case uses five fresh processes, 16 particles, and 200 timed updates after forward and backward JIT warmup. The 23-node, 50-step measurements are reused in both comparisons. Measurements were recorded on an Apple M2 Max CPU with 32 GB host memory and JAX 0.10.0.

Training cost grows approximately with solver resolution and tree size over the tested range. The 191-node case takes 0.362 seconds per update at 50 steps. Path-union subsampling can reduce the number of simulated edges; every retained continuous edge requires its numerical trajectory and backpropagation. Benchmarks 1C–1D examine the accuracy–speed trade-off of path-subsampling. Solver resolution and tree subsampling therefore affect different parts of the cost.

Table 16: Continuous-tree training cost on the recorded Apple M2 Max CPU. Each case uses 16 particles and five fresh-process repeats, with 200 timed updates per repeat after JIT warmup. Values are mean $\pm$ sample SD across repeats. The 23-node, 50-step case is shared by both views and was measured once per repeat.

Axis	Total nodes	Steps/edge	Attempts	Failed	Time/update (ms) $\downarrow$
Solver resolution	23	10	5	0	11.79 ± 0.15
	23	25	5	0	28.26 ± 0.26
	23	50	5	0	56.79 ± 0.19
	23	100	5	0	114.77 ± 0.23
	23	200	5	0	222.25 ± 1.13
Tree size	11	50	5	0	26.93 ± 0.30
	23	50	5	0	56.79 ± 0.19
	47	50	5	0	118.01 ± 0.13
	95	50	5	0	239.56 ± 0.34
	191	50	5	0	361.69 ± 0.39

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the NBFFG construction, path-subsampled training, and empirical scope, and these claims are supported by the theory in Sections 3–4 and experiments in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The conclusion discusses current limitations, including pathwise backpropagation costs and limited topology-level amortization.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The main theoretical statements list their assumptions, and the full derivations for the Gaussian proxy guide, tree ELBO, and path-subsampling estimator are provided in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The appendix specifies the model parameters, tree topologies, training settings, evaluation metrics, and baselines needed to reproduce the reported experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper is accompanied by code and experimental scripts for the synthetic benchmarks and phylogenetic experiment, with implementation details summarized in the appendix.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The appendix reports the architectures, optimizers, learning rates, iteration counts, particle counts, discretization settings, and evaluation procedures for each experiment.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Tables report mean and standard deviation over independent runs for the quantitative experiments, and the text states the number of seeds used.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports inference times for some experiments together with the computational platform.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is methodological, uses synthetic data and non-sensitive phylogenetic shape data, and does not involve prohibited data collection or deployment settings.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper focuses on foundational inference methodology and does not include a separate broader-impact discussion because it is not tied to a deployed application.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The work does not release high-risk models or datasets such as pretrained language models, image generators, or scraped web data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Existing software and scientific data sources used in the experiments are cited or identified in the paper and appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The newly introduced code, configurations, and generated benchmark outputs are documented through the experimental descriptions and accompanying release materials.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing, user studies, or experiments with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects research, so IRB approval or an equivalent review is not applicable.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as part of the proposed method, experiments, or scientific contribution.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.