A Ginkgo Generative Model

The `Ginkgo` generative process is outlined below: The 2-body decay in the parent rest frame is defined using

Algorithm 1 Toy Parton Shower Generator

```
Require: parent momentum $p_p^\mu$, parent mass squared $t_P$, cut-off mass squared $t_{\text{cut}}$, rate for the exponential
    distribution $\lambda$, binary tree tree
 1: Function NodeProcessing($p_p^\mu$, $t_P$, $t_{\text{cut}}$, $\lambda$, tree)
 2: Add parent node to tree.
 3: if $t_P > t_{\text{cut}}$ then
 4:     Sample $t_L$ and $t_R$ from the decaying exponential distribution.
 5:     Sample a unit vector from a uniform distribution over the 2-sphere.
 6:     Compute the 2-body decay of the parent node in the parent rest frame.
 7:     Apply a Lorentz boost to the lab frame to each child.
 8:     call NodeProcessing($p_p^\mu$, $t_L$, $t_{\text{cut}}$, $\lambda$, tree)
 9:     call NodeProcessing($p_p^\mu$, $t_R$, $t_{\text{cut}}$, $\lambda$, tree)
10: end if
```

momentum $p_p^\mu = p_L^\mu + p_R^\mu = (\sqrt{s}, 0, 0, 0)$. Due to energy-momentum conservation the child energies are given by

$$E_L = \frac{\sqrt{s}}{2}\left(1 + \frac{t_L}{s} - \frac{t_R}{s}\right) \tag{18}$$

$$E_R = \frac{\sqrt{s}}{2}\left(1 + \frac{t_R}{s} - \frac{t_L}{s}\right) \tag{19}$$

and the magnitude of their 3-momentum is defined

$$|\vec{p}| = \frac{\sqrt{s}}{2}\bar{\beta} = \frac{\sqrt{s}}{2}\sqrt{1 - \frac{2(t_L + t_R)}{s} + \frac{(t_L - t_R)^2}{s^2}} \tag{20}$$

The left and right child momentum are given by $p_L^\mu = (E_L, \vec{p})$ and $p_R^\mu = (E_R, -\vec{p})$ in the parent rest frame. The Lorentz boost $\gamma = \frac{E_p}{\sqrt{t_p}}$ and $\gamma\beta = |\vec{p}_p|/\sqrt{t_p}$. For more information see Cranmer, Kyle et al. (2021); Cranmer et al. (2019b).

B Combinatorial Sequential Monte Carlo

We provide an overview of the Csmc algorithm from Wang et al. (2015).

B.1 Partial States and the Natural Forest Extension

Definition 1 (Partial State)**.** A rank $r \in \{0, \cdots, N-1\}$ partial state, symbolized as $s_r = (t_i, X_i)$, represents a collection of rooted trees and adheres to the following three conditions:

1. Partial states of different ranks are disjoint, meaning that for any two distinct ranks, r and s, there is no overlap between the sets of partial states, written as $\forall r \neq s$, $\mathcal{S}_r \cap \mathcal{S}_s = \emptyset$.
2. The set of partial states at the smallest rank consists of only one element, denoted as $S_0 = \perp$.
3. The set of partial states at the final rank $R = N-1$ corresponds to the target space $\mathcal{X}$.

The likelihood, as represented in Eq. 12, and the probability measure π are specifically defined within the scope of the target space, denoted as $\mathcal{S}_R = \mathcal{X}$. It's important to note that these definitions apply exclusively to the target space of trees and not to the broader sample space encompassing partial states, denoted as $\mathcal{S}_{r<R}$, which consists of forests containing disjoint trees. The Sum-Product algorithm is primarily utilized to derive a maximum likelihood estimate for a tree. However, partial states are explicitly characterized as collections of these disjoint trees or leaf nodes. To extend the target measure π to encompass the sample space $\mathcal{S}_{r<R}$, a practical approach is to treat all elements of the jump chain as trees, as elaborated in Wang et al. (2015).

Definition 2 (Natural Forest Extension)**.** The natural forest extension, denoted as Nfe, expands the target measure π into forests by forming a product over the trees contained within the forest:

$$\pi(s) \coloneqq \prod_{(t_i, X_i)} \pi_{Y_i(x_i)}(t_i) \,. \tag{21}$$

One notable advantage of the Nfe is its ability to transmit information from non-coalescing elements to the local weight update.

Algorithm 2 Combinatorial Sequential Monte Carlo

```
  Input: $\mathbf{Y} = \{Y_1, \cdots, Y_M\} \in \Omega^{NxM}$, $\theta$
1: Initialization. $\forall k$, $s_0^k \leftarrow \perp$, $w_0^k \leftarrow 1/K$.
2: for $r = 0$ to $R = N-1$ do
3:     for $k = 1$ to $K$ do
4:         Resample
               $\mathbb{P}(a_{r-1}^k = i) = \frac{w_{r-1}^i}{\sum_{l=1}^K w_{r-1}^l}$
5:         Extend partial state
               $s_r^k \sim q(\cdot | s_{r-1}^{a_{r-1}^k})$
6:         Compute weights
               $w_r^k = w(s_{r-1}^{a_{r-1}^k}, s_r^k) = \frac{\pi(s_r^k)}{\pi(s_{r-1}^{a_{r-1}^k})} \cdot \frac{\nu^-(s_{r-1}^{a_{r-1}^k})}{q(s_r^k | s_{r-1}^{a_{r-1}^k})}$
7:     end for
8: end for
9: Output: $s_R^{1:K}$ , $w_{1:R}^{1:K}$
```

C Nested Combinatorial Sequential Monte Carlo

We provide a review of the Ncsmc algorithm from Moretti et al. (2021). The Ncsmc method performs a standard Resample step (*line 4*), similar to Csmc methods, iterating over rank events. In each iteration, Ncsmc explores all possible one-step ahead topologies ($\binom{N-r}{2}$) and samples *sub-branch* lengths for each of them (*line 5-7*). Importance *sub-weights* or *potential functions* are evaluated for these sampled look-ahead states (*line 8*). The ancestral partial state is then extended to a new partial state by choosing a topology and branch length based on their respective weights (*line 11*). The final weight for each sample is calculated by averaging over the potential functions (*line 12*). For a visual representation of this procedure, please refer to Fig. 7.

Algorithm 3 Nested Combinatorial Sequential Monte Carlo

```
  Input: $\mathbf{Y} = \{Y_1, \cdots, Y_N\} \in \Omega^{NxM}$, $\theta$
 1: Initialization. $\forall k$, $s_0^k \leftarrow \perp$, $w_0^k \leftarrow 1/K$.
 2: for $r = 1$ to $R = N - 1$ do
 3:   for $k = 1$ to $K$ do
 4:     Resample  $\mathbb{P}(a_{r-1}^k = i) = \frac{w_{r-1}^i}{\sum_{l=1}^K w_{r-1}^l}$
 5:     for $i = 1$ to $L = \binom{N-r}{2}$ do
 6:       for $m = 1$ to $M$ do
 7:         Form look-ahead partial state
              $$s_r^{k,m}[i] \sim q(\cdot|s_{r-1}^{a_{r-1}^k})$$
 8:         Compute potentials
              $$w_r^{k,m}[i] = \frac{\pi(s_r^{k,m}[i])}{\pi(s_{r-1}^{a_{r-1}^k})} \cdot \frac{\nu^-(s_{r-1}^{a_{r-1}^k})}{q(s_r^{k,m}[i]|s_{r-1}^{a_{r-1}^k})}$$
 9:       end for
10:     end for
11:     Extend partial state
              $$s_r^k = s_r^{k,J}[I],$$
              $$\mathbb{P}(I = i, J = j) = \frac{w_r^{k,j}[i]}{\sum_{l=1}^L \sum_{m=1}^M w_r^{k,m}[i]}$$
12:     Compute weights
              $$w_r^k = \frac{1}{ML} \sum_{i=1}^L \sum_{m=1}^M w_r^{k,m}[i]$$
13:   end for
14: end for
  Output: $s_R^{1:K}$ , $w_{1:R}^{1:K}$
```

D Approximate Posteriors

The proposal distribution for our point estimator adaptation of Csmc and the corresponding approximate posterior for Vcsmc corresponding to Eq. 5 can be written explicitly as follows:

$$Q_\phi\left(\mathcal{T}_{1:R}^{1:K}, a_{1:R-1}^{1:K}\right) := \left(\prod_{k=1}^K q_\phi(\mathcal{T}_1^k)\right) \cdot \prod_{r=2}^R \prod_{k=1}^K \left[\frac{w_{r-1}^{a_{r-1}^k}}{\sum_{l=1}^K w_{r-1}^l} \cdot q_\phi\left(\mathcal{T}_r^k|\mathcal{T}_{r-1}^{a_{r-1}^k}\right)\right]. \tag{22}$$

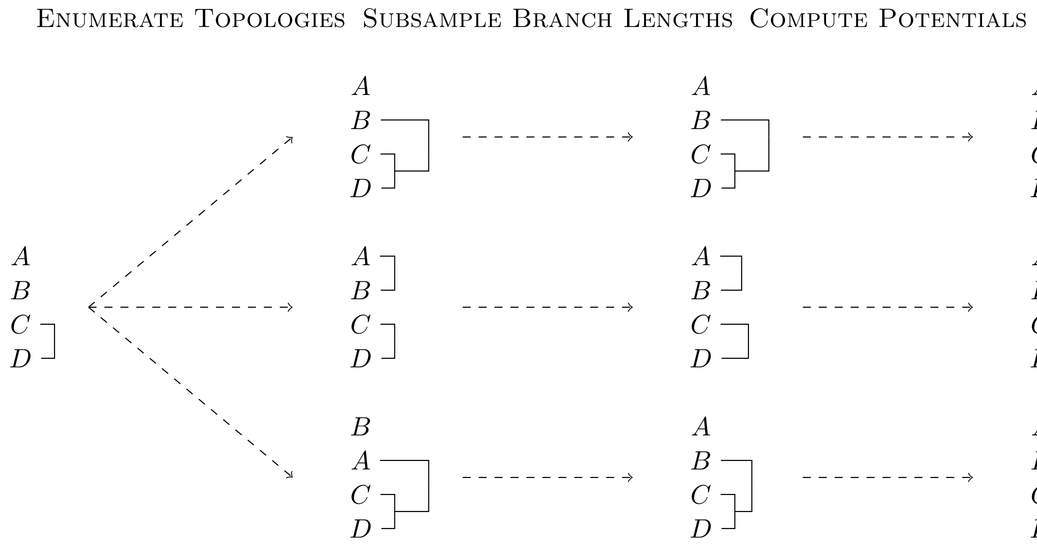

Figure 7: Illustration of the Ncsmc framework: In Ncsmc, all possible one-step ahead topologies, which amount to $\binom{N-r}{2}$ in total, are systematically enumerated. For the state A, B, C, D, the enumerated topologies include (*top*): A, B, C, D, (*center*): A, B, C, D, and (*bottom*): B, A, C, D. Subsequently, $M = 1$ *sub-branch* lengths are stochastically sampled for each edge. Following this, the *sub-weights* or *potentials* are computed (right), and a single candidate is randomly selected proportional to its sub-weight (or potential) to create the new partial state.

In the above, a_{r-1}^k denotes the ancestor index of the resampled random variable and the partial state $s_r^k = \mathcal{T}_r^k$ is sampled by proposing forest $\mathcal{T}_r^k \sim q_\phi(\cdot|\mathcal{T}_{r-1}^{a_{r-1}^k})$ from a Uniform distribution. Similarly, the proposal distribution for the global posterior defined in Eq. 14 is expressed where $\lambda_r^k \sim q_\psi(\cdot|\lambda_{r-1}^{a_{r-1}^k}) = \log N(\cdot|\tilde{\mu}, \tilde{\Sigma})$:

$$Q_{\phi,\psi}\left(\mathcal{T}_{1:R}^{1:K}, \Lambda_{1:R}^{1:K}, a_{1:R-1}^{1:K}\right) \coloneqq \left(\prod_{k=1}^{K} q_\phi(\mathcal{T}_1^k) \cdot q_\psi(\lambda_1^k)\right) \cdot \prod_{r=2}^{R} \prod_{k=1}^{K} \left[\frac{w_{r-1}^{a_{r-1}^k}}{\sum_{l=1}^{K} w_{r-1}^l} \cdot q_\phi\left(\mathcal{T}_r^k | \mathcal{T}_{r-1}^{a_{r-1}^k}\right) \cdot q_\psi\left(\lambda_r^k | \lambda_{r-1}^{a_{r-1}^k}\right)\right]. \tag{23}$$

The Ncsmc method detailed in Algorithm 3 can also be used to form an unbiased and consistent estimator of the log-marginal likelihood $\widehat{\mathcal{Z}}_{NCSMC}$ and a variational objective which we refer to as $\mathcal{L}_{NCSMC}$:

$$\mathcal{L}_{NCSMC} \coloneqq \mathop{\mathbb{E}}_{Q}\left[\log \hat{\mathcal{Z}}_{NCSMC}\right], \qquad \widehat{\mathcal{Z}}_{NCSMC} \coloneqq \prod_{r=1}^{R}\left(\frac{1}{K}\sum_{k=1}^{K} w_r^k\right). \tag{24}$$

E Log Conditional Likelihood $\hat{P}(X|\tau, \lambda)$

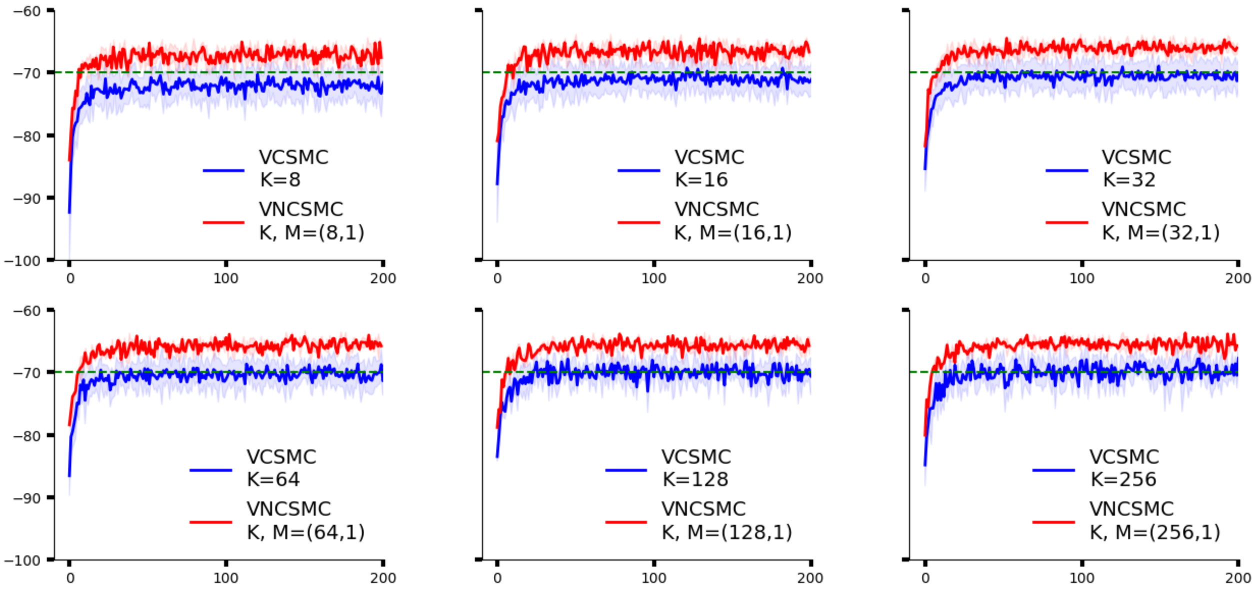

Figure 8: Log-conditional likelihood $\hat{P}(X|\tau, \lambda)$ values for VCSMC (blue) with $K = \{8, 16, 32, 64, 128, 256\}$ samples and VNCSMC (red) with $K = \{8, 16, 32, 64, 128, 256\}$ and $M = 1$ samples averaged across 5 random seeds. Greater values of K result in a more constrained ELBO and higher log-likelihood values while reducing stochastic gradient noise. VNCSMC with $K \geq 8$ explores higher probability spaces than the likelihood returned by the simulator, as depicted by the green trace for reference. VNCSMC achieves convergence in fewer epochs than VCSMC and yields higher values, all while maintaining lower stochastic gradient noise. Notably, even VNCSMC with $(K, M) = (8, 1)$ in the top-left plot (red) outperforms VCSMC with K = 256 in the bottom-right plot (blue).

F Implementation Details

F.1 Invalid Partial States when Coalescing Particles

Physics imposes several constraints on which pairs of particles are impossible to coalesce. We must consider these constraints as we are building trees from leaf to root, coalescing particles (represented as nodes) at every iteration. The following are the conditions

1. $t > 0$ for any node.
2. $t_p > t_{cut}$ for all inner nodes.
3. $t_p > max(t_l, t_r)$ for all inner nodes.

Recall that the ELBO is a function of the weight matrix, which is of dimensions (R, K), and contains all weights of the K particles across R iterations. Each entry in the matrix represents the corresponding weight of some partial state.

In VCSMC and VNCSMC, resampling ensures that we not only extend upon partial states of valid non-zero probability, but we also arrive at K valid trees at the final rank event. We note that both Greedy Search and Beam Search often fail to find any valid trees because they reach a set of partial states where no viable tree can be constructed.